Б.С. Гольдштейн, А.А. Зарубин, В.В. Саморезов

СПРАВОЧНИК
ПО ТЕЛЕКОММУНИКАЦИОННЫМ ПРОТОКОЛАМ

Протокол SIP

Б.С. Гольдштейн, А.А. Зарубин, В.В. Саморезов

Серия справочников
«Телекоммуникационные протоколы ВСС РФ»

Протокол SIP

Справочник

Санкт-Петербург
«БХВ-Петербург»
2014

УДК 621.395
Г63
ББК 32.88

Б.С. Гольдштейн, А.А. Зарубин, В.В. Саморезов

Протокол SIP: Справочник. – СПб.: БХВ-Петербург, 2014. – 456 с.: ил.

ISBN 978-5-9775-1593-1

Приводятся сведения о принципах организации и функционирования протокола SIP (Session Initiation Protocol), широко используемого сегодня в IP-телефонии и являющегося наиболее вероятным кандидатом на ведущую роль в сетях связи следующего поколения NGN. Описываются сообщения SIP, процедуры управления соединениями в IP-сети и между сетями IP и ТфОП, процедуры аутентификации, защиты информации, обеспечения безопасности. Рассматриваются расширения SIP, обеспечивающие взаимодействие сети IP с телефонной сетью при создании и поддержке сеансов связи ТфОП-IP-ТфОП, ТфОП-IP и IP-ТфОП. Излагаются задачи преобразования сигнализации SIP при взаимодействии с другими протоколами сетей NGN. Освещаются вопросы тестирования SIP и пути реализации на базе этого протокола ряда известных и новых инфокоммуникационных услуг.

Справочник

ISBN 978-5-9775-1593-1

© Гольдштейн Б.С., Зарубин А.А., Саморезов В.В., 2005, 2014

Издательство «БХВ-Петербург», 190005, Санкт-Петербург, Измайловский пр., 29

Содержание

Предисловие

«Музыкой будущего» иронически именовали опубликованную Рихардом Вагнером в 1850 теорию музыки; это выражение и сегодня используется среди музыкантов как насмешка. Столь же критически был встречен первый опубликованный Инженерной проблемной группой Интернет (IETF) в феврале 1996 года документ draft-ietf-mmusic-sip-00, с которого начинались спецификации SIP, тоже весьма медленно внедрявшиеся в телекоммуникационную индустрию. Слово MMUSIC в названии документа не связывалось непосредственно с музыкой, а представляло собой аббревиатуру выражения *Multiparty Multimedia Session Control*. Сам же документ специфицировал лишь единственный запрос установления сеанса связи, но уже тогда ориентировался на интеграцию в создававшуюся в то время мультимедийную архитектуру Интернет-конференций. В состав протоколов ядра этой мультимедийной архитектуры входили Session Announcement Protocol (SAP), Session Description Protocol (SDP) [37], Real-Time Streaming Protocol (RTSP) [64] и Real-Time Transport Protocol (RTP) [61], рассмотрение которых выходит за рамки данной книги. Следом за документом *draft-ietf-mmusic-sip-00* в декабре 1996 г. появилась следующая версия — *draft-ietf-mmusic-sip-01,* — но и в ней еще не угадывалось в полной мере будущее протокола инициирования сеансов связи SIP (Session Initiation Protocol).

Дело в том, что к началу 1996 года в IETF соперничали два протокола установления сеансов связи: *Session Invitation Protocol*, который предложили Марк Хэндли (Handley) и Ева Шулер (Schooler), и протокол *Simple Conference Invitation Protocol (SCIP)*, который предложил Хеннинг Шульцрин (Schulzrinne). Протокол Session Invitation Protocol поддерживал только установление сеанса и элементарные возможности согласования. После того как сеанс начался, этот протокол не использовался вовсе. Протокол был рассчитан на работу только поверх протокола User Datagram Protocol (UDP) и использовал SDP для описания параметров сеанса.

Протокол SCIP базировался на Transmission Control Protocol (TCP) и обеспечивал также прекращение сеанса [69]. SCIP использовал такие существующие Internet-протоколы, как Hypertext Transport Protocol (HTTP) и Simple Mail Transport Protocol (SMTP), но не использовал SDP для описания своих сеансов. В отличие от Session Invitation Protocol, протокол SCIP был разработан с оглядкой на телефонные функции. В конце 1996 года оба эти протокола были объединены в протокол Session Initiation Protocol, который позаимствовал идеи у каждого из своих прародителей. У Session Invitation Protocol протокол SIP перенял базирование на UDP и использование SDP. У SCIP протокол SIP взял поддержку TCP и его близость к другим известным протоколам IETF (таким как SMTP и HTTP). Новый протокол назвали «SIP/2.0», чтобы отличить его от «SIP/1.0», Session *Invitation* Protocol.

За три следующих года было разработано 11 версий документа *draft-ietf-mmusic-sip*, что, в конечном итоге, привело к публикации в январе 1999 г. *draft-ietf-mmusic-sip-12*, предусматривавшего 6 запросов, которые и сегодня имеются в SIP, и составившего основу опубликованного IETF (Internet Engineering Task Force) в марте 1999 г. документа RFC 2543, уже существенно отличавшегося от первого варианта, который был составлен Розенбергом (Rosenberg) и его научным руководителем в аспирантуре Колумбийского университета Хеннингом Шульцрином.

С той поры RFC 2543 продолжал претерпевать многочисленные изменения и усовершенствования; работа по дальнейшему описанию и улучшению протокола SIP продолжается и сейчас. В настоящее время действующим стандартом SIP является RFC 3261, в котором отражены все эти изменения. К чести IETF, следует отметить, что стандарт RFC 3261 совместим с предыдущей версией, т.е. с RFC 2543, и, таким образом, предшествующие реализации SIP будут согласованно работать с более новыми реализациями.

В результате, SIP стал наиболее популярным у разработчиков сетей связи следующего поколения NGN (Next Generation Network) протоколом, поддерживающим установление, изменение и завершение сеансов связи: мультимедийных конференций, телефонных соединений и соединений пользователей с разнообразными приложениями. Пользователи могут принимать участие в уже существующих сеансах связи, а также приглашать и/или быть приглашенными другими пользователями к участию во вновь создаваемом сеансе. Приглашения могут быть адресованы одному пользователю или группам пользователей.

В основу протокола SIP рабочая группа IETF MMUSIC заложила следующие принципы:

- *персональная мобильность пользователей,* основанная на присвоении пользователю уникального идентификатора, который позволяет ему перемещаться в пределах сети и получать связь в любом ее месте вне зависимости от своего местоположения, сообщаемого серверу определения местоположения при помощи специального сообщения — REGISTER;
- *масштабируемость сети* и возможность увеличения количества элементов сети, построенной на базе протокола SIP;
- *расширяемость протокола SIP,* характеризуемая возможностью дополнять протокол функциями поддержки новых услуг и его адаптации к работе с различными приложениями;
- *интеграция в стек существующих протоколов Интернет*, разработанных IETF в составе глобальной архитектуры мультимедиа и включающих в себя протокол резервирования ресурсов RSVP (Resource Reservation Protocol, RFC 2205), транспортный протокол реального времени RTP (Real-Time Transport Protocol, RFC 1889), протокол передачи потоков в реальном времени RTSP (Real-Time Streaming Protocol, RFC 2326), протокол описания параметров связи SDP (Session Description Protocol, RFC 2327);
- *взаимодействие с протоколами сигнализации* H.323, MGCP, MEGACO/H.248, DSS1 и ОКС7, включая возможность переносить в сигнальных сообщениях протокола SIP не только специфический SIP-адрес, но и телефонный номер формата E.164 или любого другого формата.

Следует отметить, что SIP не является первым и единственным протоколом IP-телефонии. Первой была зонтичная рекомендация H.323, принятая Сектором стандартизации телекоммуникаций Международного союза электросвязи (ITU–T), которая превратилась в серию довольно запутанных спецификаций, нигде не реализованных полностью, но успешно применявшихся в первых приложениях IP-телефонии при дальней связи для снижения расходов на междугородную и международную телефонную связь. И, все же, прогресс инфокоммуникаций и построение сетей связи следующего поколения NGN (Next Generation Network) связаны отнюдь не с H.323. Общепризнано, что этим целям гораздо больше соответствует протокол SIP, предложенный IETF и определяющий алгоритмы установления, модификации и завершения телефонных соединений как в процессе конвергенции сетей и услуг связи, так и в будущих конвергентных сетях NGN.

Идеологические отличия протокола SIP и свойства, общие с рассмотренными в других книгах серии «Телекоммуникационные протоколы» системами сигнализации (ОКС7, R1.5, E-DSS1, V5 и др.), представлены в табл. 1 и отчасти объясняют феноменальное распространение SIP в современных инфокоммуникациях. Среди других причин его успешного внедрения можно отметить легкость реализации и масштабируемость протокола, позволяющие пользователям с недорогостоящими ресурсами подключаться к SIP-сетям и, даже более того, — участвовать в развитии протокола путем создания разнообразных SIP-приложений и в расширении числа сторонников SIP.

Таблица 1. Сравнение систем сигнализации

Характеристики	Сигнализация в традиционных телефонных сетях	Протокол SIP
Функции протоколов	Управление установлением и разрушением соединений, управление коммутационным (транспортным) оборудованием	
Тип сети	Разработан для телефонных сетей TDM	Разработан для IP-сетей
Место применения	Существуют две группы протоколов: • между оборудованием доступа и коммутационным узлом или Softswitch — это V5.1/V5.2 или MEGACO/H.248, MGCP, SIP • между коммутационными узлами или Softswitch — это протоколы ОКС7, E-DDS1 или SIP/SIP-T, H.323	
Сетевой интеллект	Интеллект в центральных узлах сети	Интеллект может быть рассредоточен по оконечным элементам сети
Набор сигнальных сообщений	Схожий набор сигнальных сообщений: запрос установления соединения; разрушение соединения; КПВ; Занято; Ответ и пр.	
Типы соединений	Ориентация на телефонные сеансы связи	Ориентация на мультимедийные сеансы связи (речь/видео/данные)
Вид коммутации	Ориентация на коммутацию каналов	Ориентация на коммутацию пакетов (прежде всего, на IP-сети)
Открытость	Используется исключительно в сетях сигнализации	Может использоваться и оконечными терминалами, вплоть до телефонных аппаратов

При рассмотрении основ SIP в главе 1 мы вернемся к приведенным в таблице характеристикам и перепишем саму табл. 1 в более полном объеме. В главе 2 вводятся ключевые для SIP понятия *клиент агента пользователя* и *сервер агента пользователя*.

Глава 3 посвящена сообщениям SIP, а в следующих восьми главах на основе этих сообщений рассматриваются процедуры и алгоритмы, функции прокси-сервера и сервера перенаправления SIP, аспекты транспорта, защиты и безопасности, т.е. фактически рассматривается сам протокол SIP. Каждое сообщение протокола SIP содержит две части — набор заголовков и тело сообщения. Структура тела сообщения SIP обеспечивает высокую гибкость приложений. Использовавшийся первоначально только для переноса параметров SDP-сеанса, таких как параметры кодека и IP-адреса терминалов пользователей, протокол SIP может теперь переносить в составе тела сообщения многочисленные параметры. Примером такого использования тела сообщения SIP является протокол SIP-T, рассматриваемый в главе 11 и ориентированный на организацию взаимодействия ТфОП–SIP–ТфОП. Самому этому взаимодействию и преобразованию протокола общеканальной сигнализации ISUP в протокол SIP, а также SIP в ISUP, посвящены главы 12 и 13, соответственно.

Глава 14 целиком связана с телефонной нумерацией и рассматривает телефонные SIP URI. В качестве оправдания изложенным там примерам можно напомнить, что телефонная нумерация возникла в позапрошлом веке по примеру картотеки пациентов одного американского врача, причем с тех времен так и не было создано более эффективной системы преобразования адресов, которая позволила бы вызывать абонентов по именам или, например, по прозвищам. При сегодняшнем изобилии более дружественных пользовательских интерфейсов телефонная нумерация выглядит анахронизмом, но, тем не менее, и сегодня ее преобразование в SIP URI абсолютно необходимо.

Изложенные в книге материалы основаны на исследовательских работах и опыте отладки первых реализаций протокола SIP в мультисервисных коммутаторах доступа Протей-МКД на базе фрагментов NGN в сетях ряда операторов ЕСЭ РФ, за что авторы выражают признательность специалистам НТЦ Протей, коллегам-связистам Северо-Западного и Уральского регионов, инженерам-разработчикам протокол-тестеров SNT, сотрудникам сертификационного центра ЛОНИИС и кафедры телефонии СПбГУТ им. проф. М. А. Бонч-Бруевича. Авторы должны также поблагодарить своих студентов и коллег — Антона Галактионова и Александра Спирина — за существенную помощь, оказанную ими при подготовке этой книги.

Глава 1. Принципы и возможности SIP

1.1. Протокол SIP в IP-сетях

Протокол SIP (Session Initiation Protocol) является текстовым протоколом сигнализации в IP-сети и предназначен для создания, модификации и завершения сеансов связи – телефонных соединений и мультимедийных конференций, а также рассылки мультимедийной информации. Он использует многие конструктивные элементы и принципы протоколов сети Интернет, таких как HTTP и SMTP.

Согласно принципам семиуровневой модели Взаимодействия открытых систем OSI (Open Systems Interconnection) особенностью протокола SIP является независимость от технологий его смежных уровней (рис. 1.1), в частности, структура сообщений SIP не зависит от выбранной транспортной технологии. В качестве транспортных могут использоваться протоколы UDP или TCP. Протокол UDP позволяет быстрее, чем TCP, доставлять сигнальную информацию (даже с учетом повторной передачи не подтвержденных сообщений), а также вести параллельный поиск местоположения пользователей и передавать приглашения к участию в сеансе связи в режиме многоадресной рассылки. В свою очередь, протокол TCP упрощает работу с межсетевыми экранами и гарантирует надежную доставку данных. При использовании протокола TCP разные сообщения, относящиеся к одному вызову, либо могут передаваться по одному TCP-соединению, либо для каждого запроса и ответа на него может создаваться отдельное TCP-соединение.

По IP-сети может передаваться пользовательская информация практически любого вида: речь, видео и данные, а также любая их комбинация.

Во всех случаях при организации связи между терминалами пользователей необходимо известить встречную сторону о том, какого рода информация может приниматься/передаваться, а также об алгоритме ее кодирования и об адресе, по которому следует передавать информацию.

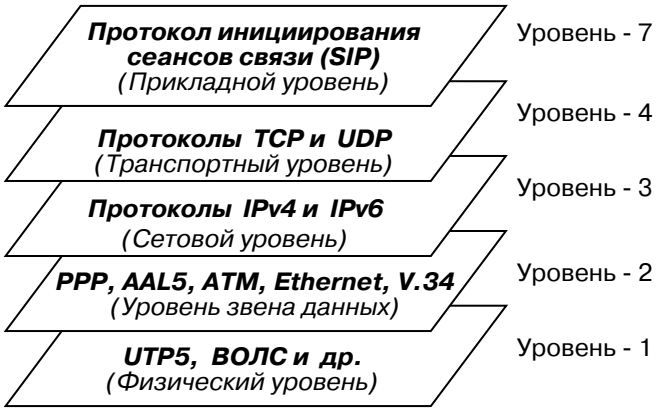


Рис. 1.1. Место протокола SIP в стеке протоколов TCP/IP

Таким образом, одним из обязательных условий организации связи при помощи протокола SIP является обмен между участниками предполагаемого сеанса связи данными об их функциональных возможностях. Для этой цели чаще всего используется отдельный *протокол описания сеансов связи SDP (Session Description Protocol)*. Сообщение протокола SDP передается в теле сообщения протокола SIP. Поскольку в течение сеанса связи может производиться его модификация (например, приглашение других пользователей к уже существующему сеансу, в частности, – к конференциям в режиме многоадресной рассылки), предусмотрена передача сообщений SIP с новыми описаниями сеанса средствами SDP. Для передачи мультимедийной информации IETF предлагает использовать протокол RTP, но сам протокол SIP не исключает возможность применения для этих целей других протоколов.

В протоколе SIP реализованы механизмы управления потоками информации и предоставления гарантированного качества обслуживания.

SIP поддерживает пять аспектов организации и завершения сеансов мультимедийной связи: определение местоположения пользователя, определение готовности пользователя участвовать в сеансе, определение функциональных возможностей терминалов пользователей (т.е. определение того, какого рода информация может использоваться, и ее параметры), установление сеанса как для вызывающей, так и для вызываемой сторон и управление сеансом связи (т.е. поддержание и завершение сеанса, модификация его параметров и активизация услуг). Кроме того, протокол SIP предоставляет возможность передачи пользовательской информации. Протокол SIP предусматривает также организацию конференций трех видов:

- в режиме многоадресной рассылки, когда информация передается на один multicast-адрес, а затем доставляется сетью конечным адресатам;
- при помощи устройства управления конференцией, к которому ее участники передают информацию в режиме точка-точка, а оно, в свою очередь, обрабатывает (т.е. смешивает или коммутирует) эту информацию и рассылает участникам конференции;
- путем соединения каждого пользователя с каждым в режиме точка-точка.

Как уже отмечалось выше, протокол SIP дает возможность подключения к уже существующему сеансу связи новых участников, т.е. двусторонний сеанс может перейти в конференцию. Расширим представленную в предисловии таблицу, приведя ее к виду табл. 1.1. Практически все строки этой новой таблицы заслуживают отдельных пояснений. Мы начнем их со строки, касающейся адресации, чему посвящен следующий параграф этой главы.

Таблица 1.1. Сравнительные характеристики протокола SIP

Характеристики	SIP	H.323	MGCP	MEGACO H.248	ISUP
Назначение	Для IP-коммуникаций	Для IP-телефонии	Для управления транспортными шлюзами	Для управления транспортными шлюзами	Для сетей TDM
Архитектура	Peer-to-Peer	Peer-to-Peer	Master-Slave	Master-Slave	Peer-to-Peer
Преемственность	Не пытается воспроизвести ТфОП	Моделирует ТфОП по аналогии с Q.931 [25]	Не пытается воспроизвести ТфОП	Наследует базовые принципы MGCP	Лежит в основе ТфОП по Q.700 [26]
Стандарты	IETF-стандарт RFC	Рекомендации ITU-T	Информац. RFC	Рек. ITU-T и IETF	Рек. ITU-T
Версии	Разные	v1-1996, v2-1998, v3 -1999	Единств. версия	Единств. версия	Национ. специфик.
Интеллект	Рассредоточен по элементам сети	В ядре сети	В ядре сети	В ядре сети	В ядре сети
Сложность	Еще простой, хотя уже содержит 13 запросов	Сложный. H.225 RAS содержит 30 сообщений, H.245 - 72 сообщения, H.255.0 – 13 сообщений	Простой	Простой	Сложный. Содержит 44 сообщения и 60 информационных элементов
Масштабируемость	Выс. степень масштаб.	Масштабируемый	–	–	Масштабируемый
Передача информации	Речь, данные и видео	Речь, данные и видео	Управление передачей речи, данных	Управление передачей речи, данных	Речь и данные
Описание функциональных возможностей оконечного оборудования	Использование протокола SDP для обмена данными о функциональных возможностях	Использование H.245 для обмена данными о функциональных возможностях	Использование SDP для обмена данными о функциях транспортных шлюзов	Использование SDP для обмена данными о функциях транспортных шлюзов	Обмен данными о функциональных возможностях с помощью информационных элементов ISUP
Контроль доступа	Контроль доступа поддерживается	Контроль доступа (управление полосой пропускания и ее контроль)	Контроль доступа на уровне IP	Контроль доступа на уровне IP	Контроль доступа посредством процедур перехода на аварийный режим
Качество обслуживания	Процедуры QoS поддерживаются	Поддержка дифферен. обслуж. (согласование скорости передачи и задержки)	Контроль QoS на уровне IP	Контроль QoS на уровне IP	QoS не требуется (предоставление выделенных некоммутируемых каналов)
Адресация	Поддержка IP-адресов и имен доменов через DNS	Поддержка IP-адресов, мультизонная, многодоменовая поддержка через привратник	Цифр. адресация терминалов пользователей, поддержка IP-адресов и имен доменов для транспорт. шлюзов	Цифр. адресация терминалов пользователей, поддержка IP-адресов и имен доменов для транспор. шлюзов	Адреса по Рек.ITU-T E.164, статические
Обнаружение закольцованных маршрутов	С помощью специальных заголовков	С помощью вектора пути	Не используется	Не используется	С помощью таймера, подсчета пересылок и сообщения Loop
Защита информации	Протоколы IPSec, TLS, SSL и HTTP Digest	Протоколы H.235, IPSec и TLS	Протоколы IPSec, TLS, SSL	Протоколы IPSec, TLS, SSL	Физическая защита
Кодирование	Текстовое кодирование	Двоичное кодирование ASN.1	Текстовое кодирование	Текстовое и двоичное кодирование	Двоичное кодирование

1.2. Адресация в сетях SIP

Для организации взаимодействия с существующими приложениями IP-сетей и для обеспечения мобильности пользователей протокол SIP использует адрес, подобный адресу электронной почты. В качестве адресов рабочих станций используются специальные универсальные указатели ресурсов – URL (Universal Resource Locators), так называемые SIP URL. SIP-адреса бывают четырех типов

- *имя@домен*,
- *имя@хост*,
- *имя@IP-адрес*,
- *№телефона@шлюз*.

Таким образом, адрес состоит из двух частей. Первая часть – это имя пользователя, зарегистрированного в домене или на рабочей станции. Если вторая часть адреса идентифицирует какой-либо шлюз, то в первой указывается телефонный номер абонента.

Во второй части адреса указывается имя домена, рабочей станции или шлюза. Для определения IP-адреса устройства необходимо обратиться к службе доменовых имен – Domain Name Service (DNS). Если же во второй части SIP-адреса размещается IP-адрес, то с узлом можно связаться непосредственно.

В начале SIP адреса ставится слово «sip:», указывающее, что это именно SIP-адрес, т.к. бывают и другие (например, «tel:»). Ниже приводятся примеры SIP-адресов:

```
sip: Alexander@niits.ru
sip: User1@192.168.0.215
sip: 333-26-27@sip-gateway.ru
```

1.3. Уровни протокола SIP

SIP представляет собой многоуровневый протокол, и если говорится, что элемент сети SIP содержит некий уровень, это означает, что поддерживается группа правил, определенных для этого уровня. Не каждый элемент, работающий по протоколу SIP, содержит все уровни, а сами элементы, специфицированные для работы в сети SIP, являются логическими, а не физическими. Физический элемент

SIP может выполнять функции разных логических элементов в зависимости от возложенных на него обязанностей.

Нижний уровень SIP отвечает за *синтаксис и кодирование*. Кодирование определено с использованием расширенных форм Бэкуса-Наура (Backus-Naur Form, BNF), кратко описанных в [26] в связи с языком SDL. Полное BNF-описание для SIP содержится в RFC 3261, а структура сообщений SIP будет рассмотрена в главе 3 книги.

Второй уровень программной реализации протокола является *транспортным*. Он определяет, как клиент передает запросы и принимает ответы и как сервер получает запросы и передает ответы по сети. Транспортный уровень протокола описан в главе 10.

Третий уровень – *уровень транзакций*. Транзакция – это запрос, переданный клиентской стороной серверной стороне с использованием транспортного уровня SIP, вместе со всеми ответами на этот запрос, переданными серверной стороной клиенту. Уровень транзакций производит повторную передачу сообщений прикладного уровня, определяет соответствие ответов запросу и уведомляет верхний уровень протокола о срабатывании таймера. Любая операция, которую выполняет клиент агента пользователя, реализуется с помощью серии транзакций. В спецификациях протокола SIP определены четыре основных функциональных элемента, которые, в зависимости от конкретных требований, либо реализуются в виде автономных компонентов, либо совмещаются на объединенной платформе :

Агенты пользователей UA (User Agents) – терминалы SIP, которые инициируют запросы, отвечают на запросы и взаимодействуют с другими агентами пользователей для организации и завершения сеансов связи. Агенты пользователей могут взаимодействовать друг с другом непосредственно, однако часто в сеанс связи бывает вовлечен один или более промежуточных серверов – прокси-серверов или серверов переадресации.

Прокси-серверы (Proxy servers) могут быть двух типов – с сохранением состояний (stateful) и без сохранения состояний (stateless); эти серверы пересылают сообщения к агентам пользователей и позволяют предоставлять такие функции, как определение местоположения пользователей, авторизация и учет. В эталонной архитектуре Международного консорциума по пакетной коммутации (IPCC) им соответствуют функция маршрутизации и функция учета.

Серверы переадресации (Redirect servers) всегда являются серверами без сохранения состояний. Они просто отвечают на запросы с указанием местоположения – адреса, по которому вызывающий пользователь может связаться прямо с требуемым вызываемым пользователем.

Серверы определения местоположения пользователей (Registration servers) позволяют агентам регистрировать свое местоположение, обеспечивая тем самым реализацию с помощью протокола SIP широкого спектра услуг мобильности.

Как будет показано далее, агенты пользователя UA и прокси-серверы с сохранением состояний транзакций (stateful proxy-servers) содержат уровень транзакций, а прокси-серверы без сохранения состояний (stateless proxy-servers), в противоположность им, уровня транзакций не содержат. Уровень транзакций имеет клиентскую часть, называемую клиентской транзакцией, и серверную часть, называемую серверной транзакцией. Каждая из них представлена конечным автоматом (state machine), связанным с обработкой запроса определенного типа.

Уровень, находящийся выше уровня транзакций, называется *пользователем транзакций (transaction user, TU)*. Каждый из объектов SIP, кроме stateless прокси-сервера, является пользователем транзакций. Когда TU желает дать запрос, он создает отдельную клиентскую транзакцию и передает ей запрос вместе с IP-адресом, данными о портах и о типе транспортного протокола для места назначения, которые определяют, куда нужно отправить запрос. Пользователь транзакций TU, который создал клиентскую транзакцию, может отменить ее. Когда клиент отменяет транзакцию, он запрашивает, чтобы сервер прекратил дальнейшую обработку запроса, возвратился в исходное состояние и передал этой транзакции ответ с определенным кодом ошибки. Это делается посредством запроса CANCEL, который создает собственную транзакцию, но выполняет свои функции в отношении отменяемой транзакции.

1.4. SIP и HTTP

Работу SIP легче всего понять с помощью модели протокола HTTP, на котором он основан. Как SIP, так и HTTP являются протоколами запроса/ответа. Клиент UA, как логический объект протокола, генерирует запросы, а сервер UA, как логический объект протокола, возвращает ответы. Когда клиенту нужен некоторый

Web-сайт, он генерирует запрос в виде URL, например www.niits.ru. Сервер, на котором размещается Web-сайт, передает в ответ Web-страничку NIITS. Протокол SIP использует ту же процедуру. Клиент UA, который передает запрос, называется UAC, а сервер UA, который передает ответ, называется UAS. Такой обмен носит название транзакция SIP.

Таким образом, для UAC протокол SIP определяет процесс создания запроса, для UAS – обработки запроса и создания ответа. Поскольку в протоколе SIP важную роль играет возможность регистрации, UAS, который может работать с запросом REGISTER, имеет свое название – сервер регистрации (registrar). Раздел 4.6 описывает работу ядер UAC и UAS при запросе REGISTER. В разделе 4.8 освещается работа UAC и UAS с запросом OPTIONS, используемым для получения информации о функциональных возможностях UA. Остальные запросы, определенные в базовом документе SIP [57], передаются в режиме диалога.

1.5. Вызовы SIP

Диалог представляет собой равноправное взаимодействие двух агентов пользователя в виде последовательности SIP-сообщений между этими UA. Запрос INVITE является единственным определенным в рекомендации RFC 3261 запросом, устанавливающим диалог. Расширения протокола определили еще два таких запроса – SUBSCRIBE и REFER.

Устройство SIP обычно функционирует и как *клиент* UA (UAC), и как сервер UA (UAS). В качестве UAC устройство SIP может инициировать запросы SIP. В качестве сервера UA устройство может принимать запросы SIP и отвечать на них. Будучи автономным устройством, UA может инициировать и принимать вызовы, которые используют SIP при равноуровневых коммуникациях. На рис. 1.2 представлен простой сценарий диалога между UAC.

Когда UAC передает запрос в режиме диалога, он, помимо выполнения общих правил UAC, описанных в следующей главе, следует правилам работы с запросами в ходе диалога. Раздел 4.1 дает понятие о диалогах и описывает процедуры их создания и поддержания, в дополнение к процедурам создания запросов в режиме диалога.

Первый запрос протокола SIP на рис. 1.2 – сообщение INVITE, которое устанавливает сеанс связи между участниками соединения. Сеанс связи – это совокупность участников соединения и медиапотоков между ними, созданных с целью обмена информацией. Раздел 4.10 описывает процедуры создания сеансов, приводящие к созданию одного или более диалогов. Раздел 4.11 повествует о том, как модифицируются параметры сеанса путем применения запроса INVITE в режиме диалога. В разделе 4.12 описано, как разрушается сеанс.

Агент пользователя вызывающей стороны преобразует имя вызываемой стороны в IP-адрес с помощью *сервера доменных имен* DNS (Domain Name System), доступного через его собственный домен. Команда INVITE передается в порт протокола UDP (User Datagram Protocol) протокола SIP и содержит информацию о формате среды и о том, от кого, кому, через кого передается запрос. Информационный ответ Trying (100) от UA вызывающей стороны аналогичен сообщению CALL PROCEEDING по стандарту Q.931 и означает, что производится маршрутизация вызова. В сценарии непосредственного вызова ответ Trying имеет другое значение, но в моделях прокси и переадресации (redirect) он используется для контроля процесса обработки вызова.

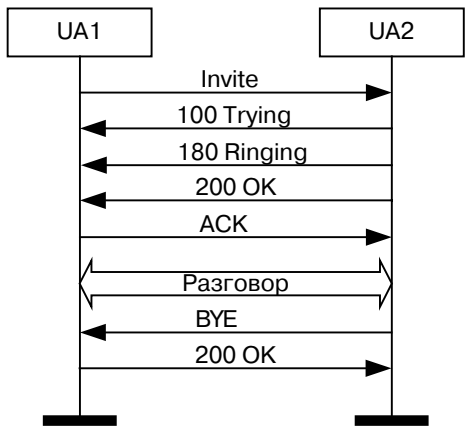


Рис. 1.2. Пример диалога UA

В табл. 1.1 сигнализация Q.931 [25] не представлена, поэтому ниже приведена табл. 1.2 соответствия сигналов SIP и DSS1 интерфейса ISDN.

Таблица 1.2. Соответствие сигналов SIP и DSS1

SIP	DSS1
INVITE	Q.931 SETUP
100 Trying	Q.931 CALL Processing
180 Ringing	Q.931 ALERTING
200 OK	Q.931 CONNECT

Когда вызов (рис. 1.2) поступает на UA2, там срабатывает вызывное устройство терминала и на UA1 передается ответ Ringing (180). Он аналогичен сообщению ALERTING, используемому в DSS1 Q.931. Временной интервал между моментом набора последней цифры и моментом получения вызывающей стороной ответа Ringing называется задержкой после набора номера PDD (postdial delay).

Когда вызываемая сторона отвечает, в UA вызывающей стороны передается ответ 200 OK. Этот UA передает сообщение ACK, подтверждающее получение ответа на запрос INVITE. В подтверждении ACK могут передаваться окончательные параметры SDP, которые поддерживает принимающий оконечный пункт. Сообщения INVITE и 200 OK аналогичны сообщениям SETUP и CONNECT по Q.931.

На этом этапе соединения транспортировка пользовательской информации обеспечивается по *протоколу RTCP* (*Real-Time Control Protocol*), который обеспечивает также контроль качества соединения и ведет статистику. Затем, как это и следует из его названия, запрос BYE с любой стороны заканчивает сеанс связи. Поскольку все сообщения переданы с помощью UDP, больше никаких действий не требуется.

В следующих главах книги будут представлены более сложные сценарии, но до этого следует подробно рассмотреть понятия UAC и UAS (глава 2) и сообщения протокола SIP (глава 3).

Глава 2. Агент пользователя

2.1. Логический объект Агент пользователя

Основным логическим объектом SIP является агент пользователя *UA (User Agent)*, который может выполнять как функции клиента агента пользователя *UAC (User Agent Client)*, генерирующего запросы, так и функции сервера агента пользователя *UAS (User Agent Server)*, который формирует ответы.

Работа UAC и UAS зависит от типа запроса и от того, происходит ли передача запроса или ответа в процессе диалога. Описанию агента пользователя посвящена следующая глава справочника.

2.2. Клиент агента пользователя UAC

Клиент агента пользователя – это часть программного обеспечения агента пользователя UA, которая создает новые запросы, отправляет их и обрабатывает принятые ответы. Запросы генерируются в результате внешних воздействий (нажатия кнопок на SIP-телефоне, сигнала из линии и т.п.).

2.2.1. Создание запроса

Запрос, составленный клиентом агента пользователя UAC, включает в себя:

- стартовую строку, в которой указан тип запроса;
- поле Request-URI и версию SIP;
- базовый набор полей заголовков: **To**, **From**, **CSeq**, **Call-ID**, **Max-Forwards** и **Via**.

Эти заголовки, как и Request-URI, обязательны для всех SIP-запросов. Они являются основными частями SIP-сообщения, поскольку совместно обеспечивают большинство требуемых услуг маршрутизации сообщений, в том числе, адресацию сообщений, маршрутизацию ответов, ограничение распространения сообщения, сохранение очередности сообщений и уникальную идентификацию транзакций. В главе 3, наряду с основными, будут подробно рассмотрены все существующие заголовки.

2.2.1.1. Формирование поля Request-URI

Поле Request-URI указывает пользователя или услугу, к которым адресован запрос. Исходное значение поля Request-URI сообщения устанавливается таким же, как URI в поле заголовка **To**. Исключение составляет запрос типа REGISTER; подробно процесс установки значения Request-URI для сообщения REGISTER описан в разделе 4.6 главы 4. По причинам обеспечения анонимности (privacy) или из соображений удобства может быть нежелательно устанавливать в полях Request-URI и заголовка **To** одинаковые значения, особенно, если инициирующий вызов UA предвидит, что Request-URI будет изменен в процессе передачи.

В некоторых случаях на значение Request-URI сообщения может повлиять наличие предустановленного маршрута. Предустановленный маршрут представляет собой упорядоченную последовательность URI, идентифицирующую последовательность серверов, через которые UAC передает исходящие запросы вне диалога. Обычно предустановленный маршрут устанавливается в UA пользователем или поставщиком услуг (service provider), или же при помощи других не-SIP механизмов. Рекомендуется задавать в качестве предустановленного маршрута один URI, соответствующий исходящему прокси-серверу. При наличии предустановленного маршрута должны выполняться процедуры заполнения полей **Request-URI** и **Route**, детализированные в § 4.3 (даже несмотря на то, что

диалога не существует), при использовании желаемого Request-URI в качестве URI удаленного узла.

2.2.1.2. Формирование заголовка To

Поле заголовка **To** определяет нужного логического получателя запроса – списочный адрес получателя (address-of-record) или ресурса, куда отправляется запрос. Значение заголовка может как быть, так и не быть адресом конечного получателя запроса. Поле **To** может содержать SIP или SIPS URI. Схема SIPS означает, что ресурсы достижимы только при условии обеспечения безопасности (например, с помощью протокола TLS).

При необходимости могут использоваться и другие URI-схемы (например, схема «tel» [75]. Все реализации SIP должны поддерживать схему SIP URI, а любая реализация, которая поддерживает протокол TLS, должна поддерживать схему SIPS URI. Поле **To** позволяет также отображать имя пользователя.

Обычно поле заголовка **To** заполняется через интерфейс пользователя вручную или с использованием адресной книги. Зачастую пользователь не вводит адрес полностью, а вместо этого вводит строку букв или цифр (например, «anton»); UA сам решает, как интерпретировать эту строку. Использование строки ввода для формирования пользовательской части SIP-адреса (user part) предполагает, что UA желает определить имя домена, находящееся по правую сторону от символа «@» SIP URI (например, *sip:anton@niits.ru*).

Запросы вне диалога не содержат в поле **To** параметр «tag». Параметр «tag» в заголовке **To** указывает один определенный из терминалов вызываемого пользователя (например, домашний, рабочий или сотовый телефон), зарегистрированных под одним SIP-адресом. «Tag» заголовка **To** в совокупности с «tag» заголовка **From** и значением поля **Call-ID** идентифицирует диалог между двумя его участниками. Если диалог не был установлен, «tag» в запросе отсутствует. Пример поля заголовка **To**:

```
To: Anton <sip:anton@niits.ru>
```

2.2.1.3. Формирование заголовка From

Поле заголовка **From** содержит логический идентификатор инициатора сообщения, как правило, списочный адрес вызывающего пользователя. Так же, как поле **To,** оно содержит URI и, опционально, отображаемое имя (display name), что

удобно для вызываемого пользователя. Заголовок используется SIP-элементами для того, чтобы определить правила обработки, применимые к запросу (например, автоматическое отклонение вызова). Важно, чтобы URI в заголовке **From** не содержал IP-адреса или FQDN (Fully Qualified Domain Name) хоста, с которым работает UA, так как это не логические имена.

Заголовок **From** предусматривает присутствие отображаемого имени (display name). Если данная информация отсутствует, UAC должен использовать отображаемое имя «Anonymous».

Обычно поле заголовка **From** запросов, которые создает UA, заполняется на основании значения, предварительно определенного пользователем или администратором локального домена. Если один UA используется несколькими пользователями, он может иметь переключаемые профили, которые содержат URI, соответствующие определенным пользователям. Получатели запросов могут аутентифицировать инициаторов запросов для того, чтобы убедиться, что они – те, кого представляют заголовки **From** этих запросов.

Поле **From** должно содержать параметр «tag», созданный клиентом UA.

Примеры:

```
From: «Anton» <sips:anton@niits.ru> 9;tag=a48s
From: sip:+79213434329@gateway.protei.ru;tag=887s
From: Anonymous <sip:c8oqz84zk7z@privacy.org>;tag=hyh8
```

2.2.1.4. Формирование заголовка Call-ID

Заголовок **Call-ID** – это уникальный идентификатор, объединяющий группу сообщений. Он должен быть одинаков у всех запросов и ответов, передаваемых любым из двух UA в процессе диалога.

При создании нового диалога, заголовок **Call-ID** должен быть выбран UAC как новый уникальный идентификатор. Все SIP-агенты пользователя должны иметь средства, гарантирующие, что **Call-ID**, созданный ими, не будет случайно генерирован другим UA. Когда запросы передаются повторно после получения ответа с кодом ошибки, требующего коррекции запроса (например, запрос отклика аутентификации), они не рассматриваются как новые и не нуждаются в новом значении заголовка **Call-ID** (см. п. «Обработка ответов класса 4xx» в § 2.2.2.5).

При генерации значений **Call-ID** рекомендуется использовать случайные криптографические идентификаторы [12]; их использование обеспечивает некоторую защиту от взлома сеансов связи и уменьшает вероятность возникновения коллизий **Call-ID**. Реализации могут использовать форму localid@host. Значения заголовка **Call-ID** чувствительны к регистру и должны сравниваться побайтно. Пример:

```
Call-ID: f81d4fae-7dec-11d0-a765-00a0c91e6bf6@niits.ru
```

2.2.1.5. Формирование заголовка CSeq

Поле заголовка **CSeq** служит средством идентификации и упорядочения транзакций. Оно содержит порядковый номер и указание типа запроса. Для запросов вне диалога, кроме REGISTER, значение порядкового номера может быть произвольным. Величина порядкового номера выражается 32-разрядным целым числом и должна быть меньше, чем 2^{31}. Для создания значений заголовка **CSeq** клиент может выбирать любой механизм. Пример:

```
CSeq: 4711 INVITE
```

2.2.1.6. Формирование заголовка Max-Forwards

Поле заголовка **Max-Forwards** служит для ограничения числа пересылок запроса на пути к месту назначения. Оно содержит целое число, которое уменьшается на единицу при каждой пересылке. Если значение этого заголовка достигнет 0 до того, как запрос достигнет места назначения, сообщение будет отклонено ответом с кодом ошибки 483 (Too Many Hops). UAC должен помещать заголовок **Max-Forwards** в каждый передаваемый запрос. Рекомендуемое его значения: 70. Оно выбрано достаточно большим для того, чтобы гарантировать, что запрос не будет отброшен сетью SIP при отсутствии петель, но и не слишком большим, чтобы не загружать ресурсы прокси-сервера при возникновении петли. Меньшие величины рекомендуется использовать с осторожностью и только в сетях, где агенту пользователя известна топология сети.

2.2.1.7. Формирование заголовка Via

Поле заголовка **Via** указывает один из узлов (например, прокси-сервер), используемых для проведения транзакции и идентифицирует место (location), куда должен быть отправлен ответ. SIP-элемент добавляет собственное значение заголовка **Via** только после выбора следующего узла, которому будет передан запрос. Когда UAC создает запрос, он должен поместить в него поле **Via**. Необходимо также

указать название протокола – SIP – и его версию – 2.0. Поле заголовка **Via** должно содержать параметр «branch», который служит для идентификации транзакции, созданной запросом. Он используется и клиентом, и сервером. Значение параметра «branch» должно быть уникальным для всех запросов, передаваемых UA. Исключение составляют запрос CANCEL и запрос-подтверждение ACK приема ответов, отличных от класса 2xx. Запрос CANCEL будет иметь то же значение параметра «branch», что и запрос, который он отменяет. Запрос-подтверждение ACK приема ответа, отличного от класса 2xx, будет иметь тот же параметр «branch», что и INVITE, прием ответа на который он подтверждает. Уникальность этого параметра облегчает его использование в качестве идентификатора транзакции. Параметр «branch», вводимый в **Via** элементом сети SIP, должен всегда начинаться с «z9hG4bK». Эти семь символов, называемых «magic cookie», используют для того, чтобы серверы, получившие запрос, могли определить, что идентификатор транзакции уникален.

Другие параметры заголовка **Via** («maddr», «ttl» и «sent-by») будут определены во время обработки запроса транспортным уровнем протокола SIP.

2.2.1.8. Формирование заголовка Contact

Поле заголовка **Contact** содержит SIP или SIPS URI, который может быть использован для связи с пользователем UA, передавшим сообщение (получатель может использовать его для своих будущих запросов). Этот заголовок должен присутствовать и содержать только один SIP или SIPS URI в любом запросе, результатом которого может стать организация диалога, каковым, например, является INVITE. Для таких запросов масштаб заголовка **Contact** должен быть глобальным, т.е. значение поля заголовка **Contact** должно содержать URI, на который UA желает получать запросы, и этот URI должен быть действительным, даже если используется в следующих запросах вне диалога. Если поле Request-URI или первое значение заголовка **Route** содержит SIPS URI, поле **Contact** тоже должно содержать SIPS URI. Заголовок **Route** служит для принудительной маршрутизации запроса в соответствии со списком прокси-серверов.

2.2.1.9. Формирование заголовков Supported и Require

Если UAC поддерживает расширения SIP, определяющие новые функции для SIP-элементов, которые могут быть использованы сервером в ответах, UAC должен включить в запрос заголовок **Supported** со списком идентификаторов

(option tag) поддерживаемых функций. Список option-tag идентифицирует новые функциональные возможности для SIP в соответствии с расширениями SIP, определенными в дополнительных RFC. Это делается с целью предотвратить требование серверов реализации клиентами не стандартизированных определенных фирмой-производителем функций для того, чтобы получить обслуживание. Расширения, которые описаны в RFC, не имеющих статуса Standard, не используются в заголовке **Supported** запроса, так как фирмы-производители зачастую используют их для реализации нестандартизованных расширений. Если UAC хочет потребовать, чтобы UAS понял расширение, которое UAC применит к запросу для его обработки, он должен поместить в запрос поле заголовка **Require**, указывающее option-tag для этого расширения. Если UAC хочет применить расширение к запросу и потребовать, чтобы каждый пройденный прокси-сервер понимал это расширение, он должен ввести в запрос заголовок **Proxy-Require**, указывающий option-tag для этого расширения.

2.2.1.10. Дополнительные компоненты сообщения

После того как новый запрос создан, и заголовки, описанные выше, составлены должным образом, в сообщения добавляются необязательные заголовки в соответствии с типом запроса.

SIP-запросы могут содержать закодированное MIME тело сообщения [18]. Независимо от типа тела сообщения определены заголовки, описывающие содержимое тела (заголовки **Content-Disposition**, **Content-Encoding**, **Content-Language**, **Content-Length**, **Content-Type**, рассматриваемые в § 3.2).

2.2.1.11. Передача запроса

При передаче запроса первоначально определяется место назначения. Если местная политика безопасности не определена по-иному, адрес места назначения должен быть определен с применением DNS-процедур, описанных в [56]: «SIP: Locating SIP Servers». Если первым в маршруте стоит strict router (прокси-сервер, удаляющий содержимое Request-URI при наличии в сообщении заголовка **Route**), то вышеуказанные DNS-процедуры должны быть применены к полю Request-URI, содержащемуся в стартовой строке запроса. В противном случае эти процедуры применяются к первому значению заголовка **Route** или к Request-URI, если заголовок **Route** отсутствует.

Процедуры устанавливают упорядоченную последовательность, состоящую из адреса, порта и типа транспортного протокола для запроса. Независимо от того, какой URI используется в качестве входящего для процедур «Locating SIP Servers», если Request-URI указывает на SIPS-ресурс, UAC должен выполнять эти процедуры, как если бы входящим URI был SIPS URI.

Местная политика может предусматривать для использования в запросах набор альтернативных мест назначения. Если Request-URI содержит SIPS URI, то соединение с любым альтернативным местом назначения должно проходить с использованием протокола TLS. Более того, ограничений для альтернативных мест назначения не существует, если запрос не содержит заголовка **Route**. Это представляет собой упрощенную альтернативу предустановленному маршруту, как способу указать исходящий прокси-сервер. Однако такой подход к конфигурации исходящего прокси-сервера не рекомендуется; вместо этого лучше использовать предустановленный маршрут с единственным URI. Если запрос содержит поле заголовка **Route**, то он будет передан на сервер, определенный в верхнем значении **Route**, но может быть направлен и на другой сервер, придерживающийся той же политики в отношении **Route** и Request-URI, что и UA. В частности, UA, сконфигурированный исходящим прокси-сервером, должен пытаться передать запрос по адресу, указанному в первом значении поля заголовка **Route**, вместо передачи всех сообщений исходящему прокси-серверу. Этим гарантируется, что исходящие прокси-серверы, которые не добавляют поле заголовка **Record-Route**, выпадут из маршрута следующих запросов. Оконечные точки, которые не могут определить первое значение для поля заголовка **Route**, делегируют выполнение этой задачи исходящему прокси-серверу.

UAC должен следовать процедурам, определенным в «SIP: Locating SIP Servers», [56] для stateful SIP-элементов, передавая запросы по каждому адресу, пока не будет установлено соединение с сервером. Каждая попытка составляет новую транзакцию, и поэтому каждый новый запрос содержит заголовок **Via** с новым параметром «branch» в первом значении.

2.2.2. Обработка ответов

Ответы сначала обрабатываются транспортным уровнем SIP, а потом направляются на уровень транзакций. Уровень транзакций производит их обработку и затем передает вышестоящему уровню – уровню пользователя транзакций

(transaction user, TU). Большая часть обработки ответов TU зависит от типа запроса, на который был передан ответ. Однако некоторые основные этапы обработки не зависят от типа запроса.

2.2.2.1. Ошибки уровня транзакций

В некоторых случаях ответ, переданный уровнем транзакций, не является SIP-сообщением; это означает уведомление об ошибке уровня транзакций. Когда с уровня транзакций приходит уведомление об ошибке, определяемой истечением времени таймера (timeout error), оно должно обрабатываться как ответ с кодом 408 (Request Timeout). Когда с транспортного уровня SIP приходит сообщение о критической ошибке (fatal transport error), его следует трактовать как ответ 503 (Service Unavailable). Обычно это означает критическую ошибку в протоколе UDP или нарушение соединения в TCP.

2.2.2.2. Неизвестные ответы

UAC должен расценивать любой неизвестный окончательный ответ, как эквивалентный x00 ответу того же класса, и должен быть готов обрабатывать ответы x00 любого класса. Например, если UAC получает неизвестный ответ с кодом 431, он делает вывод о том, что запрос содержал ошибку, и трактует его как 400 (Bad Request). UAC должен трактовать любой неизвестный предварительный ответ, отличный от 100, как ответ с кодом 183 (Session Progress). UAC должен быть способен обрабатывать ответы и 100, и 183.

2.2.2.3. Заголовки Via

Если в ответе представлено более одного значения поля заголовка **Via**, UAC должен отбросить сообщение, так как в этом случае сообщение, по-видимому, было неправильно маршрутизировано или искажено.

2.2.2.4. Обработка ответов группы 3xx

При приеме ответа перенаправления (например, ответа с кодом 301), клиенты должны использовать адрес (адреса) из поля **Contact** при составлении одного или нескольких новых запросов, основанных на перенаправленном запросе. Клиент начинает работу с начального списка адресов вызываемого пользователя (target set), включающего в себя только один URI – Request-URI оригинального запроса.

Если отправитель желает сформировать новые запросы, получив на предшествующий запрос перенаправляющий ответ класса 3xx, он помещает новые адреса в target set. UAC может выбрать, какие из URI, расположенных в поле **Contact**, поместить в target set. Клиент, обрабатывающий ответы класса 3xx, не должен добавлять один и тот же URI в target set более одного раза. Если оригинальный запрос содержит SIPS URI в Request-URI, клиент может перенаправить запрос на не-SIPS URI, но должен информировать пользователя о перенаправлении запроса на небезопасный URI.

По мере того как target set растет, UAC может генерировать новые запросы, используя адреса, выбранные из target set в любом порядке. Простейшим механизмом для этого является упорядочение в соответствии со значением параметра «q» каждого значения заголовка **Contact**. Параметр «q» определяет приоритеты среди адресов, содержащихся в заголовке **Contact**, путем варьирования их значения в пределах от 0 до 1. Запросы могут генерироваться последовательно или параллельно. Один подход состоит в обработке групп в порядке уменьшения значения параметра «q» последовательно и обработке адресов в каждой группе с определенным значением параметра «q» параллельно. Другой подход предусматривает последовательную обработку в порядке уменьшения значения параметра «q», произвольно выбирая адреса с одинаковым значением q. Если при обращении на контактный адрес из списка приходит ответ об ошибке, SIP-элемент переходит к следующему адресу в списке. Когда список заканчивается, запрос отбрасывается. Ошибки определяются по кодам ответов (коды со значением более 399). Об ошибках, произошедших при передаче по сети (ошибках транспортного уровня), пользователю транзакций (TU) сообщает клиентская транзакция. Некоторые коды ответов показывают, что запрос может быть передан снова; такие ответы не должны расцениваться как ошибки. Когда приходит ответ об ошибке при обращении на определенный контактный адрес, клиент должен попытаться передать запрос на следующий контактный адрес. Это повлечет за собой создание новой клиентской транзакции.

Для того чтобы создать запрос на основании контактного адреса, полученного в ответе класса 3xx, UAC должен полностью скопировать URI из списка адресов target set в Request-URI, за исключением параметров «method-param» и «header». Параметры «header» используются для создания значений полей заголовков новых запросов, заменяя значения, связанные с перенаправленным запросом.

В некоторых случаях в контактном адресе (адресе, содержащемся в заголовке **Contact**) указываются другие заголовки. Значения этих заголовков могут быть добавлены к значениям заголовков в оригинальном перенаправленном запросе. Как правило, если поле заголовка может принимать разделенный запятыми список значений параметров, то новое значение поля заголовка может быть добавлено к любым существующим значениям оригинального перенаправленного запроса. Если же поле заголовка не поддерживает множественность значений, то значение в оригинальном перенаправленном запросе может быть заменено значением из контактного адреса. Например, если контактный адрес возвращен со следующим значением:

```
sip:anton@niits.ru?Subject=organization&Call-Info=http://www.niits.ru,
```

то любое поле заголовка **Subject** в оригинальном перенаправленном запросе заменяется, но HTTP URL просто добавляется к существующим значениям поля заголовка **Call-Info**.

Рекомендуется, чтобы UAC использовал поля заголовков **To**, **From** и **Call-ID**, применявшиеся в перенаправленном запросе, но UAC, например, может обновить значение поля заголовка **Call-ID** для новых запросов. В результате, сформированный новый запрос отправляется с использованием новой клиентской транзакции и, следовательно, он будет иметь новое значение параметра «branch» в верхнем поле **Via**. В остальном, запросы, отправленные при получении ответа перенаправления, будут иметь те же поля заголовков и тела сообщения, что и оригинальный запрос. В некоторых случаях значения поля заголовка **Contact** могут запоминаться клиентом временно или постоянно в зависимости от кода ответа и присутствия информации о истечении времени действия.

2.2.2.5. Обработка ответов группы 4xx

Отдельные коды ответов класса 4xx при их обработке требуют от UA особых действий вне зависимости от типа запроса, на который приходит ответ.

- Ответ с кодом *401 (Unauthorized)* или *407 (Proxy Authentication Required)* означает, что запрос требует проведения процедур аутентификации. Получив ответ, UAC должен будет выполнить процедуру аутентификации для следующего запроса.
- Код ответа *413 (Request Entity Too Large)* означает, что тело запроса имеет слишком большую длину для того, чтобы UAS мог его принять. При этом UAC заново передает запрос, убирая тело сообщения или уменьшая его.

- Код ответа *415 (Unsupported Media Type)* означает, что формат данных, содержащихся в запросе – тип тела сообщения, – не поддерживается UAS. В этом случае UAC должен заново отправить запрос, приняв во внимание список поддерживаемых типов в поле заголовка **Accept**, список поддерживаемых кодеков в поле **Accept-Encoding** и список поддерживаемых языков в поле **Accept-Language**, содержащихся в ответе.
- Ответ с кодом *416 (Unsupported URI Scheme)* означает, что тип URI, использованный в Request-URI, не поддерживается сервером. В этом случае UAC должен заново послать запрос, используя SIP URI.
- Ответ с кодом *420 (Bad Extension)* означает, что запрос содержит заголовки **Require** или **Proxy-Require**, указывающие option-tag для функции, которая не поддерживается прокси-сервером или UAS. В этом UAC случае должен заново передать запрос, убрав из него все расширения, указанные в поле **Unsupported** ответа.
- Ответ с кодом *494 (Security Agreement Required)* передается сервером при выборе механизма обеспечения безопасности. Ответ должен включать в себя заголовок **Security-Server** со списком механизмов обеспечения безопасности, поддерживаемых сервером. UAC должен выбрать подходящий механизм безопасности и применить его при передаче запроса.

Во всех описанных выше случаях запрос передается заново с соответствующими изменениями. Новый запрос составляет новую транзакцию и будет иметь такие же значения заголовков **Call-ID**, **To** и **From**, как и предыдущий запрос, но заголовок **CSeq** должен содержать новый порядковый номер, на единицу больший предыдущего. При получении других ответов класса 4xx передача запроса может производиться заново в зависимости от типа запроса и от случая использования.

2.3. Сервер агента пользователя (UAS)

Сервер UAS агента пользователя принимает запросы и генерирует ответы, основываясь на действиях пользователя, полученных сообщениях, результатах выполнения программ или на каких-либо других событиях.

2.3.1. Процедура обработки запросов

При обработке сервером запроса вне диалога выполняется набор процедур обработки, не зависящих от типа запроса. Заметим, что обработка запросов элементарна. Если запрос принимается, должны быть произведены любые связанные с ним изменения состояния соединения, а если он отклоняется, ни одно из изменений производится не должно.

Серверы UAS агента пользователя обрабатывают запросы пошагово (сначала аутентификация, затем анализ типа запроса, анализ полей заголовков и так далее).

2.3.1.1. Определение типа запроса

Когда запрос прошел аутентификацию (или она была пропущена), UAS должен выяснить тип запроса. Если UAS определил тип запроса, но не поддерживает его, он должен передать ответ с кодом 405 (Method Not Allowed); в этом ответе должен присутствовать заголовок **Allow**, который содержит список типов запросов, поддерживаемых UAS. В случае поддержки сервером типа запроса, обработка сообщения продолжается.

2.3.1.2. Определение типа заголовка

Если UAS не понимает заголовка, представленного в запросе, он должен игнорировать этот заголовок и продолжать обработку сообщения. UAS игнорирует все неопознанные заголовки, которые необязательны для обработки запроса.

2.3.1.3. Обработка полей To и Request-URI

В поле заголовка **To** вызывающий пользователь указывает адрес получателя запроса. При выработке решения о приеме запроса, заголовок **To** которого идентифицирует другой UAS, данный UAS может поступать произвольным образом. Однако рекомендуется, чтобы UAS принимал запросы, даже если они содержат неизвестную схему URI (например, схему «tel») в поле **To**, или если поле **To** не содержит адреса ни текущего, ни одного из существующих пользователей этого UAS. Если UAS, все же, решает отклонить запрос, он должен создать ответ с кодом 403 (Forbidden) и отправить его серверной транзакции (понятие серверной транзакции дается в разделе 4.4) для передачи.

Поле Request-URI идентифицирует UAS, который должен обработать запрос. Если в Request-URI используется схема адресации, не поддерживаемая сервером, запрос должен быть отклонен, и должен быть отправлен ответ с кодом 416 (Unsupported URI Scheme). Если Request-URI не идентифицирует адрес, для которого UAS готов принять запрос, сообщение отбрасывается, и передается ответ с кодом 404 (Not Found). Обычно UA, который использует сообщение REGISTER, чтобы связать списочный адрес пользователя (address-of-record) с определенным контактным адресом, должен опознавать запросы, Request-URI которых совпадает с его контактным адресом. Другие возможные источники для значения полученного Request-URI – заголовки **Contact** запросов и ответов, отправленных UA для установления или обновления параметров диалога.

2.3.1.4. Обработка одинаковых запросов

Если запрос не содержит параметра «tag» в поле заголовка **To**, ядро UAS (UAS core) должно проверить запрос на предмет происходящих транзакций. Если «tag» заголовка **From**, заголовки **Call-ID** и **CSeq** точно совпадают с аналогичными полями, связанными с происходящей транзакцией, но запрос не соответствует этой транзакции, ядро UAS должно составить ответ с кодом 482 (Loop Detected) и передать его серверной транзакции. Одинаковые запросы могут придти на сервер более одного раза, следуя разными путями, вероятнее всего, из-за размножения запросов прокси-сервером. Ядро UAS обрабатывает первый такой запрос и отправляет ответ с кодом 482 (Loop Detected) на остальные такие же запросы.

2.3.1.5. Обработка заголовка Require

После того как UAS решает, что запрос предназначен для него, он приступает к анализу заголовка **Require** (в случае его наличия).

Поле заголовка **Require** используется UAC, чтобы сообщить UAS о SIP-расширениях, которые тот должен поддерживать для правильной обработки запроса. Если UAS не понимает какого-либо идентификатора расширения option-tag, указанного в поле **Require**, он должен передать ответ с кодом 420 (Bad Extension). UAS должен добавить в ответ заголовок **Unsupported** со списком непонятных опций, указанных в поле **Require** запроса. Заметим, что заголовки **Require** и **Proxy-Require** не должны использоваться в запросе CANCEL, а также в запросе ACK, подтверждающем получение ответа, отличного от класса 2xx. Эти заголовки должны игнорироваться, даже если они имеются в таких запросах.

Запрос ACK, подтверждающий прием ответа класса 2xx, должен содержать только те значения **Require** и **Proxy-Require**, которые присутствовали в начальном запросе, например:

```
UAC → UAS:      INVITE sip:vladimir@protei.ru SIP/2.0
                Require: 100rel

UAS → UAC:      SIP/2.0 420 Bad Extension
                Unsupported: 100rel
```

Это гарантирует, что взаимодействие между клиентом и сервером будет проходить без задержек, когда все опции понятны обеим сторонам, и будет замедляться, только если существуют разногласия, как в приведенном выше примере. Для хорошо согласованной пары клиент-сервер взаимодействие происходит быстро, так как не тратится время на работу механизмов согласования. К тому же, это устраняет проблемы, возникающие, когда клиент требует возможности, которые сервер не поддерживает.

2.3.1.6. Обработка содержимого тела сообщения

Допустим, что UAS понимает все расширения, требуемые клиентом; тогда UAS изучает тело сообщения и поля заголовков, которые описывают его. Если в сообщении содержится тело с непонятным типом (указывается в **Content-Type**), языком (указывается в **Content-Language**) или кодеком (указывается в **Content-Encoding**), и это тело является обязательным (указывается в **Content-Disposition**), UAS должен отбросить запрос и отправить ответ с кодом 415 (Unsupported Media Type). В случае наличия в запросе тел сообщения с типами, не поддерживаемыми сервером, ответ должен содержать заголовок **Accept** со списком всех типов тел сообщения, которые понимает UAS. Если запрос содержит типы кодеков содержимого, непонятные UAS, ответ должен содержать заголовок **Accept-Encoding** со списком кодировок, ему понятных. Если запрос имеет содержимое на непонятном для UAS языке, ответ должен содержать заголовок **Accept-Language** со списком языков, понятных UAS. После этих проверок, тело обрабатывается в зависимости от типа запроса и типа тела.

2.3.1.7. Применение расширений

При формировании ответа UAS не должен использовать расширения, если поддержка этих расширений клиентом не была указана в заголовке **Supported** запроса. Если требуемые расширения не поддерживаются, сервер должен опираться

только на основные SIP-расширения и другие расширения, поддерживаемые клиентом. В редких случаях, когда сервер не может обработать запрос без требуемого расширения, он может отправить ответ с кодом 421 (Extension Required). Этот ответ показывает, что правильный ответ не может быть сформирован без поддержки определенного расширения. Требуемое расширение или расширения должны быть включены в поле **Require** ответа. Любые расширения к ответам, кроме ответа с кодом 421, должны быть указаны в заголовке **Require**, включенном в ответ. Сервер не должен применять расширения, не указанные в заголовке **Supported** запроса. После того как все вышеописанные действия выполнены, дальнейшая обработка запроса зависит от его типа.

2.3.2. Создание ответа

Когда UAS создает ответ на запрос, он следует общим процедурам, описанным ниже. Могут потребоваться также дополнительные действия, которые зависят от кода ответа. После завершения всех процедур, связанных с созданием ответа, UAS передает ответ серверной транзакции, от которой был получен запрос.

2.3.2.1. Передача предварительного ответа

Основное правило, вне зависимости от типа запроса, состоит в том, что серверы UA должны передавать предварительные ответы только на запросы INVITE. При этом они должны как можно быстрее создавать окончательные ответы на все запросы, кроме INVITE. При формировании ответа с кодом 100 (Trying) в него из запроса должен быть в точности скопирован заголовок **Timestamp** (указывает время, когда UAC передал сообщение UAS). Это выполняется для оценки клиентом времени RTT. Если происходит задержка при генерировании ответа, UAS должен добавить значение задержки к значению, содержащемуся в поле заголовка **Timestamp** в ответе. Это значение должно содержать разницу между временем получения запроса и временем передачи ответа, выраженную в секундах.

2.3.2.2. Заголовки и параметры «tag»

Поле **From** ответа должно совпадать с аналогичным полем запроса, равно как и поля **Call-ID**, **Cseq** и **Via** ответа должны совпадать с полями **Call-ID**, **Cseq** и **Via** запроса. Помимо этого, значения поля **Via** должны совпадать со значениями поля **Via** в запросе и должны сохранять порядок следования. Если запрос содержал «tag» в поле **To**, то поле **To** в ответе должно совпадать с тем, что было в запросе. В случае

если поле **To** в запросе не содержит «tag», URI в поле **To** ответа должен совпадать с URI в поле **To** запроса; дополнительно, UAS должен добавить «tag» в поле **To** ответа (за исключением ответа с кодом 100 (Trying), в котором «tag» может уже присутствовать). Тот же «tag» должен быть использован для всех ответов на этот запрос, предварительных и окончательных (исключая ответ с кодом 100 (Trying)).

2.3.2.3. Действие UAS без сохранения состояний

UAS без сохранения состояний (stateless) – это UAS, который не запоминает состояния текущих транзакций. Он нормально обрабатывает запросы, но, в отличие от UAS с сохранением состояний (stateful), не сохраняет состояния транзакций после передачи ответов. Если stateless UAS получает повторно переданный запрос, он повторно формирует ответ и повторно его отсылает так же, как это происходило при получении первого запроса. UAS может быть stateless , только если обработка идентичных запросов одного типа приводит к генерации одинаковых ответов. Stateless UAS не использует уровень транзакций: он принимает запрос прямо от транспортного уровня SIP и передает ответ тоже непосредственно транспортному уровню. Основная роль stateless UAS состоит в поддержке не аутентифицированных запросов, которые требуют передачи ответа с запросом аутентификации. Если бы эти запросы поддерживались с сохранением состояния, то возможные потоки злонамеренных запросов, не содержащих отклика аутентификации, сопровождались бы созданием большого числа транзакций, что, в свою очередь, могло бы замедлить или остановить обработку вызовов в UAS.

Наиболее важные функции stateless UAS перечислены ниже:

- stateless UAS не должен передавать предварительные (1xx) ответы;
- stateless UAS не должен повторно передавать ответы;
- stateless UAS должен игнорировать запросы ACK;
- stateless UAS должен игнорировать запросы CANCEL;
- параметры «tag» заголовка **To** должны формироваться для ответов таким образом, что для одинаковых запросов должны генерироваться одинаковые «tag».

В остальном, stateless UAS работает так же, как stateful UAS. Для каждого нового запроса UAS может выбрать, как работать – с сохранением или без сохранения состояний.

Глава 3. Сообщения протокола SIP

3.1. Структура сообщений

В главе 1 отмечалось, что SIP является текстовым протоколом, использующим набор символов ISO 10646 в кодировке UTF-8 согласно [77]. Сообщение протокола SIP представляет собой либо *запрос от клиента серверу*, либо *ответ сервера клиенту*. Запросы и ответы используют одинаковый базовый формат сообщения и различаются наборами символов и синтаксисом. Сообщение того и другого типа (рис. 3.1) состоит из:

- стартовой строки;
- одного или нескольких полей заголовков;
- пустой строки, обозначающей конец полей заголовков;
- тела сообщения (необязательно).

Стартовая строка, каждая строка поля заголовка и пустая строка должны быть завершены символами возврата каретки CRLF. Пустая строка должна существовать независимо от того, присутствует тело сообщения или нет.

Стартовая строка представляет собой начальную строку любого SIP-сообщения. Если сообщение является запросом, в этой строке указывается тип запроса, адресат и номер версии протокола. Если сообщение является ответом на запрос, в стартовой строке указывается номер версии протокола, тип ответа и его короткая расшифровка, предназначенная только для пользователя.

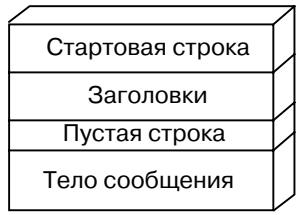


Рис. 3.1. Структура сообщения протокола SIP

Заголовки сообщений служат для передачи информации об отправителе, адресате, пути следования и других сведений, т.е. переносят необходимую для обслуживания данного сообщения информацию. О типе заголовка можно узнать из его имени. За исключением различий в наборе символов, многие SIP-сообщения и синтаксис полей заголовков схожи с используемыми в HTTP/1.1, хотя SIP и не является расширением HTTP, что уже обсуждалось в главе 1.

Сообщения протокола SIP могут содержать так называемое *тело сообщения*. В запросах ACK, INVITE и OPTIONS тело сообщения содержит описание сеансов связи, например, в формате протокола SDP, а запрос BYE, например, тело сообщения не предусматривает.

3.2. Заголовки сообщений

3.2.1. Формат заголовка

Поля заголовков SIP-сообщений похожи на поля заголовков HTTP-сообщений по синтаксису и семантике. В частности, поля заголовков SIP соответствуют описаниям синтаксиса HTTP/1.1 для заголовков сообщений и правилам для расширения полей заголовков на несколько строк. Каждое поле заголовка состоит из имени поля, символа «двоеточие» и значения поля:

```
Имя поля: значение поля
```

Формальная грамматика полей заголовков позволяет использовать любое количество пробелов (SP) для отделения двоеточия. Однако на деле стараются избегать пробелов между названием поля и двоеточием и ограничиваются одним

пробелом для отделения двоеточия от значения. В следующих примерах все четыре строки равнозначны, но последняя является общепринятой:

```
Subject:  уведомление
Subject : уведомление
Subject  :уведомление
Subject: уведомление
```

Поля заголовков могут быть расширены на несколько строк, тогда каждая следующая строка отделяется пробелом (SP) или символом горизонтальной табуляции (HT). Обрыв строки (line break) и пустое пространство (whitespase) расцениваются как один символ пробела SP. Следующие ниже поля эквивалентны.

```
Subject: Я знаю, что ты здесь, подними трубку!
Subject: Я знаю,
            что ты здесь,
            подними трубку!
```

Порядок следования заголовков не имеет значения. Однако рекомендуется размещать поля заголовков, которые требуются для обработки прокси-серверу (**Via**, **Route**, **Record-Route**, **Proxy-Require**, **Max-Forwards**, **Proxy-Authorization** и другие), в начале сообщения, чтобы ускорить анализ и обработку. Важным является порядок следования ряда заголовков с одинаковыми именами полей. Последовательности полей заголовков с одинаковыми именами могут содержаться в сообщении только в том случае, если содержимое поля представляет собой список значений, разделенных запятой. Возможно объединить такие заголовки в одну пару «имя поля: значение поля» не изменяя семантики сообщения, а путем добавления каждого следующего значения к первому значению поля заголовка; при этом все значения должны быть отделены друг от друга запятой. Исключение составляют лишь заголовки **WWW-Authenticate**, **Authorization**, **Proxy-Authenticate** и **Proxy-Authorization**. Последовательности заголовков с такими именами тоже могут присутствовать в сообщении, но объединить их невозможно, поскольку грамматика этих заголовков не подчиняется общим для SIP-заголовков правилам.

Программные реализации SIP должны быть способны обработать последовательности заголовков с одинаковым именами и со значениями, которые представлены как в форме последовательности, разделенной запятыми, так и в виде «одно значение на строку». Примеры последовательностей заголовков, приведенные ниже, правомерны и эквивалентны:

```
Route:      <sip:anton@niits.ru>
Subject:    Уведомление
Route:      <sip:vladimir@protei.ru>
```

```
Route:      <sip:alexander@niits.ru>

Route:      <sip:anton@niits.ru>, <sip:vladimir@protei.ru>
Route:      <sip:alexander@niits.ru>
Subject:    Уведомление

Subject:    Уведомление
Route:      <sip:anton@niits.ru>, <sip:vladimir@protei.ru>,
            <sip:alexander@niits.ru>
```

Следующие же последовательности заголовков правомерны, но не равнозначны:

```
Route: <sip:anton@niits.ru>
Route: <sip:vladimir@protei.ru>
Route: <sip:alexander@niits.ru>

Route: <sip:vladimir@protei.ru>
Route: <sip:anton@niits.ru>
Route: <sip:alexander@niits.ru>

Route: <sip:anton@niits.ru>,<sip:alexander@niits.ru>,
       <sip:vladimir@protei.ru>
```

Формат значения заголовка зависит от имени заголовка. Это всегда будет последовательность текстовых октетов в кодировке UTF-8 или комбинация символьных фраз (tokens), пустого пространства (whitespace), разделительных знаков и строк, заключенных в кавычки. В большинстве существующих полей заголовков исполняется общий формат, который основан на последовательности пар *имя параметра – значение параметра*, разделенных знаком «точка с запятой»:

```
Имя поля: значение поля; имя параметра=значение параметра;
имя параметра=значение параметра
```

Несмотря на то, что заголовок может содержать неограниченное число параметров, одно и то же имя параметра не может использоваться более одного раза.

Для имен полей заголовков не имеет значения, в каком регистре они написаны. Значения полей, имена параметров и значения параметров также регистронезависимы, если это не определено по-иному в описании заголовка. Если не определено иначе, значения, заключенные в кавычки, являются зависимыми от регистра.

Contact: <*sip:anton@niits.ru*>; expires=3600 эквивалентно CONTACT: <*sip:anton@niits.ru*>; ExPiReS=3600. Точно так же Content-Disposition: session; handling=optional эквивалентно *content-disposition: Session; HANDLING=OPTIONAL.*

А вот два следующих поля заголовков не равнозначны:

```
Warning:370 niits.ru «Требуется большая пропускная способность»
Warning:370 niits.ru «ТРЕБУЕТСЯ БОЛЬШАЯ ПРОПУСКНАЯ СПОСОБНОСТЬ»
```

Некоторые заголовки имеют смысл только в запросах или только в ответах. Они называются заголовками запроса и заголовками ответа, соответственно. Если заголовок появляется в сообщении не своей категории (например, заголовок запроса в ответе), то он игнорируется.

При передаче сообщений протокола SIP, упакованных в сигнальные сообщения протокола UDP, существует вероятность того, что размер запроса или ответа превысит максимально допустимый для данной сети размер, что приведет к фрагментации пакета. Во избежание этого используется сжатый формат имен основных заголовков, подобно тому, как это делается в протоколе SDP. Ниже (табл. 3.1) приведен список таких заголовков.

Таблица 3.1. Сжатые имена заголовков

Сжатая форма имени	Полная форма имени
C	Content-Type
E	Content-Encoding
F	From
I	Call-ID
K	Supported
L	Content-Length
M	Contact (от «moved»)
S	Subject
O	Event
R	Refer-To
T	To
U	Allow-Events
V	Via

3.2.2. Заголовок Accept

Заголовок **Accept** сообщает, тела сообщений каких типов принимает клиент. Серверное приложение, которое может возвращать содержимое тела в разных форматах, должно проверить содержимое поля заголовка, чтобы принять решение, какой формат содержимого и, соответственно, тип тела сообщения следует использовать. Отсутствие значений в заголовке **Accept** информирует о том, что не поддерживаются никакие типы. Если в сообщении нет заголовка **Accept**, то сервер должен применить значение по умолчанию – *application/sdp.*

```
Accept:application/sdp;level=1, application/x-private, text/html
```

3.2.3. Заголовок Accept-Encoding

Заголовок **Accept-Encoding** похож на **Accept**, он сообщает о поддерживаемых типах кодирования содержимого в ответе. Наличие пустого заголовка в сообщении также допускается. Это равнозначно: **Accept-Encoding***: identity*, что значит: кодирование запрещено. Если заголовок отсутствует в сообщении, сервер устанавливает значение по умолчанию – *identity*. В этом заключается некоторое расхождение с протоколом HTTP, где в случае отсутствия заголовка может быть использован любой тип кодирования, но значение *identity* (отсутствие кодирования) предпочтительнее. Например:

```
Accept-Encoding:gzip
```

3.2.4. Заголовок Accept-Language

Заголовок **Accept-Language** используется в запросах, чтобы указать предпочтительные языки для ключевых фраз, описаний сеансов связи, оповещений о текущем состоянии, содержащихся в ответах в качестве тел сообщения. Если заголовок отсутствует, сервер считает, что клиент поддерживает все языки. Правило упорядочения языков в списке по предпочтительности базируется на параметре «q». Например:

```
Accept-Language:da, en-gb;q=0.8, en;q=0.7
```

3.2.5. Заголовок Alert-Info

Заголовок **Alert-Info**, присутствующий в запросе INVITE, предписывает использование альтернативного сигнала вызова для UAS. Когда заголовок

содержится в ответе с кодом 180 (Ringing), он устанавливает альтернативный сигнал КПВ для UAC.

Обычно этот заголовок вводится прокси-сервером для звукового сигнала вызова. Пользователь должен иметь возможность отключить эту функцию. Это помогает избежать возможных проблем в случае использования такого заголовка SIP-элементами, с которыми не были заранее достигнуты соответствующие договоренности относительно звуковых сигналов. Пример использования этого заголовка:

```
Alert-Info:http://www.niits.ru/sounds/moo.wav
```

3.2.6. Заголовок Allow

Заголовок **Allow** содержит список типов запросов, которые поддерживает агент пользователя, сформировавший сообщение. Все типы запросов, понимаемые UA, включая ACK и CANCEL, должны входить в этот список. Отсутствие заголовка **Allow** не означает, что передающий сообщение UA не поддерживает никаких типов запросов; это подразумевает, что агент пользователя отправителя не желает передавать информацию о том, какие типы запросов он поддерживает. Применение заголовка **Allow** в ответах на запросы (за исключением ответов на запрос OPTION) приводит к уменьшению числа передаваемых сообщений.

Пример:

```
Allow:INVITE, ACK, OPTIONS, CANCEL, BYE
```

3.2.7. Заголовок Allow-Events

Заголовок **Allow-Events**, если он присутствует, содержит список, состоящий из идентификаторов функциональных возможностей информировать клиента о событиях определенного типа – event package, – поддерживаемых клиентом (если передается в запросе) или сервером (если передается в ответе). Другими словами, оконечная точка, передающая заголовок **Allow-Events**, информирует, что она может обрабатывать запросы SUBSCRIBE и создавать запросы NOTIFY для всех типов событий, идентифицированных с помощью event package в заголовке.

Терминал, поддерживающий один или несколько event package, должен помещать соответствующий заголовок **Allow-Events**, указывающий все поддерживаемые типы событий, во все типы запросов, которые инициируют диалог, такие

как INVITE, и в ответы на них, а также в ответы на запрос OPTIONS. Заметим, что заголовок **Allow-Events** не должен вводиться прокси-серверами. Пример:

```
Allow-Events:refer
```

3.2.8. Заголовок Authentication-Info

Заголовок **Authentication-Info** используется для взаимной аутентификации с использованием системы аутентификации HTTP Digest. UAS может включить этот заголовок в ответ класса 2xx на запрос, который был успешно аутентифицирован, с использованием значения отклика, содержащегося в заголовке **Authorization**. Пример:

```
Authentication-Info:nextnonce=«47364c23432d2e131a5fb210812c»
```

3.2.9. Заголовок Authorization

Поле заголовка **Authorization** содержит отклик аутентификации агента пользователя. Этот заголовок вместе с заголовком **Proxy-Authorization** отступает от общих правил, касающихся множественности значений в поле заголовка и описанных в начале данного раздела. Заголовки с одинаковыми именами могут присутствовать в сообщении в любом количестве, но не могут объединяться в один общий заголовок, как это происходит с прочими заголовками. Более подробная информация о заголовке **Authorization** содержится в 6.4. Пример:

```
Authorization:Digest username=«Anton», realm=«niits.ru»,
Nonce=«84a4cc6f3082121f32b42a2187831a9e»,
Response=«7587245234b3434cc3412213e5f113a5432»
```

3.2.10. Заголовок Call-ID

Заголовок **Call-ID** – уникальный идентификатор сеанса связи или всех регистраций отдельного клиента. Значение идентификатору присваивает сторона, которая инициирует вызов. Возможна и такая ситуация: к одной мультимедийной конференции относятся несколько соединений – все они будут иметь разные идентификаторы **Call-ID**.

Заголовок состоит из буквенно-числового значения и имени рабочей станции, которая присвоила этот идентификатор. Между ними должен стоять символ «@». Значения **Call-ID** регистрозависимы и могут сравниваться побайтно.

Примеры:

```
Call-ID:f81d4fae-7dec-11d0-a765-00a0c91e6bf6@niits.ru
i:f81d4fae-7dec-11d0-a765-00a0c91e6bf6@192.0.2.4
```

3.2.11. Заголовок Call-Info

Заголовок **Call-Info** содержит дополнительную информацию о вызывающем или вызываемом пользователе в зависимости от того, где находится заголовок: в запросе или в ответе. Назначение URI, содержащегося в заголовке, описывается параметром «purpose». Значение этого параметра *icon* определяет изображение, предназначенное для визуального представления вызывающего или вызываемого пользователя. Значение *info* дает общее описание пользователя, например, с помощью web-страницы. Значение *card* определяет электронную визитную карточку, содержащую имя пользователя, организацию, номер телефона, адрес электронной почты и т.д., например, карту формата Vcard (vCard MIME Directory Profile [10] или LDIF (The LDAP Data Interchange Format [31].

Использование заголовка **Call-Info** может представлять угрозу безопасности пользователя. Если вызываемый пользователь воспользуется URI, приведенными злонамеренным пользователем, то он может увидеть неприятную или оскорбительную информацию, получить доступ к опасным или нелегальным ресурсам и т.д. Поэтому рекомендуется, чтобы UA представлял информацию в **Call-Info** только в том случае, если может удостоверить подлинность SIP-элемента, создавшего заголовок, и доверяет этому SIP-элементу. Этот элемент не обязательно должен быть UA; заголовок может быть помещен в запрос прокси-сервером. Пример:

```
Call-Info:<http://www.serv1.niits.ru/anton/photo.jpg>;purpose=icon,
http:/www.serv1.ru/anton/;purpose=info
```

3.2.12. Заголовок Contact

Поле заголовка **Contact** несет в себе URI, значение которого зависит от типа передаваемого запроса или ответа. Как правило, в заголовке **Contact** находится текущий адрес пользователя, на который он может принимать входящие сообщения. Заголовок **Contact** может содержать отображаемое имя (display name), адрес с его параметрами и параметры заголовка. Для заголовка **Contact** определены параметры «q» и «expires». Они используются только в случае, когда заголовок присутствует в запросе REGISTER, в ответе на него или в ответе класса 3xx.

Когда значение поля заголовка содержит отображаемое имя, URI со всеми его параметрами заключается в символы «<» и «>». В противном случае все параметры, следующие за URI, будут трактоваться как параметры заголовка. Отображаемым именем может быть комбинация символьных фраз, или строка, заключенная в кавычки, если существует необходимость в более длинной характеристике.

В случае, если URI содержит запятую, точку с запятой или вопросительный знак, он заключается в угловые скобки, даже если отображаемое имя отсутствует. Между отображаемым именем и «<» допускается наличие линейного пробела (LWS). Эти правила действительны также в отношении заголовков **To** и **From**.

Заголовок **Contact** выполняет роль, похожую на роль заголовка **Location** в HTTP. Однако заголовок протокола HTTP позволяет ввести только один адрес, не заключенный в кавычки. Поскольку URI могут содержать запятые и точки с запятой в качестве скрытых знаков, они могут быть приняты за разграничители значений заголовков или параметров, соответственно. Пример:

```
Contact:«Alexander» <sip:alexander@niits.ru>
;q=0.7; expires=3600,
«Alexander» <mailto:alexander@niits.ru>;q=0.1
```

3.2.13. Заголовок Content-Disposition

Заголовок **Content-Disposition** описывает, как будет интерпретироваться клиентом или сервером агента пользователя тело сообщения, или – для состоящих из нескольких частей тел сообщений типа multipart – часть тела сообщения. Этот SIP заголовок дополняет информацию о типе тела сообщения, содержащуюся в заголовке **Content-Type**.

В протоколе SIP определено несколько значений заголовка **Content-Disposition**. Значение *session* указывает, что часть тела описывает сеанс для вызова любой из двух сторон или для одностороннего проключения речевого тракта в предответном состоянии. Значение *render* означает, что часть тела сообщения должна быть отражена на дисплее или быть представлена пользователю другим образом.

Если заголовок **Content-Disposition** отсутствует, то сервер должен для тел сообщения типа application/sdp определять значение **Content-Disposition** – session, а для тел остальных типов – значение render.

Значение *icon* указывает на то, что часть тела сообщения содержит изображение, пригодное для визуального представления вызывающего или вызываемого пользователя; это изображение может быть использовано UA, чтобы визуально информировать пользователя о полученном сообщении; помимо этого, оно может удерживаться во время диалога. Значение *alert* означает, что часть тела содержит аудиозапись, которая должна быть предоставлена агентом пользователя для того, чтобы уведомить пользователя о получении запроса, как правило, – запроса, инициирующего диалог; например, это информационное тело может быть представлено в виде вызывного сигнала для телефонного вызова, после того как был передан предварительный ответ с кодом 180 (Ringing).

Любые MIME-тела со значением заголовка **Content-Disposition**, которое предписывает передачу содержимого пользователю, могут обрабатываться только в случае, если сообщение было надлежащим образом аутентифицировано.

Параметр «handling-param» описывает действия UAS при получении сообщения с непонятными значениями заголовков **Content-Disposition** и **Content-Type**. Для этого параметра существуют значения *optional* и *required*. Если значение параметра «handling-param» отсутствует, то по умолчанию должно подразумеваться значение *required*. В случае, если заголовок **Content-Disposition** отсутствует, MIME-тип обуславливает для **Content-Disposition** значение по умолчанию. В противном случае значение должно быть *render*. Пример:

```
Content-Disposition:session
```

3.2.14. Заголовок Content-Encoding

Заголовок **Content-Encoding** используется в качестве модификатора типов тела сообщения. Когда такой заголовок присутствует, его значение указывает, какие дополнительные виды кодирования были применены к содержимому и, соответственно, какие следует применить механизмы декодирования для получения тела сообщения типа, обозначенного в поле заголовка **Content-Type**.

В первую очередь, **Content-Encoding** предназначен для того, чтобы обеспечить компрессию тела сообщения с возможностью последующего восстановления. Если содержимое было закодировано несколько раз, кодеки содержимого должны быть перечислены в том порядке, в котором они применялись. Все значения поля заголовка регистронезависимы.

Клиент может закодировать содержимое тела в запросах. Сервер может закодировать содержимое тела в ответах. При этом UAS должен использовать только типы кодирования, которые были перечисленные в поле заголовка **Accept-Encoding** запроса. Пример:

```
Content-Encoding:gzip
```

3.2.15. Заголовок Content-Language

Основное назначение заголовка **Content-Language** – определить и изменить содержимое тела сообщения в соответствии с предпочтительным языком пользователя (имеется в виду национальный язык). Если тело сообщения содержит информацию на каком-то определенном языке, то это будет указано в заголовке. В случае отсутствия в сообщении заголовка **Content-Language** подразумевается, что содержимое предназначено для пользователей любых языковых групп.

Заголовок **Content-Language** может применяться не только к текстовым телам, но и к телам других типов. Пример:

```
Content-Language:fr
```

3.2.16. Заголовок Content-Length

Заголовок **Content-Length** указывает отображенный в десятичном виде размер (в байтах) тела сообщения, переданного получателю. Приложения должны помещать в это поле размер тела сообщения, подлежащего передаче, невзирая на тип тела сообщения. Если в качестве транспорта выступает потоко-ориентированный протокол (такой как TCP), заголовок **Content-Length** должен использоваться обязательно.

Размер тела сообщения не учитывает пустой строки, отделяющей заголовки от тела сообщения. Разрешенными значениями для **Content-Length** является любое число, большее или равное нулю. Когда тело в передаваемом сообщении отсутствует, поле заголовка **Content-Length** содержит ноль.

Возможность не включать заголовок **Content-Length** в сообщение упрощает создание cgi-подобных сценариев, которые динамически генерируют ответы. Пример:

```
Content-Length:349
```

3.2.17. Заголовок Content-Type

Заголовок **Content-Type** определяет тип тела сообщения, переданного получателю. **Content-Type** должен входить в сообщение, если тело сообщения не пустое. Если же тело пустое, а **Content-Type** присутствует, он показывает, что тело определенного типа имеет нулевую длину (например, пустой аудио-файл). Примеры:

```
Content-Type:application/sdp
c: text/html;charset=ISO-8859-4
```

3.2.18. Заголовок CSeq

Заголовок **CSeq** – уникальный идентификатор запроса, относящегося к одному соединению. Он служит для корреляции запроса с ответом на него, а также для того, чтобы первоначальные запросы отличались от переадресованных. Заголовок состоит из двух частей: натурального числа из диапазона от 1 до 2^{32} и типа запроса. Часть с типом запроса является регистрозависимой. Сервер должен проверять значение **CSeq** в каждом принимаемом запросе**,** и считает его новым, если значение больше предыдущего. Этот заголовок копируется из запроса в ответ. Пример:

```
CSeq:4711 INVITE
```

3.2.19. Заголовок Date

Заголовок **Date** содержит дату и время первой отправки сообщения. В отличие от HTTP/1.1, протокол SIP поддерживает новейший среди описанных в [23] формат даты, он переводит любой часовой пояс к стандарту GMT, хотя [23] не запрещает использовать другие форматы. Поле заголовка **Date** может быть использовано любой оконечной системой без встроенных часов с автономным батарейным питанием для получения информации о текущем времени. Однако это требует, чтобы клиенты знали отклонение своих часовых поясов от GMT. Пример:

```
Date:Sat, 13 Nov 2010 23:29:00 GMT
```

3.2.20. Заголовок Error-Info

Заголовок **Error-Info** является указателем на дополнительную информацию об ответе, содержащем код ошибки. Возможности пользовательского интерфейса клиента SIP простираются от всплывающих рабочих окон и аудио в программном клиенте на персональном компьютере до функции предоставления аудиоинформации для оконечных точек, соединенных через шлюзы. Вместо того чтобы принуждать сервер, генерирующий код ошибки, делать выбор между передачей сообщения, содержащего код ошибки и указание причины, и воспроизведением звукового сигнала, существует возможность предусмотреть в поле заголовка **Error-Info** оба варианта. Впоследствии клиент самостоятельно решает, какой тип индикации ошибки предоставить вызывающему пользователю.

UAC обрабатывает SIP или SIPS URI в поле заголовка **Error-Info** так же, как контактный адрес в заголовке **Contact** перенаправляющего сообщения, и может генерировать новое сообщение INVITE, направляемое автоинформатору. Автоинформатор, предоставляющий вызывающему пользователю записанное уведомление, может находиться по адресу, использующему схему адресации, которая отлична от «sip», например «tel». Примеры:

```
SIP/2.0 404 The number you have dialed is not in service
Error-Info:<sip:not-in-service-recording@niits.ru>
```

3.2.21. Заголовок Event

В заголовке должен быть указан ровно один тип события (представленный в виде идентификатора – event package), на которое производится подписка (subscription) или о котором передается уведомление (notification).

С целью обеспечить соответствие сообщений NOTIFY и SUBSCRIBE значение типа события («event-type») и параметр «id» (в случае его присутствия) сравниваются побайтно. Заголовок **Event**, содержащий параметр «id», никогда не будет соответствовать аналогичному заголовку без такого параметра. Остальные возможные параметры при сравнении не учитываются. Пример.

```
Event:refer; id=1234
```

3.2.22. Заголовок Expires

Заголовок **Expires** устанавливает время, по истечении которого сообщение или его содержимое станет недействительным. Значение заголовка зависит от типа запроса. Присутствует в запросах REGISTER и INVITE. В запросе REGISTER заголовок указывает, сколько времени регистрация остается действительной. В запросах INVITE он ограничивает время, в течение которого URI будет оставаться действительным на приемнике – оставаться в кэш-памяти. Если этот заголовок отсутствует, буферизация на сервере производится не будет. Значение этого поля – выраженное в десятичном виде количество секунд в интервале между 0 и $(2^{32} - 1)$, измеренное с момента получения запроса. Пример:

```
Expires:5
```

3.2.23. Заголовок From

Заголовок **From** содержит URI отправителя запроса, (заметим, что адрес отправителя запроса может не совпадать с адресом инициатора диалога). Адрес из заголовка **From** запроса копируется в одноименный заголовок ответа.

Отображаемое имя должно информировать пользователя об инициаторе запроса. В случае, если не удается установить личность вызывающего пользователя, в качестве *отображаемого имени (display name)* фигурирует «Anonymous». В случае, если URI содержит запятую, точку с запятой или вопросительный знак, он заключается в угловые скобки, даже если отображаемое имя отсутствует.

Два заголовка **From** считаются эквивалентными, когда совпадают их URI и параметры. При сравнении параметры расширения, присутствующие лишь в одном из двух заголовков, во внимание не принимаются. Это означает, что отображаемое имя и наличие или отсутствие угловых скобок на результат сравнения не влияют. Примеры:

```
From:«Vladimir» <sip:vladimir@protei.ru> ;tag=a48s
From:sip:+79213434329@gateway.protei.ru;tag=887s
```

3.2.24. Заголовок In-Reply-To

В поле заголовка **In-Reply-To** перечисляются уникальные идентификаторы сеансов связи **(Call-ID)** с данным пользователем, инициированных отправителями. Эти идентификаторы по мере их поступления буферизируются клиентом

и впоследствии включаются в поле заголовка **In-Reply-To** при ответном вызове. Это позволяет системам автоматического распределения вызовов маршрутизировать ответный вызов инициатору первого вызова. Это также дает возможность вызываемому пользователю отфильтровывать вызовы, т.е. принимать только ответные вызовы от пользователей, с которыми ранее проводились сеансы связи, инициированные им самим. Однако этот заголовок не является заменой аутентификации в запросах. Пример:

```
In-Reply-To:70710@lonis.ru, 17320@niits.ru
```

3.2.25. Заголовок Max-Forwards

Заголовок **Max-Forwards** используется в любом типе SIP-запросов, чтобы ограничить число серверов или шлюзов, через которые проходит запрос. Значение заголовка должно быть целым числом в пределах от 0 до 255, отражающим оставшееся количество пересылок, которое разрешено для сообщения. Это число уменьшается каждым сервером, который пересылает запрос дальше. В качестве первоначального значения рекомендуется брать 70.

Заголовок **Max-Forwards** должен вводиться теми элементами, которые иначе не могут гарантировать обнаружение петли. Пример:

```
Max-Forwards:6
```

3.2.26. Заголовок Min-Expires

Заголовок **Min-Expires** несет в себе минимальный период обновления, который подходит для управляемых сервером SIP-элементов с временным состоянием. В первую очередь, это относится к содержимому заголовков **Contact**, которое сохраняет регистрирующий сервер registrar. Поле заголовка содержит десятичное целое число секунд от 0 до $(2^{32} - 1)$. Заголовок используется в ответе 423 (Interval Too Brief). Пример:

```
Min-Expires:60
```

3.2.27. Заголовок MIME-Version

Заголовок **MIME-Version** указывает версию стандарта MIME-информации, помещенной в тело сообщения. Пример:

```
MIME-Version:1.0
```

3.2.28. Заголовок Organization

Заголовок **Organization** содержит название организации, к которой относится SIP-элемент, передающий запросы или ответы. Это поле заголовка может использоваться клиентским программным обеспечением для фильтрации вызовов. Пример:

```
Organization:Niits
```

3.2.29. Заголовок Path

Между UA и сервером регистрации, исходя из особенностей топологии сети, может находиться один или несколько прокси-серверов. Запрос REGISTER должен проходить через эти прокси-серверы. Однако REGISTER не обеспечивает механизма обнаружения и записи последовательности этих посредников для дальнейшего использования. Такой механизм предоставляет заголовок **Path**. Заголовок используется в запросах REGISTER и ответах класса 2xx на них.

Поле заголовка **Path** может быть добавлено в запрос REGISTER любым SIP-элементом, через который проходит запрос. Значения заголовка **Path** размещаются в строгой последовательности: по мере продвижения запроса каждое новое значение добавляется прокси-серверами сверху, смещая вниз все присутствующие значения. Кроме того, так же, как в заголовке **Route**, поля заголовка могут быть объединены. Сервер регистрации отображает в заголовке ответа на REGISTER все значения в обратном порядке, и указанные посредники доставляют ответ обратно агенту пользователя. В итоге UA получает информацию об узлах, лежащих на пути прохождения сообщений регистрации, и может в дальнейшем использовать ее для своих задач.

Path отличается от **Record-Route** тем, что он применяется в запросах REGISTER и ответах класса 2xx на них, которые передаются вне диалога, в то время как заголовок **Record-Route** используется только в режиме диалога. Более того, информация из **Record-Route** используется только в рамках существующего диалога, а информация из **Path** – для использования в будущих диалогах. При этом значения заголовка **Path** придерживаются синтаксиса, определенного для заголовка **Route**. Поскольку заголовок **Path** является расширением SIP, поддержка этого заголовка агентом пользователя может быть указана с помощью option-tag «path» в заголовке **Supported**. Пример:

```
Path:<sip:P3.niits.ru;lr>,<sip:P1.niits.ru;lr>
```

3.2.30. Заголовок Priority

Заголовок **Priority** указывает на срочность запроса с точки зрения клиента. В нем содержится приоритет SIP-запроса для конечного пользователя или его UA. Например, содержимое поля данного заголовка влияет на маршрутизацию вызова и на процесс приема его сервером. Сообщения, не содержащие заголовка **Priority**, расцениваются, как сообщения с приоритетом *normal*. Заголовок **Priority** никаким образом не воздействует на использование коммуникационных ресурсов, например, на приоритет при пересылке пакетов маршрутизаторами. Поле заголовка может содержать значения: *non-urgent* (несрочное), *normal* (нормальное), *urge* (срочное) и *emergency* (экстренное); могут быть определены и дополнительные значения. Рекомендуется, чтобы значение *emergency* использовались только в случае экстренных вызовов соответствующих служб. Примеры:

```
Subject:У нас случился пожар!
Priority:emergency
```

или

```
Subject:Планы на выходные
Priority:non-urgent
```

3.2.31. Заголовок Privacy

Существуют SIP-заголовки, содержимое которых агент пользователя не может самостоятельно скрыть от просмотра другими элементами сети, например, зашифровать, поскольку их использование необходимо для маршрутизации сообщений. Это могут сделать посредники, которые несут ответственность за передачу сообщений от и к пользователю, активизировавшему функцию анонимности. Агент пользователя должен иметь средства для того, чтобы запросить *услуги анонимности (privacy services)*. Для этой цели используется заголовок **Privacy**. Агент пользователя UA помещает в сообщение заголовок **Privacy**, когда существует необходимость воспользоваться услугами анонимности, предоставляемыми сетью.

В поле заголовка могут находиться значения: *header*, *session, user, none, critical*. Значение *header* означает, что пользователь запрашивает сокрытие тех заголовков (таких как **Via** и **Contact**), которые не могут быть полностью удалены из раздела идентифицирующей информации без вмешательства посредников.

Значение *session* означает, что пользователь запрашивает анонимность для сеанса (сеансов), связи инициированного этим сообщением, описание которого, например,

содержится в SDP-теле. Выполнение запроса приведет к тому, что IP-адрес, с которого будет поступать трафик в ходе сеанса, окажется замаскированным.

Значение *user* обычно присваивается теми прокси-серверами, у которых есть предварительная договоренность с пользователем о том, что функции обеспечения анонимности уровня пользователя должны быть выполнены сетью, предположительно потому, что агент пользователя не способен их выполнить. Сам агент пользователя может помещать значение *user* только в запросы REGISTER. Значение *none* указывает, что выполнение функций обеспечения анонимности не требуется. Значение *critical* сообщает, что функции обеспечения анонимности сообщения должны быть обязательно выполнены, в противном случае запрос должен быть отклонен.

Когда заголовок сформирован, он должен состоять или из значения *none,* или из одного или более значений *user, header* и *session*, каждое из которых может быть указано один раз; значение (значения) может сопровождаться индикатором *critical*.

3.2.32. Заголовок Proxy-Authenticate

Заголовок **Proxy-Authenticate** содержит запрос аутентификации, состоящий из поля, которое отображает схему аутентификации, и ряд параметров, необходимых для проведения процедуры аутентификации с данным прокси-сервером для указанного Request-URI. Заголовок включается в состав ответа с кодом 407 (Proxy Authentication Required) или с кодом 401 (Unauthorized). Более подробная информация о заголовке приводится в § 6.4. Пример:

```
Proxy-Authenticate:Digest realm=«niits.ru»,
Qop=«auth», nonce=«f84f1cec41e6cbe5aea9c8e88d359»,
Opaque=« », stale=FALSE, algorithm=MD5
```

3.2.33. Заголовок Proxy-Authorization

Заголовок **Proxy-Authorization** идентифицирует пользователя прокси-серверу, который требует аутентификации. Значение поля заголовка состоит из отклика аутентификации, который содержит информацию аутентификации агента пользователя для прокси-сервера, обслуживающего запрашиваемый ресурс.

Если запрос идентифицирован и ресурс специфицирован, та же самая идентификационная информация может быть использована для других запросов, направляемых в эту область.

Proxy-Authorization вместе с заголовком **Authorization** отступают от общих правил, касающихся множественности значений в поле заголовка. Заголовки с одинаковыми именами могут присутствовать в сообщении в любом количестве, но не могут объединяться в один общий заголовок, как это происходит с прочими заголовками. Более подробная информация о заголовке содержится в § 6.4.

Пример:

```
Proxy-Authorization: Digest username=«Anton», realm=«niits.ru»,
Nonce=«c60f3082ee1212b402a21831ae»,
response=«245f23415f11432b3434341c022»
```

3.2.34. Заголовок Proxy-Require

Заголовок **Proxy-Require** указывает функции прокси-сервера, которые должны им поддерживаться. Пример:

```
Proxy-Require:100rel
```

3.2.35. Заголовок P-Asserted-Identity

Заголовок **P-Asserted-Identity** используется в сообщениях между логическими элементами SIP, имеющими доверительные отношения (обычно между посредниками), для переноса информации, удостоверяющей пользователя (как правило, его списочного адреса). Значение заголовка состоит из URI и, опционально, отображаемого имени. URI может иметь схему sip, sips, или tel. Возможно также присутствие в заголовке двух значений; тогда одно из них должно быть со схемой sip или sips, а второе – со схемой tel. Использование данного заголовка целесообразно только в пределах *домена доверия (Trust Domain)*.

Прокси-сервер в домене доверия может получить сообщение от узла, с которым у него установлены доверительные отношения, и от узла, с которым такие отношения не установлены. Во втором случае, при условии, что прокси-сервер хочет добавить заголовок **P-Asserted-Identity**, он должен произвести процедуру аутентификации инициатора сообщения и после этого поместить полученную информацию, идентифицирующую пользователя, в поле заголовка **P-Asserted-Identity** сообщения. Если прокси-сервер получает сообщение от узла, которому доверяет,

он может использовать информацию в заголовке **P-Asserted-Identity**, как если бы она аутентифицировала пользователя.

Прокси-сервер, передающий сообщение прокси-серверу или UA, с которым у него не установлены доверительные отношения, должен удалить все значения заголовка **P-Asserted-Identity**, если вызывающий пользователь запросил обеспечение секретности своей информации. Пример:

```
P-Asserted-Identity: «Anton» <sip:anton@niits.ru>
P-Asserted-Identity: tel:+79213434329
```

3.2.36. Заголовок P-Preferred-Identity

Заголовок **P-Preferred-Identity** используется в сообщениях, передаваемых агентом пользователя прокси-серверу, с которым у него установлены доверительные отношения. Заголовок переносит информацию, удостоверяющую пользователя и используемую по желанию инициатора сообщения в качестве значения заголовка **P-Asserted-Identity**, которое пользующийся доверием элемент введет в сообщение. Заголовок **P-Preferred-Identity** может содержать одно значение и иметь схему sip, sips, или tel. Возможно также присутствие в заголовке двух значений; тогда одно из них должно быть со схемой sip или sips, а второе – со схемой tel. Пример:

```
P-Preferred-Identity:«Anton» <sip:anton@niits.ru>
```

3.2.37. Заголовок P-Media-Authorization

Заголовок **P-Media-Authorization** предназначен для контроля доступа к услуге обеспечения гарантированного качества обслуживания (QoS) средствами SIP. Использование этого заголовка способствует предотвращению ситуаций *отказа в обслуживании (denial-of-service)*. Конечно, для такой цели могло быть использовано тело сообщения, но при этом отсутствовала бы возможность сквозного шифрования тела. Заголовок **P-Media-Authorization** включается прокси-сервером, контролирующим доступ к услуге обеспечения гарантированного QoS, во все предварительные ответы (кроме ответа с кодом 100), в первый надежный ответ 1xx или 2xx и во все повторения этого надежного ответа для UAC, или в запросы INVITE, ACK, UPDATE и PRACK – для UAS. Заголовок **P-Media-Authorization** содержит один или несколько идентификаторов для предоставления доступа к услуге гарантированного QoS.

UA использует все идентификаторы из последнего полученного запроса/ответа от прокси-сервера, предоставляющего доступ к услуге гарантированного QoS, при запросе резервирования ресурсов для медиапотоков, используемых в ходе сеанса связи (в случае использования протокола RSVP совместно с SIP идентификаторы, полученные в заголовке **P-Media-Authorization**, используются в сообщении PATH). Эти идентификаторы используются для санкционирования применения услуги обеспечения гарантированного QoS для медиапотока (потоков). Заголовок **P-Media-Authorization** может быть использован только в SIP-запросе или ответе, который допускает наличие offer или answer в теле сообщения (SDP-предложение или SDP-ответ с описанием сеанса связи).

3.2.38. Заголовок P-Associated-URI

Этот заголовок, являясь расширением SIP, позволяет регистрирующему серверу передавать агенту пользователя список существующих контактных адресов для определенного зарегистрированного списочного адреса. Заголовок **P-Associated-URI** содержится в ответе с кодом 200 (OK) на запрос REGISTER и содержит список контактных адресов. Если registrar поддерживает это расширение, он должен всегда вводить заголовок **P-Associated-URI** в ответ 200 (OK) на REGISTER, независимо от того, какую процедуру выполнял UA пользователя – регистрацию, изменение регистрации или удаление регистрации, – и независимо от числа контактных адресов, зарегистрированных для списочного адреса.

3.2.39. Заголовок P-Called-Party-ID

Заголовок **P-Called-Party-ID** вводит прокси-сервер, как правило, – в запрос INVITE. В заголовок заносится значение поля Request-URI из полученного прокси-сервером запроса (это происходит до замены начального Request-URI контактным адресом). В Request-URI, как известно, содержится списочный адрес вызываемого пользователя. Поэтому, изучая содержимое заголовка **P-Called-Party-ID**, UAS определяет, на какой списочный адрес был передан запрос INVITE (например, пользователь может одновременно использовать личный и рабочий списочные адреса для приема запросов-приглашений к сеансу связи INVITE. Сервер может использовать эту информацию для подачи пользователю разных аудиовизуальных вызывающих сигналов, в зависимости от того, на какой из списочных адресов пользователя пришел вызов. Пример:

```
P-Called-Party-ID:sip:anton-work@niits.ru
```

3.2.40. Заголовок P-Visited-Network-ID

Принципы, определенные в 3GPP (3rd Generation Partnership Progect), предусматривают так называемые домашние сети (home network) и сети временного пребывания (visited network). Каждая домашняя сеть должна иметь роуминговые соглашения (roaming agreements) с одной или несколькими сетями временного пребывания. Наличие таких соглашений позволяет мобильному терминалу, покинувшему домашнюю сеть, использовать ресурсы, предоставляемые сетью временного пребывания в прозрачном режиме. Одно из условий приема домашней сетью сообщения регистрации от UA, переместившегося в одну из сетей временного пребывания, это – наличие роуминговых соглашений между домашней сетью и сетью, в которой пребывает UA. Чтобы проверить это условие, необходимо передать домашней сети информацию о сети временного пребывания. При наличии соглашений с этой сетью временного пребывания домашняя сеть предоставит «блуждающему» UA требуемые услуги.

Заголовок **P-Visited-Network-ID** используется для доставки к серверу регистрации или к домашнему прокси-серверу (home proxy) идентификатора сети, где временно находится UA; заголовок включает в запрос прокси-сервер сети временного пребывания. Этот идентификатор должен быть известен и серверу регистрации, и прокси-серверу домашнего домена, и прокси-серверам в сети временного пребывания UA. Как правило, заголовок **P-Visited-Network-ID** используется в запросе REGISTER, хотя может применяться и в других запросах.

Для того чтобы предотвратить конфликты, связанные с дублированием идентификаторов, значение заголовка **P-Visited-Network-ID** должно выбираться с осторожностью. Идентификатор должен быть уникален. Например:

```
P-Visited-Network-ID:«Visited network number 1»
```

3.2.41. Заголовок P-Access-Network-Info

Когда UA создает SIP-запрос или ответ и знает, что он будет надежно (с подтверждением) передан прокси-серверу, предоставляющему услуги данному UA, он вводит в сообщение заголовок **P-Access-Network-Info**. Этот заголовок содержит информацию о сети, которую использует UA для IP-взаимодействия. Как правило, заголовок игнорируется прокси-серверами, находящимися между UA и прокси-сервером, предоставляющим услуги. Прокси-сервер, предоставляющий услуги, может проверить заголовок **P-Access-Network-Info** и использовать

информацию, содержащуюся там, для предоставления услуг определенного типа в зависимости от значения заголовка. Некоторые услуги более или, наоборот, менее подходят пользователю в зависимости от типа доступа; кроме того, некоторые услуги могут быть предоставлены с лучшим качеством, когда прокси-сервер владеет детальной информацией о сети доступа (поскольку они могут быть адаптированы к конкретному пользователю). Перед тем как переслать запрос дальше, этот прокси-сервер удаляет из сообщения заголовок **P-Access-Network-Info**. Агент пользователя, предоставляющий информацию, должен доверять прокси-серверу, который предоставляет услуги, – только в этом случае гарантируется, что прокси-сервер удалит заголовок перед пересылкой запроса за пределы домена, в котором он сам находится, и, тем самым, не нарушит секретности переданной информации. Такой прокси-сервер обычно находится в домашней сети.

3.2.42. Заголовок P-Charging-Function-Addresses

В процесс предоставления доступа и услуг вовлечено множество элементов распределенной архитектуры сети. Существует необходимость передать каждому прокси-серверу SIP, вовлеченному в транзакцию, информацию об адресе элементов SIP-сети, решающих задачи начисления платы. Это требуется для передачи им записей, генерируемых прокси-серверами.

В архитектуре 3GPP определено два типа функциональных элементов, занимающихся начислением платы: *Charging Collection Function (CCF)* и *Event Charging Function (ECF)*. CCF используется при кредитной системе расчетов (postpaid), ECF используется при авансовой системе (prepaid).

Прокси-сервер SIP, который получает SIP-запрос, может поместить в него заголовок **P-Charging-Function-Addresses** перед тем как переслать запрос дальше; это возможно, если в запросе не было заголовка с таким названием. Заголовок состоит из списка адресов одного или нескольких элементов сети, ведущих начисление платы CCF или ECF, куда прокси-сервер должен передать информацию, касающуюся начисления платы. Прокси-сервер, получивший сообщение с таким заголовком, тоже будет знать, по какому адресу передавать записи, содержащие информацию для начисления платы. Агенты пользователя не должны работать с данным заголовком. Пример:

```
P-Charging-Function-Addresses:ccf=192.1.1.1; ccf=192.1.1.2;
ecf=192.1.1.3; ecf=192.1.1.4
```

3.2.43. Заголовок P-Charging-Vector

Операторы должны иметь возможность начислять плату за предоставление услуг. Это требует координации действий сетевых элементов, таких как прокси-серверы, что выражается в корреляции записей для начисления платы, созданных разными элементами сети SIP, но относящихся к одному сеансу связи. Корреляционная информация включает в себя уникальный глобальный идентификатор (вектор) начисления платы, что значительно упрощает билинговые функции.

Идентификатор начисления платы определен как совокупность информации о начислении платы. Информация может быть помещена в идентификатор несколькими сетевыми элементами (включая прокси-серверы SIP) и ими же может быть удалена. В идентификаторе содержится три типа корреляционной информации: значение IMS Charging Identity (ICID), адрес прокси-сервера, который создал это значение ICID, и Inter Operator Identifiers (IOI). ICID – значение, идентифицирующее диалог или транзакцию вне диалога в целях начисления платы. Оно используется для корреляции записей для начисления платы. ICID должен быть уникальным для данной сети. Для этого он должен включать в себя два компонента: локальное уникальное значение и имя домена или IP-адрес прокси-сервера SIP, создавшего локальное значение. IOI идентифицирует обе сети, в которых находятся участники диалога или транзакции вне диалога.

Для переноса идентификатора определен заголовок **P-Charging-Vector**. В случае отсутствия заголовка **P-Charging-Vector** прокси-сервер SIP может поместить его самостоятельно в начальный запрос или ответ для диалога с теми параметрами, которые ему доступны. Прокси-сервер SIP, который получает запрос, содержащий заголовок **P-Charging-Vector,** может использовать значение ICID при создании новых записей для начисления платы. Заголовок используется либо в пределах частного административного домена, либо между доменами, имеющими друг с другом доверительные отношения. Пример:

```
P-Charging-Vector:icid-value=1234bc9876e;
icid-generated-at=192.0.6.8;
orig-ioi=home1.protei.ru
```

3.2.44. Заголовок P-DCS-Trace-Party-ID

В сети, предоставляющей услуги передачи речевой информации, может быть организована *услуга трассировки пришедшего запроса (Customer originated trace)*, которая дает вызываемой стороне возможность обратиться к правоохрани-

тельным органам в случае злонамеренного телефонного вызова. В SIP эта услуга может обеспечиваться независимо от идентификатора вызывающего пользователя и работать даже тогда, когда пользователь запрашивает услугу анонимности. Для возможности инициировать услугу трассировки с целью определения информации, идентифицирующей пользователя, который не пользуется доверием UAC, в запрос INVITE помещается дополнительный заголовок **P-DCS-Trace-Party-ID**. Этот заголовок в других запросах и ответах не присутствует.

Элемент, идентифицирванный значением поля Request-URI, выполняет определенные поставщиком услуг функции записи и передачи правоохранительным органам информации, удостоверяющей вызывающего пользователя, которая находится в заголовке **P-DCS-Trace-Party-ID. UAC**, с которым установлены доверительные отношения, заголовок **P-DCS-Trace-Party-ID** не использует.

3.2.45. Заголовок P-DCS-OSPS

Некоторые вызовы предъявляют к их обработке особые требования, которые не могут быть удовлетворены обычными агентами пользователя. Например, когда пользователь занят в сеансе связи, а в это время приходит другой вызов, то такой вызов может быть отклонен с указанием занятости. Однако некоторые услуги, предоставляемые операторами ТфОП, требуют особой обработки вызовов. В их числе услуги «Проверка занятости линии» (BLV) и «Вмешательство в разговор в случае необходимости» (EI), инициируемые оператором ТфОП с *операторской консоли OSPS (Operator Services Position System)*. Для того чтобы информировать SIP-агента пользователя о том, что входящему вызову должно быть придано особое значение, используется заголовок **P-DCS-OSPS**. Его значение указывает на то, что запрашивается определенный способ обработки вызова. На сегодня для этого заголовка определены три значения: *BLV* (busy line verification), *EI* (emergency interrupt) и *RING* (operator ringback). Если при получении запроса INVITE, содержавшего в заголовке **P-DCS-OSPS** значение *BLV* или *EI*, пользователь решил принять вызов, UAS передает UAC ответ с кодом, указывающим на незанятость абонента. Так как значения *EI* и *RING* передаются только в установленных диалогах, они могут появляться в запросах UPDATE. Заголовок **P-DCS-OSPS** не может быть передан в запросе от UAC, не пользующегося доверием. Как правило, заголовок вводится в сообщение контроллером медиа-шлюза (Media Gateway Controller). Пример:

```
P-DCS-OSPS:BLV
```

3.2.46. Заголовок P-DCS-BILLING-INFO

В процессе предоставления услуг требуется производить сбор биллинговой информации и ее доставку биллинговой системе. Эту функцию может выполнить прокси-сервер SIP . Более того, задействованные в обслуживании вызова прокси-серверы, между которыми существуют доверительные отношения, могут обмениваться биллинговой информацией, относящейся к участникам сеанса связи. Для записей об изменении *состояния баланса (accounting records)* необходимо иметь идентификатор, который связывает все записи об услугах, предоставленных в процессе конкретного сеанса. Заголовок **P-DCS-BILLING-INFO** содержит идентификатор, который может быть использован устройством, генерирующим учетные записи, чтобы ассоциировать записи разного назначения и, возможно, от разных источников, с информацией, касающейся снятия платы со счета. Заголовок содержит также информацию о состоянии счета подписчика и другую информацию, необходимую для правильного биллинга. Этот заголовок используется только в сообщениях между прокси-серверами и пользующимися доверием UA, применим только в сегменте сети с доверительными отношениями и должен быть удален при выходе за пределы этого сегмента.

3.2.47. Заголовки P-DCS-LAES и P-DCS-REDIRECT

Заголовок **P-DCS-LAES** содержит информацию, необходимую для реализации функций оперативно-розыскных мероприятий *СОРМ*, именуемых в спецификациях SIP *легальным электронным наблюдением (Lawfully Authorized Electronic Surveillance).* Информация представляет собой адрес и порт устройства, занимающегося электронным наблюдением, для доставки двустороннего потока сигнальных сообщений, относящихся к обслуживаемому вызову. Заголовок может также включать в себя дополнительный адрес и порт для доставки элементу сети SIP, занимающемуся наблюдением, медиа-информации, которая передается в ходе сеанса. Этот заголовок используется только между прокси-серверами и агентами пользователя, имеющими доверительные отношения.

Заголовок **P-DCS-Redirect** содержит идентифицирующую сеанс информацию, необходимую для поддержки требований легального электронного наблюдения за перенаправленными вызовами. Он содержит первоначально указанный адрес, адрес перенаправленного запроса и количество предпринятых перенаправлений. Этот заголовок тоже используется только между прокси-серверами и агентами пользователя, имеющими доверительные отношения.

3.2.48. Заголовок RAck

Заголовок **RAck** помещается в запрос PRACK для обеспечения надежной доставки предварительных ответов. Он включает в себя два численных значения и поле типа запроса. Первое значение – это значение, взятое из поля заголовка **RSeq** подтверждаемого предварительного ответа. Следующее значение и тип запроса копируются из поля заголовка **CSeq** подтверждаемого ответа. Поле типа запроса в заголовке **RAck** чувствительно к смене регистра. Пример:

```
RAck:776656 1 180
```

3.2.49. Заголовок Reason

Один и тот же SIP-запрос может быть передан по разным причинам. Например, запрос CANCEL может передаваться прокси-сервером, размножающим запросы, если вызов обслуживался на другой ветви или прерван клиентом агента пользователя до ответа вызываемой стороны. Несмотря на то, что протокол и поведение системы одинаковы в обоих случаях, смысл передаваемого сообщения может быть существенно разным. В тех реализациях, где не предусмотрено расширение, связанное с заголовком **Reason**, при получении CANCEL по второй причине вызов может быть расценен как упущенный; в то же время, этот запрос будет расценен так же, если вызов принят на другом терминале пользователя.

Для создания услуг часто необходимо знать причину передачи SIP-запроса. Такую информацию обеспечивает заголовок **Reason**. Заголовок **Reason** предназначен также для инкапсуляции кода окончательного ответа в предварительный ответ. Эта функция способствует решению проблемы, обозначаемой аббревиатурой HERFP (Heterogeneous Error Response Forking Problem), – ситуации, когда элемент, размножающий запросы, не может отправить окончательный отклоняющий ответ, если не получил отклоняющий ответ с каждого из направлений, по которым он отправил запрос.

Заголовок **Reason** может содержаться в любом запросе, передаваемом в режиме диалога, в любом запросе CANCEL и в любом ответе, код которого разрешает присутствие этого заголовка. Значение поля заголовка может быть *SIP* с параметром «cause», указывающим код SIP-ответа. Значение может быть также – «*Q.850*» с параметром «cause», информирующем о коде причины, по которой было передано сообщение, в десятичном представлении в соответствии с рекомендацией ITU-T Q.850 для систем сигнализации DSS1 и ОКС7 (подсистема

ISUP). SIP-сообщение может содержать более одного значения поля заголовка **Reason**, однако в этом случае значения должны быть различны (например, одно *SIP*, а другое *Q.850*). Программные реализации SIP должны игнорировать значения заголовка **Reason**, которые им непонятны.

Клиенты и серверы могут игнорировать этот заголовок, что не повлияет на процесс обработки. Пример:

```
Reason:SIP ;cause=200 ;text=«Call completed elsewhere»
Reason:Q.850 ;cause=16 ;text=«Terminated»
```

3.2.50. Заголовок Record-Route

Заголовок **Record-Route** помещается прокси-серверами в запросы для того, чтобы следующие запросы в процессе диалога маршрутизировались через эти же прокси-серверы. Пример:

```
Record-Route:<sip:serv10.protei.ru;lr>,
             <sip:site3.niits.ru;lr>
```

3.2.51. Заголовок Refer-To

Заголовок **Refer-To** применяется только в запросе REFER и указывает URI для *переадресации вызова (call transfer)*. Другими словами, значение, содержащееся в заголовке, указывает адрес третьей стороны, с которой отправитель предлагает связаться получателю запроса REFER. В заголовке может содержаться только одно значение. Вместо **Refer-To** не может быть использован заголовок **Contact**, поскольку последний является частью механизма **Route**/**Record-Route**, описанного в § 5.2, и не может указывать адрес места назначения для переадресации вызова. Пример:

```
Refer-To:sip:alexander@niits.ru
```

3.2.52. Заголовок Reply-To

Заголовок **Reply-To** содержит логически обратный URI, который может отличаться от адреса в поле заголовка **From**. Например, такой URI может использоваться для ответных действий пользователя при пропущенных вызовах и вызовах, не завершившихся установлением связи. В случае, если пользователь желает оставаться анонимным, заголовок **Reply-To** должен быть либо исключен из запроса, либо помещен таким образом, чтобы не разглашать сведения о пользователе.

Если URI содержит запятую, точку с запятой или вопросительный знак, он заключается в угловые скобки, даже если отображаемое имя отсутствует. Пример:

```
Reply-To:Vladimir <sip:vladimir@protei.ru>
```

3.2.53. Заголовок Require

Поле заголовка **Require** используется UAC для того, чтобы сообщить UAS перечень опций, которые должен поддерживать сервер для обработки запроса. Содержимое поля заголовка представляет собой список идентификаторов новых функциональных возможностей (расширений) option-tag; каждый них определяет одно из SIP-расширений, необходимых для обработки сообщения. Пример:

```
Require:100rel
```

3.2.54. Заголовок Retry-After

Заголовок **Retry-After** применяется в ответах с кодом 500 (Server Internal Error) или 503 (Service Unavailable), чтобы информировать о том, на какой срок приостановлено обслуживание для вызывающего пользователя, или в ответах 404 (Not Found), 413 (Request Entity Too Large), 480 (Temporarily Unavailable), 486 (Busy Here), 600 (Busy), 603 (Decline), чтобы указать, когда вызываемый пользователь будет снова доступен. Значения поля заголовка – любое положительное целое число секунд (в десятичном виде).

Параметр «duration» показывает интервал времени, в течение которого абонент сможет принять вызов после того, как освободится. Если этот параметр отсутствует, подразумевается, что обслуживание будет предоставлено без ограничения времени. Чтобы указать предполагаемое время обратного вызова, может быть использован опциональный комментарий. Примеры:

```
Retry-After:18000;duration=3600
Retry-After:120 (I'm in a meeting)
```

3.2.55. Заголовок Route

Заголовок **Route** служит для принудительной маршрутизации запроса в соответствии со списком прокси-серверов. Процедуры, связанные с заголовком **Route**, подробно рассмотрены в § 5.2. Примеры:

```
Route:<sip:site5.niits.ru;lr>,
      <sip:serv3.protei.ru;lr>
```

3.2.56. Заголовок RSeq

Заголовок **RSeq** используется в предварительных ответах с надежной транспортировкой. Он содержит численное значение в интервале от 1 до (2^{32}-1). Каждый надежно передаваемый предварительный ответ (информационный ответ с подтверждением) содержит **RSeq** с порядковым номером этого ответа. Значение заголовка **RSeq** в каждом следующем надежном предварительном ответе будет на единицу больше. Пример:

```
RSeq: 988789
```

3.2.57. Заголовки Security-Client, Security-Server, Security-Verify

Заголовки могут быть использованы для того, чтобы реализовать механизмы обеспечения безопасности между UAC и другими логическими элементами SIP, включая UAS, прокси-сервер и сервер регистрации, которые находятся на расстоянии одной пересылки от них. Эти новые возможности дополняют существующие методы выбора механизмов обеспечения безопасности между логическими элементами SIP. Клиент, желающий использовать данный механизм, добавляет заголовок **Security-Client** в запрос, адресованный прокси-серверу, который находится на расстоянии одной пересылки. Заголовок будет содержать список всех механизмов безопасности, поддерживаемых клиентом. Сервер должен сформировать ответ с кодом 494 (Security Agreement Required) и добавить в него заголовок **Security-Server** со списком механизмов безопасности, которые он поддерживает. Список сервера не зависит от содержимого списка клиента. Когда клиент получает ответ сервера, он выбирает механизм безопасности с наибольшим значением параметра «q» из тех, которые известны клиенту. Затем клиент применяет соответствующий механизм обеспечения безопасности. Все последующие SIP-запросы, передаваемые клиентом этому серверу, должны использовать выбранный механизм безопасности. Эти запросы должны содержать заголовок **Security-Verify**, который отражает список механизмов безопасности сервера, полученный ранее в заголовке **Security-Server**. Сервер должен проверять, соответствуют ли механизмы безопасности, перечисленные в **Security-Verify** входящих запросов, поддерживаемым механизмам безопасности, указанным в статическом списке сервера. На сегодня определены следующие значения для заголовков: *digest*, *tls*, *ipsec-ike*, *ipsec-man*. Примеры:

```
Security-Client: digest
Security-Server: tls;q=0.2
Security-Verify: ipsec-ike; q=0.1
```

3.2.58. Заголовок Server

Поле заголовка **Server** содержит информацию о программном обеспечении, которое используется сервером для обработки запросов. Но раскрытие конкретной версии программного обеспечения сервера может облегчить атаки на программные продукты, уязвимые места которых известны, поэтому разработчикам серверов рекомендуется, чтобы включение этого поля в сообщение было конфигурируемой опцией. Пример:

```
Server:HomeServer v2
```

3.2.59. Заголовок Service-Route

Заголовок **Service-Route**, являющийся расширением протокола SIP, используется совместно с ответами на запрос REGISTER для того, чтобы обеспечить механизм, с помощью которого регистрирующий сервер может информировать агента пользователя о так называемом маршруте service route, т.е. о маршруте к серверам, предоставляющим услуги. Впоследствии UA может использовать эту информацию при исходящем вызове для того, чтобы запросить предоставление определенных услуг доменом, где находится используемый сервер регистрации.

Заголовок **Service-Route** направляет запросы через определенную последовательность прокси-серверов. При использовании этого маршрута UA сможет воспользоваться услугами, предоставляемыми прокси-серверами, которые связаны с сервером регистрации. Значения заголовка **Service-Route** должны соответствовать синтаксису, определенному для заголовка **Route**. В частности, значения должны содержать параметр «lr».

В приведенном ниже примере представлена домашняя сеть, состоящая из регистрирующего сервера (R), прокси-сервера, предоставляющего услуги HSP (home service proxy), базы данных (DBMS) и пограничного прокси-сервера поставщика услуг, маршрутизирующего сообщения (P2). UA1 через исходящий прокси-сервер P1 передает сообщение REGISTER регистрирующему серверу; тот помещает в ответ заголовок **Service-Route**, указывающий, что UA1 при исходящем вызове сможет воспользоваться предоставляемыми данным доменом услугами, если направит запрос, инициирующий соединение, через P2 и HSP, т.е., например, при передаче INVITE пользователю UA2 скопирует содержимое **Service-Route** в заголовок **Route**. Пример:

```
Service-Route:<sip:p2.home.protei.ru;lr>,
<sip:hsp.home.protei.ru;lr>
```

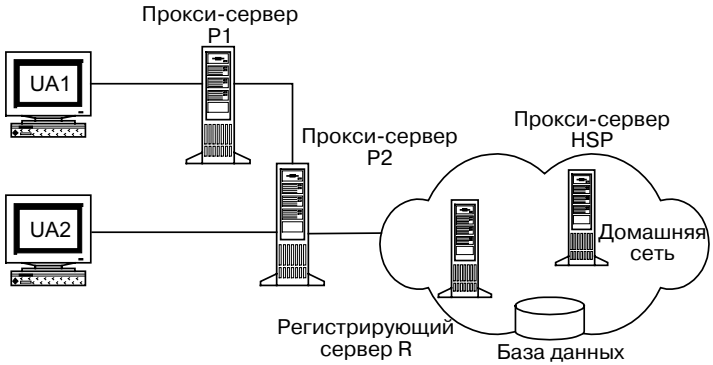


Рис. 3.2. Пример сети, использующей заголовок Service-Route

3.2.60. Заголовок Subject

Заголовок **Subject** содержит дополнительную информацию о типе и характере сеанса связи, позволяя производить фильтрацию вызовов без анализа описания сеанса. Примеры:

```
Subject: Требуется техническое описание оборудования
     s: Техническая поддержка
```

3.2.61. Заголовок Subscription-State

Заголовок **Subscription-State** содержится в сообщении NOTIFY и указывает статус подписки. Значение заголовка может быть *active*, *pending* или *terminated*. Значение *active* указывает, что подписка была принята и авторизована. Значение *pending* означает, что подписка была получена, но не может быть принята или отклонена из-за недостатка информации. Значение *terminated* сообщает, что подписка окончена. Когда значение заголовка – *active* или *pending*, заголовок включает в себя также параметр «expires», указывающий действительную продолжительность подписки; при этом подписчик должен скорректировать время действия подписки в соответствии с полученным значением. Когда значение заголовка – *terminated*, в нем присутствуют также параметры «reason» и «retry-after», информирующие о причине окончания подписки и о времени, когда подписчик может попытаться создать новую подписку. Пример:

```
Subscription-State: terminated;reason=noresource
```

3.2.62. Заголовок Supported

Заголовок **Supported** содержит перечень всех расширений, поддерживаемых UAC или UAS. Содержимое поля заголовка представляет собой список идентификаторов option-tag, которые понимает UAC или UAS. Если поле заголовка пусто, значит ни одно расширение не поддерживается. Рекомендуется, чтобы этот заголовок присутствовал во всех запросах и ответах, за исключением ACK. Пример:

```
Supported:sip-cc, sip-cc-02, timer
```

Приведенный заголовок указывает на поддержку возможностей управления сеансом связи по протоколу SIP и таймера сеанса.

3.2.63. Заголовок Timestamp

Заголовок **Timestamp** указывает, когда UAC передал сообщение к UAS. Заголовок позволяет расширениям или приложениям SIP получать оценку RTT (времени двойного пробега). Пример:

```
Timestamp:54
```

3.2.64. Заголовок To

Заголовок **To** определяет логического адресата запроса. В случае присутствия отображаемого имени, оно должно быть предоставлено вызываемому пользователю с помощью пользовательского интерфейса. Параметр «tag» является неотъемлемой частью механизма идентификации диалога.

Процедура сравнения заголовков **To** аналогична процедуре с заголовками **From**. Примеры:

```
To:The Operator <sip:operator@server10.protei.ru>;tag=287447
t:sip:+79213434329@gateway.protei.ru
```

3.2.65. Заголовок Unsupported

Заголовок **Unsupported** содержит перечень функциональных возможностей, не поддерживаемых UAS. Добавляется в ответ с кодом 420 (Bad Extension). Пример:

```
Unsupported:100rel
```

3.2.66. Заголовок User-Agent

Поле заголовка **User-Agent** несет информацию о клиенте агента пользователя, инициирующего запрос. Раскрытие конкретной версии программного обеспечения UA может облегчить атаки на программные продукты, уязвимые места которых известны. Разработчикам UA рекомендуется, чтобы включение этого поля в сообщение было настраиваемой опцией. Пример:

```
User-Agent:Softphone Beta1.5
```

3.2.67. Заголовок Via

Поле заголовка **Via** содержит список элементов сети SIP, через которые запрос на данный момент прошел. Список нужен для того, чтобы избежать ситуаций, в которых запрос пойдет по замкнутому пути, а также для тех случаев, когда необходимо, чтобы запросы и ответы обязательно проходили по одному и тому же пути. В конечном результате заголовок отображает весь путь, пройденный запросом: каждый прокси-сервер добавляет поле со своим адресом. Параметр «branch» в поле заголовка **Via** выполняет функцию идентификатора транзакции и используется прокси-серверами для обнаружения петель. **Via** содержит информацию о транспортном протоколе, посредством которого переносится сообщение, имя клиентского хоста или сетевой адрес и, возможно, номер порта, который является предпочтительным для приема ответов. В поле заголовка могут также присутствовать параметры: «maddr», «ttl», «received» и «branch». Возможно использование следующих транспортных протоколов: UDP, TCP, TLS и SCTP. Когда запрос передается на SIPS URI, указывается прикладной протокол – SIP, а транспортный протокол – TLS.

В следующем примере:

```
Via:SIP/2.0/UDP serv1.niits.ru:5060;branch=z9hG4bK87asdks7
Via:SIP/2.0/UDP 192.0.2.1:5060 ;received=192.0.2.207
;branch=z9hG4bK77asjd
```

сообщение сформировано на терминале (multi-homed host) с двумя сетевыми адресами 192.0.2.1 и 192.0.2.207. Отправитель не был уверен, какой сетевой интерфейс необходимо использовать, и указал первый. Получив сообщение, узел serv1.niits.ru обнаружил несоответствие и добавил параметр к значению заголовка **Via**, относящегося к предыдущей пересылке. Параметр содержит действительный адрес, с которого поступил пакет. К сетевому или хост-адресу и номеру порта не предъявляется требований подчинения синтаксису SIP URI, а именно, пробел не запрещается по обе стороны «:» или «/», как показано ниже:

```
Via:SIP /2.0/UDP serv3.niits.ru: 4000;ttl=16
;maddr=224.2.0.1;branch=z9hG4bKa7c6a8dlze.1
```

Два заголовка **Via** считаются равными, если их значения: название прикладного протокола, версия протокола, используемый транспортный протокол, адрес и порт узла (в указанном выше примере это – SIP/2.0/UDP serv3.niits.ru:4000) одинаковы, один и тот же набор параметров и значения одноименных параметров одинаковы.

3.2.68. Заголовок Warning

Заголовок **Warning** содержит дополнительную информацию, как правило, связанную с проблемами обработки запроса сервером. Значения поля заголовка **Warning** отправляются вместе с ответами и содержат трехзначный код, имя хоста и предупреждающий текст. Предупреждающий текст должен быть написан на национальном языке, наиболее легком для понимания конечным пользователем. Это решение может базироваться на любой доступной информации, такой как местоположение пользователя, поле заголовка A**ccept-Language** запроса или поле заголовка **Content-Language** ответа. Ниже представлены коды предупреждений (warn-code) с соответствующими уведомляющими текстами (warn-text) и расшифровка их значения. Эти предупреждения описывают сбои, вызванные неточным описанием сеанса. Все предупреждающие коды начинаются с цифры «3». Предупреждения с 300 по 329 предназначены для проблем с ключевыми словами (keywords) в описании сеанса, с 330 по 339 относятся к основным сетевым услугам, запрашиваемым в описании сеанса, с 370 по 379 относятся к количественным параметрам качества обслуживания, также запрашиваемым в описании сеанса, с 390 по 399 – разнообразные предупреждения, не относящиеся ни к одной из перечисленных категорий. Дополнительные коды могут быть определены с согласия организации IANA (Internet Assigned Numbers Authority).

Примеры:

```
Warning:307 serv1.niits.ru «Параметр сессии «example» не понят»
Warning:301 serv1.niits.ru «Несовестимый тип сетевого адреса «E.164»»
```

3.2.69. Заголовок WWW-Authenticate

Заголовок **WWW-Authenticate** содержит запрос аутентификации, который состоит из поля, отображающего схему аутентификации, и ряд параметров,

необходимых для проведения процедуры аутентификации с данным прокси-сервером для указанного Request-URI. Заголовок включается в состав ответа с кодом 407 (Proxy Authentication Required) или с кодом 401 (Unauthorized). Более подробная информация о заголовке содержится в § 6.4. Пример:

```
WWW Authenticate: Digest realm=«niits.ru»,
Qop=«auth», nonce=«f84f1cec41e6cbe5aea9c8e88d359»,
Opaque=«», stale=FALSE, algorithm=MD5
```

Информация о связи заголовков с запросами и ответами протокола SIP v. 2.0 и об обработке их прокси-серверами отображена в табл. 3.3.

Столбец «Действия прокси-сервера» описывает операции, которые может произвести прокси-сервер над заголовком.

a – прокси-сервер может объединять заголовки или добавлять их в случае отсутствия,

m – прокси-сервер может изменить значение поля заголовка,

d – прокси-сервер может удалить значение поля заголовка,

r – прокси-сервер должен быть в состоянии прочитать заголовок, и, соответственно, заголовок не может быть закодирован.

Следующие колонки устанавливают присутствие заголовков в сообщениях разного типа.

C – потребность в наличии заголовка зависит от контекста сообщения,

M – заголовок обязателен,

M* – заголовок должен вставляться при отсылке, но клиенты/серверы должны быть готовы принять сообщение без данного заголовка,

O – заголовок не обязателен,

***** – требуется наличие заголовка в случае, когда тело сообщения не пустое,

F – присутствие заголовка запрещено,

N/A – присутствие заголовка не определено,

T – заголовок должен вводиться при передаче сообщения, но клиенты-серверы должны быть готовы принять сообщение без этого заголовка. Если в качестве транспортного протокола используется потоко-ориентированный протокол, заголовок должен быть введен в сообщение при передаче.

Таблица 3.2. Список предупреждений

Код предупреждения	Текст предупреждения	Расшифровка
300	Несовместимый сетевой протокол	Один или несколько сетевых протоколов, указанных в описании сеанса, не поддерживаются
301	Несовместимые форматы сетевого адреса	Один или несколько форматов сетевых адресов, указанных в описании сеанса, не поддерживаются
302	Несовместимый транспортный протокол	Один или несколько транспортных протоколов, указанных в описании сеанса, не поддерживаются
303	Несовместимые единицы измерения пропускной способности	Одна или несколько единиц измерения пропускной способности, указанных в описании сеанса, не были поняты
304	Тип медиа-информации недоступен	Один или несколько типов медиа-информации, указанных в описании сеанса, недоступны
305	Несовместимый формат медиа-информации	Один или несколько форматов медиа-информации, указанных в описании сеанса, недоступны
306	Атрибут не понят	Один или несколько атрибутов медиа-информации, указанных в описании сеанса, не поддерживаются
307	Параметр описания сеанса не понят	Параметр, отличный от перечисленных выше, не был понят
330	Multicast-адресация не поддерживается	В месте расположения пользователя multicast-адресация не поддерживается
331	Unicast-адресация не поддерживается	В месте расположения пользователя unicast-адресация не поддерживается (обычно, в связи с присутствием сетевого экрана – firewall)
370	Недостаточная пропускная способность	Пропускная способность, указанная в описании сеанса, или обусловленная медиа-информацией, превышает возможный предел
399	Общее предупреждение	В предупреждающем тексте может содержаться любая информация для оповещения пользователя или для записи в log-файл. Система, получившая такое предупреждение, не должна производить никаких автоматических действий

Таблица 3.3. Связь заголовков с запросами и ответами протокола SIP v. 2.0

Название заголовка	Место использования заголовка	Действия прокси - сервера	ACK	BYE	CAN	INV	OPT	REG	IFO	SUB	NOT	PRK	UPD	REF	MSG
Accept	Заголовок в запросах	–	F	O	F	O	M*	O	O	O	O	O	O	O	F
Accept	Заголовок в ответах 2xx	–	F	F	F	O	M*	O	N/A	F	F	F	O	F	F
Accept	Заголовок в ответе 415	–	F	C	F	C	C	C	N/A	O	O	C	C	C	M*
Accept-Encoding	Заголовок в запросах	–	F	O	F	O	O	O	O	O	O	O	O	O	F
Accept-Encoding	Заголовок в ответах 2xx	–	F	F	F	O	M*	O	N/A	F	F	F	O	F	F
Accept-Encoding	Заголовок в ответе 415	–	F	C	F	C	C	C	N/A	O	O	C	C	C	M*
Accept-Language	Заголовок в запросах	–	F	O	F	O	O	O	O	O	O	O	O	O	F
Accept-Language	Заголовок в ответах 2xx	–	F	F	F	O	M*	O	N/A	F	F	F	O	F	F
Accept-Language	Заголовок в ответе 415	–	F	C	F	C	C	C	N/A	O	O	C	C	C	M*
Alert-Info	Заголовок в запросах	a, r	F	F	F	O	F	F	N/A	F	F	F	F	F	F
Alert-Info	Заголовок в ответе 180	a, r	F	F	F	O	F	F	N/A	F	F	F	F	F	F
Allow	Заголовок в запросах	–	F	O	F	O	O	O	N/A	O	O	O	O	O	O
Allow	Заголовок в ответах 2xx	–	F	O	F	M*	M*	O	F	O	O	O	O	O	O
Allow	Заголовок в ответах	–	F	O	F	O	O	O	N/A	O	O	O	O	O	O
Allow	Заголовок в ответе 405	–	F	M	F	M	M	M	O	M	M	M	M	M	M
Allow-Events	Заголовок в запросах	–	O	O	F	O	O	O	N/A	O	O	O	N/A	N/A	N/A
Allow-Events	Заголовок в ответах 2xx	–	F	O	F	O	O	O	N/A	O	O	O	N/A	N/A	N/A
Allow-Events	Заголовок в ответе 489	–	F	F	F	F	F	F	F	M	M	F	F	F	N/A
Authentication-Info	Заголовок в ответах 2xx	–	F	O	F	O	O	O	N/A	O	O	O	O	O	O
Authorization	Заголовок в запросах	–	O	O	O	O	O	O	O	O	O	O	O	O	O

Название заголовка	Место использов. заголовка	Действия прокси - сервера	ACK	BYE	CAN	INV	OPT	REG	IFO	SUB	NOT	PRK	UPD	REF	MSG
Call-ID	Общий заголовок, копируется из запросов в ответы	R	M	M	M	M	M	M	M	M	M	M	M	M	M
Call-Info	Общий заголовок	a, r	F	F	F	O	O	O	N/A	N/A	N/A	F	O	F	O
Contact	Заголовок в запросах	–	O	F	F	M	O	O	O	M	M	F	M	M	F
Contact	Заголовок в ответах 1xx	–	F	F	F	O	F	F	F	O	O	F	O	F	F
Contact	Заголовок в ответах 2xx	–	F	F	F	M	O	O	F	M	O	F	M	M	F
Contact	Заголовок в ответах 3xx	D	F	O	F	O	O	O	F	M	M	O	O	O	O
Contact	Заголовок в ответе 485	–	F	O	F	O	O	O	F	O	O	O	O	O	O
Contact	Заголовок в ответах 4xx-6xx	–	F	F	F	F	F	F	F	F	F	F	F	O	F
Content-Disposition	Заголовок содержания	–	O	O	F	O	O	O	N/A	O	O	O	O	O	O
Content-Encoding	Заголовки содержания	–	O	O	F	O	O	O	O	O	O	O	O	O	O
Content-Language	Заголовок содержания	–	O	O	F	O	O	O	N/A	O	O	O	O	O	O
Content-Length	Заголовок содержания	A, r	T	T	T	T	T	T	O	T	T	O	T	O	T
Content-Type	Заголовок содержания	–	*	*	F	*	*	*	*	*	*	*	*	*	*
Cseq	Общий заголовок, копир. из запросов в ответы	R	M	M	M	M	M	M	M	M	M	M	M	M	M
Date	Общий заголовок	A	O	O	O	O	O	O	O	O	O	O	O	O	O
Error-Info	Заголовок в ответах 300-699	A	F	O	O	O	O	O	N/A	O	O	O	O	O	O
Event	Заголовок в запросах	–	F	F	F	F	F	F	F	M	M	F	F	N/A	N/A
Expires	Общий заголовок	–	F	F	F	O	F	O	O	O	F	F	F	O	O
In-Reply-To	Заголовок в запросах	–	F	F	F	O	F	F	N/A	F	F	F	F	F	O
Max-Forwards	Заголовок в запросах	a, m, r	M	M	M	M	M	M	O	M	M	M	M	M	M
Expires	Заголовок в ответах 2xx		F	F	F	O	F	O	O	M	F	F	F	F	O

Название заголовка	Место использов. заголовка	Действия прокси - сервера	ACK	BYE	CAN	INV	OPT	REG	IFO	SUB	NOT	PRK	UPD	REF	MSG
From	Об. заголовок/ копируется из запросов в ответы	R	M	M	M	M	M	M	M	M	M	M	M	M	M
Min-Expires	Заголовок в ответе 423	–	F	F	F	F	F	M	N/A	M	F	F	F	F	N/A
MIME-Version	Общий заголовок	–	O	O	F	O	O	O	N/A	O	O	O	O	O	N/A
Organization	Общий заголовок	A, r	F	F	F	O	O	O	O	O	F	F	O	O	O
Path	Заголовок в запросах	a, r	F	F	F	F	F	O	F	F	F	F	F	F	F
Path	Заголовок в ответах 2xx	–	F	F	F	F	F	O	F	F	F	F	F	F	F
Priority	Заголовок в запросах	a, r	F	F	F	O	F	F	O	O	F	F	F	F	O
Privacy	Общий заголовок	a, m, r, d	O	O	O	O	O	O	O	O	O	O	O	N/A	O
Proxy-Authenticate	Заголовок в ответе 407	a, r	F	M	F	M	M	M	O	M	M	M	M	M	M
Proxy-Authenticate	Заголовок в ответе 401	a, r	F	O	O	O	O	O	N/A	N/A	N/A	O	O	O	O
Proxy-Authorization	Заголовок в запросах	d, r	O	O	F	O	O	O	O	O	O	O	O	O	O
Proxy-Require	Заголовок в запросах	a, r	F	O	F	O	O	O	O	O	O	O	O	O	O
P-Access-Network-Info	Общий заголовок	D, r	F	O	F	O	O	O	O	O	O	O	O	O	O
P-Asserted-Identity	Общий заголовок	A, d, r	F	O	F	O	O	F	F	O	O	F	F	O	N/A
P-Associated-URI	Заголовок в ответах 2xx	–	F	F	F	F	F	O	F	F	F	F	F	F	F
P-Called-Party-ID	Заголовок в запросах	A, m, r	F	F	F	O	O	F	F	O	F	F	F	O	O
P-Charging-Vector	Общий заголовок	A, d,m,r	F	O	F	O	O	O	O	O	O	O	O	O	O
P-Charging-Function-Addresses	Общий заголовок	A, d, r	F	O	F	O	O	O	O	O	O	O	O	O	O
P-DCS-Billing-Info	Общий заголовок	A, d, m, r	F	F	F	O	F	F	F	F	F	F	F	F	N/A
P-DCS-LAES	Общий заголовок	A, d, r	F	F	F	O	F	F	F	F	F	F	F	F	N/A
P-DCS-Redirect	Общий заголовок	A, d, r	F	F	F	O	F	F	F	F	F	F	F	F	N/A

Название заголовка	Место использов. заголовка	Действия прокси-сервера	ACK	BYE	CAN	INV	OPT	REG	IFO	SUB	NOT	PRK	UPD	REF	MSG
P-DCS-OSPS	Заголовок в запросах	D, r	F	F	F	O	F	F	F	F	F	F	O	F	N/A
P-DCS-Trace-Party-ID	Заголовок в запросах	D, r	F	F	F	O	F	F	F	F	F	F	F	F	N/A
P-Media-Authorization	Заголовок в запросах	A, d	O	F	F	O	F	F	F	F	F	O	O	N/A	N/A
P-Media-Authorization	Заголовок в ответах 2xx	A, d	F	F	F	O	F	F	F	F	F	O	O	N/A	N/A
P-Media-Authorization	Заголовок в ответах 101–199	A, d	F	F	F	O	F	F	N/A	N/A	N/A	N/A	N/A	N/A	N/A
P-Preffered-Identity	Общий заголовок	A, d, r	F	O	F	O	O	F	F	O	O	F	F	O	N/A
P-Visited-Network-ID	Заголовок в запросах	A, d	F	F	F	O	O	O	F	O	F	F	F	O	O
RAck	Заголовок в запросах	–	F	F	F	F	F	F	F	F	F	M	F	F	N/A
Reason	Заголовок в запросах и в ответах 1xx	–	O	O	O	O	F	F	O	O	O	O	O	O	O
Record-Route	Заголовок в запросах	A, r	O	O	O	O	O	F	N/A	O	O	O	O	O	F
Record-Route	Заголовок в ответах 18x	M, r	F	O	O	O	O	F	N/A	F	F	O	O	O	F
Record-Route	Заголовок в ответах 2xx	M, r	F	O	O	O	O	F	N/A	O	O	O	O	O	F
Record-Route	Заголовок в ответах 401, 484	M, r	F	F	F	F	F	F	F	O	O	F	F	F	F
Refer-To	Заголовок в запросах	–	F	F	F	F	F	F	F	F	F	F	F	M	F
Reply-To	Общий заголовок	–	F	F	F	O	F	F	N/A	F	F	F	F	F	O
Require	Заголовок в запросах	a, r	F	C	F	C	C	C	O	O	O	C	C	C	C
Retry-After	Заголовок в ответах 404, 413, 480, 486, 500, 503, 600 и 603	–	F	O	O	O	O	O	O	O	O	O	O	O	O
Route	Заголовок в запросах	a, d, r	C	C	C	C	C	C	O	C	C	C	C	C	O
RSeq	Заголовок в ответах 1xx	–	F	F	F	O	F	F	F	O	O	F	F	N/A	N/A
Security-Client	Заголовок в запросах	A, r, d	F	O	F	O	O	O	O	O	O	F	O	N/A	O

Название заголовка	Место использов. заголовка	Действия прокси - сервера	ACK	BYE	CAN	INV	OPT	REG	IFO	SUB	NOT	PRK	UPD	REF	MSG
Security-Server	Заголовок в ответах 421, 494	–	F	O	F	O	O	O	O	O	O	F	O	N/A	O
Security-Verify	Заголовок в запросах	A, r, d	F	O	F	O	O	O	O	O	O	F	O	N/A	O
Server	Заголовок в ответах	–	F	O	O	O	O	O	O	O	O	O	O	O	O
Subscription-State	Заголовок в запросах	–	F	F	F	F	F	F	F	F	M	F	F	F	N/A
Subject	Заголовок в запросах	–	F	F	F	O	F	F	O	F	F	F	F	F	O
Supported	Заголовок в запросах	–	F	O	O	M*	O	O	N/A	O	O	O	O	O	N/A
Supported	Заголовок в ответе 2xx	–	F	O	O	M*	M*	O	N/A	O	O	O	O	O	N/A
Timestamp	Общий заголовок	–	O	O	O	O	O	O	O	O	O	O	O	O	O
To	Общий заголовок, копируется из запросов в ответы	R	M	M	M	M	M	M	M	M	M	M	M	M	M
Unsupported	Заголовок в ответе 420	–	F	M	F	M	M	M	O	O	O	M	M	O	O
User-Agent	Общий заголовок	–	O	O	O	O	O	O	O	O	O	O	O	O	O
Via	Заголовок в запросах	a, m, r	M	M	M	M	M	M	M	M	M	M	M	M	M
Via	Заголовок в ответах, копируется из запросов в ответы	d, r	M	M	M	M	M	M	M	M	M	M	M	M	M
Warning	Заголовок в запросах	–	F	O	O	O	O	O	O	F	O	O	O	F	F
Warning	Заголовок в ответах	–	F	O	O	O	O	O	O	O	O	O	O	O	O
WWW-Authenticate	Заголовок в ответе 401	a, r	F	M	F	M	M	M	O	M	M	M	M	M	M
WWW-Authenticate	Заголовок в ответе 407	a, r	F	O	F	O	O	O	N/A	N/A	N/A	N/A	O	O	O

3.3. Назначение и формат запросов

3.3.1. Структура запросов

SIP-запросы характеризуются наличием строки Request-Line в стартовой строке. Request-Line состоит из названия типа запроса, Request-URI и версии протокола, разделенных пробелом (например, ACK *sip:anton@niits.ru SIP/2.0*). Request-Line заканчивается символами возврата каретки и перевода строки (CRLF). Оба символа, вместе или по одиночке, не должны встречаться в других частях строки. Использование линейного пробела (LWS – произвольное сочетание пробелов и символов горизонтальной табуляции согласно RFC 822) не допускается.

Рис. 3.3. Структура строки Request-Line

- **Тип запроса**

  В базовой рекомендации IETF RFC 3261 определено 6 типов запросов: REGISTER для регистрации контактной информации, INVITE, ACK и CANCEL для установления сеансов, BYE для завершения сеансов, и OPTION для запроса информации о функциональных возможностях сервера. Каждый из них предназначен для выполнения довольно широкого круга задач, что является явным достоинством протокола SIP, так как число сообщений, которыми обмениваются терминалы и серверы, сведено к минимуму. Сервер определяет тип принятого запроса по названию, указанному в стартовой строке.

- **Request-URI**

  Request-URI – это SIP или SIPS URI. Он указывает пользователя или услугу, к которой адресован запрос. Поле Request-URI не должно содержать пробелов и управляющих символов, а также не должно быть заключенным в угловые скобки «<>».

  Элементы сети SIP могут поддерживать поля Request-URI со схемами, отличными от «sip» и «sips», например «tel»; они в состоянии

преобразовать не-SIP URI с использованием любых доступных им механизмов. Результатом преобразования будет или SIP URI, или SIPS URI, или другая схема.

Содержание полей **To** и Request-URI может быть различным, например, в поле **To** может быть указан списочный адрес получателя, а в Request-URI – адрес прокси-сервера, через который проходит запрос.

- **Версия протокола**

  И запросы, и ответы содержат данные действующей версии SIP-протокола, принимая во внимание порядок, соответствие требованиям и изменение численного индекса версии. Приложения, передающие SIP-сообщения, в поле SIP-Version должны указывать «SIP/2.0». Поле SIP-Version регистронезависимо, однако при разработке оборудования рекомендуется использовать верхний регистр.

3.3.2. Запрос INVITE

Запрос INVITE – приглашает вызываемого агента пользователя принять участие в сеансе связи. Сообщение обычно содержит описание сеанса, указывающее вид информации и параметры (список возможных вариантов параметров), необходимые для приема информации. В ответе на запрос INVITE указывается вид информации, которая будет приниматься вызываемым пользователем, а кроме того, могут указываться возможные параметры передачи информации.

В этом сообщении могут также содержаться данные, необходимые для аутентификации абонента, и, следовательно, доступа клиентов к SIP-серверу. В случае необходимости изменения уже установленных характеристик передается запрос INVITE с новым описанием сеанса. Для приглашения нового участника к уже установленному соединению также используется сообщение INVITE, которое передается его агенту пользователя.

На рис. 3.4 в качестве примера приведен типичный запрос INVITE.

В этом примере пользователь Anton (*anton@niits.ru*) вызывает пользователя Alexander (alexander@niits.ru). Запрос передается на прокси-сервер (serv1.niits.ru). В полях **To** и **From** перед адресом стоит надпись, которую вызывающий пользователь желает вывести на дисплей вызываемого пользователя.

```
INVITE sip:alexander@serv1.niits.ru SIP/2.0
          Via:SIP/2.0/UDP kton.niits.ru
          From:Anton<sip:anton@niits.ru>
          To:Alexander<sip:alexander@niits.ru>
          Cseq:1INVITE
          Call-ID:3298420296@kton.niits.ru
          Content-Type:application/sdp
          Content-Length: ...

          o=bell53655765 2353687637 IN IP4 128.3.4.5
          v=0
          C=IN IP4 kton.niits.ru
          m=audio 3456 RTP/AVP 0 3 4
```

Рис. 3.4. Пример запроса INVITE

В теле сообщения оборудование вызывающего пользователя указывает в формате протокола SDP, что оно может принимать в порту 3456 упакованную в пакеты RTP речевую информацию, закодированную по одному из алгоритмов кодирования: 0 – PCMU, 3 – GSM и 4 – G.723.

3.3.3. Сообщение ACK

ACK подтверждает прием ответа на команду INVITE. Следует отметить, что подтверждение ACK используется только совместно с запросом INVITE, т.е. этим сообщением оборудование вызывающего пользователя показывает, что оно получило окончательный ответ на свой запрос INVITE. В запросе ACK может содержаться окончательное описание сеанса, передаваемое вызывающим пользователем.

3.3.4. Сообщение CANCEL

CANCEL отменяет обработку ранее переданных запросов с такими же, как и в запросе CANCEL, значениями полей **Call ID**, **To**, **From** и **CSeq**, но не влияет на те запросы, обработка которых уже завершена. Например, запрос CANCEL применяется тогда, когда прокси-сервер размножает запросы для поиска пользователя по нескольким направлениям и находит его по одному из них. Обработку запросов, разосланных по всем остальным направлениям, сервер отменяет при помощи команды CANCEL.

3.3.5. Сообщение BYE

Сообщением BYE оборудование вызываемого или вызывающего пользователя завершает соединение. Сторона, получившая запрос BYE, должна прекратить передачу речевой (мультимедийной) информации и подтвердить выполнение запроса ответом 200 (OK).

3.3.6. Сообщение REGISTER

При помощи сообщения REGISTER пользователи сообщают свое текущее местоположение. В этом сообщении содержатся следующие заголовки:

- **To** содержит адресную информацию, которую надо сохранить или модифицировать на сервере.
- **From** содержит адрес инициатора регистрации. Зарегистрировать пользователя может другое лицо, например, секретарь может зарегистрировать своего начальника.
- **Contact** содержит новый контактный адрес пользователя, по которому должны передаваться все дальнейшие запросы **INVITE**. Если в команде **REGISTER** поле **Contact** отсутствует, регистрация остается неизменной. В случае отмены регистрации здесь помещается символ «*».
- **Expires** указывается время в секундах, по истечении которого регистрация заканчивается. Если это поле отсутствует, то по умолчанию назначается время 1 час, по истечении которого регистрация завершается. Регистрацию можно также отменить сообщением REGISTER с полем **Expires**, которому присвоено значение 0, и с соответствующим полем **Contact.**

3.3.7. Сообщение OPTIONS

Сообщением OPTIONS вызываемый пользователь запрашивает информацию о возможностях терминального оборудования вызываемого пользователя или сервера. В ответ на запрос оборудование вызываемого пользователя сообщает требуемую информацию. Применение запроса ограничено теми случаями, когда существует необходимость узнать о поддерживаемых возможностях оборудования до установления соединения. Для установления соединения запрос не используется.

После экспериментальной проверки протокола SIP в реальных сетях оказалось, что для решения ряда задач рассмотренных выше шести запросов недостаточно. Поэтому организацией IETF в протокол введены новые сообщения. Ими являются: INFO, PRACK, UPDATE, SUBSCRIBE, NOTIFY, REFER, MESSAGE.

3.3.8. Сообщение INFO

Запрос INFO предназначен для обмена сигнальной информацией по сигнальному тракту SIP в процессе установления и поддержания соединения. Запрос INFO не изменяет состояния процесса обработки SIP-вызовов, как не изменяет и состояния сеансов связи, инициированных при помощи протокола SIP. Однако он обеспечивает передачу дополнительной информации прикладного уровня, которая в дальнейшем может способствовать более производительному функционированию приложений, использующих протокол SIP для доставки информации.

Сигнальным трактом для запроса INFO является сигнальное соединение, созданное в результате выполнения процедуры обслуживания определенного вызова. Им может быть тракт сигнализации непосредственно между вызывающим и вызываемым агентами пользователя или сигнальный тракт с участием прокси-серверов SIP, которые были вовлечены в процедуру установления соединения и поместили свое значение заголовка **Record-Route** в начальное сообщение INVITE.

В ходе сеанса информация может быть передана или в заголовке сообщения INFO, или в части тела сообщения.

Возможными применениями запроса INFO являются:

- Перенос текущих сигнальных сообщений ТфОП между шлюзами ТфОП в течение сеансов связи;
- Перенос DTMF-сигналов, созданных в ходе сеанса;
- Перенос защищенной сигнальной информации беспроводных систем для поддержки мобильности приложений;
- Перенос информации об остатке на счете (билинговой информации);
- Перенос изображений и другой не потоковой информации между участниками сеанса связи.

Предполагается, что в скором времени появятся другие применения сообщения INFO, как в области телефонии, так и в других областях.

На сообщение INFO сервер агента пользователя должен передать окончательный ответ. Если запрос, вне зависимости от наличия в нем тела сообщения, был успешно принят сервером для существующего сеанса, то должен быть передан ответ с кодом 200 (OK). Когда формат тела сообщения неизвестен серверу, передается ответ с кодом 415 (Unsupported Media Type).

Запрос INFO может быть отменен. UAS, получивший сообщение CANCEL для отмены INFO, должен передать ответ с кодом 487 (Request Cancelled), если до этого не был отправлен окончательный ответ.

Правила обработки запроса INFO прокси-сервером, сервером перенаправления и сервером, размножающим запросы, идентичны правилам обработки запроса BYE. Дополнительные сведения о запросе INFO содержатся в [11].

3.3.9. Сообщение PRACK

Протокол SIP определяет два типа ответов на запрос, инициирующий соединение, – предварительные и окончательные. Окончательные ответы несут результат обработки запроса и передаются надежно (с подтверждением). Предварительные ответы несут информацию о текущей стадии обработки запроса, но передаются ненадежно (без подтверждения). Однако в некоторых случаях, включая взаимодействие с ТфОП, необходим механизм обеспечения надежности передачи предварительных ответов. Для поддержки надежной транспортировки предварительных ответов требуется наличие у элемента SIP соответствующей опции (идентифицирующейся с помощью option tag «100rel»).

Механизм надежности работает по схеме, сходной с существующими механизмами надежности для окончательных ответов класса 2xx на запрос INVITE. Надежные предварительные ответы повторно передаются TU (пользователем транзакций) с интервалом, возрастающим по экспоненциальному закону. Передача повторных ответов прекращается, когда приходит сообщение PRACK. Запрос типа PRACK играет ту же роль, что и ACK, но для предварительных ответов. Однако между ними имеется принципиальное различие – PRACK является обычным SIP-сообщением, как запрос BYE, т.е. его надежность обеспечивается последовательно (hop-by-hop) при прохождении через каждый stateful прокси-сервер.

Так же, как BYE (но в отличие от ACK), PRACK требует при его получении передачи ответа. На устанавливающий диалог запрос UAS может передать ответ класса 1xx (кроме 100) надежно при условии, что запрос содержит заголовок **Supported** с option tag «100rel». Это означает, что UAC поддерживает расширение для надежной передачи предварительных ответов и может принимать и обрабатывать такие ответы. В то же время, при создании запроса UAC может потребовать надежной доставки предварительных ответов на него. Для этого UAC вводит в запрос заголовок **Require** с option tag «100rel». Каждый предварительный ответ имеет порядковый номер, содержащийся в заголовке **RSeq** ответа (от 1 до $2^{31} - 1$). После формирования предварительного ответа ядро UAS периодически передает его серверной транзакции для пересылки. Предварительный ответ устанавливает диалог, если тот еще не был установлен.

При получении надежного ответа UAC создает запрос PRACK и передает его в находящемся на ранней стадии диалоге, который связан с предварительным ответом. Сообщение PRACK содержит заголовок **RAck**, который указывает порядковый номер подтверждаемого предварительного ответа. Повторная передача PRACK не зависят от получения ответов класса 1xx, т.е. UAC не должен повторно передавать запрос PRACK, когда получает повторно уже подтвержденный им предварительный ответ.

Повторения передачи надежного предварительного ответа прекращаются при получении ядром UAS запроса PRACK. Если PRACK соответствует неподтвержденному предварительному ответу, передается ответ класса 2xx.

После того как надежный предварительный ответ был подтвержден, UAS может передавать дополнительные надежные предварительные ответы. Значение заголовка **RSeq** в каждом следующем предварительном ответе будет на единицу больше.

Дополнительные сведения о запросе PRACK содержатся в [55]. Пример использования запроса PRACK приведен на рис. 3.5.

3.3.10. Сообщение UPDATE

Часто возникают случаи, когда необходимо изменить некоторые параметры сеанса (например, кодеки) до прихода окончательного ответа на начальное сообщение INVITE. Например, предответное состояние (early media) – ситуация, когда

сеанс устанавливается для передачи информации о текущем состоянии процесса обслуживания вызова до того, как на запрос INVITE сервером будет передан ответ. Важно, чтобы и вызывающая, и вызываемая стороны могли изменять параметры сеанса до того как поступит ответ на вызов. Запрос re-INVITE (INVITE, передаваемый в ходе сеанса, см. раздел 4.11) не может быть использован для этих целей, поскольку re-INVITE оказывает влияние не только на состояние сеанса, но и на состояние диалога. В противоположность ему, запрос UPDATE может быть передан агентом пользователя в режиме диалога (находящегося на ранней стадии или установленного) для обновления параметров сеанса без воздействия на состояние диалога.

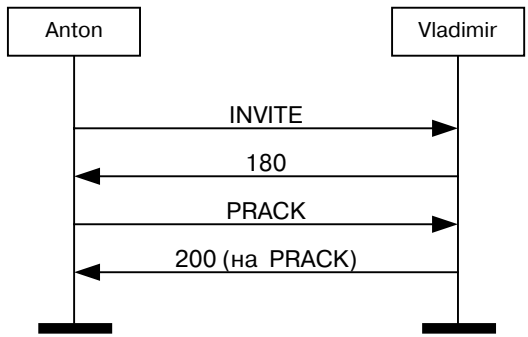


Рис. 3.5. Пример использования запроса PRACK

Запрос UPDATE используется следующим образом. Вызывающая сторона создает начальную INVITE-транзакцию. В поле заголовка **Allow** запроса INVITE, среди прочих типов запроса, указывается UPDATE для того, чтобы указать на способность вызывающей стороны принимать запросы этого типа. INVITE-транзакция протекает надлежащим образом. Любой ответ (предварительный или окончательный) от вызываемой стороны также содержит заголовок **Allow** с указанным в нем значением *UPDATE*.

После перехода в режим диалога (находящегося на ранней стадии или установленного) вызывающая сторона может создать запрос UPDATE, который содержит информацию offer (предложение с описанием сеанса связи в формате SDP), предназначенную для обновления параметров сеанса. Ответ на этот запрос

переносит информацию answer (ответ на предложение с указанием принятых параметров, также в формате SDP). Подобным образом, после установления диалога вызываемая сторона может передать запрос UPDATE с информацией offer, а вызывающая сторона – поместить в **answer** в ответ класса 2xx на UPDATE. Если UA получает окончательный ответ не-2xx на UPDATE, параметры сеанса должны оставаться неизменными.

Запрос UPDATE с новой информацией offer не может быть передан (и, соответственно, не может быть принят) до тех пор, пока на информацию offer, переданную в начальном INVITE, ненадежном предварительном ответе, надежном предварительном ответе, запросе PRACK или предыдущем UPDATE не получена информация answer. Это действительно для случая, когда начальная INVITE-транзакция не завершена. Когда она заканчивает свое существование, то же относится к информации answer, передаваемой в ответ на информацию offer, содержащуюся в re-INVITE или в предыдущем UPDATE.

UPDATE является запросом, обновляющим текущий адрес удаленного пользователя (target refresh request). Дополнительные сведения о запросе UPDATE содержатся в [51]. Пример использования запроса UPDATE приведен на рис. 3.6.

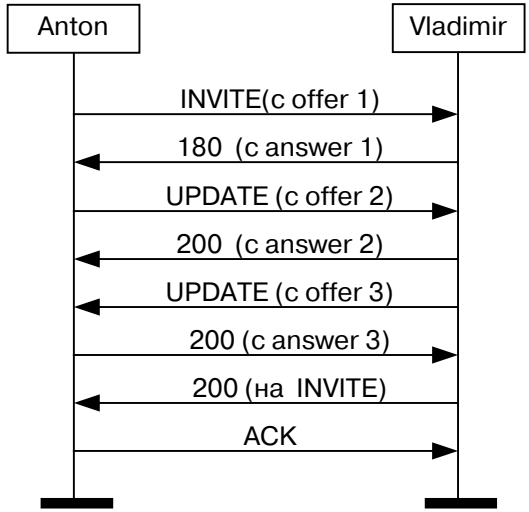


Рис. 3.6. Пример использования запроса UPDATE

3.3.11. Сообщения SUBSCRIBE и NOTIFY

Для большинства реализованных на базе протокола SIP услуг, которые требуют взаимодействия между конечными точками, представляет интерес возможность запрашивать асинхронное уведомление о событиях. Эти услуги включают в себя: услуги автоматического обратного вызова (связанные с изменениями состояния терминала пользователя), интерактивные списки контактов «buddy lists» (связанные с определением присутствия пользователя), оповещения об ожидающем сообщении (связанные с событиями изменения состояния почтового ящика) и передача информации о состоянии процесса обслуживания вызова при взаимодействии сетей Интернет и ТфОП.

Объекты сети SIP могут подписаться на предоставление информации о состоянии определенного ресурса или процесса обслуживания вызова в сети, и объекты, располагающие этими сведениями (или объекты, действующие от их лица), будут передавать уведомления каждый раз, когда это состояние изменится. Типичный вариант обмена сообщениями представлен на рис. 3.7.

1 - запрос подписки на предоставление информации о состоянии
2 - подтверждение подписки
3 - передача информации о текущем состоянии
4 - подтверждение приёма
5 - передача информации о текущем состоянии
6 - подтверждение приёма

Рис. 3.7. Процедура подписки на предоставление информации

Запрос SUBSCRIBE используется для запроса информации о текущем состоянии и информации об обновлениях состояния удаленного ресурса. SUBSCRIBE – это запрос, создающий диалог. Когда подписчик (subscriber – агент пользователя, запрашивающий и получающий информацию о состоянии ресурса) желает подписаться на получение информации о состоянии ресурса, он формирует сообщение SUBSCRIBE. Если начальное сообщение SUBSCRIBE представляет собой запрос вне диалога, как зачастую и бывает, его формирование проходит с привлечением процедур создания запроса вне диалога для UAC. Идентификация событий, на уведомления о которых производится подписка, обеспечивается с помощью трех компонентов запроса SUBSCRIBE: поля Request URI, заголовка **Event** и, опционально, тела сообщения.

Request URI содержит исчерпывающую информацию идентификации ресурса, в отношении которого требуется провести процедуру уведомления (notification), но не обязательно содержит достаточно информации для того, чтобы уникально идентифицировать тип события (например, *sip:vladimir@protei.ru* был бы подходящим URI как для подписки на информацию о состоянии присутствия пользователя (user presence state), так и для подписки на информацию о состоянии ящика речевой почты пользователя).

В запросе SUBSCRIBE должен присутствовать ровно один заголовок **Event**, указывающий событие или класс событий, на уведомление о котором производится подписка. Заголовок содержит значение, указывающее тип состояния, информация о котором затребована подписчиком. Это значение носит название event package и описывает значение события или класса событий. Заголовок **Event** может также содержать параметр «id». Параметр идентифицирует конкретную подписку в пределах диалога. Большинство event package потребуют наличия в запросе SUBSCRIBE *тел сообщения* (синтаксис и семантику для таких тел определяют event package). Эти тела, как правило, будут модифицировать, уточнять, отфильтровывать, прерывать и/или устанавливать границы для запрашиваемого класса событий. В запросе SUBSCRIBE может присутствовать заголовок **Allow**, указывающий, какие типы event package он поддерживает. Когда время действия подписки истекает, подписчик может обновить таймер этой подписки путем передачи еще одного сообщения SUBSCRIBE с таким же значением параметра «id» заголовка **Event** в том же диалоге, где существует начальная подписка. Пользователь может также отказаться от подписки. Процедура отказа протекает так же, как и процедура обновления подписки с единственным отличием, заключающемся в том, что значение поля заголовка **Expires** – нулевое.

Запрос SUBSCRIBE должен быть подтвержден окончательным ответом. При этом уведомитель (notifier – агент пользователя, информирующий его о состоянии ресурса) не должен ожидать от пользователя подтверждения, прежде чем передать окончательный ответ. Если уведомитель способен незамедлительно определить, что он поддерживает event package, что аутентифицированный подписчик авторизован на подписку и что не существует других препятствий, чтобы создать подписку, он создает подписку, и в случае необходимости – диалог, а затем передает ответ с кодом 200 (OK). Может быть также передан ответ с кодом 202 (Accepted). Такой ответ означает, что запрос был получен и понят, но подписка, возможно, еще не авторизована. После того как подписка была успешно создана или обновлена, уведомитель должен незамедлительно передать сообщение NOTIFY, чтобы сообщить подписчику текущее состояние ресурса.

Запрос NOTIFY передается в том же диалоге, который был создан ответом на запрос SUBSCRIBE (если не существовало заранее установленного диалога). Когда происходит изменение состояния, на уведомление о котором была открыта подписка, подписчику также передается запрос NOTIFY. Таким образом, тип запроса NOTIFY используется для уведомления узла SIP о том, что событие, информация о котором запрашивалась в запросе SUBSCRIBE, произошло. Он может также содержать подробности, связанные с этим событием.

Запрос NOTIFY, так же, как SUBSCRIBE, содержит заголовок **Event** со значением event package, для которого производится уведомление; параметр «id» заголовка должен совпадать с аналогичным параметром в SUBSCRIBE. В запросах NOTIFY могут присутствовать тела сообщения, их семантику определяют event package. Тела содержат дополнительную информацию о типе произошедшего события и о новом состоянии ресурса. Сообщение NOTIFY должно содержать заголовок **Subscription-State** со значением либо *active*, либо *pending*, либо *terminated*.

Значение *active* указывает, что запрос подписки был принят и авторизован. Значение *pending* означает, что запрос подписки был получен, но не может быть принят или отклоняется из-за недостатка информации. Значение *terminated* сообщает, что подписка окончена. После получения запроса NOTIFY подписчик должен проверить, соответствует ли запрос хотя бы одной из его существующих подписок (соответствие будет иметь место, если NOTIFY относится к тому же диалогу и имеет соответствующее значение заголовка **Event**). После того как подписчик принял уведомление, он должен передать ответ с кодом 200 (OK). Дополнительные сведения о запросах SUBSCRIBE и NOTIFY даны в [50].

3.3.12. Сообщение REFER

Запрос REFER, передаваемый отправителем, предписывает получателю (идентифицированному адресом в поле Request-URI) связаться с третьей стороной, используя контактную информацию, которая содержится в сообщении. Такой механизм может быть использован для многих целей, включая переадресацию вызова (Call Transfer). Например, если Anton установил сеанс связи с пользователем Vladimir и решает, что Vladimir должен поговорить с пользователем Alexander, Anton может поручить своему SIP UA передать запрос REFER SIP-агенту пользователя Vladimir, снабдив его контактной информацией пользователя Alexander. Если Vladimir даст свое согласие, его UA попытается связаться с пользователем Alexander, используя контактный адрес. После этого UA пользователя Vladimir уведомит UA пользователя Anton, насколько успешно прошла попытка установить связь с пользователем Alexander.

Если не определено иначе, правила передачи запросов REFER и ответов на них те же, что и для запроса BYE. В запрос REFER включается заголовок **Refer-To**. Значение, содержащееся в заголовке, указывает адрес третьей стороны, с которой отправитель предлагает связаться получателю запроса REFER. Например, *sip:anton@niits.ru.*

Если запрос принят, UAS должен передать ответ с кодом 202 (Accepted) до того, как истечет время действия REFER-транзакции. Вслед за этим UA получателя создает подписку. Подписка, создаваемая запросом REFER, по своей сути является такой же, как подписка, создаваемая запросом SUBSCRIBE. REFER – это только механизм, который создает подписку на уведомление о событии переадресации – «refer». Создание подписки влечет за собой незамедлительную передачу запроса NOTIFY. Механизм передачи сообщений NOTIFY используется для того, чтобы информировать UA, передавший REFER, о статусе переадресации. Идентифицирующие диалог значения заголовков **To**, **From** и **Call-ID** в каждом сообщении NOTIFY должны соответствовать аналогичным заголовкам в запросе REFER, как если бы вместо REFER был запрос SUBSCRIBE. Каждое сообщение NOTIFY должно содержать заголовок **Event** со значением *refer*. NOTIFY содержит также тело сообщения типа message/sipfrag (тип тела сообщения message/sipfrag подробно описан в [67]), содержащее исчерпывающую информацию о состоянии действия переадресации вызова; в теле сообщения может также содержаться дополнительная информация о результате действия.

Запрос NOTIFY может быть создан всякий раз, когда появляется новая информация о статусе переадресации вызова. Уведомления о событии «refer» не могут передаваться чаще, чем один раз в секунду. Возможен и такой вариант, когда за время подписки передается ровно два сообщения NOTIFY – безотлагательный запрос NOTIFY и NOTIFY, содержащий окончательный результат переадресации вызова. Окончательный запрос NOTIFY содержит заголовок **Subscription-State** со значением terminated (подписка окончена) и со значением параметра «reason» – noresource. Подробнее сведения о запросе REFER приведены в [68]. На рис. 3.8 представлен пример обмена сообщениями между терминалом пользователя Anton и терминалом пользователя Alexander при успешной переадресации вызова, где пользователь Alexander переадресует вызов другому абоненту.

3.3.13. Сообщение MESSAGE

Интерактивный обмен текстовыми сообщениями (Instant Messaging) происходит между группой участников в режиме, близком к реальному времени. Как правило, сообщения имеют малый размер и не сохраняются. От электронной почты эти сообщения отличается тем, что они обычно относятся к коротким сеансам взаимодействия (группируются вокруг них) с большим числом коротких сообщений, передаваемых в обе стороны.

Запрос типа MESSAGE предназначен для передачи мгновенных текстовых сообщений (instant messages), используя модель, походящую на функционирование двустороннего пейджера или работу телефона при отправке SMS. Прямой связи между сообщениями не существует. Каждое текущее сообщение (IM) независимо – информация о том, что происходит взаимодействие, может быть отражена только в пользовательском интерфейсе клиента или, возможно, в воображении самого пользователя. Такой подход полностью противоположен сеансовой модели взаимодействия участников с четко определенным началом и концом.

Когда один пользователь решает послать другому пользователю текущее текстовое сообщение (IM), отправитель формирует запрос типа MESSAGE и передает его. Поле запроса Request-URI , как обычно, будет указывать списочный адрес получателя текущего сообщения, однако оно может содержать и адрес устройства в случаях, когда клиент владеет информацией о текущем местонахождении получателя. Например, клиент может быть совмещен с системой присутствия, которая при вводе списочного адреса дает новейшую контактную информацию устройства, где в данный момент находится пользователь.

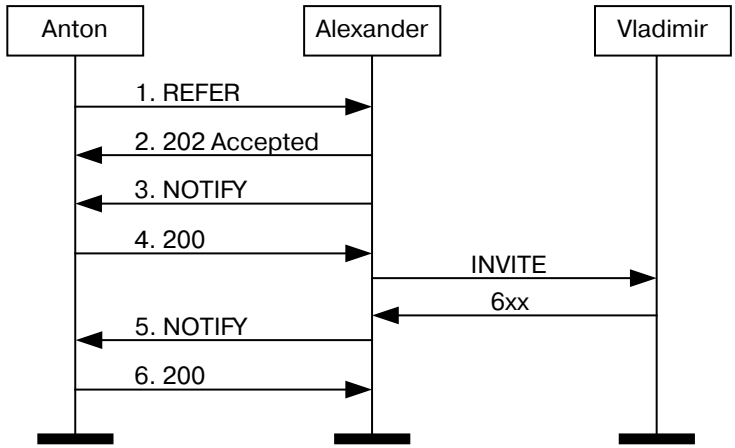


Рис. 3.8. Пример использования запроса REFER

Содержание сообщения 1

```
REFER sip:alexander@niits.ru SUP/2.0
Via:SIP/2.0/UDP ant.niits.ru;branch=z9hG4bK2293940223
To:<sip:alexander@niits.ru>
From:<sip:anton@niits.ru>;tag=193402342
Call-ID: 898234234@ant.niits.ru
CSeq:93809823 REFER
Max-Forwards:70
Refer-To:(любой URI)
Contact:sip:anton@niits.ru
Content-Length:0
```

Содержание сообщения 5

```
NOTIFY sip:anton@niits.ru SIP/2.0
To:<sip:anton@niits.ru>;tag=193402342
From:<sip:alexander@niits.ru>;tag=4992881234
Call-ID:898234234@ant.niits.ru
CSeq:1993403 NOTIFY
Max-Forwards:70
Event:refer
Subscription-State: terminated;reason=noresource
Contact:sip:alexander@niits.ru
Content-Type: message/sipfrag;version=2.0
```

Тело сообщения будет включать в себя текстовое сообщение, которое необходимо доставить. Это тело может быть любого MIME-типа (зачастую – text/plain), включая тип message/cpim. Поскольку ожидается, что формат message/cpim будет поддерживаться другими IM-протоколами, терминалы, использующие разные IM-протоколы, но, в то же время, поддерживающие тип тела message/cpim, будут способны обмениваться текстовыми сообщениями без модификации содержания шлюзом или другим посредником. Агент пользователя не помещает в запрос MESSAGE заголовок **Contact**. Запрос MESSAGE пройдет через группу прокси-серверов и будет доставлен получателю. Получив запрос, UA получателя перейдет к его обработке и, в случае успеха, передает окончательный ответ с кодом 200 (OK); это означает, что текстовое сообщение было доставлено пользователю, но не указывает на то, что пользователь с ним ознакомился. Запросы MESSAGE не устанавливают диалога. UAC может передавать запрос MESSAGE в существующем диалоге. Текущие текстовые сообщения (IM) могут адресоваться с помощью Instant Message URI в форме «im:user@domain». URI со схемой «im» абстрактные, поэтому они должны преобразовываться в конкретные URI в зависимости от протокола (в данном случае SIP URI). Если в качестве адреса UA-получателя сообщения представлен IM URI, он должен быть переведен в SIP URI и помещен в поле Request-URI запроса MESSAGE перед отправкой.

Ниже представлен пример обмена сообщениями. В этом примере Anton передает пользователю Alexander начальное сообщение, оба пользователя находятся в одном домене niits.ru; между ними действует посредник – прокси-сервер. Дополнительные сведения о запросе MESSAGE даны в [7].

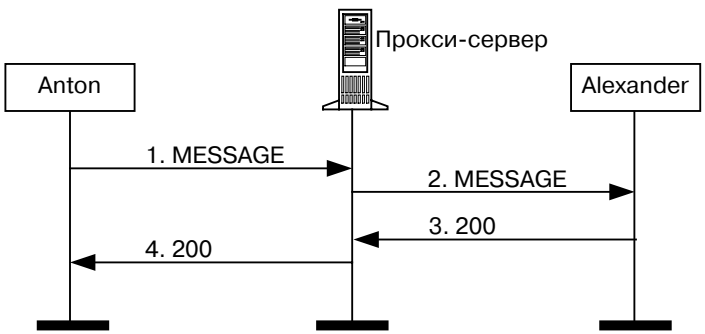


Рис. 3.9. Пример использования запроса MESSAGE

```
MESSAGE sip:alexander@niits.ru SIP/2.0
Via: SIP/2.0/TCP serv3.niits.ru;branch=z9hG4bK776sgdkse
Max-Forwards:70
From:sip:anton@niits.ru;tag=49583
To:sip:alexander@niits.ru
Call-ID: asd88asd77a@1.2.3.4
CSeq:1 MESSAGE
Content-Type:text/plain
Content-Length:30

Alexander, подойди пожалуйста.
```

Все запросы SIP с их кратким описанием сведены в табл. 3.4.

Таблица 3.4. Типы запросов SIP

Название запроса	Описание запроса
INVITE	Приглашает пользователя к сеансу связи.
ACK	Подтверждает прием окончательного ответа на запрос INVITE
BYE	Завершает сеанс. Посылается любой из сторон, участвующих в соединении.
CANCEL	Отменяет обработку запросов с такими же заголовками **Call-ID**, **To**, **From** и **Cseq**, как и в запросе CANCEL
REGISTER	Переносит адресную информацию для регистрации пользователя на сервере определения местоположения
OPTION	Запрашивает информацию о функциональных возможностях сервера
INFO	Переносит по сигнальному тракту управляющую и прочую информацию
PRACK	Используется для надежной транспортировки предварительных ответов
UPDATE	Служит для изменения параметров сеанса до прихода окончательного ответа (без воздействия на состояние диалога)
SUBSCRIBE	Запрашивает информацию о текущем состоянии и информацию об обновлении состояния удаленного ресурса
NOTIFY	Сообщает подписчику текущее состояние ресурса и уведомляет о том, что интересующее его событие произошло
REFER	Предписывает получателю связаться с третьей стороной, используя контактную информацию, которая содержится в сообщении
MESSAGE	Переносит мгновенные текстовые сообщения (instant massages)

3.4. Назначение и формат ответов на запросы

Характерное отличие SIP-ответов от запросов – это наличие строки Status-Line в стартовой строке. Status-Line составляют: версия протокола и код ответа (Status-Code) со связанной с ним текстовой расшифровкой (Reason-Phrase), разделенные пробелом (SP). Символы возврата каретки (CR) и перевода строки (LF) могут использоваться только совместно в завершающей строку последовательности CRLF.

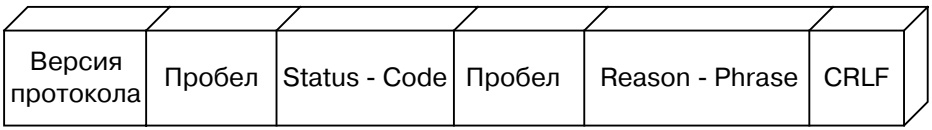


Рис. 3.10. Структура строки Status-Line

Код ответа – это целое трехзначное число, отражающее результат обработки запроса сервером. Reason-Phrase дает краткое описание кода ответа и предназначена для визуального восприятия пользователем, в отличие от Status-Code, который служит для оповещения технических устройств. К формулировке Reason-Phrase не предъявляется жестких требований: фирмы-производители вправе выбрать другой текст на произвольном национальном языке, указанном в поле заголовка **Accept-Language** запроса. Первая цифра кода ответа определяет класс ответа. Оставшиеся две цифры носят дополнительный характер и служат для упорядочения кодов в пределах категории. В некоторых случаях оборудованию даже необязательно знать все коды ответов, но оно обязательно должно интерпретировать первую цифру ответа. После приема и интерпретации запроса адресат передает ответ на полученный запрос. Назначение ответов бывает разным, в том числе, подтверждение установления соединения, передача запрашиваемой информации, сообщение о неисправностях и т.д. Структуру и виды ответов протокол SIP унаследовал от протокола HTTP.

Определено шесть классов ответов, которые несут разную функциональную нагрузку. Все ответы делятся на два типа: информационные и окончательные. Информационные ответы показывают, что запрос находится в стадии обработки. Информационные ответы кодируются трехзначным числом, начинающимся с единицы, – 1xx. Окончательные ответы кодируются трехзначными числами, начинающимися с цифр 2, 3, 4, 5 и 6. Все они означают завершение обработки запроса, а каждый из них в отдельности – результат обработки запроса.

3.4.1. Информационные ответы 1xx

Информационные или предварительные ответы содержат информацию о том, что запрашиваемый сервер находится на стадии обработки запроса и не может в данный момент дать окончательный ответ. Сервер передает ответ класса 1xx, если он предполагает, что формирование окончательного запроса займет более 200 мс. Предварительные ответы не требуют от клиента подтверждающего сообщения ACK; однако при поддержке обоими UA расширения функциональных возможностей «100rel» возможна передача ответов класса 1xx с подтверждением PRACK. Предварительные ответы могут содержать тело сообщения с описанием сеанса связи.

Таблица 3.5. Информационные ответы протокола SIP

Код	Назначение
100	**Trying.** Запрос обрабатывается. Например, сервер обращается к базе данных, но местоположение терминала вызываемого пользователя в настоящий момент не определено. Этот ответ, как и прочие предварительные ответы, пресекает повторную передачу клиентом сообщения INVITE. От остальных ответов 1xx его отличает то, что он не может быть пропущен stateful прокси-сервером. Будет, вероятнее всего, передаваться от прокси-сервера или от другого промежуточного SIP-сервера
180	**Ringing.** Местоположение вызываемого пользователя определено. Вызываемый пользователь получает сигнал о входящем вызове от своего UA
181	**Call Is Being Forwarded.** Прокси-сервер переводит вызов к другому пользователю. Если прокси-сервер передает этот ответ, он может также указывать в теле сообщения, к какому пользователю он переводит вызов
182	**Queued.** Вызываемый пользователь временно недоступен, но входящий вызов поставлен в очередь. Когда вызываемый пользователь станет доступен, он передаст окончательный ответ. Reason-Phrase может дать более детальное описание текущего состояния обслуживания вызова, например, «В очереди находится 5 вызовов. Предполагаемое время ожидания 15 минут». Сервер периодически будет передавать ответы 182 (Queued), чтобы информировать вызывающего пользователя о статусе вызова, находящегося в очереди
183	**Session Progress.** Этот ответ используется для того, чтобы заранее получить описание сеанса информационного обмена от шлюзов на пути к вызываемому пользователю таким образом, чтобы речевой тракт мог быть проключен в предответном состоянии еще до того, как вызывающий пользователь получит сигнал КПВ. Этот ответ используется, например, при взаимодействии протокола SIP с сетью ТфОП, при котором передача ответа *Session Progress* с SDP-описанием шлюза ТфОП позволяет входящей АТС послать сигнал КПВ. Среди других вариантов использования этого ответа – воспроизведение приветственного объявления или музыкальной фразы при входе в домен передустановлением соединения

3.4.2. Ответы об успешной обработке запроса 2xx

Ответы класса **2xx** означают, что запрос был успешно обработан. Варианты ответов приведены в табл. 3.6.

Таблица 3.6. SIP – ответы об успешной обработке запроса

Код	Назначение
200	**OK.** Запрос успешно выполнен Ответ **200** на запрос INVITE означает, что вызываемый пользователь согласен принять участие в сеансе связи, в теле ответа указываются возможности оборудования вызываемого пользователя Ответ **200** на запрос BYE означает завершение сеанса связи, в теле ответа не переносится никакой информации Ответ **200** на запрос CANCEL означает отмену поиска, в теле ответа не переносится никакой информации Ответ **200** на запрос REGISTER означает, что регистрация прошла успешно Ответ **200** на запрос OPTIONS означает согласие вызываемого пользователя сообщить функциональные возможности своего оборудования, которые должны содержаться в теле ответа
202	**Accepted.** Запрос был принят для обработки, но обработка еще не завершена. Неизвестно, будет ли выполнен запрос, поскольку после завершения обработки запрос может быть отклонен Значение ответа 202 намеренно неопределенное. Цель такого ответа – позволить серверу принять запрос для обработки без предъявления требования, чтобы соединение агента пользователя с сервером существовало до завершения обработки запроса. Ответ должен содержать информацию о текущем статусе запроса и предположительное время, когда обработка запроса будет завершена

3.4.3. Ответы о перенаправлении вызова 3xx

Ответы класса **3xx** (табл. 3.7) информируют оборудование вызывающего пользователя о новом местоположении вызываемого пользователя или об альтернативных услугах, с помощью которых может быть обслужен вызов пользователя.

Таблица 3.7. SIP-ответы перенаправления вызова

Код	Назначение
300	**Multiple Choices**. Вызываемый пользователь доступен по нескольким адресам. Эти адреса передаются вызывающему пользователю, и тот может выбрать один из них и направить вызов по этому адресу. Ответ может содержать тело сообщения, включающее в себя список доступных ресурсов с основными характеристиками и с их местоположением. UA выбирает из списка наиболее подходящий вариант при условии, что это разрешено в поле заголовка **Accept**
301	**Moved Permanently**. Вызываемый пользователь больше не находится по указанному в запросе адресу, и вызывающий пользователь должен направлять запросы на новый адрес, указанный в заголовке **Contact** ответа. Возможно получение списка возможных адресов вызываемого пользователя, и вызывающий пользователь может задавать порядок, в котором будут последовательно «обзваниваться» номера из этого списка для установления соединения. Если сервер не может найти в своей памяти никакой информации о местоположении вызываемого пользователя, он передает ответ отказа 404 (Not Found)
302	**Moved Temporarily**. Вызываемый пользователь временно изменил свое местоположение и может быть найден по адресу, указанному в заголовке **Contact** ответа. Может использоваться при ручной переадресации вызова, потому что сам сервер не переадресует вызов. Поле Request-URI переадресованного запроса имеет то же значение, что и заголовок **Contact** ответа. Время действия контактного адреса указывается в поле заголовка **Expires** или в качестве значения параметра «expires» в заголовке **Contact**. Оба прокси-сервера и агенты пользователя могут буферизировать этот URI на время действия. Если время действия не ограничено, то данный адрес предназначен лишь для единственного использования и не должен заноситься в кэш-память для последующих транзакций. Временный URI может измениться раньше, чем закончится время действия контактного адреса
305	**Use Proxy**. Вызываемый пользователь не доступен непосредственно, входящий вызов должен обязательно пройти через прокси-сервер. Вызывающей стороне рекомендуется повторить запрос, передав его через прокси-сервер, адрес которого указан в поле заголовка **Contact**. Этот ответ передает только UAS
380	**Alternative Service**. Запрошенная услуга недоступна, но доступны альтернативные варианты обслуживания, которые описаны в теле сообщения ответа

3.4.4. Ответы об ошибке в запросе 4xx

Ответы класса **4xx** (табл. 3.8) информируют о том, что в запросе обнаружена ошибка. После получения такого ответа пользователь не должен передавать тот же самый запрос без его модификации.

Таблица 3.8 SIP-ответы об ошибке в запросе

Код	Назначение
400	**Bad Request**. В запросе обнаружена синтаксическая ошибка. Это означает, что запрос не понят принимающим узлом. Ответ с этим кодом должен передаваться при обнаружении любой синтаксической ошибки. Reason-Phrase должна характеризовать ошибку более детально, например: «Потеря заголовка **Call-ID**». Дальнейшее поведение системы зависит от конкретной программной реализации SIP
401	**Unauthorized**. Запрос требует проведения процедуры аутентификации пользователя. Этот ответ передает UAS или registrar. Когда получен этот ответ, к формированию сообщений применяются специальные правила
402	**Payment Required**. Требуется предварительная оплата услуг. Ответ с этим кодом резервирован для будущего использования
403	**Forbidden**. Запрещенный запрос – запрос не будет обрабатываться сервером и не должен передаваться повторно. Запрос был понят, но не будет обслужен. Такой ответ может быть получен, к примеру, при попытке дозвониться по номеру, который не принимает звонки с данного номера телефона
404	**Not Found.** Вызываемый пользователь не обнаружен. Сервер не обнаружил вызываемого пользователя в домене, указанном в поле Request-URI. Этот ответ передается, когда вызываемый пользователь либо никогда не существовал, либо данные об этом пользователе были стерты с этого сервера
405	**Method Not Allowed.** Не разрешается передавать запрос этого типа на адрес, указанный в поле Request-URI. Ответ содержит заголовок **Allow** со списком типов запросов, возможных для данного адреса
406	**Not Acceptable.** Вызываемая сторона будет генерировать ответы, которые не будут поняты вызывающей стороной. Форматы передаваемых данных не соответствуют требованиям, предъявленным в заголовке **Accept** запроса, поскольку форматы, указанные в запросе, не были поняты
407	**Proxy Authentication Required.** Перед вызовом вам требуется аутентифицировать себя прокси-серверу
408	**Request Timeout.** Сервер не может передать ответ в течение промежутка времени, специфицированного вызывающим пользователем в заголовке **Expires** запроса. Время от времени эти ответы могут передавать загруженные сетевые серверы. Вызывающий пользователь может заново передать запрос через некоторое время
410	**Gone.** Сервер больше не имеет доступа к запрашиваемому ресурсу и не знает, куда переадресовать запрос. Ответ передается в случае, когда вызываемый пользователь изменил свое местонахождение на длительный срок. Если сервер не знает или не в состоянии определить, как долго адресат будет отсутствовать, он передает ответ с кодом 404 (Not Found)

Код	Назначение
413	**Request Entity Too Large**. Размер запроса слишком велик для обработки на сервере. Сервер может завершить соединение, чтобы прекратить прием такого запроса. Если обслуживание приостановлено временно, сервер добавляет в сообщение заголовок **Retry-After**. В поле заголовка указано время, по истечении которого вызывающий пользователь может предпринять новую попытку
414	**Request-URI Too Long**. У сервера возникли трудности с интерпретацией адреса, указанного в поле Request-URI, поскольку он слишком длинный
415	**Unsupported Media Type**. Сервер не может принять запрос из-за того, что формат содержимого тела сообщения не поддерживается сервером для запроса данного типа. Сервер должен предоставить клиенту список доступных форматов, используя заголовки **Accept**, **Accept-Encoding**, или **Accept-Language**, в зависимости от характера проблемы
416	**Unsupported URI Scheme**. Сервер не может обработать запрос из-за того, что схема URI в поле Request-URI ему непонятна
420	**Bad Extension**. Сервер не понимает расширение протокола SIP, которое содержится в поле заголовка **Proxy-Require** или **Require**. Сервер должен поместить список не поддерживаемых расширений в заголовок Unsupported ответа
421	**Extension Required**. Для правильной обработки запроса UAS вынужден применить определенное расширение, однако оно не указано в поле заголовка **Supported** запроса. Ответы с этим кодом могут содержать заголовок **Require** со списком требуемых расширений. Сервер передает ответ с кодом 421 только в тех случаях, когда не может предоставить вызываемому пользователю обслуживание с приемлемым качеством. В остальных случаях, если необходимые расширения отсутствуют в заголовке **Supported** запроса, сервер вместо передачи ответа с кодом 421 должен обрабатывать запрос, используя стандартные возможности SIP и другие расширения, поддерживаемые клиентом
423	**Interval Too Brief**. Сервер отклоняет запрос, потому что время действия ресурса, слишком короткое. Этот ответ использует registrar, чтобы отклонить сообщение регистрации, для заголовка **Contact** время действия которого очень мало
480	**Temporarily Unavailable**. Соединение с оконечной системой было установлено успешно, но пользователь в данное время недоступен (к примеру, находится вне системы или находится в системе, но в состоянии, препятствующем установлению соединения с вызывающим абонентом, или активизировал опцию «Не беспокоить»). Ответ может информировать о предпочтительном времени для повторного вызова в поле заголовка **Retry-After**. Вызываемый пользователь может быть доступен по другому адресу, не известному серверу. Reason-Phrase формируется агентом пользователя и обозначает более точную причину недоступности пользователя
481	**Call/Transaction Does Not Exist**. Сервер получил запрос, не относящийся к текущему диалогу или транзакции. Запрос отбрасывается
482	**Loop Detected**. Обнаружен замкнутый маршрут передачи запроса. Это тот случай, когда используется поле **Via**, позволяющее выявлять закольцованные маршруты, наличие которых могли пропустить протоколы сетевого и транспортного уровней. Если сервер видит свой собственный адрес в поле **Via** принятого им запроса, относящегося к еще не установленному соединению, это говорит о том, что где-то в сети имеется петля
483	**Too Many Hops**. Запрос на своем пути к вызываемому пользователю прошел через большее количество прокси-серверов, чем разрешено. Сервер получает запрос с нулевым значением заголовка **Max-Forwards**. Это является защитой от длинных и нестабильных маршрутов

Код	Назначение
484	**Address Incomplete**. Принят запрос с неполным адресом в поле Request-URI. Дополнительная информация представлена в комментарии Reason-Phrase. Может использоваться для реализации постепенной передачи дополнительной адресной информации
485	**Ambiguous**. Адрес вызываемого пользователя в поле Request-URI неоднозначен. Сервер может предложить вызывающему пользователю список адресов в поле заголовка **Contact**, по которым можно передать запрос. Обычно вызывающему пользователю предоставляется несколько адресов на выбор. Использование списка адресов может повлечь за собой нежелательные последствия – раскрытие частной информации о местонахождении того или иного пользователя или организации. Должна существовать возможность конфигурировать сервер таким образом, чтобы он передавал ответ с кодом 404 (Not Found) по получении неоднозначного запроса или изымал список адресов в поле **Contact**. Ниже приведен пример ответа на неоднозначный запрос с адресом *sip:anton@niits.ru*: SIP/2.0 485 Ambiguous Contact: Anton Zarubin <*sip:anton-zarubin@niits.ru*> Contact: Anton Ivanov <*sip:anton-i@niits.ru*> Contact: Petrov Anton <*sips: anton-petr@niits.ru*>
486	**Busy Here**. Вызываемый пользователь в данный момент либо не желает, либо не имеет возможности принять данный вызов в дополнение к существующим. В заголовке **Retry-After** ответа может быть указано подходящее для вызова время. Для того чтобы связаться с пользователем, можно применить другие средства, например, воспользоваться услугой речевой почты
487	**Request Terminated**. Запрос был отменен сообщением BYE или CANCEL
488	**Not Acceptable Here**. Соединение с сервером было установлено, но отдельные элементы описания сеанса связи, такие как тип запрашиваемой информации, полоса пропускания, вид адресации недопустимы. Существует возможность связаться с пользователем по другому адресу, или используя прочие средства
489	**Bad Event**. Ответ используется, чтобы указать, что сервер не понял тип event package, указанного в заголовке **Event**
491	**Request Pending**. Запрос поступил на сервер, который к этому времени не закончил обработку другого запроса, относящегося к тому же диалогу
493	**Undecipherable**. Запрос, полученный UAS, содержит зашифрованное MIME-тело сообщения, для которого получатель не в состоянии подобрать подходящий ключ дешифрирования. Ответ может иметь тело сообщение, несущее соответствующий открытый ключ шифрования; при использовании этого ключа клиентом запросы будут без затруднений обрабатываться UA
494	**Security Agreement Required**. Ответ передается сервером при выполнении процедуры выбора механизма обеспечения безопасности. Если запрос клиента, помимо заголовка **Security-Client** со списком механизмов содержит идентификатор option-tag «sec-agree» в заголовке **Supported**, сервер передает ответ с кодом 494. Ответ должен включать в себя заголовок **Security-Server** со списком механизмов обеспечения безопасности, поддерживаемых сервером. Клиент должен выбрать подходящий механизм безопасности и применить его при передаче запроса

3.4.5. Ответы об отказе сервера 5xx

Ответы класса **5xx** информируют о том, что запрос не может быть обработан из-за ошибки сервера.

Таблица 3.9. SIP-ответ об отказе сервера

Код	Назначение
500	**Server Internal Error.** Ошибка сервера. Это может быть аппаратная, программная или любая внутренняя ошибка. Вызывающий пользователь имеет возможность повторить свой запрос через несколько секунд. Если ошибка временная, то сервер отображает в поле заголовка **Retry-After** время, через которое рекомендуется отправить сообщение повторно
501	**Not Implemented.** Сервер не может обслужить запрос, потому что в нем не реализованы соответствующие функции. Этот ответ необходим, когда UAS не в состоянии определить тип запроса и не может принять сообщение
502	**Bad Gateway.** Сервер принял некорректный ответ от шлюза или прокси-сервера на пути к адресату вызова. Это – типичный отказ для соединений, устанавливаемых через многочисленные сетевые сегменты и серверы
503	**Service Unavailable.** Обслуживание временно невозможно вследствие перегрузки или проведения технического обслуживания. Сервер может указать в поле заголовка **Retry-After** время, по истечении которого следует повторить передачу запроса UAC или прокси-сервер, получивший ответ 503, должен попытаться направить запрос по иному пути через другой сервер. Не исключены случаи, когда сервер вместо передачи ответа 503 отказывает в обслуживании или отбрасывает запрос
504	**Server Time-out.** Сервер, функционирующий в качестве шлюза или прокси-сервера, не получил ответа в течение установленного промежутка времени от сервера, к которому он обратился для завершения вызова, например, от сервера определения местоположения пользователей
505	**Version Not Supported.** Сервер не поддерживает или отказывается поддерживать версию протокола SIP, используемую в запросе. При надлежащем обеспечении совместимости снизу вверх эта ошибка маловероятна. Такой ответ могут передавать старые серверы или терминалы, когда они видят более новую версию протокола SIP в заголовке запроса
513	**Message Too Large.** Сервер не в состоянии обработать запрос из-за большой длины сообщения
580	**Precondition Failure.** Когда UAS не может или не желает принимать параметры, предлагаемые в описании сеанса – информации offer, он должен отклонить запрос, передав ответ с кодом 580. При этом информация answer не передается. Этот ответ используется для отказа от предложения установить сеанс связи (offer), содержащегося в запросе INVITE или UPDATE. В ответе содержится SDP-описание, основанное на последней принятой от удаленного агента пользователя информации offer или answer, которое указывает причину отказа. SDP-описание этого ответа не является ни информацией answer, ни offer, поскольку не направлено на установление сеанса

3.4.6. Ответы о полной невозможности установить соединение 6xx

Передаваемый пользователем запрос не может обслужить ни один сервер (табл. 3.10). Соединение с вызываемым пользователем установить невозможно.

Таблица 3.10. SIP-ответы о полной невозможности установить соединение

Код	Назначение
600	**Busy Everywhere.** Вызываемый пользователь занят и не желает принимать вызов в данный момент. Ответ может указывать подходящее для вызова время. Если с пользователем можно связаться по другому адресу или, к примеру, оставить сообщение на речевой почтовый ящик, то используется ответ 486 (Busy Here)
603	**Decline.** Вызываемый пользователь не может или не желает принять входящий вызов без указания причины отказа
604	**Does Not Exist Anywhere.** Вызываемый пользователь не существует.
606	**Not Acceptable.** Соединение с сервером было установлено, но отдельные элементы описания сеанса связи, такие как тип запрашиваемой информации, полоса пропускания, вид адресации не допустимы. Ответ может содержать заголовок **Warning** с указанием причин невозможности установить сеанс связи

Глава 4. Процедуры управления соединением

4.1. Диалоги

Диалог представляет собой равноправное взаимодействие двух агентов пользователя по протоколу SIP, которое длится определенное время. Диалог устанавливает последовательность сообщений между UA и обеспечивает верную маршрутизацию запросов. В этом разделе будет рассмотрен процесс использования запросов и ответов для организации диалога и их пересылки в режиме диалога.

Диалог идентифицируется каждым агентом пользователя с помощью идентификатора диалога (dialog ID), который состоит из значения Call-ID, локальной метки (local tag) и удаленной метки (remote tag). У участвующих в диалоге сторон локальная метка одного UA идентична удаленной метке другого и наоборот. Идентификатор диалога непосредственно связан со всеми запросами, имеющими параметр «tag» в поле заголовка **To**.

Правила вычисления идентификатора диалога зависят от того, чем является элемент сети SIP–клиентом или сервером агента пользователя. Для UAC значение Call-ID устанавливается по заголовку **Call-ID** сообщения, удаленная метка – по параметру «tag» в поле заголовка **To**, а локальная метка по параметру «tag» в поле заголовка **From**. Для UAS, соответственно, значение Call-ID копируется из заголовка **Call-ID** сообщения, удаленная метка определяется по параметру «tag» в заголовке **From**, а локальная метка – по параметру «tag» в заголовке **To**.

Диалог включает в себя ряд компонентов, которые описывают его состояние и используются для передачи сообщений в ходе диалога. Это – идентификатор диалога, локальный порядковый номер (для упорядочения запросов UA, направляемых собеседнику), удаленный порядковый номер (для упорядочения запросов от собеседника к UA), локальный URI, удаленный URI, текущий адрес удаленного пользователя (remote target), булев флаг «secure» и конфигурация маршрута (route set), представляющая собой упорядоченный список URI. Конфигурация маршрута – это перечень серверов, через которые должен пройти запрос, отправленный второму участнику диалога.

Диалог может находиться на так называемой «ранней стадии», которая имеет место при создании диалога в результате передачи предварительного ответа; только после получения клиентом окончательного ответа, диалог переходит в установленное состояние. Если помимо предварительных ответов 2xx никаких ответов не приходит, диалог, находящийся на «ранней стадии», завершается.

4.2. Процедура создания диалога

Диалоги создаются путем передачи не информирующих об ошибках ответов на запросы определенных типов. В данной версии протокола установить диалог можно передачей на запрос INVITE только ответов 101–199 и 2xx с параметром «tag» в поле заголовка **To**. Диалог, установленный предварительным ответом на запрос, называется диалогом «на ранней стадии» (early dialog).

4.2.1. Действия UAS

Когда UAS, получив запрос, передает ответ, инициирующий создание диалога (как, например, ответ класса 2xx на запрос INVITE), он должен скопировать в ответ содержимое заголовка **Record-Route** запроса (включая адреса, параметры адресов и все параметры заголовка, вне зависимости от того, известны они серверу или нет) с сохранением порядка следования этих величин. UAS должен также поместить в ответ заголовок **Contact**; в этом заголовке указывается предпочтительный адрес для последующих запросов диалога. Обычно компонент URI, идентифицирующий хост, является его IP-адресом или FQDN. URI, представленный в поле заголовка **Contact**, может быть либо SIP URI, либо SIPS URI.

Если запрос, инициировавший диалог, содержал SIPS URI в поле Request-URI или в качестве первого значения заголовка **Record-Route** или заголовка **Contact** (в случае отсутствия Record-Route), то контактный адрес в ответе также должен быть SIPS URI. Адрес должен быть уникальным для того, чтобы тот же URI мог быть использован в сообщениях вне диалога. Так, например, URI в заголовке **Contact** сообщения INVITE может впоследствии использоваться для передачи сообщений вызывающему пользователю.

Далее UAS формирует состояние диалога, которое должно поддерживаться на протяжении всего диалога.

Если запрос пришел через TLS, и поле Request-URI содержит SIPS URI, флаг «secure» устанавливается в состояние «TRUE».

Конфигурация маршрута (route set) определяется из перечня URI в заголовке **Record-Route** запроса в установленном порядке и с сохранением всех параметров каждого URI. Когда заголовок **Record-Route** отсутствует в сообщении, маршрут устанавливается в «пустое» состояние. Даже если маршрут пуст, он аннулирует для последующих запросов текущего диалога маршруты, определенные ранее.

Текущий адрес удаленного пользователя (remote target) определяется по URI в поле заголовка **Contact** запроса.

Удаленный порядковый номер устанавливается в соответствии с порядковым номером в заголовке **CSeq** запроса. Локальный порядковый номер должен быть пуст.

Call-ID идентификатора диалога копируется из заголовка **Call-ID** запроса. Локальная метка (local tag) идентификатора определяется по параметру «tag» в заголовке **To** (который всегда содержит этот параметр) ответа на запрос, а удаленная метка (remote tag) – по параметру «tag» в заголовке **From** запроса. UAS должен быть готов получить запрос без параметра «tag» в заголовке **From**; в таком случае метке присваивается значение 0.

Удаленный URI переносится из заголовка **From**, а локальный URI – из заголовка **To**.

4.2.2. Действия UAC

Когда UAC передает запрос, который может привести к созданию диалога (например, INVITE), он должен обеспечить наличие уникального SIP или SIPS URI в поле заголовка **Contact**. Если в запросе значение поля Request-URI или верхнее значение заголовка **Route** есть SIPS URI, в поле заголовка **Contact** также записывается SIPS URI.

Когда UAC получает ответ, устанавливающий диалог, он приступает к формированию состояния диалога.

Если запрос был передан через TLS, и поле Request-URI содержит SIPS URI, флаг «secure» устанавливается в состояние «TRUE».

Конфигурация маршрута определяется из перечня URI в заголовке **Record-Route** ответа, взятых в обратном порядке и с сохранением всех параметров каждого URI. Когда заголовок **Record-Route** отсутствует в сообщении, маршрут переходит в «пустое» состояние. Даже если маршрут пуст, он аннулирует для последующих запросов текущего диалога маршруты, определенные ранее.

Текущий адрес удаленного пользователя (remote target) определяется по URI в поле заголовка **Contact** ответа. Локальный порядковый номер устанавливается в соответствии с порядковым номером в заголовке **CSeq** ответа. Удаленный порядковый номер должен быть пуст.

Call-ID идентификатора диалога копируется из заголовка **Call-ID** запроса. Локальная метка идентификатора определяется по параметру «tag» в заголовке **From** запроса, а удаленная метка – по параметру «tag» в заголовке **To** ответа. UAC должен быть готов получить ответ без параметра «tag» в заголовке **To**; в таком случае метке присваивается значение 0. Удаленный URI переносится из заголовка **To**, а локальный URI – из заголовка **From**.

4.3. Процедура передачи и приема запросов в ходе диалога

Как только соединение между двумя агентами пользователя установлено, любой из них может стать инициатором новых транзакций, необходимых в рамках диалога. UA, передающий запросы, будет выполнять в транзакции роль кли-

ента, а UA, принимающий запросы, будет выполнять роль сервера. Заметим, что эти роли могут отличаться от тех, которые исполняли агенты пользователя в транзакции создания диалога.

Запросы в режиме диалога могут включать в себя заголовки **Record-Route** и **Contact**. Эти запросы не могут привести к изменению маршрута (route set), однако в состоянии модифицировать текущий адрес удаленного пользователя (remote target). Для изменения remote target применяются определенные типы запросов, которые используют для этой цели вышеупомянутые заголовки. Они носят название запросов, обновляющих текущий адрес удаленного пользователя (target refresh requests). Такими запросами являются re-INVITE и UPDATE.

4.3.1. Действия UAC при создании запроса

Запрос в состоянии диалога формируется с использованием большинства компонентов, описывающих состояние диалога. URI в заголовке **To** запроса устанавливается в соответствии с удаленным URI диалога, параметр «tag» в заголовке **To** – по удаленной метке идентификатора диалога. URI в заголовке **From** запроса устанавливается в соответствии с локальным URI состояния диалога, параметр «tag» в заголовке **From** – по локальной метке идентификатора диалога. Если удаленная или локальная метка имеет нулевое значение, параметр «tag» должен быть исключен из состава заголовка **To** или **From**, соответственно.

Заголовок **Call-ID** запроса должен быть составлен в соответствии с Call-ID диалога. Запросы в режиме диалога имеют монотонно возрастающие порядковые номера (пошагово увеличивающиеся на единицу), которые содержатся в поле заголовка **Cseq**, исключая запросы ACK и CANCEL, чьи номера совпадают с номерами подтверждаемого или отменяемого запроса. Таким образом, если локальный порядковый номер присутствует, его значение увеличивается на единицу, и результат этой операции помещается в заголовок **CSeq** создаваемого запроса. Если же локальный порядковый номер отсутствует, то выбирается начальный порядковый номер.

Содержимое поля типа запроса в заголовке **CSeq** должно соответствовать типу запроса. Верхняя граница порядкового номера – 2^{32}, этого хватит на то, чтобы на протяжении 136 лет клиент в рамках одного сеанса генерировал запросы со скоростью один запрос в секунду. Начальный порядковый номер выбирается таким образом, чтобы порядковые номера следующих запросов текущего сеанса не

перекрылись с данным. Ненулевое начальное значение порядкового номера дает возможность клиенту привязать его к текущему времени. Например, клиент может выбрать в качестве начального порядкового номера значение, отражающее 31 наиболее значимых битов 32-битового секундного счетчика времени.

UAC использует remote target и route set для формирования поля Request-URI и заголовка **Route** запроса. В случае отсутствия конфигурации маршрута UAC копирует remote target в поле Request-URI, заголовок **Route** не добавляется. Если же конфигурация маршрута содержит список URI, и первый из них содержит параметр «lr», UAC также помещает значение remote target в поле Request-URI и включает в состав запроса заголовок **Route**, содержащий конфигурацию маршрута в прямом порядке и со всеми параметрами. Когда конфигурация маршрута содержит список URI, и первый из них не имеет параметра «lr», UAC помещает первый URI маршрута в поле Request-URI, исключая любые запрещенные для этого поля параметры. Клиент должен добавить заголовок **Route**, содержащий оставшуюся часть маршрута. Далее значение remote target копируется в заголовок **Route** в качестве последнего значения.

Например, если значение remote target – *sip:user@remoteua,* и route set содержит:

```
<sip:proxy1>,<sip:proxy2>,<sip:proxy3;lr>,<sip:proxy4>,
```

запрос будет включать в себя поле Request-URI и заголовок **Route** в следующем виде:

```
Request-URI sip:proxy1
Route:<sip:proxy2>,<sip:proxy3;lr>,<sip:proxy4>,<sip:user@remoteua>
```

Если первый URI в списке, определяющем маршрут, не имеет параметра «lr», это означает, что первый прокси-сервер не поддерживает стандартного механизма маршрутизации и заменяет Request-URI первым значением в заголовке **Route** сообщения. Такой прокси-сервер называется strict router. Предварительное перемещение Request-URI в конец заголовка **Route** сохраняет информацию, содержащуюся в этом поле, при прохождении через strict router.

UAC должен включить заголовок **Contact** в состав любого запроса, обновляющего текущий адрес удаленного пользователя в режиме диалога, и в случае необходимости изменить значение remote target – таким образом UA может изменить контактный адрес в ходе диалога. В случае, когда флаг «secure» установлен в состояние «TRUE», адрес должен быть SIPS URI.

Как только сформирован запрос, определяется адрес сервера и происходит передача запроса с использованием тех же процедур, что и при работе вне диалога (см. раздел 2.2.1). Итогом выполнения этих процедур обычно является передача запроса на адрес, обозначенный первым в заголовке **Route** или на адрес, указанный в поле Request-URI, если заголовок **Route** отсутствует.

4.3.2. Действия UAC при обработке ответов

UAC получает ответы на запросы от уровня транзакций. Если клиентская транзакция информирует об истечении времени ожидания ответа, это уведомление обрабатывается как ответ с кодом 408 (Request Timeout). Поведение UAC при получении ответов 3xx в режиме диалога то же, что и вне диалога, как это описано в § 2.2.2. Когда UAC получает ответ класса 2xx на запрос, обновляющий remote target, он должен заменить свой remote target значением URI в заголовке **Contact** ответа. Если на запрос приходит ответ с кодом 481 (Call/Transaction Does Not Exist) или 408 (Request Timeout), UAC завершает диалог. UAC также должен завершить диалог, если на запрос не пришло никакого ответа.

4.3.3. Действия UAS

UAS получает запросы от уровня транзакций. Если в заголовке **To** запроса имеется параметр «tag», UAS вычисляет идентификатор диалога, соответствующий запросу, и сравнивает его с идентификаторами существующих диалогов. При совпадении UAS использует основные правила обработки запросов, которые применяются как при работе в диалоге, так и вне диалога, как это описано в § 2.3.1. Если в заголовке **To** запроса имеется параметр «tag», но идентификатор диалога, соответствующий запросу, не совпадает ни с одним из существующих идентификаторов, значит UAS вышел из строя и перезагрузился или, возможно, он получил запрос, предназначенный для другого UAS. Другая причина – неправильная маршрутизация входящего запроса. UAS, основываясь на параметре «tag» заголовка **To**, может принять или отклонить запрос. Использование механизма, основанного на параметре «tag» заголовка **To**, обеспечивает высокую надежность, которая предотвращает разрыв диалога даже при возникновении неполадок оборудования. В режиме диалога могут быть получены запросы, не изменяющие состояния диалога (например, запрос OPTION). Эти запросы обрабатываются так же, как если бы они были переданы вне диалога. Если UAS не содержит удаленного порядкового номера, то этот номер должен быть установлен в соответствии с порядковым

номером в поле заголовка **CSeq** запроса. Если же удаленный порядковый номер имеется, но его величина больше порядкового номера в запросе, это значит, что запрос содержит ошибку; запрос отклоняется, и передается ответ с кодом 500 (Server Internal Error). При величине удаленного порядкового номера, меньшей порядкового номера в **Cseq**, запрос принимается. Это не является ошибкой, и UAS должен быть готов принимать и обрабатывать запросы с заголовком **CSeq**, значение которого превосходит более чем на единицу значение соответствующего заголовка в предыдущем запросе. В такой ситуации удаленный порядковый номер принимает значение порядкового номера в **CSeq** запроса.

Прокси-сервер может потребовать аутентификации при получении запроса, сформированного клиентом. При этом UAC включает в сообщение отклик аутентификации, и запрос передается снова. У этого запроса будет уже другой номер в **CSeq**. UAS никогда не узнает о первом запросе и поэтому обнаружит разрыв в последовательности номеров. Такой разрыв не расценивается как ошибка.

Когда UAS получает запрос, обновляющий текущий адрес удаленного пользователя, он заменяет значение remote target диалога значением URI в поле заголовка **Contact** запроса.

4.3.4. Процедура завершения диалога

Когда на любой запрос вне диалога приходит окончательный ответ класса, отличного от 2xx, находящиеся на «ранней стадии» диалоги, созданные посредством предварительных ответов, разрушаются.

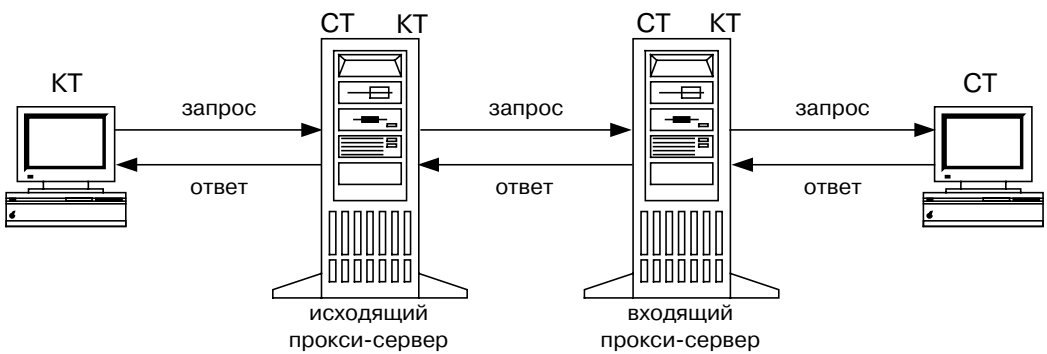
Для разрушения установленных диалогов используется запрос BYE.

4.4. Транзакции

SIP – это протокол, ориентированный на транзакции: взаимодействие между элементами сети происходит путем периодического обмена сообщениями. Транзакция состоит из запроса и любого количества ответов на него (ни одного или один и более предварительных ответов и один и более окончательных ответов). В транзакцию запроса INVITE (называемую INVITE-транзакцией) входит запрос ACK в случае, если окончательный ответ был класса, отличного от 2xx;

в противном случае ACK не является частью INVITE-транзакции. Причиной этого разделения является важность доставки UAC всех ответов с кодом 200 (OK) на запрос INVITE. Чтобы доставить клиенту все ответы с кодом 200, ядро UAS берет на себя ответственность за их повторную передачу, а ядро UAC, соответственно, берет на себя ответственность за их подтверждение запросами ACK.

Транзакция имеет клиентскую сторону и серверную сторону; соответственно, они носят название клиентской транзакции и серверной транзакции. Клиентская транзакция занимается передачей запросов, а серверная транзакция – передачей ответов. Они создаются в агентах пользователя и прокси-серверах с сохранением состояний (stateful). На рис. 4.1, представленном ниже, UAC выполняет клиентскую транзакцию, а его исходящий прокси-сервер выполняет серверную транзакцию. Исходящий сервер выполняет также клиентскую транзакцию, которая передает запрос серверной транзакции входящего прокси-сервера. Входящий прокси-сервер выполняет клиентскую транзакцию, которая, в свою очередь, передает запрос серверной транзакции UAS.



СТ - серверная транзакция
КТ - клиентская транзакция

Рис. 4.1. Взаимодействие клиентских и серверных транзакций

Прокси-сервер без сохранения состояний не создает ни клиентских, ни серверных транзакций. Транзакция выполняется между UA или прокси-сервером с сохранением состояний с одной стороны и UA или прокси-сервером с сохранением состояний с другой стороны. Цель клиентской транзакции состоит в том, чтобы получить запрос от ядра клиента (этот элемент носит название пользователь

транзакции – Transaction User (TU); это может быть ядро UA или ядро прокси-сервера с сохранением состояния) и надежно доставить запрос серверной транзакции. Клиентская транзакция также отвечает за получение ответов и доставку их TU, отфильтровывая ответы, переданные повторно, и запрещенные ответы (такие как ответ на запрос ACK). Кроме того, в случае с запросом INVITE в полномочия клиентской транзакции входит передача запроса ACK, подтверждающего любой окончательный ответ класса, отличного от 2xx. Подобным образом, цель серверной транзакции – принимать запросы от транспортного уровня протокола SIP и передавать их TU. Серверная транзакция отфильтровывает повторные запросы, переданные из сети. Серверная транзакция принимает ответы от TU и передает их на транспортный уровень SIP для пересылки по сети; в случае INVITE-транзакции, она принимает запрос ACK на любой окончательный ответ, исключая ответ класса 2xx. Ответ класса 2xx и его подтверждение ACK имеют особое значение. Ответ 2xx может передаваться повторно только сервером UA, а запрос ACK формируется только клиентом агента пользователя. Сквозное обращение требуется для того, чтобы вызывающий пользователь знал число пользователей, принявших вызов. Поэтому передача повторных ответов 2xx, как и генерация запроса ACK, это прерогатива ядра UA, а не уровня транзакций. Прокси-сервер только пересылает каждый ответ класса 2xx на запрос INVITE и соответствующий ему ACK.

4.4.1. Процедуры функционирования клиентских транзакций

Пользователь транзакции (TU) взаимодействует с клиентской транзакцией следующим образом. Когда TU нужно инициировать новую транзакцию, он создает клиентскую транзакцию и передает ей SIP-запрос, предназначенный для отправки, IP-адрес, номер порта и тип транспортного протокола, на которые должен быть передан запрос.

Существует два конечных автомата клиентских транзакций, в зависимости от типа запроса, который передает TU. Один из них поддерживает клиентские транзакции для запросов INVITE (клиентская INVITE-транзакция). Второй поддерживает клиентские транзакции запросов всех типов, исключая INVITE и ACK (клиентская не-INVITE-транзакция). Не существует отдельной клиентской транзакции для запроса ACK: когда у TU возникает необходимость передать этот запрос (при подтверждении ответа класса 2xx на запрос INVITE), он отправляет его непосредственно транспортному уровню SIP для доставки по сети.

INVITE-транзакция отличается от транзакций запросов других типов своей большей продолжительностью: в случае успешного установления сеанса окончательный ответ (класса 2xx) может быть передан только после приема вызова пользователем. Длительные задержки в процессе передачи ответа говорят о протекающей процедуре трехэтапного согласования (three-way handshake). Запросы других типов обрабатываются быстро. TU незамедлительно отвечают на не-INVITE-запросы, поскольку очевидно, что согласование должно быть двухэтапным.

4.4.2. Клиентская INVITE-транзакция

INVITE-транзакция реализуется в процессе трехэтапного согласования. Клиентская транзакция передает запрос, серверная транзакция передает ответ, а затем клиентская транзакция передает подтверждение ACK. В случае использования ненадежного транспортного протокола (такого как UDP) клиентская транзакция повторно передает запросы через отрезок времени T1, который удваивается после каждой повторной передачи. T1 – это оценка периода кругового обращения, т.е. времени, необходимого на передачу и на подтверждение приема запроса (RTT); по умолчанию значение T1 – 500 мс. Практически все транзакционные таймеры связаны с T1, их настройку можно произвести, изменяя значение T1. При надежном транспортном протоколе запрос не требует повторной передачи. После получения информационного ответа (класса 1xx) все повторные запросы прекращаются, клиент пребывает в ожидании дальнейших ответов. Серверная транзакция может передать дополнительные ответы 1xx , которые передаются ненадежно (без подтверждения). В итоге серверной транзакцией передается окончательный ответ. При использовании надежного транспортного протокола он передается однократно, при использовании ненадежного – периодически повторяется. Для каждого полученного окончательного ответа клиентская транзакция передает ACK, назначение которого – предотвратить повторную передачу ответа.

4.4.3. Конечный автомат клиентской INVITE-транзакции

Конечный автомат (машина состояний) клиентской INVITE-транзакции показан на рис. 4.2. Начальное состояние «Calling» наступает, когда TU создает новую клиентскую транзакцию по запросу INVITE. Клиентская транзакция должна передать запрос на транспортный уровень SIP для его дальнейшей транспортировки по сети.

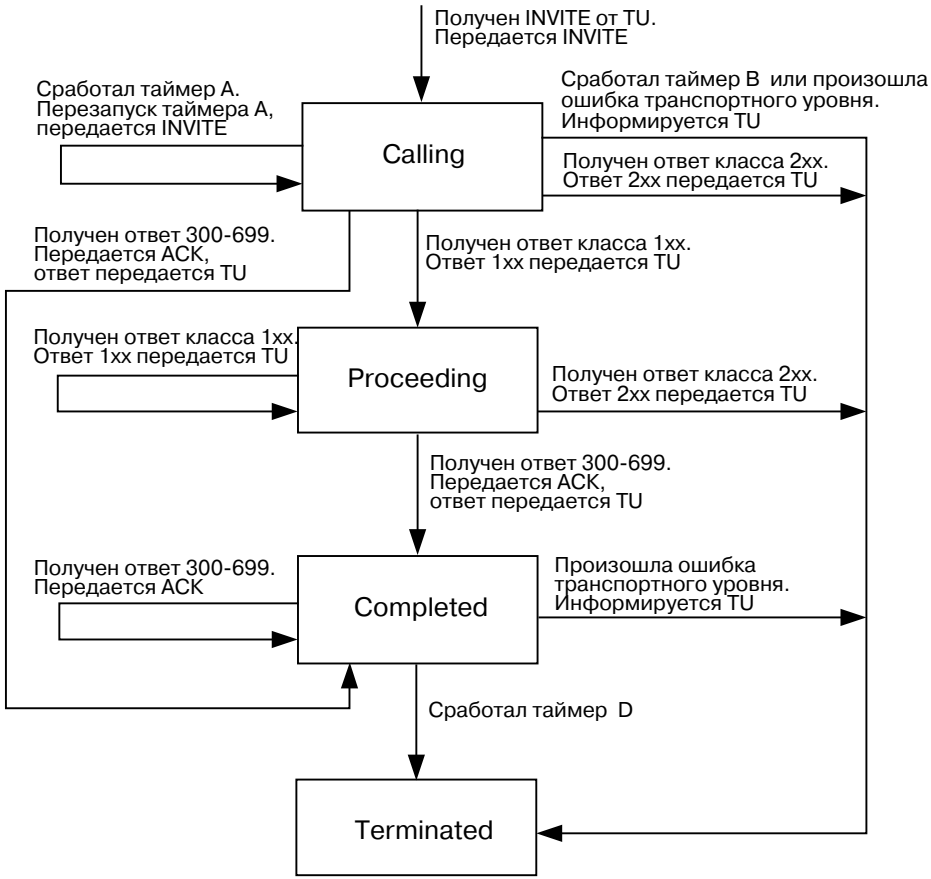


Рис. 4.2. Клиентская INVITE-транзакция

Если при этом используется ненадежный транспортный протокол, клиентская транзакция запускает таймер A со значением T1. В противном случае таймер A не используется – он предназначен для контроля времени повторной передачи запросов). Вне зависимости от используемого транспортного протокола должен быть также запрещен таймер B со значением 64 T1 (он ограничивает время ожидания окончательного ответа клиентской INVITE-транзакции).

Когда таймер A срабатывает, клиентская транзакция должна повторно передать запрос, отправив его на транспортный уровень SIP, и заново запустить таймер, но уже со значением 2 T1. Этот процесс продолжается, и запрос передается повторно через интервал времени, удваивающийся после каждой передачи. Повторения передачи запроса возможны только в тот отрезок времени, когда клиентская транзакция находится в состоянии «Calling».

Как уже упоминалось, рекомендуемое значение T1 – 500 мс. T1 – это оценка RTT между клиентской и серверной транзакциями. Использование SIP-элементами величин меньше T1 не рекомендуется, однако в некоторых случаях это может оказаться целесообразным. T1 может быть выбрано больше значения по умолчанию, если заранее известно, что RTT (round-trip time) больше. Если клиентская транзакция находится в состоянии «Calling», то когда срабатывает таймер B, она должна информировать TU об истечении времени ожидания. Значение 64 T1 выбрано с учетом возможности передачи семи запросов за этот интервал времени при использовании ненадежного транспортного протокола. Если клиентская транзакция, находясь в состоянии «Calling», получает предварительный ответ, она переходит в состояние «Proceeding». В этом состоянии клиентской транзакции запрещается дальнейшая передача повторных запросов. Кроме того, предварительный ответ должен быть направлен TU. Ему отправляются все предварительные ответы, пришедшие за время пребывания в состоянии «Proceeding».

Прием ответов с кодами 300 – 699 в двух вышеописанных состояниях приводит к переходу в состояние «Completed». Клиентская транзакция предоставляет TU полученный ответ, формирует подтверждение ACK и передает его на транспортный уровень SIP. ACK должен использовать тот же адрес, номер порта и тип транспортного протокола, что и оригинальный запрос. При переходе в состояние «Completed» запускается таймер D со значением, по меньшей мере, 32 секунды для ненадежного транспортного протокола и 0 для надежного. Он отражает, сколько времени серверная транзакция может оставаться в состоянии «Completed», когда используется ненадежный транспортный протокол. Таймер D идентичен таймеру H в серверной INVITE-транзакции, значение которого по умолчанию равно 64 T1. Однако клиентская транзакция не располагает сведениями о значении T1, используемом серверной транзакцией, поэтому значение таймера D принято равным 32 с.

На все повторные окончательные ответы, пришедшие во время пребывания в состоянии «Completed», должен быть повторно передан запрос ACK, однако эти ответы не передаются TU.

При срабатывании таймера D клиентская транзакция переходит в состояние «Terminated». Получение ответа класса 2xx в состояниях «Calling» и «Proceeding» также приводит к переходу клиентской транзакции в состояние «Terminated», принятый ответ направляется для обработки к TU. Обращение с ответом зависит от того, чем является TU: программным ядром прокси-сервера или ядром клиента агента пользователя. Ядро UAC сформирует запрос ACK на этот ответ, а ядро прокси-сервера будет пересылать ответ 200 (OK) по сети.

В момент перехода в состояние «Terminated» клиентская транзакция должна быть разрушена. Это необходимо для того, чтобы гарантировать правильность работы. Дело в том, что ответы 2xx на INVITE трактуются по-разному. Каждый ответ класса 2xx должен быть передан ядру прокси-сервера (для того, чтобы быть пересланным) и ядру UAC (для того, чтобы быть подтвержденным). Уровень транзакций не принимает участия в обработке. Всякий раз, когда транспортный уровень SIP получает ответ и не находит при этом соответствующей клиентской транзакции, ответ поступает прямо к TU. А так как клиентская транзакция разрушается первым ответом класса 2xx, последующие ответы 2xx будут непосредственно передаваться ядру.

4.4.4. Формирование запроса ACK

Далее рассматривается конструкция запросов ACK, передаваемых в ходе клиентской транзакции. Ядро UAC, генерирующее ACK на ответ класса 2xx, должно следовать описанным в главе 2 правилам, определенным для запроса данного типа. Запрос ACK, формируемый клиентской транзакцией, должен содержать значения заголовков **Call-ID**, **From** и поля Request-URI, идентичные тем, что были переданы в составе оригинального запроса – запроса, для которого создавалась данная клиентская транзакция (в данном случае – для запроса INVITE). Значение заголовка **To** копируется из аналогичного заголовка подтверждаемого ответа и, следовательно, всегда будет отличаться от оригинального запроса дополнительным параметром «tag». В ACK должен присутствовать единственный заголовок **Via**, равный верхнему заголовку **Via** в оригинальном запросе. Поле заголовка **CSeq** содержит тот же порядковый номер, что и оригинальный запрос, но поле типа запроса имеет значение ACK. В случае, если запрос INVITE, на который получен требующий подтверждения ответ, содержал поля заголовка **Route**, они также должны быть и в запросе ACK. Это гарантирует, что ACK будет правильно маршрутизироваться при прохождении через прокси-серверы без сохранения состояний (stateless).

Как и другие запросы, ACK может содержать тело сообщения. Однако существуют трудности, связанные с тем, что запрос ACK не может быть отклонен в случае, если тело сообщение не понято. Поэтому не рекомендуется помещать тело сообщения в запрос ACK, который подтверждает ответ, отличный от класса 2xx, но если это происходит, типы тела должны соответствовать типам, указанным в запросе INVITE, при условии, что ответ был не с кодом 415. В случае, если на оригинальный запрос пришел ответ с кодом 415, тело в сообщении ACK должно придерживаться типов, перечисленных в заголовке **Accept** ответа. Пример:

Оригинальный запрос

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKkjshdyff
To: Vladimir <sip:vladimir@protei.ru>
From: Anton <sip:anton@niits.ru>;tag=88sja8x
Max-Forwards: 70
Call-ID: 987asjd97y7atg
CSeq: 986759 INVITE
```

Запрос ACK, подтверждающий прием окончательного ответа, отличного от класса 2xx

```
ACK sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKkjshdyff
To: Vladimir <sip:vladimir@protei.ru>;tag=99sa0xk
From: Anton <sip:anton@niits.ru>;tag=88sja8x
Max-Forwards: 70
Call-ID: 987asjd97y7atg
CSeq: 986759 ACK
```

4.4.5. Клиентская не-INVITE-транзакция

Клиентские не-INVITE-транзакции не используют запрос ACK. Они строятся по схеме запрос – ответ. При использовании ненадежного транспортного протокола запросы передаются повторно через интервал T1, который последовательно удваивается пока не достигнет значения T2. Если приходит предварительный ответ, повторная передача продолжается, но не выходит за временные рамки T2. Серверная транзакция начинает повторять последний переданный предварительный/окончательный ответ только после получения повторно переданного запроса. Поэтому необходимо продолжать повторную передачу запроса после получения предварительного ответа, это гарантирует надежную доставку окончательного ответа.

4.4.6. Конечный автомат клиентской не-INVITE-транзакции

Конечный автомат (машина состояний) клиентской не-INVITE-транзакции показан на рис. 4.3. Состояние «Trying» наступает, когда TU передачей запроса инициирует создание клиентской транзакции. В тот же момент запускается таймер F со значением 64 T1. Запрос должен быть передан на транспортный уровень SIP для передачи по сети.

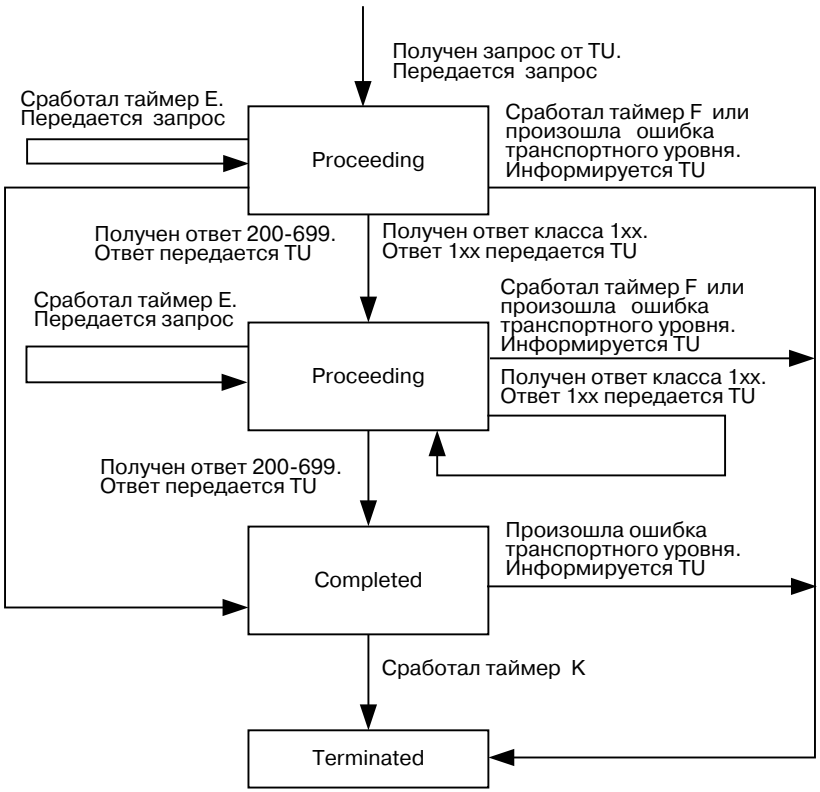


Рис. 4.3. Клиентская не-INVITE-транзакция

При использовании ненадежного транспортного протокола запускается таймер E на время T1. Если этот таймер срабатывает, когда транзакция еще не перешла в другое состояние, устанавливается новая величина таймера, равная MIN (2 T1, T2). После очередного срабатывания значение таймера становится MIN (4 T1, T2). Этот процесс продолжается так, что повторения передачи происходят с экспоненциально увеличивающимся интервалом времени; возрастание интервала прекращается при достижении им величины T2. Значение T2 по умолчанию – 4 секунды. В течение этого времени серверная не-INVITE-транзакция может формировать ответ на запрос, если ответ не был передан немедленно. Для значений T1 и T2 по умолчанию это выражается следующим образом: 500 мс, 1 с, 2 с, 4 с, 4 с, 4 с и т.д.

В случае срабатывания таймера F клиентская транзакция должна информировать TU об истечении времени ожидания и перейти в состояние «Terminated». При получении предварительного ответа в состоянии «Trying», он передается TU, а затем клиентская транзакция переходит в состояние «Proceeding». При получении окончательного ответа (ответы 200 – 699) в состоянии «Trying», он также передается TU, а клиентская транзакция переходит в состояние «Completed». Когда в состоянии «Proceeding» срабатывает таймер E, запрос поступает на транспортный уровень SIP, а таймер принимает значение T2. В случае срабатывания в состоянии «Proceeding» таймера F клиентская транзакция должна информировать TU об истечении времени ожидания и перейти в состояние «Terminated».

При получении окончательного ответа (ответы 200 – 699) в состоянии «Proceeding», он также передается TU, и клиентская транзакция переходит в состояние «Completed». При переходе клиентской транзакции в состояние «Completed», запускается таймер K со значением T4 при использовании ненадежного транспортного протокола и со значением 0 – при использовании надежного. Состояние «Completed» предназначено для буферизации дополнительных повторных ответов, которые могут придти, а потому оно используется только при ненадежном транспортном протоколе. T4 представляет интервал времени, необходимый сети для завершения передачи сообщений между клиентской и серверной транзакциями. Значение T4 по умолчанию – 5 секунд. Ответ будет считаться повторно переданным, если он относится к той же транзакции. Когда срабатывает таймер K, клиентская транзакция переходит в состояние «Terminated». Как только клиентская транзакция оказывается в этом состоянии, она тут же разрушается.

4.4.7. Соответствие ответов клиентским транзакциям

Когда транспортный уровень клиента принимает ответ, он должен выяснить, какой клиентской транзакции ответ принадлежит. Это необходимо для того, чтобы вступили в силу процедуры, описанные выше. Для этой цели существует параметр «branch» в верхнем заголовке **Via**. Ответ соответствует клиентской транзакции при выполнении двух условий:

- Верхний заголовок **Via** ответа имеет то же значение параметра «branch», что и аналогичный верхний заголовок запроса, инициировавшего транзакцию.
- Поле типа запроса в заголовке **CSeq** соответствует типу запроса, создавшего транзакцию. Это условие необходимо, поскольку запрос CANCEL составляет новую транзакцию, но при этом содержит тот же параметр «branch», т.е. выполняет первое условие.

Если запрос передается в режиме многоадресной рассылки, то возможна доставка нескольких ответов с разных серверов. Все эти ответы будут иметь одинаковый параметр «branch» в верхнем заголовке **Via**, но будут различаться параметром «tag» в заголовке **To**. Первый из них, который будет получен, подвергнется обработке, а остальные будут рассматриваться, как повторные. Когда клиентская транзакция передает запрос транспортному уровню SIP для доставки, и если при этом транспортный уровень SIP сообщает об ошибке, то должны выполняться следующие процедуры. Клиентская транзакция информирует TU о транспортной ошибке и переходит в состояние «Terminated».

4.4.8. Процедуры функционирования серверных транзакций

Серверная транзакция отвечает за доставку запросов TU и надежную передачу ответов. Это достигается с помощью механизма функционирования серверной транзакции. Серверные транзакции создаются при условии, что создание транзакции желательно для запроса. Как и в клиентских транзакциях, механизм функционирования зависит от типа запроса.

4.4.9. Серверная INVITE-транзакция

Конечный автомат состояний серверной INVITE-транзакции показан на рис. 4.4.

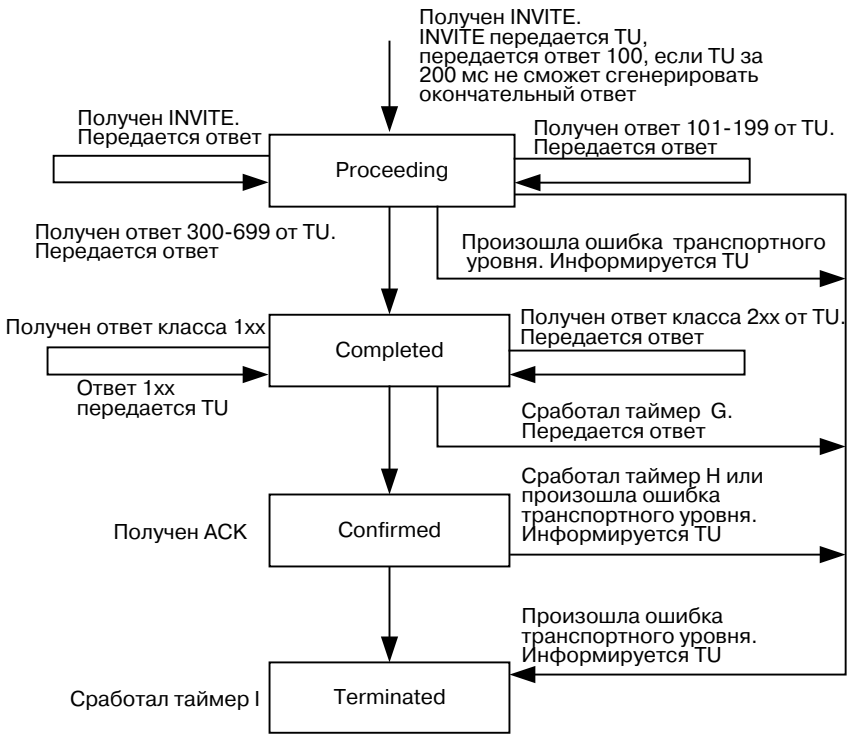


Рис. 4.4. Серверная INVITE-транзакция

Когда приходит запрос, клиентская транзакция создается и переходит в состояние «Proceeding». Серверная транзакция должна передать ответ с кодом 100 (Trying), если не обладает сведениями о том, что TU сформирует предварительный или окончательный ответ в течение 200 мс. Этот предварительный ответ нужен для того, чтобы быстро прекратить повторную передачу запросов и, тем самым, предотвратить перегрузки сети. Полученный запрос должен быть передан TU.

TU может передать серверной транзакции любое число предварительных ответов. Пока серверная транзакция находится в состоянии «Proceeding», она должна направлять все эти ответы на транспортный уровень SIP. Т.о., серверная транзакция не занимается повторной передачей предварительных ответов, они передаются ненадежно без ее участия. Поэтому их передача не вызывает изменения состояния серверной транзакции. Если приходит повторный запрос, серверная транзакция направляет на транспортный уровень SIP последний полученный от TU предварительный ответ. Запрос расценивается как повторно переданный, если выполняются условия его соответствия серверной транзакции, которые будут приведены ниже. В случае, когда TU в состоянии «Proceeding» передает серверной транзакции ответ класса 2xx, серверная транзакция должна, опять же, отправить его на транспортный уровень SIP для передачи по сети. Серверная транзакция не занимается повторной передачей ответов 2xx, это – прерогатива TU. После этого серверная транзакция переходит в состояние «Terminated».

При получении от TU ответа с кодом 300 – 699 серверная транзакция передает его транспортному уровню SIP и переходит в состояние «Completed». При использовании ненадежного транспорта запускается таймер G со значением T1.

После того как серверная транзакция перешла в состояние «Completed», запускается таймер H со значением 64 T1 для любого транспорта. Таймер H определяет, когда серверная транзакция должна прекратить повторную передачу ответов. Его значение выбирается равным значению таймера B, определяющего время, в течение которого клиентская транзакция будет повторно передавать запросы. Если срабатывает таймер G, ответ отправляется к транспортному уровню SIP для повторной передачи, а таймер G принимает новое значение MIN (2 T1, T2). При очередном срабатывании таймера следуют аналогичные действия, и значение таймера G удваивается; это происходит до тех пор, пока оно не достигает T2; после этого значение таймера всегда остается равным T2. Кроме того, если в состоянии «Completed» приходит повторный запрос, серверу следует отправить ответ на транспортный уровень SIP для передачи. Когда приходит запрос ACK, серверная транзакция должна перейти в состояние «Confirmed». Поскольку в этом состоянии таймер G игнорируется, повторение передачи будет прекращено.

Срабатывание таймера H в состоянии «Completed» говорит о том, что запрос ACK не был получен; серверная транзакция переходит в состояние «Terminated» и информирует TU об ошибке транзакции.

Назначение состояния «Confirmed» – принимать дополнительные запросы ACK, являющиеся реакцией на повторно переданные окончательные ответы. Как только это состояние наступает, таймер I запускается со значением T4 для случая с ненадежным транспортным протоколом и со значением 0 для случая с надежным транспортом. При срабатывании таймера I серверная транзакция переходит в состояние «Terminated».

В состоянии «Terminated» серверная транзакция немедленно разрушается. Как и при клиентских транзакциях, это нужно для того, чтобы обеспечить надежность передачи ответов класса 2xx на запрос INVITE.

4.4.10. Серверная не-INVITE-транзакция

Конечный автомат (машина состояний) серверной не-INVITE-транзакции показан на рис. 4.5. При поступлении запроса (исключая INVITE и ACK) серверная транзакция входит в состояние «Trying». Запрос передается TU, а все повторные запросы отбрасываются (запрос считается повторным, если относится к той же серверной транзакции). Если TU передает серверной транзакции предварительный ответ, она переходит в состояние «Proceeding». Ответ должен быть направлен транспортному уровню SIP для передачи по сети. Те же действия предпринимаются при поступлении дальнейших предварительных ответов от TU. При поступлении повторного запроса в состоянии «Proceeding» серверная транзакция передает на транспортный уровень SIP предварительный ответ, получаемый последним.

Транзакция переходит в состояние «Completed» при поступлении от TU окончательного ответа (ответы 200 – 699), ответ передается на транспортный уровень SIP. При переходе в состояние «Completed» запускается таймер J со значением 64 T1 секунд для ненадежного транспортного протокола и 0 секунд – для надежного. В этом состоянии серверная транзакция передает окончательный ответ на транспортный уровень SIP для повторной передачи, когда бы ни пришел повторный запрос.

Любые другие окончательные ответы, поступающие от TU в состоянии «Completed», отклоняются серверной транзакцией. Транзакция остается в этом состоянии до тех пор, пока не сработает таймер J, после чего серверная транзакция переходит в состояние «Terminated». В этом состоянии серверная транзакция прекращает свое существование.

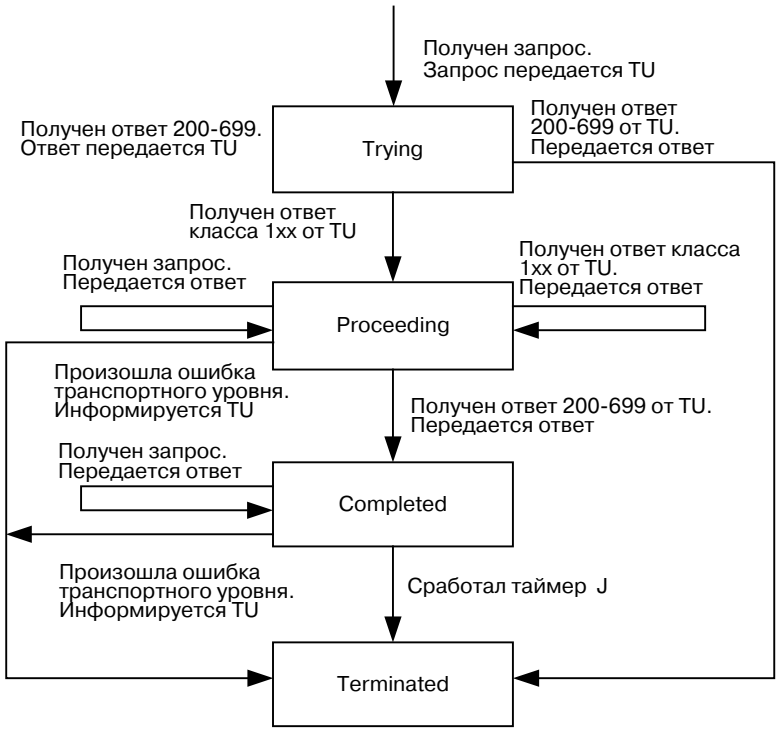


Рис. 4.5. Серверная не-INVITE-транзакция

4.4.11. Соответствие запросов серверным транзакциям

Когда из сети приходит запрос, сервер должен установить принадлежность запроса серверной транзакции. Это проводится следующим образом. Проверяется параметр «branch» первого заголовка **Via**. Если он присутствует, и его значение начинается с комбинации «z9hG4bK» (magic cookie), то запрос был отправлен серверной транзакцией, функционирующей по общим правилам протокола SIP v. 2.0. Следовательно, параметр «branch» уникален для каждой транзакции, в которой принимает участие клиент.

Запрос соответствует транзакции, когда:

- параметр «branch» запроса совпадает с аналогичным параметром в первом заголовке **Via** запроса, создавшего транзакцию;
- имя хоста и номер порта в первом значении заголовка **Via** совпадают с аналогичными в первом значении заголовка **Via** запроса, создавшего транзакцию;
- тип запроса соответствует типу запроса, создавшего транзакцию, за исключением запроса ACK, для которого тип запроса, создавшего транзакцию, будет INVITE;

Это правило распространяется и на INVITE-транзакции, и на не-INVITE-транзакции.

Сравнение значений имени хоста и номера порта в заголовках **Via** используется в процессе сопоставления в связи с возможностью возникновения случайного или злонамеренного дублирования параметра «branch»: запросы с одинаковым параметром могут придти от разных клиентов.

Если в заголовке **Via** запроса параметр «branch» отсутствует или не начинается с «magic cookie», вышеописанные процедуры также выполняются; это делается для того, чтобы обеспечить согласование с более ранними рекомендациями.

Запрос INVITE соответствует транзакции в случае, если его поле Request-URI, параметры «tag» в заголовках **To** и **From**, заголовки **Call-ID**, **CSeq** и первый заголовок **Via** идентичны аналогичным составляющим запроса INVITE, инициировавшего транзакцию. В этом случае имеет место повторная передача запроса INVITE.

Запрос ACK соответствует транзакции, когда поле Request-URI, параметр «tag» заголовка **From**, заголовок **Call-ID**, порядковый номер в заголовке **CSeq** и первый заголовок **Via** идентичны аналогичным составляющим запроса INVITE, инициировавшего транзакцию, и параметр «tag» заголовка **To** совладает с параметром «tag» заголовка **To** в ответе, переданном серверной транзакцией. Добавление параметра «tag» заголовка **To** в процесс сопоставления помогает прокси-серверу отличить ACK, являющегося реакцией на ответ класса 2xx, от подтверждения ACK, являющегося реакцией на другие ответы.

Для сообщений всех остальных типов, запрос соответствует транзакции, когда его поле Request-URI, параметры «tag» в заголовках **To** и **From**, заголовки **Call-ID**, **CSeq** и первый заголовок **Via** идентичны аналогичным составляющим запроса, инициировавшего транзакцию. В этом случае имеет место повторная передача не-INVITE-запроса.

Таблица 4.1. Таймеры в протоколе SIP

Таймер	Величина	Назначение
T1	500 мс (по умолчанию)	RTT (время двойного оборота по сети)
T2	4 с	Максимальный интервал между повторными не-INVITE-запросами и ответами на INVITE
T4	5 с	Максимальное время, в течение которого сообщение будет оставаться в сети
Таймер A	Начальная величина = T1	Время передачи повторного запроса INVITE (только при использовании UDP)
Таймер B	64*T1	Время ожидания окончательного ответа INVITE-транзакцией
Таймер D	> 3 мин	Время ожидания повторных ответов
Таймер E	32 с для UDP 0 с для TCP/SCTP	Время передачи повторного не-INVIITE-запроса (только при использовании UDP)
Таймер F	64*T1	Время ожидания окончательного ответа не-INVITE транзакцией
Таймер G	Начальная величина = T1	Время передачи повторного ответа на запрос INVITE
Таймер H	64*T1	Время ожидания подтверждения ACK
Таймер I	T4 для UDP 0 с для TCP/SCTP	Время ожидания повторных подтверждений ACK
Таймер J	64*T1 для UDP 0 с для TCP/SCTP	Время ожидания повторных не-INVITE-запросов
Таймер K	T4 для UDP 0 с для TCP/SCTP	Время ожидания повторных ответов

4.5. SDL-диаграммы для конечных автоматов транзакций

4.5.1. Клиентская INVITE-транзакция

На рис. 4.6. – 4.9. представлены SDL-диаграммы переходов для конечных автоматов клиентских INVITE-транзакций.

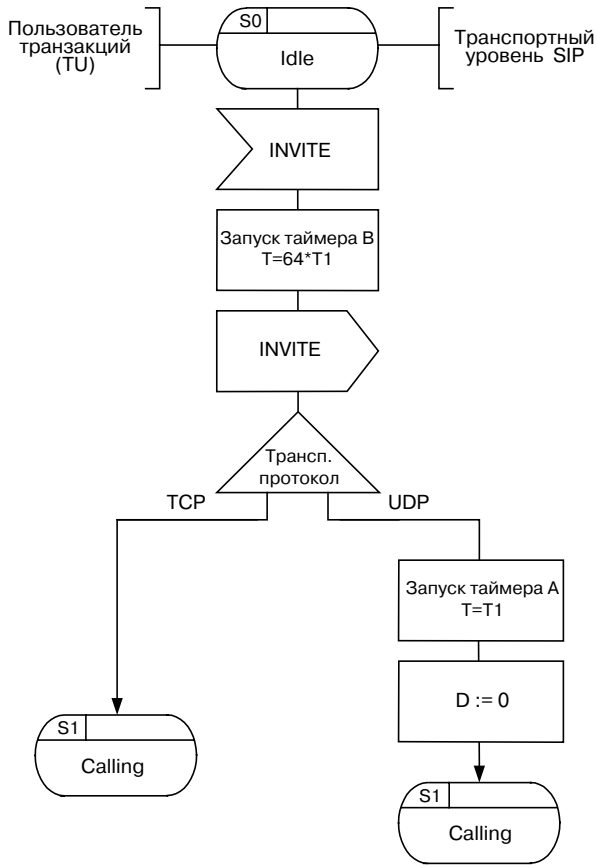


Рис. 4.6. Переход в состояние Calling

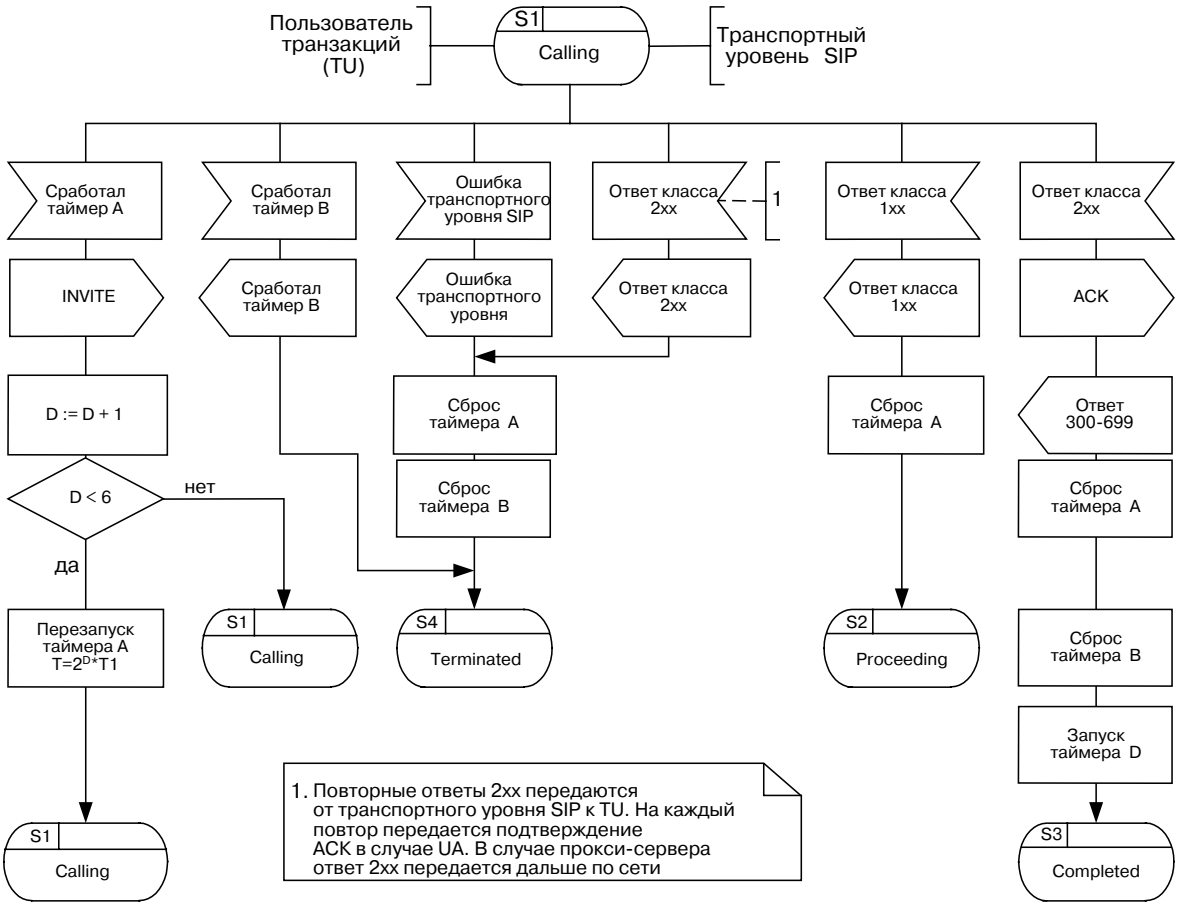


Рис. 4.7. Переход в состояние Proceeding

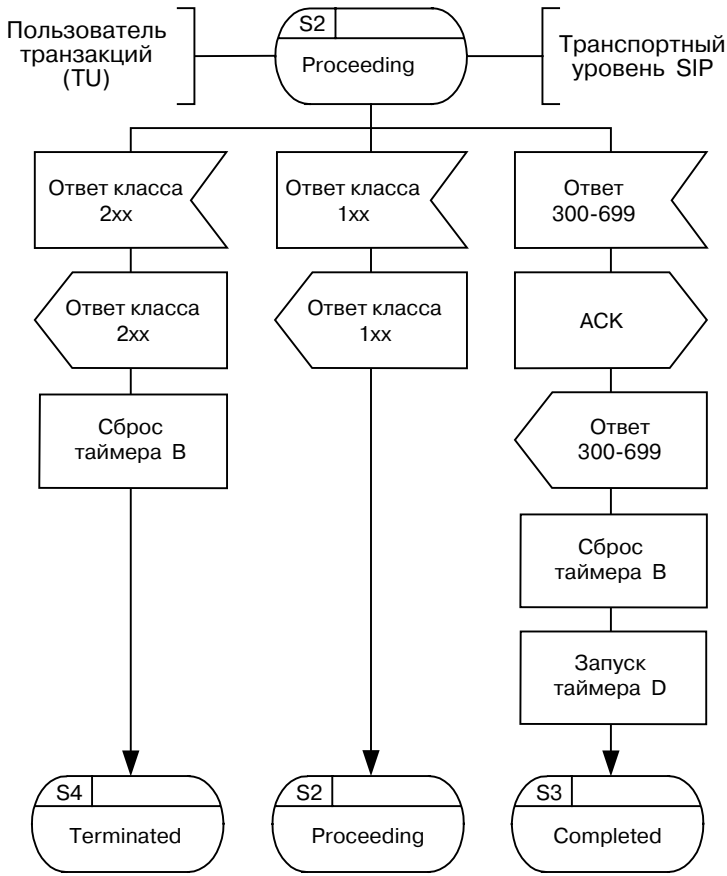


Рис. 4.8. Переход в состояние Completed

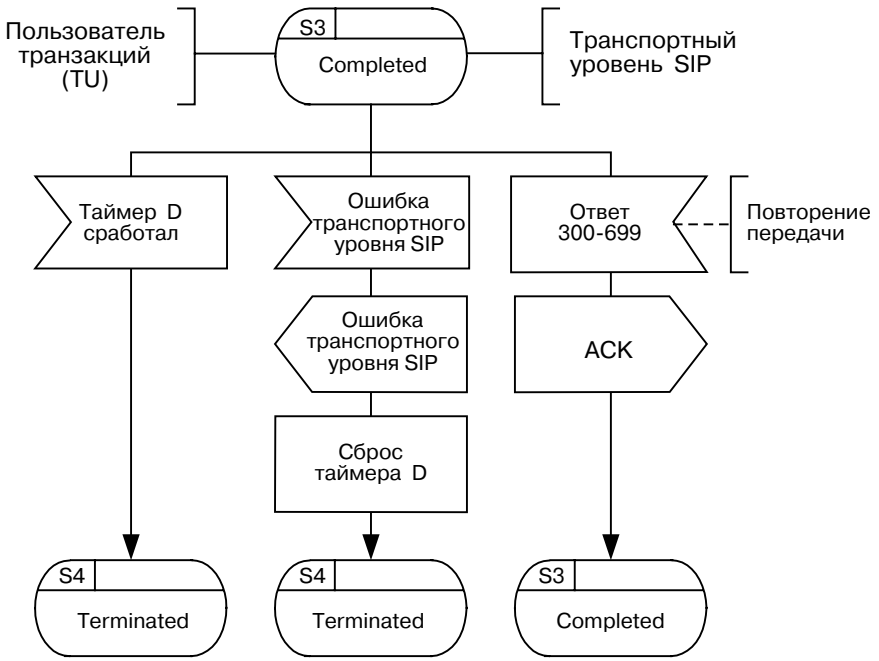


Рис. 4.9. Переход в состояние Terminated

Состояния:

S0 – «Idle» – исходное состояние – транзакции не существует.

S1 – «Calling» – вызывное состояние – периодически передается запрос INVITE, ожидание ответов.

S2 – «Proceeding» – состояние приема ответов – получен ответ 1xx, повторения передачи INVITE прекращены, ожидание окончательного ответа.

S3 – «Completed» – состояние окончания обработки сообщений – ожидание повторных окончательных ответов 300 – 699, передача подтверждений ACK.

S4 – «Terminated» – состояние разрушения транзакции – транзакция разрушена.

4.5.2. Клиентская не-INVITE-транзакция

На рис. 4.10. – 4.13. представлены SDL-диаграммы переходов для конечных автоматов клиентских не-INVITE-транзакций.

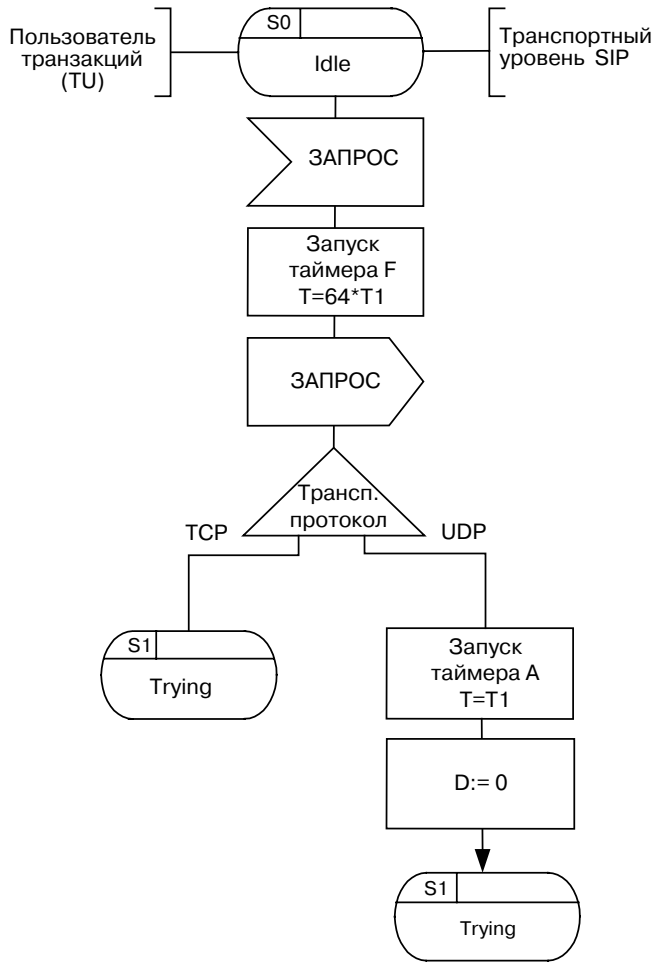


Рис. 4.10. Переход в состояние Trying

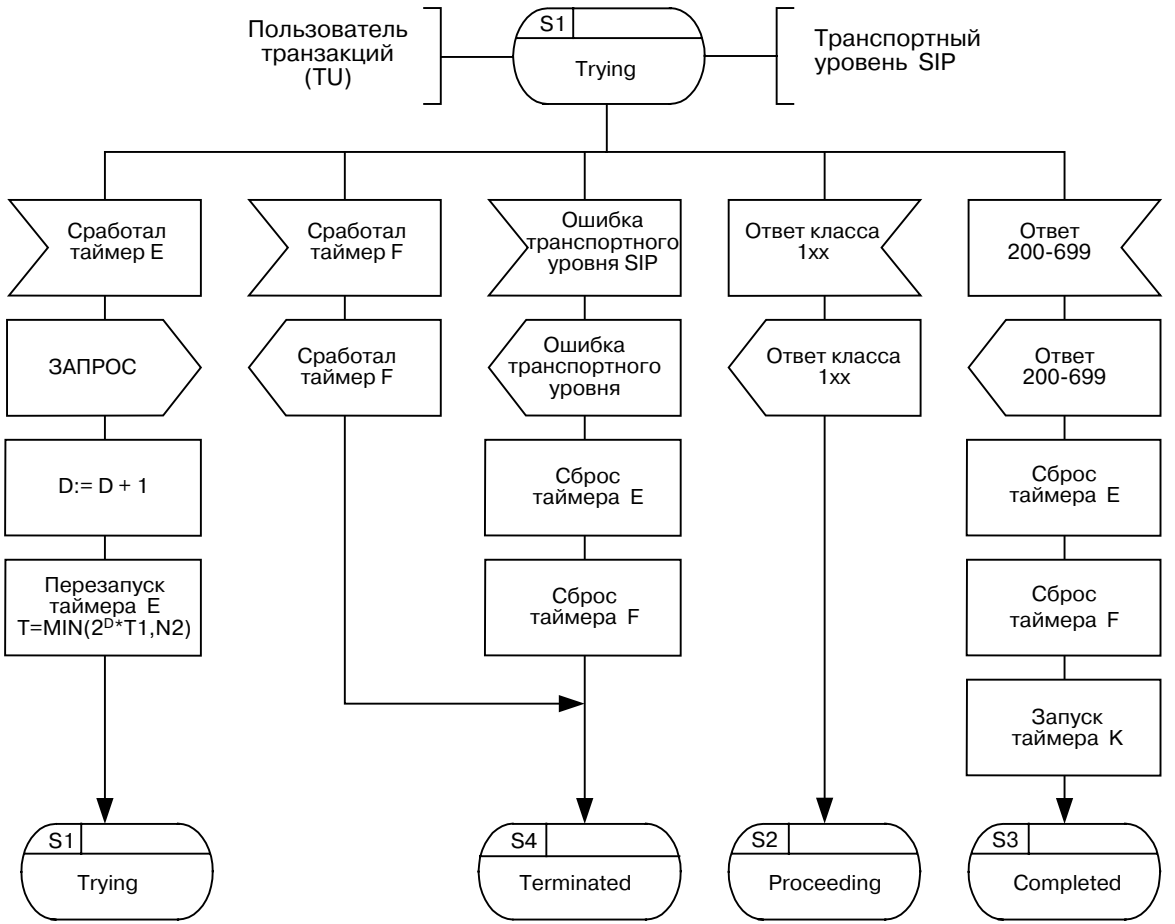


Рис. 4.11. Переход в состояние Proceeding

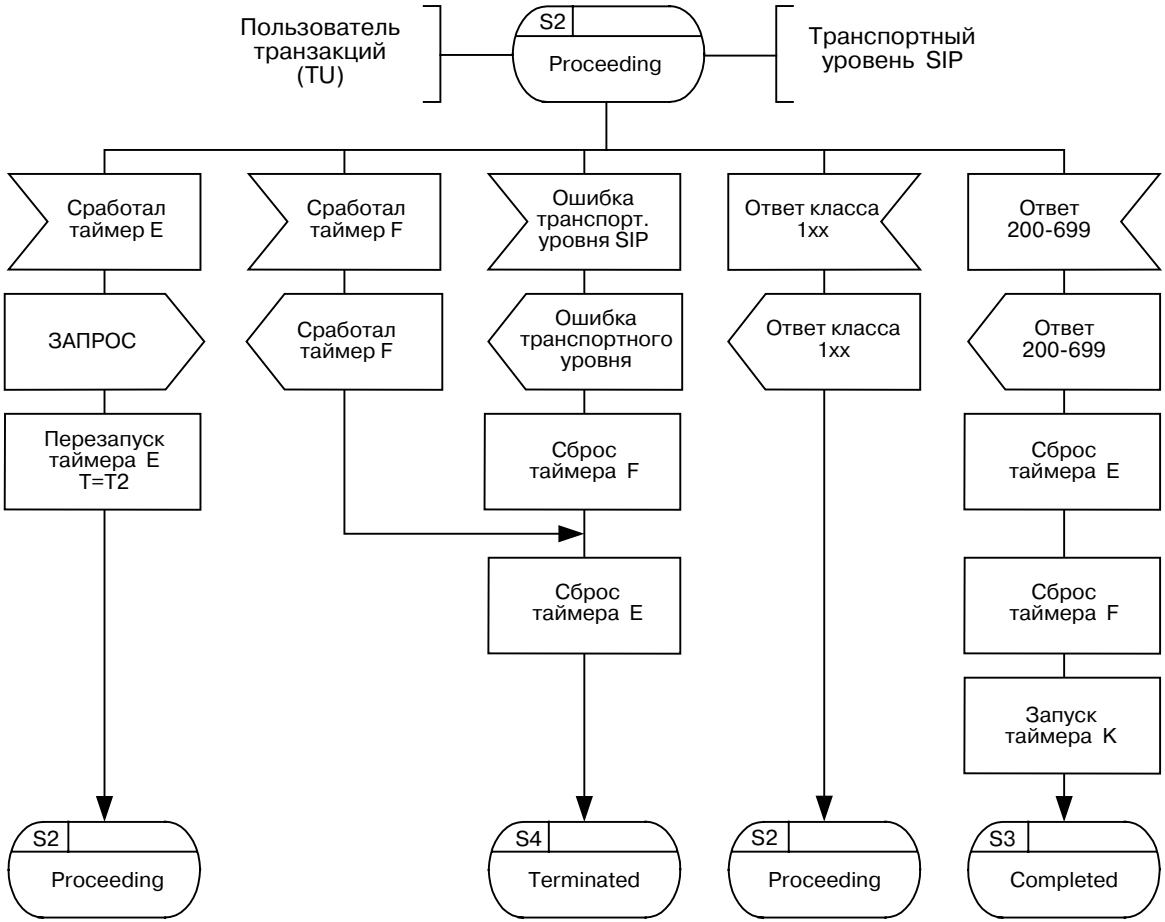


Рис. 4.12. Переход в состояние Completed

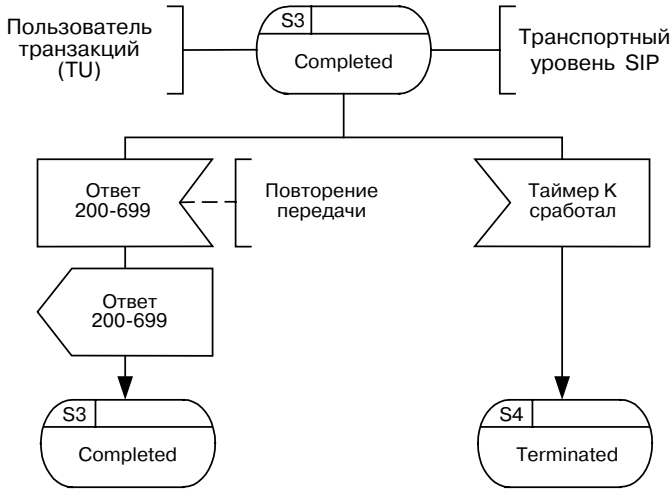


Рис. 4.13. Переход в состояние Terminated

Состояния:

S0 – «Idle» – исходное состояние – транзакции не существует.

S1 – «Trying» – состояние передачи запроса – периодически передается запрос, ожидание ответов.

S2 – «Proceeding» – состояние приема ответов – получен ответ 1xx, продолжается повторная передача запроса, ожидание окончательного ответа.

S3 – «Completed» – состояние окончания обработки сообщений – ожидание повторных окончательных ответов 200 – 699.

S4 – «Terminated» – состояние разрушения транзакции – транзакция разрушена.

4.5.3. Серверная INVITE-транзакция

На рис. 4.14. – 4.16. представлены SDL-диаграммы переходов для конечных автоматов серверных INVITE-транзакций.

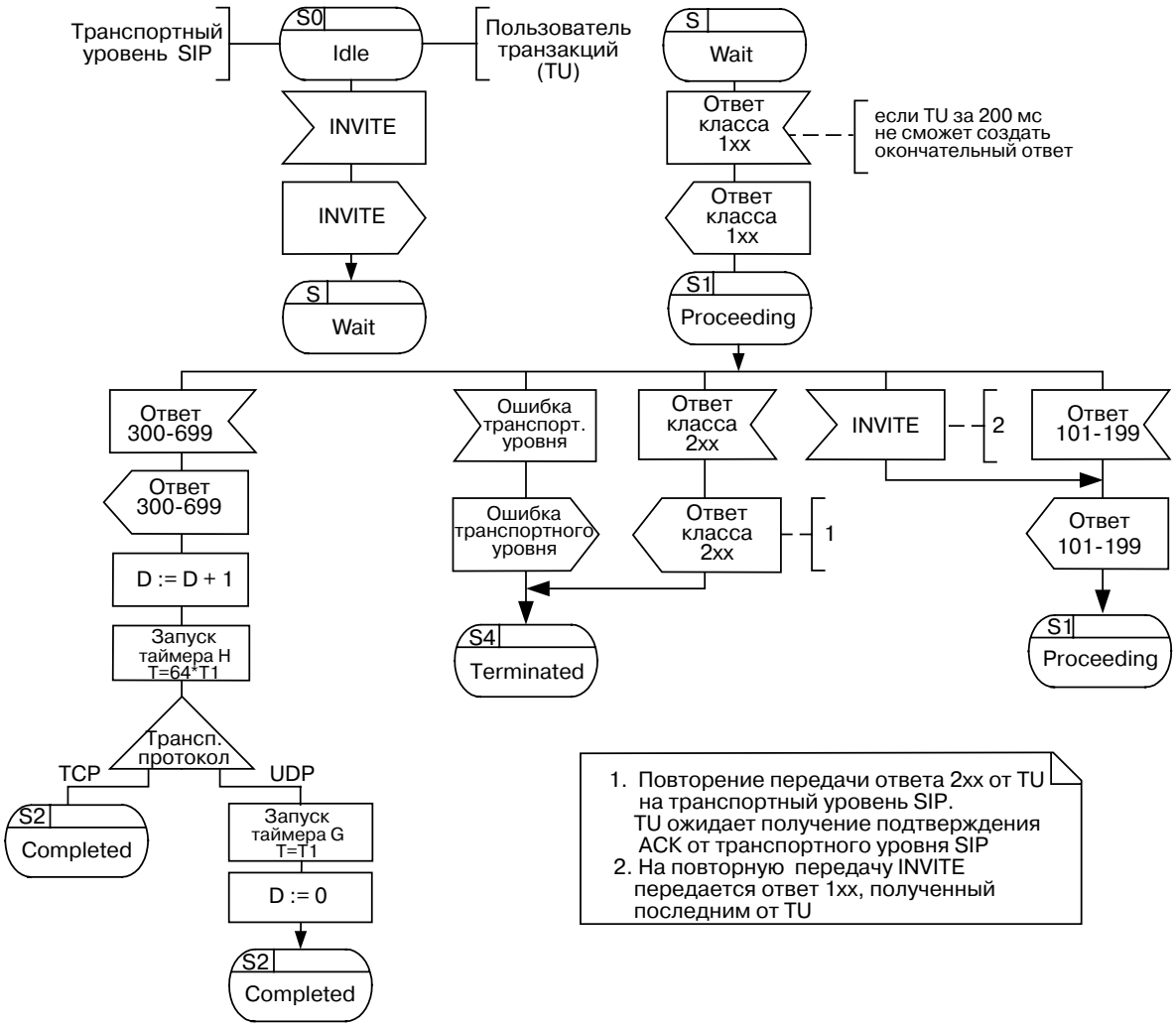


Рис. 4.14. Переход в состояние Proceeding

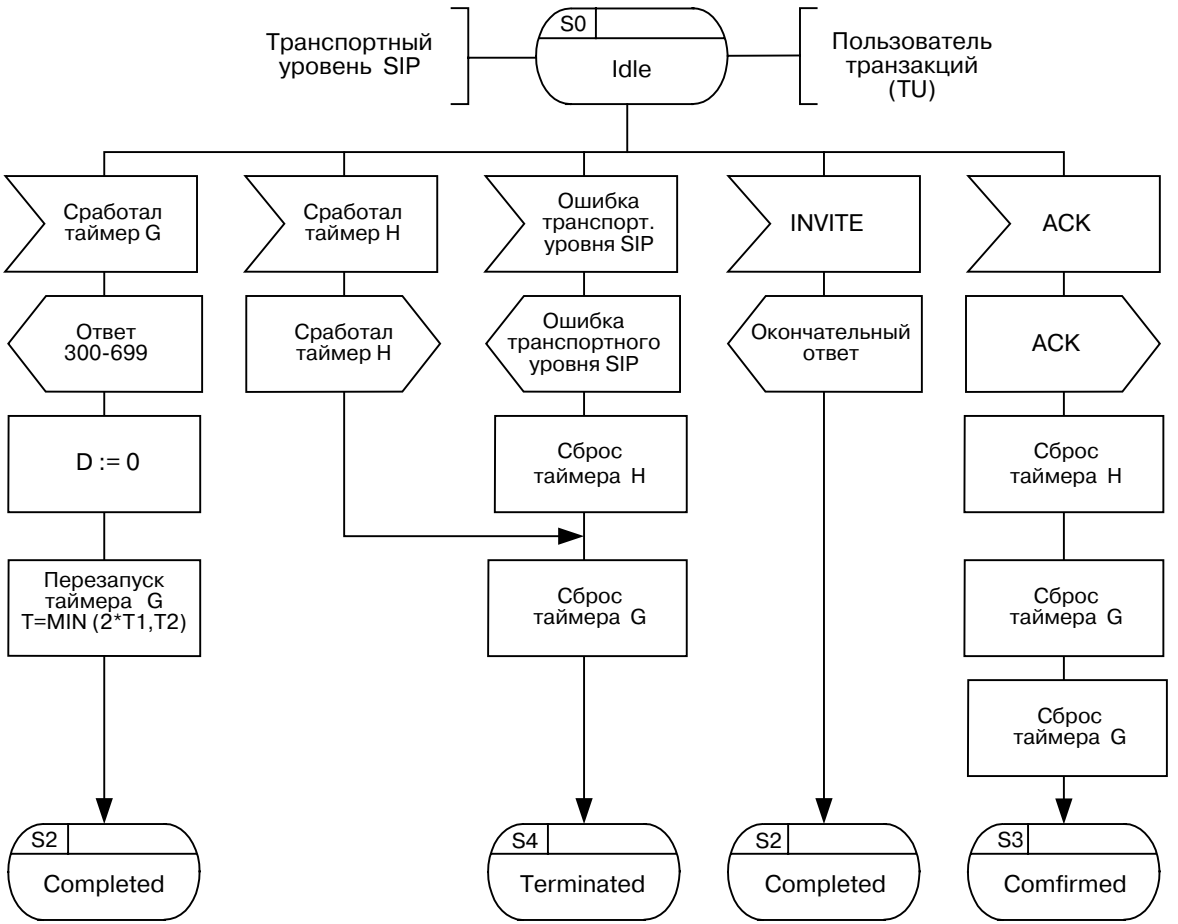


Рис. 4.15. Переход в состояние Completed

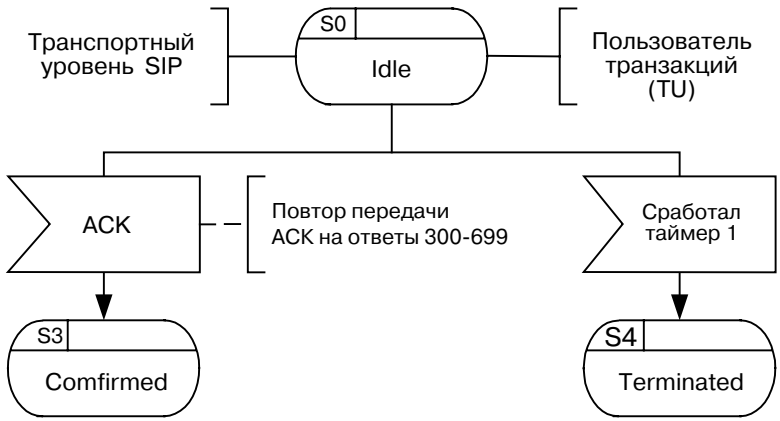


Рис. 4.16. Переход в состояние Confirmed

Состояния:

S0 – «Idle» – исходное состояние – транзакции не существует.

S1 – «Proceeding» – состояние обработки запроса – получен запрос INVITE, после его обработки формируется и передается окончательный ответ.

S2 – «Completed» – состояние окончания обработки сообщений – повторная передача ответов 300 – 699, ожидание подтверждения ACK.

S3 – «Confirmed» – подтвержденное состояние – прием повторений передачи ACK на ответы 300 – 699.

S4 – «Terminated» – состояние разрушения транзакции – транзакция разрушена.

4.5.4. Серверная не-INVITE-транзакция

На рис. 4.17 – 4.19 представлены SDL-диаграммы переходов для конечных автоматов серверных не-INVITE-транзакций.

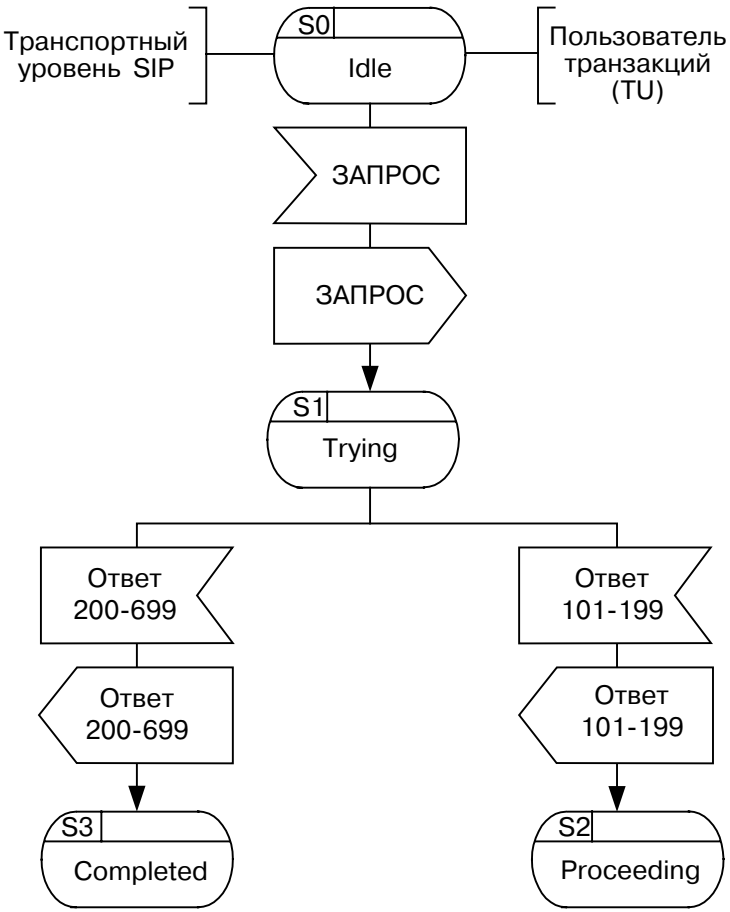


Рис. 4.17. Переход в состояние Proceeding

Состояния:

S0 – «Idle» – исходное состояние – транзакции не существует.

S1 – «Trying» – состояние обработки запроса – получен запрос, после его обработки формируется и передается окончательный ответ.

S2 – «Proceeding» – состояние ожидания окончательного ответа – от TU получен ответ 1xx, ожидание окончательного ответа.

S3 – «Completed» – состояние окончания обработки сообщений – прием повторных передач запросов, передача ответов 200 – 699 на них.

S4 – «Terminated» – состояние разрушения транзакции – транзакция разрушена.

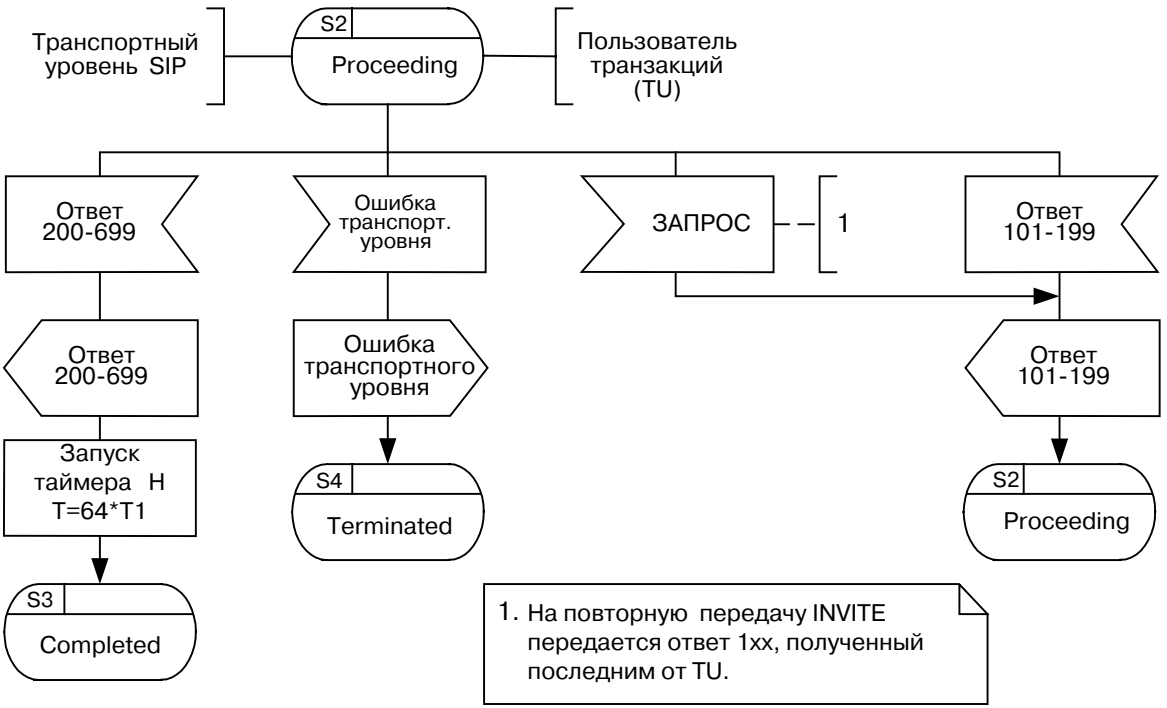


Рис. 4.18. Переход в состояние Completed

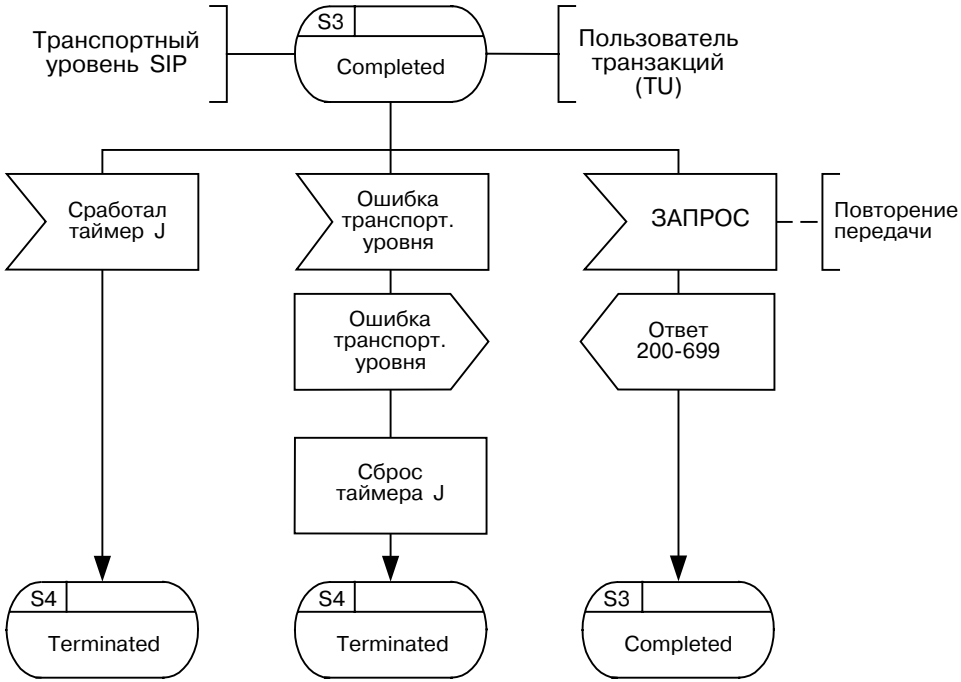


Рис. 4.19. Переход в состояние Terminated

4.6. Процедура регистрации

Если пользователь хочет инициировать сеанс связи с другим пользователем, необходимо определить адрес узла (узлов), по которому доступен адресат. Процесс определения адреса, как правило, выполняется элементами сети SIP, такими как прокси-серверы и серверы перенаправления, которые ответственны за получение запроса, определение его адресата на основании информации о местонахождении пользователя и его передачу на установленный адрес. Для этого элементы сети обращаются к серверу определения местоположения (location server), который предоставляет информацию о текущем адресе вызываемого пользователя.

Сервер определения местоположения использует базу данных, в которой каждому запрошенному адресу, например *sip:anton@niits.ru,* поставлено в соответствие один или несколько адресов, связанных с вызываемым пользователем, например *sip:anton@serv1.niits.ru.* В конечном счете, прокси-сервер с помощью услуги определения местоположения узнает адрес агента (агентов) пользователя, где в это время пребывает вызываемый пользователь.

Процедура регистрации вносит изменения в базу данных услуги определения местоположения нужного домена, где каждая запись представляет собой связку (binding) – сочетание списочного URI (address-of-record) и сопоставленных ему одного или нескольких контактных адресов. Таким образом, когда прокси-сервер этого домена получает запрос, значение поля Request-URI которого совпадает с каким либо списочным адресом, зарегистрированном в данном домене, он направляет его по контактным адресам, зарегистрированным для этого списочного адреса. Необходимость регистрировать списочный адрес у доменового сервера определения местонахождения существует только в том случае, если запросы на этот адрес будут маршрутизироваться в этот домен. В большинстве случаев это означает, что домен регистрации должен совпадать с доменом в URI списочного адреса. Существует много способов внесения изменений в базу данных сервера определения местоположения. Протокол SIP обеспечивает механизм внесения изменений непосредственно агентом пользователя. Этот механизм получил название регистрация. Регистрация подразумевает передачу сообщения REGISTER серверу определенного типа, который называется сервером регистрации (registrar). Он принимает запросы REGISTER и предоставляет информацию из них тому серверу определения местоположения домена, который он контролирует. Впоследствии информацией, сохраненной на сервере определения местоположения, пользуется прокси-сервер, ответственный за доставку запросов в данный домен.

Процесс регистрации показан на рис. 4.20. Заметим, что функции сервера регистрации и прокси-сервера в сети может выполнять одно устройство. На рисунке они изображены раздельно для облегчения понимания. Заметим также, что UAS может передавать запросы для сервера регистрации через прокси-сервер, когда они являются отдельными элементами.

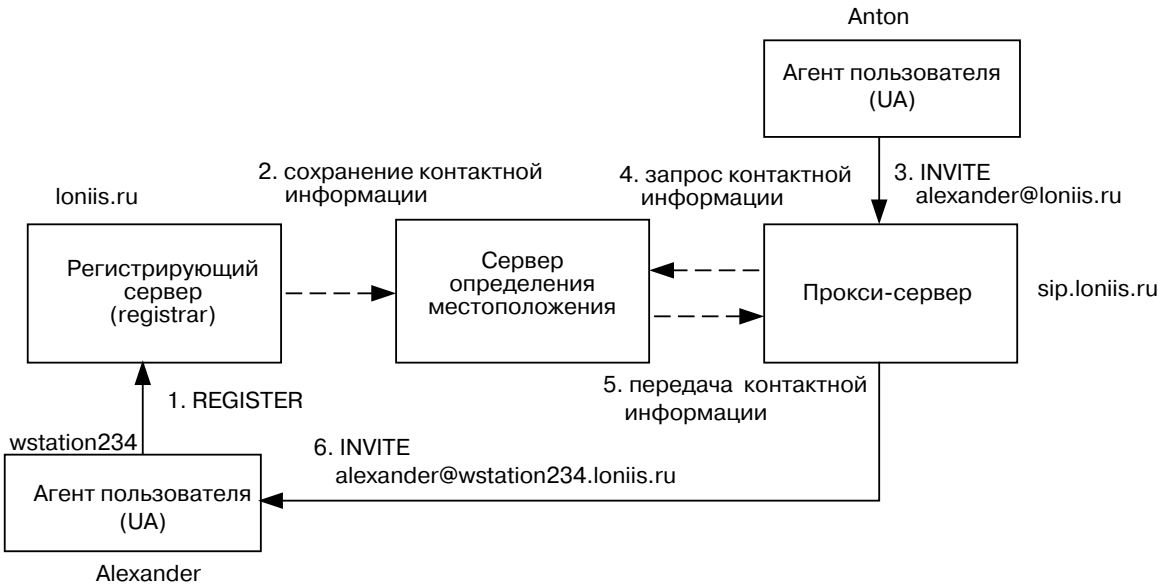


Рис. 4.20. Использование регистрации для услуги определения местоположения

Протокол SIP не предписывает определенных механизмов для реализации услуги определения местоположения. Единственное требование к серверу регистрации – возможность читать и записывать информацию в базу данных, а к прокси-серверу и серверу перенаправления домена – способность читать эту информацию. Registrar может быть физически совмещен с SIP прокси-сервером одного и того же домена.

4.6.1. Процедура формирования запроса REGISTER

Запросы REGISTER добавляют, удаляют и изменяют связки в базе данных сервера определения местоположения. Запрос REGISTER может также ввести новое соответствие между списочным адресом и одним или несколькими контактными (создать новые связки). Регистрация может быть произведена независимой третьей авторизованной стороной (которая не является ни владельцем списочного адреса, ни пользователем, инициирующим вызов на этот адрес).

За исключением моментов, описанных в этом пункте, процесс формирования запроса REGISTER и поведение клиента, передающего его, идентичны рассмотренным ранее общим правилам поведения UAC.

Запрос REGISTER не устанавливает диалог. UAC может включить в состав запроса REGISTER заголовок **Route**, базирующийся на предустановленном маршруте. Заголовок **Record-Route** не играет никакой роли в процедуре регистрации и в случае его появления должен быть игнорирован. В частности, UAC не должен создавать новый маршрут (route set) по факту наличия или отсутствия заголовка **Record-Route** в ответе на запрос REGISTER.

Запрос REGISTER должен содержать следующие основные составляющие:

- **Request-URI**

Поле Request-URI сообщает имя домена, в котором собирается зарегистрироваться пользователь (к примеру, *sip:niits.ru*). Компоненты «userinfo» (пользовательская часть) и «@» SIP-адреса не должны присутствовать.

- **To**

Заголовок **To** указывает тот списочный адрес, в отношении которого проводится процедура регистрации. Заголовок **To** и поле Request-URI различаются, т.к. первый содержит имя пользователя. Списочный адрес может быть как SIP, так и SIPS URI.

- **From**

Заголовок **From** содержит списочный адрес лица, ответственного за регистрацию. Значение заголовка совпадает со значением заголовка **To**, за исключением случаев, когда регистрация производится третьей стороной.

- **Call-ID**

Все запросы регистрации от UAC должны использовать единое значение заголовка **Call-ID** при передаче сообщения на соответствующий registrar.

- **CSeq**

Заголовок **CSeq** гарантирует правильную очередность запросов REGISTER. Агент пользователя увеличивает на единицу значение **CSeq** для каждого из запросов REGISTER с одинаковым **Call-ID**.

- **Contact**

Запрос REGISTER может включать в себя заголовок **Contact**, содержащий ноль и более контактных адресов. Агент пользователя не должен передавать новых сообщений регистрации (запросов, содержащих новые значения в заголовке **Contact**), пока не получит окончательный ответ от сервера регистрации на предыдущий запрос или пока не истечет время его ожидания для предыдущего запроса REGISTER.

В отношении параметров «action» и «expires» заголовка **Contact** в запросах REGISTER существуют особые условия.

`«action»`

Параметр «action» присутствует в сообщениях только в ходе процедуры регистрации. С его помощью клиент указывает свои предпочтения в отношении действий сервера, ответственного за данный домен – сервер будет либо пересылать, либо перенаправлять запросы, предназначенные регистрирующемуся UA. Не рекомендуется, чтобы UAC использовал этот параметр. Элементы SIP поддерживают данный параметр в целях обратной совместимости с предыдущей версией рекомендаций.

`«expires»`

Этот параметр определяет время действия связки – поставленные в соответствие списочный и контактный адрес (контактные адреса). Значение параметра – величина времени в секундах. Если этот параметр не поддерживается, вместо него используется значение заголовка **Expires**. Реализации могут трактовать значение, большее $2^{32} - 1$ (4294967295 секунд или 136 лет), как величину, эквивалентную $2^{32} - 1$. Непонятные значения должны заменяться величиной 3600.

4.6.2. Создание связок

Запрос REGISTER, отправляемый серверу регистрации, содержит контактный адрес (адреса), на который должны направляться SIP-запросы с соответствующим списочным адресом. Списочный адрес указан в заголовке **To** запроса. Заголовок **Contact**, как правило, состоит из SIP или SIPS URI, которые идентифицирует SIP терминал пользователя (к примеру, *sip:anton@serv1.niits.ru),* однако они могут использовать и другие схемы URI. UA может предложить для регистрации телефонные номера (со схемой URI «tel») или адреса электронной почты (со схемой URI «mailto») в качестве контактных номеров для определенного списочного адреса.

К примеру, пользователь Vladimir со списочным адресом *sip:vladimir@protei.ru* должен провести процедуру регистрации с сервером регистрации домена protei.ru. Информация регистрации будет впоследствии использована прокси-сервером этого домена для маршрутизации запросов, предназначенных пользователю Vladimir, на его терминал. После того как клиент зарегистрировался на сервере регистрации, он может посылать сообщения регистрации для создания новых связок или модификации существующих в случае необходимости. Ответ класса 2xx на запрос REGISTER будет содержать в заголовке **Contact** список адресов, зарегистрированных для списочного адреса на этом сервере регистрации.

Если списочный адрес в заголовке **To** запроса REGISTER является SIPS URI, то все значения в заголовке **Contact** тоже должны быть SIPS URI. Существует возможность регистрировать контактные не-SIPS URI для списочного SIPS URI только тогда, когда безопасность ресурса, представленного контактным адресом, обеспечивается с помощью других средств. Это может подойти для URI, которые используют протоколы, отличные от SIP, или для SIP-устройств, защищенных с помощью протоколов, отличных от TLS.

4.6.2.1. Продолжительность действия контактного адреса

Когда клиент передает запрос REGISTER, он может предложить срок действия регистрации (в конечном счете registrar принимает решение о выборе срока действия самостоятельно в соответствии со своей внутренней политикой). Это может производиться двумя способами: через заголовок **Expires** и параметр «expires» заголовка **Contact**. Последний позволяет указать желаемый срок действия индивидуально для каждого контактного адреса, когда в запросе REGISTER присутствует более одного контактного адреса, тогда как первый дает возможность дать одно общее значение для всех адресов, обозначенных в заголовке **Contact** и не имеющих параметра «expires». Если в сообщении не предлагается значений срока действия контактных адресов, это означает, что клиент полагается на выбор сервера.

4.6.2.2. Приоритеты среди контактных адресов

Если запрос REGISTER содержит более одного контактного адреса в **Contact**, то это значит, что UA, сформировавший запрос, предполагает ассоциировать все эти контактные адреса со списочным адресом в заголовке **To**. В списке контактных адресов в заголовке **Contact** приоритеты определяются с помощью параметра «q».

Параметр «q» указывает приоритет одного значения в поле заголовка **Contact** по отношению другим для соответствующего списочного адреса.

4.6.3. Удаление связок

Регистрации носят временный характер и теряют свою силу по истечении срока действия, однако они могут быть также удалены пользователем. Клиент может воздействовать на срок действия регистрации, установленной сервером регистрации, как описано выше. Поэтому UA запрашивает немедленное удаление связки в базе данных сервера определения местоположения, определяя срок действия равным нулю для контактного адреса в запросе REGISTER. Агенты пользователя должны поддерживать этот механизм, чтобы пользователь в случае необходимости имел возможность удалить регистрацию до истечения срока ее действия.

Значение * заголовка **Contact** воздействует одновременно на все регистрации отдельного пользователя, однако оно не может быть использовано, пока в составе сообщение присутствует заголовок **Expires** со значением 0. Использование значения * в заголовке **Contact** позволяет UA удалить все связки, относящиеся к определенному списочному адресу, не зная точных значений контактных адресов.

4.6.4. Обновление связок

Каждый UA отвечает за обновление связок, созданных ранее. Агент пользователя не должен обновлять регистрации, созданные другими UA. Ответ класса 2xx от registrar содержит заголовок **Contact** со списком всех текущих контактных адресов, поставленных в соответствие списочному адресу. UA изучает список на предмет наличия там требуемого контактного адреса. Если контактный адрес присутствует, то UA обновляет значение срока действия контактного адреса в соответствии со значением параметра «expires» или, в случае его отсутствия, – заголовка **Expires**. UA передает запрос REGISTER для каждой своей связки, пока срок действия не истек; в одном запросе REGISTER может быть совмещено несколько обновлений. UA должен использовать одно значение **Call-ID** для всех регистраций пользователя между перезагрузками. Запросы обновления регистрации посылаются на тот же сетевой адрес, что и оригинальное регистрационное сообщение, за исключением случая переадресации.

4.6.5. Определение адреса registrar

У UA есть три способа для определения адреса, на который следует передать регистрационное сообщение: адрес определяется из конфигурации, адрес определяется из списочного адреса, адрес определяется с помощью многоадресной рассылки. В конфигурацию UA может быть внесен адрес конкретного registrar. Если конфигурация UA не содержит адреса registrar, UA использует значение, находящееся справа от «@» в списочном адресе; он помещает его в поле Request-URI запроса и передает, используя обычные SIP-механизмы определения местоположения сервера, описанные в «SIP: Locating SIP Servers», [56]. К примеру, UA для пользователя *sip:vladimir@protei.ru* адресует запрос REGISTER на адрес *sip:protei.ru.*

В конечном счете, UA может быть настроен на определение адреса сервера регистрации с помощью многоадресной рассылки. Запрос регистрации передается на известный адрес многоадресной рассылки и доставляется всем серверам регистрации группы. Ответственный за данный домен сервер регистрации откликнется на запрос UAC.

4.6.6. Отправка запроса

После того как сообщение REGISTER сформировано, и определен адрес его назначения, UAC приступает к процедурам передачи сообщения на уровень транзакций, как это описано выше в разделе о работе UAC вне диалога. Если уровень транзакций сообщает об ошибке, связанной с истечением времени ожидания ответа на REGISTER, UAC не должен немедленно передавать повторный запрос REGISTER на тот же registrar; эту попытку, скорее всего, постигнет та же участь. Ожидание в течение разумного промежутка времени позволит избежать ошибок при повторной передаче и предотвратит излишнюю загрузку сети.

Если UA получает ответ с кодом ошибки 423 (Interval Too Brief), он может предпринять повторную попытку регистрации после того, как установит срок действия для каждого контактного адреса в запросе REGISTER равным или большим величины в поле заголовка **Min-Expires** ответа.

4.7. Процедура обработки запроса REGISTER

Registrar – это UAS, который отвечает на запросы REGISTER и содержит перечень связок, который доступен прокси-серверам и серверам перенаправления, находящимся в пределах данного административного домена. Registrar производит с запросами те же действия, что и обычный UAS, но принимает только запросы REGISTER. Registrar не должен формировать ответы класса 6xx. Если в запрос REGISTER входит заголовок **Record-Route**, registrar должен его игнорировать (сервер регистрации может принять запрос REGISTER, прошедший через прокси-сервер, который не опознал тип запроса и добавил значение в **Record-Route**). В свою очередь, сервер регистрации сам не должен помещать заголовок **Record-Route** в ответы на REGISTER.

Registrar обрабатывает запросы REGISTER в порядке их поступления. Запросы REGISTER обрабатываются автоматически, т.е. они либо обрабатываются полностью, либо не обрабатываются совсем. Обработка каждого сообщения производится независимо от прочих регистраций или изменений связок.

При получении запроса REGISTER сервер регистрации последовательно выполняет следующие шаги:

- Registrar анализирует поле Request-URI, чтобы выявить право на доступ к связкам домена, указанного в Request-URI. Если он определяет, что в Request-URI указан другой домен, и registrar, помимо своих функций, выполняет функции прокси-сервера, сервер должен передать запрос в домен, когда тот адресован по всем правилам работы прокси-сервера.
- Следуя процедурам функционирования UAS вне диалога, registrar должен обработать значения в поле заголовка **Require** для того, чтобы гарантировать поддержку всех необходимых расширений.
- Registrar должен провести аутентификацию UAC. Механизм аутентификации SIP-агентов пользователя раскрыт в главе 6. Процедура регистрации никоим образом не влияет на процедуру SIP-аутентификации. Если механизм аутентификации не доступен, registrar может использовать адрес из заголовка **From** для определения инициатора запроса.
- Registrar должен установить, уполномочен ли аутентифицированный пользователь модифицировать регистрации для данного списочного адреса. К примеру, сервер регистрации может обратиться к базе данных автори-

зации, где отображается список всех списочных адресов, для которых этот пользователь имеет право изменять связки. Если пользователь не в праве изменять регистрации, registrar должен передать ответ с кодом 403 (Forbidden) и пропустить все оставшиеся шаги.

- Списочный адрес registrar получает из заголовка **To** запроса. Если он не согласуется с доменом, указанным в Request-URI, registrar должен передать ответ 404 (Not Found) и пропустить оставшиеся шаги. Затем из адреса должны быть удалены все параметры, и все символы в видоизмененной escaped-кодировке («%» + код символа в шестнадцатеричном виде) должны быть преобразованы в стандартную форму. Значение, полученное в результате, используется в качестве указателя в списке связок базы данных.
- Registrar проверяет запрос на наличие в нем заголовка **Contact**. Если его нет, процесс обработки REGISTER переходит на завершающую стадию (п. 8). Когда **Contact** присутствует, registrar определяет, содержит ли заголовок **Contact** хоть одно значение *, а также проверяет присутствие заголовка **Expires**. Если запрос содержит дополнительные поля **Contact** или срок действия отличен от нуля, запрос считается неверным, сервер передает ответ с кодом 400 (Invalid Request) и все оставшиеся шаги пропускаются. В противном случае registrar проверяет, совпадает ли **Call-ID** со значением, сохраненным для каждой связки. Если совпадения нет, связка удаляется; если есть – связка удаляется только в том случае, когда значение **CSeq** в запросе больше значения, сохраненного для данной связки. В противном случае обновление должно быть отменено, и запрос отбрасывается.
- Registrar по очереди обрабатывает каждый контактный адрес в заголовке **Contact**. Для каждого адреса он определяет срок его действия, исходя из следующего:
- если значение заголовка имеет параметр «expires», величина должна быть выбрана равной запрашиваемой для данного контактного адреса;
- если параметр «expires» отсутствует, но запрос содержит заголовок **Expired**, величина должна быть выбрана равной запрашиваемой для всех контактных адресов;
- Registrar может выбрать срок действия регистрации меньше запраши-

ваемого. Только в случае, когда интервал больше нуля и меньше одного часа и при этом меньше минимума, установленного конфигурацией сервера регистрации, registrar может отклонить регистрацию с ответом 423 (Interval Too Brief). Этот ответ должен включать в себя заголовок **Min-Expires**, указывающий минимально возможный интервал. После этого оставшиеся шаги пропускаются.

- при отсутствии и параметра «expires», и заголовка **Expires** сервер регистрации берет значение, принятое по умолчанию, исходя из своей конфигурации.
- Срок действия регистрации зачастую используется при создании услуг. Пример тому – follow-me service, где пользователь может быть доступен на определенном терминале в течение короткого периода времени. Поэтому registrar должен принимать регистрации на короткий срок; запрос отклоняется только в том случае, если срок этот настолько короткий, что его обновления ухудшают качество функционирования registrar.
- Для каждого адреса registrar изучает список текущих связок, используя правила сравнения URI. Если связка отсутствует, сервер регистрации условно добавляет новую, если присутствует – проверяет значение **Call-ID**. Когда значения **Call-ID** в существующей связке и в запросе различаются, связка должна быть удалена, если срок действия равен нулю, и обновлена, если не равен. При условии, что значения **Call-ID** одинаковы, registrar производит сравнение значений **CSeq**: если величина в запросе больше значения в связке, связка должна быть обновлена или удалена так же, как приведено выше. В противном случае обновление будет прекращено и запрос получит отказ. Каждая учетная запись сохраняет значения **Call-ID** и **CSeq** запроса.
- Обновления связок должны быть видны прокси-серверам и серверам перенаправления только тогда, когда все обновления и добавления закончились успешно. Если любое из них терпит неудачу, запрос отклоняется с ответом 500 (Server Error), и все условные связки удаляются.
- Registrar передает ответ 200 (OK). Ответ должен содержать заголовок **Contact** со списком всех активных на данный момент контактных адресов. Каждое значение должно иметь параметр «expires», отображающее срок действия, который выбрал registrar. В ответ должен также входить заголовок **Date**.

Рассмотрим пример процедуры регистрации.

Пользователь Vladimir регистрируется при запуске терминального оборудования. Обмен сообщениями показан на рис. 4.21. С целью упрощения процедура аутентификации, необходимая при проведении регистрации, на рисунке не показана.

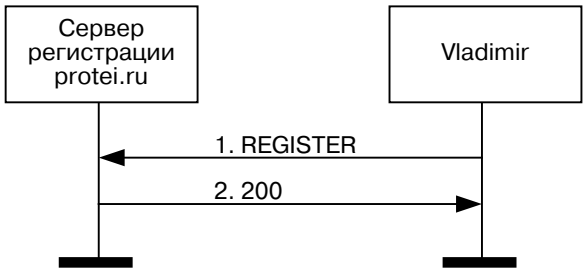


Рис. 4.21. Процедура регистрации

Запрос REGISTER (1) имеет следующий вид:

```
REGISTER sip:registrar.protei.ru SIP/2.0
Via:SIP/2.0/UDP serv3.protei.ru:5060;branch=z9hG4bKnashds7
Max-Forwards:70
To:Bob <sip:vladimir@protei.ru>
From:Bob <sip:vladimir@protei.ru>;tag=456248
Call-ID:843817637684230@998sdasdh09
CSeq:1826 REGISTER
Contact:<sip:vladimir@192.0.2.4>
Expires:7200
Content-Length:0
```

Срок действия регистрации истекает через два часа. REGISTER передает ответ 200 (OK), который выглядит следующим образом (2):

```
SIP/2.0 200 OK
Via:SIP/2.0/UDP serv3.protei.ru:5060;branch=z9hG4bKnashds7
;received=192.0.2.4
To: Bob <sip:vladimir@protei.ru>;tag=2493k59kd
From: Bob <sip:vladimir@protei.ru>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:vladimir@192.0.2.4>
Expires: 7200
Content-Length: 0
```

4.8. Процедура запроса информации о функциональных возможностях

SIP-запрос OPTIONS позволяет агенту пользователя запрашивать у другого UA или у прокси-сервера информацию о функциональных возможностях. С помощью этой процедуры клиент может, не вызывая пользователя, узнать о поддерживаемых его клиентом пользователя типах запросов, типах тел сообщения, расширениях, кодеках и др. Например, перед тем как поместить в запрос INVITE заголовок **Require**, содержащий option-tag определенной опции, клиент может проверить поддержку данной опции сервером UA адресата, передав запрос OPTIONS. Если в заголовке **Supported** ответа будет содержаться соответствующий option-tag, то UAS поддерживает требуемую опцию. Запрос OPTIONS должны поддерживать все UA. Адресат запроса OPTIONS изначально указывается в поле Request-URI. Если запрос OPTIONS адресован прокси-серверу, URI в поле Request-URI не содержит пользовательской части, подобно значению поля Request-URI при передаче запроса REGISTER.

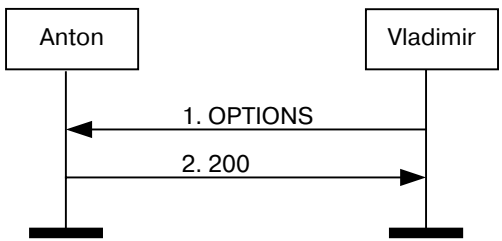


Рис. 4.22. Процедура запроса информации о функциональных возможностях

В случае, если какой-либо сервер получает запрос OPTIONS со значением заголовка **Max-Forwards**, равным 0, он может ответить на данный запрос, несмотря на то, что поле Request-URI идентифицирует другой сервер. Такое поведение системы заимствовано у протокола HTTP. Используя функцию трассировки маршрута (утилита «traceroute» для HTTP), UA может выяснить функциональные возможности отдельных серверов путем передачи последовательности запросов OPTIONS с заголовком **Max-Forwards**, значение которого каждый раз уменьшается на единицу. Когда истекает время ожидания ответа на запрос OPTIONS, уровень транзакций может передать TU соответствующее уведомление. Это может означать, что запрос не будет доставлен к месту назначения и, соответственно не будет обрабо-

тан адресатом. Запрос OPTIONS может передаваться в ходе уже установленного диалога для запроса функциональных возможностей UA собеседника, сведения о которых могут понадобиться впоследствии.

4.8.1. Создание запроса OPTIONS

Запрос OPTIONS формируется с использованием основных правил создания запросов для UAC. Использование заголовка **Contact** для запроса OPTIONS не обязательно. В сообщение должен быть помещен заголовок **Accept**, чтобы указать тип тела сообщения, которое UAC предпочел бы видеть в ответе. Как правило это тип application/sdp, описывающий характеристики медиапотока. Ответ на запрос данных о функциональных возможностях приходит от сервера, указанного в поле Request-URI, однако гарантия того, что будущие запросы будут получены сервером, ответившим на запрос OPTIONS, может быть дана только тогда, когда OPTIONS передается в ходе уже установленного диалога. Ниже представлен пример запроса OPTIONS:

```
OPTIONS sip:alexander@niits.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.tu;branch=z9hG4bKhjhs8ass877
Max-Forwards: 70
To: <sip:alexander@niits.ru>
From: Anton <sip:anton@niits.ru>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:anton@pc33.niits.ru>
Accept: application/sdp
Content-Length: 0
```

4.8.2. Обработка запроса OPTIONS

Ответ на запрос OPTIONS формируется с использованием стандартных для UAS правил формирования SIP-ответа. Код ответа должен быть выбран так же, как при поступлении запроса INVITE, т.е. – 200 (OK), если UAS готов принять вызов, 486 (Busy Here), если UAS занят, и т.д. Это позволяет использовать запрос OPTIONS еще и для определения общего состояния UAS – результат обработки OPTIONS указывает, сможет ли UAS принять запрос INVITE. Запрос OPTIONS, полученный в ходе диалога, приводит к созданию ответа с кодом 200 (OK), который идентичен аналогичному ответу, передаваемому вне диалога, и не оказывает влияния на состояние диалога. Несмотря на то, что так же, как INVITE, запрос OPTIONS может передаваться вне диалога, при размножении запросов прокси-сервером на

него может придти только один ответ с кодом 200 (OK) (на INVITE их может прийти несколько – по одному с каждого направления). Это происходит потому, что только запрос INVITE обладает особым статусом при обработке прокси-серверами, запросы остальных типов обрабатываются по общему механизму. Если запрос OPTIONS предназначен прокси-серверу, то он создает ответ с кодом 200 (OK), указывающий функциональные возможности прокси-сервера. При этом ответ не содержит тела сообщения.

В ответе 200 (OK) на OPTIONS должны присутствовать заголовки **Allow**, **Accept**, **Accept-Encoding**, **Accept-Language** и **Supported**. Если ответ создается прокси-сервером, заголовок **Allow**, указывающий поддерживаемые типы запросов, исключается за ненадобностью. Заголовок **Contact** может присутствовать в ответе 200 (OK); при этом он будет иметь ту же семантику, что и ответе класса 3xx, т.е. он будет включать в себя список адресов, по которым предположительно можно связаться с требуемым пользователем. Может также присутствовать заголовок **Warning**. Сообщение может содержать тело сообщения, тип которого указан в заголовке **Accept** полученного запроса OPTIONS (в случае отсутствия заголовка **Accept** по умолчанию устанавливается тип application/sdp). Если среди типов тел сообщения в заголовке **Accept** запроса OPTIONS содержится тип, который предназначен для описания функциональных возможностей в части медиа-информации, UAS должен передать в ответе тело именного этого типа с указанными функциональными возможностями. Пример ответа UAS на запрос OPTIONS представлен ниже (тело сообщения в примере не показано).

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKhjhs8ass877;received=192.0.2.4
To: <sip:alexander@niits.ru>;tag=93810874
From: Anton<sip:anton@niits.ru>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:alexander@niits.ru>
Contact: <mailto:alexander@niits.ru>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE
Accept: application/sdp
Accept-Encoding: gzip
Accept-Language: en
Supported: 100rel
Content-Type: application/sdp
Content-Length: 274
```

4.9. Процедура отмены запроса

Запрос CANCEL, как следует из его названия, используется для отмены предшествующего запроса, переданного клиентом. Точнее, он требует от UAS прекратить обработку запроса и создать ответ на него с определенным кодом. Запрос CANCEL не оказывает воздействия на запрос, на который UAS уже передал окончательный ответ. Поэтому отмена запросов представляет наибольшую важность для запросов, формирование ответов на которые требует больших затрат серверного времени. По этой причине CANCEL лучше всего подходит для запросов INVITE, которые требуют длительного времени для генерации ответа. В этом случае UAS, который получает CANCEL для INVITE, но еще не передал окончательного ответа, прекращает передавать пользователю сигнал вызова и посылает ответ на INVITE с определенным кодом ошибки (487).

Запросы CANCEL могут быть сформированы и переданы и прокси-серверами, и клиентами агента пользователя. Раздел 4.4 описывает, при каких условиях UAC отменит запрос INVITE, в разделе 5.2 дано описание использования запроса CANCEL прокси-сервером.

4.9.1. Действия клиента

Запросы CANCEL передаются для отмены запросов INVITE. Поскольку на прочие запросы ответы приходят незамедлительно, передача CANCEL для не-INVITE-запросов всегда будет приводить к явлению, когда с равной вероятностью первым могут быть доставлен и окончательный ответ – клиенту, и запрос CANCEL – серверу.

Для формирования запроса CANCEL применяются следующие процедуры. Поля Request-URI, **Call-ID**, **To**, числовая часть поля **CSeq** и **From** в запросе CANCEL должны быть идентичны полям в отменяемом запросе, включая параметры «tag». CANCEL, формируемый клиентом, должен иметь только одно значение заголовка **Via**, соответствующее верхнему значению **Via** отменяемого запроса. Использование тех же значений для этих полей позволяет установить соответствие CANCEL отменяемому запросу. Однако поле типа запроса в заголовке **CSeq** должно иметь значение CANCEL. Это приводит к тому, что сообщение будет идентифицировано и обработано как отдельная транзакция.

Запрос CANCEL не должен содержать заголовков **Require** и **Proxy-Require**.

Если отменяемый запрос содержит заголовок **Route**, CANCEL тоже должен содержать значения этого заголовка. Это нужно для того, чтобы stateless прокси-серверы могли правильно маршрутизировать запросы CANCEL.

После формирования запроса CANCEL клиент должен проверить, не получил ли он ответа (предварительного или окончательного) на отменяемый запрос. Если предварительного ответа не было получено, запрос CANCEL не должен передаваться; точнее, клиент должен дождаться прихода предварительного ответа, прежде чем передать запрос CANCEL. Если на оригинальный запрос пришел окончательный ответ, нет смысла передавать CANCEL, так как он не оказывает влияния на запросы, на которые уже создан окончательный ответ. Когда клиент решает передать запрос CANCEL, он создает для него клиентскую транзакцию и передает ей запрос вместе с адресом назначения, номером порта и типом транспортного протокола. Адрес назначения, порт и тип транспортного протокола для CANCEL должны быть идентичными тем, которые использовались при передаче оригинального запроса. Если бы было разрешено передать CANCEL до получения ответа на предыдущий запрос, сервер мог бы получить CANCEL раньше оригинального запроса.

Заметим, что и транзакция, соответствующая оригинальному запросу, и CANCEL-транзакция будут завершены независимо. Однако UAC, отменяющий запрос, не может быть уверен в получении ответа с кодом 487 (Request Terminated) на оригинальный запрос, так как UAS, функционирующий в соответствии с более ранней версией спецификации, может не создать такого ответа. В связи с этим, если за интервал времени 64 T1 секунд не пришло окончательного ответа на оригинальный запрос, клиент расценивает оригинальную транзакцию отмененной и разрушает клиентскую транзакцию оригинального запроса.

4.9.2. Действия сервера

Сообщение CANCEL запрашивает TU на стороне сервера отмену незавершенной транзакции. Чтобы определить транзакцию, которую требуется отменить, TU принимает запрос CANCEL, а затем применяет процедуры сопоставления транзакций, описанные в § 4.4.8. Отменена может быть только единственная транзакция.

Обработка запроса CANCEL сервером зависит от типа сервера. Stateless прокси-сервер его перешлет, stateful прокси-сервер может ответить на него и самостоятельно создать несколько запросов CANCEL, а UAS ответит на него.

Порядок работы прокси-сервера с запросом CANCEL описывается в следующей главе (§ 5.2.8.)

Сперва UAS производит обработку CANCEL в соответствии с общими правилами обработки для UAS, описанными в § 2.3. Однако поскольку запросы CANCEL передаются последовательно от одного ретрансляционного участка к другому и не могут быть переданы повторно, сервер не требует от них подтверждения подлинности (для того, чтобы они имели надлежащий отклик аутентификации в заголовке **Authorization**). Заметим также, что запросы CANCEL не содержат заголовка **Require**.

Если UAS, выполняя процедуры описанные выше, не нашел соответствующую транзакцию для CANCEL, он передает на запрос CANCEL ответ с кодом 481 (Call Leg/Transaction Does Not Exist). Если транзакция для оригинального запроса все еще существует, действия UAS при получении запроса CANCEL зависят от того, передал ли он уже окончательный ответ на оригинальный запрос или нет. Если передал, то CANCEL не оказывает влияния на обработку оригинального запроса, на состояние сеанса и на ответы на оригинальный запрос. Если UAS не передавал окончательного ответа на оригинальный запрос, его поведение зависит от типа оригинального запроса. Если таковым являлся INVITE, UAS незамедлительно передает ответ на INVITE с кодом 487 (Request Terminated). Запрос CANCEL не влияет на обработку транзакций других типов сообщений. Вне зависимости от типа оригинального запроса, поскольку было установлено соответствие между CANCEL и существующей транзакцией, UAS передает ответ с кодом 200 (OK). Этот ответ составлен с использованием правил, описанных в § 2.3.2, с учетом того, что «tag» в заголовке **To** ответа на CANCEL и «tag» в заголовке **To** ответа на оригинальный запрос должны быть одинаковыми. Ответ на CANCEL передается серверной транзакции для передачи по сети. На рис. 4.23 представлен пример отмены запроса INVITE с использованием запроса CANCEL.

4.10. Инициирование сеансов связи

Когда клиент агента пользователя желает установить сеанс связи (аудио или видео), он формирует запрос INVITE. INVITE – запрос сервера установить сеанс связи. Он пересылается прокси-серверами и, в конечном счете, приходит на один или несколько UAS, которые потенциально могут принять предложение клиента.

Эти UAS, как правило, требуют от пользователя подтверждения приема вызова. По прошествии некоторого времени UAS может принять предложение путем передачи ответа 2xx (OK); после этого сеанс связи считается установленным. Если предложение не принято, передаются ответы с кодами 3xx, 4xx, 5xx или 6xx, в зависимости от причины отказа. Перед передачей окончательного ответа UAS может передавать предварительные ответы (1xx) для того, чтобы уведомлять UAC о состоянии процесса обработки вызова на стороне вызываемого пользователя.

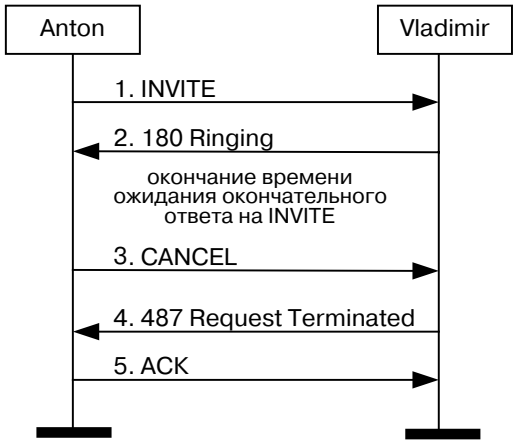


Рис. 4.23. Процедура отмены запроса INVITE

После возможного получения одного или нескольких предварительных ответов UAC получит один или более ответов 2xx или один окончательный ответ не-2xx. Из-за того, что ожидание окончательного ответа на запрос INVITE может занять длительное время, механизмы INVITE-транзакций отличаются от механизмов для других запросов (например, OPTIONS). После того как UAC получает окончательный ответ, он должен передавать ACK на каждую принятую повторную передачу окончательного ответа. Процедура передачи подтверждения ACK зависит от типа ответа. Для окончательных ответов с кодами 300 по 699 работа с запросом ACK происходит на уровне транзакций в соответствии с общим набором правил, как это описано в § 4.4. Для ответов класса 2xx работу с запросом ACK выполняет ядро UAC.

Ответ класса 2xx на запрос INVITE устанавливает сеанс связи, а также создает диалог между UA, который отправляет INVITE, и UA, который формирует ответ класса 2xx. Поэтому когда ответы класса 2xx приходят от разных удаленных UA (из-за размножения запроса INVITE на прокси-сервере), каждый из ответов устанавливает свой отдельный диалог. Все эти диалоги являются частью одного сеанса.

В этом разделе описываются подробности установления сеанса с использованием запроса INVITE. UA, который поддерживает INVITE, поддерживает также запросы ACK, CANCEL и BYE.

4.10.1. Процедуры UAC. Создание начального запроса INVITE

Формирование первоначального запроса INVITE происходит вне диалога с использованием соответствующих стандартных процедур, описанных в главе 2. Дополнительная обработка требуется лишь в особых случаях.

Как правило, INVITE содержит заголовок **Allow**. Он указывает, какие типы запросов могут быть использованы вызывающей стороной в процессе диалога. Скорее всего, в запросе будет также присутствовать заголовок **Supported**. Он перечисляет все расширения, поддерживаемые клиентом агента пользователя. Возможно присутствие заголовка **Accept**, который указывает поддерживаемые данным UA типы тел сообщения из предложенных в принятом ответе или запросе в ходе диалога. Заголовок **Accept** особенно важен для индикации поддержки различных форматов описания сеансов. UAC может также добавить заголовок **Expires**, чтобы ограничить время действия приглашения к установлению соединения. Если время, указанное в **Expires**, истечет до получения окончательного ответа на INVITE, UAC передает сообщение CANCEL. Кроме этого, UAC может посчитать нужным добавить заголовки **Subject**, **Organization** и **User-Agent**; все они содержат информацию, относящуюся к запросу INVITE.

UAC может добавить в запрос INVITE тело сообщения. Существуют отдельные правила для тел, содержащих описание сеанса связи, – их заголовок **Content-Disposition** имеет значение session. Протокол SIP использует модель типа «запрос/ответ» (offer/answer), где один UA посылает предложение – запрос с описанием сеанса. Запрос предлагает желаемые средства общения (аудио, видео), их параметры (такие как типы кодека) и адреса для получения медиа-информации от отвечающей стороны. Другой UA отвечает своим описанием сеанса,

указывающим, какие средства общения приняты, параметры, применяемые к ним, и адреса для получения медиа-информации от инициатора запроса. Обмен offer/answer происходит в контексте диалога, поэтому когда передача запроса INVITE приводит к созданию нескольких диалогов, обмен происходит отдельно для каждого. Имеются следующие правила использования модели offer/answer для начальной INVITE-транзакции:

- Начальное offer с описанием сеанса должно содержаться в сообщении INVITE или в первом надежном сообщении от UAS, не информирующем об ошибке (ответе класса 2xx).
- Если offer содержится в сообщении INVITE, то answer должен присутствовать в надежном ответе на INVITE от UAS. Точно такой же ответ с описанием сеанса (answer) может быть помещен в предварительные ответы, предшествующие окончательному. UAC трактует первый пришедший answer как описание сеанса связи и игнорирует описания сеанса во всех следующих ответах на начальное сообщение INVITE.
- Если offer содержится в первом надежном сообщении от UAS, не информирующем об ошибке, answer должен быть в подтверждении ACK на ответ класса 2xx.
- После передачи или получения answer на первый offer UAC может помещать следующие offer в запросы, следуя правилам для запроса данного типа, но только если он получил answer на предыдущие offer и не передал offer, на которые answer еще не получены.
- После того как UAS передал или получил answer на начальный offer, он не должен включать новые offer в состав ответов на начальное сообщение INVITE. Это означает, что UAS никогда не сможет в одиночку создать новые offer до завершения начальной транзакции.

Приведенные выше правила определяют для агентов пользователя два вида обмена:

- offer в запросе INVITE и answer в ответе класса 2xx (и, возможно, в ответе класса 1xx с тем же значением),
- offer в ответе класса 2xx и answer в подтверждении ACK,

причем все агенты пользователя, поддерживающие запрос INVITE, должны поддерживать оба этих вида обмена. К тому же, протокол описания сеансов связи Session Description Protocol (SDP) должен поддерживаться любыми UA [37]. Указанные выше ограничения для модели offer/answer применимы лишь к тем телам сообщения, в которых заголовок **Content-Disposition** имеет значение session. Поэтому возможен случай, когда и INVITE, и ACK содержат тело сообщения (например, запрос INVITE переносит фотографию (*Content-Disposition: render*), а ACK – описание сеанса (*Content-Disposition: session*)).

Если заголовок **Content-Disposition** отсутствует, то для сообщений со значением заголовка **Content-Type** application/sdp для заголовка **Content-Disposition** по умолчанию устанавливается значение session, в противном случае устанавливается значение render.

После создания запроса INVITE UAC отправляет его, следуя процедурам, определенным для передачи запросов вне диалога. Это приводит к формированию клиентской транзакции, которая, в конечном счете, передает запрос и доставляет ответы клиенту.

4.10.2. Процедуры UAC. Обработка ответов на запрос INVITE

Как только UAC передает запрос INVITE клиентской транзакции, он переходит в режим ожидания ответов на запрос. Если клиентская INVITE-транзакция вместо передачи ответа сообщает об истечении времени ожидания окончательного ответа, пользователь транзакции (TU) действует, как в случае получения ответа с кодом 408 (Request Timeout).

4.10.2.1. Ответы 1xx

До получения окончательного ответа может придти любое число предварительных ответов. Предварительные ответы на запрос INVITE могут привести к созданию диалогов, находящихся на «ранней стадии». Такие диалоги потребуются только в том случае, если у клиента возникнет необходимость передать запрос серверу до завершения начальной INVITE-транзакции. Заголовки, содержащиеся в предварительном ответе, действительны только на «ранней стадии» диалога (например, заголовок **Allow** в предварительном ответе содержит типы запросов, которые могут быть использованы в процессе диалога, пока тот находится на «ранней стадии»).

4.10.2.2. Ответы 3xx

Ответы класса 3xx могут содержать в поле заголовка **Contact** одно или несколько значений, указывающих адреса, по которым вызываемый пользователь может быть доступен. В зависимости от кода ответа класса 3xx UAC может предпринять попытку вызова пользователя по новым адресам.

4.10.2.3. Ответы 4xx, 5xx и 6xx

На запрос INVITE могут быть получены окончательные ответы класса, отличного от 2xx. Ответы класса 4xx, 5xx и 6xx могут содержать в поле заголовка **Contact** значение, указывающее местоположение дополнительной информации об ошибке. При получении окончательного ответа класса, отличного от 2xx, все диалоги, находящиеся на «ранней стадии», разрушаются. Кроме того, когда приходит окончательный ответ класса, отличного от 2xx, клиентская транзакция переходит в состояние «Completed», предусматривающее ожидание дополнительных повторных ответов.

4.10.2.4. Ответы 2xx

UAC может получить несколько ответов класса 2xx на один запрос INVITE, что происходит при прохождении запросов через прокси-сервер, размножающий запросы. Каждый ответ отличается параметром «tag» в поле заголовка **To** и представляет отдельный диалог со своим идентификатором диалога. Если идентификатор диалога в ответе 2xx соответствует идентификатору организуемого диалога, диалог переходит в установленное состояние и route set должен быть пересчитан на основе заголовка **Record-Route** ответа класса 2xx (это делается с целью обеспечить обратную совместимость с предыдущей версией рекомендаций).

Ядро UAC должно создавать запрос ACK при получении каждого ответа класса 2xx. Заголовки запроса ACK формируются так же, как и заголовки любых других сообщений, передаваемых в режиме диалога, за исключением **CSeq** и заголовков, относящихся к процедуре аутентификации. Порядковый номер в заголовке **CSeq** должен совпадать с номером в первоначальном запросе INVITE, но в поле типа запроса заголовка должно быть значение ACK. Запрос ACK содержит то же значение отклика аутентификации, что и INVITE.

Если ответ класса 2xx содержит offer (предложение с описанием сеанса), в соответствии с правилами, приведенными выше, в теле сообщения ACK должен

находиться answer (ответ с описанием сеанса). Если offer в ответе 2xx не может быть принято, ядро UAC формирует answer в подтверждении ACK и затем незамедлительно передает сообщение BYE.

После формирования подтверждения ACK определяются адрес места назначения, номер порта и тип транспортного протокола. Запрос направляется непосредственно на транспортный уровень SIP, минуя клиентскую транзакцию. Это происходит потому, что за повторные запросы ACK отвечает не уровень транзакций, а ядро UAC. Подтверждение ACK должно направляться клиентской стороне транспортного уровня SIP каждый раз, когда приходит очередной повторно переданный ответ класса 2xx.

Если после подтверждения ответа 2xx на INVITE UAC не желает продолжать текущий диалог, он должен разорвать диалог передачей сообщения BYE.

4.10.3. Процедуры UAS. Обработка запроса INVITE

Ядро UAS получает запросы INVITE от уровня транзакций. Вначале выполняются общие процедуры обработки для UAS (как для запросов INVITE вне диалога, так и для запросов INVITE в режиме диалога). Допуская, что эти процедуры обработки выполняются без создания ответа, ядро UAC выполняет дополнительные шаги обработки:

- Если поступает запрос INVITE, содержащий заголовок **Expires**, ядро UAS запускает таймер со значением, указанным в этом заголовке. Срабатывание таймера сигнализирует об истечении времени действия приглашения к установлению сеанса связи. Если это происходит до того, как UAS сформировал окончательный ответ, передается ответ с кодом 487 (Request Terminated).
- Если запрос приходит в ходе диалога, то первоначально производятся процедуры обработки запроса, не зависящие от его типа, как описано в разделе 4.3. Может также произойти модификация сеанса (см. раздел 4.11).
- Если запрос содержит параметр «tag» в заголовке **To**, но идентификатор диалога не соответствует ни одному существующему диалогу, это говорит о том, что UAS перезагрузился, находится в аварийном состоянии или получил запрос, предназначавшийся другому UAS.

Дальнейшее описание процедур обработки предполагает, что передача запроса INVITE происходит вне диалога, другими словами, запрос направлен на установление сеанса.

Запрос INVITE может содержать описание сеанса, в этом случае UAS посредством offer приглашается к участию в сеансе связи. Возможно, что пользователь уже является участником данного сеанса, несмотря на то, что INVITE передается вне диалога. Это может произойти, когда пользователь приглашается к одной и той же multicast-конференции разными ее участниками. При желании UAS может использовать идентификаторы внутри описания сеанса, чтобы обнаружить такое дублирование. Например, SDP-описание включает в себя идентификатор сеанса (session id) и порядковый номер SDP-описания, указанный в поле origin, – так называемый номер версии. В случае, когда пользователь уже является участником сеанса, и параметры сеанса, содержащиеся в SDP-описании запроса INVITE, те же, UAS принимает INVITE, не информируя об этом пользователя, и передает ответ класса 2xx.

Если запрос INVITE не содержит описания сеанса, это значит, что UAC обращается к UAS с просьбой принять участие в сеансе и передать информацию offer. UAS должен предоставить offer в своем первом надежном ответе, не информирующем об ошибке (ответ класса 2xx).

UAS может информировать о текущей стадии обработки, принятии, перенаправлении или отклонении информации offer. Во всех этих случаях он формирует ответ в соответствии с процедурами для создания ответов вне диалога.

4.10.3.1. Текущая стадия обработки

Если UAS не в состоянии ответить на приглашение немедленно, он может предоставить UAC некоторую информацию о текущей стадии обработки запроса (например, оповестить о том, что в данный момент пользователю передается вызывной сигнал). Это выполняется путем передачи предварительных ответов с кодами от 101 по 199. Передача таких предварительных ответов приводит к установлению диалогов, находящихся на «ранней стадии». UAS может передать неограниченное число предварительных ответов. Каждый из них должен иметь одно и тоже значение идентификатора диалога (dialog ID).

Если UAS нуждается в отсрочке передачи ответа на запрос INVITE, он должен запросить «увеличение времени обработки» для того, чтобы предотвратить

аннулирование транзакции прокси-серверами. Прокси-сервер может разрушить транзакцию, когда интервал между ответами достигает трех минут. Чтобы не допустить этого, UAS должен передавать предварительные ответы (за исключением ответа с кодом 100) каждую минуту с учетом того, что некоторые из них могут быть утеряны (т.к. по умолчанию предварительные ответы передаются ненадежно). INVITE-транзакция может продолжать свое существование на время действия отсрочки в случаях, когда пользователь поставлен на ожидание или при межсетевом взаимодействии с системами ТфОП, которые поддерживают соединения в предответном состоянии (например, интерактивные системы речевого ответа IVR).

4.10.3.2. Перенаправление запроса INVITE

Если UAS решает перенаправить вызов, он передает ответ класса 3xx. Ответ с кодом 300 (Multiple Choices), 301 (Moved Permanently) или 302 (Moved Temporarily) должен содержать заголовок **Contact** с одним или несколькими новыми адресами, на которые следует перенаправить вызов. Ответ передается серверной INVITE-транзакции, которая производит его повторную передачу.

4.10.3.3. Отклонение запроса INVITE

Наиболее простой сценарий имеет место, когда вызываемый пользователь не может или не желает принять дополнительный вызов на свой терминал. При этом вызывающему пользователю передается ответ с кодом 486 (Busy Here). Если UAS знает о том, что никакой другой терминал не в состоянии принять этот вызов, передается ответ с кодом 600 (Busy Everywhere). Так как маловероятно, что UAS может располагать такими широкими сведениями, ответ с кодом 600 обычно не используется. Ответ передается серверной INVITE-транзакции, которая занимается его повторной передачей.

UAS, отклоняющий информацию offer, которая содержится в запросе INVITE, должен передать ответ с кодом 488 (Not Acceptable Here). Такой ответ должен включать в себя заголовок **Warning** с содержимым, объясняющим причину отказа.

4.10.3.4. Прием запроса INVITE

Ядро UAS формирует ответ класса 2xx, который устанавливает диалог. Ответ класса 2xx на запрос INVITE, как правило, содержит заголовки **Allow** и **Supported** и, возможно, заголовок **Accept**. Содержимое этих заголовков позволяет UAC определить типы запросов и расширения, поддерживаемые UAS на протяжении

сеанса, без дополнительных проб. Если запрос INVITE содержит offer, и UAS еще не передал answer, ответ класса 2xx должен включать в себя answer. Если в запросе INVITE нет offer, ответ класса 2xx должен содержать offer в случае, если UAS еще не передал offer. Как только ответ сформирован, он передается серверной INVITE-транзакции. Заметим, что серверная INVITE-транзакция будет разрушена, как только она получит окончательный ответ и передаст его на транспортный уровень SIP. Следовательно, необходимо периодически передавать ответы непосредственно на транспортный уровень SIP, пока не придет подтверждение ACK. Ответ класса 2xx передается на транспортный уровень SIP с интервалом, начинающимся с T1 и удваивающимся после каждой повторной передачи, пока не достигнет T2 (величины T1 и T2 указаны в п. 4.4).

Повторения передачи ответа прекращаются, как только приходит подтверждение ACK, не зависимо от того, какие транспортные протоколы используются для доставки ответов. Повторная передача ответа класса 2xx является «сквозной», в тоже время, не исключается возможность наличия в сети звеньев, использующих для передачи ответов протокол UDP. Чтобы гарантировать надежную доставку на этих участках, повторная передача ответа класса 2xx производится даже при использовании UAS надежного транспортного протокола, такого как TCP. Если за время повторной передачи ответа класса 2xx, равное 64 T1, подтверждение ACK не приходит, диалог устанавливается, но сеанс должен быть разрушен. Это происходит путем передачи сообщения BYE, как описано в п. 4.12.

4.11. Процедуры модификации сеансов связи

Успешный запрос INVITE устанавливает диалог между двумя агентами пользователя и сеанс связи, используя модель offer/answer. В разделе 4.1. описывалось, как модифицировать существующий диалог, используя запрос, обновляющий текущий адрес удаленного пользователя (remote target). В этом параграфе рассматривается вопрос модификации действующего сеанса. Такая модификация может предусматривать изменение адресов или портов, добавление или удаление медиапотока и т.д. Это выполняется путем передачи запроса INVITE в рамках того же диалога, что установил сеанс связи. Запрос INVITE, передаваемый в рамках существующего диалога, называется re-INVITE. Заметим, что одно сообщение re-INVITE может одновременно модифицировать и диалог, и параметры сеанса. Модифицировать сеанс способен как вызывавший, так и вызванный пользователь.

4.11.1. Действия UAC

Механизм offer/answer, используемый для обмена информацией описания сеанса при работе с запросом INVITE, применяется и для запроса re-INVITE. UAC, который, к примеру, хочет создать новый медиапоток, формирует новое предложение offer, описывающее параметры этого медиапотока, и передает его в сообщении INVITE другому участнику сеанса. Важно заметить, что передается полное описание сеанса, а не его изменение.

Конечно, UAC может передать re-INVITE без описания сеанса; в этом случае первый надежный ответ на re-INVITE, не информирующий об ошибке (класса 2xx), будет содержать offer.

Если формат описания сеанса предусматривает возможность нумерации версий – присвоения порядкового номера SDP-описанию, предложенному в ходе сеанса, – инициатор модификации должен указать, что версия описания сеанса изменилась.

Заголовки **To**, **From**, **Call-ID**, **CSeq**, и поле Request-URI формируются в соответствии с правилами для запросов в рамках существующего диалога.

UAC может решить не добавлять в запрос re-INVITE заголовок **Alert-Info** или тело сообщения со значением alert в заголовке **Content-Disposition**, поскольку UAS обычно не извещает пользователя о получении re-INVITE.

В отличие от INVITE, re-INVITE не может быть размножен при прохождении по сети и, следовательно, на него всегда приходит один окончательный ответ. Причиной является то, что поле Request-URI идентифицирует текущий адрес вызываемого пользователя, а не его списочный адрес.

Заметим, что UAC не должен инициировать новую INVITE-транзакцию в диалоге, пока протекает другая INVITE-транзакция в любом из двух направлении, а именно:

- Если существует действующая клиентская INVITE-транзакция, TU должен подождать, пока она перейдет в состояние «Completed» или «Terminated», а уже затем инициировать новый запрос INVITE.
- Если существует действующая серверная INVITE-транзакция, TU должен подождать, пока она перейдет в состояние «Confirmed» или «Terminated», и лишь потом инициировать новый запрос INVITE.

Однако во время выполнения INVITE-транзакции UA может инициировать транзакцию для запроса любого типа, кроме INVITE, ACK и CANCEL. И, наоборот, UA также может инициировать INVITE-транзакцию во время действия транзакции запросов любого типа, кроме INVITE, ACK и CANCEL.

Если UA получает на запрос re-INVITE окончательный ответ, отличный от класса 2xx, параметры сеанса останутся неизменными, как будто запрос re-INVITE не передавался. Если полученный ответ является 481 (Call/Transaction Does Not Exist) или 408 (Request Timeout) или если ответа на re-INVITE не получено (то есть от клиентской транзакции пришло уведомление о срабатывании таймера), UAC завершит диалог. Если UAC на запрос re-INVITE получает ответ с кодом 491, он должен запустить таймер со значением T, выбранным в соответствии с нижеописанным.

- Если именно UAC сформировал значение **Call-ID** идентификатора диалога, T выбирается случайным образом в интервале от 2,1 до 4 секунд с шагом 10 мс.
- Если значение **Call-ID** идентификатора диалога сформировал не UAC, то T выбирается случайным образом в интервале от 0 до 2 секунд с шагом 10 мс.

Когда таймер сработает, UAC должен предпринять еще одну попытку передать re-INVITE, если до этих пор сохранилась необходимость модификации сеанса. Например, в случае, если произошло завершение сеанса связи, сопровождающееся передачей сообщения BYE, re-INVITE не будет передан.

Правила для передачи запроса re-INVITE и для создания подтверждения ACK ответа 2xx на re-INVITE те же, что для начального INVITE.

4.11.2. Поведение UAS

UAS, получающий второй запрос INVITE до того, как в том же диалоге передан окончательный ответ на первый запрос INVITE с меньшим порядковым номером в заголовке **Cseq**, передает на второй INVITE ответ с кодом 500 (Server Internal Error) и включает в него заголовок **Retry-After** со случайно выбранной величиной в интервале от 0 до 10 секунд.

UAS, который получает второй запрос INVITE до прихода окончательного ответа на первый запрос INVITE, инициированный им самим, передает на второй

запрос ответ с кодом 491 (Request Pending). Если UAS получает re-INVITE для существующего диалога, он должен проверить все идентификаторы версии в описании сеанса, а при их отсутствии – содержимое описания сеанса, чтобы определить, изменилось оно или нет.

Если описание сеанса изменилось, UAS должен настроить соответствующие параметры, возможно, после запроса подтверждения пользователя. Если новое описание сеанса неприемлемо, UAS может отклонить его, передав ответ на re-INVITE с кодом 488 (Not Acceptable Here). Такой ответ должен содержать заголовок **Warning**.

Если UAS неоднократно передает ответ класса 2xx и ни разу не получает подтверждение ACK, он должен передать сообщение BYE для разрушения диалога.

UAS может не передавать ответ с кодом 180 (Ringing) на запрос re-INVITE, поскольку UAC обычно не передает эту информацию пользователю. По той же причине UAS может не использовать в ответе на re-INVITE заголовок **Alert-Info** или тело сообщения, для которого заголовок **Content-Disposition** имеет значение alert.

UAS, помещающий offer в ответ класса 2xx (из-за того, что re-INVITE не содержал offer), должен формировать offer так же, как если бы UAS создавал новый вызов, подчиняясь правилам передачи информации offer, обновляющей существующий сеанс в случае использования протокола SDP, как описано в [54]», RFC 3264. Это значит, что информация offer должна включать в себя столько форматов и типов медиа-информации, сколько поддерживает UA. UAS должен гарантировать, что описание сеанса частично совпадает с предыдущим описанием – форматами медиа-информации, типом транспортного протокола, или другими параметрами, которые поддерживаются UAC. Это делается, чтобы исключить возможность того, что описание сеанса будет отключено UAC. Если, все же, это описание не приемлемо для UAC, он должен сформировать answer с надлежащим описанием сеанса и затем передать BYE.

4.12. Процедуры разрушения сеансов связи

Этот параграф описывает процедуры прерывания сеансов связи, установленных по протоколу SIP. Состояние сеанса и состояние диалога очень тесно

связаны. Когда сеанс инициируется сообщением INVITE, каждый ответ 1xx или 2xx от отдельного UAS создает диалог, и, если ответ завершает обмен offer/answer, создает сеанс. В итоге, каждый сеанс связывается с одним диалогом. Если на начальный запрос INVITE придет окончательный ответ, отличный от 2xx, он разрушит все сеансы и диалоги (если они существовали), созданные посредством ответов на запрос. Завершая транзакцию, окончательный ответ не-2xx предотвращает также создание сеанса, как результата передачи запроса INVITE.

Запрос BYE используется для завершения отдельных сеансов. В этом случае отдельный сеанс – это сеанс с другим UA, участником диалога. Когда участнику диалога приходит сообщение BYE, все сеансы, связанные с этим диалогом, должны быть разрушены. UA не может передать BYE вне диалога. UA вызывающего пользователя может передать BYE как в установленном диалоге, так и в диалогах, находящихся на «ранней стадии», а UA вызываемого пользователя может передать BYE только в установленном диалоге. Однако UA вызываемого пользователя не должен передавать BYE до тех пор, пока не получит подтверждения ACK на свой ответ класса 2xx. Если нет расширений SIP, которые определяют другие состояния прикладного уровня, связанные с диалогом, сообщение BYE также разрушает диалог.

Влияние на диалоги и сеансы ответов на запрос INVITE, отличных от класса 2xx, делает привлекательным использование запроса CANCEL. Он вынуждает UAS передавать ответ не-2xx на запрос INVITE (в частности, ответ с кодом 487). Поэтому если в определенный момент UAC приходит к выводу об ошибочности попытки вызова, он вправе передать CANCEL. Если на INVITE приходит окончательные ответ (ответы) 2xx, это значит, что UAS принял приглашение установить связь до того, как к нему пришел CANCEL. В связи с этим UAC может продолжать работать в сеансах, установленных ответами класса 2xx, а может разорвать их передачей сообщения BYE.

Если пользователь кладет трубку, это обычно указывает на то, что он хочет завершить попытку установить сеанс связи и разрушить все существующие сеансы связи. Для UA вызывающего пользователя это будет подразумевать передачу сообщения CANCEL, если ответ 2xx на первоначальный INVITE еще не пришел, или передачу сообщения BYE, если окончательный ответ был принят. Для UA вызываемого пользователя это обычно подразумевает передачу сообщения BYE. Однако вызываемый пользователь может положить трубку до прихода подтверждения ACK ответа класса 2xx – тогда передача сообщения BYE будет удержана программным модулем SIP до прихода ACK в целях правильного завершения соединения.

Если определенный интерфейс пользователя позволяет отклонить вызов, не отвечая на него, то это выполняется передачей ответа с кодом 403 (Forbidden) или 486 (Busy Here); сообщение BYE в соответствии с приведенными выше правилами передано быть не может.

4.12.1. Разрушение сеанса с помощью запроса BYE. Работа UAC

Запрос BYE формируется так же, как любой другой запрос внутри диалога. Как только сообщение BYE сформировано, ядро UAC создает новую клиентскую не-INVITE-транзакцию и передает ей сформированный запрос. UAC должен считать сеанс разрушенным и, следовательно, закончить передавать и ожидать прихода медиа-информации, как только запрос BYE передан клиентской транзакции. Если, передав BYE, UAC получает ответ с кодом 481 (Call/Transaction Does Not Exist) или 408 (Request Timeout), или если ответа на BYE не получено вообще, UAC должен считать диалог и сеанс разрушенными.

4.12.2. Разрушение сеанса с помощью запроса BYE. Работа UAS

Сначала UAS обрабатывает запрос BYE с использованием общих процедур обработки. Ядро UAS, получившее запрос BYE, проверяет его соответствие существующему диалогу. Если запрос не соответствует диалогу, ядро UAS должно сформировать ответ с кодом 481 (Call/Transaction Does Not Exist) и передать его серверной транзакции. Это означает, что если UAC передал запрос BYE без параметров «tag» в соответствующих заголовках, UAS его отклоняет. Ядро UAS, получившее запрос BYE для существующего диалога, должно выполнить процедуры обработки запроса в диалоговом режиме. После этого UAS разрушает сеанс и заканчивает передавать и ожидать прихода медиа-информации. Ядро UAS формирует ответ класса 2xx и передает его серверной транзакции для дальнейшей доставки по сети.

Глава 5. Прокси-серверы SIP

5.1. Назначение прокси-сервера SIP

Прокси-серверы – это элементы сети SIP, которые маршрутизируют SIP-запросы к серверам агента пользователя и SIP-ответы – к клиентам агента пользователя. На пути к UAS запрос может проходить несколько прокси-серверов. Каждый из них будет принимать решение о дальнейшей маршрутизации, внося изменения в запрос перед его пересылкой следующему элементу сети. Ответы будут маршрутизироваться через ту же группу прокси-серверов, которая была пройдена запросами, но в обратном порядке.

Прокси-сервер – это логический элемент сети SIP. Когда приходит запрос, элемент, выполняющий роль прокси-сервера, первоначально решает, существует ли необходимость отвечать на запрос самостоятельно. Например, запрос может содержать ошибки, или прокси-сервер может нуждаться в аутентификации клиента для выполнения своих функций. Элемент может передать ответ с подходящим кодом ошибки. Отвечая на запрос непосредственно, SIP-элемент выполняет роль UAS и должен действовать в соответствии с общими правилами для UAS вне диалога.

Прокси-сервер может функционировать с сохранением состояний (stateful) и без сохранения состояний (stateless) для каждого нового запроса. Сервер без сохранения состояний работает как ретранслирующий узел сети. Он пересылает каждый запрос следующему элементу, принимая решение о маршрутизации на основе информации, содержащейся в запросе. Полученные ответы он просто возвращает обратно.

Прокси-серверы без сохранения состояний удаляют информацию о полученном сообщении, как только оно было ретранслировано. Прокси-серверы с сохранением состояний хранят информацию (состояние транзакции) о каждом входящем запросе и о каждом переданном запросе, возникающем вследствие обработки входящего запроса. Прокси-сервер с сохранением состояний может принять решение о размножении запроса, передавая его на несколько разных адресов, где может находиться вызываемый пользователь. Любой запрос, направляемый более чем на один SIP-элемент, должен обрабатываться с сохранением состояний.

В некоторых случаях прокси-сервер может пересылать запросы с использованием транспортного протокола с сохранением состояния (например, TCP), не используя возможность сохранения транзакционных состояний. Например, прокси-сервер может передавать запрос через определенное TCP-соединение без сохранения транзакционных состояний до тех пор, пока он способен поместить в сообщение объем информации, достаточный для того, чтобы ответ мог быть передан обратно через то соединение, по которому пришел запрос. Запросы, передающиеся с заменой транспортного протокола, когда TU прокси-сервера играет ведущую роль в обеспечении надежной доставки, должны передаваться с сохранением транзакционных состояний.

Stateful прокси-сервер может перейти в stateless режим в любой момент обработки запроса при условии, что он ранее не сделал ничего, что могло бы предотвратить переход в stateless режим (например, размножение запросов или создание ответа с кодом 100). При выполнении такого перехода вся информация о состоянии транзакций удаляется.

Большинство процедур обработки запроса в режимах stateless и stateful идентичны. Следующий раздел описывает действия прокси-сервера в режиме stateful. В разделе, описывающем режим stateless, будут указаны отличия от действий в режиме stateful.

5.2. Функции прокси-сервера с сохранением состояний

В режиме с сохранением состояний прокси-сервер действует как механизм обработки SIP-транзакций. При функционировании прокси-сервер использует серверную транзакцию и одну или несколько клиентских транзакций, которые связаны друг с другом посредством ядра прокси-сервера – компонента верхнего уровня, выполняющего обработку сигнальной информации (рис. 5.1). Входящие запросы обрабатываются серверной транзакцией. От серверной транзакции запросы направляются ядру прокси-сервера. Ядро прокси-сервера определяет, куда маршрутизировать запрос, выбирая одно или несколько мест для следующей пересылки запроса.

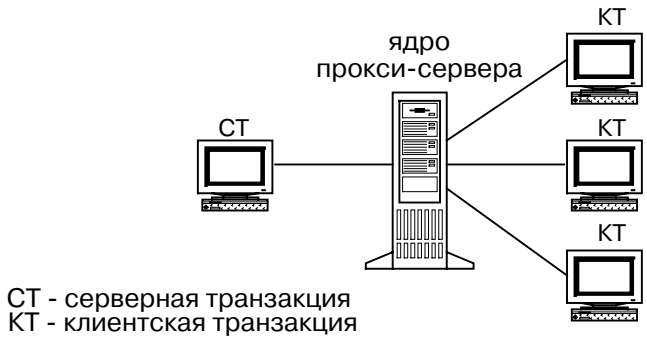


Рис. 5.1. Модель stateful прокси-сервера

Исходящий запрос, адресованный одному пользователю, но рассылаемый по нескольким направлениям, для каждого направления обрабатывается средствами связанной с ним клиентской транзакции. Ядро прокси-сервера собирает ответы клиентских транзакций и использует их для передачи ответов к серверной транзакции.

Stateful прокси-сервер создает новую серверную транзакцию для каждого полученного запроса. Все повторения передачи запроса будут поддерживаться этой серверной транзакцией. Ядро прокси-сервера должно действовать как UAS в отношении передачи предварительных ответов на эту серверную транзакцию (таких как 100 (Trying)). Таким образом, stateful прокси-сервер не должен создавать ответов с кодом 100 (Trying) на не-INVITE-запросы.

Для каждого запроса элемент, выполняющий роль прокси-сервера, должен выполнять следующие функции:

- проверку правильности составления запроса;
- предварительную обработку маршрутной информации;
- определение адреса (адресов) для пересылки;
- пересылку запроса;
- обработку всех ответов.

5.2.1. Проверка правильности составления запроса

Перед тем как прокси-сервер начнет работать с запросом, он должен проверить, правильно ли запрос составлен. Сообщение должно пройти:

- проверку корректности синтаксиса;
- проверку схемы URI;
- проверку заголовка **Max-Forwards**;
- проверку наличия замкнутого пути (опционально);
- проверку заголовка **Proxy-Require**;
- проверку заголовка **Proxy-Authorization**.

Если хотя бы одна проверка выявляет ошибку, элемент должен действовать как сервер агента пользователя и передать ответ с кодом ошибки.

От прокси-сервера не требуется обнаружение запросов, идентичных уже принятым, но не соответствующих его транзакции; он не должен расценивать такие запросы как ошибку. Оконечная точка, получив запросы, решит этот вопрос самостоятельно.

5.2.1.1. Проверка корректности синтаксиса

Чтобы запрос был поддержан серверной транзакцией, он должен быть правильно составлен. Все компоненты запроса, к которым применяются функции «Проверка правильности составления запроса» и «Пересылка запроса», должны быть сформированы корректно.

Все остальные компоненты, правильно составленные или нет, должны игнорироваться и оставаться без изменений при пересылке запроса. К примеру, прокси-сервер не отклонит запрос, если в нем содержится неверно составленный заголовок **Date**. Более того, прокси-сервер не будет удалять такой заголовок при пересылке запроса. Существующая версия протокола SIP предусматривает возможность дальнейшего расширения. Новые расширения могут определять новые типы запросов и заголовки. Прокси-сервер не должен отклонять запросы из-за того, что ему не известны их типы или заголовки.

5.2.1.2. Проверка схемы URI

Если поле Request-URI содержит URI, схема которого не понятна прокси-серверу, он должен отклонить запрос, передав ответ с кодом 416 (Unsupported URI Scheme).

5.2.1.3. Проверка заголовка Max-Forwards

Заголовок **Max-Forwards** используется для того, чтобы ограничить число узлов сети SIP, через которые может пройти запрос. Если запрос не содержит заголовка **Max-Forwards**, или запрос содержит заголовок **Max-Forwards**, и его значение больше нуля, то проверка успешно завершается. Если запрос содержит заголовок **Max-Forwards** и его значение равно нулю, прокси-сервер не должен пересылать запрос. Если тип принятого запроса – OPTIONS, прокси-сервер может функционировать как конечный получатель. В других случаях он передает ответ с кодом 483 (Too many hops).

5.2.1.4. Проверка наличия замкнутого пути

Прокси-сервер может проверить запрос на предмет наличия петли в маршруте его продвижения. Если запрос содержит заголовок **Via**, и в значении этого заголовка имя хоста и номер порта совпадают со значениями, помещенными данным прокси-сервером в предыдущие запросы, то запрос проходил через этот прокси-сервер ранее и движется либо по замкнутому пути, либо «по спирали». Во втором случае это будет запрос, который был маршрутизирован к прокси-серверу, прошел через него и спустя некоторое время снова вернулся на него измененным. Различие в поле Request-URI определит решение прокси-сервера о маршрутизации к другому месту назначения. Распознать замкнутый путь можно путем вычисления параметра «branch» сообщения (процедура вычисления будет

описана позже) и сравнения его с аналогичным параметром в заголовке **Via**. Если значения параметров совпадают, значит имеет место замкнутый путь, если они различны – запрос движется «по спирали» и, следовательно, обработка продолжается. При обнаружении замкнутого пути прокси-сервер может передать ответ с кодом 482 (Loop Detected).

5.2.1.5. Проверка заголовка Proxy-Require

Будущие расширения протокола SIP могут предоставлять новые функциональные возможности и требовать, чтобы прокси-сервер их также поддерживал. Оконечные точки включают заголовок **Proxy-Require** в состав запросов, использующих эти функции, предписывая прокси-серверу не обрабатывать запрос, если данные функции ему непонятны. Если запрос содержит заголовок **Proxy-Require** с одним или несколькими значениями, непонятными прокси-серверу, он должен передать ответ с кодом 420 (Bad Extension). Ответ должен включать в себя заголовок **Unsupported**, перечисляющий те идентификаторы option-tag, которые не были распознаны прокси-сервером.

5.2.1.6. Проверка заголовка Proxy-Authorization

Если прокси-сервер требует проведения аутентификации для передачи запроса, он должен просмотреть заголовок **Proxy-Authorization**; при отсутствии отклика аутентификации в заголовке запрос отклоняется, и передается ответ с кодом 407 (Proxy Authentication Required).

5.2.2. Предварительная обработка маршрутной информации

Прокси-сервер должен изучить поле Request-URI запроса. Если поле содержит значение, идентифицирующее данный прокси-сервер, он заменяет значение поля Request-URI последним значением заголовка **Route** с последующим удалением последнего значения заголовка **Route** (см. пример в § 5.5.2). Затем прокси-сервер продолжает работать так, как если бы он получил этот измененный запрос. Такое возможно, если SIP-элементом, передающим запрос прокси-серверу, является strict-router. Кроме того, механизм перезаписи позволяет сохранить значение поля Request-URI при прохождении запроса через strict-router.

Если адрес в Request-URI содержит параметр «maddr», прокси-сервер должен проверить, присутствует ли его значение в списке адресов или доменов,

за которые он отвечает, исходя из своей конфигурации. Если это так, и запрос был получен с использованием порта и транспортного протокола, указанных в значении поля Request-URI, прокси-сервер удаляет параметр «maddr» и параметры, содержащие номер порта и транспортный протокол, установленные не по умолчанию, и продолжает обработку так, как если бы эти значения отсутствовали. Запрос, который содержит значение параметра «maddr», соответствующее адресу прокси-сервера, может придти на порт или использовать транспортный протокол, отличные от указанных в URI. Такой запрос долен быть передан прокси-серверу с использованием указанного порта и транспортного протокола. Если первое значение заголовка **Route** запроса является адресом данного прокси-сервера, он должен удалить это значение.

5.2.3. Определение адресов пересылки

Прокси-сервер определяет адреса для дальнейшей передачи запроса. Перечень адресов (target set) для запроса будет либо назначен в содержимом запроса, либо получен при обращении к серверу определения местоположения.

Если Request-URI в запросе содержит параметр «maddr», Request-URI должен быть помещен в перечень, как единственный адрес, и прокси-сервер должен перейти к выполнению функции пересылки запроса. Если домен в Request-URI является доменом, за который данный прокси-сервер не несет ответственности, выполняются аналогичные действия. Существует множество условий, при которых прокси-сервер может получить запрос для пользователя, находящегося в домене, который обслуживается другим прокси-сервером. Прокси-сервер с функциями брандмауэра, поддерживающий исходящие вызовы (равно как и прокси-сервер HTTP, поддерживающий исходящие запросы), – один из примеров, когда это может произойти.

Если перечень адресов не был назначен, это подразумевает, что данный элемент ответственен за домен, указанный в Request-URI, и элемент может использовать любой механизм для того, чтобы определить, куда переслать запрос. Любой из этих механизмов базируется на обращении к серверу определения местоположения. Механизм может предусматривать получение информации от сервера определения местоположения, просмотр базы данных, обращение к серверу присутствия (presence server), использование других протоколов, или просто выполнение алгоритмической подстановки Request-URI. При обращении к сер-

веру определения местоположения Request-URI должен быть сперва приведен к каноническому виду для дальнейшего поиска в базе данных. Результат действия механизмов используется для формирования перечня адресов (target set). Если Request-URI не обеспечивает достаточной информации для того, чтобы определить перечень адресов, прокси-сервер передает ответ с кодом 485 (Ambiguous). Ответ должен содержать заголовок, содержащий URI новых адресов для следующих попыток. Например, запрос INVITE, адресованный на *sip*:*vladimir*@*protei.ru*, может быть неясным прокси-серверу, поскольку в базе данных услуги определения местоположения присутствует несколько адресов с именем пользователя Vladimir.

При формировании перечня адресов может использоваться информация о запросе, или содержащаяся в запросе, или сведения о текущем окружении прокси-сервера. Например, различные перечни адресов могут быть созданы в зависимости от содержимого или присутствия заголовков и тел сообщения, от времени поступления запроса, от интерфейса, на который пришел запрос, от ошибки предыдущего запроса и даже от текущего уровня загруженности прокси-сервера. Если перечисленные причины каким-либо образом обуславливают наличие определенных адресов, то эти адреса добавляются в перечень. URI могут быть помещены в перечень единожды.

Прокси-сервер не должен помещать в перечень дополнительные URI, если Request-URI оригинального запроса указывает ресурс, который обслуживается другим прокси-сервером. Прокси-сервер может изменить Request-URI запроса в процессе пересылки только тогда, когда он обслуживает домен, указанный в Request-URI. В противном случае прокси-сервер не будет добавлять в перечень контактные адреса из ответов класса 3xx или ответов с кодом 416, как это описано ниже.

Если Request-URI оригинального запроса указывает ресурс, за который данный элемент ответственен, прокси-сервер может продолжать добавлять адреса в перечень в процессе пересылки запроса. Для определения новых адресов, он может использовать любую информацию, полученную при обработке. Например, прокси-сервер может добавлять в перечень адресов контактные адреса, полученные в ответе перенаправления класса 3xx. Если прокси-сервер во время создания перечня адресов (к примеру, если он обращается к SIP-серверу регистрации) использует динамический источник информации, он должен следить за изменениями в источнике на протяжении всего времени обработки запроса. Поэтому, как только у пользователя появится новый контактный адрес, он сразу должен быть включен

в перечень адресов.

Если поле Request-URI указывает ресурс на прокси-сервере, которого не существует, он должен передать ответ с кодом 404 (Not Found). Если после проведения вышеперечисленных операций перечень адресов остается пуст, прокси-сервер должен передать ответ с кодом ошибки 480 (Temporarily Unavailable).

5.2.4. Пересылка запроса

Если перечень адресов не пуст, прокси-сервер может начать пересылку запроса. Stateful прокси-сервер может обрабатывать перечень в произвольной очередности. Он может обрабатывать URI последовательно, позволяя каждой клиентской транзакции завершиться перед началом следующей. Он может также создавать клиентские транзакции для каждого URI параллельно. Кроме того, прокси-сервер может произвольно разбивать перечень на группы, обрабатывая группы последовательно, а адреса в каждой из них – параллельно.

Простейшим механизмом упорядочения является механизм, основанный на использовании значения параметра «q», который получен из поля заголовка **Contact**. Адреса обрабатываются в очередности от большего значения параметра «q» к меньшему. Адреса, имеющие равные значения параметра, могут обрабатываться параллельно.

Stateful прокси-сервер должен иметь механизм для сохранения перечня адресов до прихода ответа и для связывания ответов на каждый переданный запрос с оригинальным запросом. Этот механизм реализуется через так называемый буфер ответов (response context, см. глоссарий). Он создается ядром прокси-сервера перед пересылкой первого запроса.

Для каждого адреса прокси-сервер передает запрос, последовательно выполняя следующие шаги:

- создает копию полученного запроса;
- обновляет содержимое поля Request-URI;
- обновляет содержимое заголовка **Max-Forwards**;
- опционально добавляет значение заголовка **Record-route**;

- опционально добавляет дополнительные заголовки;
- выполняет заключительную обработку маршрутной информации;
- определяет адрес, номер порта и тип транспортного протокола для пересылки;
- добавляет значение в заголовок **Via**;
- добавляет заголовок **Content-Length** (в случае необходимости);
- пересылает новый запрос;
- устанавливает таймер C.

5.2.4.1. Копирование запроса

Прокси-сервер начинает работу с копирования полученного запроса. Копия должна содержать все заголовки запроса, кроме того, в копии не должен быть нарушен порядок следования заголовков. Поля с одинаковым именем заголовка должны содержаться в том же порядке. Прокси-сервер не должен добавлять, модифицировать и удалять тела сообщения.

5.2.4.2. Обновление содержимого поля Request-URI

Значение поля Request-URI в стартовой строке копии должно быть заменено значением URI из перечня адресов. Если URI содержит параметры, запрещенные для Request-URI, они должны быть удалены. При некоторых обстоятельствах полученный Request-URI помещается в перечень адресов без изменений.

5.2.4.3. Обновление содержимого заголовка Max-Forwards

Если копия содержит заголовок **Max-Forwards**, прокси-сервер должен уменьшить его значение на единицу. В противном случае прокси-сервер сам добавляет заголовок со значением 70, поскольку некоторые существующие UA не помещают в запросы заголовок **Max-Forwards**.

5.2.4.4. Добавление значения Record-Route (опционально)

Если прокси-сервер требует, чтобы путь следования запросов диалога, который будет создан данным запросом, проходил через него, он должен поместить в копию свое значение заголовка **Record-Route** перед существующими значени-

ями заголовка, даже если заголовок **Route** уже присутствует. Запросы, устанавливающие диалог, могут содержать предустановленный заголовок **Route**.

Если запрос передается в режиме диалога, прокси-сервер помещает также свое значение заголовка **Record-Route**, если он желает, чтобы следующие запросы этого диалога прошли через него. Значения поля этого заголовка не окажут никакого влияния на маршруты route set, используемые оконечными терминалами. Прокси-сервер останется на пути следования запросов, если он решит не помещать значение **Record-Route** в запросы, которые передаются в режиме диалога. Однако при этом прокси-сервер будет удален из пути, когда конечная точка восстановит диалог после возможного сбоя. Прокси-сервер может поместить значение заголовка **Record-Route** в любой запрос. Если запрос не инициирует диалог, терминалы игнорируют это значение.

Каждый прокси-сервер на пути запросов принимает решение о добавление значения **Record-Route** независимо – наличие заголовка **Record-Route** в запросе не обязывает прокси-сервер добавлять свое значение.

URI, помещенный в качестве значения в заголовок **Record-Route**, должен быть SIP или SIPS URI. Этот URI должен содержать параметр «lr», обозначающий, что данный прокси-сервер поддерживает стандартные механизмы маршрутизации, определенные в [57]. Адрес прокси-сервера может быть разным для каждого направления, куда пересылается запрос. URI не должен содержать параметр, указывающий транспортный протокол, за исключением случая, когда прокси-сервер знает, что следующий элемент, находящийся на пути запросов, поддерживает этот тип транспортного протокола. URI, указанный прокси-сервером, используется другим SIP-узлом для того, чтобы принять решение о маршрутизации. Данный прокси-сервер не имеет средств для получения информации о функциональных возможностях другого элемента, поэтому при помещении своего значения в **Record-Route** он ограничивается обязательными компонентами: SIP URI и транспортный протокол – TCP или UDP. URI, помещаемый в поле **Record-Route**, должен однозначно определять прокси-сервер SIP, поместивший его, для того чтобы следующие запросы прошли через этот прокси-сервер.

Если Request-URI содержит SIPS URI или верхнее значение заголовка **Route** (после выполнения заключительной обработки, см. п. 5.2.4.6) содержит SIPS URI, то адрес, помещенный в заголовок **Record-Route,** также должен быть SIPS URI. Более того, если при этом запрос был получен не через TLS, прокси-сервер не

может не вставить заголовок **Record-Route**. Подобным образом прокси-сервер, который получает запрос через TLS, но создает запрос, не содержащий SIPS URI в поле Request-URI или в верхнем значении заголовка **Route** (после выполнения заключительной обработки), должен поместить заголовок **Record-Route** со значением не SIPS URI.

URI, помещаемый прокси-сервером в поле заголовка **Record-Route**, действителен только на время жизни диалога. Прокси-сервер, сохраняющий информацию о состоянии диалога, например, может отказать в приеме будущих запросов с таким URI в поле Request-URI после разрушения диалога. Прокси-сервер, не сохраняющий информацию о состоянии диалога, не знает, когда разрушается диалог, однако он может создать значение, хранящее некоторую информацию; это значение используется для сравнения с идентификатором диалога (dialog ID) будущих запросов, вследствие чего прокси-сервер может отклонять запросы, не соответствующие этой информации. Терминалы не должны использовать URI, полученные из заголовка **Record-Route** вне диалога, для которого он был предназначен. В разделе 4.1 дана более детальная информация о том, как конечные точки используют значения заголовка **Record-Route**.

Запись маршрута посредством заголовка **Record-Route** может быть востребована некоторыми услугами, когда прокси-сервер должен наблюдать за всеми сообщениями диалога. Однако это замедляет процесс обработки и ухудшает масштабируемость; таким образом, прокси-серверы должны вести запись маршрута, только если это требуется для услуги.

5.2.4.5. Добавление дополнительных заголовков

На данном этапе прокси-сервер может добавить в копию любые необходимые заголовки.

5.2.4.6. Заключительная обработка маршрутной информации

Прокси-сервер может вести внутреннюю политику, которая предусматривает, чтобы запрос прошел через группу прокси-серверов перед тем, как быть доставленным к месту назначения. Прокси-сервер должен гарантировать, что все эти серверы придерживаются стандартных процедур обработки заголовка **Route**, предусматривающих разделение места назначения запроса (содержащееся в Request-URI) и списка прокси-серверов на пути к месту назначения (содержащийся в заголовке **Route**). Это может быть достоверно известно, только если

прокси-серверы находятся в одном административном домене. Список прокси-серверов представлен перечнем URI, каждый из которых содержит параметр «lr». Перечень URI должен быть помещен в заголовок **Route** копии над существующими значениями (в случае их присутствия). Если заголовок **Route** отсутствует, он должен быть добавлен с содержащимся в нем перечнем URI.

Если прокси-сервер ведет внутреннюю политику, которая предписывает, чтобы запрос прошел через один определенный прокси-сервер, альтернативным вариантом помещения значения в заголовок **Route** является обход логики пересылки путем лишь передачи запроса на адрес, порт и по транспортному протоколу определенному прокси-серверу. Если запрос уже содержит заголовок **Route**, этот альтернативный вариант может быть использован, только если известно, что следующий прокси-сервер придерживается стандартных процедур обработки заголовка **Route**, как описано выше. Однако механизм введения значений в заголовок **Route** предпочтительнее данного подхода из-за своей надежности, гибкости, универсальности и последовательности действия. Более того, если поле Request-URI содержит SIPS URI, при взаимодействии с этим прокси-сервером должен быть использован протокол TLS. Если копия содержит заголовок **Route**, прокси-сервер должен проанализировать URI в его первом значении. Если он не содержит параметр «lr», прокси-сервер модифицирует копию следующим образом:

- помещает содержимое поля Request-URI в поле заголовка **Route**, в качестве последнего значения;
- затем помещает первое значение заголовка **Route** в поле Request-URI и удаляет его из заголовка **Route**.

Перемещение содержимого Request-URI в заголовок **Route** является частью механизма передачи информации в Request-URI при прохождении через strict-router. «Выталкивание» первого значения заголовка **Route** в поле Request-URI приводит сообщение в вид, в котором strict-router ожидает его получить (со своим собственным URI в поле Request-URI и URI следующего элемента в первом значении заголовка **Route**).

5.2.4.7. Определение адреса, порта и транспортного протокола для пересылки следующему элементу

Внутренняя политика прокси-сервера может предписывать пересылать запрос на определенный IP-адрес, порт и с использованием транспортного про-

токола независимо от значений в заголовке **Route** и поле Request-URI. Такая политика не должна использоваться, если прокси-серверу достоверно не известно, что IP-адрес, порт и тип транспортного протокола соответствуют серверу, придерживающемуся стандартных процедур обработки заголовка **Route**. Однако такой механизм пересылки запроса на определенном звене сети не является наилучшим; вместо него рекомендуется использовать описанный выше механизм работы с заголовком **Route**.

Если нет возможности воспользоваться им, прокси-сервер приступает к выполнению процедур поиска SIP-сервера, которому следует переслать запрос [56]. Если прокси-сервер модифицировал запрос для его пересылки на strict-router в соответствии с § 5.5.2, процедуры должны быть применены к значению в поле Request-URI. Если запрос не был изменен, процедуры применяются к первому значению заголовка **Route**. Выполнение процедур DNS-поиска даст упорядоченные наборы взаимосвязанных величин (адрес, порт, транспортный протокол). Независимо от того, какой тип URI будет использоваться при поиске, если Request-URI определяет SIPS-ресурс, то прокси-сервер будет выполнять процедуры поиска, как если бы исходным адресом для этого был SIPS URI. Прокси-сервер должен попытаться доставить сообщение в соответствии с первым набором взаимосвязанных величин, а в случае неудачи двигаться вниз по списку, производя новые попытки вплоть до получения успешного результата. Для каждого нового набора прокси-сервер форматирует в соответствии с ним сообщение и пересылает запрос, используя новую клиентскую транзакцию, как показано в следующих пп. 5.2.4.8.–5.2.4.10. Поскольку каждая попытка приводит к созданию новой транзакции, она представляет новую ветвь. Таким образом, параметр «branch», в поле заголовка **Via**, помещаемый туда на следующем, 8-м этапе, будет разным при разных попытках.

Если клиентская транзакция сообщает о невозможности передать запрос или об окончании времени ожидания окончательного ответа, прокси-сервер продолжает работу, используя следующий в списке набор взаимосвязанных величин. Если список заканчивается, это означает, что запрос не может быть передан этому серверу в списке адресов (target set). Прокси-сервер не помещает ничего в буфер ответов, но в остальном работает так, как если бы сервер передал окончательный ответ с кодом 408 (Request Timeout).

5.2.4.8. Добавление значения в заголовок Via

Прокси-сервер должен добавить значение в заголовок **Via** копии перед уже существующими значениями. Это подразумевает, что прокси-сервер подсчитает параметр «branch» для этой ветви, который будет уникальным для данной сети и будет содержать необходимую комбинацию «magic cookie».

Прокси-серверы, конфигурированные таким образом, чтобы обнаруживать петли, имеют дополнительное ограничение значения, которое они используют для формирования параметра «branch». Такой прокси-сервер в реализации должен создавать параметр «branch», разделенный на две части. Первая часть должна удовлетворять требованиям, описанным в 2.2.1, вторая – служит для того, чтобы отличать спирали от петель.

Обнаружение петель производится путем проверки изменения полей, влияющих на обработку вернувшегося на прокси-сервер запроса (если изменений нет, то замкнутый маршрут присутствует). Значение, помещенное во вторую часть параметра «branch», должно отражать все эти поля (включая поля заголовков **Route**, **Proxy-Require** и **Proxy-Authorization**). Это гарантирует, что если запрос был маршрутизирован обратно на прокси-сервер и одно из полей изменилось, он расценивается как прошедший по спирали, а не по петле. Общепринятый способ создания этого значения – подсчет криптографической контрольной суммы по алгоритму хэширования параметров «tag» заголовков **To** и **From**, заголовка **Call-ID**, поля Request-URI полученного запроса (перед преобразованием), верхнего заголовка **Via** и порядкового номера из заголовка **Cseq**. При подсчете контрольной суммы могут также учитываться значения заголовков **Proxy-Require** и **Proxy-Authorization** (в случае их присутствия). Алгоритм хэширования, используемый для подсчета контрольной суммы, зависит от реализации, но предпочтительным является алгоритм MD5 [49], формирующий строку в шестнадцатеричном виде.

Если прокси-сервер хочет обнаруживать петли, параметр «branch», который он добавляет, должен зависеть от всей информации, влияющей на обработку запроса, включая поле Request-URI (во входящем запросе) и любые заголовки, влияющие на прием запроса или маршрутизацию. Это необходимо для того, чтобы отличить запросы с петлей от запросов, чьи параметры маршрутизации изменились перед возвращением на этот сервер (запросы, проходящие по спирали). Тип запроса не должен использоваться при подсчете параметра «branch». В частности, запрос CANCEL и запрос ACK на ответ, отличный от класса 2xx, имеют

те же значения параметра, что и соответствующий запрос, который они отменяют или подтверждают. Параметр «branch» используется при корреляции этих запросов на сервере, обеспечивающем их обработку.

5.2.4.9. Добавление заголовка Content-Length (в случае необходимости)

Если запрос будет передан следующему элементу с использованием потоко-ориентированного транспортного протокола (например, TCP), и копия при этом не содержит заголовка **Content-Length**, прокси-сервер должен добавить его с правильным значением для тела запроса.

5.2.4.10. Пересылка запроса

Stateful прокси-сервер создает новую клиентскую транзакцию для этого запроса и дает указание транзакции переслать запрос, используя адрес, порт и тип транспортного протокола, определенные в п. 5.2.4.7.

5.2.4.11. Установка таймера C

На случай возникновения ситуации, при которой на запрос INVITE не приходит окончательного ответа, TU использует таймер C. Он запускается для каждой клиентской транзакции, когда запрос проходит через прокси-сервер. Значение таймера должно быть больше 3 минут. Параграф 5.2.5 описывает, как значение этого таймера обновляется с помощью предварительных ответов, а § 5.2.6 описывает обработку при его срабатывании.

5.2.5. Обработка ответов

При получении ответа прокси-сервер первым делом пытается определить клиентскую транзакцию, соответствующую этому ответу. Если прокси-сервер ее не находит, он обрабатывает ответ (даже если это информационный ответ) как stateless прокси-сервер. Если соответствие определено, ответ передается клиентской транзакции. Пересылка запросов, для которых не найдена клиентская транзакция (или любая информация о передаче запроса, связанного с этим ответом), улучшает надежность передачи. В частности, это гарантирует, что повторения передачи ответа класса 2xx на запрос INVITE ведутся правильно.

Как только клиентская транзакция передает ответы ядру прокси-сервера, вступают в силу следующие процедуры обработки:

- обнаружение буфера ответов;
- перезапуск таймера C при помощи предварительных ответов;
- удаление верхнего значения заголовка **Via**;
- добавление ответа в буфер ответов;
- проверка необходимости немедленной пересылки ответа;
- выбор наилучшего окончательного ответа из буфера ответов (при необходимости).

Если к тому времени, когда завершились все транзакции, связанные с одним буфером ответов, не было принято ни одного окончательного ответа, прокси-сервер должен выбрать и передать «наилучший» ответ из тех, которые были получены.

Следующие этапы обработки должны быть произведены каждым таким ответом:

- объединение значений в заголовке **Authorization** (при необходимости);
- перезапись значений заголовка **Record-Route** (опционально);
- пересылка ответа;
- создание необходимых запросов CANCEL.

5.2.5.1. Обнаружение буфера ответов

Прокси-сервер находит буфер ответов, который он создал перед пересылкой оригинального запроса. Остальные шаги обработки связаны с этим буфером ответов.

5.2.5.2. Перезапуск таймера C при помощи предварительных ответов

Для INVITE-транзакции, если ответ является предварительным с кодом от 101 по 199 включительно, прокси-сервер должен перезапустить таймер C клиентской транзакции. Таймер может быть перезапущен с любым значением, которое больше 3 минут.

5.2.5.3. Удаление верхнего значения заголовка Via

Прокси-сервер удаляет верхнее значение заголовка **Via** из ответа. Если в заголовке **Via** ответа не осталось значений, значит ответ предназначен этому элементу и не должен пересылаться.

Обработка, описанная далее, не производится, вместо этого выполняются стандартные процедуры обработки UAC ответов вне диалога (обработка транспортного уровня SIP уже выполнена).

5.2.5.4. Добавление ответа в буфер ответов

Полученные окончательные ответы сохраняются в буфере ответов. Любой из ответов в буфере может быть выбран в качестве «наилучшего» и передан на серверную транзакцию, связанную с этим буфером. Даже если ответ не был выбран, часть информации из него может потребоваться для формирования «наилучшего» ответа.

Если прокси-сервер решит передать запрос на часть адресов, указанных в заголовке **Contact** полученного ответа класса 3xx, путем добавления их в перечень адресов (target set), он должен удалить эти контактные адреса из ответа 3xx перед его сохранением в буфере. Однако если Request-URI оригинального запроса был SIPS URI, прокси-сервер не должен игнорировать контактные адреса со схемой, отличной от «sips». Если прокси-сервер решит передать запрос на все контактные адреса, полученные из ответа класса 3xx, он не должен добавлять в буфер этот ответ, как не содержащий контактных адресов. Удаление контактного адреса перед добавлением ответа в буфер предотвращает обращение следующего узла сети SIP по адресам, к которым данный прокси-сервер уже обращался.

Ответы класса 3xx могут одновременно содержать SIP, SIPS и не SIP URI. Прокси-сервер может передать запрос на ряд адресов со схемами «sip» и «sips» и поместить ответ с оставшимися контактными адресами в буфер ответов; впоследствии эта контактная информация, возможно, будет включена в пересылаемый окончательный ответе.

Если прокси-сервер получает ответ с кодом 416 (Unsupported URI Scheme) на запрос, у которого в поле Request-URI содержится адрес со схемой, отличной от «sip», но схема оригинального запроса была «sip» или «sips» (т.е. прокси-сервер при пересылке изменил схему адреса), прокси-сервер добавляет новый URI

в перечень адресов (target set). Этот URI должен быть адресом из поля Request-URI переданного запроса, но преобразованный под схему «sip». В случае использования схемы «tel» это выполняется путем перемещения части номера телефона из tel URL в пользовательскую часть SIP URI и указанием в части хоста SIP URI имени домена, куда был передан предыдущий запрос.

Так же, как и в случае с ответом класса 3xx, если прокси-сервер обращается по контактному SIP или SIPS URI, содержащемуся в ответе с кодом 416, этот ответ не добавляется в буфер ответов.

5.2.5.5. Проверка необходимости немедленной пересылки ответа

До тех пор пока серверной транзакцией прокси-сервера не получен окончательный ответ, следующие ответы должны пересылаться незамедлительно:

- Любые предварительные ответы, кроме ответа с кодом 100 (Trying).
- Любые ответы класса 2xx.

Если по одному из направлений получен ответ класа 6xx, он не пересылается сразу; вместо этого stateful прокси-сервер должен отменить все незавершенные клиентские транзакции и не должен создавать новые ветви для данного вызова. Для отмены клиентских транзакций прокси-сервер передает по соответствующим направлениям запрос CANCEL, что, как правило, приводит к получению ответов с кодом 487 от всех незавершенных клиентских транзакций, и только после этого передается ответ класса 6xx. После получения окончательного ответа серверной транзакцией должны незамедлительно пересылаться ответы класса 2xx на запрос INVITE.

Stateful прокси-сервер не должен незамедлительно пересылать прочие ответы. Они собираются в буфер ответов, как это описано в п. 5.2.4.4, для последующего выбора и передачи «наилучшего» ответа.

Любой ответ, требующий немедленной пересылки, должен быть обработан, как описано в пп. 5.2.4.7 и 5.2.4.8.

Этот шаг, совмещенный со следующим, гарантирует, что stateful прокси-сервер перешлет ровно один окончательный ответ на запрос не-INVITE, или ровно один ответ не-2xx, или один и более ответов 2xx на запрос INVITE.

5.2.5.6. Выбор «наилучшего» окончательного ответа из буфера ответов

Stateful прокси-сервер должен передать окончательный ответ на серверную транзакцию, которой соответствует определенный буфер ответов, если ни один окончательный ответ до этого момента не был передан, и все клиентские транзакции, связанные с этим буфером ответов, были завершены. Stateful прокси-сервер выбирает «наилучший» окончательный ответ среди тех, которые были получены и сохранены в буфере ответов.

Если в буфере ответов отсутствуют окончательные ответы, прокси-сервер должен передать серверной транзакции ответ с кодом 408 (Request Timeout). В противном случае прокси-сервер должен передать один из ответов, сохраненных в буфере ответов. Он должен выбрать один из ответов класса 6xx, если такие присутствуют в буфере ответов. Если ответы этого класса отсутствуют, прокси-сервер должен выбирать из ответов буфера ответов, класс которых имеет наименьшее значение. В пределах выбранного класса прокси-сервер должен давать преимущество тем ответам, которые доставляют информацию, влияющую на очередную передачу запроса, таким как 401, 407, 415, 420 и 484, если выбран класс 4xx.

Прокси-сервер, получивший ответ с кодом 503 (Service Unavailable), не должен пересылать его далее до тех пор, пока не сможет определить, что любые следующие запросы, которые он может переслать, также приведут к получению ответа с кодом 503. Другими словами, пересылка ответа с кодом 503 означает, что прокси-сервер не может обслужить ни один запрос, а не только запрос на адрес, который указан в Request-URI запроса, приведшего к созданию ответа с кодом 503. Если единственным полученным ответом является ответ с кодом 503, прокси-сервер должен создать и передать ответ с кодом 500 (Server Internal Error). К примеру, если прокси-сервер передал запрос на 4 адреса и получил ответы с кодами 503 (Service Unavailable), 407 (Proxy Authentication Required), 501 (Not Implemented) и 404 (Not Found), он выберет для передачи ответ с кодом 407. Пересылаемый ответ должен быть обработан, как описано в пп. 5.5.5.7 и 5.5.5.8.

Ответы класса 1xx и 2xx могут быть вовлечены в процесс организации диалогов. Когда запрос не содержит параметр «tag» в заголовке **To**, UAC использует параметр «tag» в ответе для того, чтобы различать ответы разных терминалов на запрос, инициирующий диалог. Прокси-сервер не должен помещать параметр «tag» в заголовок **To** ответа 1xx или 2xx, если запрос его не содержал. Прокси-сервер не должен модифицировать параметр «tag» в заголовке **To** ответа 1xx или 2xx.

Поскольку прокси-сервер не помещает параметр «tag» в заголовок **To** ответа класса 1xx на запрос, который не содержал «tag», он не может самостоятельно создавать предварительные ответы, кроме ответа 100 (Trying) (ответы 101–199 должны содержать параметр «tag» в заголовке **To**). Однако прокси-сервер может переслать запрос серверу агента пользователя. Этот UAS передает предварительные ответы, входя при этом в режим диалога с инициатором запроса на «ранней стадии».

Ответы класса с 3xx по 6xx доставляются последовательно, путем ретрансляции (hop-by-hop). Во время передачи такого ответа прокси-сервер работает как UAS, передающий ответ. Прокси-сервер не должен изменять «tag» в заголовке **To** при пересылке ответа 3xx – 6xx на запрос, который не содержал «tag». Прокси-сервер не должен изменять «tag» в любом пересылаемом ответе на запрос, содержащий «tag» в заголовке **To**.

Несмотря на то, что для ряда SIP-элементов не имеет значения, заменяет ли прокси-сервер «tag» в заголовке **To** пересылаемого ответа 3xx – 6xx, сохранение оригинального «tag» может способствовать поиску ошибок и неисправностей. Когда прокси-сервер при формировании окончательного ответа объединяет информацию из нескольких ответов, процесс выбора параметра «tag» заголовка **To** – произвольный, а создание нового «tag» для заголовка **To** может сделать процесс поиска ошибок и неисправностей проще. Это происходит, например, при объединении ответов с кодами 401 (Unauthorized) и 407 (Proxy Authentication Required), или при объединении значений заголовков **Contact** из незакодированных и неаутентифицированных ответов класса 3xx.

5.2.5.7. Объединение значений в заголовке Authorization

Если выбранный ответ является 401 (Unauthorized) или 407 (Proxy Authentication Required), прокси-сервер должен собрать все значения из заголовков **WWW-Authenticate** и **Proxy-Authenticate** всех остальных ответов с кодом 401 и 407, полученных до настоящего времени, в буфер ответов и перед пересылкой добавить их без изменений в этот ответ. Итоговый ответ 401 или 407 может содержать несколько значений в заголовках **WWW-Authenticate** и **Proxy-Authenticate**.

Это необходимо, поскольку некоторые или все направления, куда был переслан запрос, могут требовать аутентификации. Клиент должен получить все запросы аутентификации и при следующей передаче запроса снабдить его требуемой информацией.

5.2.5.8. Перезапись значений заголовка Record-Route

Если выбранный ответ содержит значение заголовка **Record-Route**, помещенное прокси-сервером при прохождении запроса, этот прокси-сервер может изменить значение перед пересылкой ответа. Это позволяет прокси-серверу иметь два разных адреса – для SIP-элементов, находящихся на расстоянии одной пересылки со стороны клиента и со стороны сервера.

Если прокси-сервер получил запрос через TLS, а передал его не через TLS-соединение, он должен перезаписать URI в заголовке ответа **Record-Route** так, чтобы он стал SIPS URI. Если прокси-сервер получил запрос не через TLS-соединение и передал его через TLS, прокси-сервер должен перезаписать URI в **Record-Route** ответа так, чтобы он стал SIP-URI.

Новый URI, помещенный в заголовок прокси-сервером, должен удовлетворять ограничениям, наложенным на значения заголовка **Record-Route** запросов (п. 2.4.1.4) с такими изменениями: URI не должен содержать параметра «transport» в случае, если прокси-сервер не знает, что следующий элемент сети SIP, находящийся на пути следующих запросов, поддерживает этот транспортный протокол.

Когда прокси-сервер принимает решение изменить значение заголовка **Record-Route** в ответе, одна из операций, которые он совершает, – поиск значения заголовка **Record-Route**, которое он поместил при пересылке запроса. Если запрос проходил по спирали, и прокси-сервер вставлял значение в заголовок **Record-Route** при каждом повторном прохождении, обнаружение нужного значения в ответе затруднительно. Для того чтобы процедура поиска проходила быстро и безошибочно, рекомендуется, чтобы в пользовательской части URI находилось значение, уникально идентифицирующее данный прокси-сервер. Когда приходит ответ, прокси-сервер изменяет значение заголовка **Record-Route**, идентификатор которого в пользовательской части URI соответствует этому прокси-серверу.

Запись значения заголовка **Record-Route** в запрос прокси-сервером не обязательно приведет к тому, что ответ будет включать в себя заголовок **Record-Route**. Если ответ содержит заголовок **Record-Route**, он будет содержать значение, добавленное прокси-сервером.

5.2.5.9. Пересылка ответа

Прокси-сервер не должен добавлять, изменять или удалять тело сообщения. Если это не определено иным способом, прокси-сервер не удаляет значения

заголовков, кроме значения заголовка **Via**, описанного в п. 5.2.4.3. В частности, прокси-сервер не должен удалять параметр «received», который он мог добавить к следующему значению заголовка **Via** во время обработки запроса, связанного с этим ответом. Прокси-сервер должен передать ответ серверной транзакции, связанной с буфером ответов. Это приведет к передаче ответа в соответствии с местонахождением, указанным в верхнем значении заголовка **Via**.

Если серверная транзакция не может быть больше доступна для передачи, прокси-сервер должен переслать ответ без сохранения состояний (statelessly), передав его серверной стороне транспортного уровня SIP. Серверная транзакция может информировать об ошибке при передаче ответа или сигнализировать об окончании времени ожидания окончательного ответа. Эти ошибки могут быть записаны в целях последующей диагностики.

Прокси-сервер должен сохранять буфер ответов до тех пор, пока не завершатся все связанные с ним транзакции, даже после пересылки окончательного ответа.

5.2.5.10. Создание запросов CANCEL

Если пересланный ответ был окончательным, прокси-сервер должен создать запрос CANCEL для всех незавершенных клиентских транзакций, связанных с буфером ответов. Аналогичные действия прокси-сервер предпринимает, когда получает ответ класса 6xx. Незавершенная клиентская транзакция – это транзакция, получившая предварительный ответ, но не получившая окончательного ответа (находится в состоянии «Proceeding») и не передавшая созданный для нее запрос CANCEL. Создание запроса CANCEL описано в п. 4.9.

Отмена незавершенных клиентских транзакций при пересылке окончательного ответа не гарантирует, что оконечная точка не получит на запрос INVITE ответы с кодом 200 (OK). Ответы 200 могут быть созданы более чем на одной ветви до того, как запрос CANCEL будет передан и обработан.

5.2.6. Обработка таймера C

Если таймер C сработает, прокси-сервер должен либо перезапустить таймер с любым выбранным значением, либо завершить клиентскую транзакцию. Если клиентская транзакция к этому времени получила предварительный ответ,

прокси-сервер создает запрос CANCEL, соответствующий этой транзакции. В противном случае прокси-сервер должен вести себя так же, как в случае получения транзакцией ответа с кодом 408 (Request Timeout).

Возможность перезапуска таймера позволяет прокси-серверу увеличить время жизни транзакции в зависимости от своего текущего состояния, например, при чрезмерной загрузке.

5.2.7. Обработка ошибок транспортного уровня SIP

Если транспортный уровень SIP уведомляет прокси-сервер об ошибке, когда тот пытается переслать запрос (см. § 10.7), прокси-сервер должен вести себя так, как если бы на пересылаемый запрос пришел ответ с кодом 503 (Service Unavailable). Если прокси-сервер извещается об ошибке при попытке переслать ответ, он отбрасывает ответ. После получения извещения прокси-сервер не должен отменять незавершенные клиентские транзакции, связанные с данным буфером ответов.

5.2.8. Обработка запроса CANCEL

Stateful прокси-сервер может сформировать запрос CANCEL, относящийся к любому другому запросу, созданному им ранее. Прокси-сервер должен отменить все незавершенные клиентские транзакции, связанные с буфером ответов, когда получает соответствующий запрос CANCEL.

Stateful прокси-сервер может создать запросы CANCEL для незавершенной клиентской INVITE-транзакции по истечении интервала времени, определенного в заголовке **Expires** запроса INVITE. Однако это часто не требуется, поскольку оконечные точки, участвующие в соединении, информируют друг друга об окончании транзакции самостоятельно без помощи прокси-сервера.

Если запрос CANCEL обрабатывается stateful прокси-сервером в рамках собственной серверной транзакции, буфер ответов для него не создается. Вместо этого ядро прокси-сервера ищет существующий буфер ответов для серверной транзакции запроса, отменяемого данным CANCEL. Если соответствующий буфер ответов найден, прокси-сервер должен незамедлительно передать ответ с кодом 200 (OK) на запрос CANCEL. В этом случае прокси-сервер выполняет функции

в соответствии с правилами для UAS, описанными в разделе 2.3. Более того, прокси-сервер должен создать запросы CANCEL для всех незавершенных клиентских транзакций, связанных с данным буфером ответов, как описано в п. 5.2.5.10.

Если буфер ответов не найден, прокси-сервер не в состоянии определить, какой запрос требуется отменить с помощью сообщения CANCEL. Поэтому он ретранслирует его без сохранения состояний (причиной того, что буфер ответов не был найден, может быть то, что прокси-сервер передал отменяемый запрос без сохранения состояний).

5.3. Функции прокси-сервера без сохранения состояний

В режиме без сохранения состояний прокси-сервер работает как простой ретранслятор сообщений. Большая часть процедур обработки, выполняемая stateful прокси-сервером, характерна для работы и stateless прокси-сервера. Отличия изложены ниже.

У stateless прокси-сервера нет никакого представления о механизме транзакций или о буфере ответов, который используется для описания поведения stateful прокси-сервера. Вместо этого stateless прокси-серверы принимают сообщения – запросы и ответы – прямо от транспортного уровня протокола SIP (см. главу 10). В результате, они не могут передавать повторные сообщения, созданные самостоятельно. Они только пересылают все повторные сообщения, которые получают; повторения сообщений пересылаются так же, как оригинальные сообщения, поскольку stateless прокси-серверы не различают их. Более того, при пересылке запроса без сохранения состояния прокси-сервер не должен самостоятельно создавать ответ с кодом 100 (Trying) или другие предварительные ответы.

Stateless прокси-сервер проверяет правильность составления запроса, как описано в § 5.2.1, и выполняет этапы обработки, описанные в § 5.2.2 и 5.2.3, со следующим исключением. Stateless прокси-сервер выбирает только один адрес из перечня адресов target set. Этот выбор должен зависеть только от информации, заложенной в сообщении, и статических свойств сервера. В частности, повторный запрос должен при обработке каждый раз передаваться по одному и тому же направлению. Кроме того, запрос ACK, не содержащий заголовка **Route**, и запрос CANCEL должны быть передан по тому же направлению, что и связанный с ними запрос INVITE.

Stateless прокси-сервер выполняет этапы обработки, описанные в § 5.2.4, со следующим исключением. Требования для уникальных идентификаторов, содержащихся в параметре «branch», применимы также к stateless прокси-серверам. Однако, stateless прокси-сервер не может просто использовать генератор случайных чисел для того, чтобы подсчитать первую составную часть параметра «branch», как описано в п. 5.2.4.8. Это происходит из-за того, что повторные запросы должны иметь то же значение параметра, а stateless прокси-сервер не в силах отличить повторный запрос от оригинального. Для того чтобы компонент параметра «branch», который делает его уникальным, был одинаков при каждой повторной пересылке запроса, для stateless прокси-сервера параметр «branch» должен быть подсчитан как комбинаторная функция тех параметров сообщения, которые при повторной передаче не изменяются. Stateless прокси-сервер может использовать любой механизм для того, чтобы гарантировать уникальность значения параметра «branch» среди существующих транзакций. Однако рекомендуется следующая процедура. Прокси-сервер проверяет параметр «branch» в верхнем заголовке **Via** полученного запроса. Если он начинается с комбинации, называемой «magic cookie», то первая составная часть параметра «branch» исходящего запроса подсчитывается путем хэширования значения полученного параметра «branch». В противном случае первый компонент значения параметра «branch» подсчитывается путем хэширования верхнего заголовка **Via**, параметров «tag» заголовков **To** и **From**, заголовка **Call-ID**, порядкового номера из заголовка **CSeq** и поля Request-URI из полученного запроса. Одно из этих полей всегда будет изменяться при переходе от одной транзакции к следующей. Все остальные преобразования сообщения, определенные в § 5.2.4, имеют результатом аналогичное преобразование повторно переданного запроса. В частности, если прокси-сервер добавляет значение заголовка **Record-Route** или помещает значение в заголовок **Route**, он должен поместить те же значения в повторные запросы. Для параметра «branch» заголовка **Via** это подразумевает, что преобразования должны основываться на статической конфигурации прокси-сервера или на свойствах запроса, не изменяющихся при повторной передаче.

Stateless прокси-сервер определяет, куда переслать запрос так же, как это описано для stateful прокси-сервера в п. 5.2.4.10. Вместо того чтобы проходить через клиентскую транзакцию, запрос передается непосредственно транспортному уровню SIP. Так как stateless прокси-сервер должен пересылать повторно переданные запросы по одному и тому же направлению и добавлять идентичные параметры «branch» к каждому из них, он может использовать для этих подсчетов только информацию из самого сообщения и данные из статической конфигурации

прокси-сервера. Если состояние конфигурации не статично (например, если таблица маршрутизации обновляется), часть запросов, на которые могут повлиять изменения, могут быть не переданы в течение интервала времени, равного значению таймера транзакции до или после изменений. Способ обработки подверженных влиянию запросов в течение этого интервала зависит от реализации. Наиболее простое решение – пересылать их с сохранением транзакционных состояний.

Stateless прокси-серверы не должны выполнять специальную обработку запросов CANCEL. Они обрабатываются в соответствии с общими правилами, как остальные запросы. В частности, прокси-сервер выполняет обработку заголовка **Route** запроса CANCEL так же, как он делает это для любых других запросов. Обработка ответов, как описано в § 5.2.5, не производится прокси-серверами, работающими в режиме без сохранения состояний. Когда ответ приходит на stateless прокси-сервер, он должен проверить адрес и порт в верхнем значении заголовка **Via**. Если адрес соответствует значению, которое данный прокси-сервер поместил в предшествующие запросы, он должен удалить это значение из ответа и переслать результат на адрес, указанный в следующем значении заголовка **Via**. Прокси-сервер не должен добавлять, модифицировать или удалять тело сообщения. Если это не определено иначе, прокси-сервер не должен удалять никакие другие значения заголовков. Если адрес не соответствует прокси-серверу, сообщение должно быть отброшено.

5.4. Работа с заголовком Route и полем Request-URI

Действия, которые выполняет прокси-сервер с запросом, содержащим заголовок **Route**, выполняются по следующим шагам:

- Прокси-сервер анализирует поле Request-URI. Если оно указывает ресурс, обслуживаемый данным прокси-сервером, он заменяет его значениями URI, полученными в результате обращения к серверу определения местоположения. В противном случае прокси-сервер не изменяет значение в поле Request-URI.
- Прокси-сервер анализирует URI в верхнем значении заголовка **Route**. Если URI определяет данный прокси-сервер, он удаляет это значение из заголовка **Route**.

- Прокси-сервер отошлет запрос на ресурс, указанный в верхнем значении заголовка **Route** или в поле Request-URI при отсутствии заголовка **Route**. Прокси-сервер определяет используемый при пересылке адрес, порт и транспортный протокол с помощью применения к данному URI процедур определения местоположения сервера [56].

Если на пути запроса не встречается **strict-router**, поле Request-URI всегда будет отражать URI адресата запроса.

5.5. Примеры

5.5.1. Взаимодействие через исходящий и входящий прокси-серверы

Данный сценарий (рис. 5.2) является примером обмена сообщениями через два прокси-сервера: исходящий и входящий, в котором прокси-серверы записывают свои значения в заголовок **Record-Route**. Здесь U1 (Агент пользователя 1) передает прокси-серверу P1:

```
INVITE sip:vladimir@protei.ru SIP/2.0
Contact: sip:anton@u1.niits.ru
```

Рис. 5.2. Взаимодействие через исходящий и входящий прокси-серверы

P1 – исходящий прокси-сервер. P1 не отвечает за protei.ru, поэтому он отыскивает адрес прокси-сервера, обслуживающего protei.ru, с помощью DNS-процедур и передает запрос по полученному адресу. Он добавляет также значение заголовка **Record-Route**.

```
INVITE sip:vladimir@protei.ru SIP/2.0
Contact: sip:anton@u1.niits.ru
Record-Route: <sip:p1.niits.ru;lr>
```

P2 (входящий прокси-сервер) получает запрос. P2 обслуживает домен protei.ru, поэтому он обращается к серверу определения местонахождения и перезаписывает значение в поле Request-URI. Он добавляет также значение заголовка **Record-Route**. Заголовок **Route** отсутствует, поэтому новое значение Request-URI определяет, куда передать запрос:

```
INVITE sip:vladimir@u2.protei.ru SIP/2.0
Contact: sip:anton@u1.niits.ru
Record-Route: <sip:p2.protei.ru;lr>
Record-Route: <sip:p1.niits.ru;lr>
```

Вызываемый пользователь *Vladimir на u2.protei.ru* получает запрос и передает ответ с кодом 200 (OK):

```
SIP/2.0 200 OK
Contact: sip:vladimir@u2.protei.ru
Record-Route: <sip:p2.protei.ru;lr>
Record-Route: <sip:p1.niits.ru;lr>
```

Агент пользователя U2 присваивает компоненту состояния диалога remote target значение *sip:anton@u1.niits.ru*, и другому компоненту – route set – значение <*sip:p2.protei.ru;lr*>,<*sip:p1.niits.ru;lr*>.

Ответ пересылается от прокси-сервера 2 к прокси-серверу 1 и к Агенту пользователя 1. Теперь UA устанавливает свое значение remote target диалога –

```
sip:vladimir@u2.protei.ru,
```

и значение **route set** –

```
<sip:p1.niits.ru;lr>,<sip:p2.protei.ru;lr>
```

Так как все элементы в значении route set содержат параметр «lr», U1 может создать свой следующий запрос BYE:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
Route: <sip:p1.niits.ru;lr>,<sip:p2.protei.ru;lr>
```

Так же, как поступили бы другие элементы сети SIP (включая прокси-серверы), U1 устанавливает URI в верхнем значении заголовка **Route** с использованием

DNS-процедур, чтобы определить, куда передавать запрос. Запрос передается на P1. Прокси-сервер P1 определяет, что не является ответственным за ресурс, указанный в Request-URI, и поэтому не вносит изменений в это поле. Прокси-сервер обнаруживает, что его адрес является первым значением в заголовке **Route**, поэтому он удаляет это значение и пересылает запрос к P2:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
Route: <sip:p2.protei.ru;lr>
```

P2 определяет также, что он не является ответственным за ресурс, указанный в Request-URI (он ответственен за protei.ru, но не за u2.protei.ru), и поэтому не вносит изменений в это поле. Прокси-сервер обнаруживает свой адрес в первом значении заголовка **Route**, поэтому он удаляет это значение и пересылает запрос на u2.protei.ru на основании DNS-поиска по Request-URI:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
```

5.5.2. Прохождение сообщений через strict-router

В этом сценарии (рис. 5.3) диалог устанавливается через 4 прокси-сервера, каждый из которых добавляет свое значение в заголовок **Record-Route**. Третий прокси-сервер – strict-router.

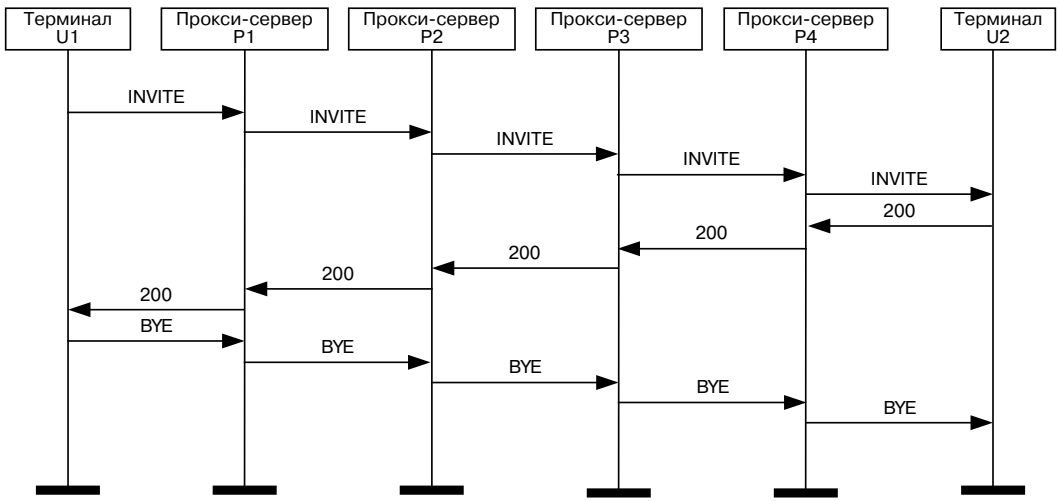


Рис. 5.3. Прохождение сообщений через strict-router

Запрос INVITE, полученный U2, содержит:

```
INVITE sip:vladimir@u2.protei.ru SIP/2.0
Contact: sip:anton@u1.niits.ru
Record-Route: <sip:p4.protei.ru;lr>
Record-Route: <sip:p3.loniis.ru>
Record-Route: <sip:p2.niits.ru;lr>
Record-Route: <sip:p1.niits.ru;lr>
```

U2 передает ответ с кодом 200 (OK). Позже U2 передает BYE прокси-серверу 4 на основании первого значения в заголовке **Route**.

```
BYE sip:anton@u1.niits.ru SIP/2.0
Route: <sip:p4.protei.ru;lr>
Route: <sip:p3.loniis.ru>
Route: <sip:p2.niits.ru;lr>
Route: <sip:p1.niits.ru;lr>
```

P4 не ответственен за ресурс, указанный в Request-URI, и поэтому не будет его изменять. Прокси-сервер обнаруживает, что его адрес является первым значением в заголовке **Route**, и удаляет это значение. Затем он готовится передавать запрос на основании нового первого значения заголовка **Route** – *sip:p3.loniis.ru*, но обнаруживает, что этот URI не содержит параметра «lr». Поэтому до передачи он переформатирует запрос следующим образом:

```
BYE sip:p3.loniis.ru SIP/2.0
Route: <sip:p2.niits.ru;lr>
Route: <sip:p1.niits.ru;lr>
Route: <sip:anton@u1.niits.ru>
```

Прокси-сервер 3 – strict-router; он пересылает запрос прокси-серверу 2 в следующем виде:

```
BYE sip:p2.niits.ru;lr SIP/2.0
Route: <sip:p1.niits.ru;lr>
Route: <sip:anton@u1.niits.ru>
```

P2 обнаруживает, что в Request-URI указано значение, записанное им ранее в заголовок **Record-Route**. Поэтому перед дальнейшей обработкой он производит перезапись значений:

```
BYE sip:anton@u1.niits.ru SIP/2.0
Route: <sip:p1.niits.ru;lr>
```

P2 не ответственен за *u1.niits.ru*, поэтому он пересылает запрос P1 на основании первого значения в заголовке **Route**.

P1 обнаруживает свой адрес в первом значении заголовка **Route**, поэтому он удаляет это значение, получая:

```
BYE sip:anton@u1.niits.ru SIP/2.0
```

Так как P1 не отвечает за u*1.niits.ru*, и заголовок **Route** отсутствует, P1 перешлет запрос на *u1.niits.ru* на основании Request-URI.

5.5.3. Прохождение сообщений через прокси-сервер с перезаписью значения в заголовке Record-Route

В приведенном на рис. 5.4 сценарии U1 и U2 находятся в разных частных пространствах имен; они устанавливают диалог через прокси-сервер P1, который работает как шлюз между пространствами имен.

U1 передает:

```
INVITE sip:vladimir@gateway.niits.ru SIP/2.0
Contact: <sip:anton@u1.niits.ru>
```

P1 обращается к серверу определения местоположения и передает к U2 следующее:

```
INVITE sip:vladimir@protei.ru SIP/2.0
Contact: <sip:anton@u1.niits.ru>
Record-Route: <sip:gateway.protei.ru;lr>
```

Рис. 5.4. Прохождение сообщений через прокси-сервер с перезаписью значения в заголовке Record-Route

U2 передает к P1 ответ с кодом 200 (OK):

```
SIP/2.0 200 OK
Contact: <sip:vladimir@u2.protei.ru>
Record-Route: <sip:gateway.protei.ru;lr>
```

P1 перезаписывает свое значение в заголовке **Record-Route**, чтобы обеспечить значение, которое будет полезно для U1, и передает к U1следующее:

```
SIP/2.0 200 OK
Contact: <sip:vladimir@u2.protei.ru>
Record-Route: <sip:gateway.niits.ru;lr>
```

Позже U1 передает к P1 следующий запрос BYE:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
Route: <sip:gateway.niits.ru;lr>
```

который P1 передает к U2 в виде:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
```

5.6. Назначение и функции сервера перенаправления

Для некоторых сетевых архитектур может оказаться полезным снизить загрузку прокси-серверов, которые отвечают за маршрутизацию запросов, и производить доставку начального сообщения, используя сервер перенаправления. Сервер перенаправления помещает информацию маршрутизации для поступившего запроса в ответ, предназначенный клиенту. Клиент, получивший перенаправляющий ответ от этого сервера, снова передает запрос, используя новый, только что полученный адрес (адреса).

Логическая структура сервера перенаправления предусматривает наличие уровня серверных транзакций и уровня пользователя транзакций, имеющего доступ к серверу определения местоположения – базе данных, в которой каждому URI соответствует один или более адресов, где может находиться пользователь, идентифицируемый представленным URI. Сервер перенаправления не может создавать своих запросов. После получения любого запроса, отличного от CANCEL, сервер перенаправления либо отклоняет запрос, либо обращается к серверу определения местоположения, который передает ему список альтернативных адресов; затем он передает клиенту ответ класса 3xx. Для правильно составленных запросов CANCEL он передает ответ класса 2xx. Окончательный ответ завершает эту SIP-транзакцию. На протяжении всей транзакции сервер перенаправления сохраняет ее состояние.

В перенаправляющем ответе класса 3xx на запрос присутствует заголовок **Contact**, в котором содержится список контактных адресов. Значения заголовка могут содержать параметр «expires», указывающий время действия контактного адреса. Указанные в **Contact** адреса характеризуют места, где может находиться

пользователь-адресат, или просто снабжают исходный адрес дополнительными транспортными параметрами. Например, ответ с кодом 301 (Moved Permanently) или 302 (Moved Temporarily) может сообщить адрес, совпадающий по местоположению с начальным, но доопределенный транспортными параметрами, указывающими другое имя сервера или multicast-адрес, которые должны быть использованы в новом запросе, или замену транспортного протокола UDP протоколом TCP (или наоборот).

Сервер перенаправления не должен переадресовывать запрос на URI, полностью идентичный начальному адресу, указанному в Request-URI запроса, поскольку это может явиться причиной возникновения петель. Сервер переадресации может либо отклонить запрос, передав ответ с кодом 404 (Not found), либо, если он совмещен с исходящим прокси-сервером, произвести пересылку запроса к месту назначения.

Заголовок **Contact** содержит URI, указывающие возможное текущее местоположение вызываемого пользователя, причем это могут быть не только SIP-адреса. Заголовок может включать в себя URI для телефона, факса, электронной почты. Значение заголовка **Contact** может направлять на ресурс, не имеющий связи с запрашиваемым. Например, для SIP-вызовов, связанных со шлюзом ТфОП, может оказаться необходимым доставить определенное информационное уведомление, такое как «Номер изменился».

Параметр «expires» заголовка **Contact** указывает время, в течение которого действителен контактный адрес. Значение параметра – целое число, указанное в секундах. В случае отсутствия параметра, его роль выполняет заголовок **Expires**. Не интерпретируемые значения по умолчанию равны *3600*.

Сервер перенаправления игнорирует все непонятные ему компоненты запроса, включая не интерпретируемые заголовки, любые идентификаторы функциональных возможностей option-tag в заголовке **Require** и даже типы запросов, и выполняет только функции перенаправления.

Глава 6. Процедуры HTTP-аутентификации

6.1. Аутентификация в SIP

Протокол SIP обеспечивает для аутентификации stateless-механизм на основе запроса аутентификации, который базируется на принципах аутентификации для протокола HTTP. Всегда, когда прокси-сервер или UA получает запрос (кроме исключений, описываемых ниже), он может потребовать от инициатора запроса удостоверить свою подлинность.

Для удостоверения подлинности в протоколе SIP определен только один механизм – аутентификация на базе схемы «Digest». Дело в том, что в соответствии с последней рекомендацией для протокола SIP [57] схема «Basic» ввиду своей недостаточной безопасности была запрещена для использования. Серверы не должны принимать значения отклика аутентификации, использующие схему аутентификации «Basic», а также не должны запрашивать аутентификацию с использованием этой схемы.

В протоколе SIP UAS использует ответ с кодом 401 (Unauthorized) для того, чтобы запросить аутентификацию UAC. Кроме того, registrar и сервер переадресации могут использовать для запроса аутентификации ответ с кодом 401 (Unauthorized), а прокси-серверы – нет; вместо этого они могут использовать ответ с кодом 407 (Proxy Authentication Required).

Во время прохождения процедуры аутентификации должна быть четко определена область аутентификации, т.е. та область, для доступа к которой клиент должен передать отклик аутентификации. Область указывается в поле realm запроса аутентификации, переданного сервером клиенту. Для удобства к значению поля realm предъявляются следующие требования:

- Значение realm должно быть уникально. Поле realm должно содержать имя хоста или имя домена.
- Значение realm должно представлять собой удобный для восприятия человеком идентификатор, который может быть отображен пользователю.

Например:

```
INVITE sip:vladimir@niits.ru SIP/2.0
Authorization: Digest realm=«niits.ru», <...>
```

Каждая область аутентификации имеет свой список имен пользователя (usernames) и паролей (passwords). Если сервер не требует аутентификации для определенного запроса, он может по умолчанию принять имя пользователя «anonymous» и не требовать пароля. Подобным образом, клиенты агента пользователя, представляющие многих пользователей, такие как шлюзы ТфОП, могут иметь свое собственное, зависящее от устройства имя пользователя и пароль для их области.

Когда UAC получает запрос подтверждения подлинности, он должен отобразить пользователю содержимое поля realm, содержащегося в нем, если программному модулю UAC не известно значение отклика аутентификации для области, указанной в запросе аутентификации.

Несмотря на то, что сервер может запрашивать проведение процедуры аутентификации при поступлении большинства SIP-запросов, существуют два запроса, требующие особого похода к решению задачи аутентификации, – ACK и CANCEL.

При использовании схемы аутентификации, использующей ответы для переноса запроса процедуры аутентификации (например, схемы «Digest»), могут возникнуть трудности для запросов, на которые не приходит ответ; на данное время таким запросом является ACK. По этой причине любое значение отклика аутентификации в запросе INVITE, который был принят сервером, должен быть принят тем же сервером и для подтверждения ACK. Клиенты агента пользователя, создающие

сообщение ACK, продублируют значения всех полей заголовков **Authorization** и **Proxy-Authorization** запроса INVITE, в которых содержатся отклики аутентификации. Серверы не должны пытаться запрашивать аутентификацию при получении подтверждения ACK.

Несмотря на то, что на сообщение CANCEL приходит ответ, серверы также не должны запрашивать значение отклика аутентификации при получении запросов этого типа, так как они не могут быть переданы заново.

Если сервер получает верное значение отклика аутентификации, процедура аутентификации успешно завершается. Если же значение отклика не является действительным, или сервер, требующий подтверждения подлинности, не принимает значение отклика по какой-либо другой причине, сервер может повторить свой запрос или передать ответ с кодом 403 (Forbidden). UAC не должен повторно пытаться передавать запрос со значением отклика аутентификации, которое было отклонено.

6.2. Процедуры аутентификации «пользователь-пользователь»

Когда UAS получает запрос от UAC, ему может потребоваться аутентифицировать инициатора запроса перед обработкой сообщения. Если значение отклика аутентификации отсутствует в заголовке **Authorization** запроса, UAS может запросить у отправителя подтверждение подлинности, отклонив запрос и передав ответ с кодом 401 (Unauthorized).

В ответное сообщение 401 (Unauthorized) должен быть включен заголовок **WWW-Authenticate**. Значение поля состоит, как минимум, из одного запроса аутентификации, который включает в себя схему (схемы) аутентификации и параметры, применимые для данной области аутентификации. Ниже представлен пример заголовка **WWW-Authenticate** в ответе с кодом 401.

```
WWW-Authenticate: Digest
realm=«niits.ru»,
qop=«auth,auth-int»,
nonce=«dcd98b7102dd2f0e8b11d0f600bfb0c093»,
opaque=«5ccc069c403ebaf9f0171e9517f40e41»
```

Когда инициирующий запрос UAC получает ответ с кодом 401 (Unauthorized), он, если это возможно, должен снова передать запрос, снабдив его надлежащим значением отклика аутентификации. UAC может потребовать от пользователя ввести свое имя и пароль. Как только значение отклика аутентификации будет получено (от пользователя, или обнаружено во внутреннем наборе ключей), UA должен занести в кэш-память значение отклика, соответствующее адресу в заголовке **To** и полю realm, и использовать это значение для следующих запросов.

Если для данной области не было найдено значения отклика, UAC может попытаться передать запрос, используя имя пользователя «anonymous» без пароля. Если значение отклика было обнаружено, UA, желающий себя аутентифицировать перед UAS или сервером регистрации (обычно, но необязательно, после получения ответа с кодом 401 (Unauthorized)), он может это сделать, добавив в запрос заголовок **Authorization**.

Заголовок **Authorization** содержит значение отклика аутентификации, о информациях аутентификации агента пользователя для области, в которой находится запрашиваемый ресурс, а также с параметрами, требуемыми для аутентификации и защиты от злонамеренного воздействия. Пример поля заголовка **Authorization** представлен ниже (для запроса REGISTER, направляемого на сервер регистрации).

```
Authorization: Digest username= «anton»,
realm=«niits.ru»,
nonce=«dcd98b7102dd2f0e8b11d0f600bfb0c093»,
uri=«sip:anton@niits.ru»,
qop=auth,
nc=00000001,
cnonce=«0a4f113b»,
response=«6629fae49393a05397450978507c4ef1»,
opaque=«5ccc069c403ebaf9f0171e9517f40e41»
```

После получения ответа, содержащего запрос аутентификации, UAC передает запрос, снабженный значением отклика аутентификации; он должен также увеличить значение в поле заголовка **CSeq**.

6.3. Процедуры аутентификации «прокси-сервер – пользователь»

Подобным образом, когда прокси-сервер получает запрос от UAC, он может потребовать проведения процедуры удостоверения подлинности, прежде чем приступить к обработке. Если значение отклика аутентификации отсутствует в заголовке **Proxy-Authorization** запроса, прокси-сервер может запросить у инициатора запроса подтверждение подлинности, отклонив запрос и передав ответ с кодом 407 (Proxy Authentication Required). В ответное сообщение прокси-сервер должен поместить заголовок **Proxy-Authenticate** с запросом аутентификации, содержащим схему (схемы) аутентификации и параметры, применимые для данной области аутентификации.

Обязанность UAC состоит в том, чтобы добавить заголовок **Proxy-Authorization** со значением, содержащим отклик аутентификации для области прокси-сервера, который желает удостоверить подлинность. Когда инициирующий запрос UAC получает ответ с кодом 407 (Proxy Authentication Required), он, если это возможно, должен снова передать запрос, снабдив его надлежащим значением отклика аутентификации. Он должен выполнять те же процедуры отображения пользователю поля realm так же, как указано в п. 6.2 при получении ответа с кодом 401.

Если значение отклика аутентификации для данной области не найдено, UAC может предпринять еще одну попытку – передать запрос с именем пользователя «anonymous» и без пароля.

UAC должен заносить в кэш-память значение отклика аутентификации, используемое в переданном запросе. Значение отклика будет включено во все запросы с идентичным значением заголовка **Call-ID**.

Значение заголовка **Proxy-Authorization** используется только тем прокси-сервером, чья область идентифицирована в поле realm. Когда используется последовательность нескольких прокси-серверов, значение заголовка **Proxy-Authorization** не удаляется прокси-сервером, чья область не соответствует полю realm. При получении размноженного запроса прокси-серверы могут запросить у UAC подтверждение подлинности. В этом случае прокси-сервер, размножающий запросы, должен сгруппировать все запросы аутентификации в одном ответе. Каждое значение заголовков **WWW-Authenticate** и **Proxy-Authenticate**,

полученное в ответах на размноженный запрос, должно быть помещено в единый ответ, передаваемый размножающим запросы прокси-сервером агенту пользователя. Очередность следования этих заголовков не имеет значения.

Прокси-сервер передает запрос только в случае, когда на его ответ с кодом 407 UAC передаст запрос, содержащий действительное значение отклика аутентификации. Прокси-сервер, размножающий запросы, может пересылать запрос одновременно нескольким прокси-серверам, требующим проведения процедуры аутентификации, которые, в свою очередь, не примут запрос, пока UAC не аутентифицирует себя в соответствующих им областях. Если UAC не передаст отклик аутентификации для каждого запроса аутентификации, запрос не дойдет до агента пользователя, где, возможно, находится адресат и, следовательно, эффективность размножения запросов в значительной степени понижается.

При передаче запроса, который должен содержать отклики аутентификации, UAC добавляет значение заголовка **Authorization** для каждого значения заголовка **WWW-Authenticate** и значение заголовка **Proxy-Authorization** для каждого значения заголовка **Proxy-Authenticate**. Оклики аутентификации в запросе должны быть различимы по значению поля realm. Не исключается возможность получения нескольких запросов аутентификации, связанных с одной областью, в одном ответе с кодом 401 (Unauthorized) или 407 (Proxy Authentication Required). Например, это может произойти, когда несколько прокси-серверов в пределах одного административного домена, использующих общее значение realm, получают размноженный запрос. Следовательно, при очередной попытке передать запрос UAC может поместить несколько откликов аутентификации в заголовок **Authorization** или **Proxy-Authorization** с одним и тем же значением поля realm. Для одной области должен быть использован один и тот же отклик аутентификации.

6.4. Схема аутентификации «Digest»

Этот параграф описывает схему аутентификации HTTP Digest, адаптированную для использования в SIP. Использование этой схемы в SIP практически полностью совпадает с применением ее в протоколе HTTP. Отличия состоят в следующем: URI может иметь как схему SIP, так и SIPS; при формировании значения nonce не может использоваться значение HTTP-заголовка **Etag**; невозможно применение операций буферизации, специфицированных для HTTP.

Рассмотрим пример проведения процедуры аутентификации «пользователь–пользователь» (рис. 6.1).

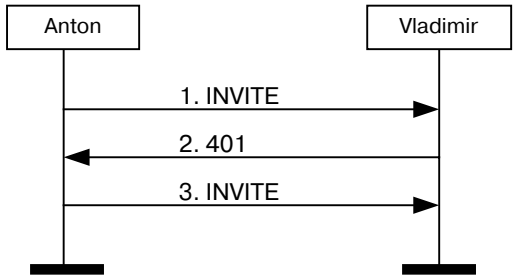


Рис. 6.1. Процедура аутентификации

Вызывающий пользователь Anton желает установить с пользователем Vladimir сеанс связи и передает ему запрос INVITE. UAS, конфигурация которого предусматривает контроль доступа, не находит в сообщении заголовка **Authorization** и поэтому передает агенту пользователя Anton ответ с кодом 401 (Unauthorized). В заголовке **WWW-Authenticate** ответа будет находится запрос аутентификации, содержащий, как минимум, схему аутентификации – Digest, поле realm, указывающее область аутентификации – niits.ru – и уникальное значение в поле nonce. UAC формирует комбинацию, включающую в себя имя пользователя, пароль, полученное значение nonce и другие компоненты, и преобразует ее с помощью хэш-функции в отклик аутентификации. Значение отклика помещается в заголовок **Authorization** запроса INVITE и заново передается UAS. Сервер примет запрос, самостоятельно подсчитает значение отклика и сравнит его с указанным в сообщении. Если значение отклика окажется действительным, UAS приступит к обработке запроса.

Рассмотрим состав заголовков, участвующих в процедуре аутентификации более подробно.

В поле заголовка **WWW-Authenticate** содержится запрос аутентификации (challenge), который включает в себя наименование схемы (Digest) и компоненты – realm, domain, nonce, opaque, stale, algorithm, qop. При этом необходимыми для аутентификации являются только realm и nonce.

- **realm**

Строка, которая отображается пользователям для того, чтобы они могли правильно выбрать имя пользователя (username) и пароль (password). Содержит имя хоста, который выполняет аутентификацию, или домена, для которого она проводится.

- **domain**

Заключенный в кавычки список URI, который определяет область защиты. Список сообщает клиенту о группе URI, которым может быть передана одинаковая информация аутентификации. Если поле domain отсутствует или существует, но не содержит значения, клиент должен понимать, что область защиты состоит только из адресов отвечающего сервера. Поле domain не имеет смысла в заголовке **Proxy-Authenticate**, для которого область защиты определяет конкретный прокси-сервер.

- **nonce**

Строка данных, сформированная сервером, которая уникально генерируется каждый раз при создании ответа с кодом 401. Строка может быть в шестнадцатеричном представлении или закодирована с помощью base64. Содержимое nonce зависит от реализации. Как правило, значение nonce обрабатывается при помощи алгоритма хэширования. Значение поля nonce копируется в заголовок **Authorization** последующего запроса; после получения запроса сервер пересчитывает значение nonce и сравнивает его с полученным в запросе. При несоответствии запрос отклоняется. Если в содержимое nonce входит метка времени (time-stamp), сервер имеет возможность ограничить время действия значения nonce. Значения nonce не прозрачно для клиента.

- **opaque**

Сформированная сервером строка данных, которая должна передаваться клиентом неизменной в заголовке **Authorization** следующих запросов, адресованных на URI, которые находятся в той же области защиты. Рекомендуется, чтобы строка была в шестнадцатеричном представлении или закодирована с помощью base64.

- **username**

Имя пользователя в указанной области аутентификации.

- **stale**

Флаг, указывающий, что предыдущий запрос клиента был отклонен, поскольку значение nonce было просрочено. Если флаг установлен в состояние «TRUE», клиент может повторить попытку передать сообщение без запроса у пользователя нового имени и пароля. Сервер присваивает флагу stale значение «TRUE» только в случае, когда он получает запрос, содержащий неверное значение nonce, но верное значение отклика (указывающее, что клиенту известны верные имя пользователя и пароль). Если значение флага – «FALSE», или любое другое, отличное от «TRUE», или поле stale отсутствует, это означает, что имя пользователя и/или пароль неверны, и поэтому должны быть введены новые значения.

- **algorithm**

Поле, указывающее доступный алгоритм хэширования. В случае отсутствия поля по умолчанию подразумевается значение «MD5». Если алгоритм не понят, запрос аутентификации должен быть игнорирован (и использован следующий запрос аутентификации, если в ответе присутствует несколько запросов отклика аутентификации).

- **qop**

Это поле является опциональным по причине обратной совместимости с более старой рекомендацией [17]. Qop – строка, заключенная в кавычки, которая указывает уровни «quality of protection» (качества защиты), поддерживаемые сервером. Значения «auth» предусматривает применение только процедуры аутентификации, значение «auth-int» – процедур аутентификации и защиты целостности сообщения. В заголовке **Authorization** содержится отклик аутентификации, который включает в себя наименование схемы аутентификации (Digest) и компоненты: username, realm, nonce, uri, response, algorithm, cnonce, opaque, qop, nc. При этом необходимыми являются username, realm, nonce, uri, response. Значения полей opaque и algorithm должны быть скопированы из заголовка **WWW-Authenticate** соответствующего ответа. Назначение realm, nonce, algorithm, opaque описано выше.

- **uri**

Содержит URI из поля Request-URI, входящего в состав Request-Line. URI дублирован здесь, поскольку при транспортировке запроса значение Request-URI может быть изменено прокси-серверами. В контексте протокола SIP эти два URI

могут предназначаться разным пользователям, в зависимости от действий определенного прокси-сервера при продвижении запроса.

- **response**

Строка, состоящая из 32 шестнадцатеричных разрядов и удостоверяющая, что пользователю известен пароль. Формируется с помощью применения функции хэширования к значениям nonce, nc, cnonce, qop, uri, username, realm, типу запроса и паролю password. По умолчанию хэширование производится по алгоритму MD5.

- **qop**

Указывает, какой уровень защиты клиент применил для своего сообщения. В случае присутствия, значение поля должно соответствовать одному из поддерживаемых вариантов, предложенных сервером в заголовке **WWW-Authenticate**. Это значение влияет на подсчет отклика аутентификации.

- **cnonce**

Значением поля cnonce является заключенная в кавычки строка, которая создается клиентом и используется клиентом и сервером для предотвращения атак, обеспечения взаимной аутентификации и некоторого уровня защиты целостности сообщения. Это поле должно присутствовать, если в запрос помещается поле qop, и отсутствует, когда заголовок **WWW-Authenticate** ответа не содержит поля qop.

- **nc**

Значение поля nc (nonce count) указывает число запросов (включая текущий) в шестнадцатеричном выражении, которые были переданы клиентом с использованием значения nonce, применяемом в данном запросе. Например, в первом запросе, передаваемом после получения ответа с новым значением nonce, поле nc будет выглядеть как «nc=00000001». Цель этого поля – дать возможность серверу обнаружить атаки воспроизведения (replay attack): если одно и тоже значение nc появляется дважды, сервер расценивает ситуацию как атаку воспроизведения. Это поле должно присутствовать, если в запрос помещается поле qop, и отсутствует, когда заголовок **WWW-Authenticate** ответа не содержит поля qop. Если одно из полей содержит неверное значение, или если требуемое поле отсутствует, передается ответ с кодом 400 (Bad Request). Если значение поля response неверно, указание об ошибке должно быть занесено в журнал регистрации, пос-

кольку повторные ошибки такого плана, произошедшие с одним и тем же клиентом, могут говорить о том, что злоумышленник пытается подобрать пароль.

В схеме аутентификации Digest применяется заголовок **Authentication-Info**. Заголовок **Authentication-Info** используется сервером в ответе для того, чтобы сообщить некоторую информацию, относящегося к процедуре аутентификации. Информация аутентификации представлена в нескольких полях: nextnonce, qop, rspauth, cnonce, nc. Обязательным является только поле nextnonce.

- **nextnonce**

Значение nextnonce – это строка nonce, которую сервер предписывает клиенту использовать в своем следующем отклике аутентификации. По сути, поле nextnonce является средством, реализующим выбор между использованием старого значения nonce или нового, измененного значения. Если клиент пренебрегает требованием сервера и не использует значение из поля nextnonce при создании содержимого поля заголовка **Authorization**, от сервера может придти запрос повторной аутентификации с флагом «stale=TRUE».

- **qop**

Указывает уровень защиты, примененный сервером к ответу. Сервер должен использовать то же значение, которое было передано в соответствующем запросе.

- **rspauth**

Поле rspauth содержит отклик аутентификации сервера и обеспечивает взаимную аутентификацию – сервер удостоверяет в ответе, что знает секретную информацию пользователя, а при использовании значения auth-int в поле qop обеспечивает также определенный уровень защиты целостности и ответа. Отклик аутентификации сервера подсчитывается для ответа так же, как формируется клиентом для запроса, за исключением того, что при хэшировании не используется значение типа запроса. Значения uri, cnonce, nc берутся из заголовка **Authorization** соответствующего запроса. Схема аутентификации Digest может также применяться для аутентификации типа пользователь – прокси-сервер, прокси-сервер – прокси-сервер, прокси-сервер – сервер. Для этого используются заголовки **Proxy-Authenticate** и **Proxy-Authorization**. Принцип действия механизма аутентификации с использованием **Proxy-Authenticate** и **Proxy-Authorization** аналогичен описанному выше механизму с применением заголовков **WWW-Authenticate** и **Authorization**.

Глава 7. Защита тела сообщения средствами S/MIME

7.1. S/MIME сертификаты

Прежде всего отметим, что в старом документе [36] для шифрования заголовков и тел сообщений протокола SIP рекомендовался механизм PGP, но этот механизм более не используется и поэтому здесь не рассматривается.

SIP-сообщения могут содержать тела сообщения в кодировке MIME. Стандарт MIME, в свою очередь, предусматривает механизмы защиты MIME-содержимого для гарантии целостности и конфиденциальности (включая multipart/signed и application/pkcs7-mime MIME-типы, подробно описанные в документах [20, 48]. Однако разработчики должны учитывать, что иногда могут существовать сетевые посредники (не типичные для протокола SIP прокси-серверы), в обязанности которых входит анализ и модификация тел SIP-сообщений (особенно, содержащих описание сеанса связи в формате SDP). Защита MIME-тел может помешать функционированию таких посредников, например, межсетевых экранов некоторых типов.

Для подписи или шифрования тел SIP-сообщений используются открытый и секретный ключи. Тела подписываются секретным ключом отправителя (который, в зависимости от ситуации, может поместить в сообщение открытый ключ), а шифруются тела открытым ключом определенного получателя. Отправители должны заранее обладать открытыми ключами получателей для шифрования тел сообщения. Открытые ключи могут храниться в наборе ключей UA.

Использование открытых ключей требует дополнительной их защиты и идентификации для определения связи с секретным ключом. Без такой дополнительной защиты злоумышленник может представить себя как отправителем подписанных данных, так и получателем зашифрованных данных, заменив значение открытого ключа или нарушив его идентификацию. Все это приводит к необходимости верификации открытого ключа. Для этих целей используется электронный сертификат.

Электронный сертификат представляет собой цифровой документ, который связывает открытый ключ с определенным пользователем или приложением. Для заверения электронного сертификата используется электронная цифровая подпись доверенного центра – центра сертификации. Используя открытый ключ центра сертификации, каждый пользователь может проверить достоверность электронного сертификата, выпущенного центром, и воспользоваться его содержимым.

Сертификаты, которые используются для идентификации конечного пользователя в S/MIME, отличаются от сертификатов, которые используются серверами, тем, что информация, удостоверяющая пользователя, является не именем хоста, а адресом пользователя. Этот адрес формируется путем связывания частей SIP или SIPS URI: «userinfo» (информация пользователя), «@» и «domainname» (имя домена) (например, *anton@niits.ru)*; в большинстве случаев он соответствует списочному адресу пользователя.

Каждый агент пользователя, который поддерживает S/MIME, должен иметь в своем распоряжении набор ключей, содержащий сертификаты конечных пользователей. Этот набор ключей устанавливает соответствие между списочными адресами и соответствующими им сертификатами. Каждый раз, когда пользователи используют в качестве URI инициатора вызова (находящегося в поле заголовка **From**) один и тот же списочный адрес, они должны использовать один и тот же сертификат.

Любые механизмы, зависящие от наличия сертификатов конечного пользователя, серьезно ограничены тем, что сегодня не существует единого сертификационного центра, который обеспечивал бы сертификаты для нужд пользователей. Тем не менее, пользователи могут получать сертификаты от известных сертификационных центров. В качестве альтернативы, пользователи могут создавать собственные сертификаты (self-signed certificates). Реализации также могут использовать предустановленные сертификаты в сетях SIP, где между всеми логическими элементами сети уже установлены доверительные отношения.

Для распространения сертификатов конечных пользователей существует несколько известных централизованных каталогов (сетевых справочников). Владелец сертификата может поместить свой сертификат в таком каталоге. В свою очередь, клиенты агента пользователя должны поддерживать механизм импортирования (вручную или автоматически) найденных в публичных каталогах сертификатов, которые соответствуют адресу места назначения SIP-запроса.

7.2. Обмен ключами S/MIME

Открытые ключи также могут быть переданы средствами протокола SIP. Для этой цели средствами S/MIME в тело сообщения помещается цифровая подпись, содержащая сертификат, который переносит открытый ключ пользователя, необходимый для верификации подписи.

Когда UAC передает вне диалога запрос, содержащий тело сообщения с цифровой подписью S/MIME, он должен структурировать тело сообщения, как состоящее из нескольких частей и содержащее цифровую подпись – multipart/signed. Если, помимо этого, пользователь желает зашифровать тело с использованием известного открытого ключа собеседника, UAC должен преобразовать тело сообщения в зашифрованное тело, которое будет заверено цифровой подписью.

Когда UAS получает запрос, содержащий тело сообщения с цифровой подписью, которая содержит сертификат, UAS первоначально должен проверить достоверность сертификата. Затем UAS обнаруживает информацию, удостоверяющую владельца сертификата, и сравнивает ее со значением заголовка **From** запроса. Если сертификат не может быть верифицирован, UAS должен уведомить своего пользователя о таком статусе сертификата и запросить его разрешение перед началом каких-либо действий. Если сертификат был успешно верифицирован, и информация, удостоверяющая владельца сертификата, соответствует значению в заголовке **From** запроса, или если пользователь (после уведомления) разрешает дальнейшие действия, то UAS должен добавить этот сертификат к внутреннему набору ключей; указателем сертификата в наборе будет служить списочный адрес владельца сертификата.

Когда на запрос, переданный вне диалога, UAS передает ответ, содержащий тело сообщения с цифровой подписью S/MIME, он так же, как UAC, должен

структурировать тело сообщения, как состоящее из нескольких частей и содержащее цифровую подпись – multipart/signed. Если, помимо этого, пользователь желает зашифровать тело с использованием известного открытого ключа собеседника, UAC должен преобразовать тело сообщения в зашифрованное тело, которое будет заверено цифровой подписью.

Когда UAC получает ответ, который содержит тело сообщения с цифровой подписью S/MIME, включающей в себя сертификат, UAC первоначально должен проверить достоверность сертификата. UAC должен также найти данные, удостоверяющие владельца сертификата, и сравнить их со значением заголовка **To** ответа; в случае отрицательного результата, UAC должен запросить разрешение пользователя для дальнейших действий. Если сертификат был успешно верифицирован, и информация, удостоверяющая его владельца, соответствует значению заголовка **To**, или если пользователь (после уведомления) разрешает использование сертификата, UAC должен добавить этот сертификат к своему внутреннему набору ключей. Если UAC еще не передал UAS собственного сертификата в предыдущих транзакциях, он должен поместить в свое следующее сообщение тело с цифровой подписью S/MIME, содержащей сертификат.

Впоследствии, когда UA получает запросы или ответы, которые содержат значение заголовка **From**, соответствующее значению в его наборе ключей, UA должен сравнивать сертификат, предложенный в этих сообщениях, с сертификатом, находящимся в его наборе ключей. Если обнаруживается расхождение, UA должен уведомить своего пользователя об изменении сертификата (предпочтительно сделать это так, чтобы указать, что, возможно, произошло нарушение защиты), и запросить разрешение пользователя перед тем, как продолжить процесс обработки сигнализации. Если пользователь, все же, принимает такой сертификат, он должен добавить его в набор ключей параллельно предыдущим значениям для этого списочного адреса.

Механизм обмена ключами не гарантирует безопасный обмен при использовании собственных сертификатов (self-signed) или сертификатов, подписанных неизвестным центром. Однако защита, обеспечиваемая этим механизмом, лучше, нежели отсутствие всякой защиты.

Если UA получает сообщение с телом, зашифрованным неизвестным открытым ключом, он должен отклонить запрос с помощью ответа с кодом 493 (Undecipherable).

Этот ответ должен содержать действительный сертификат отвечающей стороны (как правило, соответствующий списочному адресу, который указан в заголовке **To** отклоненного запроса); сертификат находится в теле сообщения, для которого значение параметра «smime-type» заголовка **Content-Type** – *certs-only*. Передача ответа с кодом 493 (Undecipherable) без сертификата сигнализирует о том, что отвечающая сторона не поддерживает шифрование средствами S/MIME.

Агент пользователя, который получает запрос, содержащий защищенное средствами S/MIME тело, для которого значение параметра «handling» в заголовке **Content-Disposition** – *required*, должен отклонить запрос, передав ответ с кодом 415 (Unsupported Media Type), если тип тела сообщения ему не понятен. Агент пользователя, получивший такой ответ, уведомляет своего пользователя о том, что удаленное устройство не поддерживает S/MIME. Затем пользователь может заново передать запрос без использования S/MIME. Однако следует иметь в виду, что ответ с кодом 415 может представлять собой атаку, направленную на снижение уровня безопасности.

Если агент пользователя получает на запрос, содержащий защищенное средствами S/MIME тело, ответ с незащищенным телом сообщения, UAC должен уведомить своего пользователя о том, что в ходе данного сеанса S/MIME не может быть использован. Подобным образом, если UA, который поддерживает S/MIME, получает запрос с незащищенным телом, он не должен передавать ответ с защищенным телом и должен уведомить своего пользователя, что в ходе данного сеанса S/MIME не может быть использован.

Все описанные условия требуют уведомлять пользователя о возникновении аномальных ситуаций, связанных с сертификатами. При этом пользователь должен осознавать, что неожиданные изменения в сертификате или отсутствие защиты, когда она ожидается, являются основанием для предупреждения, но не обязательно говорят о вмешательстве злоумышленника. Пользователи могут прервать любую попытку соединения или отказаться принять полученный запрос соединения; на языке телефонии это означает, что они могут положить трубку и произвести обратный вызов (call back). Заметим, что иногда пользователи вынуждены изменить свои сертификаты, например, когда они подозревают, что их закрытый ключ более не секретен. Когда это происходит, пользователи должны создать новые открытый и закрытый ключи и переустановить доверительные отношения со всеми пользователями, в распоряжении которых находится их старый открытый ключ.

Если в ходе диалога UA получает в теле сообщения сертификат с цифровой подписью, который не соответствует используемому сертификату, UA должен уведомить своего пользователя об этом, чтобы указать, что новый сертификат, возможно, передан злоумышленником.

7.3. Защита тела сообщения

В соответствии с S/MIME защищенные тела сообщения могут быть только двух типов – application/pkcs7-mime и multipart/signed. К телу сообщения любого типа могут быть применены механизмы обеспечения безопасности. При этом данное тело инкапсулируется в тело типа application/pkcs7-mime или как защищенная часть тела multipart/signed. Защищенное тело сообщения может быть зашифровано, иметь цифровую подпись, или цифровая подпись может быть передана для переноса сертификата. В приведенном ниже примере сообщение содержит тело типа application/pkcs7-mime, которое является зашифрованным телом типа application/sdp. Параметр «smime-type» заголовка **Content-Type** содержит значение enveloped-data, что означает, что к телу было применено шифрование. В примере текст, обрамленный точками (•••), зашифрован.

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKnashds8
To: Vladimir <sip:vladimir@protei.ru>
From: Anton <sip:anton@niits.ru> tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Contact: <sip:anton@pc33.niits.ru>
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
name=smime.p7m
Content-Disposition: attachment; filename=smime.p7m
handling=required

•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
•  Content-Type: application/sdp                                               •
•                                                                              •
•  v=0 o=anton 53655765 2353687637 IN IP4 pc33.niits.ru                        •
•  s=-                                                                         •
•  t=0 0                                                                       •
•  c=IN IP4 pc33.niits.ru                                                      •
•  m=audio 3456 RTP/AVP 0 1 3 99                                               •
•  a=rtpmap:0 PCMU/8000                                                        •
•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
```

В следующем примере тело в формате application/sdp снабжено цифровой подписью и помещено в виде части тела сообщения типа multipart/signed.

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKnashds8
To: Vladimir <vladimir@protei.ru>
From: Anton <anton@niits.ru>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Contact: <sip:anton@pc33.niits.ru>
Content-Type: multipart/signed;
protocol=«application/pkcs7-signature»;
micalg=sha1; boundary=boundary42

--boundary42
Content-Type: application/sdp

v=0
o=anton 2890844526 2890844526 IN IP4 pc33.niits.ru
s=Session SDP
c=IN IP4 pc33.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

--boundary42
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s;

handling=required
ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756

--boundary42-
```

7.4. SIP-туннелирование

Для обеспечения защиты SIP-заголовков S/MIME предусматривает инкапсуляцию сообщений SIP в тела сообщений типа message/sip с последующим применением к ним механизмов обеспечения безопасности так же, как это делается с обычными телами сообщений SIP. Эти инкапсулированные запросы и ответы SIP не создают отдельный диалог или транзакцию, они являются копией «внешнего»

сообщения, которая используется для контроля целостности или обеспечения дополнительной информации. Если UAS получает запрос, который содержит тело сообщения message/sip, он тоже помещает в ответ тело типа message/sip с инкапсулированным сообщением.

Заметим, что если в сообщении должны быть переданы, помимо защищенного, незащищенные тела сообщения, то тело типа message/sip может быть передано в качестве части тела типа multipart/mixed.

7.4.1. Обеспечение целостности SIP-сообщений

Туннелирование SIP сообщений средствами S/MIME может обеспечить целостность SIP-заголовков, которые отправитель желает защитить, продублировав их в теле типа message/sip, подписанном отдельной цифровой подписью. Любые тела сообщения, которые требуют защиты целостности, могут быть включены во «внутреннее» сообщение.

На приемном конце целостность заголовка должна быть определена путем сопоставления значения заголовка «внутреннего» сообщения со значением аналогичного заголовка во «внешнем» сообщении. Заголовками, которые могут быть изменены прокси-сервером, являются: **Via**, **Record-Route**, **Route**, **Max-Forwards**, и **Proxy-Authorization**; кроме того, может быть модифицировано поле Request-URI. Если эти заголовки претерпевают изменения при сквозной передаче, реализации не должны расценивать это как нарушение защиты. Изменения в других заголовках сигнализируют о нарушении целостности, и пользователи должны быть об этом извещены своими агентами UA. Если в сообщении с подписанным телом присутствует заголовок **Date**, UA, получивший сообщение, должен сравнить значение заголовка со своими внутренними часами. Если обнаруживается значительное расхождение во времени (порядка часа или более), UA должен предупредить пользователя о возможном нарушении защиты.

Если получателем обнаружено нарушение целостности сообщения, это сообщение должно быть отклонено, и должен быть передан ответ с кодом 403 (Forbidden), если это запрос, или же могут быть завершены все существующие диалоги. Ниже представлен пример использования тела типа message/sip:

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKnashds8
To: Vladimir <sip:vladimir@protei.ru>
```

```
From: Anton <sip:anton@niits.ru> ;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Fri, 30 Apr 2004 13:02:03 GMT
Contact: <sip:anton@pc33.niits.ru>
Content-Type: multipart/signed;
protocol=«application/pkcs7-signature»;
micalg=sha1; boundary=boundary42
Content-Length: 568

--boundary42
Content-Type: message/sip

INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKnashds8
To: Vladimir <vladimir@protei.ru>
From: Anton <anton@niits.ru>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Fri, 30 Apr 2004 13:02:03 GMT
Contact: <sip:anton@pc33.niits.ru>
Content-Type: application/sdp
Content-Length: 147

v=0
o=anton 2890844526 2890844526 IN IP4 pc33.niits.ru
s=Session SDP
c=IN IP4 pc33.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

--boundary42
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s;
handling=required

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756

--boundary42-
```

7.4.2. Шифрование при туннелировании

Существует возможность применить шифрование информации, находящейся в теле сообщения типа message/sip. На практике большинство заголовков требуется для прохождения сообщения по сети, поэтому основное назначение шифрования средствами S/MIME – защитить тела сообщения, например тела в формате SDP. Шифрование заголовков является второстепенной задачей.

Одним из возможных вариантов применения шифрования заголовков является функция выборочной анонимности. Запрос может быть сформирован таким образом, что заголовок **From** не содержит информации, удостоверяющей пользователя (к примеру, *sip:anonymous@anonymizer.invalid*). Однако второй заголовок **From**, содержащий подлинный списочный адрес инициатора, может быть зашифрован в теле сообщения типа message/sip, где он будет виден только участникам диалога.

Для того чтобы обеспечить сквозную целостность сообщений, зашифрованные тела типа message/sip должны быть подписаны пользователем. Для этого должно быть создано тело сообщения типа multipart/signed, которое включает в себя часть тела, содержащую зашифрованное тело сообщения типа message/sip, и часть тела, содержащую цифровую подпись. Обе части являются телами типа application/pkcs7-mime.

Ниже представлен пример зашифрованного и заверенного подписью сообщения; текст, обрамленный точками (•••), зашифрован.

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKnashds8
To: Vladimir <sip:vladimir@protei.ru>
From: Anonymous <sip:anonymous@niits.ru>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 314159 INVITE
Max-Forwards: 70
Date: Thu, 30 Apr 2004 13:02:03 GMT
Contact: <sip:pc33.niits.ru>
Content-Type: multipart/signed;
protocol=«application/pkcs7-signature»;
micalg=sha1; boundary=boundary42
Content-Length: 568

--boundary42
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;
name=smime.p7m
```

```
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7m
handling=required
Content-Length: 231
```

```
  Content-Type: message/sip

  INVITE sip:vladimir@protei.ru SIP/2.0
  Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKnashds8
  To: Vladimir <vladimir@protei.ru>
  From: Anton <anton@niits.ru>; tag=1928301774
  Call-ID: a84b4c76e66710
  CSeq: 314159 INVITE
  Max-Forwards: 70
  Date: Thu, 30 Apr 2004 13:02:03 GMT
  Contact: <sip:anton@pc33.niits.ru>
  Content-Type: application/sdp

  v=0
  o=anton 53655765 2353687637 IN IP4 pc33.niits.ru
  s=Session SDP
  t=0 0
  c=IN IP4 pc33.niits.ru
  m=audio 3456 RTP/AVP 0 1 3 99
  a=rtpmap:0 PCMU/8000
```

```
--boundary42
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s;
handling=required

ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfHfYT6
4VQpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756tbB9HGTrfvbnj
n8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfHfYT6ghyHhHUujpfyF4
7GhIGfHfYT64VQbnj756

--boundary42-
```

Глава 8. Процедуры обеспечения безопасности

Использование посредников, сложность доверительных отношений, использование протокола SIP между узлами при отсутствии доверительных отношений и работа в режиме пользователь-пользователь – все это сильно усложняет процедуры обеспечения безопасности. Для того чтобы обеспечить безопасность с учетом всех особенностей SIP, требуется несколько отдельных механизмов, применимых к разным аспектам протокола.

Обеспечение безопасности сигнализации SIP не опирается на безопасность протоколов, используемых совместно с SIP, таких как RTP, и не связано с процедурами обеспечения безопасности отдельных тел сообщений, поддерживаемым в SIP.

Ниже рассматриваются традиционные типы возможных угроз, исходя из которых определяются меры безопасности для SIP. Затем, в соответствии с несколькими механизмами обеспечения безопасности, детализируется комплекс процедур, требуемых для устранения этих угроз. Далее перечисляются требования для разработчиков оборудования вместе с типичными примерами реализаций, в которых эти механизмы могут быть использованы для улучшения безопасности SIP.

8.1. Типы угроз

В этом параграфе описываются некоторые типы угроз, которые являются общими для большинства реализаций сетей SIP. Они были выбраны специально, чтобы показать работу каждой процедуры обеспечения безопасности, требуемой для SIP.

Примеры демонстрируют полный список угроз для сетей SIP; точнее, они представляют собой «классические» угрозы, указывающие на необходимость применения определенных услуг обеспечения безопасности, которые потенциально могут предотвратить все категории угроз.

Эти угрозы приводят к возникновению условий, при которых злонамеренные пользователи могут производить чтение пакетов в сети, – подразумевается, что протокол SIP может быть использован в сетях общего пользования, например Интернет. Злонамеренные пользователя в сети могут также модифицировать пакеты (возможно, – на скомпрометированном посреднике). Злоумышленники могут заниматься кражей услуг, подслушиванием, разрушать сеансы связи.

8.1.1. Злоумышленная регистрация

Механизм регистрации SIP позволяет агенту пользователя идентифицировать себя серверу регистрации, как терминал, который в данный момент эксплуатирует пользователь (с присвоенным адресом). Registrar идентифицирует пользователя по информации, указанной в заголовке **From** сообщения REGISTER, чтобы определить, может ли этот запрос изменять контактные адреса, связанные со списочным адресом, приведенном в заголовке **To**. Несмотря на то, что зачастую эти два поля одинаковы, существуют реализации SIP, когда зарегистрировать контакты от имени пользователя может третья сторона.

Однако заголовок **From** SIP-запроса может быть произвольно изменен владельцем UA, и это открывает дверь злонамеренным регистрациям. Злонамеренный пользователь, который успешно имитирует сторону, уполномоченную на изменение контактных адресов, связанных со списочным адресом, например, удаляет из регистрации все существующие контактные адреса, а затем регистрирует свое собственное устройство в качестве надлежащего контактного адреса, тем самым направляя все запросы, адресованные пользователю, на свое устройство.

Эта угроза принадлежит семейству, которое основывается на отсутствии криптографической проверки инициатора запроса. Любой SIP UAS, который предоставляет услуги (например, шлюз, обеспечивающий межсетевой обмен SIP-запросов с традиционными телефонными вызовами), может производить контроль доступа к своим ресурсам путем аутентификации получаемых запросов. Даже UA конечного пользователя, например SIP-телефоны, заинтересованы в верификации инициатора запроса.

Этот тип угроз демонстрирует необходимость процедур обеспечения безопасности, которые дают возможность устройствам SIP аутентифицировать инициаторов запросов.

8.1.2. Имитация сервера

Имя домена, на которое передается запрос, как правило, указано в поле Request-URI. Агенты пользователя обычно связываются непосредственно с сервером в этом домене для того, чтобы доставить запрос. Однако всегда существует вероятность того, что злоумышленник может имитировать удаленный сервер, и запрос будет перехвачен.

Например, рассмотрим случай, когда сервер перенаправления в одном домене, loniis.ru, имитирует сервер перенаправления в другом домене, protei.ru. Агент пользователя передает запрос в protei.ru, но сервер перенаправления в loniis.ru передает фальшивый ответ, позаимствовав SIP-заголовки для ответа у protei.ru. Ложные контактные адреса в ответе перенаправления могут направить инициировавший запрос UA на несоответствующие или ненадежные ресурсы, или просто не допускать попадание запросов в protei.ru.

Это семейство угроз обширно, а многие из них являются критическими. В противоположность угрозе злонамеренной регистрации, рассмотрим случай, когда сообщение регистрации, отправленное в домен protei.ru, перехватывается сетевым элементом в loniis.ru, и тот передает ложный ответ с кодом 301 (Moved Permanently). Этот перенаправляющий ответ, который, как кажется, пришел из домена protei.ru, назначил узел в домене loniis.ru соответствующим сервером регистрации. Все будущие запросы REGISTER от данного UA, будут направляться в домен loniis.ru. Предотвращение подобной угрозы требует средств, с помощью которых агенты пользователя могли бы аутентифицировать серверы, которым они передают запросы.

8.1.3. Порча тела сообщения

SIP UA маршрутизируют запросы через прокси-серверы, которым они доверяют. Не обращая внимания на то, как устанавливаются доверительные отношения, UA может доверять прокси-серверу маршрутизировать запрос, но не изучать и не изменять тела, содержащиеся в запросе.

Рассмотрим UA, который использует тела SIP-сообщения для того, чтобы передать ключи шифрования мультимедийного сеанса. Несмотря на то, что он доверяет прокси-серверу домена в плане доставки сигнальной информации, для UA может быть нежелательно, чтобы администраторы домена были способны расшифровывать все последующие мультимедийные сеансы. Еще хуже, если прокси-сервер будет предпринимать активные действия, например, менять ключ сеанса, перехватывая его, или изменять характеристики безопасности, запрашиваемые инициирующим UA.

Это семейство угроз относится не только к ключам сеанса, но и ко всем возможным типам тел сообщения, передаваемым «насквозь» по протоколу SIP. Среди прочих, содержимым могут являться MIME-тела, которые должны быть отображены пользователю, SDP информация или инкапсулированные сигналы ТфОП. Злонамеренные пользователи могут пытаться изменить SDP-тела, например, для того, чтобы направлять RTP-медиапотоки на устройство прослушивания для подслушивания последующих переговоров.

Заметим также, что для некоторых заголовков в SIP чрезвычайно важна сквозная передача, например для заголовка **Subject**. Агенты пользователя могут защищать эти заголовки так же, как тела (например, злоумышленный посредник, изменяя заголовок **Subject**, может представить важный запрос, как спам). Однако поскольку многие заголовки правомерно просматриваются или изменяются прокси-серверами во время маршрутизации запроса, сквозная безопасность должна быть обеспечена не для всех заголовков.

По этим причинам UA может желать обеспечить сквозную безопасность для тел сообщения SIP и, в определенных случаях, – для заголовков. Услуги обеспечения безопасности, требуемые для тел, включают в себя обеспечение конфиденциальности, целостности и аутентификации. Эти сквозные услуги должны быть независимы от средств, используемых для обеспечения безопасности при взаимодействии с посредниками, такими как прокси-сервер.

8.1.4. Срыв сеансов связи

После установления диалога в результате обмена инициирующими сообщениями могут передаваться последующие запросы, изменяющие состояние диалога и/или сеанса. Необходимо, чтобы участники сеанса были уверены, что такие запросы не могут быть поделаны злоумышленниками.

Рассмотрим случай, когда третья сторона – злонамеренный пользователь – овладевает некоторыми инициирующими сообщениями в диалоге между двумя другими сторонами для того, чтобы узнать параметры сеанса («tag» заголовков **To** и **From** и так далее) и затем «вставляет» в сеанс запрос BYE. Злонамеренный пользователь подделает запрос так, чтобы существовала видимость, что он пришел от второго участника сеанса. Как только BYE достигнет места назначения, сеанс будет преждевременно разорван.

Подобные угрозы, возникающие во время сеанса, включают в себя передачу ложных запросов re-INVITE, видоизменяющих сеанс (возможно, для снижения его безопасности или для перенаправления медиапотоков в рамках атаки, направленной на прослушивание разговоров).

Наиболее эффективная контрмера против такой угрозы – это аутентификация отправителя запроса BYE. В рассматриваемом случае получатель должен знать только, что BYE пришел от той же стороны, с которой был установлен соответствующий диалог, для чего не требуется знать полную информацию, идентифицирующую отправителя. Кроме того, если злонамеренный пользователь не в состоянии узнать параметры сеанса из-за предпринятых мер обеспечения конфиденциальности, подделать запрос BYE будет невозможно. Однако некоторым посредникам (таким как прокси-серверы) требуется просматривать параметры установленного сеанса.

8.1.5. Отказ в обслуживании (DoS-атаки)

Атаки, вызывающие отказ в обслуживании (denial-of-service attacks), сосредоточены на приведении сетевого элемента в недоступное состояние, как правило, путем направления чрезмерного сетевого трафика на его интерфейсы. Распределенные атаки, вызывающие отказ в обслуживании, позволяют одному пользователю сети оказать влияние на несколько сетевых узлов так, чтобы они «затопили» целевой узел большим сетевым трафиком.

Во многих архитектурах SIP прокси-серверы выполняют функции взаимодействия с сетью общего пользования Интернет для того, чтобы принять запросы от IP-терминалов. Таким образом, протокол SIP создает возможности проведения распределенных атак, вызывающих отказ в обслуживании, что должно быть учтено разработчиками и операторами SIP-систем.

Злонамеренные пользователи могут создавать поддельные запросы, содержащие IP-адрес фальсифицированного источника и соответствующий заголовок **Via**, идентифицирующий хост, на который направлена атака, как инициатора запроса, а затем рассылать этот запрос большому числу сетевых SIP-элементов, тем самым используя SIP-агенты пользователя или прокси-серверы для создания чрезмерного трафика, направляемого на определенный узел.

Подобным образом злоумышленники могут использовать фальсифицированные значения заголовка **Route** запроса, которые идентифицируют целевой хост, и затем передавать такие запросы на прокси-серверы, которые, размножая запросы, увеличивают поток сообщений, направляемых по указанному адресу. Для этого может использоваться и заголовок **Record-Route**, тогда инициированный злонамеренным запросом диалог выльется в множественные транзакции, создаваемые в обратном направлении.

Часть атак denial-of-service становится возможной, если запросы REGISTER не аутентифицируются и не авторизуются должным образом серверами регистрации. Злонамеренные пользователи могут удалить регистрации некоторых или всех пользователей в административном домене, что повлечет за собой отсутствие у пользователей возможности быть приглашенными к установлению новых сеансов. Они могут также зарегистрировать большое число контактных адресов, определяющих один и тот же хост для определенного списочного адреса. Это делается для того, чтобы использовать registrar и связанные с ним прокси-серверы как средства увеличения трафика при данном типе атак. Злонамеренные пользователи могут также попытаться исчерпать доступный объем памяти и ресурсы диска сервера регистрации путем регистрации огромного количества связок.

Эти проблемы демонстрируют общую необходимость создания сетевых архитектур, минимизирующих риск возникновения угроз denial-of-service.

8.2. Механизмы обеспечения безопасности

Исходя из типов угроз, описанных выше, для протокола SIP требуются следующие основные услуги обеспечения безопасности: сохранение конфиденциальности и целостности при обмене сообщениями, предотвращение атак воспроизведения (replay-attacks) или получения доступа к сообщению путем фальсификации (message spoofing), обеспечение аутентификации и анонимности участников сеанса связи, предотвращение атак denial-of-service. Тела, находящиеся внутри SIP-сообщений, требуют применения услуг безопасности, обеспечивающих конфиденциальность, целостность и аутентификацию.

Вместо того, чтобы определять новые, специфические для SIP механизмы безопасности, SIP, по возможности, использует существующие методы, заимствованные из протоколов HTTP и SMTP.

Полное шифрование сообщений является наилучшим средством для сохранения конфиденциальности сигнальной информации, оно также может гарантировать, что сообщения не были модифицированы злонамеренными посредниками. Однако запросы и ответы **SIP** не могут быть просто полностью зашифрованы по сквозному принципу (end-to-end), потому что поля сообщения, такие как Request-URI, **Route** и **Via** должны оставаться видимыми для прокси-серверов в большинстве сетевых архитектур, чтобы SIP-запросы правильно маршрутизировались.

Прокси-серверы SIP должны также изменять некоторые заголовки сообщений (например, добавлять значения заголовка **Via**). Следовательно, прокси-серверы должны пользоваться какой-то степенью доверия SIP-агентов пользователя. Для этой цели рекомендуется использовать SIP-механизмы обеспечения безопасности низкого уровня, которые полностью зашифровывают SIP-запросы или ответы на уровне ретрансляционных участков. Это позволяет конечным точкам идентифицировать прокси-серверы, которым они передают запросы.

Логические объекты SIP тоже требуют взаимной идентификации в безопасной форме. Когда SIP-терминал передает информацию, удостоверяющую пользователя, агенту пользователя, находящемуся на другой стороне, или прокси-серверу, должны существовать механизмы, позволяющие произвести проверку этой информации. Чтобы удовлетворить этим требованиям, в протоколе SIP используется механизм криптографической аутентификации.

Кроме этого, существующий независимый механизм обеспечения безопасности для тел сообщений SIP предусматривает альтернативный способ сквозной взаимной аутентификации, а также предельную степень доверия посредникам агентов пользователя.

8.2.1. Безопасность транспортного и сетевого уровней

В сферу обеспечения безопасности на транспортном или сетевом уровне входит шифрование сигнального трафика и обеспечение конфиденциальности и целостности сообщения.

Существует два распространенных средства для обеспечения безопасности на транспортном и сетевом уровнях, это – TLS и IPSec, соответственно.

IPSec – это комплекс инструментальных средств протокола сетевого уровня, которые могут быть совместно использованы как дополнение традиционного протокола IP в области обеспечения безопасности. В большинстве случаев IPSec используется в архитектурах, где совокупности хостов или административных доменов имеют между собой доверительные отношения. IPSec обычно реализуется на хосте на уровне операционной системы или в маршрутизаторе, который обеспечивает конфиденциальность и целостность для всего трафика, получаемого с определенного интерфейса (как в архитектуре VPN). IPSec может быть также использован при последовательной передаче через ретрансляционные участки.

Во многих архитектурах IPSec не требует интеграции с SIP-приложениями. IPSec, возможно, более всего подходит для реализаций SIP, в которых добавление безопасности непосредственно к SIP-хостам было бы проблематичным. Участок сети между агентом пользователя и прокси-сервером, находящимся на расстоянии одной пересылки, также хорошо подходят для использования IPSec.

TLS обеспечивает безопасность на транспортном уровне поверх протоколов, ориентированных на соединение (например, TCP); «tls» (т.е. TLS по TCP) может быть определен, как желательный транспортный протокол, в значении заголовка **Via** или в SIP URI. TLS лучше всего подходит для архитектур, где необходимо обеспечить безопасность при последовательной передаче сообщений между хостами, где отсутствуют доверительные отношения. Например, UA пользователя Anton доверяет локальному прокси-серверу в своем домене, который после обмена сертификатами принимает решение

доверять локальному прокси-серверу в домене, где находится UA пользователя Vladimir. Vladimir, в свою очередь, доверяет своему локальному прокси-серверу – поэтому Vladimir и Anton могут безопасно общаться. Заметим, что транспортные механизмы для протокола SIP реализуются на основе последовательной передачи через ретрансляционные участки; поэтому UA, который передает запросы первому прокси-серверу с использованием TLS, не имеет гарантии, что TLS будет использоваться на протяжении всего пути сообщения.

8.2.2. Схема SIPS URI

Схема SIPS позволяет указывать для ресурсов, что они достижимы только при условии выполнения процедур обеспечения безопасности. Схема SIPS URI придерживается синтаксиса SIP URI, за исключением того, что поле типа схемы имеет значение «sips» вместо «sip». Однако семантика схемы SIPS сильно отличается от схемы SIP URI.

SIPS URI может использоваться как списочный адрес для определенного пользователя, т.е. быть адресом, который закреплен за пользователем и каждый раз помещается в заголовок **From** его запросов. Присутствие адреса со схемой SIPS URI в поле Request-URI запроса означает, что каждый ретрансляционный участок, через который пересылается запрос, должен быть защищен с помощью TLS. После того как этот запрос достигнет домена, в котором находится вызываемый пользователь, он пересылается UAS в соответствии с правилами внутренней безопасности (вполне возможно, с использованием TLS). При использовании инициатором запроса SIPS URI в качестве списочного адреса вызываемого пользователя схема SIPS предписывает, чтобы весь путь запроса до прокси-сервера, обслуживающего требуемый ресурс, был защищен.

Адреса со схемой SIPS могут быть использованы, помимо поля Request-URI запроса, во многих других случаях – в списочных адресах, в контактных адресах в заголовке **Contact** и в заголовке **Route**. В каждом из перечисленных случаев схема SIPS URI позволяет данным полям сообщения определять безопасные ресурсы.

При использовании схемы SIPS URI транспортный протокол (и, соответственно, параметр «transport») не зависит от того, используется TLS или нет, и поэтому оба адреса «*sips:anton@niits.ru*; *transport=tcp*» и «*sips:anton@niits.ru; transport=sctp*» верны. Протокол UDP не должен использоваться в качестве транспортного протокола при использовании схемы SIPS.

8.2.3. HTTP-аутентификация

Протокол SIP обеспечивает возможность запроса подтверждения подлинности, базирующуюся на HTTP-аутентификации. Это выражается в передаче ответов с кодом 401 и 407, включающих в себя заголовки, которые используются для переноса запросов аутентификации, с последующей передачей сообщений – запросов, заголовки которых содержат отклики аутентификации. Применение схемы аутентификации Digest с введением незначительных изменений для использования в протоколе SIP позволяет обеспечивать одностороннюю аутентификацию и защиту от атак воспроизведения.

Использование Digest-аутентификации в SIP описано в главе 6.

8.2.4. S/MIME

Как сказано выше, сквозное шифрование SIP-сообщений в целях конфиденциальности неприемлемо, поскольку сетевые посредники (такие как прокси-серверы) должны иметь доступ к определенным заголовкам для правильной маршрутизации сообщений; в противном случае SIP-сообщения будут не маршрутизируемы.

S/MIME позволяет SIP-агентам пользователя шифровать тела в кодировке MIME, обеспечивая сквозную безопасность этих тел без воздействия на заголовки сообщения. S/MIME может обеспечивать сквозную конфиденциальность и целостность тел сообщения, равно как и взаимную аутентификацию. Можно также использовать S/MIME, чтобы обеспечить целостность и конфиденциальность SIP-заголовков путем туннелирования сообщений SIP. Использование S/MIME в протоколе SIP описано в главе 7.

8.3. Реализация механизмов обеспечения безопасности

8.3.1. Требования для разработчиков оборудования SIP

Прокси-серверы, серверы перенаправления и регистрации должны реализовывать TLS и должны поддерживать взаимную и одностороннюю аутентификацию. Настоятельно рекомендуется, чтобы агенты пользователя могли инициировать TLS-соединения; агенты пользователя должны также иметь возможность принимать сообщения, переданные с использованием TLS. Прокси-серверы, серверы перенаправления и регистрации должны располагать сертификатами, в которых идентифицирующая их информация должна соответствовать их хост-именам. Агенты пользователя могут иметь собственные сертификаты для двусторонней аутентификации с использованием TLS. Все элементы сети SIP, которые поддерживают TLS, должны иметь механизм проверки достоверности сертификатов, полученных в ходе TLS-согласования. Это приводит к необходимости наличия одного или нескольких центров сертификации – широко известных организаций, занимающихся распределением сертификатов для узлов. Все SIP-элементы, которые поддерживают TLS, должны также поддерживать схему SIPS URI.

Прокси-серверы, серверы перенаправления, серверы регистрации и агенты пользователя могут реализовывать протокол IPSec или другие протоколы обеспечения безопасности низкого уровня.

Прокси-серверы, серверы перенаправления, серверы регистрации и агенты пользователя должны реализовывать аутентификацию по схеме Digest.

8.3.2. Решения по обеспечению безопасности

Совместное функционирование механизмов обеспечения безопасности может, в определенной степени, придерживаться существующих моделей безопасности Web и E-Mail. На верхнем уровне агенты пользователя аутентифицируют себя серверам (таким как прокси-сервер, сервер перенаправления и сервер регистрации), используя имя пользователя (username) и пароль (password). На транспортном уровне серверы аутентифицируют себя агентам пользователя или прокси-серверам, находящимся в радиусе одной пересылки, и, наоборот, UA и прокси-серверы аутентифицируют себя серверам, используя сертификат узла, доставленный средствами TLS.

На уровне взаимодействия оконечного оборудования аутентификация может быть выполнена средствами S/MIME, когда UA не доверяют элементам SIP-сети.

Ниже представлен пример, в котором эти механизмы обеспечения безопасности используются различными агентами пользователя и серверами для предотвращения угроз, описанных ранее.

8.3.2.1. Регистрация

При регистрации UA в своем локальном административном домене, он должен установить TLS-соединение со своим сервером регистрации. Registrar передает сертификат агенту пользователя. Узел, определяемый сертификатом, должен находится в пределах домена, в котором UA намеревается зарегистрироваться. Например, если UA намеревается выполнить регистрацию для списочного адреса «*anton@niits.ru*», сертификат узла должен идентифицировать хост, находящийся в домене niits.ru (такой как sip.niits.ru). После того как UA получает TLS-сообщение, содержащее сертификат, он производит проверку подлинности сертификата и определяет узел, обозначенный в сертификате. Если сертификат неверный, аннулированный или идентифицирует не подходящий узел, UA не должен передавать сообщение REGISTER и, следовательно, продолжать процесс регистрации. Когда registrar дает верный сертификат, UA знает, что registrar не является злоумышленником, который мог бы перенаправить UA на ложный ресурс, произвести кражу пароля аутентификации или попытаться предпринять другие подобные атаки.

Затем UA создает запрос REGISTER, содержащий в поле Request-URI адрес, который соответствует сертификату узла, полученному от сервера регистрации. После того как UA передал запрос REGISTER по существующему TLS-соединению, registrar должен запросить подтверждение подлинности запроса, передав ответ с кодом 401 (Proxy Authentication Required). Значение поля realm в заголовке **Proxy-Authenticate** ответа должно соответствовать домену, определенному из сертификата узла. Когда UAC получает запрос аутентификации, он должен либо обратиться к пользователю за значением отклика аутентификации, либо выбрать в наборе ключей значение отклика, соответствующее полю realm в запросе аутентификации. Имя пользователя (username) в значении отклика аутентификации должно соответствовать пользовательской части (userinfo) в URI, который содержится в заголовке **To** запроса REGISTER. После того как значение отклика помещается в соответствующий заголовок **Proxy-Authorization**, сообщение REGISTER снова передается серверу регистрации.

Так как registrar требует от агента пользователя аутентифицировать себя, для злоумышленника будет представлять трудность подделать запросы REGISTER для конкретного списочного адреса пользователя. Кроме того, поскольку сообщение REGISTER передается по конфиденциальному TLS-соединению, у злонамеренных пользователей не будет возможности перехватить запрос REGISTER, чтобы записать значение отклика аутентификации для атак воспроизведения (replay-атак).

Далее, после того как registrar выполнит регистрацию, UA должен оставить это соединение открытым при условии, что registrar функционирует также и как прокси-сервер, которому передаются запросы, предназначенные для пользователей в этом административном домене. Существующее TLS-соединение будет использовано для доставки входящих запросов агенту пользователя, который только что завершил процедуру регистрации. Поскольку UA уже аутентифицировал сервер на противоположной стороне TLS-соединения, все запросы, которые приходят по этому соединению, пройдут через прокси-сервер, и злоумышленники не могут создать поддельные запросы, которые могли бы быть пересланы через этот прокси-сервер.

8.3.2.2. Междоменные запросы

Теперь предположим, что агент пользователя Anton желает инициировать сеанс связи с пользователем в удаленном административном домене, имеющим списочный адрес *vladimir@protei.ru*. При этом локальный административный домен вызывающего пользователя (niits.ru) имеет локальный исходящий прокси-сервер.

Прокси-сервер, выполняющий прием входящих запросов для административного домена, может также работать как исходящий прокси-сервер; это допускается для niits.ru с целью простоты (в противном случае UA будет вынужден создать новое TLS-соединение, чтобы отделить сервер в этой точке). Предполагая, что клиент завершил процесс регистрации, описанный в предыдущем разделе, он должен использовать то же TLS-соединение с локальным прокси-сервером при передаче запроса INVITE другому пользователю. UA использует для сообщения INVITE отклик аутентификации, сохраненный в кэш-памяти, чтобы лишний раз без необходимости не обращаться к пользователю.

Когда локальный исходящий прокси-сервер аутентифицировал UA посредством отклика аутентификации в сообщении INVITE, он должен проанализировать содержимое поля Request-URI, чтобы определить, как сообщение должно мар-

шрутизироваться. Если часть имени домена в поле Request-URI соответствует локальному домену (niits.ru), а не домену protei.ru, тогда прокси-сервер обратится к серверу определения местоположения, чтобы определить наилучший способ для связи с запрашиваемым пользователем. Так, например, при обращении пользователя *anton@niits.ru* сервер определения местоположения сообщает контактный адрес *alexander@niits.ru*. Затем локальный прокси-сервер передаст запрос по TLS-соединению, установленному между пользователем Alexander и сервером регистрации во время процедуры регистрации. Поскольку Alexander получит этот запрос по его аутентифицированному каналу, он будет уверен, что запрос пользователя Anton был авторизован прокси-сервером локального административного домена.

Однако в рассматриваемом случае поле Request-URI определяет удаленный домен. Следовательно, локальный исходящий прокси-сервер в niits.ru должен установить TLS-соединение с удаленным прокси-сервером в домене protei.ru. Поскольку оба участника TLS-соединения являются серверами, располагающими сертификатами узла, должна быть произведена процедура взаимной аутентификации. Каждая сторона соединения должна проверить подлинность сертификата другой стороны и внимательно его проанализировать, сравнивая имя домена, обозначенное в сертификате, с заголовками SIP-сообщений. Например, прокси-сервер в niits.ru должен проверить на этом этапе, что сертификат, полученный от удаленной стороны, соответствует домену protei.ru. После того как это было выполнено, и TLS-согласование, выражающееся в создании безопасного канала между прокси-серверами, завершилось, прокси-сервер домена niits.ru готов к передаче запроса INVITE в домен protei.ru.

Прокси-сервер в protei.ru должен, в свою очередь, осмотреть сертификат прокси-сервера домена niits.ru и сравнить имя домена, указанное в сертификате, с частью имени домена адреса в заголовке **From** запроса INVITE. Прокси-сервер домена protei.ru может иметь жесткую политику безопасности, которая требует отклонять запросы, которые не соответствуют административному домену, откуда они были переданы. Такая политика безопасности может применяться, например, для предотвращения получения спама.

Однако такая политика гарантирует только то, что запрос пришел от домена, указанного в нем; это не позволяет домену protei.ru узнать, как niits.ru аутентифицировал пользователя Anton. Только если protei.ru будет иметь с помощью других средств доступ к информации аутентификации домена niits.ru, он,

вероятно, сможет узнать, как Anton себя идентифицировал. В определенных случаях в домене protei.ru может быть установлена еще более жесткая политика, которая запрещает прием запросов от доменов, которые не предоставляют информации аутентификации домену protei.ru.

После того как подлинность сообщения INVITE была установлена прокси-сервером домена protei.ru, он должен идентифицировать существующий TLS-канал (если таковой существует), связанный с пользователем, которому адресован запрос (в данном случае – *vladimir@protei.ru*). INVITE должен быть передан по этому каналу пользователю Vladimir. Поскольку запрос приходит по TLS-соединению, которое было ранее аутентифицировано как соединение с прокси-сервером protei.ru, Vladimir уверен, что содержимое заголовка не было подделано и что niits.ru проверил подлинность пользователя Anton, хотя не обязательно доверяет информации, идентифицирующей его.

Перед тем как переслать запрос, оба прокси-сервера должны добавить в него значение заголовка **Record-Route** так, чтобы все будущие запросы в этом диалоге прошли через эти прокси-серверы. Таким образом, прокси-серверы могут продолжать обеспечивать услуги безопасности на протяжении времени жизни всего диалога. Если прокси-серверы не добавляют свои значения заголовка **Record-Route**, будущие сообщения будут передаваться непосредственно между пользователями без применения каких-либо услуг обеспечения безопасности (если действующие стороны не договорились об использовании для обеспечения сквозной безопасности других средств, таких как S/MIME).

Злоумышленник, столкнувшись с такой архитектурой, не сможет, например, подделать запрос BYE и поместить его в поток сигнальной информации между Vladimir и Anton, поскольку не имеет возможности узнать параметры сеанса, а также потому, что механизмы обеспечения целостности защищают трафик между этими двумя пользователями.

8.3.2.3. Запросы при отсутствии локального прокси-сервера

В качестве альтернативы рассмотрим работу UA, который не имеет локального исходящего прокси-сервера. Пользователь UA – Alexander – идентифицируется адресом *alexander@loniis.ru*. Если Alexander хочет передать запрос INVITE по адресу *vladimir@protei.ru*, его UA должен инициировать TLS-соединение непосредственно с прокси-сервером protei.ru (используя механизмы, описанные в [56]).

Когда UA пользователя Alexander получает сертификат от прокси-сервера домена protei.ru, он должен быть проверен до того, как UA передаст свой запрос INVITE по TLS-соединению. UA пользователя Alexander не располагает средствами для проведения процедуры собственной аутентификации прокси-серверу в домене protei.ru, однако он имеет цифровую подпись, заверяющую тело сообщения типа message/sip (см. раздел 7.4) в сообщении INVITE. Прокси-сервер домена protei.ru также может иметь жесткую политику, касающуюся запросов, которые не содержат protei.ru в доменовой части адреса в заголовке **From**. На такие запросы прокси-сервер может даже не передавать запросы аутентификации – он расценивает этих пользователей как не аутентифицированных.

В отношении пользователя Vladimir прокси-сервер в домене protei.ru имеет следующую политику: все не аутентифицированные запросы перенаправляются на контактный адрес <sip:vladimir@192.0.2.4>, зарегистрированный для списочного адреса *vladimir@protei.ru*. Alexander получает ответ перенаправления вызова по TLS-соединению, которое он установил с прокси-сервером домена protei.ru; поэтому он доверяет достоверности контактного адреса.

Затем Alexander должен установить TCP-соединение по назначенному адресу и передать новое сообщение INVITE с полем Request-URI, содержащим полученный контактный адрес (пересчитав подпись, заверяющую тело сообщения). Vladimir получает INVITE через интерфейс, обозначенный как «небезопасный», но его UA анализирует и, в данном случае, опознает содержимое заголовка **From** запроса и затем сопоставляет локально буферизированный сертификат с сертификатом, присутствующим в подписи тела сообщения INVITE. Vladimir создает ответное сообщение, аутентифицируя себя подобным образом, и передает его пользователю Alexander ; после этого безопасный диалог вступает в силу.

Иногда межсетевой экран или NAT (Network Address Translation) в административном домене могут препятствовать установлению прямого TCP-соединения с UA. В этих случаях прокси-серверы могут пересылать запросы агентам пользователя, но уже без доверительных отношений, например, путем отказа от существующего TLS-соединения и передачи запроса по не защищенному TCP-соединению, как предписывает внутренняя политика.

8.3.2.4. Защита от DoS-атак

Для минимизации риска атак, направленных на отказы в обслуживании сервером, требуется выполнение следующих правил.

Когда узел, на котором функционирует прокси-сервер SIP, доступен для маршрутизации из сети общего пользования Интернет, он должен находиться в административном домене, обеспеченном защитными механизмами (блокирующими маршрутизированный от источника трафик, а также отфильтровывающими ping-трафик). Возможно также включение в архитектуру сети защитных хостов, установленных на границе административного домена, которые могут противостоять denial-of-service атакам, гарантируя, что SIP-хосты в пределах административного домена не будут «завалены» чрезмерным количеством сообщений.

Вне зависимости от того, какие решения обеспечения безопасности реализуются, чрезмерные потоки сообщений, направленные на прокси-серверы, могут привести к закрытию ресурсов прокси-сервера; соответственно, это помешает трафику достичь места своего назначения. Затраты вычислительной мощности на прокси-сервере связаны с обработкой SIP-транзакций; эти затраты для stateful прокси-серверов выше, чем для stateless прокси-серверов. Следовательно, stateful прокси-серверы более чувствительны к перегрузке, чем stateless прокси-серверы.

Агенты пользователя и прокси-серверы должны запрашивать аутентификацию отправителей запросов путем передачи только одного ответа с кодом 401 (Unauthorized) или 407 (Proxy Authentication Required), пренебрегая алгоритмом повторной передачи ответов и, таким образом, работая по отношению к не аутентифицированным запросам, как stateless элементы. Повторная передача ответов с кодом 401 или 407 усиливает действие атаки злоумышленника, когда тот использует поддельное значение поля заголовка (такого как **Via**), чтобы направить трафик третьей стороне.

В заключение подчеркнем, что взаимная аутентификация прокси-серверов с помощью таких механизмов, как TLS, значительно снижает потенциальные возможности злонамеренного пользователя передавать поддельные запросы или ответы, которые могут привести к отказу из-за перегрузки.

Глава 9. Алгоритмы установления соединения

9.1. Установление соединения с участием прокси-сервера

В представленном на рис. 9.1 сценарии пользователь Anton вызывает пользователя Vladimir с использованием двух прокси-серверов – Proxy 1 и Proxy 2. Первоначальный запрос INVITE (F1) содержит в заголовке **Route** предустановленный маршрут с адресом Proxy 1. Proxy 1 сконфигурирован как исходящий прокси-сервер для пользователя Anton. Запрос не содержит информации аутентификации, поэтому Proxy 1 передает ответ с кодом 407 (Proxy Authorization), содержащий запрос подтверждения подлинности.

Затем передается новый запрос INVITE (F4), содержащий надлежащий отклик аутентификации, и происходит установление соединения. Соединение разрушается, когда Vladimir отсоединяется и передает сообщение BYE.

Прокси-сервер Proxy 1 помещает в сообщение INVITE заголовок **Record-Route** для гарантии того, что он будет принимать участие в последующем обмене сообщениями. Proxy 2 также вносит свое имя в заголовок **Record-Route**. Сообщения ACK (F15) и BYE (F18) содержат заголовок **Route**.

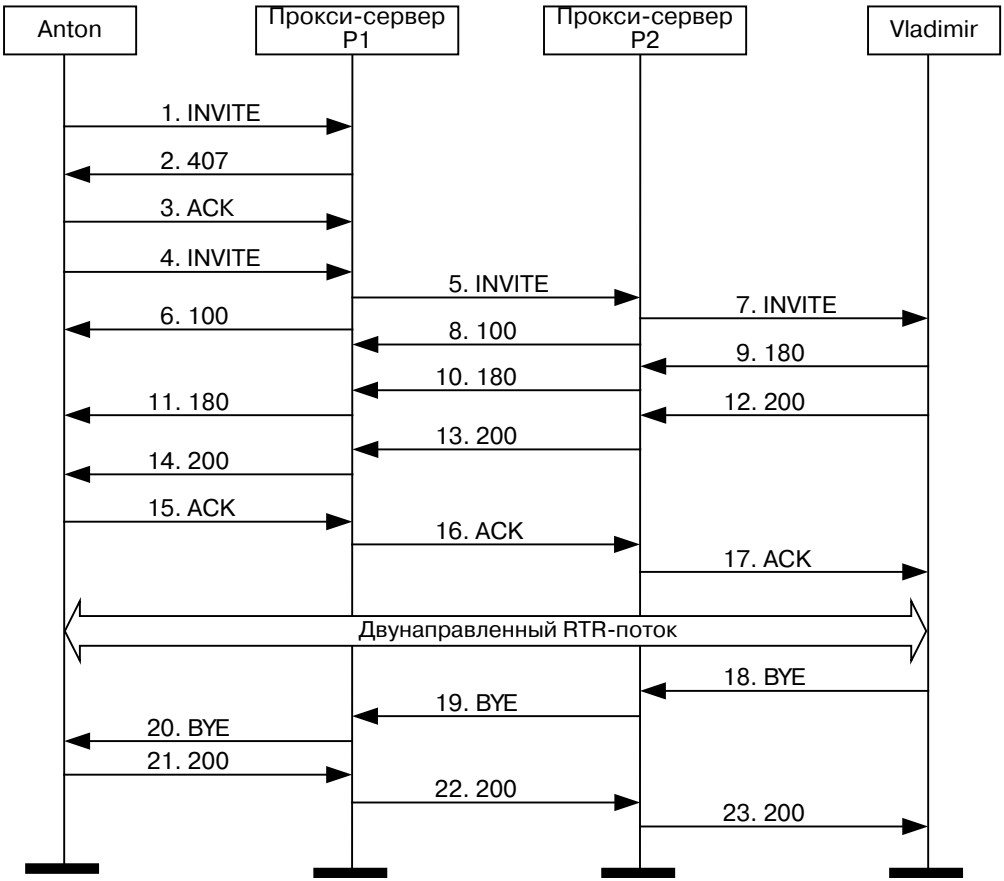


Рис. 9.1. Установление соединения с участием прокси-серверов

Рассмотрим содержание сообщений на рис. 9.1.

1. INVITE Anton → Proxy 1

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74b43
Max-Forwards: 70
Route: <sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@serv1.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=anton 2890844526 2890844526 IN IP4 serv1.niits.ru
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Далее прокси-сервер Proxy 1 запрашивает аутентификацию.

2. 407 (Proxy Authorization Required) Proxy 1 → Anton

```
SIP/2.0 407 Proxy Authorization Required
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74b43
;received=192.0.2.101
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=3flal12sf
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 INVITE
Proxy-Authenticate: Digest realm="niits.ru", qop="auth",
nonce="f84f1cec41e6cbe5aea9c8e88d359",
opaque="", stale=FALSE, algorithm=MD5
Content-Length: 0
```

3. ACK Anton → Proxy 1

```
ACK sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74b43
Max-Forwards: 70
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=3flal12sf
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 ACK
Content-Length: 0
```

Здесь Anton предпринял новую попытку передать запрос INVITE, содержащий отклик аутентификации.

4. INVITE Anton → Proxy 1

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:anton@serv1.niits.ru;transport=tcp>
Proxy-Authorization: Digest username=»anton»,
realm="niits.ru",
nonce="wf84f1ceczx41ae6cbe5aea9c8e88d359", opaque="",
uri="sip:vladimir@protei.ru",
response="42ce3cef44b22f50c6a6071bc8"
Content-Type: application/sdp
Content-Length: 151

v=0
o=anton 2890844526 2890844526 IN IP4 serv1.niits.ru
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 принимает отклик аутентификации и пересылает INVITE прокси-серверу Proxy 2. Клиент пользователя Anton готовится принимать из сети пользовательскую информацию на порт 49172.

5. INVITE Proxy 1 → Proxy 2

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:anton@serv1.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=anton 2890844526 2890844526 IN IP4 serv1.niits.ru
s=-
c=IN IP4 192.0.2.101
```

```
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

6. 100 (Trying) Proxy 1 → Anton

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Content-Length: 0
```

7. INVITE Proxy 2 → Vladimir

```
INVITE sip:vladimir@serv3.protei.ru SIP/2.0
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 68
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:anton@serv1.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=anton 2890844526 2890844526 IN IP4 serv1.niits.ru
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

8. 100 (Trying) Proxy 2 → Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Content-Length: 0
```

9. 180 (Ringing) Vladimir → Proxy 2

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
Contact: <sip:vladimir@serv3.protei.ru;transport=tcp>
CSeq: 2 INVITE
Content-Length: 0
```

10. 180 (Ringing) Proxy 2 → Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
Contact: <sip:vladimir@serv3.protei.ru;transport=tcp>
CSeq: 2 INVITE
Content-Length: 0
```

11. 180 (Ringing) Proxy 1 → Anton

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
Contact: <sip:vladimir@serv3.protei.ru;transport=tcp>
CSeq: 2 INVITE
Content-Length: 0
```

12. 200 (OK) Vladimir → Proxy 2

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
```

```
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv3.protei.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=vladimir 2890844527 2890844527 IN IP4 serv3.protei.ru
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

13. 200 (OK) Proxy 2 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv3.protei.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=vladimir 2890844527 2890844527 IN IP4 serv3.protei.ru
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

14. 200 (OK) Proxy 1 → Anton

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
```

```
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv3.protei.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 147

v=0
o=vladimir 2890844527 2890844527 IN IP4 serv3.protei.ru
s=-
c=IN IP4 192.0.2.201
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

15. ACK Anton → Proxy 1

```
ACK sip:vladimir@serv3.protei.ru SIP/2.0
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74b76
Max-Forwards: 70
Route: <sip:ss1.niits.ru;lr>,
<sip:ss2.protei.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 ACK
Content-Length: 0
```

16. ACK Proxy 1 → Proxy 2

```
ACK sip:vladimir@serv3.protei.ru SIP/2.0
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74b76
;received=192.0.2.101
Max-Forwards: 69
Route: <sip:ss2.protei.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 ACK
Content-Length: 0
```

17. ACK Proxy 2 → Vladimir

```
ACK sip:vladimir@serv3.protei.ru SIP/2.0
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74b76
;received=192.0.2.101
Max-Forwards: 68
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 3848276298220188511@niits.ru
```

```
CSeq: 2 ACK
Content-Length: 0
```

Между терминалами пользователей Anton и Vladimir созданы RTP-потоки. Спустя определенное время Vladimir кладет трубку. Заметим, что значение CSeq не равно 3. Терминалы пользователей Anton и Vladimir поддерживают раздельный порядок счета CSeq.

18. BYE Vladimir → Proxy 2

```
BYE sip:anton@serv1.niits.ru SIP/2.0
Via: SIP/2.0/TCP serv3.protei.ru:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
Route: <sip:ss2.protei.ru;lr>,
<sip:ss1.niits.ru;lr>
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

19. BYE Proxy 2 → Proxy 1

```
BYE sip:anton@serv1.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
Via: SIP/2.0/TCP serv3.protei.ru:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
Max-Forwards: 69
Route: <sip:ss1.niits.ru;lr>
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

20. BYE Proxy 1 → Anton

```
BYE sip:anton@serv1.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP serv3.protei.ru:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
Max-Forwards: 68
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

21. 200 (OK) Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP serv3.protei.ru:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

22. 200 (OK) Proxy 1 → Proxy 2

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss2.protei.ru:5060;branch=z9hG4bK721e4.1
;received=192.0.2.222
Via: SIP/2.0/TCP serv3.protei.ru:5060;branch=z9hG4bKnashds7
;received=192.0.2.101
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

23. 200 (OK) Proxy 2 → Vladimir

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP serv3.protei.ru:5060;branch=z9hG4bKnashds7
;received=192.0.2.201
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 3848276298220188511@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

9.2. Установление соединения с участием сервера перенаправления

В этом сценарии, приведенном на рис. 9.2, пользователь Anton производит вызов пользователя Vladimir с помощью сервера перенаправления. Первоначально сообщение INVITE передается серверу перенаправления. Он дает ответ с кодом 302 (Moved Temporarily), содержащий заголовок **Contact** с текущим SIP-адресом пользователя Vladimir. Затем Anton создает новый запрос INVITE и передает его пользователю Vladimir через прокси-сервер; далее соединение устанавливается по стандартному сценарию. В данном примере INVITE не содержит SDP-описания сеанса связи, поэтому оно присутствует в сообщении ACK. Соединение разрушается, когда Vladimir передает сообщение BYE.

Рассмотрим содержание сообщений для этого сценария.

1. INVITE Anton → Сервер перенаправления

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP serv1.niits.ru:5060;branch=z9hG4bKbf9f44
Max-Forwards: 70
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@serv1.niits.ru>
Content-Length: 0
```

2. 302 (Moved Temporarily) Сервер перенаправления → Anton

```
SIP/2.0 302 Moved Temporarily
Via: SIP/2.0/UDP serv1.niits.ru:5060;branch=z9hG4bKbf9f44
;received=192.0.2.101
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=53fHlqlQ2
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 INVITE
Contact: <sip:vladimir@loniis.ru;transport=tcp>
Content-Length: 0
```

3. ACK Anton → Сервер перенаправления

```
ACK sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP serv1.niits.ru:5060;branch=z9hG4bKbf9f44
Max-Forwards: 70
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=53fHlqlQ2
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 ACK
Content-Length: 0
```

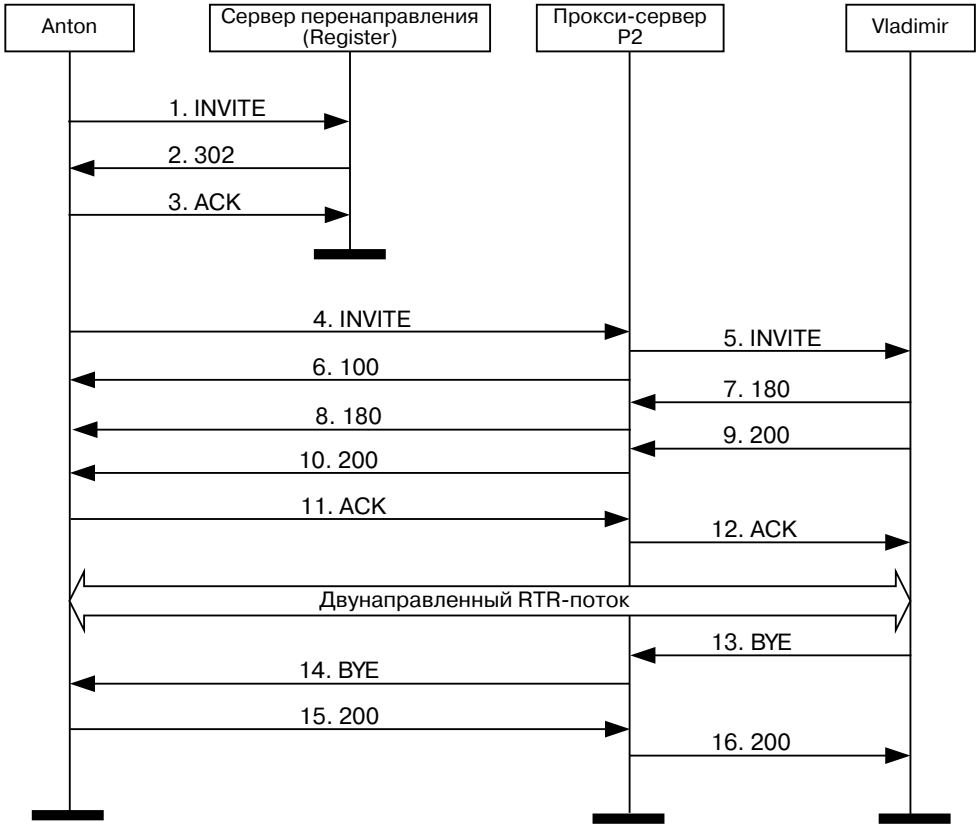


Рис. 9.2. Установление соединения с участием сервера перенаправления

4. INVITE Anton → Proxy 2

```
INVITE sip:vladimir@loniis.ru SIP/2.0
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 INVITE
Contact: <sip:anton@serv1.niits.ru;transport=tcp>
Content-Length: 0
```

5. INVITE Proxy 2 → Vladimir

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/TCP ss1.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.niits.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 3848276298220188511@niits.ru
CSeq: 2 INVITE
Contact: <sip:anton@serv1.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151
```

6. 100 (Trying) Proxy 2 → Anton

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 INVITE
Content-Length: 0
```

7. 180 (Ringing) Vladimir → Proxy 2

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss3.loniis.ru:5060;branch=z9hG4bK721e.1
;received=192.0.2.233
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.loniis.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv5.loniis.ru;transport=tcp>
Content-Length: 0
```

8. 180 (Ringing) Proxy 2 → Anton

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.loniis.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv5.loniis.ru;transport=tcp>
Content-Length: 0
```

9. 200 (OK) Vladimir → Proxy 2

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss3.loniis.ru:5060;branch=z9hG4bK721e.1
;received=192.0.2.233
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.loniis.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv5.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 148

v=0
o=vladimir 2890844527 2890844527 IN IP4 serv5.loniis.ru
s=-
c=IN IP4 192.0.2.100
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

10. 200 (OK) Proxy2 → Anton

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.loniis.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 INVITE
Contact: <sip:vladimir@serv5.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 148

v=0
o=vladimir 2890844527 2890844527 IN IP4 serv5.loniis.ru
s=-

c=IN IP4 192.0.2.100
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Подтверждение ACK содержит SDP-описание сеанса агента пользователя Anton, так как оно отсутствует в запросе INVITE.

11. ACK Anton → Proxy 2

```
ACK sip:vladimir@serv5.loniis.ru SIP/2.0
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bq9
Max-Forwards: 70
Route: <sip:ss3.loniis.ru;lr>
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 ACK
Content-Type: application/sdp
Content-Length: 151

v=0
o=anton 2890844526 2890844526 IN IP4 serv1.niits.ru
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

12. ACK Proxy 2 → Vladimir

```
ACK sip:vladimir@serv5.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss3.loniis.ru:5060;branch=z9hG4bK721e.1
Via: SIP/2.0/TCP serv1.niits.ru:5060;branch=z9hG4bK74bq9
;received=192.0.2.101
Max-Forwards: 69
From: Anton <sip:anton@niits.ru>;tag=9fxced76sl
To: Vladimir <sip:vladimir@protei.ru>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 2 ACK
Content-Type: application/sdp
Content-Length: 151
v=0
o=anton 2890844526 2890844526 IN IP4 serv1.niits.ru
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

После того как Vladimir получает подтверждение ACK, между пользователями Anton и Vladimir создаются RTP-потоки. Спустя какое-то время Vladimir кладет трубку.

13. BYE Vladimir → Proxy 2

```
BYE sip:anton@serv1.niits.ru SIP/2.0
Via: SIP/2.0/TCP serv5.loniis.ru:5060;branch=z9hG4bKfgaw2
Max-Forwards: 70
Route: <sip:ss3.loniis.ru;lr>
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
```

```
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

14. BYE Proxy 2 → Anton

```
BYE sip:anton@serv1.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss3.loniis.ru:5060;branch=z9hG4bK721e.1
;received=192.0.2.100
Via: SIP/2.0/TCP serv5.loniis.ru:5060;branch=z9hG4bKfgaw2
Max-Forwards: 69
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

15. 200 (OK) Anton → Proxy 2

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss3.loniis.ru:5060;branch=z9hG4bK721e.1
;received=192.0.2.233
Via: SIP/2.0/TCP serv5.loniis.ru:5060;branch=z9hG4bKfgaw2
;received=192.0.2.100
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

16. 200 (OK) Proxy 2 → Vladimir

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP serv5.loniis.ru:5060;branch=z9hG4bKfgaw2
;received=192.0.2.100
From: Vladimir <sip:vladimir@protei.ru>;tag=314159
To: Anton <sip:anton@niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@niits.ru
CSeq: 1 BYE
Content-Length: 0
```

Глава 10. Транспортный уровень протокола SIP

10.1. Назначение транспортного уровня

Транспортный уровень протокола SIP отвечает за перенос запросов и ответов через сеть с использованием ее транспортных протоколов. Кроме того, в случае применения ориентированных на соединение транспортных протоколов транспортный уровень SIP производит выбор соединения, используемого для запроса или ответа.

Транспортный уровень SIP отвечает за управление соединениями таких транспортных протоколов как TCP и SCTP. Транспортный уровень SIP имеет клиентскую и серверную сторону, поэтому при создании соединения оно контролируется транспортными функциями и клиента, и сервера. Соединения идентифицируются указателем, состоящим из адреса, порта и транспортного протокола на удаленном конце соединения. Когда соединение создается транспортным уровнем SIP-отправителя, этот указатель принимает значение IP-адреса, номера порта и транспортного протокола места назначения. Когда соединение создается транспортным уровнем SIP-получателя, указатель принимает значение IP-адреса, номера порта и транспортного протокола источника.

Заметим, что номер порта может являться временным, и не существует возможности узнать, является ли он временным или выбран с помощью DNS-процедур, описанных в RFC 3263 «Locating SIP servers»[56]. Из-за этого соединения, принятые

транспортным уровнем SIP, могут не использоваться повторно. В результате два прокси-сервера, взаимодействующих с использованием ориентированных на соединение транспортных протоколов, зачастую будут одновременно использовать два соединения – для транзакций, инициированных в том и в другом направлении.

Соединение должно сохраняться в течение некоторого интервала времени после того, как последнее сообщение было передано или получено через это соединение. Этот интервал определяется конкретной реализацией и должен быть, как минимум, равным наибольшему времени, требующемуся элементу для того, чтобы перевести транзакцию в завершенное состояние Terminated. Это нужно для того, чтобы сделать возможным выполнение транзакционных функций в том же соединении, в котором транзакции были инициированы. Обычно этот интервал, по меньшей мере, равен 64*T1. Однако значение интервала может быть больше, например, в SIP элементе, содержащем TU, который использует большее значение таймера C (см. п. 5.2.4.11.).

Все узлы SIP должны уметь работать с протоколами UDP и TCP. Могут применяться также и другие протоколы.

10.2. Работа клиента при передаче запросов

Клиентская сторона транспортного уровня SIP отвечает за передачу запросов и прием ответов. Пользователь транспортного уровня SIP, которым может являться либо транзакция, либо ядро SIP-приложения, передает клиентской стороне транспортного уровня SIP-запрос, IP-адрес, номер порта, название транспортного протокола, и, возможно, TTL при многоадресной рассылке.

Если длина запроса находится в пределах 200 байтов (максимальная величина передаваемого блока данных MTU), или если он больше 1300 байтов, а MTU не известна, запрос должен передаваться с использованием транспортного протокола, предусматривающего контроль перегрузки, такого, как TCP. Если такие действия вызывают изменение транспортного протокола (протокол не совпадает со значением, указанным в верхнем заголовке **Via**), значение в верхнем заголовке **Via** должно быть изменено. Это предотвратит возможную фрагментацию сообщений, передающихся по протоколу UDP, и обеспечит контроль перегрузки при передаче сообщений большей длины. Однако реализации должны быть способны

работать с сообщениями вплоть до максимального размера пакета дейтаграммы. Для UDP этот размер составляет 65,535 байтов, включая заголовки IP и UDP.

Существование 200-байтового «буфера» между размером сообщения и MTU обусловлено тем, что ответ в протоколе SIP может быть больше запроса. Например, это происходит при добавлении значений заголовка **Record-Route** в ответ на запрос INVITE. Размер 1300 байтов выбирается, когда path MTU не известна, исходя из допущения, что Ethernet MTU составляет 1500 байтов.

Иногда элемент сети SIP вынужден передать запрос по протоколу TCP из-за большого размера сообщения (однако при этом существует возможность передать сообщение и по UDP). Тогда, если в результате попытки установить TCP-соединение произошел сбой или выяснилось, что протокол ICMP не поддерживается, элемент сети SIP должен совершить повторную попытку передать запрос, но уже по UDP. Это нужно для обеспечения совместимости с реализациями, выполненными по более ранней версии рекомендаций и не поддерживающими протокол TCP. Ожидается, что такое поведение будет запрещено в следующих версиях рекомендаций.

Клиент, использующий многоадресную рассылку, должен добавить параметр «maddr» в значение заголовка **Via**, содержащее URI многоадресной рассылки места назначения, и при использовании IPv4 должен добавить параметр «ttl» со значением, равным 1.

Перед тем как передать запрос, клиентская сторона транспортного уровня SIP должна поместить в заголовок **Via** значение, состоящее из IP-адреса/имени хоста и номера порта. Если номер порта отсутствует, значение по умолчанию для него устанавливается в зависимости от используемого транспортного протокола. Оно равно 5060 для UDP, TCP и SCTP, и 5061 для TLS.

При применении надежных транспортных протоколов ответ передается через соединение, по которому был получен запрос. Поэтому клиентская сторона транспортного уровня SIP должна быть готова получить ответ по тому же соединению, которое использовалось для передачи запроса. В случае возникновения ошибки сервер может попытаться создать для передачи ответа новое соединение. Чтобы предусмотреть этот случай, транспортный уровень SIP также должен быть готов принять входящее соединение на IP-адрес с которого был передан запрос, и на номер порта, указанный в значении заголовка **Via**.

Транспортный уровень SIP должен быть также готов получить входящее соединение на любой адрес и порт, которые будут выбраны сервером с использование процедур, описанных в главе 5, [56].

При использовании ненадежных транспортных протоколов клиентская сторона транспортного уровня должна быть готова принять ответы на IP-адрес источника, от которого передан запрос (поскольку ответы передаются обратно на адрес ресурса) и номер порта, указанный в значении заголовка **Via**. Более того, так же, как и для надежных транспортных протоколов, в определенных случаях ответ может быть передан по другому адресу. Клиент должен быть готов принять ответы на любой адрес и порт, которые будут определены сервером при использовании процедур, описанных в главе 5, [56].

При многоадресной рассылке клиентская сторона транспортного уровня SIP должна быть готова получить ответы на ту же multicast-группу и порт, на которые передается запрос (то есть для того, чтобы получить ответ, требуется быть членом той multicast-группы, которой был передан запрос).

Если запрос предназначен для передачи на IP-адрес, порт и транспортный протокол, для которых уже существует соединение, он передается по этому соединению, однако возможность создания другого соединения не исключается.

Если запрос предусматривает многоадресную рассылку, он передается с использованием группового адреса, номера порта и TTL, предоставленных пользователем транспортного уровня SIP. Если используется ненадежный транспортный протокол, запрос передается на IP-адрес и порт, указанные пользователем транспортного уровня.

10.3. Работа клиента при получении ответов

Когда получен ответ, клиентская сторона транспортного уровня SIP проверяет верхнее значение заголовка **Via**. Если в этом значении заголовка **Via** адрес и порт не соответствуют значению, на которое была конфигурирована клиентская сторона транспортного уровня, по умолчанию ответ отклоняется.

Если существуют действующие клиентские транзакции, то клиентская сторона транспортного уровня SIP использует процедуры, описанные в § 4.4.1, чтобы сопос-

тавить пришедший ответ с существующей транзакцией. Если обнаруживается соответствие, ответ должен быть передан этой транзакции. В противном случае ответ должен быть передан для дальнейшей обработки ядру (вне зависимости от того, будет ли это stateless прокси-сервер, stateful прокси-сервер или UA). Обработка ответов, не соответствующих ни одной из транзакций, зависит от типа ядра (например, ядро прокси-сервера перешлет их, в то время как ядро UA их отклонит).

10.4. Работа сервера при получении запросов

Сервер должен быть готов получать запросы на любую комбинацию IP-адреса, порта и транспортного протокола, на которую клиент может передать запрос после DNS-поиска по обнаруженному адресу. Адрес может быть определен посредством заголовка **Contact** запроса REGISTER ответа класса 3xx или заголовка **Record-Route** сообщения. Сервер должен всегда ожидать запросы, приходящие на установленные по умолчанию SIP-порты (5060 для TCP и UDP, 5061 для TLS по TCP) по всем общедоступным интерфейсам.

Типичным исключением являются частные сети, или случай, когда на одном узле функционирует несколько серверных приложений. Для всех портов и интерфейсов, на которые сервер ожидает поступления UDP-пакетов, он должен ожидать также поступления TCP-пакетов. Это происходит потому, что если сообщение слишком длинное, оно может требовать передачи по протоколу TCP, а не UDP. Обратное не верно. Серверу нет смысла ожидать поступления UDP-пакетов на определенный адрес и порт, если он ожидает поступления на этот адрес и порт TCP-пакетов. Однако могут возникнуть определенные причины, когда при этом сервер будет ожидать UDP-пакеты.

Когда серверная сторона транспортного уровня SIP получает запрос, пришедший по любому транспортному протоколу, она должна проанализировать адрес и порт в верхнем значении заголовка **Via**. Если часть URI, отражающая имя узла, содержит имя домена или содержит IP-адрес, который отличается от адреса источника пакетов, сервер должен добавить в значение заголовка **Via** параметр «received». Этот параметр должен содержать адрес источника, от которого был получен пакет. Параметр будет использован серверной стороной транспортного уровня SIP при передаче ответа на IP-адрес источника, от которого поступил запрос.

К примеру, рассмотрим запрос, полученный серверной стороной транспортного уровня. Интересующая нас часть выглядит так:

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP serv3.protei.ru:5060
```

Запрос получен от IP-адреса источника 192.0.2.4. Перед тем как передать запрос дальше, транспортный уровень SIP добавляет в значение заголовка **Via** параметр «received» так, чтобы запрос выглядел следующим образом.

```
INVITE sip:vladimir@protei.ru SIP/2.0
Via: SIP/2.0/UDP serv3.protei.ru:5060;received=192.0.2.4
```

Далее, серверная сторона транспортного уровня SIP пытается сопоставить запрос с серверной транзакцией. Для этого используются процедуры, описанные в § 4.4.8. Если соответствующая серверная транзакция найдена, запрос передается этой транзакции для обработки. Если таковой не найдено, запрос передается ядру, которое может принять решение о формировании для данного запроса новой серверной транзакции. Заметим, что когда ядро UAS передает ответ класса 2xx на INVITE, серверная транзакция разрушается. Это означает, что к тому моменту, когда приходит подтверждение ACK, соответствующая серверная транзакция уже не будет существовать. Поэтому запрос ACK направляется ядру UAS, где он и обрабатывается.

10.5. Работа сервера при передаче ответов

Серверная сторона транспортного уровня использует верхнее значение заголовка **Via** для того, чтобы определить, куда отправить ответ. Транспортный уровень SIP производит следующие процедуры обработки.

- Если в значении заголовка **Via** указан надежный транспортный протокол, такой как TCP, SCTP или TLS (по TCP или SCTP), ответ должен быть передан по существующему соединению к источнику оригинального запроса, который создал транзакцию. Для этого серверная сторона транспортного уровня SIP должна удерживать связь между серверными транзакциями и транспортными соединениями. Если соединение разрушено, сервер должен создать соединение в соответствии с IP-адресом в параметре «received» (если он присутствует), используя порт, указанный в значении, или порт, заданный по умолчанию для данного транспортного протокола

(в случае, если значение порта в значении заголовка **Via** не определено). Если попытка создать соединение терпит неудачу, сервер должен найти адрес и номер порта с помощью выполнения процедур, специфицированных в [56], чтобы создать нужное соединение и отправить по нему ответ.

- В случае, когда значение заголовка **Via** содержит параметр «maddr», ответ должен быть передан на адрес, указанный в этом параметре, с использованием порта, обозначенного в значении заголовка или порта 5060, в случае его отсутствия. Если адрес является адресом многоадресной рассылки, ответ передается с использованием значения TTL, указанного в параметре «ttl», или значения TTL, равного 1, если этот параметр отсутствует.
- В случае, если верхнее значение **Via** имеет параметр «received» (для ненадежных транспортных unicast-протоколов), ответ должен быть передан на адрес, указанный в этом параметре с использованием порта, определенного в значении заголовка, или порта 5060, если заголовок отсутствует. Если это не удается, например, приводит к получению ответа протокола ICMP – «port unreachable» (порт недоступен), для того, чтобы определить, куда передать ответ, должны быть применены процедуры, описанные в главе 5, [56].
- В противном случае ответ должен быть передан на адрес, указанный в значении заголовка, с использованием процедур, описанных в разделе 5, [56].

10.6. Длина тела сообщения

При использовании транспортных протоколов, не ориентированных на соединение (таких как UDP), если сообщение содержит заголовок **Content-Length**, предполагается, что тело сообщения имеет большую длину. Если сообщение не имеет заголовка **Content-Length**, предполагается, что тело сообщения заканчивается там же, где заканчивается транспортный пакет. Если пакет транспортного протокола содержит дополнительные байты, помещенные за телом сообщения, они должны быть удалены. В случае, когда транспортный пакет заканчивается до окончания тела сообщения, это рассматривается как ошибка. Если сообщение является ответом, оно должно быть отброшено. Когда же сообщение является запросом, элемент создает ответ с кодом 400 (Bad Request).

В случае потоко-ориентированных протоколов, таких как TCP, заголовок **Content-Length**, указывающий размер тела, используется всегда.

10.7. Обработка ошибок

Обработка ошибок не зависит от того, является ли сообщение запросом или ответом. Если пользователь транспортного уровня SIP, например, уровень транзакций или ядро, запрашивает, чтобы сообщение было передано по ненадежному транспортному протоколу, и в результате происходит ICMP-ошибка, поведение транспортного уровня зависит от типа ICMP-ошибки. Ошибки типа «хост, сеть, порт или протокол недоступен» (host, network, port, protocol unreachable) или ошибки, связанные с параметрами, обязывают транспортный уровень SIP информировать своего пользователя о неудаче. Ошибки протокола ICMP «Source quench» и «TTL exceeded» должны игнорироваться.

Если пользователь транспортного уровня SIP запрашивает, чтобы сообщение было передано по надежному транспортному протоколу, и в результате происходит ошибка соединения, транспортный уровень SIP должен информировать пользователя транспортного уровня об ошибке.

Глава 11. Протокол SIP-T для телефонии

11.1. Назначение и особенности протокола SIP-T

В предыдущих главах был подробно рассмотрен ключевой протокол IP-телефонии – протокол SIP. Но в сегодняшних условиях конвергенции сетей и услуг связи SIP-сети не могут существовать изолированно от традиционных телефонных сетей, поэтому необходим простой в управлении и легкий для понимания интерфейс взаимодействия между ними. Для решения этой задачи было разработано описание протокола SIP для взаимодействия с телефонными сетями, получившее название SIP-T.

Спецификации SIP-T описывают способы передачи общеканальной сигнализации ОКС7 по сети SIP, для чего используются два механизма переноса сигнализации, известных как *инкапсуляция* и *трансляция*. В шлюзах SIP-ISUP сообщение ISUP ОКС7 инкапсулируется в сообщение протокола SIP, при этом сохраняется только информация, необходимая для обслуживания. Однако некоторые посредники, например, прокси-серверы, принимающие решения о продвижении запроса SIP дальше по сети, не могут правильно распознать сигнальные единицы ISUP. Наряду с инкапсуляцией, важная информация транслируется из сообщения ISUP в заголовки сообщений SIP в порядке, определяемом тем, как дальше будет маршрутизироваться SIP-запрос. Процедуры, необходимые при взаимодействии сетей SIP и ТфОП, приведены в табл. 11.1.

Таблица 11.1.

Требования к интерфейсу при взаимодействии ТфОП-SIP	Функции протокола SIP-T
Прозрачность сети SIP для сигнализации ISUP	Инкапсуляция сообщений ISUP в тело запросов SIP
Маршрутизация запросов SIP по информации, содержащейся в сообщениях ISUP	Трансляция параметров сообщений ISUP в заголовки запросов SIP
Передача сигнальной информации ISUP во время мультимедийного сеанса	Использование запроса INFO

Заметим, что в отечественной ТфОП используется ряд различных протоколов сигнализации [27, 28, 29, 30], но в этом справочнике рассматривается только взаимодействие SIP и ISUP [28].

11.2. Сценарии организации взаимодействия

В этом параграфе рассмотрен ряд сценариев взаимодействия в реальных условиях отечественных сетей связи.

11.2.1. Применение SIP-T при транзите (ТфОП–IP–ТфОП)

Сценарий, когда сеть SIP связывает двух абонентов ТфОП, т.е. является транзитной сетью, называют также *SIP bridging*. Для сообщения ISUP сеть SIP является прозрачной, т.е. сообщение проходит через сеть SIP, не изменяясь. Это достигается путем инкапсуляции сообщения ISUP в тело SIP запроса. На рис. 11.1 показана схема установления соединения для сценария ТфОП–SIP–ТфОП. Когда вызов, предназначенный для сети SIP, идет от абонента ТфОП, сообщение IAM ISUP будет получено шлюзом сигнализации в MGC (Media Gateway Controller), который является точкой взаимодействия ТфОП и сети SIP. Контроллер медиашлюза MGC стандартными средствами протокола SIP передает запрос в SIP-сеть. Протокол SIP выполняет стандартную маршрутизацию внутри сети, чтобы определить для вызова нужную точку выхода (в нашем случае – другой MGC). Найдя эту точку SIP начинает диалог установления медиа-соединения между начальной и конечной точками SIP-сети. Оконечный MGC, который является точкой выхода из сети SIP в ТфОП, передает в сеть ТфОП сообщение IAM ISUP, пришедшее в запросе SIP, не проверяя его, т.е. просто транслирует сообщение в сеть ТфОП без анализа содержимого. Пример простого соединения для такого сценария приведен на рис. 11.2.

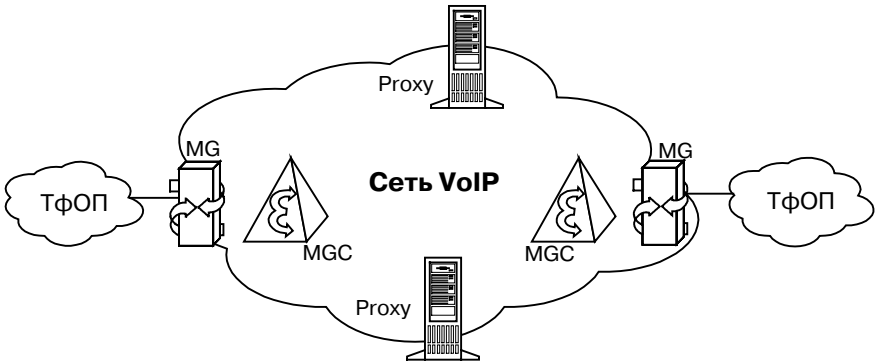


Рис. 11.1. Транзитная связь ТфОП–SIP–ТфОП

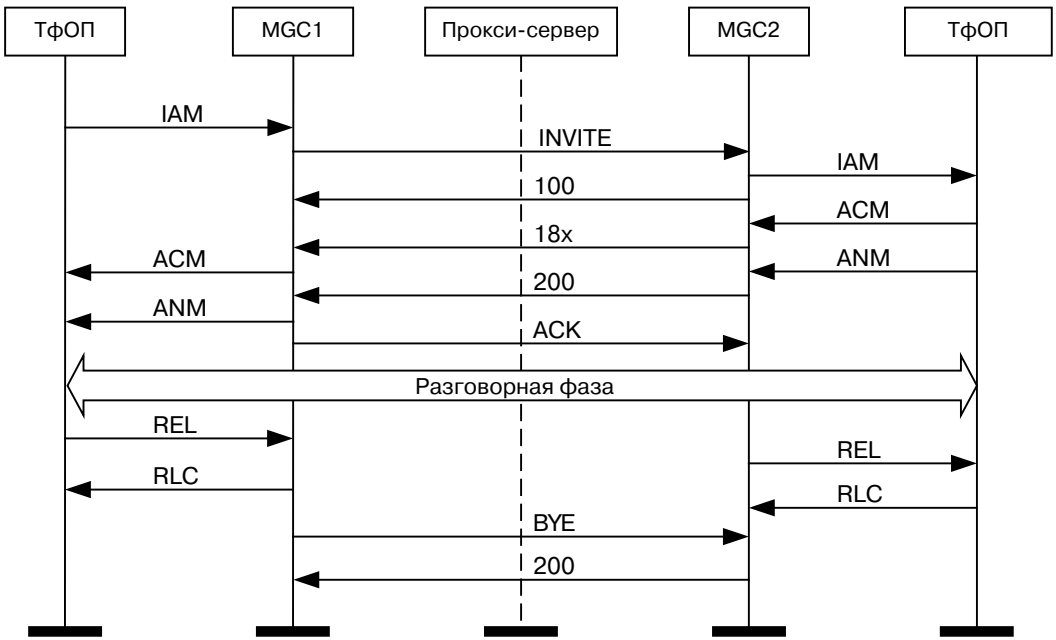


Рис. 11.2. Сценарий ТфОП–SIP–ТфОП

11.2.2. Процедуры организации связи из ТфОП в IP-сеть

В этом сценарии начальной точкой является телефонный аппарат ТфОП, а конечной точкой – терминальное оборудование пользователя сети SIP.

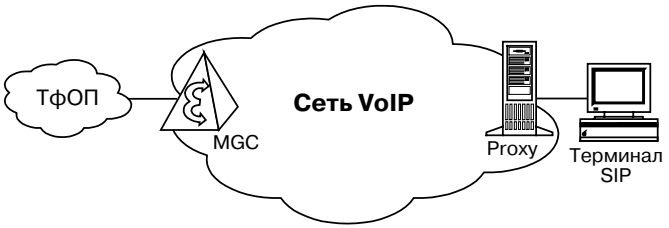


Рис. 11.3. Связь ТфОП–SIP

На рис. 11.3 приведена схема организации взаимодействия для этого сценария. Здесь требуется только один MGC, который при получении сообщения IAM ISUP из ТфОП транслирует его в SIP-запрос, который передается в SIP-сеть. Этот запрос обрабатывается в сети SIP стандартным образом, и в результате устанавливается медиа-соединение через MGC и прокси-сервер к SIP-телефону. Порядок обмена сообщениями в таком сценарии приведен на рис. 11.4.

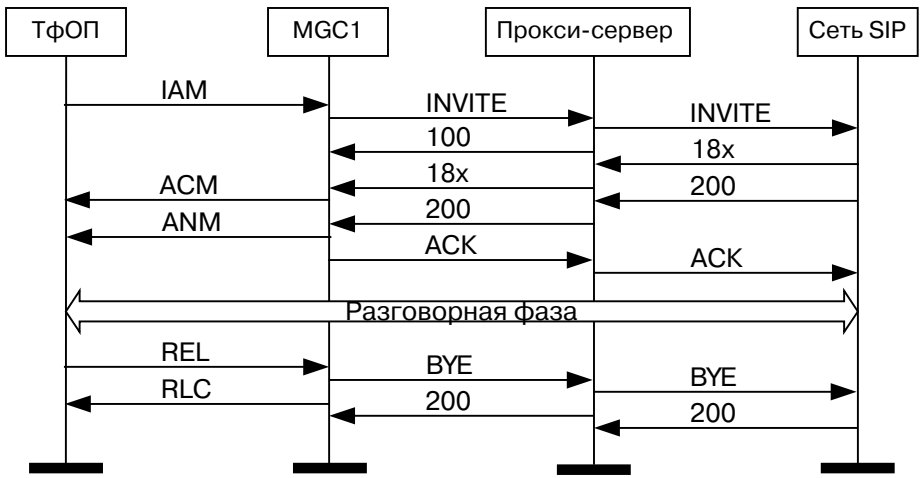


Рис. 11.4. Сценарий ТфОП–SIP

11.2.3. Процедуры организации связи из IP-сети в ТфОП

В этом сценарии начальной точкой является терминал сети SIP, а конечной точкой – телефонный аппарат ТфОП.

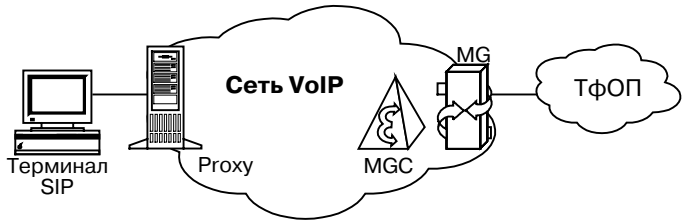


Рис. 11.5. Связь SIP–ТфОП

Схема организации взаимодействия для этого сценария показана на рис. 11.5. В отличие от предыдущих двух случаев, где сообщение IAM ISUP инкапсулировалось в тело запроса SIP, и часть этого сообщения транслировалась в заголовок запроса, в данном сценарии MGC, являющийся точкой выхода из сети SIP, только транслирует заголовок запроса SIP в соответствующие параметры сообщения IAM. Порядок организации взаимодействия в таком сценарии показан на рис. 11.6.

От терминала SIP поступает запрос INVITE, который через прокси-сервер сети SIP попадает на MGC. MGC анализирует полученное сообщение и определяет, что его необходимо передать дальше в сеть ТфОП. После этого он транслирует часть заголовка SIP-сообщения в сообщение IAM ISUP и оно передается в ТфОП.

Пока сигнальная информация анализируется и доставляется до нужного абонента ТфОП, пользователю сети SIP передается сообщение 100 (Trying).

Как только из сети ОКС7 ТфОП приходит сообщение ACM, вызывающему пользователю передается сообщение 180 (Ringing) или 183 (Session Progress).

Далее абонент поднимает трубку, и из ТфОП к MGC приходит сообщение ANM. MGC передает в сеть SIP ответ 200 OK. После этого к MGC приходит окончательное подтверждение ACK.

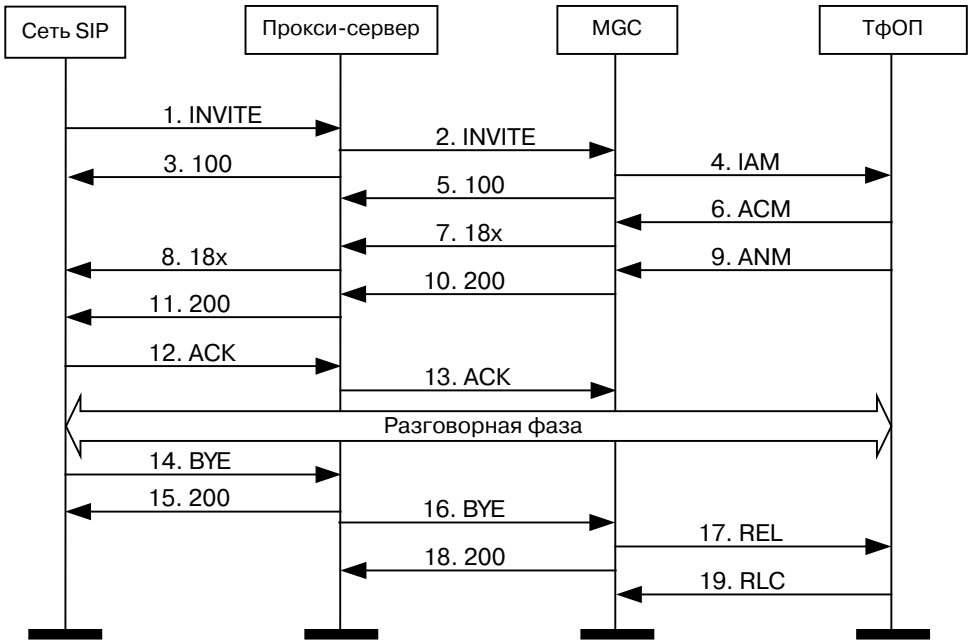


Рис. 11.6. Сценарий ТфОП – SIP

11.3. Компоненты протокола SIP–T

11.3.1. Использование протокола SIP

SIP-T использует методы и процедуры протокола SIP, описанные в предыдущих главах справочника.

11.3.2. Процедуры инкапсуляции сигнальных сообщений

Возможность инкапсуляции сигнализации ТфОП является одним из основных требований к SIP-T. Протокол SIP-T использует разделяемое на необходимое число

частей тело сообщения в кодировке MIME, что позволяет включать в сообщения SIP различную информацию: данные протоколов SDP, ISUP и т.д. Существует много версий ISUP, и поэтому для определения используемой версии введен специальный MIME-тип – ISUP Media Type, позволяющий удобно получать информацию об используемом варианте ISUP. ISUP Media Type содержит информацию, представленную в табл. 11.2.

Таблица 11.2. ISUP Media Type

Media type name:	Application
Media subtype name:	ISUP
Required parameters:	version
Optional parameters:	base
Encoding scheme:	binary
Security considerations:	SIP

Использование параметра «version» позволяет системным администраторам узнать тип ISUP. Это дает возможность каждому Softswitch/MGC корректно обработать сообщение, или передать пользователю сообщение о том, что данный тип ISUP не поддерживается. Спецификация не ограничивает значения, которые могут быть использованы в «version»; это оставлено на усмотрение системных администраторов.

Параметр «version» может быть использован для идентификации реализации ISUP в сети (например, X-NetxProprietaryISUPv3), или для определения известных стандартных версий ISUP, таких как версия ITU-T, ISUP-R или ANSI.

Параметр «base» может включаться опционально в некоторые сообщения, если обязательно требуется, чтобы получатель правильно распознал используемый тип ISUP, т.к. параметр «version» может быть не понят. Таблица 11.3 представляет возможные значения параметра «base», поддерживаемых телом сообщения типа «application/ISUP».

Заголовок **Content-Disposition** может служить для описания процесса обработки вложенного сообщения ISUP, в частности, того, какие действия необходимо предпринять, если приемником не было понято содержимое заголовка **Content-Type**. По умолчанию значение заголовка **Content-Disposition** для ISUP-сообщений – «signal». Это показывает, что данная часть тела сообщения содержит сигнальную информацию, но не содержит описания соединения.

Таблица 11.3. Параметры «base»

Значение параметра «base»	Протокол
Itu-t88	ITU-T Q.761-4 (1988)
Itu-t92+	ITU-T Q.761-4 (1992)
Ansi88	ANSI T1.113-1988
Ansi00	ANSI T1.113-2000
Etsi121	ETS 300 121
Etsi356	ES 300 356
Gr317	BELLCORE GR-317
Ttc87	JT-Q761-4 (1987-1992)
Ttc93+	JT-Q761-4 (1993-)

Полное описание заголовка **Content-Disposition** находится в главе 3 и в RFC 2046 [18]. Для вложенных ISUP-сообщений заголовок **Content-Disposition** может иметь следующие параметры и значения: значение заголовка «signal», параметр *handling*, допустимые значения параметра – *optional* и *required*. Пример типичного заголовка (параметр «base» может отсутствовать):

```
Content-Type: application/ISUP; version=nxv3; base=etsi121
Content-Disposition: signal; handling=optional
```

Ниже приведен пример сообщения INVITE, которое содержит информацию SDP и инкапсулированное сообщение ISUP IAM. К этому примеру необходимо сделать следующее примечание. Части сообщения разделяются специальной строкой, задаваемой параметром boundary (согласно RFC 2046). В примере для разделения используется пустая строка «unique-boundary-1».

```
INVITE sip:78123877658@max.loniis.ru SIP/2.0
Via: SIP/2.0/UDP anton.loniis.ru
From: sip:78124513355@anton.loniis.ru
To: sip:78123877658@max.loniis.ru
Call-ID: MAX123199902171209550099@max.loniis.ru
CSeq: 8348 INVITE
Contact: <sip:anton@loniis.ru>
Content-Length: 436
Content-Type: multipart/mixed; boundary=unique-boundary-1
MIME-Version: 1.0
--unique-boundary-1
Content-Type: application/SDP; charset=ISO-10646

v=0
o=jpeterson 2890844526 2890842807 IN IP4 126.16.64.4
s=SDP seminar
c=IN IP4 MG122.loniis.ru
```

```
t= 2873397496 2873404696
m=audio 9092 RTP/AVP 0 3 4
--unique-boundary-1
Content-Type: application/ISUP; version=nxv3;
base=etsi121
Content-Disposition: signal; handling=optional

01 00 49 00 00 03 02 00 07 04 10 00 33 63 21
43 00 00 03 06 0d 03 80 90 a2 07 03 10 03 63
53 00 10 0a 07 03 10 27 80 88 03 00 00 89 8b
0e 95 1e 1e 1e 06 26 05 0d f5 01 06 10 04 00
--unique-boundary-1
```

11.3.3. Процедуры преобразования сигнальных сообщений

Речь идет о преобразовании сигнальной информации между протоколами ISUP и SIP. По существу, такое преобразование включает в себя два компонента:

- Преобразование сигнализации ISUP в SIP на уровне сообщений. В SIP-T предполагается использование MGC, которые создают сообщения ISUP из поступающих сообщений SIP и наоборот. Для этого необходимо точное определение правил преобразования сообщений: каждое сообщение ISUP должно быть преобразовано в определенное сообщение SIP, например, IAM в INVITE, REL в BYE и т.д.
- Преобразование параметров сообщения ISUP в заголовки сообщения SIP. Запрос SIP, который используется для установления соединения, должен содержать необходимую для маршрутизации прокси-серверами информацию, например, это может быть телефонный номер, набранный вызывающим абонентом.

На практике очень важно стандартизировать процедуры преобразования информации из ISUP в SIP (например, Called Party Number в ISUP IAM должен быть записан в заголовок **To** и поле Request-URI в SIP и т.д.).

Одной из проблем преобразования при транзите трафика через сеть SIP является то, что параметр ISUP, переведенный в заголовок сообщения SIP, может изменяться промежуточными узлами сети. Оконечный MGC (точка выхода из сети SIP) может получить сообщение, в котором параметры заголовка сообщения SIP не соответствуют параметрам вложенного сообщения ISUP. Например, параметр заголовка **To** и поля Request-URI запроса SIP могут отличать от параметра Called Party Number (номер вызываемого абонента) во вложенном сообщении ISUP.

В этом случае приоритет имеют значения заголовков, т.е. при создании нового сообщения параметры будут заполнятся значениями из заголовков запроса SIP, а недостающая информация будет взята из вложенного сообщения ISUP, если оно присутствует.

11.3.4. Поддержка передачи сигнальных сообщений во время сеанса

Базовые сообщения протокола SIP не могут обеспечить передачу сигнальных сообщений во время сеанса связи. Для этой цели необходимо использовать дополнительное сообщение INFO (согласно [11]). На практике этот запрос используется для передачи сигналов, когда телефонный номер из ТфОП передается по частям (overlap dialing). Это сообщение можно использовать для передачи сигналов DTMF. Описание механизмов передачи сигналов DTMF также представлено в [63].

11.4. Согласование содержимого сообщений протокола SIP

Шлюз, который передает запрос, может включить в сообщение тело, состоящее из нескольких частей разных типов: например, описание SDP и сообщение ISUP. Если получатель не поддерживает разделения тела сообщения на несколько частей (формат multipart/mixed) и/или пришедший в сообщении тип ISUP MIME (application/ISUP), то он отклоняет запрос и передает сообщение 415 (Unsupported Media Type), в котором содержатся поддерживаемые форматы (по умолчанию «application/SDP»). Шлюз, который передал сообщение, должен впоследствии передать его еще раз, предварительно удалив из него часть тела сообщения с сообщением ISUP (т.е. оставить только SDP), и такой запрос будет принят.

Это приводит к необходимости иметь механизм, при котором устройство, передающее сообщение, отмечает, какие части тела сообщения в запросе обязательные, а какие – опциональные. На принимающей стороне, если устройство не поддерживает определенный формат, оно проверяет, к какому классу относится данная часть тела сообщения: если это необязательная часть, то она отклоняется, а сообщение анализируется дальше, если часть является обязательной, то отклоняется все сообщение. Так, например, в терминале SIP потеря части тела, в которой содержится сообщение ISUP, не приведет к утрате SIP-сообщения. Рассмотрим примеры реализации такого механизма.

На рис. 11.7 представлен сценарий, в котором поддержка ISUP необязательна, а UA 2 принимает INVITE независимо от того, может он обработать ISUP или нет.

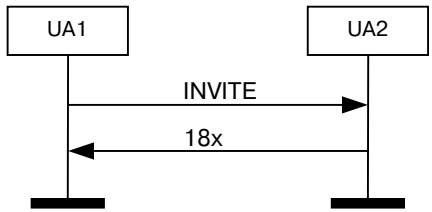


Рис. 11.7. Сценарий без поддержки ISUP

```
UA1          UA2
INVITE →
Content-type:multipart/mixed;
Content-type: application/sdp;
Content-disposition: session; handling=required;
Content-type: application/isup;
Content-disposition: signal; handling=optional;)
← 18x
```

На рис. 11.8 представлен сценарий, в котором поддержка ISUP предпочтительна. Здесь UA 2 не поддерживает ISUP и отклоняет запрос, передавая сообщение об ошибке 415 (Unsupported Media Type), а UA 1 удаляет из запроса часть с ISUP и опять передает сообщение, содержащее теперь только SDP, и оно принимается UA 2.

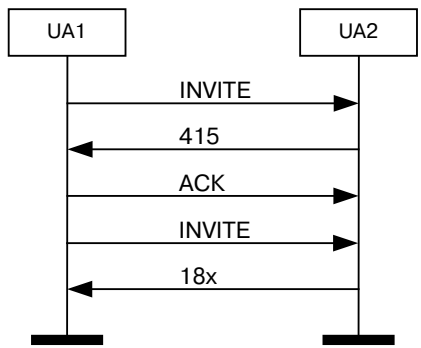


Рис. 11.8. Сценарий с предпочтительной поддержкой ISUP

```
UA1         UA2
INVITE →
Content-type: application/sdp;
Content-disposition: session; handling=required;
Content-type: application/isup;
Content-disposition: signal; handling=required;)

← 415
(Accept: application/sdp)
ACK →

INVITE →
(Content-type: application/sdp)
← 18x

UA1         UA2
INVITE → (Content-type:multipart/mixed;
```

И наконец, на рис.11.9 представлен сценарий, для которого поддержка ISUP необходима для входящего вызова. Здесь UA2 не поддерживает ISUP и передает сообщение об ошибке 415 (Unsupported Media Type). Тогда UA1 перенаправляет запрос к UA 3.

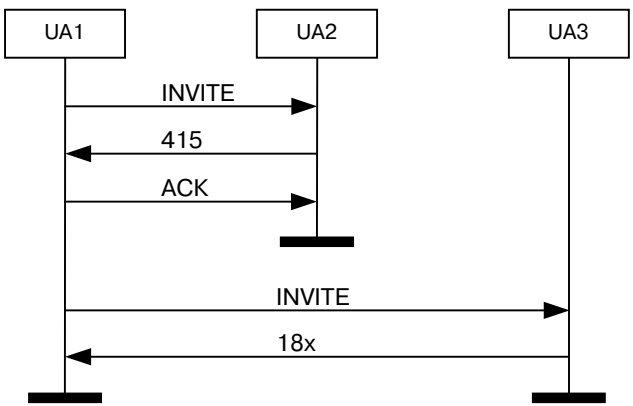


Рис. 11.9. Сценарий с обязательной поддержкой ISUP

```
UA1          UA2
INVITE → (Content-type:multipart/mixed;
Content-type: application/sdp;
Content-disposition: session; handling=required;
Content-type: application/isup;
Content-disposition: signal; handling=required;)
← 415
(Accept: application/sdp)
ACK →
UA1          UA3
INVITE → (Content-type:multipart/mixed;
Content-type: application/sdp;
Content-disposition: session; handling=required;
Content-type: application/isup;
Content-disposition: signal; handling=required;)
```

11.5. Процедуры обеспечения безопасности

Протокол SIP-T может работать как внутрисетевой сигнальный механизм, подчиняясь установленным правам доступа между административными доменами. В большинстве сетей связи использование ISUP ограничивается лицензиями операторов связи, использование же протокола SIP-T может привести к значительному усложнению механизма передачи сообщения конечному пользователю. Каждый административный домен, использующий SIP-T, должен иметь соответствующие механизмы защиты, включающие в себя элементы обнаружения и исключения несанкционированных запросов, чтобы гарантировать, что передача вложенного сообщения ISUP не приводит к нарушению безопасности.

Сообщения ISUP, инкапсулированные в запросы SIP, должны быть защищены с использованием средств защиты тел сообщения S/MIME, обсуждавшихся в главе 7 и описанных в основной спецификации протокола SIP в RFC 3261 [57], а также в RFC 2976 [11]. Процедуры аутентификации S/MIME гарантируют конечному пользователю, что сообщение ISUP было отправлено зарегистрированным терминалом. Шифрование же обеспечивает то, что правильно распознать вложенное ISUP-сообщение смогут только операторы, имеющие необходимый ключ.

Глава 12. Преобразование ISUP–SIP

12.1. Общие принципы взаимодействия

Протокол SIP обычно работает поверх протокола IP, разговор пользователей рассматривается как мультимедийный сеанс связи, включающий в себя передачу аудиоинформации. ISUP работает в стеке протоколов ОКС7 поверх подистемы MTP [28], но может также функционировать на базе технологий IP согласно [69]. Как подробно изложено в другом справочнике [27] этой серии, протокол ISUP предназначен для управления соединениями, а также для эксплуатационного управления сетью.

Устройство, содержащее модуль преобразования сообщений между протоколами ISUP и SIP, называется *Softswitch*; используются также термины *Call Agent, Telephone Server, Media Gateway Controller (MGC)*. Softswitch имеет логический интерфейс для работы с сетями обоих типов – коммутации каналов с протоколом ISUP и коммутации пакетов с протоколом SIP. Преобразованием аудиоинформации из формата, принятого в сети SIP, в формат ТфОП занимается Media Gateway (MG) с интерфейсом E1 со стороны ТфОП и интерфейсом IP со стороны пакетной сети; он выполняет это преобразование под управлением Softswitch, который используется как связующее звено между сетями ТфОП и SIP. Речь идет о том, что вызовы из телефонной сети могут поступать на SIP-телефоны и наоборот; кроме того, вызовы из ТфОП могут проходить транзитом через сеть SIP.

Взаимодействие этих двух сетей основано на инкапсуляции сообщений ISUP в тело запросов SIP и на преобразовании части информации сообщения ISUP, необходимой для правильной маршрутизации, в заголовок запроса SIP.

12.2. Требования к SIP при взаимодействии с ТфОП

Для того чтобы преобразование сообщений из формата ISUP в формат SIP и наоборот проходило правильно и без ошибок, используются несколько механизмов, рассматриваемых ниже. Если SIP UAC/UAS получает сообщение в том формате, которого он не поддерживает, установление соединения еще возможно, но при этом сценарий обмена сообщениями будет отличаться от стандартного.

12.2.1. Процедуры прозрачной передачи сообщений ISUP

Чтобы позволить шлюзам использовать при транзите сообщений ТфОП через сеть SIP (ТфОП–SIP–ТфОП) полный набор услуг, предоставляемых существующей телефонной сетью, запрос SIP должен быть способен транспортировать полезную нагрузку от шлюза к шлюзу в виде инкапсулированного сообщения ISUP.

12.2.2. Процедуры поддержки формата MIME

В большинстве случаев взаимодействия с ТфОП, сообщение SIP транспортирует информацию для установления медиа-соединения (по протоколу SDP), а также информацию ISUP и/или биллинговые данные. Узлы в сети SIP должны поддерживать тип тела сообщения «multipart/mixed», описанный в [18]. Поддержка этого формата для клиента обеспечивается путем добавления значения «multipart/mixed» в заголовок **Accept**.

12.2.3. Процедуры передачи многочастотного набора DTMF

При взаимодействии сети SIP и сети ТфОП часто возникает задача передавать сигналы DTMF. Передача таких сигналов через сеть SIP вызывает ряд проблем.

Кодек, который используется в сети SIP для сжатия речи, не может использоваться для передачи сигналов DTMF, т.к. эти сигналы могут сильно искажаться. Поэтому должен применятся символьный метод внутриполосной передачи. Этот метод описан в [63].

12.2.4. Процедуры проключения речевых трактов в предответном состоянии

В предответном состоянии пользователю может передаваться аудиоинформация. Предответным, по идущей еще от сигнализации R1.5 [30] традиции, называют состояние соединения до того, как оно будет установлено полностью, т.е. будет получен окончательный ответ 2xx.

При работе с сетью ТфОП часто бывает необходимо проключение медиапотока в обратном направлении для того, чтобы пользователю могли передаваться мелодии и речевые сообщения. Нужно отметить, что сообщение INVITE практически всегда содержит вложение SDP, которое достаточно для того, чтобы проключить речевой тракт в обратном направлении, т.е. описание SDP может содержать указание UA быть готовым принимать медиа-данные сразу после того, как будет получено сообщение INVITE. Основные процедуры протокола SIP позволяют создавать простые однонаправленные речевые тракты в предответном состоянии. Однако этот механизм имеет множество ограничений и недостатков – например, медиапотоки, описанные данными протокола SDP в запросе INVITE, не могут быть изменены или удалены, и двунаправленный тракт RTCP, необходимый для организации сеанса, не может быть установлен. Поэтому шлюзы должны поддерживать более сложные методы проключения речевых трактов. Один из таких методов описан в [51].

12.2.5. Обмен транзакциями во время активного сеанса

Здесь речь идет об обмене во время активного сеанса транзакциями, не изменяющими состояния конечного автомата протокола SIP. Дело в том, что при взаимодействии с ТфОП могут возникать ситуации, когда необходимо передать сигнальные сообщения в то время, когда речевой тракт уже проключен. Для этого используется запрос INFO, так как использование одного из 6 основных запросов SIP приведет к обрыву соединения. Подробное описание INFO приведено в § 3.3.8. Там же отмечено, что все шлюзы в сети должны уметь обрабатывать этот запрос.

Шлюзы должны также интерпретировать ответы 405 (Method Not Allowed) и 501 (Not Implemented) не как ответы на запросы INFO, приводящие к разрушению соединения, то есть если второй шлюз не поддерживает полученный запрос INFO, то соединение не прерывается.

12.2.6. Поддержка механизмов обеспечения конфиденциальности

ISUP имеет средства, с помощью которых пользователь может определить, хочет ли он, чтобы его номер мог узнать вызываемый абонент. Когда шлюзы получают запрос с запретом предоставления номера, они должны скрыть идентификатор отправителя сообщения.

Базовые возможности протокола SIP позволяют вводить анонимных пользователей. Однако эта система имеет множество ограничений – например, в этом случае идентификатор шлюза передается в открытом виде, что может привести к раскрытию анонимности. Поэтому шлюзы могут поддерживать более сложные методы защиты информации. Такие механизмы, поддерживающие полностью засекреченную передачу информации, уже рассматривались выше.

12.2.7. Информация о причинах, вызвавших передачу запроса CANCEL

В ISUP имеется только одно сообщение, которое может прервать процесс установления соединения – сообщение REL общего назначения. Подобная концепция существует и в SIP – запрос CANCEL передается для того, чтобы прекратить диалог установления соединения. Однако запрос CANCEL не может содержать вложений, и потому сообщение REL не может быть инкапсулировано в этот запрос.

Описанная проблема возникает редко, так как на практике REL передается, если абонент не дождался ответа и положил трубку, или после завершения разговора. Но бывают исключительные случаи, такие как отказ сети и т.п., в которых код причины, содержащийся в REL, отличен от обычного значения 16 (Normal clearing). Для таких случаев шлюзы могут поддерживать механизмы передачи соответствующего кода причины. Один из этих механизмов – заголовок **Reason** – описан в § 3.2.48.

12.3. Процесс обмена сообщениями

Приведенные далее сценарии иллюстрируют порядок следования сообщений для типичного примера успешного установления соединения или незавершенного в результате ошибки соединения в случаях, когда соединение

инициируется сетью ТфОП. Ответ 100 (Trying), передаваемой в то время, пока не придет ответ на INVITE, на рисунках этого параграфа не изображается, так как его наличие не обязательно и зависит от конфигурации сети.

На диаграммах вся сигнализация (ISUP и SIP) идет через контроллер медеашлюза MGC (Softswitch); управление медиапотоком (например, освобождение каналов) производится медиашлюзом MG под управлением MGC. Для упрощения диаграмм здесь представлено одно устройство, названное «MGC/MG».

12.3.1. Установление соединения (быстрый ответ не используется)

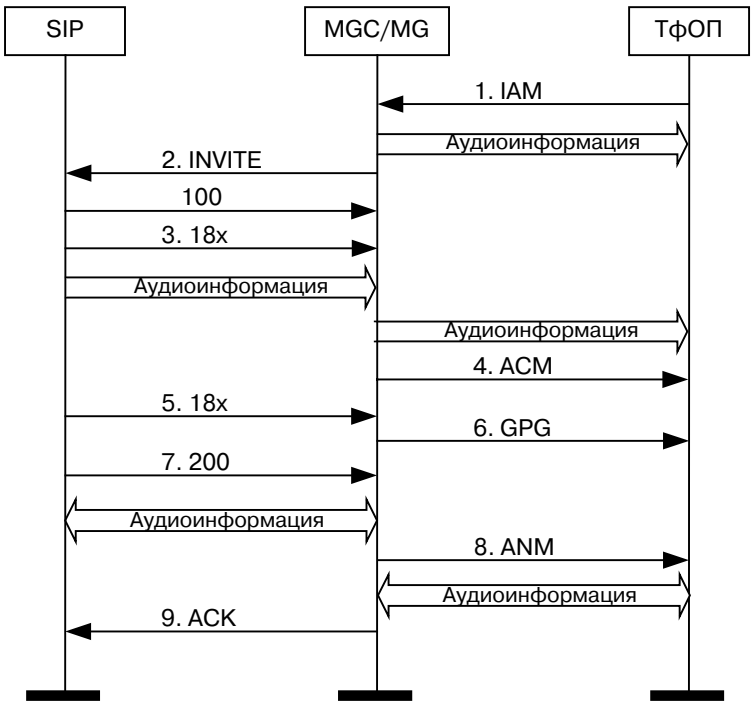


Рис. 12.1. Установление соединения (быстрый ответ не используется)

1. При вызове абонентом сети ТфОП пользователя сети SIP из ТфОП (см. рис. 12.1.) передается сообщение IAM в направлении нужного шлюза.
2. После получения IAM шлюз передает INVITE соответствующему узлу в сети SIP.
3. Если адресной информации достаточно, узел SIP генерирует предварительный ответ 18x.
4. При получении ответа группы 18x шлюз передает в сеть ТфОП сообщение ACM. Если получен ответ, отличный от 180 (Ringing), то ACM будет содержать в параметре called party status (статус вызываемого абонента) значение no indication (нет признака).
5. Чтобы указывать состояние процесса обслуживания вызова, узел сети SIP использует предварительные ответы.
6. После того как ACM будет передано, все временные ответы преобразуются в сообщения ISUP CPG, как будет показано далее в § 12.3.2.
7. После того как узел в сети SIP ответит на вызов, будет передано сообщение 200 OK.
8. При получении 200 OK шлюз передает в сеть ТфОП сообщение ANM.
9. Шлюз передает узлу сети SIP сообщение ACK с подтверждением параметров, переданных в 200 OK.

12.3.2. Установление соединения (быстрый ответ)

1. В сценарии на рис.12.2, когда абонент ТфОП вызывает пользователя сети SIP, из ТфОП передается сообщение IAM в направлении нужного шлюза.
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и передает его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
3. Далее, если в узле сети SIP установлен автоматический ответ, шлюзу передается сообщение 200 OK.
4. После получения сообщения 200 OK шлюз передает сообщение CON тому узлу ТфОП, который запрашивал установление соединения.
5. Шлюз передает узлу сети SIP сообщение ACK с подтверждением параметров, переданных в 200 OK.

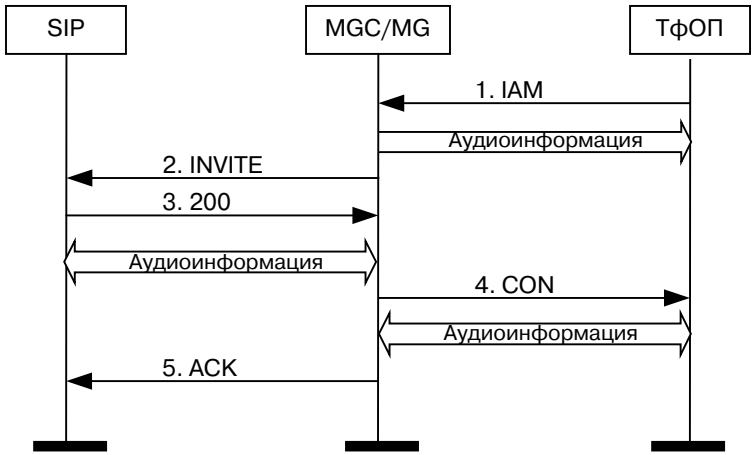


Рис. 12.2. Установление соединения (быстрый ответ)

12.3.3. Таймеры протокола SIP

Теперь рассмотрим работу таймеров при установлении соединения из ТфОП в SIP-сеть.

1. Когда абонент ТфОП вызывает пользователя сети SIP, в направлении соответствующего шлюза из ТфОП передается сообщение IAM .
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и передает его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера. Включаются таймер для ISUP T11 и таймер для SIP T1.
3. Каждый раз при срабатывании таймера T1 сообщение INVITE передается узлу SIP еще раз. В основном документе по протоколу SIP [57] определено, что сообщение INVITE может передаваться 7 раз.
4. При срабатывании таймера T11 к узлу ТфОП передается сообщение ACM для начала прерывания устанавливаемого соединения. При этом в удаленном узле ISUP включается таймер T7. В поле «Called Party Status» сообщения ACM записано значение «no indication» (нет признака).

5. Как только будет передано максимально возможное число сообщений INVITE, шлюз передает узлу ТфОП сообщение REL для прекращения установления соединения.
6. Шлюз передает также сообщение CANCEL узлу SIP для прекращения тем любых попыток ответа.
7. После получения сообщения REL узел ТфОП передает шлюзу сообщение RLC для подтверждения.

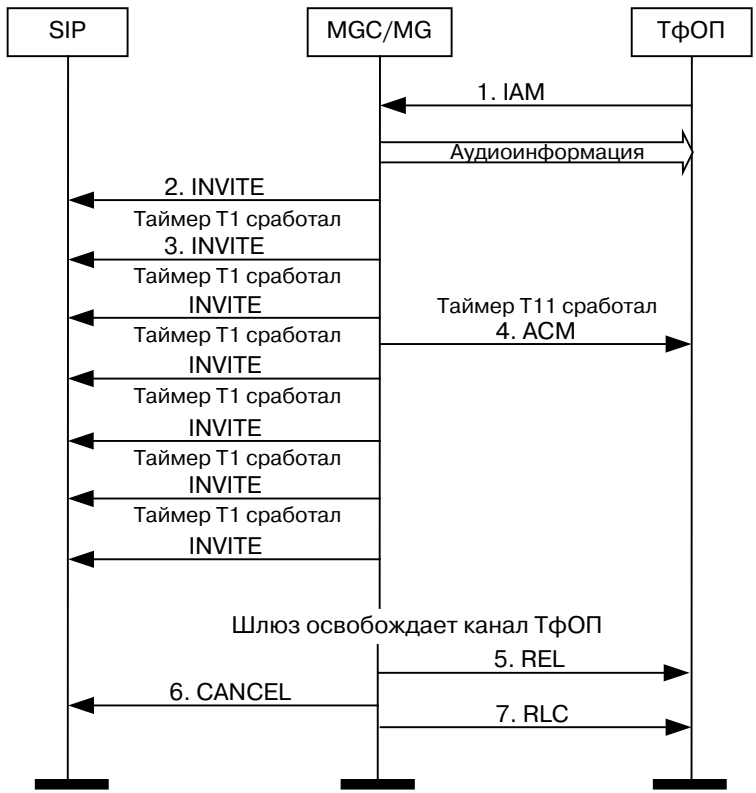


Рис. 12.3. Таймеры SIP

12.3.4. Срабатывание таймера ISUP T9

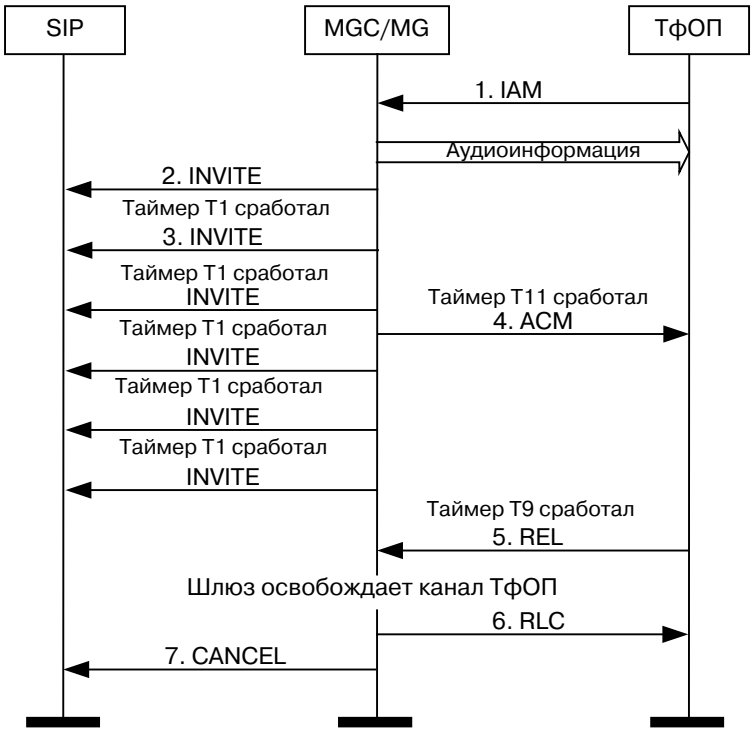


Рис. 12.4. Срабатывание таймера T9

1. Когда абонент ТфОП вызывает пользователя сети SIP, в направлении нужного шлюза из ТфОП передается сообщение IAM.
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и передает его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера. Таймер ISUP T11 и таймер SIP T1 начинают отсчитывать время.

3. Каждый раз при срабатывании таймера T1 сообщение INVITE передается еще раз. С момента запуска T1 проходит 0.5 секунды или больше, после чего сообщение передается опять. Если для T1 задана длительность более 500 мс, то возможно, что времени на передачу 7 запросов INVITE потребуется больше, чем нужно для срабатывания таймеров ISUP T11 + ISUP T9.
4. При срабатывании таймера T11 узлу ТфОП передается сообщение ACM для начала прерывания устанавливаемого соединения. При этом на удаленном узле ISUP включается таймер T7. В поле Called Party Status (статус вызываемой стороны) сообщения ACM записано значение «no indication» (нет признака).
5. Когда таймер T9 в узле ТфОП срабатывает, шлюзу передается сообщение REL.
6. После получения сообщения REL шлюз передает узлу ТфОП сообщение RLC для подтверждения.
7. Сообщение REL указывает шлюзу также, что надо передать CANCEL узлу сети SIP.

12.3.5. Ошибки в сети SIP при установлении соединения

Теперь рассмотрим ситуацию возникновения ошибки в сети SIP в процессе установления соединения (рис. 12.5).

1. Когда абонент ТфОП вызывает пользователя сети SIP, в направлении нужного шлюза из ТфОП передается сообщение IAM.
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и передает его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
3. Узел сети SIP сообщает об ошибке шлюзу, передавая ответ 400.
4. Шлюз передает узлу сети SIP сообщение ACK для подтверждения параметров, переданных в запросе INVITE.
5. Шлюз передает сообщение REL узлу ТфОП.
6. Узел ТфОП передает RLC в подтверждение приема REL.

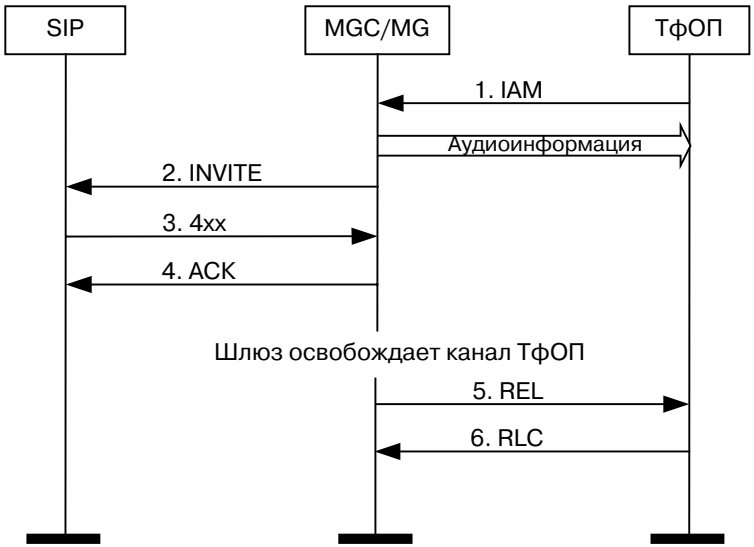


Рис. 12.5. Ошибка в сети SIP

12.3.6. Перенаправление запросов в сети SIP

На рис. 12.6 представлен сценарий перенаправления запросов в сети SIP.

1. Когда абонент ТфОП вызывает пользователя сети SIP, в направлении нужного шлюза из ТфОП передается сообщение IAM.
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и передает его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
3. Передачей к шлюзу сообщения 3xx, узел сети SIP указывает, что пользователь, с которым хочет установить соединение абонент ТфОП, имеет другой адрес. Предполагается, что адресом, на который нужно направлять вызов, является Contact URL.

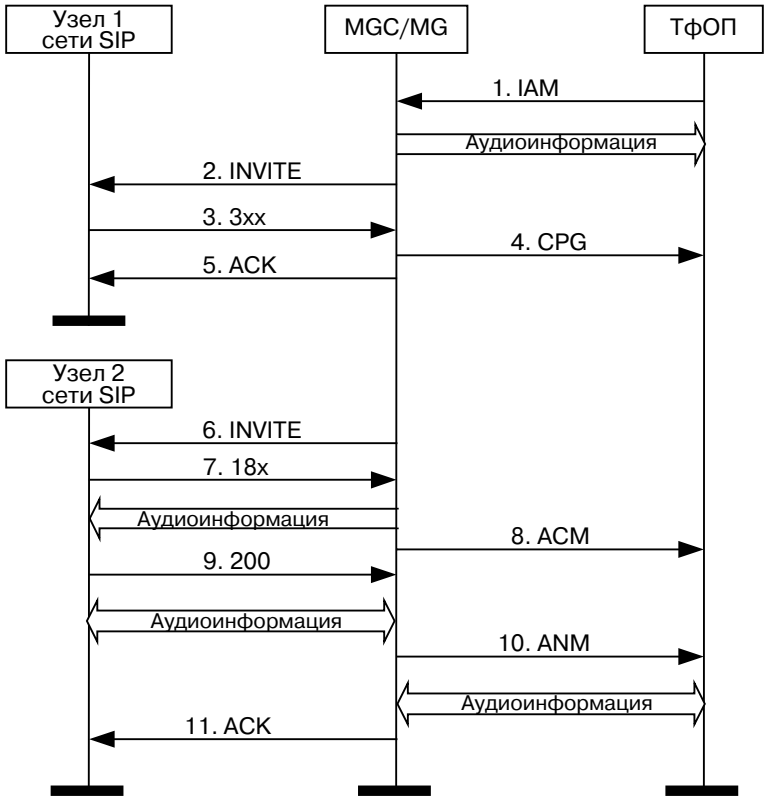


Рис. 12.6. Перенаправление запросов в сети SIP

4. Шлюз передает сообщение CPG с извещением о событии, полученным в сообщении 3xx. Однако при этом надо следить, сможет ли узел корректно распознать запрос, т.к. не все реализации ISUP поддерживают передачу CPG до того, как пришло сообщение ACM.
5. Шлюз передает узлу сети SIP сообщение ACK для подтверждения приема ответа 3xx.
6. Шлюз пересылает запрос по адресу, указанному в поле **Contact** сообщения 3xx.

7. Когда узел сети SIP примет адресную информацию, достаточную для того, чтобы определить, ему ли адресован запрос, он передает предварительное сообщение группы 18x, если адрес правильный.
8. После получения сообщения 180 (Ringing) шлюз передает узлу ТфоП сообщение ACM с кодом этого события.
9. Когда узел сети SIP ответил на вызов, передается сообщение 200 OK.
10. После получения 200 OK шлюз передает сообщение ANM в направлении узла ТфОП.
11. Шлюз передает узлу сети SIP сообщение ACK для подтверждения параметров, переданных в 200 OK.

12.3.7. Установление соединения прерывается со стороны ТфОП

1. Когда абонент ТфОП вызывает пользователя сети SIP, в направлении нужного шлюза из ТфОП передается сообщение IAM.
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и передает его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
3. Когда узел сети SIP примет адресную информацию, достаточную для подтверждения правильности доставки запроса INVITE, он передает шлюзу промежуточное сообщение 180 (Ringing).
4. После получения 180 (Ringing) шлюз передает в ТфОП сообщение ACM.
5. Если вызывающая сторона (т.е. абонент ТфОП) положит трубку раньше, чем пользователь сети SIP ответит на вызов, генерируется сообщение REL.
6. Шлюз освобождает тракт передачи в ТфОП и идентифицирует его как доступный для дальнейшего использования, передавая сообщение RLC.
7. При получении сообщения REL до того, как был получен ответ на запрос INVITE, шлюз передает узлу сети SIP сообщение CANCEL.
8. При получении сообщения CANCEL узел сети SIP передает в подтверждение ответ 200 OK.

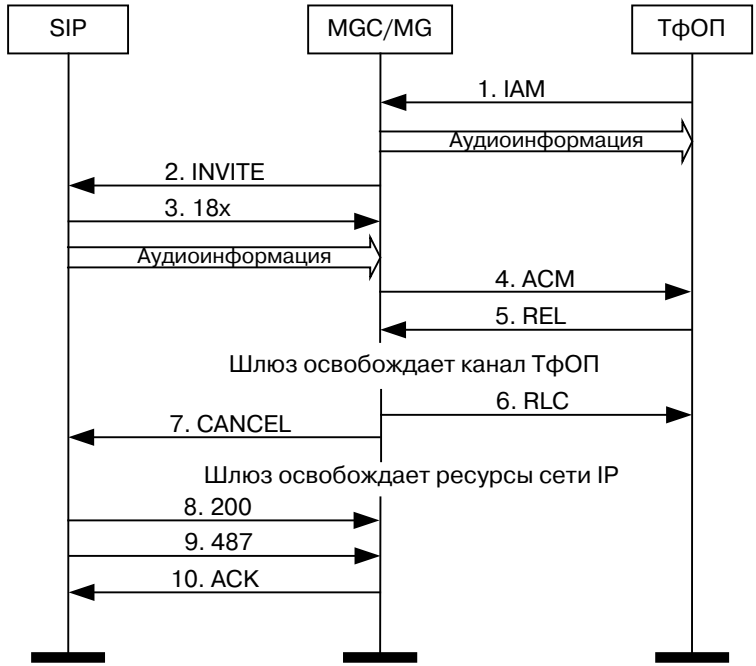


Рис. 12.7. Установление соединения прерывается со стороны ТфОП

9. Узел сети SIP передает ответ 487 (Call Cancelled) для подтверждения окончания обработки запроса INVITE.
10. Шлюз передает узлу сети SIP сообщение ACK для подтверждения приема ответа 487.

12.4. Конечные автоматы SIP-T при взаимодействии ISUP–SIP

12.4.1. Начало получения адресной информации

После получения сообщения IAM, шлюз должен резервировать подходящий внутренний ресурс, например, цифровой сигнальный процессор DSP и подобные ему устройства, необходимые для управления передачей сообщений в IP-части устанавливаемого соединения.

12.4.2. Процедуры преобразования сообщения IAM в запрос INVITE

Если шлюз получает сообщение IAM, он должен создать сообщение INVITE для передачи в сеть SIP. Далее описывается, каким образом шлюз заполняет все поля INVITE, основываясь на параметрах пришедшего сообщения IAM.

Первое, что должно быть взято из IAM при создании INVITE, это два универсальных идентификатора ресурса (URI), один – адрес вызывающего абонента ТфОП, второй – адрес вызываемого пользователя сети SIP. Первый адрес помещается в поле **From** сообщения INVITE, после того как нужная информация будет извлечена из формата ISUP, а второй адрес в большинстве случаев помещается как в заголовок **To**, так и в поле Request URI.

Адрес вызываемого пользователя должен быть помещен в tel URL, который используется в поле Request URI сообщения INVITE. Шлюз может также попытаться использовать функцию Telephone Number Mapping преобразования телефонного номера (ENUM) вызываемого пользователя в SIP URI согласно [14]. После завершения преобразования к телефонному URI должны быть добавлены некоторые дополнительные поля ISUP.

Речь идет о следующем. Если сеть поддерживает услугу, обеспечивающую перенос номера согласно документу «Number Portability in the Global Switched Telephone Network (GSTN): An Overview. RFC 3482» [15], то при анализе сообщения шлюз должен обработать еще несколько параметров. Бит «number translated» параметра FCI показывает, что услуга переноса номера была использована. Информация об этом должна быть отражена в сообщении SIP. Для этого к полю tel URL добавляется параметр «npdi=yes». Если в сообщении IAM содержится параметр GAP, то содержимое поля номера вызываемого абонента (CPN) (в данном

случае Location Routing Number-RN) должно быть переведено из формата ISUP и скопировано в поле «rn=», которое также добавляется в поле tel URL.

Значение параметра GAP также должно быть переведено из формата ISUP и использовано для заполнения поля главного телефонного номера в tel URL (т.е. tel URI). В некоторых национальных спецификациях (например, ANSI) оба номера (LRN и номер вызываемого абонента) могут находиться в параметре CPN и должны быть разделены для размещения в разных полях tel URL. LRN всегда используется на территории одной страны, и, следовательно, поле «rn=» не должно содержать знака «+». Кроме того, могут добавляться поля, которые используются при маршрутизации по идентификатору сети/оператора согласно [6].

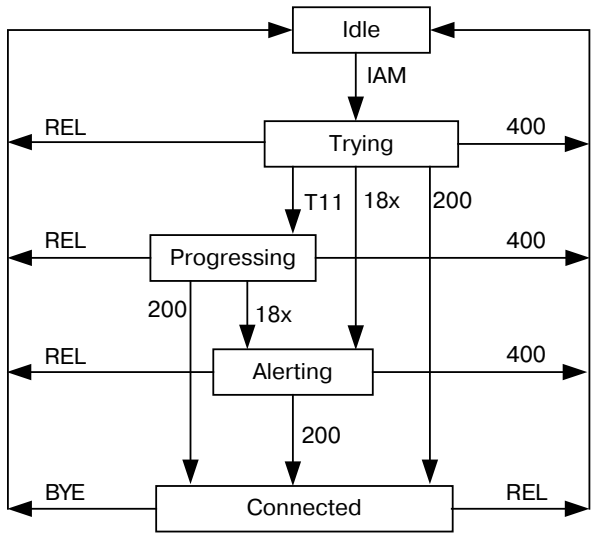


Рис. 12.8. Конечный автомат SIP-T при взаимодействии ISUP–SIP

В большинстве случаев окончательное значение tel URL содержится в поле To и Request-URI, переданным шлюзом. Однако когда в сообщении IAM содержится параметр OCN, то поле **To** должно быть образовано переводом номера из OCN, и тогда заголовок **To** и поле **Request-URI** могут различаться. Конструкция полей заголовка **From** зависит от наличия параметра ISUP Calling Party's Number (CIN). Если CIN отсутствует, шлюз должен сконструировать поля заголовка **From**,

содержащего SIP URI, без пользовательской части, а только с адресом шлюза (например, *sip:gw.protei.ru*). Если CIN присутствует, то он должен быть преобразован в tel URI, содержащийся в поле заголовка **From**.

12.4.3. Сообщение 100

При приеме сообщения 100 в ТфОП не должно передаваться никаких сообщений; это служебный ответ, который говорит лишь о том, что надо прекратить передавать INVITE.

12.4.4. Сообщения группы 18x

Если до получения ответа 18x, не содержащего вложений ISUP, сообщение ACM еще не было передано, то сообщения, передаваемые в сторону абонента ТфОП от шлюза, генерируются согласно табл. 12.1. Если сообщение ACM было передано до получения ответа 18x, соединение переходит в состояние «Progressing» вместо «Alerting».

Таблица 12.1

Пришедший ответ	Сообщение, передаваемое шлюзом
180 Ringing	ACM (BCI = subscriber free)
181 Call is being forwarded	Early ACM and CPG, event=6
182 Queued	ACM (BCI = no indication)
183 Session progress message	ACM (BCI = no indication)

Если сообщение ACM уже передано, и 18x не содержит вложений ISUP, то сообщения от шлюза генерируются согласно табл. 12.2.

Таблица 12.2

Пришедший ответ	Сообщение, передаваемое шлюзом
180 Ringing	CPG, event = 1 (Alerting)
181 Call is being forwarded	CPG, event = 6 (Forwarding)
182 Queued	CPG, event = 2 (Progress)
183 Session progress message	CPG, event = 2 (Progress)

При получении ответа 18x шлюз должен послать КПВ вызывающему абоненту ТфОП (за исключением случаев, когда шлюз знает, что КПВ будет передаваться от узла ТфОП). Шлюз может получать медиа-данные в любой момент после того, как будет передан запрос INVITE с вложением SDP, даже до получения временного ответа 18x. Следовательно, шлюз должен быть готов воспроизвести эти данные вызывающему абоненту ТфОП (если необходимо, то прекратить подачу КПВ, и вместо этого передавать медиа-данные).

Если шлюз получает ответ 183 (Session progress) в то время, когда абоненту ТфОП воспроизводятся медиа-данные, он должен преобразовывать это сообщение, если в нем не содержится вложенного пакета ISUP. Созданное сообщение должно содержать параметр, указывающий, что в данный момент передаются медиа-данные (например, параметр Event Information (информация о событии) в сообщении CPG или параметр Optional Backward Call Indicators (необязательные индикаторы, передаваемые в обратном направлении) в сообщении ACM). Перед тем как сообщение ACM будет передано, шлюз должен принудительно установить параметр Backward Call Indicators (обратные индикаторы условий обслуживания вызовов, BCI) и все остальные необязательные параметры в соответствии с политикой безопасности шлюза.

Таблица 12.3

Backward Call Indicators	
Charge indicator:	10 charge
Called party's status indicator:	01 subscriber free or 00 no indication
Called party's category indicator:	01 ordinary subscriber
End-to-end method indicator:	00 no end-to-end method
Interworking indicator:	0 no interworking
End-to-end information indicator:	0 no end-to-end info
ISDN user part indicator:	1 ISUP used all the way
Holding indicator:	0 no holding
ISDN access indicator:	0 No ISDN access
Echo control device indicator:	It depends on the call
SCCP method indicator:	00 no indication

При отсутствии вложенного сообщения параметр BCI должен быть установлен согласно табл. 12.3. Когда параметр BCI в поле Interworking indicator (индикатор наличия взаимодействия) содержит значение interworking encountered (взаимодействие на некоторых участках), это означает, что сеть ISDN взаимодействует с сетью, которая не поддерживает большинство услуг ISDN, и не может использовать функции, которые обычно применяются. Когда параметр BCI в поле Interworking indicator (индикатор наличия взаимодействия) содержит значение interworking encountered (взаимодействие на некоторых участках), это означает, что сеть ISDN взаимодействует с сетью, которая не поддерживает большинство услуг ISDN, и не может использовать функции, которые обычно применяются.

12.4.5. Сообщения группы 2xx

После того как получен ответ 200 OK, шлюз должен создать медиапоток в нужном направлении, а также передать сообщение ANM абоненту ТфОП и сообщение ACK-пользователю сети SIP.

Таблица 12.4

Пришедший ответ	Сообщение, передаваемое шлюзом
200 OK	ACM, ACK

Если ответ 200 OK придет на шлюз до того, как передано сообщение ACM, то вместо ANM будет передано сообщение CON, если используемый вариант ISUP поддерживает такую возможность.

Если в пришедшем ответе 200 OK содержалось понятное шлюзу инкапсулированное сообщение ANM, то шлюз должен заполнить все параметры представляемого абоненту ТфОП сообщения ANM, используя вложенное сообщение.

Обратите внимание, что шлюз может получать 200 OK не только в ответ на запрос INVITE, но также, например, в ответ на запрос INFO. Процедуры, описанные в этом разделе, относятся только к тем 200 OK, которые были получены в ответ на запрос INVITE. Если 200 OK пришел в ответ не на запрос INVITE, шлюз не должен передавать абоненту ТфОП никаких сообщений.

12.4.6. Сообщения группы 3xx

Когда шлюз получает ответ 3xx, он должен определить новое местоположение пользователя, передав один или несколько запросов нового соединения. При этом используются URI, находящиеся в любых полях **Contact** пришедшего ответа. Ответы 3xx обычно передаются сервером переадресации. Если в поле **Contact** ответа 3xx содержится URI, который относится к абоненту ТфОП (т.е. шлюз используется как транзит), шлюз должен передать новый запрос и работать как обычный коммутатор ТфОП (без применения SIP). Кроме этого, в ответ на сообщение 3xx шлюз может передать в ТфОП сообщение REL, содержащее redirection indicator (индикатор перенаправления) и diagnostic field (поле диагностики) с новым телефонным номером из URI. Если же новый адрес пользователя является SIP URI, MGC должен сформировать новое сообщение IAM и инкапсулировать его в новые сообщения INVITE, после чего разослать их по адресам из поля **Contact**. Пока шлюз обрабатывает список адресов из поля **Contact** (генерирует и рассылает новые INVITE), он может передать абоненту ТфОП сообщение CPG с кодом события 6 (перенаправление), если это разрешено в используемом варианте ISUP. Все случаи переадресации должны тщательно анализироваться, так как они могут создавать сложные ситуации, например при биллинге услуг.

12.4.7. Сообщения групп 4xx–6xx

Если на шлюз поступает сообщение групп 4xx–5xx, это означает, что переданный запрос INVITE был отклонен. В этом случае шлюз должен освободить соответствующие ресурсы, передать сообщение REL с нужными параметрами абоненту ТфОП и запрос ACK пользователю сети SIP. В некоторых случаях, описываемых ниже, шлюз может попытаться решить проблему путем повторной передачи исправленного запроса. После того как шлюз передает сообщение REL в ТфОП, он ожидает обратного ответа RLC с подтверждением освобождения всех ресурсов.

12.4.8. Преобразование кодов ответов SIP в коды событий ISUP

Если в сообщении 4xx, пришедшем на шлюз, содержалось вложенное REL, то из него должен быть извлечен параметр Cause Indicator (индикатор причины, CAI), а его значение использовано для заполнения этого же параметра сообщения REL, передаваемого далее в ТфОП.

Таблица 12.5

Полученный ответ	Код значения в сообщении REL
400 Bad Request	41 Temporary Failure
401 Unauthorized	21 Call rejected (*)
402 Payment required	21 Call rejected
403 Forbidden	21 Call rejected
404 Not found	1 Unallocated number
405 Method not allowed	63 Service or option unavailable
406 Not acceptable	79 Service/option not implemented (+)
407 Proxy authentication required	21 Call rejected (*)
408 Request timeout	102 Recovery on timer expiry
410 Gone	22 Number changed (w/o diagnostic)
413 Request Entity too long	127 Interworking (+)
14 Request-URI too long	127 Interworking (+)
415 Unsupported media type	79 Service/option not implemented (+)
416 Unsupported URI Scheme	127 Interworking (+)
420 Bad extension	127 Interworking (+)
421 Extension Required	127 Interworking (+)
423 Interval Too Brief	127 Interworking (+)
480 Temporarily unavailable	18 No user responding
481 Call/Transaction Does not Exist	41 Temporary Failure
482 Loop Detected	25 Exchange - routing error
483 Too many hops	25 Exchange - routing error
484 Address incomplete	28 Invalid Number Format (+)
485 Ambiguous	1 Unallocated number
486 Busy here	17 User busy
487 Request Terminated	--- (no mapping)
488 Not Acceptable here	--- by Warning header
500 Server internal error	41 Temporary failure
501 Not implemented	79 Not implemented, unspecified
502 Bad gateway	38 Network out of order
503 Service unavailable	41 Temporary failure
504 Server time-out	102 Recovery on timer expiry
504 Version Not Supported	127 Interworking (+)
513 Message Too Large	127 Interworking (+)
600 Busy everywhere	17 User busy
603 Decline	21 Call rejected
604 Does not exist anywhere	1 Unallocated number
606 Not acceptable	--- by Warning header

Если ответ пришел без вложенного сообщения, рекомендуется использовать преобразование кодов ответов SIP в коды причины ISUP. Любой код ответа сети SIP, не указанный выше, должен быть преобразован в код причины 31 (Normal, unspecified). Это относится только к ответам. Запросы BYE и CANCEL, которые используются для завершения диалога, в большинстве случаев должны быть преобразованы в код 16 (Normal clearing). Точно так же, как некоторые коды причины ISUP используются только в ТфОП и не имеют аналогов в сети SIP, так и некоторые коды SIP не преобразуются в коды ISUP. Эти коды отмечены знаком (+) в табл. 12.5.

(*) В некоторых случаях шлюз SIP может передать отклик аутентификации к UAS сети SIP, который отклонил запрос INVITE из-за требования аутентификации. Если шлюз может сам авторизовать пользователя, он должен сделать это и продолжить установление соединения. Только в том случае, если шлюзу не удается провести авторизацию пользователя, должно быть передано сообщение с кодом 21 (Call rejected).

(+) Если это возможно, шлюз должен реагировать на ошибки протокола, устраняя их и делая попытку возобновить процесс установления соединения. Процесс следует прервать только в том случае, если шлюз не сможет корректно обработать ошибку.

Расположение участка сети, связанного с параметром CAI, должно быть указано по умолчанию таким образом, что их коды 6xx определяли местоположение пользователя, а коды 4xx и 5xx – расположение сети. Исключения отмечены ниже. Когда в ответах 606 (Not acceptable) и 488 (Not Acceptable here) содержится заголовок **Warning**, при преобразовании в ISUP должна учитываться информация в этих заголовках. В большинстве случаев при преобразовании рекомендуется использовать код события 31 (Normal, unspecified). Если код **Warning** несет информацию о том, что информацию невозможно доставить, т.к. запрошенные средства передачи не поддерживаются (т.е. технически не реализованы), для преобразования рекомендуется использовать код причины 65 (Bearer Capability Not Implemented).

12.4.9. Получение сообщения REL

Сообщение может передаваться, когда абонент ТфОП кладет трубку до того, как был получен ответ на вызов, поэтому шлюз прерывает установление соединения. Пользователю SIP необходимо передать запрос CANCEL; сообщение BYE в данном случае не используется, т.к. не было получено финального сообщения

от пользователя сети SIP. На запрос CANCEL должен придти ответ 200 OK. После этого на переданный запрос INVITE ожидается окончательный ответ 487 (Request Terminated). Шлюз должен сохранять историю обмена сообщениями для этого соединения в течение некоторого времени, начиная с прихода ответа 200 OK на переданный ранее INVITE (даже после приема 200 OK в ответ на запрос CANCEL). В этой ситуации, вслед за соответствующим запросом BYE, шлюз должен передать ACK.

В ситуациях, когда разговорный сеанс связи устанавливается через сеть SIP транзитом, сообщение REL не может быть вложено в запрос CANCEL (так как CANCEL не имеет тела сообщения). Обычно сообщение REL содержит CAI со значением 16 (Normal clearing). Если значение CAI отличается от 16 (Normal clearing), шлюз может использовать другой метод обработки кодов.

12.4.10. Срабатывания таймера ISUP T11

Чтобы не допустить срабатывания удаленного таймера ISUP T7, шлюз может иметь свой собственный управляющий таймер. ISUP определяет этот таймер как T11. Значение таймера T11 тщательно подобрано таким образом, что оно всегда меньше значения таймера T7 любого узла, взаимодействующего со шлюзом в данный момент. Если шлюз поддерживает использование таймера T11, то при его срабатывании должно быть передано предварительное сообщение ACM (early ACM) (т.е. параметр called party status – статус вызываемой стороны – имеет значение «no indication»), чтобы предотвратить срабатывание таймера T7 удаленного узла ТфОП, если используемый вариант ISUP это поддерживает. Если после этого приходит ответ 180 (Ringing), должно быть передано соответствующее сообщение CPG, как это было показано выше в § 12.4.4.

12.5. Примеры сценариев для случая соединения ТфОП–SIP

В этом разделе приводится несколько примеров типичных сценариев с подробным описанием сообщений. Аппарат абонента Maxim находится в ТфОП, использующей вариант сигнализации ISUP, и подключается к сети SIP через АТС A и шлюз NGW 1. Для сокращения количества сообщений на диаграммах показан только один прокси-сервер, т.е. не показано взаимодействие прокси-сервера и сервера определения местоположения.

12.5.1. Успешное соединение абонента ТфОП с пользователем сети SIP

Пользователь Maxim через АТС А, шлюз NGW 1 и прокси-сервер Proxy 1 вызывает пользователя Anton. После того как пользователь Anton отвечает на вызов, устанавливается прямое двухстороннее мультимедийное соединение между абонентом ТфОП и пользователем сети SIP.

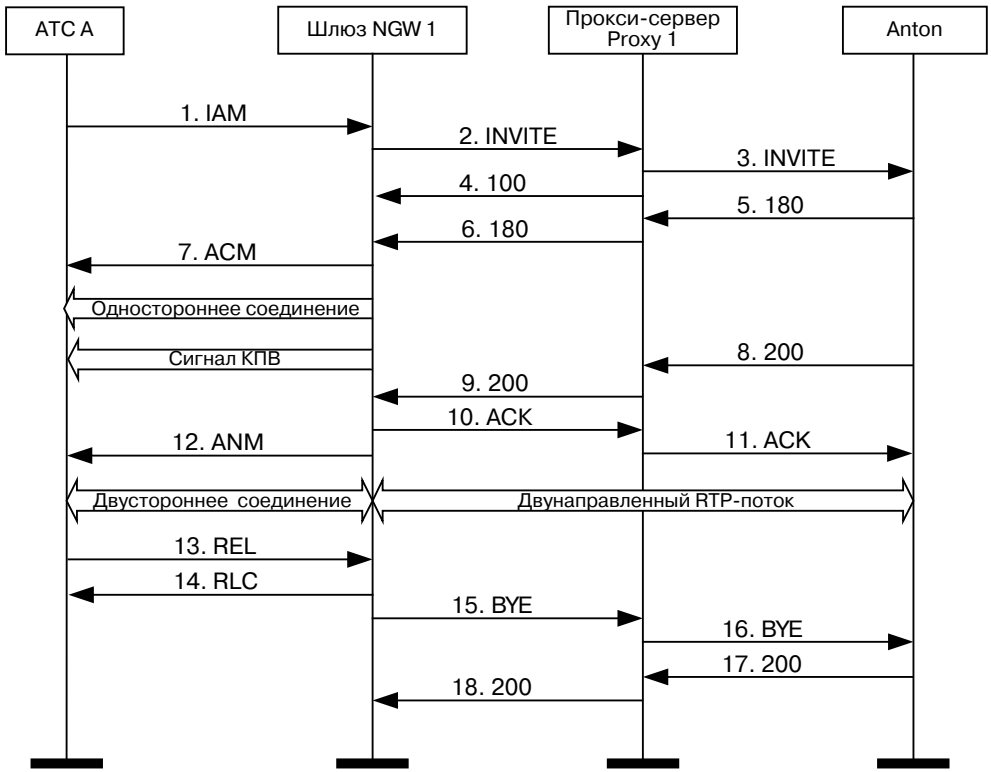


Рис. 12.9. Установление соединения абонента ТфОП с пользователем сети SIP

Разговор прерывается, когда абонент Maxim кладет трубку, АТС передает сообщение REL на NGW 1, где оно преобразуется в BYE для сети SIP. Содержимое сообщений:

1. IAM ATC A → NGW 1

```
Получение сообщения IAM
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
```

2. INVITE NGW → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения местонахождения, чтобы найти пользователя Anton. Анализ местоположения пользователя проводится на основании номера вызываемого пользователя Anton. NGW 1 готовится принимать данные на порт 3456 от терминала абонента Maxim.

3. INVITE Proxy 1 → Anton

```
INVITE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
```

```
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying Proxy 1 → NGW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. 180 Ringing Anton → Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Length: 0
```

6. 180 Ringing Proxy 1 → NGW 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Length: 0
```

7. ACM NGW 1 → ATC A

```
Получение сообщения ACM
```

8. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
```

```
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
Contact: <sip:anton@client.b.niits.ru>
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

9. 200 OK Proxy 1 → NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

10. ACK NGW 1→ Proxy 1

```
ACK sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

11. ACK Proxy 1 → Anton

```
ACK sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
```

```
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

12. NGW 1 → ATC A

Получение сообщения ANM.
Между пользователем Maxim и пользователем Anton установлено соединение для RTP-потока (через NGW).
Пользователь Maxim кладет трубку.

13. REL ATC A → NGW 1

Получение сообщения REL с CauseCode=16 Normal.

14. RLC NGW 1 → ATC A

Получение сообщения RLC

15. BYE NGW 1→ Proxy 1

```
BYE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

16. BYE Proxy 1 → Anton

```
BYE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

17. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
```

```
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

18. 200 OK Proxy 1 → NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

12.5.2. Соединение от абонента ТфОП к пользователю сети SIP, быстрый ответ

Представленный на рис. 12.10 сценарий *быстрого ответа* похож на предыдущий сценарий на рис. 12.9. Разница в том, что терминал пользователя Anton отвечает на вызов мгновенно, т.е. от терминала пользователя Anton сразу поступает 200 OK, без предварительного ответа 180 Ringing. Шлюз передает сообщение ANM, пропуская сообщение ACM. Обратите внимание на то, что для протокола ISUP версии ETSI и некоторых других вариантов ISUP вместо ANM должно быть передано сообщение CON.

Содержимое сообщений:

1. IAM ATC A → NGW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
```

2. INVITE NGW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
```

```
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения местонахождения, чтобы найти пользователя Anton. Анализ местоположения проводится на основании номера вызываемого пользователя. NGW 1 готовится принимать данные на порт 3456 от терминала пользователя Maxim.

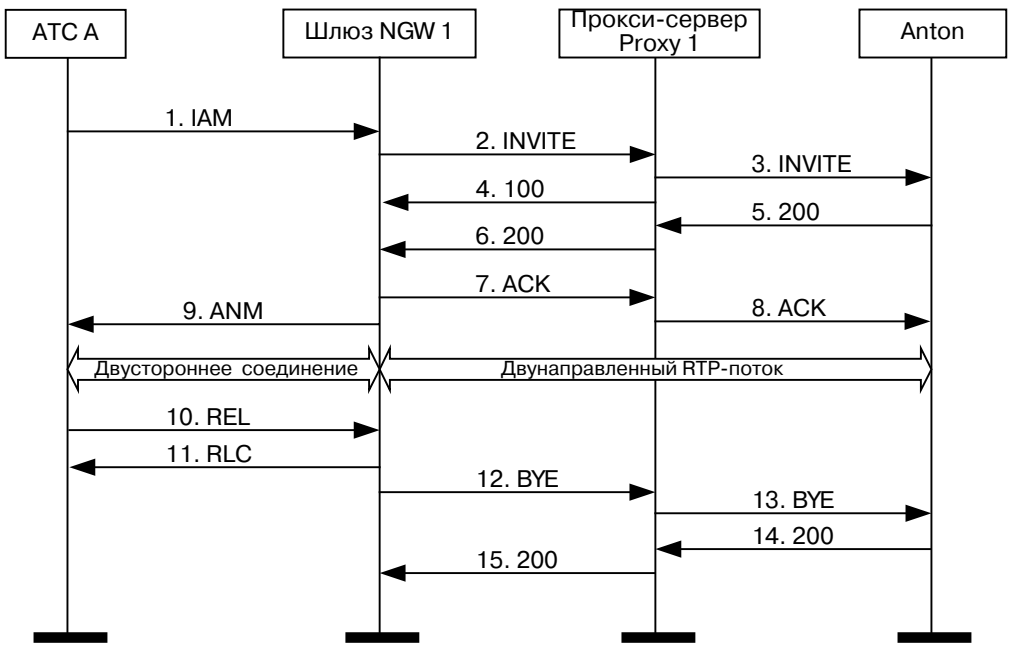


Рис. 12.10. Успешное установление соединения абонента ТфОП с пользователем сети SIP (быстрый ответ)

3. INVITE Proxy 1 → Anton

```
INVITE anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
```

```
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying Proxy 1 → NGW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

6. 200 OK Proxy 1 → NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

7. ACK NGW 1 → Proxy 1

```
ACK anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

8. ACK Proxy 1 → Anton

```
ACK anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=130.131.132.14
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

9. NGW 1 → ATC A

Передается сообщение ANM. Между абонентом Maxim и пользователем Anton установлено соединение для RTP потока (через NGW). Пользователь Maxim кладет трубку.

10. REL ATC A → NGW 1

Передается сообщение REL с CauseCode=16 Normal.

11. RLC NGW 1 → ATC A

```
Передается сообщение RLC.
F12 BYE NGW 1 → Proxy 1
BYE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

13. BYE Proxy 1 → Anton

```
BYE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

14. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

15. 200 OK Proxy 1 → NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

12.5.3. Соединение от абонента УПАТС к пользователю сети SIP

В этом сценарии Maxim вызывает пользователя Anton через УПАТС, шлюз GW1 и Proxy 1. УПАТС и шлюз могут взаимодействовать, используя различные протоколы, а также могут быть объединены физически в одно устройство. Поэтому в данном случае на диаграмме при взаимодействии УПАТС и GW не показаны сообщения определенного протокола, а описаны типы передаваемых сигналов. Шлюз может опознать только группу линий, по которой пришел вызов, и не может идентифицировать линию в УПАТС, по которой он пришел. Поэтому в примере для идентификации номера вызывающего абонента в SIP URI используется sip: *551313@gw1.a.niits.ru.*

Содержимое сообщений:

1. Запрос и набор номера

УПАТС → GW 1

```
Занятие линии.
```

GW 1 → УПАТС

```
Подтверждение занятия линии.
```

УПАТС → GW 1

```
Передача номера вызываемого абонента.
```

2. INVITE GW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
Max-Forwards: 70
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:551313@gw1.a.niits.ru;user=phone>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

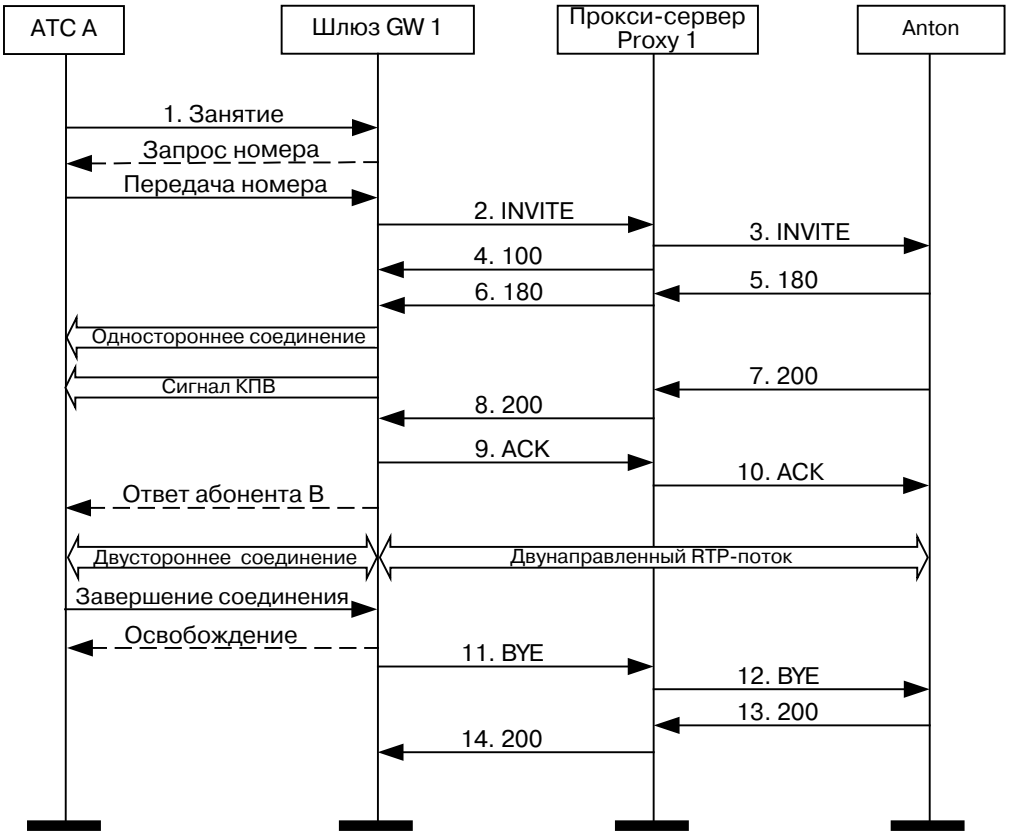


Рис. 12.11. Соединение абонента УПАТС с пользователем сети SIP

4. 100 Trying Proxy 1 → GW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. 180 Ringing Anton → Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Length: 0
```

6. 180 Ringing Proxy 1 → GW 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Length: 0
```

7. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
Contact: <sip:anton@client.b.niits.ru>
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

8. 200 OK Proxy 1 → GW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

9. ACK GW 1 → Proxy 1

```
ACK sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

10. ACK Proxy 1→ Anton

```
ACK sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Max-Forwards: 69
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

11. BYE GW 1 → Proxy 1

```
BYE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
```

```
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

12. BYE Proxy 1 → Anton

```
BYE sip:anton@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Max-Forwards: 69
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

13. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

14. 200 OK Proxy 1 → GW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP gw1.a.niits.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
From: <sip:551313@gw1.a.niits.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

12.5.4. Неуспешное установление соединения из ТфОП в сеть SIP. Пользователь не найден

Абонент Maxim пытается вызвать пользователя Anton через АТС A, GW1 и Proxy 1. Прокси-сервер не находит пользователя с заданным номером и завершает установление соединения, передавая ответ с соответствующим кодом ошибки, который переводится в сообщение REL для абонента Maxim с соответствующим кодом причины.

Рис. 12.12. Неуспешное установление соединения из ТфОП в сеть SIP. Пользователь не найден

Содержимое сообщений:

1. IAM АТС А → GW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-100-2516,NPI=E.164,NOA=National
```

2. INVITE GW1→ Proxy 1

```
INVITE sip:+78121002516@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@gw1.a.niits.ru;user=phone>;tag=076342s
To: <sip:+78121002516@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Contact:
<sip:+70953864515@gw1.a.niits.ru;user=phone;transport=tcp>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.niits.ru
s=-
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует сервер определения местоположения, чтобы найти пользователя +7-812-100-2516. Пользователь с таким идентификатором отсутствует, поэтому Proxy 1 завершает установление соединения.

3. 604 (Does Not Exist Anywhere) Proxy 1→ GW 1

```
SIP/2.0 604 Does Not Exist Anywhere
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@gw1.a.niits.ru;user=phone>;tag=076342s
To: <sip:+78121002516@ss1.a.niits.ru;user=phone>;tag=6a34d410
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 INVITE
Error-Info: <sip:does-not-exist@ann.a.niits.ru>
Content-Length: 0
```

4. ACK GW 1 → Proxy 1

```
ACK sip:+78121002516@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@gw1.a.niits.ru;user=phone>;tag=076342s
To: <sip:+78121002516@ss1.a.niits.ru;user=phone>;tag=6a34d410
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

5. REL GW 1 → ATC A

```
Получение сообщения REL с CauseCode=1.
```

6. RLC ATC A → GW 1

```
Получение сообщения RLC.
```

12.5.5. Неуспешное установление соединения из ТфОП в сеть SIP. Линия занята

В этом сценарии абонент Maxim вызывает пользователя Anton через АТС А, шлюз NGW 1 и Proxy 1. Прокси-сервер находит пользователя Anton и направляет ему вызов. Терминал пользователя Anton отклоняет вызов, передавая ответ 600 (Busy Everywhere). Шлюз передает сообщение REL, содержащее код причины соответствующий пришедшему ответу. Поскольку в сообщении IAM (F1) не содержалось параметра Interworking, сигнал «Занято» будет генерироваться локальным узлом (например, АТС А). В некоторых сценариях, где указано межсетевое взаимодействие, сигнал «занято» подается из шлюза.

Содержимое сообщений:

1. IAM ATC A → NGW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
```

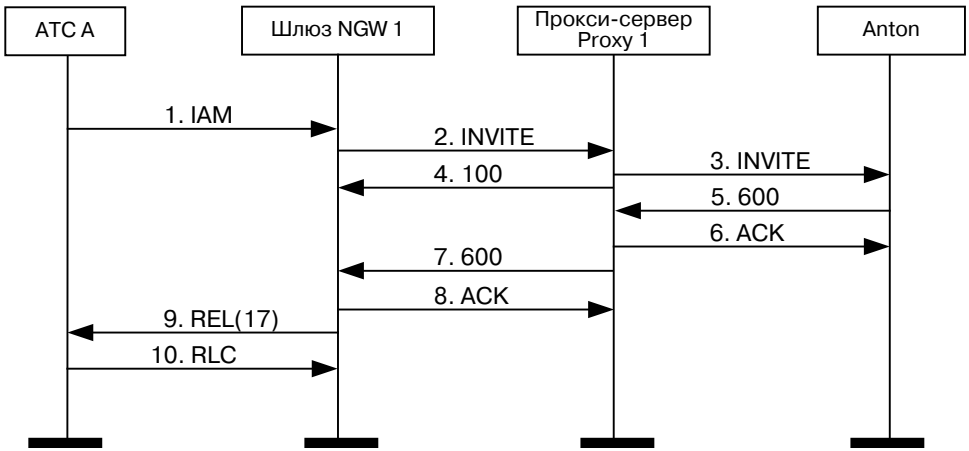


Рис. 12.13. Неуспешное установление соединения из ТфОП в сеть SIP. Линия занята

2. INVITE NGW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 144
v=0

o=GW 2890844527 2890844527 IN IP4 gw1.a.niits.ru
s=-
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

3. INVITE F3 Proxy 1 → Anton

```
INVITE anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
Max-Forwards: 69
```

```
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.niits.ru
s=-
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying Proxy 1 → NGW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. 600 (Busy Everywhere) Anton → Proxy 1

```
SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

6. ACK Proxy 1 → Anton

```
ACK anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

7. 600 (Busy Everywhere) Proxy 1 → NGW 1

```
SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

8. ACK NGW 1 → Proxy 1

```
ACK anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

9. REL NGW 1 → ATC A

```
Получение сообщения REL с CauseCode=17 Busy.
```

10. RLC ATC A → NGW 1

```
Получение сообщения RLC.
```

12.5.6. Неуспешное установление соединения. Линия занята. IAM содержит параметр interworking

Абонент Maxim вызывает пользователя Anton через АТС А, шлюз NGW 1 и Proxy 1. Прокси-сервер находит пользователя Anton и направляет к нему вызов. Его терминал отклоняет пришедший вызов, передавая ответ с кодом ошибки. NGW 1 создает односторонний речевой тракт к абоненту Maxim и передает ему сигнал «занято». Вызывающий абонент кладет трубку. Шлюз передает сигнал «занято», т.к. в сообщении IAM (1) был указан параметр interworking. В предыдущем сценарии в сообщении REL был указан код сигнала «занято», и он передавался абоненту ближайшей к нему АТС, т.к. параметр interworking не был указан.

Содержимое сообщений:

1. IAM ATC A → NGW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
Interworking=encountered
```

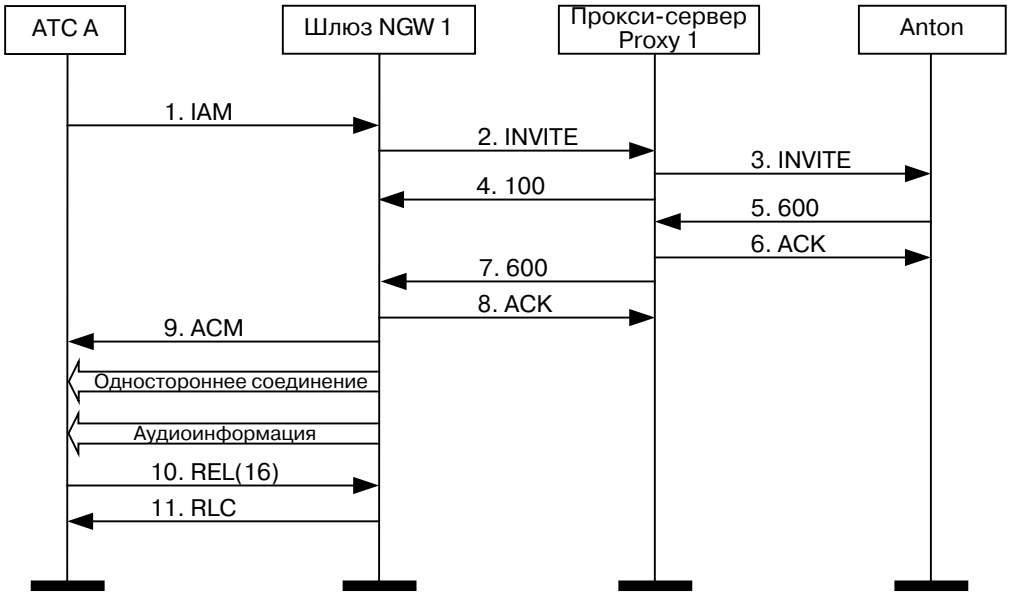


Рис. 12.14. Неуспешное установление соединения из ТфОП в сеть SIP. Линия занята. IAM содержит параметр interworking

2. INVITE NGW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

3. INVITE Proxy 1 → Anton

```
INVITE anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying Proxy 1 → NGW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. 600 Busy Everywhere Anton → Proxy 1

```
SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

6. ACK Proxy 1 → Anton

```
ACK anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

7. 600 Busy Everywhere Proxy 1 → NGW 1

```
SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

8. ACK NGW 1 → Proxy 1

```
ACK sip:ngw1@a.niits.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

9. ACM NGW 1 → ATC A

Передается сообщение ACM.
NGW 1 устанавливает односторонний разговорный тракт к абоненту Maxim. Соединение разрушается после того, как Maxim кладет трубку.

10. REL ATC A → NGW 1

Передается сообщение REL с CauseCode=16

11. RLC NGW 1 → ATC A

Передается сообщение RLC

12.5.7. Неуспешное установление соединения. Срабатывает таймер

Абонент Maxim вызывает пользователя Anton через ATC A, NGW 1 и Proxy 1. Сообщение INVITE передается еще раз, если Anton не ответил на предыдущее до срабатывания таймера T1. Anton не отвечает на все запросы INVITE до срабатывания таймера в ТфОП. После того как таймер в ТфОП сработал, установление соединения прекращается, и передается сообщение REL. Шлюз преобразует это сообщение в CANCEL.

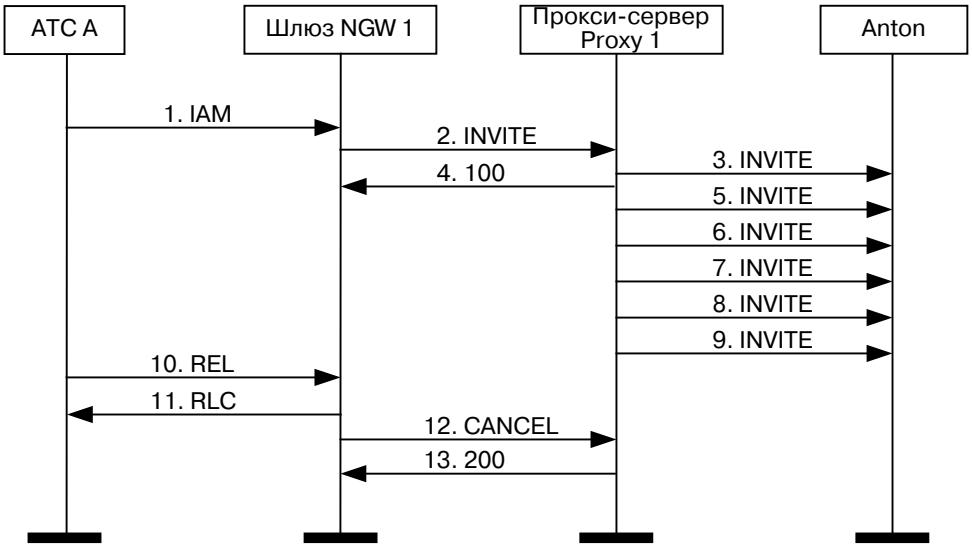


Рис. 12.15. Неуспешное установление соединения. Срабатывает таймер

Содержимое сообщений:

1. IAM АТС А → NGW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
```

2. INVITE NGW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
```

```
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

3. INVITE Proxy 1→ Anton

```
INVITE sip:anton@b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
c c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap
:0 PCMU/8000
```

4. 100 Trying Proxy 1 → NGW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. – 9. INVITE Proxy 1 → Anton

```
Аналогично сообщению 3.
Сработал таймер в ТфОП.
```

10. REL ATC A→ NGW 1

```
Передается сообщение REL с CauseCode=16 Normal.
```

11. RLC NGW 1 → ATC A

```
Передается сообщение RLC.
```

12. CANCEL NGW 1 → Proxy 1

```
CANCEL sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
```

```
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

13. 200 OK Proxy 1 → NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

12.5.8. Неуспешное установление соединения. Срабатывает таймер. Прокси-сервер в режиме без сохранения состояния

В этом сценарии абонент Maxim вызывает пользователя Anton через АТС A, NGW 1 и Proxy 1. Так как прокси-сервер находится в режиме без сохранения состояний, он не использует ответ 100 (Trying). NGW 1 повторно передает INVITE, когда срабатывает таймер SIP T1. Терминал пользователя Anton не отвечает на запросы. При срабатывании таймера в сети ТфОП установление соединения завершается, и передается сообщение REL (CauseCode=102 Timeout).

Содержимое сообщений:

1. IAM АТС A → NGW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
```

2. INVITE NGW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146
```

```
v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

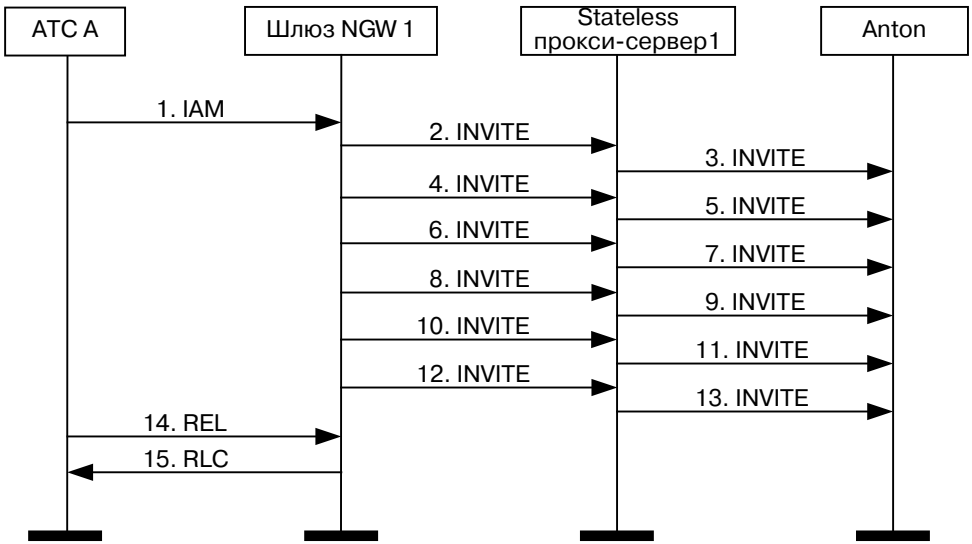


Рис. 12.16. Неуспешное установление соединения. Срабатывает таймер. Прокси-сервер в режиме без сохранения состояния

3. INVITE Proxy 1 → Anton

```
INVITE sip:anton@b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146
```

```
v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. –13. INVITE NGW 1 → Proxy 1

```
Аналогично сообщению 2.
Срабатывает таймер в ТфОП.
```

14. REL ATC A → NGW 1

```
Получение сообщения REL с CauseCode=102 Timeout.
```

15. RLC NGW 1 → ATC A

```
Получение сообщения RLC.
```

12.5.9. Неуспешное установление соединения. Вызывающий абонент кладет трубку, не дождавшись установления соединения

Абонент Maxim вызывает пользователя Anton через ATC A, NGW 1 и Proxy 1. Вызов достиг вызываемого абонента, но тот не снимает трубку. NGW 1 проключает односторонний тракт и передает абоненту Maxim акустический сигнал «контроль посылки вызова», т.к. в сообщении IAM присутствует параметр interworking. Maxim кладет трубку, не дождавшись ответа, после чего передается сообщение REL, которое преобразуется в сообщение CANCEL. Если терминал пользователя Anton передает ответ 200 OK после того, как пришло сообщение REL, NGW 1 сначала передает сообщение ACK, а только потом BYE для правильного завершения установления соединения.

Содержимое сообщений:

1. IAM ATC A → NGW 1

```
Получение сообщения IAM.
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National
```

2. INVITE NGW 1 → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
```

```
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

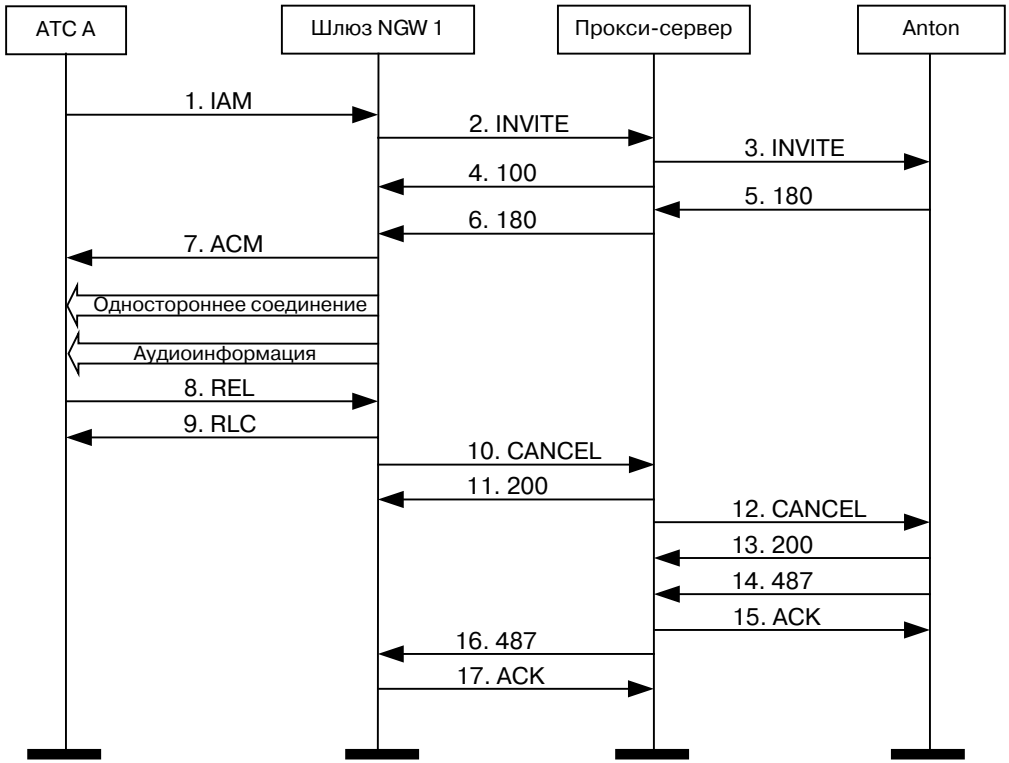


Рис. 12.17. Неуспешное установление соединения. Вызывающий абонент кладет трубку, не дождавшись установления соединения

3. INVITE Proxy 1 → Anton

```
INVITE sip:anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying Proxy 1 → NGW 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. 180 Ringing Anton → Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru;transport=tcp>
Content-Length: 0
```

6. 180 Ringing Proxy 1 → NGW 1

```
SIP/2.0 180 Ringing
```

```
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.niits.ru;lr>
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.niits.ru>
Content-Length: 0
```

7. ACM NGW 1 → ATC A

Передается сообщение ACM.
Maxim кладет трубку.

8. REL ATC A → NGW 1

Передается сообщение REL с CauseCode=16 Normal.

9. RLC NGW 1 → ATC A

Передается сообщение RLC.

10. CANCEL NGW 1 → Proxy 1

```
CANCEL sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

11. 200 OK Proxy 1 → NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

12 CANCEL Proxy 1 → Anton

```
CANCEL sip:anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

13. 200 OK Anton → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

14. 487 (Request Terminated) Anton → Proxy 1

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

15. ACK Proxy 1 → Anton

```
ACK sip:anton@b.niits.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

16. 487 Request Terminated Proxy 1 → NGW 1

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

17. ACK NGW 1 → Proxy 1

```
ACK sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.niits.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.niits.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

Глава 13. Преобразование SIP–ISUP

13.1. Процесс обмена сообщениями

Следующие примеры обмена сообщениями иллюстрируют типичные случаи установления соединения в сети SIP. Для простоты на диаграммах не показан ответ 100 (Trying) на запрос INVITE, хотя в большинстве сценариев он применяется. Как это уже отмечалось в предыдущей главе, все сигнальные сообщения (SIP и ISUP) проходят через MGC, а управление медиапотоком (например, подача аудиоданных пользователю, освобождение канала и т.д.) выполняется MG под управлением MGC. Так же, как это было сделано в предыдущей главе, на рисунках, представленных ниже, для упрощения MGC и MG представлены одним узлом, обозначенным MGC/MG.

13.1.1. Установление соединения (быстрый ответ не используется)

На рис. 13.1 представлен сценарий установления успешного соединения в SIP-сети к абоненту ТфОП без процедуры «быстрого ответа».

1. Когда пользователь SIP хочет вызвать абонента ТфОП, терминал SIP передает запрос INVITE.
2. Шлюз получает запрос, переданный терминалом SIP, преобразует его в сообщение IAM для сети ТфОП и передает это сообщение.
3. Когда узел ТфОП получает необходимую адресную информацию, он начинает установление соединения с абонентом, а к шлюзу передается сообщение ACM.

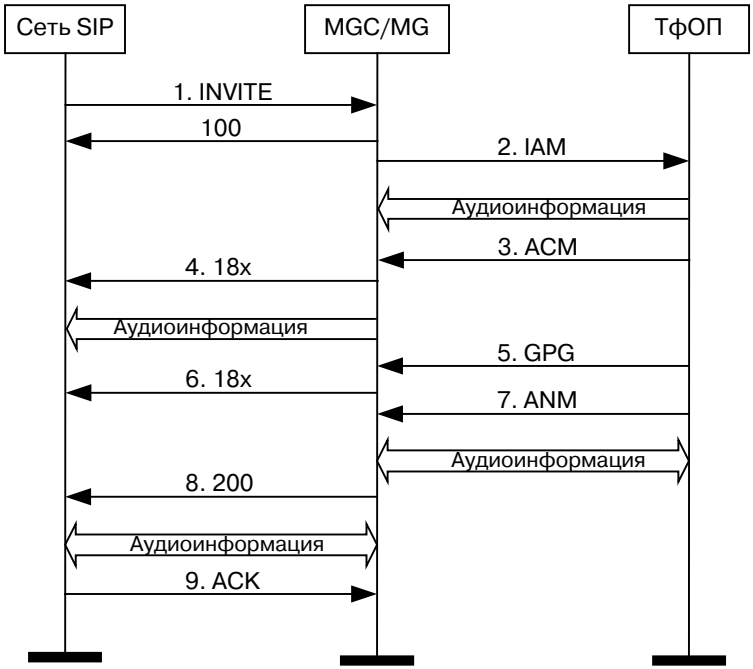


Рис. 13.1. Установление соединения (быстрый ответ не используется)

4. Параметр called party status (состояние вызываемой стороны) в сообщении ACM преобразуется в один из предварительных ответов SIP, который передается терминалу пользователя сети. Этот ответ может содержать вложение SDP для однонаправленного медиапотока (как показано на диаграмме). Если вложение SDP не содержится, то после шага 8 проключается тракт для двухстороннего медиапотока.
5. Если исполь зуемый вариант ISUP это позволяет, узел ТфОП может передавать сообщения для индикации особенностей маршрута в ТфОП.
6. Если сообщение пришло на шлюз, тот преобразует его в один из предварительных ответов и передает этот ответ к терминалу пользователя SIP.
7. Как только абонент ТфОП ответит на вызов (поднимет трубку), к шлюзу будет передано сообщение ANM.

8. После того, как будет получено сообщение ANM, шлюз передает пользователю SIP ответ 200 OK.
9. В ответ на это терминал пользователя SIP передает сообщение ACK, подтверждая все параметры и успешное установление соединения.

13.1.2. Установление соединения (быстрый ответ)

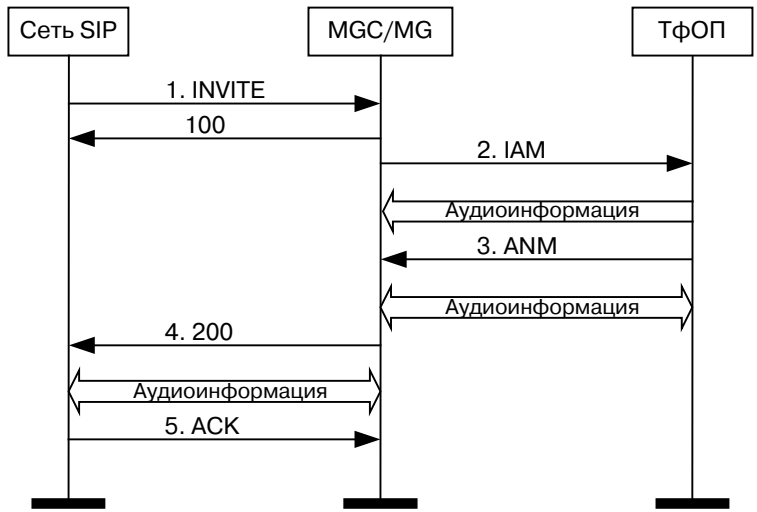


Рис. 13.2. Установление соединения (используется быстрый ответ)

Отметим, что этот сценарий предусмотрен в европейской версии ISUP, но не поддерживается в версии ISUP ANSI. На рис.13.2 показаны следующие шаги:

1. Когда пользователь SIP хочет получить соединение с абонентом ТфОП, терминал передает запрос INVITE.
2. Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое передается в ТфОП.
3. Поскольку удаленный терминал ТфОП предусматривает автоответ, узел ТфОП в ответ на пришедшее сообщение IAM передает сообщение ANM .

4. После получения сообщения ANM шлюз передает к терминалу SIP ответ 200.
5. После получения этого ответа терминал SIP считает соединение установленным и, подтверждая все параметры, передает сообщение ACK.

13.1.3. Срабатывание таймера T7

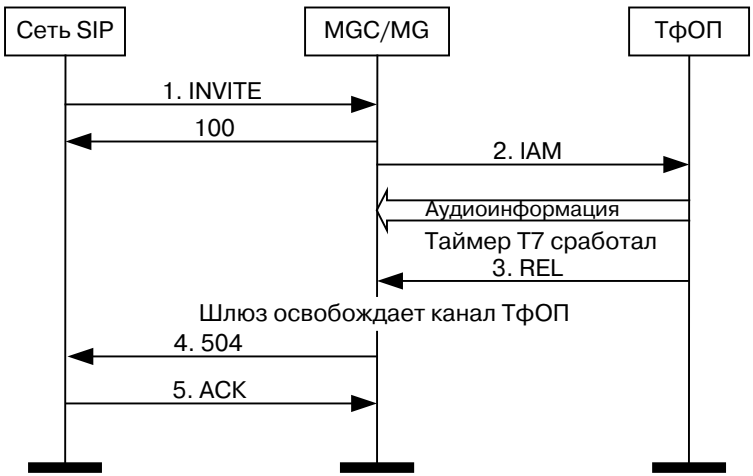


Рис. 13.3. Срабатывание таймера T7

1. Когда пользователь SIP хочет установить соединение с абонентом ТфОП, его терминал передает запрос INVITE.
2. Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое передается в ТфОП. С этого момента таймер T7 начинает отсчет.
3. Таймер срабатывает прежде, чем приходит сообщение ACM или ANM, и поэтому к шлюзу передается REL.
4. От шлюза к терминалу SIP передается ответ 504 (Server Time-out) с кодом ошибки – срабатывание таймера.
5. Получив ответ, терминал SIP передает подтверждение приема ACK.

13.1.4. Срабатывание таймеров SIP

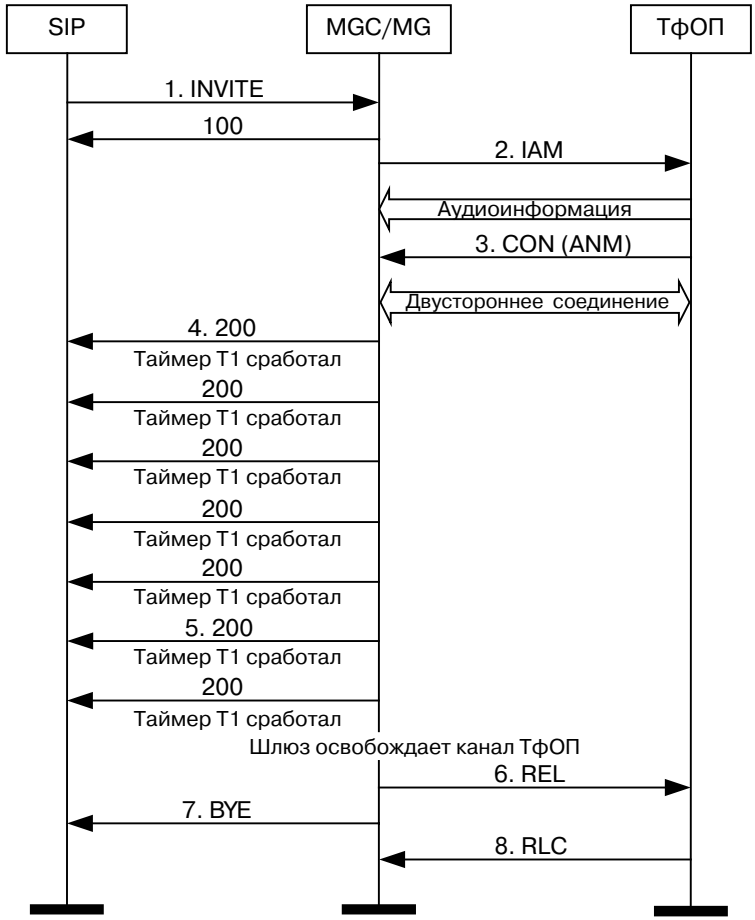


Рис. 13.4. Срабатывание таймеров SIP

1. Когда пользователь SIP хочет установить соединение с абонентом ТфОП, терминал передает запрос INVITE.

2. Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое передается в ТфОП.
3. Поскольку терминал ТфОП предусматривает быстрый ответ, он передает сообщение CON, как только получит IAM. В варианте ISUP ANSI вместо CON передается ANM (без ACM).
4. После получения сообщения CON (ANM) шлюз передает терминалу SIP ответ 200 OK, и таймер T1начинает отсчет.
5. Ответ передается повторно всякий раз, когда срабатывает таймер.
6. После семи повторений установление соединения завершается, к узлу ТфОП передается сообщение REL с кодом причины 102 (recover on timer expiry).
7. Терминалу SIP передается сообщение BYE для завершения соединения. Дальнейший процесс завершения соединения не показан, поскольку неизвестно состояние терминала SIP.
8. На сообщение REL узел ТфОП отвечает сообщением RLC.

13.1.5. Ошибка установления соединения на стороне ТфОП

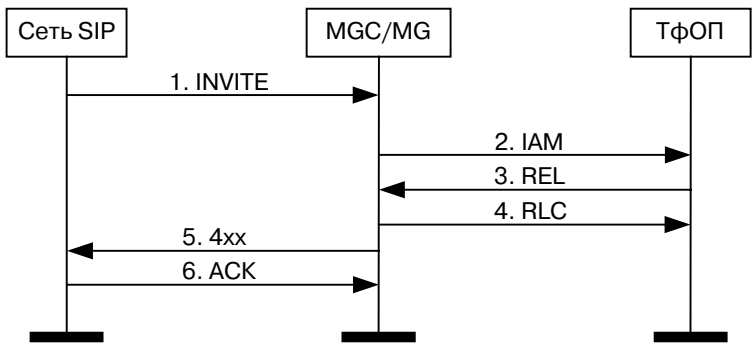


Рис. 13.5. Ошибка установления соединения на стороне ТфОП

1. Когда пользователь SIP хочет начать соединение с абонентом ТфОП, терминал передает запрос INVITE.

2. Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое передается в ТфОП.
3. Поскольку узел ТфОП не может установить соединение, он передает к шлюзу сообщение REL.
4. Шлюз подтверждает отмену установления соединения сообщением RLC и освобождает занятые ресурсы.
5. Терминалу SIP передается ответ с кодом ошибки, соответствующей коду причины в сообщении REL.
6. Терминал SIP подтверждает завершение процедуры установления соединения сообщением ACK.

13.1.6. В сообщении ACM содержится код причины

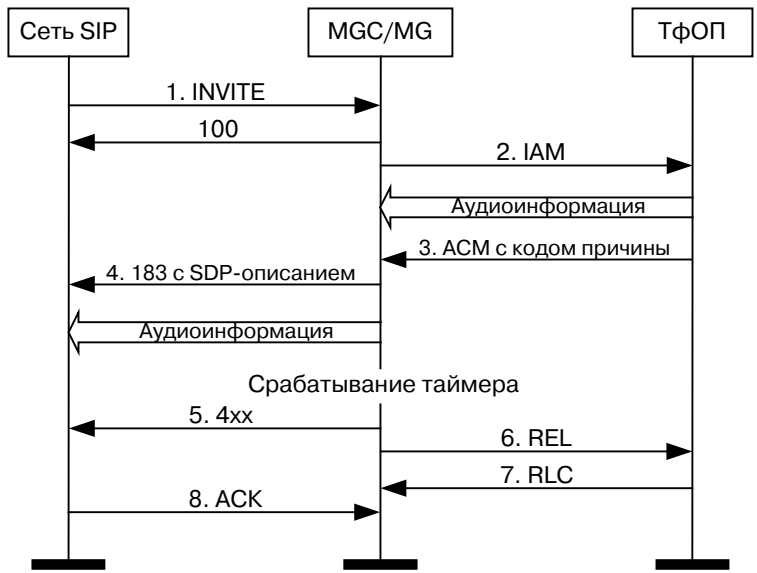


Рис. 13.6. В сообщении ACM содержится код причины

1. Когда пользователь SIP хочет получать соединение с абонентом ТфОП, терминал передает запрос INVITE.
2. Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое отправляется в ТфОП.
3. Поскольку узел ТфОП не может установить соединение (например, абонент находится вне зоны действия сети), он передает шлюзу сообщение ACM с соответствующим кодом причины. Шлюз запускает таймер.
4. После получения ACM с кодом причины (содержащимся в параметре CAI) шлюз передает к терминалу SIP ответ 183, содержащий вложение SDP для создания на непродолжительное время одностороннего аудиоканала.
5. Генерируется основанный на коде причины заключительный ответ на INVITE и передается к терминалу SIP для прекращения установления соединения.
6. При срабатывании таймера узлу ТфОП передает сообщение REL для прерывания установления соединения. Однако если пользователь SIP прервет установление соединения раньше, чем сработает таймер (терминал передает сообщение CANCEL), то процесс обмена сообщениями будет соответствовать следующему сценарию.
7. На полученное сообщение REL узел ТфОП отвечает сообщением RLC.
8. Терминал SIP передает ACK, завершая процесс обмена сообщениями.

13.1.7. Пользователь SIP прерывает установление соединения

1. Когда пользователь SIP хочет установить соединение с абонентом ТфОП, терминал передает запрос INVITE.
2. Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое передается в ТфОП.
3. Когда узел ТфОП получит необходимую адресную информацию, он начнет процесс установления соединения с абонентом, а к шлюзу отправит сообщение ACM.
4. Параметр called party status (состояние вызываемой стороны) в сообщении ACM преобразуется в один из предварительных ответов SIP, который передается к терминалу пользователя сети. Этот ответ может содержать вложение SDP для установления однонаправленного мультимедийного соединения.

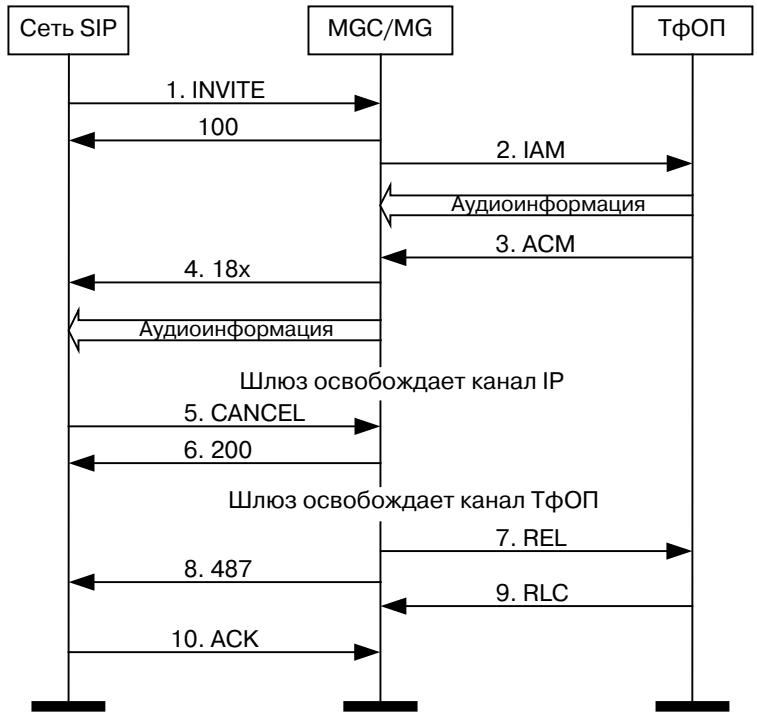


Рис. 13.7. Пользователь SIP прерывает установление соединения

5. Для разрушения установленного соединения до того, как абонент ТфОП ответит на вызов, терминал SIP передает сообщение CANCEL.
6. Это сообщение подтверждается ответом 200 OK.
7. После получения CANCEL шлюз передает к узлу ТфОП сообщение REL.
8. Шлюз передает сообщение 487 (Request Terminated) терминалу SIP, как сигнал о завершении обработки запроса INVITE.
9. На пришедшее сообщение REL узел ТфОП отвечает сообщением RLC.
10. Терминал SIP подтверждает прием ответа 487 (Call Cancelled) сообщением ACK.

13.2. Конечные автоматы SIP-T при взаимодействии SIP-ISUP

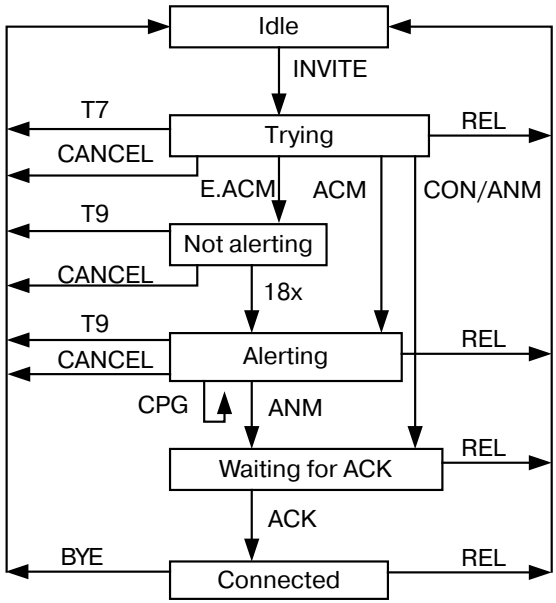


Рис. 13.8. Конечный автомат SIP-T-ISUP

13.2.1. Получение запроса INVITE

При получении запроса INVITE шлюз может передать к терминалу SIP ответ 100 (Trying) для индикации того, что вызов обслуживается.

При получении запроса шлюз должен также зарезервировать ресурсы для создания мультимедийного соединения, поскольку до этого сообщение IAM не может быть передано. Обычно ресурсами являются тайм-слот в первичном цифровом потоке E1/T1 и RTP/UDP порт на стороне IP. Для резервирования ресурсов могут использоваться любые процедуры обеспечения заданного качества обслуживания. После того как будет передано сообщение IAM, шлюз запускает таймер T7.

13.2.2. Процедуры преобразования INVITE в IAM

В сообщении IAM должны присутствовать пять обязательных параметров: Called Party Number (CPN), Nature of Connection Indicator (NCI), Forward Call Indicators (FCI), Calling Party's Category (CPC), и параметр, который указывает желаемые характеристики канала передачи – в некоторых вариантах ISUP это задается параметром Transmission Medium Requirement (TMR), в остальных вариантах – с помощью User Service Information (USI) или обоих параметров. Все сообщения IAM должны содержать, как минимум, пять названных параметров.

Таким образом, любой шлюз должен преобразовывать заголовки пришедшего запроса INVITE в значения этих параметров. Значения большинства параметров (таких как NCI и USI) шлюз создает самостоятельно, исходя из параметров устанавливаемого соединения. Значение остальных, таких как CPN, получаются на основе информации, пришедшей в запросе INVITE.

В сообщении IAM может также содержаться ряд необязательных параметров. При переводе содержания сообщений из SIP в ISUP их рекомендуется использовать, но можно и не учитывать. Как было отмечено ранее, преобразование позволяет аппаратуре сети SIP понимать содержимое сообщений узлов ТфОП, когда инкапсулированное сообщение ISUP отсутствует или его невозможно извлечь. Параметры, которые важны только для ТфОП при сценарии ТфОП-SIP-ТфОП, инкапсулируются в тело запросов SIP; преобразовывать их значения не обязательно. Из всех необязательных параметров преобразованы могут быть следующие: Calling Party's Number (CIN, он обычно присутствует), Transit Network Selection (TNS), Carrier Identification Parameter (CIP, только для сети ANSI), Original Called Number (OCN), и Generic Digits (или, в некоторых вариантах, Generic Address Parameter (GAP)).

Когда на шлюз приходит сообщение INVITE, он должен использовать для формирования IAM вложенное сообщение ISUP, если оно существует. Если это возможно, то при создании нового IAM шлюз должен попытаться использовать значения параметров из вложенного сообщения. В некоторых случаях шлюз не может использовать инкапсулированное сообщение – например, не может обработать данный вариант вложенного ISUP-сообщения и поэтому вынужден его игнорировать. Даже если использование инкапсулированного сообщения возможно, при формировании нового сообщения IAM значения параметров, полученные из заголовков сообщения SIP, заменяют значения параметров, полученные из инкапсулированного сообщения. Другими словами, в сценарии ISUP-SIP-ISUP значения

параметров в заголовке сообщения SIP имеют приоритет перед значениями параметров инкапсулированного сообщения. Это позволяет вводить множество услуг в сети SIP, в частности, таких как разные варианты переадресации вызовов.

Например, пришедшее на шлюз сообщение INVITE содержит инкапсулированное IAM с параметром CPN, содержащим телефонный номер +78125332699, а поле Request-URI сообщения INVITE содержит «tel:+70955550110». Тогда при создании нового сообщения IAM для ТфОП шлюз должен использовать телефонный номер, содержащийся в Request-URI, а не тот, который находится в поле CPN вложенного IAM. Более подробное описание преобразования заголовков SIP в параметры ISUP описано ниже.

Шлюз должен иметь установленные заранее значения по умолчанию для обязательных параметров ISUP и использовать их в тех случаях, когда значение параметра не может быть получено путем преобразования сообщения SIP (например, таких параметров, как NCI или TMR), а инкапсулированное сообщение ISUP отсутствует. Параметр FCI также должен иметь заранее введенное значение по умолчанию, только бит «M» может быть заменен в процессе преобразования, если используются механизмы переносимости номера, описание которых следует ниже.

Первым шагом преобразования полей и заголовков сообщения INVITE в параметры IAM является проверка поля Request-URI. Если шлюзом поддерживается услуга переноса номера, то следующие шаги связаны с обработкой параметров «npdi» и «rn» поля Request-URI. Подробнее процедуры услуг переноса номера описаны в уже упоминавшемся документе RFC 3482 «Number Portability in the Global Switched Telephone Network (GSTN): An Overview» [15].

Если внутри поля Request-URI не существует записи «npdi=yes», то основной телефонный номер в tel URL (цифры номера следуют непосредственно за «tel:») должен быть преобразован в формат ISUP и занесен в значение параметра CPN.

Если запись «npdi=yes» существует, тогда бит «number translated» параметра FCI, содержащегося в IAM, должен отразить то, что была использована услуга переноса номера.

Если в дополнение к записи «npdi=yes» не существует поля «rn», тогда основной телефонный номер в tel URL должен быть преобразован в формат ISUP и занесен в значение параметра CPN. Это означает, что переадресация

была произведена, но номер вызываемого абонента не изменился. Если существуют оба параметра «npdi=yes» и «rn», то телефонный номер из параметра «rn» должен быть преобразован в формат ISUP и занесен в значение параметра CPN. Основной телефонный номер в tel URL тоже должен быть преобразован в формат ISUP и занесен в значение параметра Generic Digits Parameter (или GAP в варианте ANSI ISUP). В некоторых вариантах ISUP номер, получаемый из параметра «rn», присоединяется к основному телефонному номеру (с префиксом или без, или с сепаратором), и комбинированный результат используется как значение параметра CPN.

Все дальнейшие обязательные при преобразовании действия выполняются после проверки использования услуги переносимости номера и разбора параметров, связанных с этой функцией. Если переносимость номера не поддерживается шлюзом, то основной телефонный номер из tel URL должен быть преобразован в формат ISUP и после этого использован в качестве значения параметра CPN.

Если основной телефонный номер в поле Request-URI и значение заголовка **To** не совпадают, то заголовок **To** должен быть использован для заполнения параметра OCN. В остальных случаях заголовок **To** игнорируется.

Встроенные функции протокола SIP-T позволяют использовать маршрутизацию по идентификатору сети (оператора). Поддержке такого типа маршрутизации посвящен документ RFC 3398 [6].

Когда вызов из сети SIP маршрутизируется шлюзом, то поле Request-URI должно содержать tel URL (или SIP URI с пользовательской частью адреса в виде tel URL). Если шлюз принимает сообщение, в поле Request-URI которого не содержится правильного телефонного номера, то такое сообщение должно быть отклонено, а отправителю передан ответ с кодом ошибки.

Исключением из этого правила является случай, когда поле **From** не содержит телефонного номера. Такое может быть, если вызов производился с телефона в сети SIP, и когда адрес отправителя обычно имеет вид user@host. Тогда при создании нового сообщения IAM параметр CIN должен быть опущен. Шлюз может также использовать для преобразования SIP URI в телефонный номер альтернативные не стандартизированные процедуры.

Когда шлюз получает запрос с вложенным в него проверенным и доступным для обработки сообщением ISUP, то он должен установить в генерируемом IAM

индикатор FCI в соответствии с тем, какие значения имеют во вложенном сообщении ISUP все биты, связанные с взаимодействием. В большинстве случаев встречается значение индикатора «no interworking».

Если в запросе, пришедшем на шлюз, нет доступного для обработки вложенного сообщения ISUP, при создании нового IAM рекомендуется присваивать биту Interworking Indicator параметра FCI значение «no interworking», а индикатору ISDN User Part Indicator – в значение «ISUP used all the way»; шлюз может также присвоить параметру Originating Access Indicator значение «Originating access non-ISDN».

Когда в параметре FCI установлено значение «interworking encountered», это показывает, что сеть ISDN взаимодействует с сетью, которая не поддерживает большинство из услуг, существующих в сетях ISDN. Поэтому сеть ISDN вынужденно ограничивает свои возможности, которые она обычно использует, включая использование кодов причины [cause code] при неудачном установлении соединения.

Если необходимо, производители могут снабдить шлюз настраиваемыми опциями, используемыми, по их усмотрению, провайдерами услуг в том случае, когда в параметре FCI указано взаимодействие, а вложенное сообщение отсутствует или недоступно.

13.2.3. Срабатывание таймера ISUP T7

Так как узел ТфОП не отвечает в течение установленного времени, MG освобождает свои ресурсы. Терминалу SIP передается ответ 504 (Server Timeout). Узлу ТфОП передается сообщение REL с кодом причины 102 (ошибка протокола, основанная на срабатывании таймера). Шлюз может ожидать, что ТфОП ответит сообщением RLC, а сеть SIP – сообщением ACK, которые подтверждают освобождение ресурсов.

13.2.4. Получение сообщения CANCEL или BYE

Если запросы CANCEL и BYE были получены до того, как произошло установление соединения, в сеть SIP должен быть передан ответ 200 OK, подтверждающий прием и того, и другого; должен быть передан также ответ 478 для прекращения передачи запросов INVITE. Далее все ресурсы освобождаются, ик

узлу ТфОП передается сообщение REL с кодом причины 16 (normal clearing). Шлюз может ожидать сообщения RLC от узла ТфОП, подтверждающего освобождение ресурсов.

В сценарии, когда сеть SIP является транзитной, сообщение REL может быть инкапсулировано в тело запроса BYE. Хотя обычно запрос BYE преобразуется в код причины 16 (normal clearing), в некоторых случаях код причины в инкапсулированном REL может быть другим. Поэтому параметр Cause Indicator из вложенного сообщения должен быть использован в сообщении REL, передаваемом к узлу ТфОП.

Запросы CANCEL и BYE могут содержать заголовок **Reason**, значение которого должно преобразовываться в значение параметра Cause Indicator. Если BYE содержит и заголовок **Reason** и инкапсулированное сообщение ISUP, то значение заголовка **Reason** является более значащим. Шлюз должен освободить все ресурсы до того, как он передаст сообщение REL.

13.2.5. Получение сообщения REL

В данном разделе описано, что происходит, когда на шлюз приходит сообщение REL до того, как соединение было установлено. Обычно это происходит в тех случаях, когда узел ТфОП отклоняет пришедший на него вызов.

В этом случае все ресурсы шлюза должны быть немедленно освобождены, а в ТфОП передано сообщение RLC.

Если запрос INVITE, который инициировал сеанс связи, содержал проверенное и доступное для обработки сообщение ISUP, то соответствующее вложенное сообщение ISUP должно быть передано в ответе на этот INVITE. Следовательно, как только сообщение REL приходит на шлюз, его нужно включить в тело ответа SIP. Шлюз не должен передавать ответ с инкапсулированным сообщением ISUP, если инициатор создания сеанса не использовал вложенные сообщения ISUP в запросе INVITE.

Получение некоторых эксплуатационно-управляющих сообщений ISUP в ответе на IAM, таких как Blocking Message (BLO), Reset Message (RSC) или их эквивалентов, может привести к прекращению установления соединения на данном этапе.

13.2.6. Преобразование кодов причины ISDN в коды ответов сети SIP

Сообщение REL в сети ОКС7 выполняет множество функций и поэтому имеет громоздкую структуру, создающую трудности при его обработке. В то же время, SIP имеет ряд специальных инструментов, которые вместе играют ту же роль, что и REL, а именно – BYE, CANCEL и различные коды состояний/ответов. В сети ISDN сообщение REL может быть передано, чтобы прекратить установление соединения (BYE в сети SIP), чтобы отменить переданный ранее запрос, пока его обработка не завершена (CANCEL в сети SIP), или чтобы отклонить запрос соединения, который был недавно получен (различные типы ответов в сети SIP).

Не все коды событий ISUP нужно переводить в сообщения SIP, т.к. некоторые из них предназначаются только для интерфейса ISUP шлюза. Хорошим примером является код причины 44 (Request circuit or channel not available). Он означает, что CIC, которому было передано IAM, не в состоянии обработать новый запрос. Шлюз в этом случае должен переслать IAM на другой CIC, удерживая вызов на ожидании. Ясно, что в этом случае нет и не может быть соответствующего кода состояния SIP, т.к. такая ситуация может возникнуть только в шлюзе.

Следовательно, при получении кода причины ISUP 44 (Request circuit or channel not available) преобразование в SIP производиться не будет.

Если значение кода события отличается от приведенных ниже, то используется ответ по умолчанию 500 (Server internal error).

В дополнение к коду причины, в ISUP используется параметр CAI, содержащий поле location, которое показывает, в какой сети произошло прерывание обслуживания вызова. В большинстве случаев поле location не влияет на преобразование в коды состояния SIP; некоторые исключения отмечены ниже. В кодах причины ISUP может также присутствовать поле диагностики, которое содержит дополнительную информацию, имеющую отношение к прерыванию обслуживания вызова.

Сообщение REL с кодом причины ISUP 22 (number changed) может содержать информацию о новом номере вызываемого абонента в поле diagnostics.

Если шлюз способен обработать эту информацию, то этот номер должен быть добавлен в заголовок **Contact** ответа SIP (301).

Рекомендуются представленные в табл.13.1 варианты преобразования.

Таблица 13.1

Коды причины ISUP	Ответы SIP
1 unallocated number	404 Not Found
2 no route to network	404 Not found
3 no route to destination	404 Not found
16 normal call clearing	– (*)
17 user busy	486 Busy here
18 no user responding	408 Request Timeout
19 no answer from the user	480 Temporarily unavailable
20 subscriber absent	480 Temporarily unavailable
21 call rejected	403 Forbidden (+)
22 number changed (w/o diagnostic)	410 Gone
22 number changed (w/ diagnostic)	301 Moved Permanently
23 redirection to new destination	410 Gone
26 non-selected user clearing	404 Not Found (=)
27 destination out of order	502 Bad Gateway
28 address incomplete	484 Address incomplete
29 facility rejected	501 Not implemented
31 normal unspecified	480 Temporarily unavailable

(*) – Код причины ISUP 16 (normal call clearing) обычно преобразуется в запросы BYE или CANCEL.

(+) – Если значением поля location является «user», то вместо ответа класса 4xx передается ответ класса 6xx (вместо ответа 403 – ответ 603).

(=) – Процедура ANSI – в сети ANSI коду причины 26 присваивается значение misrouted port number. Предполагается, что в основной сети была использована услуга переноса номера. В остальных случаях код причины 26 процедурами ISUP обычно не используется.

Вид причины «ресурсы недоступны» указывает на временный отказ в обслуживании, так как все ресурсы в данный момент задействованы (табл. 13.2). Поэтому при преобразовании к ответу SIP может быть добавлен заголовок **«Retry After»**. Значение причины «услуга или опция недоступны» указывает, что имеются временные проблемы при обработке запроса, которые оборудование самостоятельно устранит через какое-то время. Эти и другие причины приведены в табл. 13.2. – 13.6.

Таблица 13.2. Ресурсы недоступны

Код причины ISUP	Ответ SIP
34 no circuit available	503 Service unavailable
38 network out of order	503 Service unavailable
41 temporary failure	503 Service unavailable
42 switching equipment congestion	503 Service unavailable
47 resource unavailable	503 Service unavailable

Таблица 13.3. Услуга или опция недоступны

Код причины ISUP	Ответ SIP
55 incoming calls barred within CUG	403 Forbidden
57 bearer capability not authorized	403 Forbidden
58 bearer capability not presently available	503 Service unavailable
65 bearer capability not implemented	488 Not Acceptable Here
70 only restricted digital availability	488 Not Acceptable Here
79 service or option not implemented	501 Not implemented

Таблица 13.4. Неверное сообщение

Код причины ISUP	Ответ SIP
87 user not member of CUG	403 Forbidden
88 incompatible destination	503 Service unavailable

Таблица 13.5. Ошибка протокола

Код причины ISUP	Ответ SIP
102 recovery of timer expiry	504 Gateway timeout
111 protocol error	500 Server internal error

Таблица 13.6. Взаимодействие с другими сетям

Код причины ISUP	Ответ SIP
127 interworking unspecified	500 Server internal error

13.2.7. Получение предварительного сообщения ACM

Сообщение ACM передается в тех случаях, когда необходимо показать, что вызов еще обрабатывается, и перезапустить таймеры ISUP: при этом процесс обслуживания вызова не попадает в стадию alerting. Такой метод применяется, например, в мобильных сетях, где перемещение пользователя может создавать большие задержки при установлении соединения. Предварительное ACM передается перед тем, как процесс обслуживания перейдет в состояние Alerting, для того, чтобы сбросить таймер T7 и запустить таймер T9. Сообщение ACM называется «предварительным ACM» в том случае, если значение параметра Called Party's Status Indicator равно 00 (no indication).

После того как было передано предварительное ACM, сеть ISDN ожидает сообщения CPG, чтобы перевести процесс в следующую стадию.

Когда шлюз получает предварительное ACM, он должен передать ответ 183 (Session Progress) в сеть SIP. В сценарии, когда сеть SIP является транзитной, предварительное ACM должно быть также включено в тело ответа.

Передача ответа 183 (Session Progress) до того, как было получено сообщение о достаточности адресной информации (ACM), создает проблемы при транзите трафика через сеть SIP (SIP – bridging), и, следовательно, не должна производиться.

13.2.8. Получение сообщения ACM

В большинстве случаев при получении сообщения ACM шлюз должен передать предварительный ответ (из группы 18x) в сеть SIP. Если запрос INVITE, полученный от инициатора сеанса, содержал проверенное и доступное для обработки инкапсулированное сообщение ISUP, ACM, полученное шлюзом, должно быть присоединено к предварительному ответу, передаваемому в сеть SIP.

Если ACM содержит параметр Backward Call Indicators со значением «subscriber free», шлюз должен передать ответ 180 (Ringing). Если до этого ответа разговорный тракт не проключался, все акустические сигналы передаются агентом пользователя (SIP user agent) в терминале. Шлюз тоже может выдавать акустические сигналы.

Если параметр Backward Call Indicators (BCI) в сообщении ACM показывает, что в соединении встречается межсетевое взаимодействие (обычно в ACM

указывается взаимодействие с более простой сетью, которая не поддерживает сигнализацию по общему каналу), то может быть использована внутриканальная передача служебных сигналов о состоянии процесса («занято» и др). Кроме того, если это возможно, должен быть создан односторонний разговорный тракт в обратном направлении. Такой тракт должен быть создан обязательно, если значение параметра сообщения ACM Optional Backward Call Indicators показывает, что речевые объявления и служебные сигналы будет передаваться внутри основного канала. После того как будут созданы односторонние тракты, шлюз должен передать код ответа 183 (Session Progress) в сеть SIP.

При получении сообщения ACM коммутаторы в большинстве версий ISUP запускают таймер T9, значение которого обычно варьируется от 90 секунд до 3 минут. В случае, когда созданы односторонние разговорные тракты, и передаются речевые объявления или служебные сигналы в предответном состоянии, этот таймер определяет, сколько времени вызывающий пользователь может прослушивать передаваемую ему информацию (например, КПВ или речевые сообщения) от удаленного ISUP без начисления платы за соединение. КПВ и речевые сообщения генерирует ближайшая к пользователю АТС ТфОП. Если необходимо, чтобы речевые сообщения продолжались дольше, чем допускает таймер T9, сеть передает сообщение ANM, которое инициирует создание двухстороннего тракта на неопределенный промежуток времени. Обычно в сетях с ISUP процедуры биллинга начинают использоваться после получением сообщения ANM. Некоторые сети могут не поддерживать таймер T9.

13.2.9. Получение сообщений CON или ANM

Когда шлюз получает сообщение CON или ANM, это означает, что вызываемый абонент ответил на вызов, и в сеть SIP должен быть передан ответ 200 OK. В сценарии, когда сеть SIP используется как транзитная (в данном случае запрос INVITE инициатора сеанса содержит разрешенное и доступное вложение), пришедшее сообщение CON или ANM должно быть инкапсулировано в ответ 200 OK, передаваемый в сеть SIP. После этого должен быть установлен двухсторонний медиатракт, если этого не было сделано заранее.

Когда межсетевое взаимодействие ведется с сетями того же класса, что и сеть OKC7 с ISUP, то АТС ТфОП может получить ANM сразу же после получения предварительного ACM (без CPG или других подобных сообщений), или даже вовсе без получения ACM (когда используется быстрый ответ).

В этом случае пользователю SIP не будет передан предварительный ответ класса 18x, и он не услышит звуковых сигналов (КПВ и др.) перед тем, как вызываемый абонент снимет трубку. Это может привести к потере начального содержимого мультимедийного потока, так как передача потока не может начаться до того, как будут получены данные SDP.

13.2.10. Срабатывание таймера T9

Этот таймер может использоваться не во всех сетях. Его срабатывание показывает, что сообщение ANM для обслуживаемого вызова не было получено в течение длительного времени (с момента передачи ACM). Обычно это означает, что терминал вызываемого абонента получал сигнал ПВ длительное время, но не ответил. Это может происходить также в случае межсетевого взаимодействия, когда сеть передает информацию о статусе абонента (например, абонент находится вне зоны действия сети), повторяя ее несколько раз. Любая причина задержки обслуживания вызова в этом состоянии до срабатывания таймера T9 приводит к тому, что обработка запроса прекращается. Все ресурсы шлюза, связанные с данным мультимедийным потоком, освобождаются. В сеть SIP должен быть передан ответ 480 (Temporarily Unavailable), а в сеть ТфОП – сообщение REL с кодом причины 19 (no answer from the user). Шлюз ожидает сообщения RLC из сети ТфОП и запроса ACK из сети SIP в подтверждение освобождения ресурсов.

13.2.11. Получение сообщения CPG

CPG – это предварительное сообщение, передаваемое, в частности, в следующих случаях:

- передача информации об особенностях обслуживания вызова (например, с переадресацией в сети ISDN) ;
- передается сигнал КПВ;
- передаются речевые объявления.

Если CPG показывает, что идет передача речевых объявлений, шлюз должен создать односторонний аудиотракт в обратном направлении и начать передачу. При транзите трафика через сеть SIP, сообщение CPG должно быть инкапсулировано в ответ группы 18x, передаваемый в сеть SIP и определяемый по коду события CPG с помощью табл. 13.7.

Если сообщение CPG не показывает, что вызываемому абоненту подается ПВ, текущее состояние процесса обслуживания вызова не изменяется.

Таблица 13.7

Код события ISUP	SIP response
1 Alerting	180 Ringing
2 Progress	183 Session progress
3 In-band information	183 Session progress
4 Call forward; line busy	181 Call is being forwarded
5 Call forward; no reply	181 Call is being forwarded
6 Call forward; unconditional	181 Call is being forwarded
Нет кода события	183 Session progress

13.2.12. Получение ACK

В этом состоянии соединение уже полностью установлено, и имеет место разговорная фаза. При получении этого запроса в ТфОП ничего не передается.

13.3. Примеры сценариев и сообщений для вызовов SIP–ТфОП

В следующих сценариях пользователь Maxim (*sip:max@niits.ru*) пользуется SIP-телефоном или любым другим устройством, подключенным к сети SIP. Абонент Anton подключен к ТфОП и имеет международный телефонный номер +78122625326. Вызов пользователя Maxim проходит через прокси-сервер Proxy1 и шлюз Network Gateway. В некоторых сценариях Maxim звонит абоненту Alexey, который подключен к УПАТС и имеет 2 номера: внутренний номер 444 – 3333 [private extension] и международный номер +7-812-100-2516. Заметьте, что Maxim использует в заголовке **From** запроса INVITE международный номер +7-812-262-5326. Шлюз использует этот номер при определении данных для занесения в параметр calling party number сообщения ISUP.

В сценариях с ошибками соединение не устанавливается. Однако в некоторых случаях тракты для медиапотоков все же проключаются. Это бывает в тех случаях, когда ТфОП передает акустические сигналы о состоянии абонента («занято»

и различные объявления, например «Номер, по которому вы звоните, изменен. Новый номер…»). Это требуется для того, чтобы пользователь мог понять, по какой причине завершился его вызов. Такие тракты устанавливаются с использованием ответа 183 (Session Progress), который содержит в себе данные SDP.

Передача служебных объявлений может быть завершена пользователем, или шлюзом после срабатывания таймера. В остальных сценариях коды причины ISUP преобразуются в ответы SIP. В этих сценариях медиапотоки не передаются, но коды ошибок доводятся до пользователя с помощью SIP User Agent Client.

13.3.1. Успешное соединение из сети SIP в ТфОП

Пользователь Maxim набирает международный номер +78122625326 стандарта E.164 для того, чтобы позвонить абоненту Anton. Особое значение имеют последние 7 цифр номера (или меньшее число, в зависимости от нумерации в местной сети). Предполагается, что SIP UA может преобразовать набранный номер в международный формат и поместить его в SIP URI. Вместо SIP URI может использоваться tel URI.

Maxim может указать в заголовке **From** свой SIP-адрес (*sip:max@niits.ru*) или телефонный номер SIP (*sip:+70953864515@ss1.niits.ru;user=phone*). В данном примере используется телефонный номер, этот номер будет использован также при трансляции на NGW 1 как номер вызывающей стороны (calling party identification). Чтобы этот номер прошел в сеть с ОКС7, его корректность обязательно должна быть проверена и подтверждена.

В этом сценарии Anton отвечает на вызов, а после разговора первым кладет трубку пользователь Maxim.

В данном сценарии в сообщениях 7 – 11, передаваемых шлюзом, в поле **Contact** записано также sip:ngw1@a.niits.ru. Это сделано потому, что NGW 1 принимает только запросы, пришедшие от Proxy1; остальные сигнальные сообщения игнорируются. Так как этот Contact URI может использоваться не только в данном диалоге, то и за пределами этого сценария Contact URI для NGW 1 должен назначать Proxy1. Этот Contact URI назначается через DNS для Proxy1 (*sip:ss1.a.niits.ru*), который потом преобразует его в *sip:ngw1.a.niits.ru*, что и является адресом NGW1.

Для транспортировки сообщений используется TCP.

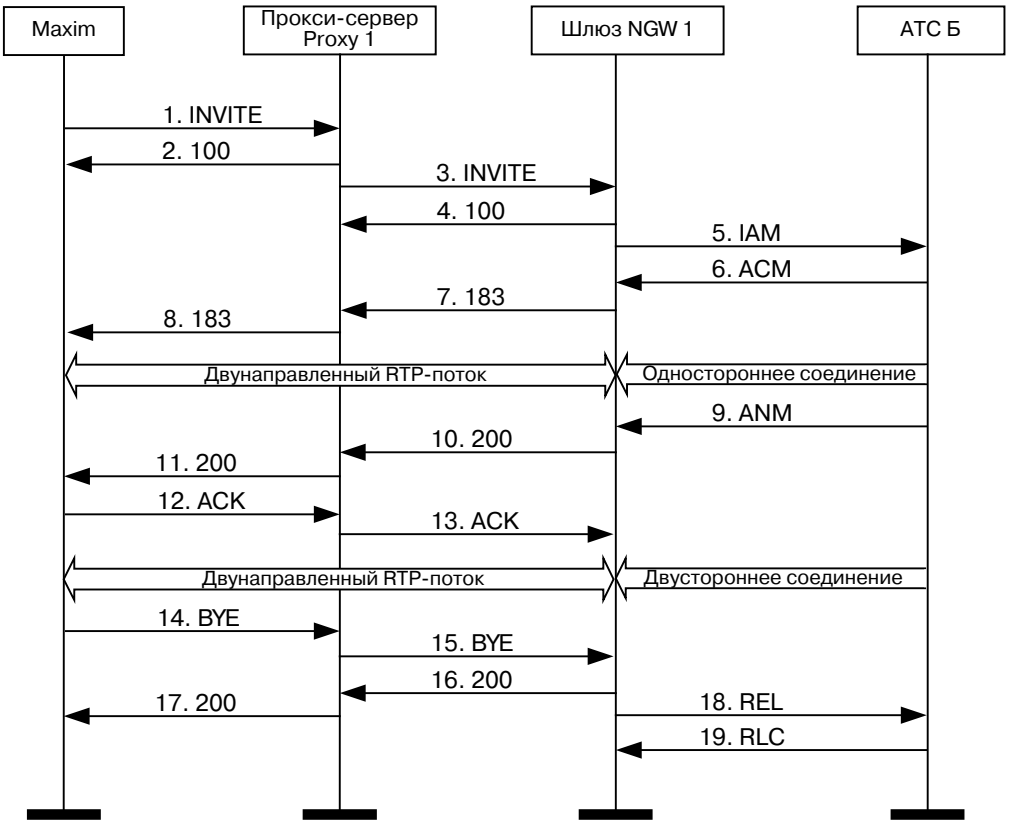


Рис. 13.9. Успешное установление соединения из сети SIP в ТфОП

Содержимое сообщений:

1. INVITE Maxim → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
```

```
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru;transport=tcp>
Proxy-Authorization: Digest username=«Max», realm=«a.niits.ru»,
nonce= «dc3a5ab25302aa931904ba7d88fa1cf5», opaque=«»,
uri=«sip:+78122625326@ss1.a.niits.ru;user=phone»,
response=«ccdca50cb091d587421457305d097458c»
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

2. 100 Trying Proxy 1 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

Proxy 1 использует функцию определения местоположения для поиска адреса шлюза, к которому нужно отправить вызов. Вызов направляется к шлюзу NGW1. Клиент, установленный на терминале пользователя Maxim, готовится принимать данные из сети на порт 49172.

3. INVITE Proxy 1 → NGW 1

```
INVITE sip:+78122625326@ngw1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 154
```

```
v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying NGW 1 → Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. IAM NGW 1 → АТС Б

```
Передается сообщение IAM
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National
```

6. ACM АТС Б → NGW 1

```
Передается сообщение ACM.
```

7. 183 Session Progress NGW 1 → Proxy 1

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
```

```
a=rtpmap:0 PCMU/8000
```

NGW 1 передает аудиоданные из ТфОП (КПВ) по каналу RTP Maxim.

8. 183 Session Progress Proxy 1 → Maxim

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

9. ANM АТС Б → NGW 1

Передается сообщение ANM.

10. 200 OK NGW 1 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
```

```
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

11. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

12. ACK Maxim → Proxy 1

```
ACK sip:ngw1@a.niits.ru SIP/2.0
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

13. ACK Proxy 1 → NGW 1

```
ACK sip:ngw1@a.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
```

```
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

Maxim кладет трубку, завершая разговор с пользователем Anton.

14. BYE Maxim → Proxy 1

```
BYE sip:ngw1@a.niits.ru SIP/2.0
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

15. BYE Proxy 1 → NGW 1

```
BYE sip:ngw1@a.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

16. 200 OK NGW 1 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

17. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
```

```
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

18. REL NGW 1 → АТС Б

Передается сообщение REL с CauseCode=16 Normal.

19. RLC АТС Б → NGW 1

Передается сообщение RLC.

13.3.2. Успешное соединение из сети SIP к абоненту УПАТС

Пользователь Maxim находится в сети SIP, а абонент Alexey подключен к УПАТС. Между шлюзом и УПАТС используется сигнализация DSS1. УПАТС подключается через группу каналов ISDN. Maxim набирает международный номер Alexey (+7-812-387-5333), и он помещается в SIP URI.

Часть адреса Request URI запроса INVITE F3, идентифицирующая узел в сети, используется для определения подключаемого контекста (пользователь, направление или линия), для которого доступен номер 444-3333. Иначе запрос INVITE F3 при обработке на шлюзе будет отправлен по другому адресу.

Proxy 1 на основе телефонного номера определяет шлюз, к которому подключена УПАТС. Терминал абонента Alexey определяется номером 444-3333, который находится в Request URI запроса, отправляемого шлюзу.

Заметьте, что Contact URI для GW 1, используемый в сообщениях 8, 9, 12 и 13 – sips:4443333@gw1.a.niits.ru, который переадресует все вызовы прямо на сервер.

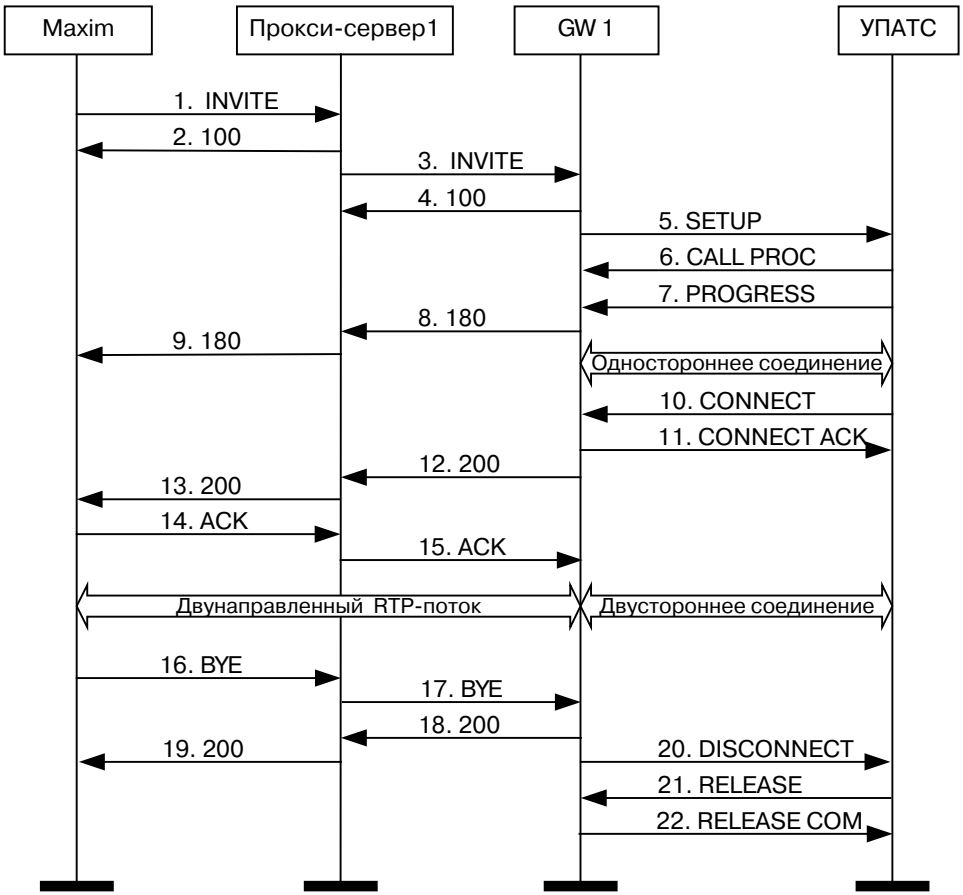


Рис. 13.10. Успешное установление соединения из сети SIP к абоненту УПАТС

В данном сценарии используется SIPS URI.

Содержимое сообщений:

1. INVITE Maxim → Proxy 1

```
INVITE sips:+78123875333@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sips:max@client.a.niits.ru>
Proxy-Authorization: Digest username=«Max»,
realm=«a.niits.ru», nonce= «qo0dc3a5ab22aa931904badfa1cf5j9h»,
opaque=«», uri= «sips:+78123875333@ss1.a.niits.ru;user=phone»,
response= «6c792f5c9fa360358b93c7fb826bf550»
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

2. 100 Trying Proxy 1 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Content-Length: 0
```

3. INVITE Proxy 1 → GW 1

```
INVITE sips:4443333@gw1.a.niits.ru SIP/2.0
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
```

```
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sips:max@client.a.niits.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying GW 1 → Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
;received=192.0.2.111
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Content-Length: 0
```

5. SETUP GW 1 → УПАТС

```
Protocol discriminator=Q.931
Message type=SETUP
Bearer capability: Information transfer capability=0 (Speech) or 16
(3.1 kHz audio)
Channel identification=Preferred or exclusive B-channel
Progress indicator=1 (Call is not end-to-end ISDN;further call
progress information may be available inband)
Called party number:
Type of number unknown
Digits=444-3333
```

6. CALL PROCEEDING УПАТС → GW 1

```
Protocol discriminator=Q.931
Message type=CALL PROC
Channel identification=Exclusive B-channel
```

7. PROGRESS УПАТС → GW 1

```
Protocol discriminator=Q.931
Message type=PROG
Progress indicator=1
```

8. 180 Ringing GW 1 → Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
```

```
;received=192.0.2.101
Record-Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.niits.ru>
Content-Length: 0
```

9. 180 Ringing Proxy 1 → Maxim

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.niits.ru>
Content-Length: 0
```

10. CONNECT УПАТС → GW 1

```
Protocol discriminator=Q.931
Message type=CONN
```

11. CONNECT ACK GW 1 → УПАТС

```
Protocol discriminator=Q.931
Message type=CONN ACK
```

12. 200 OK GW 1 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.niits.ru>
Content-Type: application/sdp
Content-Length: 144
```

```
v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.niits.ru
s=-
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

13. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.niits.ru>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.niits.ru
s=-
c=IN IP4 gw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

14. ACK Maxim → Proxy 1

```
ACK sips:4443333@gw1.a.niits.ru SIP/2.0
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 ACK
Content-Length: 0
```

15. ACK Proxy 1 → GW 1

```
ACK sips:4443333@gw1.a.niits.ru SIP/2.0
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
```

```
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 ACK
Content-Length: 0
```

Далее Maxim кладет трубку, завершая разговор с пользователем Alexey.

16. BYE Maxim → Proxy 1

```
BYE sips:4443333@gw1.a.niits.ru SIP/2.0
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sips:ss1.a.niits.ru;lr>
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 3 BYE
Content-Length: 0
```

17. BYE Proxy 1 → GW 1

```
BYE sips:4443333@gw1.a.niits.ru SIP/2.0
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 3 BYE
Content-Length: 0
```

18. 200 OK GW 1 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS ss1.a.niits.ru:5061;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 3 BYE
Content-Length: 0
```

19. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS client.a.niits.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sips:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 3 BYE
Content-Length: 0
```

20. DISCONNECT GW 1 → УПАТС

```
Protocol discriminator=Q.931
Message type=DISC
Cause=16 (Normal clearing)
```

21. RELEASE УПАТС → GW 1

```
Protocol discriminator=Q.931
Message type=REL
```

22. RELEASE COMPLETE GW 1 → УПАТС

```
Protocol discriminator=Q.931
Message type=REL COM
```

13.3.3. Соединение из сети SIP в ТфОП в условиях перегрузки шлюза

Пользователь Maxim отправляет вызов абоненту Anton через Proxy 1. Proxy 1 перенаправляет вызов на шлюз NGW 1. Пусть данный шлюз в этот момент недоступен (загружен обработкой других вызовов) и отклоняет пришедший запрос, передавая ответ с кодом ошибки 503 (Service Unavailable). Получив этот ответ, Proxy 1 направляет вызов на Network Gateway NGW 2. Anton отвечает на вызов. Соединение завершается, когда Maxim кладет трубку. В данном сценарии для транспортировки сообщений используется протокол UDP.

Содержимое сообщений:

1. INVITE Maxim → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru>
```

```
Proxy-Authorization: Digest username= «Max»,
realm=«a.niits.ru», nonce=«b59311c3ba05b401cf80b2a2c5ac51b0»,
opaque=«», uri=«sip:+78122625326@ss1.a.niits.ru;user=phone»,
response= «ba6ab44923fa2614b28e3e3957789ab0»
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения местоположения, чтобы найти текущий адрес абонента Anton. Proxy 1 получает два варианта расположения абонента – основной NGW 1 и дополнительный NGW 2. Первый запрос передается на NGW 1.

2. INVITE Proxy 1 → NGW 1

```
INVITE sip:+78122625326@ngw1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

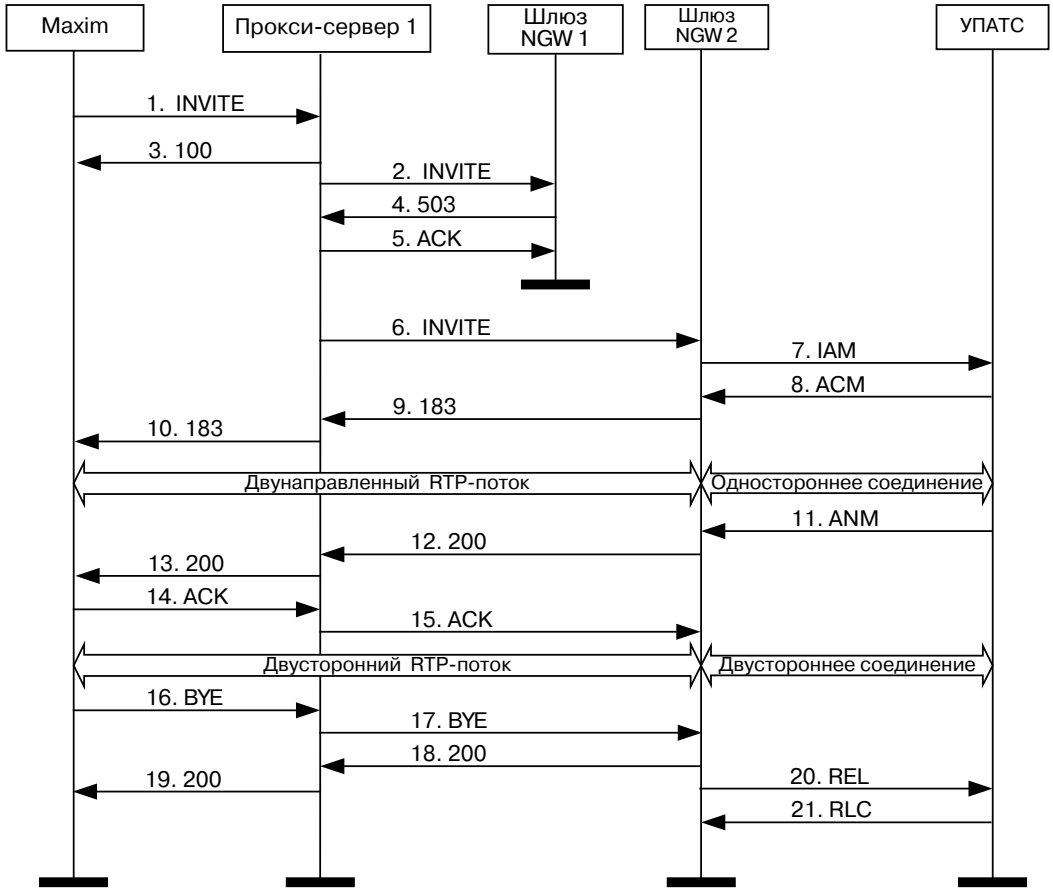


Рис. 13.11. Успешное установление соединения из сети SIP в сеть ТфОП в условиях перегрузки шлюза

3. 100 Trying Proxy 1 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

4. 503 (Service Unavailable) NGW 1 → Proxy 1

```
SIP/2.0 503 Service Unavailable
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=123456789
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. ACK Proxy 1 → NGW 1

ACK sip:ngw1@a.niits.ru SIP/2.0

Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1

```
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru>;user=phone>
;tag=123456789
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

Proxy 1 передает запрос на шлюз NGW 2.

6. INVITE Proxy 1 → NGW 2

```
INVITE sip:+78122625326@ngw2.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.2
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
```

```
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

7. IAM NGW 2 → АТС Б

Получение сообщения IAM.

```
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National
```

8. ACM АТС Б → NGW 2

Получение сообщения ACM.

9. 183 (Session Progress) NGW 2 → Proxy 1

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.2
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.niits.ru
s=-
c=IN IP4 ngw2.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Пользователю Maxim от шлюза передаются RTP пакеты с аудиоданными (КПВ).

10. 183 Session Progress Proxy 1 → Maxim

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.niits.ru
s=-
c=IN IP4 ngw2.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

11. ANM АТС Б → NGW 2

Получение сообщения ANM.

12. 200 OK NGW 2 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.2
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.niits.ru
s=-
c=IN IP4 ngw2.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

13. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.niits.ru
s=-
c=IN IP4 ngw2.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

14. ACK Maxim → Proxy 1

```
ACK sip:ngw2@a.niits.ru SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

15. ACK Proxy 1 → NGW 2

```
ACK sip:ngw2@a.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.2
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

Между пользователем Maxim и абонентом Anton проключается тракт для передачи RTP потока (через NGW). Через некоторое время Maxim кладет трубку, завершая разговор с абонентом Anton.

16. BYE Maxim → Proxy 1

```
BYE sip:ngw2@a.niits.ru SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

17. BYE Proxy 1 → NGW 2

```
BYE sip:ngw2@a.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.2
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

18. 200 OK NGW 2 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.2
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

19. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
```

```
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 BYE
Content-Length: 0
```

20. REL NGW 2 → АТС Б

Получение сообщения REL с CauseCode=16 Normal.

21. RLC АТС Б → NGW 2

Получение сообщения RLC.

13.3.4. Соединения из сети SIP в SIP с использованием ENUM Query

Пользователь Maxim пытается вызвать абонента Anton, набирая его телефонный номер (9722625326). UA пользователя Maxim преобразует введенный номер в формат номера E.164 (+78122625326) и выполняет процедуру ENUM с номером в формате E.164 (2.2.2.2.5.5.5.2.7.9.1.e164.arpa). Потом номер обрабатывается процедурой Naming Authority Pointer (NAPTR) на DNS сервере, и UA пользователя Maxim получает списочный адрес абонента Anton (*sip:+78122625326@b.niits.ru*). Если адрес имеет такой вид, это означает, что в данный момент пользователь Anton находится в сети SIP. Результатом всего этого является то, что UA пользователя Maxim передает запрос INVITE, и соединение устанавливается по IP, минуя ТфОП. Разговор заканчивается, когда терминал абонента Anton передает сообщение BYE.

Содержимое сообщений.

1. ENUM Query Maxim → DNS Server

```
2.2.2.2.5.5.5.2.7.9.1.e164.arpa
```

2. ENUM NAPTR Set DNS Server → Maxim

```
$ORIGIN 2.2.2.2.5.5.5.2.7.9.1.e164.arpa.
IN NAPTR 100 10 «u» «sip+E2U»
«!^.*$!sip:+78122625326@b.niits.ru!».
```

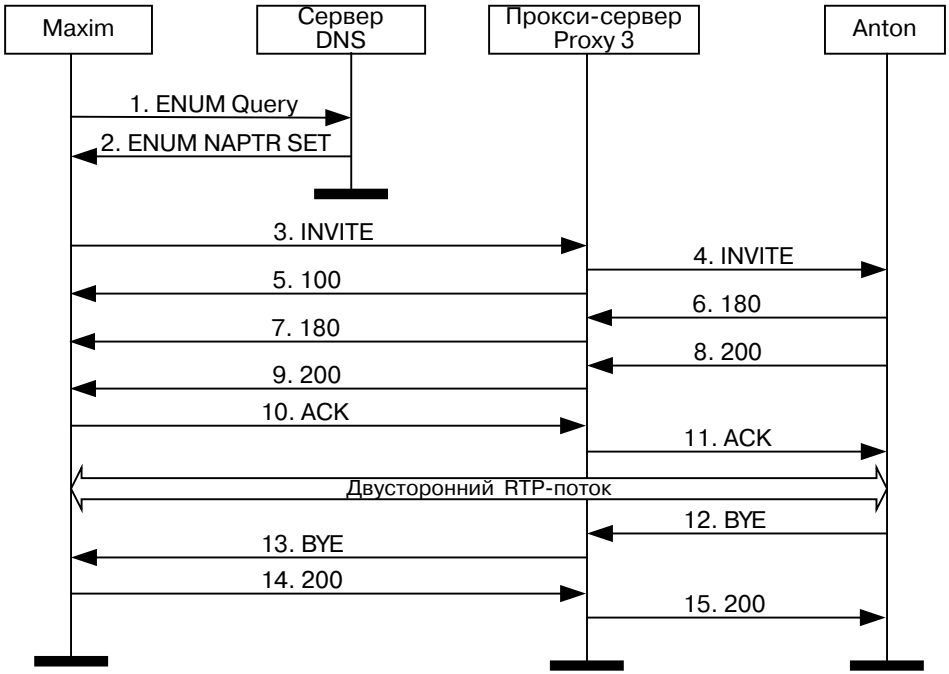


Рис. 13.12. Успешное установление соединения из сети SIP в сеть SIP с использованием ENUM Query

3. INVITE Maxim → Proxy 3

```
INVITE sip:+78122625326@b.niits.ru SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76s1
To: <tel:+78122625326>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sip:+70953864515@client.a.niits.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
```

```
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. INVITE Proxy 3 → Anton

```
INVITE sip:+78122625326@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss3.b.niits.ru:5060;branch=z9hG4bK721e418c4.1
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss3.b.niits.ru;lr>
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sip:+70953864515@client.a.niits.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=UserA 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

5. 100 Trying Proxy 3 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Content-Length: 0
```

6. 180 Ringing Anton → Proxy 3

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ss3.b.niits.ru:5060;branch=z9hG4bK721e418c4.1
;received=192.0.2.233
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.b.niits.ru;lr>
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.niits.ru>
Content-Length: 0
```

7. 180 Ringing Proxy 3 → Maxim

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.b.niits.ru;lr>
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.niits.ru>
Content-Length: 0
```

8. 200 OK Anton → Proxy 3

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss3.b.niits.ru:5060;branch=z9hG4bK721e418c4.1
;received=192.0.2.233
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.b.niits.ru;lr>
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
s=-
c=IN IP4 client.b.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

9. 200 OK Proxy → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss3.b.niits.ru;lr>
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.niits.ru>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.niits.ru
```

```
s=-
c=IN IP4 192.0.2.100
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

10. ACK Maxim → Proxy 3

```
ACK sip:+78122625326@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bq9
Max-Forwards: 70
Route: <sip:ss3.b.niits.ru;lr>
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 ACK
Content-Length: 0
```

11. ACK Proxy 3 → Anton

```
ACK sip:+78122625326@client.b.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss3.b.niits.ru:5060;branch=z9hG4bK721e418c4.1
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bq9
;received=192.0.2.101
Max-Forwards: 69
From: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 2 ACK
Content-Type: application/sdp
Content-Length: 0
```

Между пользователем Maxim и абонентом Anton проключается тракт передачи RTP-потока. Затем Anton кладет трубку, завершая разговор с пользователем Maxim.

12. BYE Anton → Proxy 3

```
BYE sip:+70953864515@client.a.niits.ru SIP/2.0
Via: SIP/2.0/UDP client.b.niits.ru:5060;branch=z9hG4bKfgaw2
Max-Forwards: 70
Route: <sip:ss3.b.niits.ru;lr>
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 BYE
Content-Length: 0
```

13. BYE Proxy 3 → Maxim

```
BYE sip:+70953864515@client.a.niits.ru SIP/2.0
Via: SIP/2.0/UDP ss3.b.niits.ru:5060;branch=z9hG4bK721e418c4.1
;received=192.0.2.100
Via: SIP/2.0/UDP client.b.niits.ru:5060;branch=z9hG4bKfgaw2
Max-Forwards: 69
From: <tel:+78122625326>;tag=314159
```

```
To: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 BYE
Content-Length: 0
```

14. 200 OK Maxim → Proxy 3

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss3.b.niits.ru:5060;branch=z9hG4bK721e418c4.1
;received=192.0.2.233
Via: SIP/2.0/UDP client.b.niits.ru:5060;branch=z9hG4bKfgaw2
;received=192.0.2.100
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 BYE
Content-Length: 0
```

15 .200 OK Proxy 3 → Anton

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.b.niits.ru:5060;branch=z9hG4bKfgaw2
;received=192.0.2.100
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.niits.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 BYE
Content-Length: 0
```

13.3.5. Неуспешное соединение из SIP в ТфОП: сообщение об ошибке из ТфОП

Maxim вызывает абонента Anton через Proxy 1 и шлюз NGW 1. Вызов отклоняется АТС в ТфОП, а абоненту передается звуковое сообщение об ошибке (или звуковой сигнал). Пользователь Maxim, прослушав звуковое сообщение, кладет трубку, в результате чего его терминал передает запрос CANCEL для прекращения обработки вызова. Передается CANCEL, а не BYE, т.к. не было получено финального ответа на INVITE.

Содержимое сообщений:

1. INVITE Maxim → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
```

```
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru>
Proxy-Authorization: Digest username=«Max»,
realm=«a.niits.ru», nonce=«01cf8311c3b0b2a2c5ac51bb59a05b40»,
opaque=«», uri=«sip:+78122625326@ss1.a.niits.ru;user=phone»,
response=«e178fbe430e6680a1690261af8831f40»
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

2. 100 Trying Proxy 1 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

Proxy 1 использует функцию определения местоположения чтобы найти, где находится абонент Anton. На основе этого анализа вызов направляется к шлюзу NGW 1. Клиент терминала пользователя Maxim готовится принимать данные из сети на порт 49172.

3. INVITE Proxy 1 → NGW 1

```
INVITE sip:+78122625326@ngw1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru>
Content-Type: application/sdp
Content-Length: 154
```

```
v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

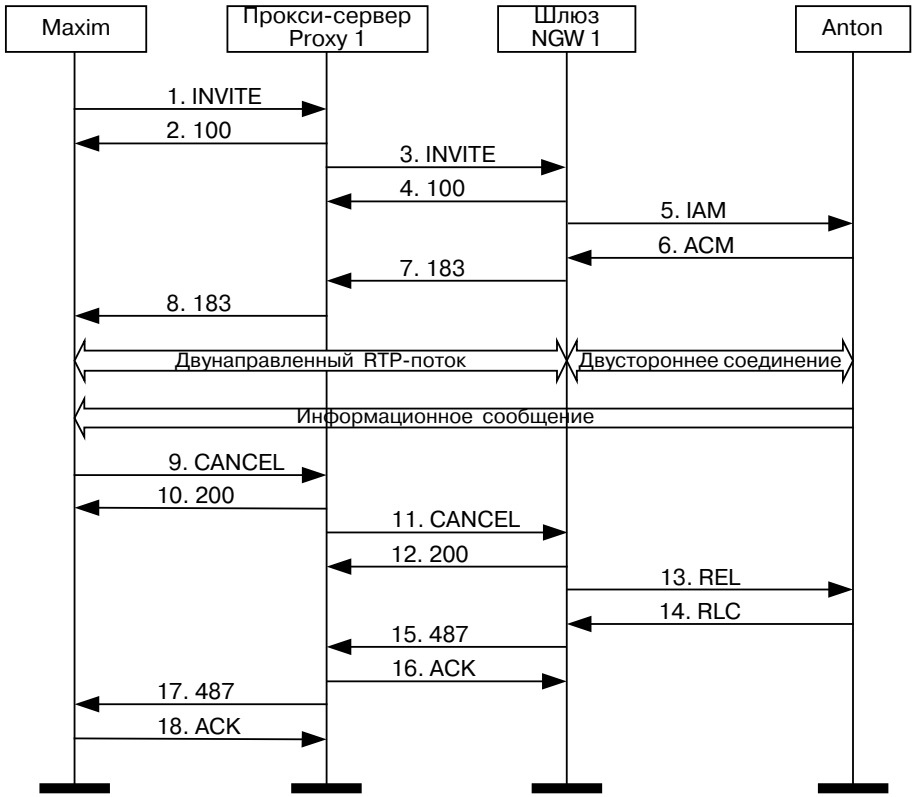


Рис. 13.13. Неуспешное установление соединения из сети SIP в ТфОП: сообщение об ошибке из ТфОП

4. 100 Trying NGW 1 → Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. IAM NGW 1 → АТС Б

Передается сообщение IAM.

```
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National
```

6. ACM АТС Б → NGW 1

Передается сообщение ACM.

7 .183 Session Progress NGW 1 → Proxy 1

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

8. 183 (Session Progress) Proxy 1 → Maxim

```
SIP/2.0 183 Session Progress
```

```
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Вызывающий абонент прослушивает речевые объявления, после чего кладет трубку.

9. CANCEL Maxim → Proxy 1

```
CANCEL sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

10. 200 OK Proxy 1 → Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

11. CANCEL Proxy 1 → NGW 1

```
CANCEL sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
```

```
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

12. 200 OK NGW 1 → Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 CANCEL
Content-Length: 0
```

13. REL NGW 1 → АТС Б

Передается сообщение REL с CauseCode=18 No user responding.

14. RLC АТС Б → NGW 1

Передается сообщение RLC.

15. 487 (Request Terminated) NGW 1 → Proxy 1

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

16. ACK Proxy 1 → NGW 1

```
ACK sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

17 .487 (Request Terminated) Proxy 1 → Maxim

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

18. ACK Maxim → Proxy 1

```
ACK sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

13.3.6. Неуспешное соединение из сети SIP в ТфОП: ТфОП отклоняет вызов, передавая REL с кодом причины

Maxim вызывает абонента Anton через Proxy 1, шлюз NGW 1 и АТС Б. АТС в ТфОП отклоняет вызов, передав сообщение REL с соответствующим кодом причины. Этот код преобразуется в ответ SIP 404 (Not Found), который передается к терминалу пользователя Maxim.

Содержимое сообщений.

1. INVITE Maxim → Proxy 1

```
INVITE sip:+44-1234@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru;transport=tcp>
Proxy-Authorization: Digest username=«Max»,
realm=«a.niits.ru», nonce=«j1c3b0b01cf832da2c5ac51bb59a05b40»,
opaque=«», uri=«sip:+44-1234@ss1.a.niits.ru;user=phone»,
```

```
response=«a451358d46b55512863efe1dccaa2f42»
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

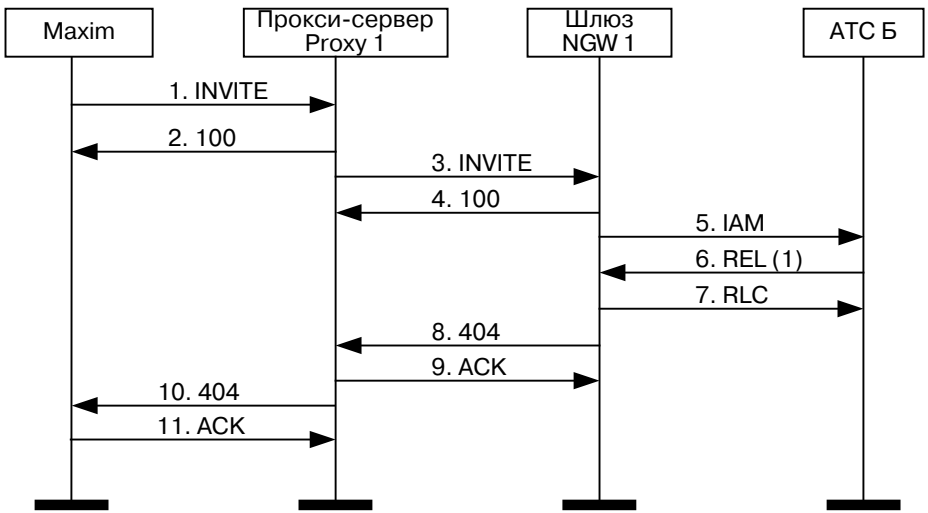


Рис. 13.14. Неуспешное установление соединения из сети SIP в ТфОП: ТфОП отклоняет вызов, передавая REL с кодом причины

2. 100 (Trying) Proxy 1 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

Proxy 1 использует функцию определения местоположения чтобы найти, где в данный момент находится Anton. Далее вызов передается к шлюзу NGW 1. Клиент терминала Maxim готовится принимать данные из сети на порт 49172.

3. INVITE Proxy 1 → NGW 1

```
INVITE sip:+44-1234@ngw1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 (Trying) NGW 1 → Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. IAM NGW 1 → АТС Б

Передается сообщение IAM.

```
CdPN=44-1234,NPI=E.164,NOA=International
CgPN=095-386-4515,NPI=E.164,NOA=National
```

6 .REL АТС Б → NGW 1

Передается сообщение REL с CauseValue=1 Unallocated number.

7. RLC NGW 1 → АТС Б

Передается сообщение RLC.

Шлюз преобразует CauseValue=1 в ответ SIP 404 (Not Found).

8. 404 (Not Found) NGW 1 → Proxy 1

```
SIP/2.0 404 Not Found
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Error-Info: <sip:not-found-ann@ann.a.niits.ru>
Content-Length: 0
```

9. ACK Proxy 1 → NGW 1

```
ACK sip:+44-1234@ngw1.a.niits.ru;user=phone SIP/2.0
Max-Forwards: 70
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

10 .404 (Not Found) Proxy 1 → Maxim

```
SIP/2.0 404 Not Found
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Error-Info: <sip:not-found-ann@ann.a.niits.ru>
Content-Length: 0
```

11. ACK Maxim → Proxy 1

```
ACK sip:+44-1234@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.niits.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

13.3.7. Неуспешное соединение из сети SIP в ТфОП: срабатывает таймер ожидания шлюзом сообщения ANM

Пользователь Maxim вызывает абонента Anton через Proxy 1, шлюз NGW 1 и АТС Б. Шлюз прерывает установление соединения после того, как сработал таймер ожидания сообщения ANM от АТС ТфОП (т.е. вызываемый абонент в течение определенного времени не отвечает на вызов). Шлюз передает сообщение REL в ТфОП и ответ 480 (Temporarily Unavailable) к терминалу пользователя Maxim. Содержимое сообщений:

1. INVITE Maxim → Proxy 1

```
INVITE sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru;transport=tcp>
Proxy-Authorization: Digest username=«Max»,
realm=«a.niits.ru», nonce=«da2c5ac51bb59a05j1c3b0b01cf832b40»,
opaque=«», uri=«sip:+78122625326@ss1.a.niits.ru;user=phone»,
response=«579cb9db184cdc25bf816f37cbc03c7d»
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения местоположения, чтобы найти, где находится абонент Anton. На основе этого анализа вызов передается к шлюзу NGW 1. Клиент терминала пользователя Maxim готовится принимать данные из сети на порт 49172.

2. 100 (Trying) Proxy 1 → Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
```

```
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

3. INVITE Proxy 1 → NGW 1

```
INVITE sip:+78122625326@ngw1.a.niits.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.niits.ru
s=-
c=IN IP4 client.a.niits.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

4. 100 Trying NGW 1 → Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Content-Length: 0
```

5. IAM NGW 1 → АТС Б

Передается сообщение IAM.

```
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National
```

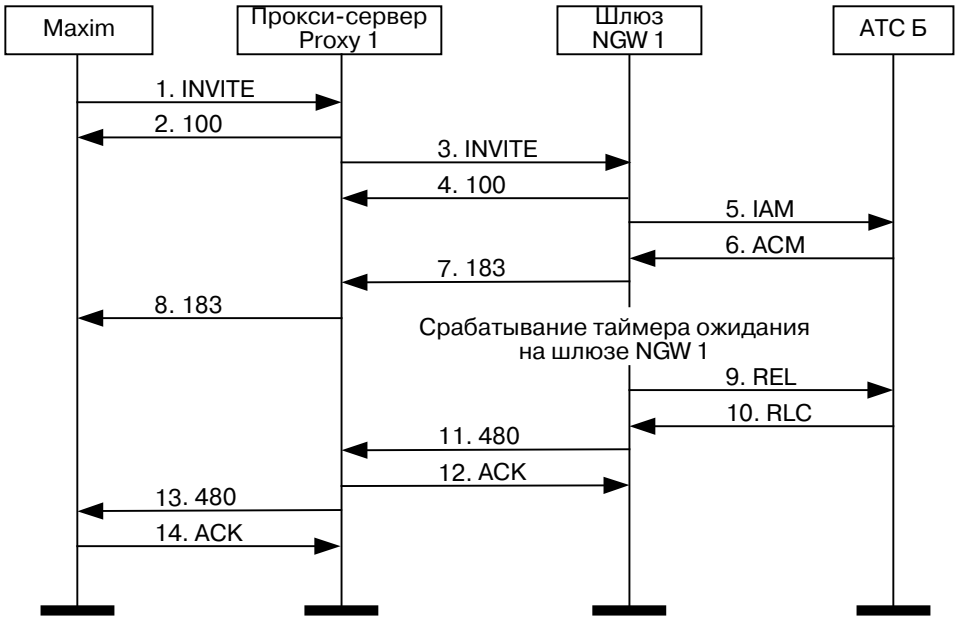


Рис. 13.15. Неуспешное установление соединения из сети SIP в ТфОП: срабатывает таймер ожидания шлюзом сообщения ANM

6. ACM АТС Б → NGW 1

Передается сообщение ACM.

7. 183 (Session Progress) NGW 1 → Proxy 1

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
```

```
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

8. 183 (Session Progress) Proxy 1 → Maxim

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.niits.ru;lr>
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.niits.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.niits.ru
s=-
c=IN IP4 ngw1.a.niits.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

После того как истекает время ожидания на шлюзе, он передает REL в ТфОП и ответ 480 (Temporarily Unavailable) в сеть SIP.

9. REL NGW 1 → АТС Б

Передается сообщение REL с CauseCode=18 No user responding.

10. RLC АТС Б → NGW 1

Передается сообщение RLC.

11. 480 (Temporarily Unavailable) NGW 1 → Proxy 1

```
SIP/2.0 480 Temporarily Unavailable
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
```

```
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Error-Info: <sip:temp-unavail-ann@ann.a.niits.ru>
Content-Length: 0
```

12. ACK Proxy 1 → NGW 1

```
ACK sip:ngw1@a.niits.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.niits.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

13. 480 (Temporarily Unavailable) Proxy 1 → Maxim

```
SIP/2.0 480 Temporarily Unavailable
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 INVITE
Error-Info: <sip:temp-unavail-ann@ann.a.niits.ru>
Content-Length: 0
```

14. ACK Maxim → Proxy 1

```
ACK sip:+78122625326@ss1.a.niits.ru;user=phone SIP/2.0
Max-Forwards: 70
Via: SIP/2.0/TCP client.a.niits.ru:5060;branch=z9hG4bK74bf9
From: Max <sip:+70953864515@ss1.a.niits.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.niits.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.niits.ru
CSeq: 1 ACK
Content-Length: 0
```

Глава 14. Формирование телефонных URI

14.1. Телефонные SIP URI

Прокси-сервер в сети SIP может маршрутизировать сообщения по любому признаку, заданному администратором сети. Обычно для этой цели используется поле Request-URI. Поля заголовков **To** и **From** также могут быть важны при решении задач маршрутизации. Преобразование SIP-ТфОП предполагает, что прокси-сервер использует, как минимум, эти три поля, каждое из которых содержит URI.

При преобразовании SIP–ТфОП часто требуется получить из SIP URI телефонный номер. Но может также потребоваться из телефонного номера в сообщении ISUP получить SIP URI.

Формат, наиболее часто используемый в SIP для представления телефонных номеров, – это tel URL. При преобразовании формата tel URL может целиком занимать поле URI, т.е. tel URL используется как URI, или может быть пользовательской частью адреса в SIP URI. Например, поле **To** может выглядеть так:

```
To: tel:+78123875605
```

или так:

```
To: sip:+78123875605@protei.ru
```

Знак «+», предшествующий телефонному номеру в tel URL, показывает, что цифры, следующие за ним, – полный телефонный номер в формате E.164. Это

означает, что за кодом страны следует код зоны или города. Отсутствие знака «+» может означать, что это местный или специальный номер, использующийся для обеспечения анонимности пользователя. Когда знак «+» отсутствует, но телефонный номер используется как пользовательская часть адреса URI, то SIP URI должен содержать необязательный параметр «;user=phone», например:

```
To: sip:83000@sip.niits.net;user=phone
```

При работе между двумя шлюзами SIP-T рекомендуется использовать международный формат номера E.164, особенно в тех случаях, когда эти шлюзы находятся в разных областях или административных доменах. В большинстве сетей, где не используется SIP-T, шлюзы не отвечают за сквозную маршрутизацию вызова. Фактически это означает, что шлюз не сможет узнать, в локальном или в удаленном административном домене и/или регионе был отклонен вызов, и, следовательно, шлюз должен всегда использовать международный формат номера. Нет никакой гарантии того, что оборудование сети SIP сможет правильно понять национальный формат телефонного номера, набранного вызывающим абонентом. Если шлюз, закрепленный за вызывающим абонентом, переведет номер в международный формат, то этот номер сможет быть правильно распознан удаленными элементами сети.

В сигнализации ISUP формат телефонного номера определяется рядом параметров, таких, например, как Called Party Number (CPN) и Calling Party's Number (CIN); когда номер вызывающего абонента представляется этими параметрами, к ним добавляется некоторая дополнительная информация.

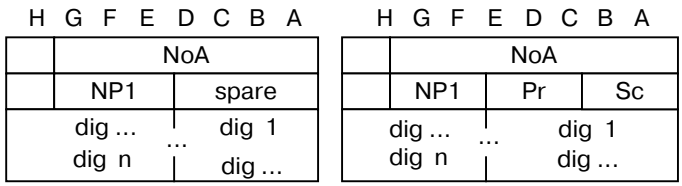


Рис. 14.1. Форматы номеров ISUP по [6]

В одном из стандартов по SIP-T [6] CPN определяется как ISUP format, а формат номера вызывающего абонента CIN – как ISUP calling format. Формат содержит байт, называемый индикатором типа адреса – Nature of Address (NoA) indicator, который сопровождается другим байтом, содержащим индикатор плана нумерации – Numbering

Plan Indicator (NPI). Оба они являются префиксами к серии переменной длины, содержащей цифры телефонного номера в двоичной форме BCD (Binary Coded Decimal). В случае номера вызывающего абонента поле NPI содержит также биты таких полей, как Presentation Indicator, который определяет, будет ли отображен номер на дисплее пользователя или нет, а также Screening Indicator, который показывает, проводилась ли проверка этого номера и ее результаты.

В поле NPI обычно помещается значение «ISDN (Telephony) numbering plan (Recommendation E.164)», но это не означает, что цифры, следующие за полем NPI, содержат код страны; поле NoA определяет, какой формат имеет телефонный номер – национальный или международный. В случае, когда формат номера отличается от международного, поле NoA обычно содержит дополнительную информацию, определяющую специфику национального плана нумерации; основываясь на этой информации, можно определить, каким образом перевести номер в международный формат. Поле NPI может содержать значение, отличное от «ISDN numbering plan»; тогда tel URI может не подходить для переноса адресной информации, а описание обработки вызовов такого типа выходит за рамки данного справочника.

14.2. Процедура преобразования формата ISUP в формат tel URL

На основе сказанного ранее преобразование из формата ISUP в tel URL производится следующим образом. Сначала, если поле NPI показывает, что используется формат номера E.164, происходит обращение к полю NoA. Если значение этого поля указывает на то, что используется международный формат номера, то при преобразовании перед цифрами номера должна быть добавлена строчка «tel:+». Если поле NoA указывает на то, что номер – в национальном формате, то до того как номер будет преобразован в tel URL, перед его цифрами необходимо добавить код страны.

Если в сообщении присутствует параметр, содержащий специфическую информацию об имени вызывающего абонента (например, Generic Name Parameter в ANSI), то если параметр Presentation Indicator не имеет значение presentation restricted, эта информация добавляется в поле отображаемого имени заголовка **From**. Если при преобразовании адресов используется ISUP calling format, то нужно принять во внимание два параметра: presentation indicators и screening indicators.

Если параметр presentation indicators имеет значение «presentation restricted», шлюзом должен быть создан специальный URI, который сообщит удаленному терминалу, что идентификация пользователя не проводится. Этим URI должен быть SIP URI, у которого в поле отображаемого имени и имени пользователя записано «Anonymous», например:

```
From: Anonymous <sip:anonymous@anonymous.invalid>
```

Если параметр presentation indicators имеет значение «address unavailable», шлюз должен обрабатывать сообщение IAM так, будто параметр CIN отсутствует. Параметр screening indicators в данном случае не преобразуется.

14.3. Процедура преобразования формата tel URL в формат ISUP

Преобразование из tel URL в формат ISUP выполняется проще. Если URI находится в международном формате, шлюз должен проанализировать код страны из URI. Если код страны является для шлюза локальным (шлюз имеет один или больше каналов, через которые можно установить речевое соединение, не выходя за пределы данного кода страны), то полю NoA должно быть присвоено значение «national (significant) number», и код страны должен быть удален из URI перед трансляцией. Если область, определяемая кодом страны, не является для шлюза локальной, в поле NoA помещается значение «international number», и код страны оставляется в URI. В любом случае поле NPI должно иметь значение «ISDN numbering plan». Если URI находится не в международном формате, шлюз может попытаться обработать URI, как будто информация записана в национальном или специфическом для данной сети формате номера. Если это приведет к внутренней ошибке шлюза, или если шлюз не поддерживает такие процедуры обработки, к вызывающему оборудованию передается ответ с соответствующим кодом состояния SIP, отражающим, что URI не был понят шлюзом (если этот URI является Request-URI, передается ответ 484 (Address incomplete)).

При преобразовании tel URL в ISUP calling format процедура идентична приведенной выше, но дополнительно параметру presentation indicator присваивается значение «presentation allowed», а параметру screening indicator – значение «network provided», если настройки сети или параметры пользователя не указывают другие значения.

Глава 15. Преобразование, тестирование и реализация SIP

15.1. Подходы к преобразованию сигнализации SIP

При взаимодействии разных сетей и телекоммуникационного оборудования часто возникает потребность в *конвертерах сигнализации*, выполняющих преобразование (на основании определенных правил) одних сигнальных сообщений в другие с соответствующей обработкой содержимого полей в этих сообщениях. Такое преобразование называют также *мэппингом (mapping*). Подразумевается, что при приеме определенного сигнального сообщения одной системы сигнализации конвертер передает во вторую систему сигнализации другое сигнальное сообщение, однозначно соответствующее принятому. Например, при приеме сообщения IAM сигнализации ОКС7 [27] необходимо передать сообщение INVITE протокола SIP. При обработке полей сигнальных сообщений также используются определенные правила. Так, на основании поля «Номер вызываемого пользователя» сообщения IAM формируется поле To сообщения INVITE. Более детально процессы взаимодействия сетей на базе протоколов SIP и ОКС7 рассмотрены в главах 12 и 13 этой книги.

Концепция сетей NGN предполагает использование различных протоколов на разных уровнях сети. Как показано на рис. 15.1, протокол SIP обеспечивает как взаимодействие между несколькими программными коммутаторами Softswitch, так и взаимодействие между Softswitch и оконечными устройствами (IP-телефонами, soft-телефонами, SIP IAD и т.п.). Кроме того, в Softswitch сети NGN требуется преобразование протокола SIP в протокол H.323, MGCP или H.248/MEGACO.

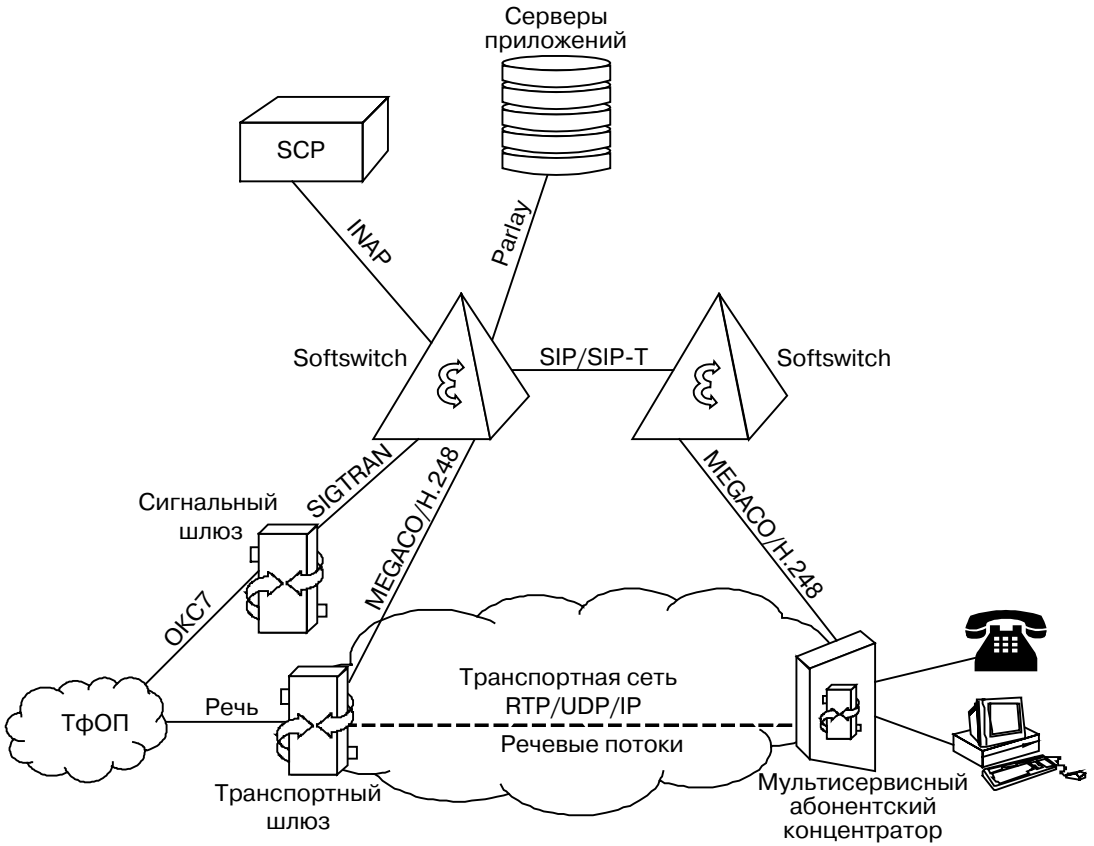


Рис. 15.1. Преобразование сигнализации в NGN

В зависимости от принципа построения сети и используемого оборудования для этого преобразования могут использоваться *шлюзы сигнализации*. Для поддержки ОКС7 и преобразования SIP/ОКС7 в Softswitch разработана специальная технология SIGTRAN. В ней сообщения ОКС7 передаются по IP-сети, причем вместо протоколов UDP и TCP используется протокол SCTP (*Stream Control Transport Protocol*), обеспечивающий как надежную доставку сообщений, так и высокую скорость передачи. Стек протоколов для этого случая показан на рис. 15.2.

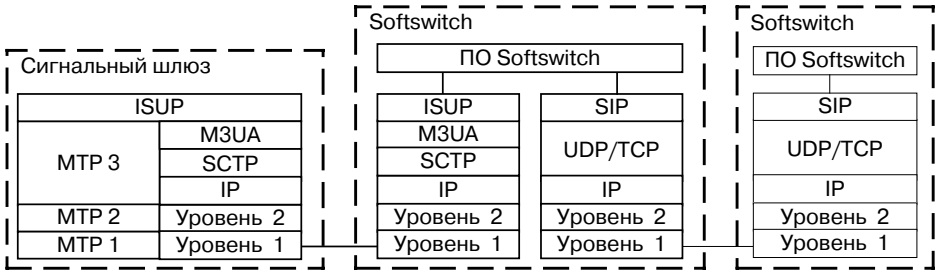


Рис. 15.2. Стек протоколов при взаимодействии сигнального шлюза и программного коммутатора Softswitch по технологии SIGTRAN

В шлюзе сигнализации сообщения ОКС7 извлекаются из сигнального канала, упаковываются в IP-пакеты и передаются через транспортную сеть к программному коммутатору Softswitch. Программное обеспечение Softswitch определяет направление вызова и дает запрос на установление сеанса связи по протоколу SIP. Стоит отметить, что в этом примере целесообразно применять технологию SIP-T, так как она предусматривает сквозную передачу сообщений ОКС7, информация которых может потребоваться на других узлах.

Для представленного на рис. 15.2 преобразования используется конвертер сигнализации, называемый *шлюзом сигнализации (Signaling Gateway*), роль которого может выполнять шлюз IP-телефонии, содержащий в себе и функции транспортного шлюза (кодирование/декодирование речи, упаковка/извлечение речевых пакетов в/из IP-датаграммы, работа с джиттер-буфером и др.). Существует ряд различных шлюзов, имеющих разнообразные интерфейсы и поддерживающих разные протоколы. В качестве наглядного примера рассмотрим алгоритм работы и пример настройки шлюза IP-телефонии Протей-ITG.

Из ТфОП с сигнализацией ОКС7 в шлюз поступают телефонные вызовы, которые необходимо направить на определенное устройство в IP-сети (Softswitch, другой шлюз, IP-телефон и др.), уже с применением протокола SIP. Прежде всего, необходимо определить алгоритм маршрутизации вызова, для чего создается ряд правил, описывающих соответствие между телефонным номером вызываемого абонента и IP-адресом узла, который будет обрабатывать вызов:

```
{1111; SIP={ PrimaryIP=192.168.100.71; SecondaryIP=192.168.100.72;}}
{2222; SIP={ PrimaryIP=192.168.100.73; SecondaryIP=192.168.100.74;}}
{3333; SIP={ PrimaryIP=192.168.100.75; SecondaryIP=192.168.100.76;}}
```

Эти правила означают, что вызовы с номером 1111 будут отправляться на узел с адресом 192.168.100.71, а в случае его недоступности – на узел 192.168.100.72, вызовы с номером 2222 – на узлы 192.168.100.73 и 192.168.100.74, вызовы с номером 3333 – на узлы 192.168.100.75 и 192.168.100.76. В случае необходимости, номер можно модифицировать (удалить цифры, добавить префикс и т.д.) или заменить. Если вызовы в сети обслуживает прокси-сервер, то в специальном поле записывается его адрес (в приведенном ниже примере – 192.168.100.169) и номер порта (5060), и тогда все вызовы будут обрабатываться этим прокси-сервером:

```
Proxy = {192.168.100.169:5060}
```

Отметим, что номер порта может отличаться от стандартного 5060 по соображениям безопасности или для разделения трафика разных приложений. Шлюз Протей-ITG поддерживает процедуру проключения разговорных каналов в предответном состоянии, что позволяет пользователю получить информацию или воспользоваться определенной услугой без начисления платы. С точки зрения протокола SIP это требует возможности передачи SDP в информационных ответах 18x. Поддержка такого режима работы в шлюзе необходима по следующим причинам:

- некоторые телефонные службы могут работать в предответном состоянии, например, спецслужбы «01», «02», «03»;
- информационные подсказки абоненту передаются, как правило, в предответном состоянии;
- сигналы «Контроль посылки вызова» и другие могут передаваться в предответном состоянии.

Существует также ряд конвертеров, которые преобразуют сигнализацию SIP в сигнализацию H.323 и обратно. Они требуются в случае взаимодействия разнотипного оборудования, работающего на базе этих двух разных протоколов. Одна из сложностей такого преобразования заключается в том, что протокол H.323 может создавать речевые каналы после ответа вызывающего пользователя. В протоколе SIP информация о речевых каналах передается уже в первом запросе INVITE. Решить эту задачу можно, использовав в протоколе H.323 процедуру FastConnect, а также с помощью повторной передачи запроса INVITE с описанием речевых каналов. Кроме того, для этих целей можно использовать запрос ACK.

На рис. 15.3 показан простейший сценарий взаимодействия этих протоколов. Получив сообщение H.323 Setup, сигнальный шлюз передает запрос INVITE.

После приема ответа 180 Ringing сигнальный шлюз передает сообщение H.323 Alerting, которое информирует о передаче вызываемому пользователю сигнала «Посылка вызова». После того как этот пользователь снял трубку, в ответе 200 OK передается информация о параметрах речевого канала вызванной стороны. Далее инициируются процедуры протокола H.245, целью которых является проключение речевых каналов между терминалами пользователей. В запросе ACK содержится описание характеристик сеанса на вызвавшей стороне.

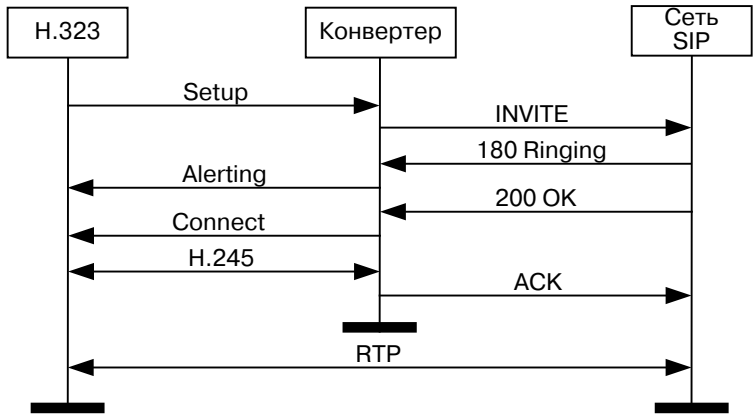


Рис. 15.3. Пример преобразования сообщений H.323 в сообщения SIP

15.2. Тестирование протокола SIP

Стандартом де-факто для тестирования современных систем сигнализации ЕСЭ РФ в целом и протокола SIP в частности является отечественная платформа SNT, портативный вариант которой изображен на рис. 15.4. В определенной степени, протокол-тестер SNT может рассматриваться и как система обучения SIP благодаря удобному и интуитивно понятному пользовательскому интерфейсу на русском языке, значительному числу сервисных функций и возможности задавать значения параметров протокола в процессе тестирования и конфигурации системы.

Рис. 15.4. Протокол-тестер SNTlite

Протокол-тестеры SNT достаточно подробно рассмотрены в других книгах серии «Телекоммуникационные протоколы». Здесь отметим лишь, что SNT оснащен набором разных библиотек кодирования/декодирования рассмотренных в предыдущих главах сообщений SIP и позволяет тестировать реализации версий протокола в разных сетевых режимах.

15.3. Реализация услуг на базе SIP

В самом начале книги отмечалось, что первые версии протокола SIP учитывали только одну услугу – постановку вызова на ожидание. В этом случае сторона, которая желала поставить вызов на ожидание, отправляла запрос INVITE с нулевым значением (0.0.0.0.) параметра, определяющего IP-адрес терминала пользователя (параметра c) в SDP-описании сеанса. Описание других услуг для них полностью отсутствовало.

Активное развитие IP-технологий и движение телекоммуникационных технологий по пути перехода к сетям связи следующего поколения вынудили разработчиков обратить серьезное внимание на расширение возможностей протокола SIP в создании новых инфокоммуникационных услуг. Если рассматривать предоставление дополнительных услуг с использованием протокола SIP с точки зрения управления, то можно выделить два подхода: на основе децентрализованного и на основе централизованного управления.

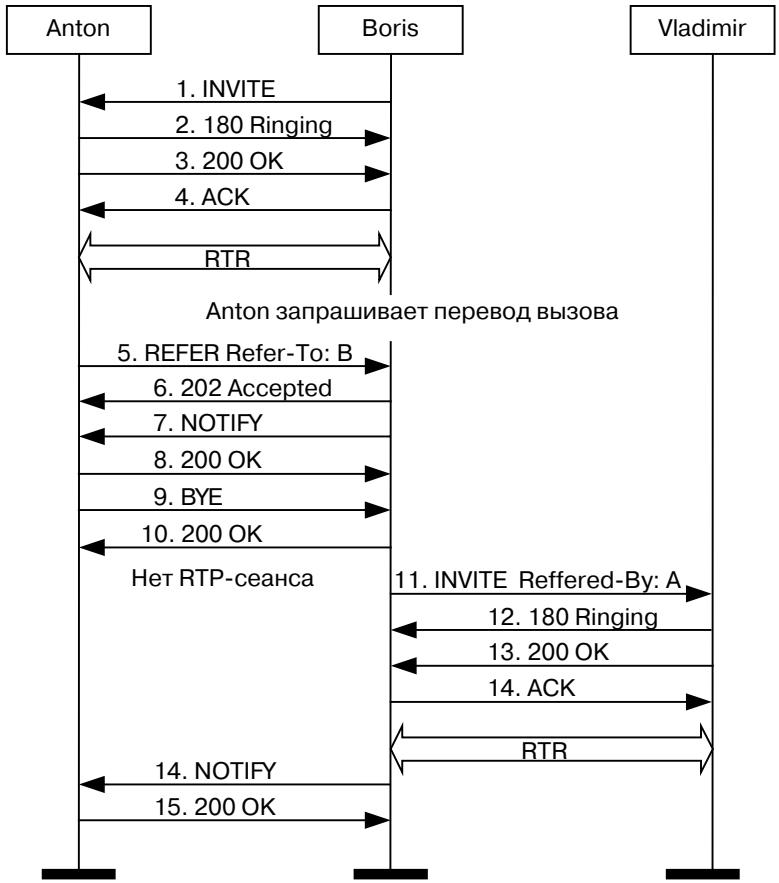


Рис. 15.5. Услуга «Перевод вызова» при децентрализованном способе управления

В случае *децентрализованного управления услугами* каждое оконечное оборудование UA, например, SIP-телефоны и оборудование доступа, содержит в себе алгоритмы предоставления услуг и выполняет их по запросу пользователя. При этом, в зависимости от услуги, оно может вызывать других пользователей, воспроизводить подсказки, запрашивать и принимать от пользователей цифровую информацию и др.

В качестве примера реализации децентрализованного способа можно привести применение запроса REFER в изображенном на рис. 15.5 сценарии перевода вызова на другого пользователя. В этом примере после разговора пользователей Boris и Anton последний требует с помощью запроса REFER (F5) переводит вызов к пользователю Vladimir. На основании адреса, указанного в REFER, терминал Boris вызывает терминал Vladimir. Как видно из рис. 15.5, каждый терминал по команде пользователя выполняет определенные действия по обработке вызова.

Отметим, что децентрализованный способ не исключает использование прокси-серверов, которые в этом случае отвечают только за маршрутизацию вызовов по IP-сети. Это достаточно простой способ, однако он имеет ряд существенных недостатков: усложнение оконечного оборудования; невозможность организации услуг, которые предусматривают групповую работу (т.е. объединение пользователей в группы и предоставление им единых услуг); сложность управления услугами, так как у обслуживающего персонала нет доступа к услугам на терминале пользователя.

Централизованное управление услугами предусматривает использование единого сервера, управляющего предоставлением услуг многим пользователям сети. Для этого он содержит специализированное программное обеспечение, поддерживающее реализацию логики работы услуг. Терминал пользователя выполняет простейшие функции установления базового соединения, принимает информацию от пользователя (как правило, цифры номера) и передает ее на центральный сервер для обработки.

На рис. 15.6 показан сценарий обмена сообщениями протокола SIP для услуги «Перевод вызова» при управлении со стороны центрального сервера. Показан случай, когда терминал пользователя Anton при подъеме трубки автоматически делает вызов на сервер, чтобы тот мог управлять обработкой вызова. После набора цифр номера, которые в примере передаются с помощью запросов INFO, сервер устанавливает соединение с пользователем Boris. Для запроса услуги «Перевод вызова» Anton вводит комбинацию цифр, на основании которой сервер запускает логику предоставления услуги и устанавливает соединение между пользователями Vladimir и Boris.

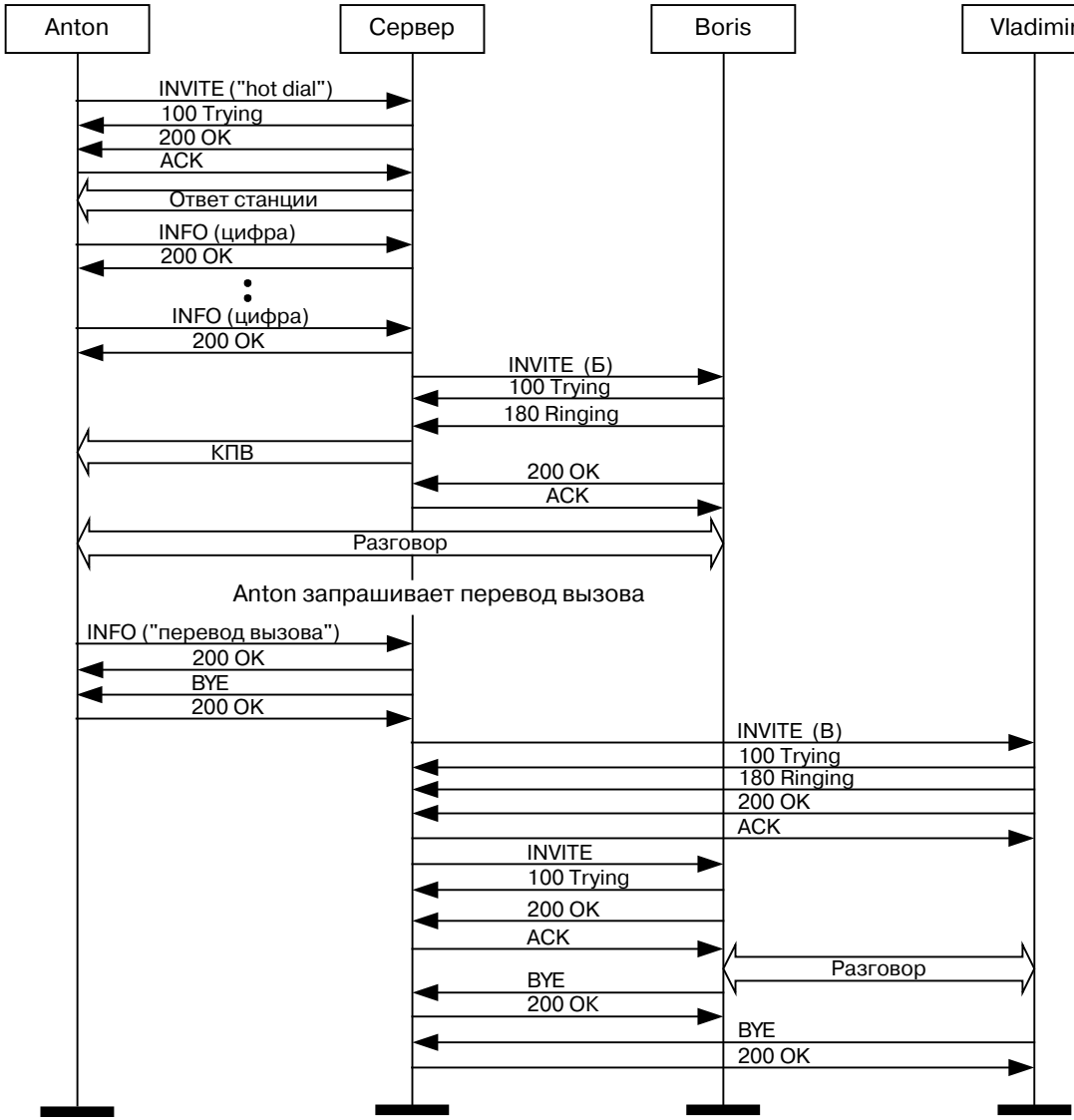


Рис. 15.6. Услуга «Перевод вызова» при централизованном способе управления

Для предоставления услуг сервер на рис. 15.7 может обращаться за инструкциями к внешним устройствам, например, к узлу управления услугами SCP Интеллектуальной сети, серверу приложения по API Parlay и др. В сетях следующего поколения этот сервер будет представлять собой программный коммутатор Softswitch. Оператор может настроить правила обработки вызовов для каждого пользователя. После получения вызова сервер предоставления услуг на основании конфигурации обращается к соответствующему устройству за инструкциями, которые затем выполняет. Наиболее совершенным в этом отношении является интерфейс Parlay; этот гибкий и эффективный механизм позволяет реализовать огромное количество услуг на базе единого стандарта.

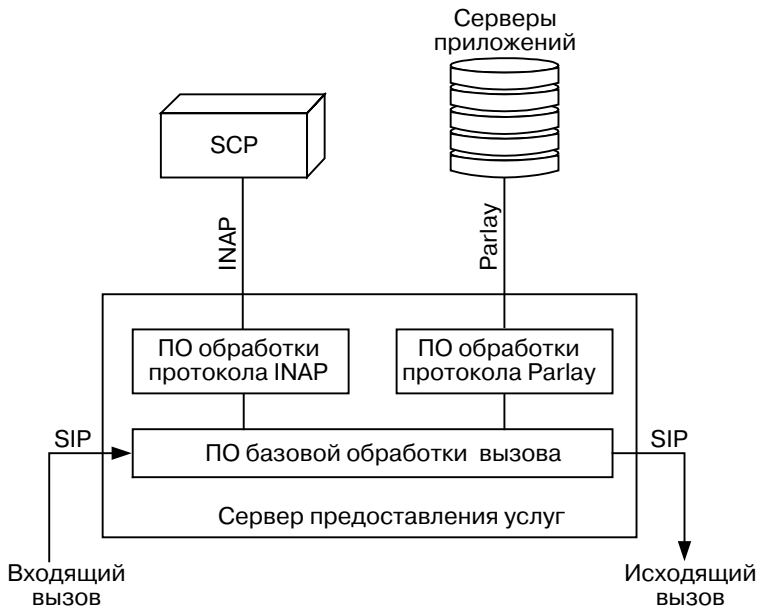


Рис. 15.7. Функциональная схема сервера предоставления услуг

Организация IETF прорабатывает вопросы взаимодействия Интеллектуальных сетей и сетей на базе протокола SIP. Это нужно, в частности, при предоставлении в NGN услуги 800 (Бесплатный вызов) и других подобных услуг. Если услугами управляет SCP, то сервер представляет собой «виртуальный» SSP со всеми его стандартизированными функциями.

15.4. PINT и SPIRITS

Эволюция технологий предоставления интеллектуальных услуг связи рассмотрена в [22]. Классическая концепция Интеллектуальной сети предполагает сосредоточение логики услуг в отдельном сетевом устройстве – узле управления услугами SCP. При этом услуги могут действовать в любых пределах: в местной, междугородной или международной сети. Другим направлением реализации интеллектуальных услуг является подход, основанный на применении узлов услуг SN (Service Node), характерным примером которого является интеллектуальная платформа Протей. Первоначально оба вышеназванных подхода не учитывали одного важного фактора – всемирной сети Интернет, развитие которой и привело к процессам конвергенции ТфОП и Интернет. Проблематика новых инфокоммуникационных услуг на основе взаимодействия этих сетей стала предметом исследований международных организаций по стандартизации. Концепция построения таких услуг была предложена в IETF (Internet Engineering Task Force) рабочими группами PINT (Public Switched Telephone Network (PSTN)/Internet Inter-Networking) и SPIRITS (Service in the PSTN/IN Requesting InTernet Service). Примерами услуг, предоставляемых конвергентными сетями, являются услуги Click-to-Dial-Back, Internet Call Waiting, Internet Call Forwarding и т.д.

Услуга Internet Call Waiting, например, позволяет уведомить о входящем вызове пользователя, терминал которого занят Интернет-сеансом. При этом пользователь может заранее определить алгоритм обслуживания поступающих вызовов и:

- отказаться от вызова;
- принять вызов прямо на персональный компьютер, оснащенный звуковой картой и гарнитурой, с использованием IP-телефонии;
- переадресовать вызов на другой телефонный номер;

- переадресовать вызов на речевую почту, причем вызываемый пользователь сохраняет Интернет-сеанс, а вызывающий слышит подсказку, приглашающую его оставить речевое сообщение;
- заранее определить речевое сообщение, которое будет передано вызывающему пользователю, например: «Извините, пользователь не может ответить на Ваш вызов, позвоните позже».

Концепция рабочей группы SPIRITS предусматривает возможность обращения из ТфОП к функциям выполнения услуг в IP-сети. Другими словами, если какое-либо событие происходит на телефонной станции, выполняющей функции SSP, о нем сообщается терминалу пользователя, который и принимает решение о дальнейших действиях. Однако этого недостаточно, потому что терминал должен уметь и сам запрашивать выполнение определенных действий в ТфОП/IN, например, запрограммировать SSP так, чтобы он сообщал обо всех вызовах, поступающих на заданную абонентскую линию. Для этого специалистами SPIRITS был взят материал, разработанный группой PINT, в связи с чем, как правило, две системы изображают совместно.

Рассмотрим услугу Internet Call Waiting (рис.15.8). В основе архитектуры для предоставления этой услуги лежит классическая Интеллектуальная сеть, но в SCP добавлена возможность взаимодействия с узлами сети Интернет. Основным протоколом сети коммутации пакетов для подобной сети является протокол SIP [3-5]. Как уже отмечалось, SIP, в классическом его варианте, – это протокол установления мультимедийный соединений в сетях с коммутацией пакетов (в IP-сетях, как самых распространенных).

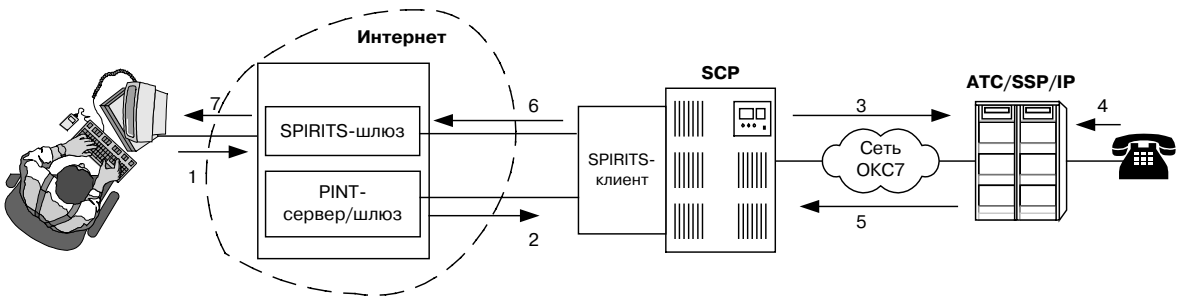


Рис. 15.8. Услуга ICW на основе архитектуры, предложенной SPIRITS

В функции специального узла – прокси-сервера – входит обслуживание заявок от терминалов пользователей и пересылка сообщений от SCP к терминалам и обратно. Для этого в его состав входят SPIRITS-шлюз, который обслуживает сообщения от SPIRITS-клиента и SPIRITS-сервера, и PINT-сервер/шлюз, обслуживающий сообщения от PINT-клиента.

Для представленной на рис. 15.8 услуги ICW терминал пользователя (PINT-клиент) сразу после установления модемного соединения с Интернет передает запрос запустить триггерную точку в SSP, которая будет отслеживать поступающие вызовы (1); прокси-сервер перенаправляет сообщение к SCP (2); SCP с помощью протокола INAP активизирует триггерную точку в SSP (3). При поступлении вызова (4) информация о нем передается в SCP (5). SCP знает о том, что услугу запросил определенный терминал в сети Интернет, и передает сообщение на прокси-сервер (6), который пересылает его к терминалу пользователя (7). Дальнейшие действия зависят от пользователя, о возможных вариантах уже было написано ранее. Сигнальные сообщения в IP-сети представляют собой запросы и ответы протокола SIP.

Глава 16. Quo Vadis?

16.1. Дальнейшее развитие SIP

Включение этой последней главы в книгу, названную справочником, может показаться несколько неуместным. Для такого решения у авторов было два аргумента. Во-первых, своего рода прецедент, имевший место в культовом справочнике 80-х годов [60]. Во-вторых, уже неоднократно звучавшие аналогии между SIP и общеканальной сигнализацией ОКС7, позволяющие предполагать, что SIP будет развиваться не менее успешно, чем это было со стеком протоколов ОКС7.

Благодаря своей открытой организации и проистекающей из этого гибкости и расширяемости, протокол SIP получил широчайшее признание в качестве основного протокола сигнализации IP-телефонии для проводных и беспроводных сетей. К тому же, SIP с самого начала был задуман как Интернет-протокол, и чтобы найти местоположение пользователя, он мог использовать любые имеющиеся в его распоряжении средства. Он может ретранслировать приглашение к сеансу связи между серверами SIP, перенаправлять приглашение другим серверам, параллельно или последовательно разветвлять приглашение (повторять приглашение к сеансу по многим ветвям). SIP переносит описание сеанса как тело (body) и производит минимальное количество согласований сеанса, поддерживает примитивы для установления и прекращения сеансов. Вместо числовых идентификаторов SIP применяет для представления пользователей распространенные и хорошо известные идентификаторы, аналогичные идентификаторам электронной почты. Все эти достоинства и постепенно исчезающая, к сожалению, простота протокола (текстовая база, парадигма запрос-ответ) привлекали и продолжают привлекать к SIP многих сторонников.

Поэтому в ближайшей перспективе от протокола SIP ожидается не только поддержка традиционных услуг коммутируемой телефонной сети общего пользования, но и организация передачи видеоинформации, чатов, интерактивных игр и виртуальной реальности.

В этой главе кратко обозначаются открытые проблемы SIP, среди которых уже упоминавшееся ранее использование SIP проектом 3GPP и преобразование сетевых адресов NAT.

16.2. SIP в мобильных сетях 3G

Подобно технологиям стационарных сетей связи, мобильные сети тоже не стояли на месте. Они прошли в своем развитии такой же путь: от аналоговых систем (первое поколение – 1G) через сети коммутации каналов (второе поколение – 2G) к сетям коммутации пакетов (3G). Существует несколько версии сетей 3G, однако наиболее распространенной и перспективной является версия, которую стандартизировала организация 3GPP (3rd Generation Partnership Project). На рис. 16.1. показана архитектура такой сети.

UTRAN (UMTS Terrestrial Radio Access Network – наземная сеть радиодоступа UMTS). Она отвечает за выполнение всех функций на радио-участке между мобильным телефоном и сетью связи. Концепция сетей 3G подразумевает, что ядро сети полностью изолировано от радиоинтерфейса, что дает свободу в выборе технологии, применяемой в UTRAN.

Сеть коммутации пакетов (PS CN – Packet Switch Core Network). В рекомендациях определяется, что на первом этапе в качестве базовой будет использоваться сеть GSM с технологией GPRS, поэтому в сети коммутации пакетов применяются устройства SGSN (Service GPRS Support Node) и GGSN (Gateway GPRS Support Node). SGSN выполняет маршрутизацию пакетов, управление логическими каналами, аутентификацию, шифрование и проверку по регистру IMEI, отвечает за управление мобильностью пользователя. GGSN назначает IP-адреса для мобильных терминалов и управляет качеством обслуживания.

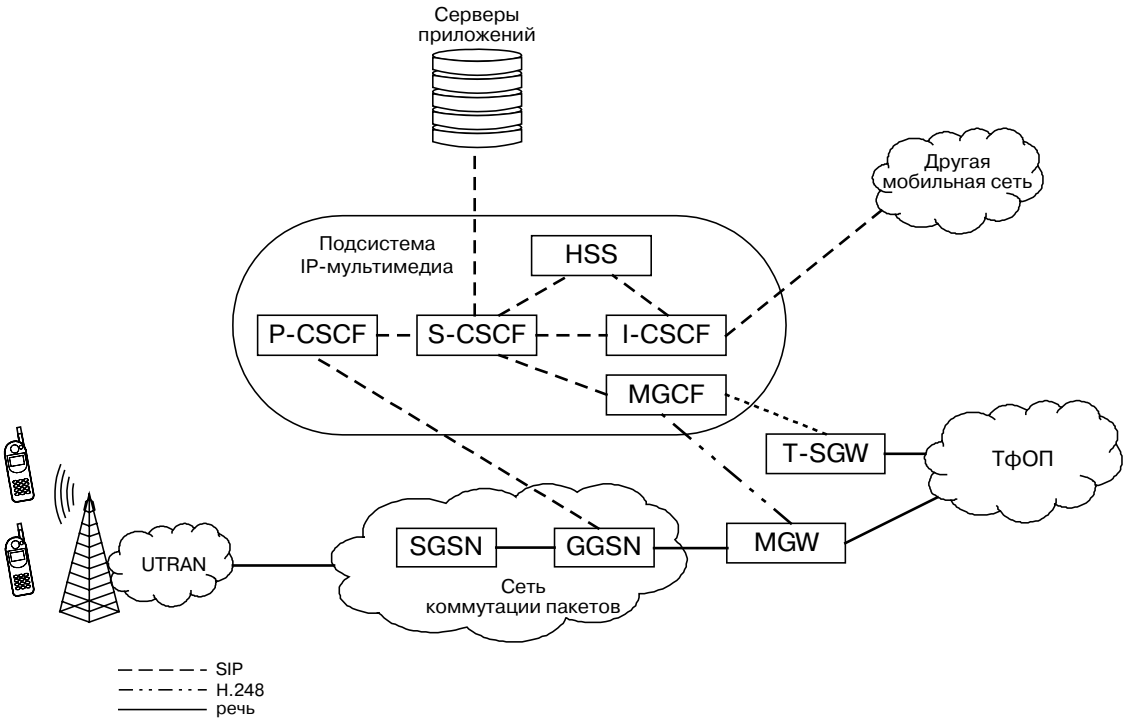


Рис. 16.1. Архитектура сети по рекомендациям 3GPP

С точки зрения сигнализации для нас наиболее важной является подсистема IP-мультимедиа (IMS–IP Multimedia Subsystem). IMS объединяет все элементы сети коммутации пакетов для предоставления мультимедийных услуг, включая речь, видео, текст, чат, и т.д. В IMS находятся устройства (функциональные блоки) CSCF (Call Session Control Function – функции управления обработкой вызовов). Существует три типа таких устройств:

- Proxy-CSCF (P-CSCF) является аналогом прокси-сервера в архитектуре SIP, т.е. принимает запросы и либо обслуживает их локально, либо перенаправляет. Все вызовы из PS CN сначала поступают на P-CSCF. Терминал узнает адрес P-CSCF во время процедуры регистрации. К основным функциям P-CSCF относятся:

  - Перенаправление SIP-запросов регистрации REGISTER от терминала пользователя на устройство I-CSCF.
  - Перенаправление SIP-запросов от терминала пользователя на устройство S-CSCF, и, при необходимости, модификация полей в этих запросах.
  - Перенаправление SIP-ответов и запросов на терминалы пользователей.
  - Генерация записей о вызовах (CDR-Call Detailed Record).
  - Управление обработкой вызовов к спецслужбам.
  - Перенаправление вызовов в домашнюю сеть при роуминге.
  - Управление качеством обслуживания и безопасностью.
- Interrogating-CSCF (I-CSCF) принимает вызовы в домашней сети (т.е. в той, где зарегистрирован пользователь) из гостевой (сеть, в которую переместился пользователь), т.е. является шлюзом на входе сети. Это необязательный элемент сети, так как P-CSCF может направлять запросы непосредственно на S-CSCF. Его основная функция – сокрытие структуры сети от внешних узлов. К его функциям также относятся:
  - Маршрутизация SIP-запросов из других сетей на S-CSCF.
  - Определение адреса S-CSCF по базе данных пользователей (HSS).
  - Генерация записей о вызовах.
- Serving-CSCF (S-CSCF) управляет обслуживанием вызовов пользователей в своей сети. Он хранит информацию обо всех вызовах для поддержки разнообразных услуг. В зависимости от требуемых функциональных возможностей и производительности сети в ней может быть несколько S-CSCF. Основные функции S-CSCF во время управления сеансами связи:
  - Функции сервера регистрации (registrar) (см. п. 4.6) и взаимодействие с базой данных пользователей (HSS).
  - Управление обработкой вызовов зарегистрированных пользователей.
  - Управление предоставлением дополнительных услуг, для чего может понадобиться взаимодействие с серверами приложения.
  - Поиск и маршрутизация к P-CSCF запросов и ответов в гостевой сети при роуминге.

Одним из больших достоинств использования CSCF является поддержка концепции Виртуального домашнего окружения (VHE-Virtual Home Environment), которая подразумевает, что управление услугами производится домашним S-CSCF. Это значит, что реализовывать услуги нужно только в домашней сети, а пользователи при роуминге получают к ним доступ в любой гостевой сети (даже в той, которая не поддерживает эти услуги).

Для хранения информации о пользователях в IMS организуется Домашний сервер пользователей (HSS-Home Subscriber Server). HSS хранит следующую информацию о пользователях:

- Идентификацию пользователя, его номер и адрес.
- Информацию для аутентификации и авторизации пользователя.
- Информацию о местоположении пользователя.
- Профили пользователей.

Для взаимодействия с ТфОП используются устройства трех типов:

– транспортный шлюз (MGW-Media Gateway) предназначен для преобразования речевой информации в вид, поддерживаемый ТфОП, для эхокомпенсации и др.;

– сигнальный шлюз (T-SGW-Transport Signaling Gateway) предназначен для упаковки сигнальных сообщений (например, ОКС7) в IP-пакеты, а также для обратного преобразования;

– функция управления транспортными шлюзами (MGCF-Media Gateway Control Function) управляет (по протоколу H.248) транспортными шлюзами. По сути, она преобразует сообщения протокола SIP в сообщения H.248.

Как видно из рис. 16.1, в качестве базового протокола установления соединения в сети 3G используется протокол SIP. Для иллюстрации его работы приведем пример регистрации пользователя в сети (рис. 16.2), а также установление исходящего и входящего соединения в домашней сети.

Как и в мобильных сетях второго поколения, в сетях 3G после включения мобильного терминала происходит процедура регистрации. Для этого терминал передает на P-CSCF (его адрес он узнает с помощью специальной процедуры поиска P-CSCF) уже хорошо известный запрос REGISTER (1).

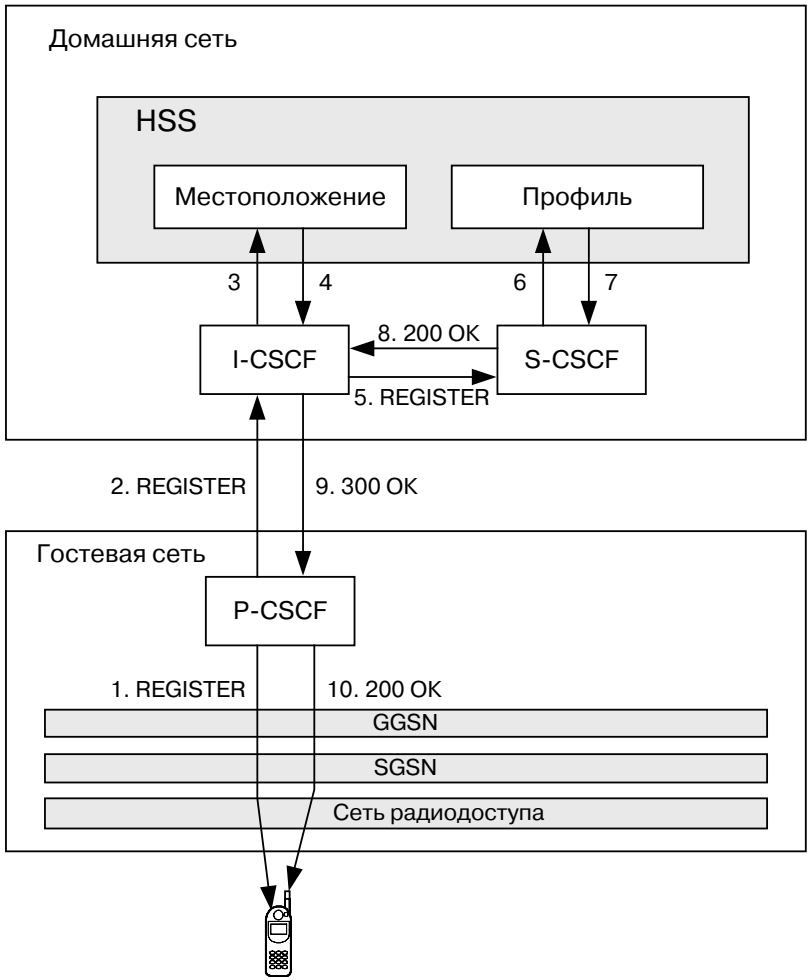


Рис. 16.2. Регистрация пользователя

На основании информации из Request-URI через DNS-запрос P-CSCF определяет адрес I-CSCF в домашней сети пользователя, после чего пересылает туда запрос REGISTER (2). Сервер I-CSCF передает информацию (3) о пользователе на HSS, который в ответ (4) сообщает адрес обслуживающего сервера S-CSCF, куда пересылается запрос REGISTER (3). S-CSCF регистрирует пользователя на HSS (6,7). Если все проходит успешно, пользователю передается ответ 200 OK (8, 9,10).

При исходящем вызове (рис. 16.3) после набора номера терминал пользователя отправляет запрос INVITE (1) на P-CSCF, адрес которого определяет механизм обнаружения CSCF. В запросе терминал указывает параметры сеанса (в SDP) для того, чтобы сеть могла проверить доступность ресурсов, необходимых пользователю. P-SCSF проверяет, имеет ли пользователь право на участие в сеансе связи, и перенаправляет запрос INVITE на S-CSCF (3). S-CSCF проверяет права пользователя и список разрешенных ему услуг. Кроме того, он удаляет все параметры сеансов, которыми пользователь не может пользоваться (например, видео). S-CSCF перенаправляет запрос (6) на соответствующий узел (например, на другой P-CSCF). В ответе 183 Session Progress (8-9) приходит информация от вызываемого пользователя о параметрах поддерживаемого им сеанса. На P-CSCF проверяется (10), доступны ли пользователю ресурсы, которые он запросил.

Оконечный терминал 1 выбирает, какие параметры сеанса будут использоваться. Он подтверждает прием информационного ответа с помощью запроса PRACK (12) и резервирует ресурсы в транспортной сети для передачи выбранных медиа-потоков (18).

В отличие от предыдущих, в этом сценарии используется процедура резервирования ресурсов сети. До того, как передать на вызывающую сторону ответ 180 Ringing, необходимо убедится, что ресурсы сети, которые требуются для сеанса, в данный момент доступны. Иначе может возникнуть ситуация, когда после снятия вызывающим пользователем трубки разговор будет невозможен ввиду загруженности сети, что создает определенное неудобство. Как видно из рис. 16.3, с помощью запроса INVITE и ответа 183 Session Progress обе стороны сообщают о параметрах сеанса, в котором они желают принять участие. Для окончательного подтверждения резервирования требуемых ресурсов используется запрос COMET. Ранее этот запрос не описывался, т.к. он упоминается только в черновиках, и в финальных версиях стандартов его пока нет.

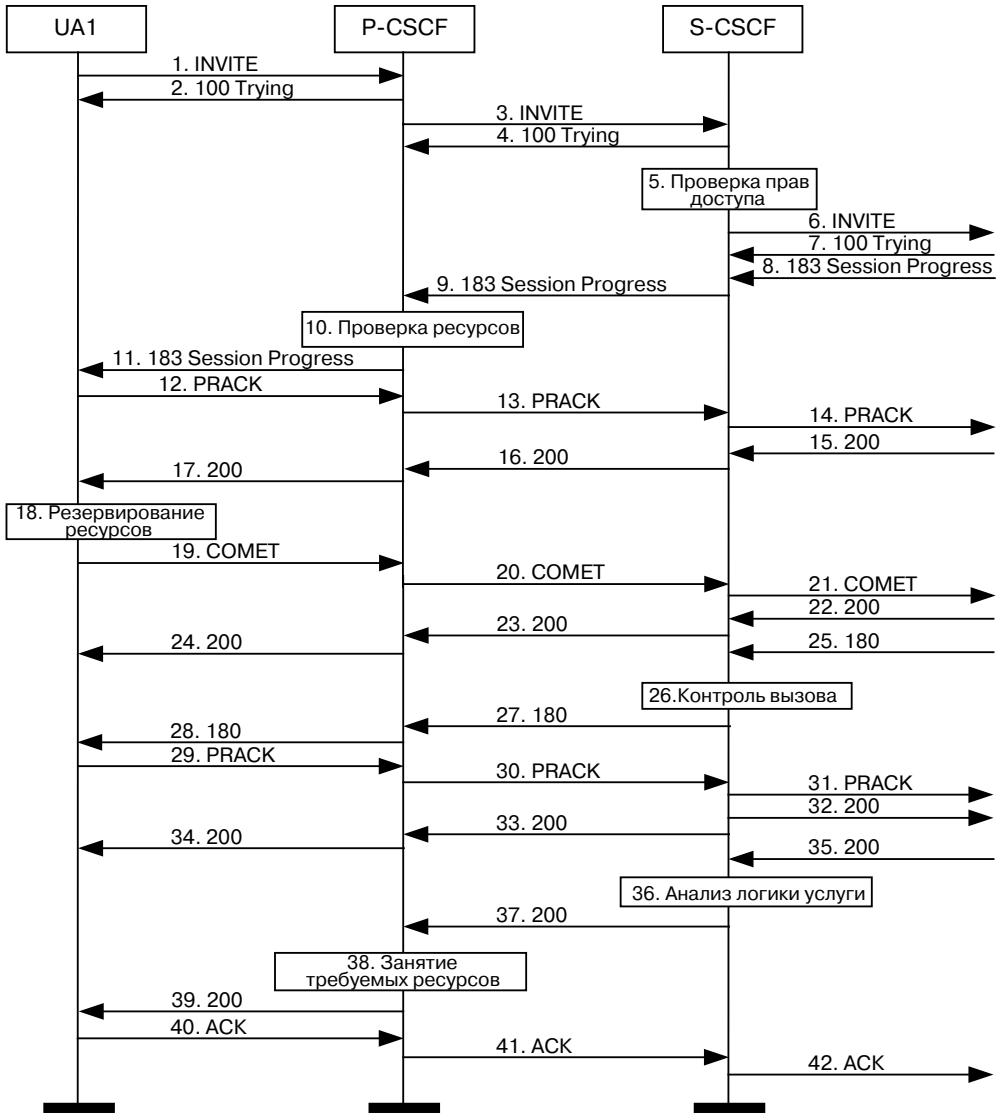


Рис. 16.3. Исходящий вызов с мобильного телефона

В RFC 3312 «Integration of Resource Management and SIP» [5] вместо запроса COMET используется запрос UPDATE, описание которого можно найти в предыдущих разделах справочника. Поэтому, вероятнее всего, в будущих стандартах организации 3GPP для извещения об успешном резервировании ресурсов будет использован запрос UPDATE.

После приема запроса COMET терминал вызываемого пользователя информирует пользователя о входящем вызове и генерирует ответ 180 Ringing (25, 27, 28), который подтверждается запросом PRACK (29-31). Вызываемый пользователь поднимает трубку, и его терминал передает ответ 200 OK (35). После окончательного занятия ресурсов пользователи могут разговаривать между собой (или видеть друг друга, если сеанс связи предусматривает передачу видео-потоков).

На рис. 16.4 показан сценарий входящего вызова на мобильный телефон. Как видно, он зеркально похож на предыдущий сценарий. На входящей стороне тоже резервируются ресурсы сети для того, чтобы качество обслуживания пользователей было на высоком уровне.

Как упоминалось в начале справочника, протокол SIP обладает свойством персональной мобильности, суть которой состоит в том, что пользователь имеет единый номер, а сеть по этому номеру «находит» его, независимо от того, где он находится. Это реализуется за счет регистрации пользователя при смене своего физического подключения к сети. По сути, этот принцип в полной мере применим к мобильным сетям. При переезде пользователя в другую сеть при установлении IP-соединения его терминал (1) регистрируется в новой сети, отправляя запрос REGISTER (1) на P-CSCF. Последний на основе информации из запроса определяет, какой сети принадлежит пользователь, и пересылает туда (на I-CSCF) запрос REGISTER (2). Сервер I-CSCF запрашивает в базе данных абонентов HSS адрес сервера S-CSCF, обслуживающего этого абонента (3,4). Далее запрос REGISTER пересылается на S-CSCF (5). Сервер S-CSCF регистрирует на HSS новое местоположение своего абонента (6,7).

Как видно из материала этого параграфа, мобильные сети третьего поколения эффективно используют преимущества, которые дает протокол SIP, а именно: работу с мультимедийными сеансами, принцип персональной мобильности, гибкую интеграцию с IP-сетями и др. Организация 3GPP широко использует результаты работы IETF в этом направлении, что дает надежду на то, что взаимодействие между стационарными и мобильными сетями нового поколения (NGN) будет простым.

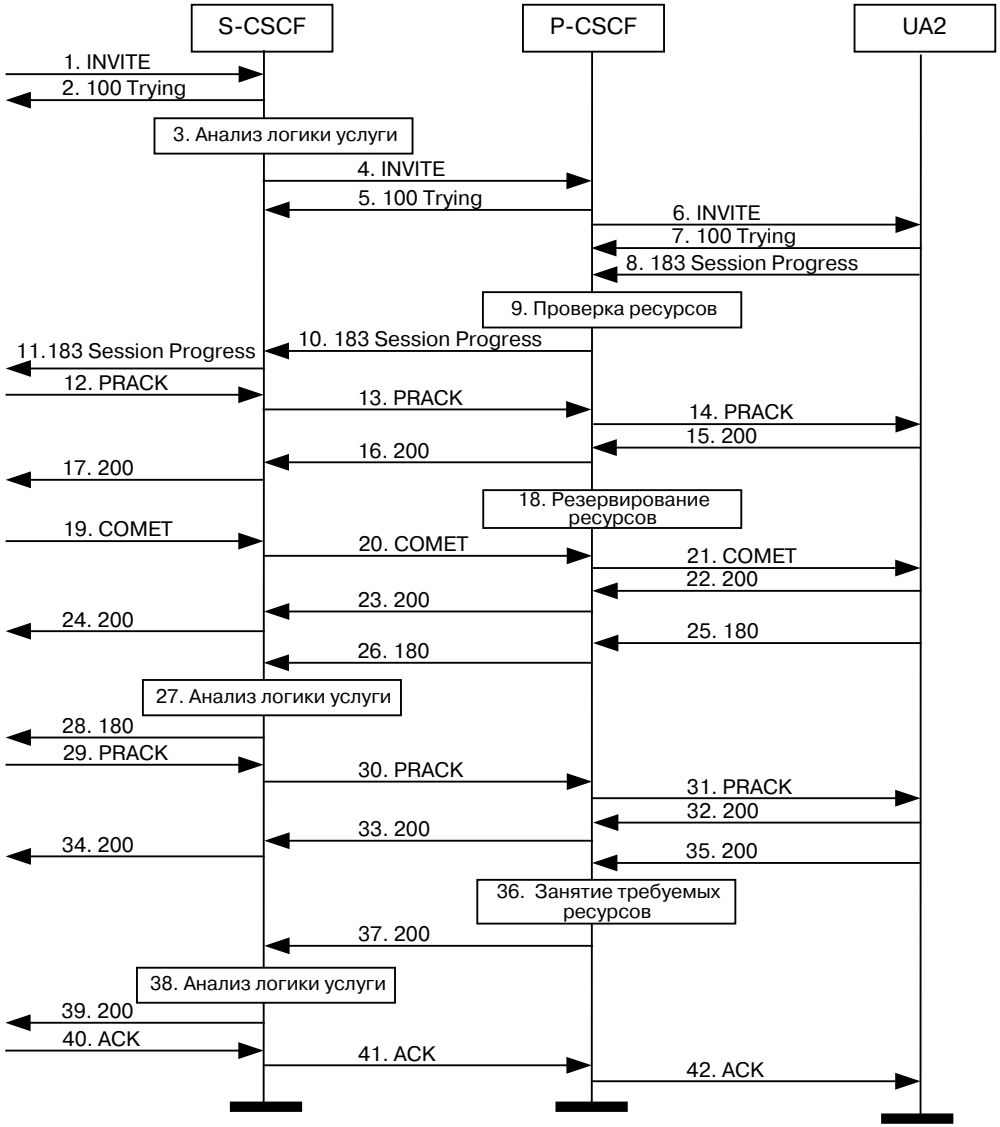


Рис. 16.4. Входящий вызов на мобильный телефон

16.3. Работа с NAT

В последние годы NAT (Network Address Translator – преобразователь сетевых адресов) получил широкое признание как промежуточное решение в целях сокращения незанятого адресного пространства IPv4 и масштабирования в маршрутизации. NAT сохраняет пространство адресов IPv4, позволяя использовать глобальные IP-адреса совместно несколькими хостами. Для этого NAT заменяет заголовки IP-пакетов, что затрудняет (или делает невозможной) работу с NAT многих прикладных протоколов, таких как SIP и H.323. В условиях повсеместного распространения NAT протокол SIP и создаваемые с его помощью медиа-потоки должны быть способны преодолевать NAT, чтобы SIP мог активно использоваться в сети Интернет.

Проблема прохождения через NAT и брандмауэр является для SIP сложной, так как в сообщениях SIP передаются IP-адреса или имена серверов имен доменов (DNS), а также информация о номерах портов для медиа-потоков. NAT не способен соотносить сообщения протокола SIP с медиа-сеансами, для которых сообщения передаются. Таким образом, решить эту проблему необходимо как для сигнализации, так и для медиа-потоков. Отсутствие логически последовательного решения проблемы отчасти обусловлено множеством предлагаемых решений:

- Протокол Simple Traversal of UDP Through NAT (STUN);
- Traversal using relay NAT (TURN);
- SIP Application Layer Gateways (ALGs);
- Протокол Middlebox Communication (MIDCOM);
- SIP Symmetric Response Routing, RFC 3581 [53];
- Управляемые протоколом Media Gateway Control Protocol (MGCP) шлюзы;
- Virtual private networks (VPNs);
- Firewall Enhancement Protocol, RFC 3093 [21];
- Universal Plug and Play (UPnP);
- Interactive Connectivity Establishment (ICE).

SIP ALGs – это, по-видимому, наиболее легко внедряемые технически и наиболее надежные решения. SIP ALGs являются, по существу, SIP B2BUA, которые

обрабатывают каждое сообщение SIP и подставляют соответствующие адреса. Они не требуют каких-либо расширений протоколов SIP или SDP. Ясно, что поддержка в NAT протокола SIP эффективна, но из-за необходимости реализации полного стека SIP, который требуется для функционирования B2BUA, SIP ALGs вносят задержки, и вызывают вопросы в отношении пропускной способности NAT. Кроме того, существует огромное количество функционирующих NAT, в которых отсутствуют функциональные возможности SIP ALG.

Решение MIDCOM подобно решению ALG, за исключением того, что в MIDCOM интеллектуальные возможности протокола уровня приложений встроены не в NAT/брандмауэр, а в агент MIDCOM, который динамически управляет всеми NAT, используемыми MIDCOM, чтобы создавать каналы для сигнализации и для медиа-потоков. Основное достоинство этого решения состоит в том, что, будучи используемыми MIDCOM, NAT/брандмауэры не нуждаются в постоянном обновлении для поддержки разных протоколов уровня приложений. Однако обновление существующих NAT для поддержки MIDCOM – трудная задача.

Большинство прочих решений для прохождения NAT использует общую идею, суть которой состоит в попытке расширения функций NAT и дополнения его другими протоколами, позволяющими SIP UA создавать на NAT IP-адрес/порт общего пользования. Это позволяет SIP UA устанавливать сеансы с использованием маршрутизируемых адресов и портов общего пользования и гарантировать сквозную передачу информации. На запрос SIP UA об IP-адресе и порте общего пользования может ответить либо сам NAT, либо сервер, функционирующий в сети общего пользования.

Протокол STUN позволяет SIP-клиенту определить, используется ли NAT или нет, а также узнать тип NAT. Клиент с возможностями STUN передает запрос на внешний сервер STUN с целью определить параметры NAT. Сервер STUN сообщает клиенту глобальный IP-адрес и номера портов, которые он использует. Клиент может использовать эту информацию для создания запроса SIP. Сервер STUN не участвует в сигнализации или в передаче медиа-потоков. Протокол STUN не работает с NAT наиболее общего типа – симметричным NAT, – что является самым большим его недостатком.

Протокол TURN решает проблему прохождения медиа-информации через симметричные NAT. TURN использует сервер, который находится на пути медиа-потоков. Сервер TURN расположен либо в сети пользователя, либо в сети

провайдера услуг. Клиент SIP с возможностями TURN передает запрос на сервер TURN. В ответ сервер сообщает IP-адрес общего пользования и номер порта, которые должны применяться для данного сеанса. Эта информация используется в сообщениях протокола SIP и в последующих медиа-потоках. Достоинство этого метода в том, что в адрес пункта назначения, видимый преобразователем NAT, не вносятся изменения, поэтому можно использовать симметричные NAT. Однако TURN, как и STUN, требует обновления существующих агентов SIP UA.

Как уже говорилось, проблема прохождения NAT для SIP – двойная проблема, включающая в себя прохождение самой сигнализации SIP и прохождение медиа-информации протокола RTP. В документе [53] рассматривается решение для прохождения NAT протоколом SIP поверх UDP с использованием техники под названием «симметричная маршрутизация ответов». Она включает в себя новый параметр для поля заголовка **Via**, называемый rport, который позволяет клиенту запрашивать передачу ответа из сервера обратно по IP-адресу и номеру порта источника, откуда поступил запрос, вместо использования номера порта в поле самого верхнего заголовка **Via** запроса.

Упоминавшееся решение ICE явилось попыткой определить общую методологию, которая строится на достоинствах предыдущих частных решений и обеспечивает единое решение проблемы функционирования сети на базе протокола SIP при использовании NAT. ICE не предлагает каких-либо расширений STUN и TURN, но расширяет SDP (добавляет специальный атрибут), чтобы решить проблему прохождения медиа-информации в RTP.

NAT считаются краткосрочным решением для сохранения адресного пространства IPv4 путем совместного использования IP-адресов общего пользования, но они породили серьезные проблемы для SIP. Появилось несколько решений, позволяющих SIP проходить через NAT. Каждое из них подходит для определенной топологии сети, но унифицированное решение, которое удовлетворяет всем потребностям, отсутствует. Архитектура ICE пытается сопоставить несопоставимые решения в единой методологии, но описывающий методологию документ пока далек от статуса стандарта. NAT будут нужны вплоть до полного перехода на IPv6, которое должно освободить Интернет от NAT и от связанных с этим проблем.

Список сокращений

3GPP, 3rd Generation Partnership Project, – совместный проект разработчиков оборудования сетей 3-го поколения

A

AAL, ATM Adaptation Layer, – уровень адаптации ATM

ANSI, American National Standards Institute, – институт американских национальных стандартов

ATM, Asynchronous Transfer Mode, – асинхронный режим переноса информации

B

BCI, Backward Call Indicators, – обратные индикаторы условий обслуживания вызовов. Параметр сообщений ISUP

BNF, Backus-Naur Form, – форма Бэкуса-Наура

C

CCF, Charging Collection Function, – функция накопления данных о начисленной плате

CIC, Carrier Identification Code, – идентификатор оператора (сети)

CIN, Calling Party's Number, – номер вызывающего абонента. Параметр сообщений ISUP

CON, Connect Message, – сообщение ISUP

CPG, Call ProGress, – сообщение ISUP

CPN, Called Party Number, – номер вызываемого абонента. Параметр сообщений ISUP

CPS, Called Party Status, – статус вызываемой стороны. Параметр сообщений ISUP

CSCF, Call Session Control Function, – функции управления обслуживанием вызовов

D

DNS, Domain Name Service, – протокол поддержки (служба) доменовых имен

DoS, Denial of Service, – отказ в обслуживании (один из типов угроз или атак)

DSP, Digital Signal Processor, – процессор цифровой обработки сигналов

DTMF, Dual-Tone MultiFrequency signaling, – многочастотная сигнализация кодом «2 из n»

E

ECF, Event Charging Function, – функция разового начисления платы

ETS, European Telecommunications Standard, – европейский телекоммуникационный стандарт

ETSI, European Telecommunications Standards Institute, – институт европейских телекоммуникационных стандартов

F

FCI, Forward Call Indicators, – прямые индикаторы условий обслуживания вызовов. Параметр сообщений ISUP

FQDN, Fully Qualified Domain Name, – полностью определенное имя домена

G

GAP, Generic Address Parameter, – параметр сообщений ISUP

GGSN, Gateway GPRS Support Node, – узел GPRS, поддерживающий шлюзы

GMT, Greenwich Mean Time, – время по Гринвичу

GSM, Global System for Mobile communications, – глобальная система мобильной связи

GPRS, General Packet Radio Service, – технология пакетной передачи данных в сетях GSM

GSTN, Global Switched Telephone Network, – всемирная коммутируемая телефонная сеть

GW, Gateway, – шлюз

H

HSS, Home Subscriber Server, – сервер пользователей домашней сети (база данных)

HTTP, Hypertext Transfer Protocol, – протокол переноса гипертекста (в Интернет)

I

ICID, IMS Charging Identity, – идентификатор диалога или транзакции для начисления платы

ICMP, Internet Control Message Protocol, – протокол обмена контрольными сообщениями в сети Интернет

ICW, Internet Call Waiting, – услуга, информирующая о входящем телефонном вызове пользователя, который занят в сеансе модемной связи с Интернет

IETF, Internet Engineering Task Force, – комитет инженерных задач Интернет

IM, Instant Messaging, – интерактивный обмен текстовыми сообщениями между несколькими участниками связи в режиме, близком к реальному времени

IMEI, International Mobile Equipment Identity, – международный идентификатор мобильного оборудования

IMS, IP Multimedia Subsystem, – подсистема поддержки мультимедийной IP-связи

IN, Intelligent Network, – Интеллектуальная сеть

IOI, Inter Operator Identifiers, –идентификаторы сетей для расчетов между операторами

IP, Internet Protocol, – протокол Интернет (уровень 3 OSI)

IPSec, IP Security, – технология обеспечения сетей IP средствами безопасности

ISDN, Integrated Services Digital Network, – цифровая сеть интегрального обслуживания

ISO, International Standardization Organization, – международная организация стандартов

ISUP, ISDN User Part, – подсистема-пользователь MTP (система ОКС7), поддерживающая сигнализацию ISDN

ITU, International Telecommunications Union, – международный союз электросвязи

ITU-T, Telecommunication Standardization Sector, – сектор стандартизации в составе ITU

IVR, Interactive Voice Response, – интерактивная система речевых ответов

L

LRN, Locating Routing Number, – номер для маршрутизации к адресату

M

MEGACO, MEdia GAteway COntrol – протокол управления транспортными шлюзами и название рабочей группы IETF, занимающейся его стандартизацией

MG, Media Gateway, – транспортный шлюз (медиашлюз)

MGC, Media Gateway Controller, – контроллер транспортного шлюза. Может также выполнять функции шлюза сигнализации

MGCF, Media Gateway Control Function, – функции управления транспортными шлюзами

MGCP, Media Gateway Control Protocol, – протокол управления транспортными шлюзами

MIDCOM, Middlebox Communication, – протокол связи через промежуточный сервер

MIME, Multipurpose Internet Mail Extension, – многоцелевое расширение Интернет-почты

MTU, Maximum Transmission Unit, – максимально допустимая для передачи длина пакета

N

NAPTR, Naming Authority PoinTeR, – процедура преобразования номера в формат SIP URL на DNS сервере

NAT, Network Address Translator, – преобразователь сетевых адресов

NCI, Nature of Connection Indicator, – параметр сообщений ISUP

NGN, Next Generation Network, – сеть связи следующего поколения

NGW, Network Gateway, – шлюз

O

OCN, Original Called Number, – параметр сообщений ISUP

OSI, Open Systems Interconnection, – взаимодействие открытых систем

P

PDD, PostDial Delay, – задержка после набора номера

PGP, Pretty Good Privacy, – система защиты информации

PINT, PSTN/Internet Inter-Networking, – рабочая группа комитета IETF «Взаимодействие сетей ТфОП/Интернет»

PPP, Point-to-Point Protocol, – протокол «точка – точка»

PSCN, Packet Switch Core Network, – опорная сеть коммутации пакетов

PSTN, Public Switched Telephone Network, – коммутируемая телефонная сеть общего пользования

Q

Qop, Quality of protection, – качество защиты

QoS, Quality of Service, – качество обслуживания, качество услуг(и)

R

RFC – серия стандартов и рекомендаций организации IETF

RSVP, resource ReSerVation Protocol, – протокол резервирования ресурсов

RTCP, Real-Time Control Protocol, – протокол управления передачей в реальном времени

RTP, Real-Time Protocol, – протокол передачи информации в реальном времени

RTT, Round-Trip Time, – длительность цикла транзакции (с момента передачи запроса до момента получения подтверждения его приема)

S

SCP, Service Control Point, – узел управления услугами (Интеллектуальной сети)

SCTP, Stream Control Transport Protocol, – транспортный протокол с управлением потоком

SDL, Specification and Description Language, – язык спецификации и описания

SDP, Session Description Protocol, – протокол описания сеансов связи

SGSN, Service GPRS Support Node, – узел GPRS, поддерживающий услуги

SIGTRAN, SIGnaling TRANsport – технология передачи сигнальных сообщений по IP-сетям

SIP, Session Initiation Protocol, – протокол инициирования сеансов связи

SIP ALG, SIP Application Layer Gateway, – шлюз прикладного уровня SIP

SIP IAD, SIP Integrated Access Device – интегрированное устройство доступа, работающее по протоколу SIP

SIPS URI, SIP Secured URI, – схема, гарантирующая использование только тех ресурсов, которые обеспечены средствами защиты

SIP-T, SIP extension for Telephony, – описание SIP для взаимодействия с ТфОП

S/MIME, Security for MIME, – система защиты заголовков и MIME-тел сообщений SIP

SMTP, Simple Mail Transfer Protocol, – простой протокол доставки почты

SN, Service Node, – узел услуг (Интеллектуальной сети)

SPIRITS, Service in the PSTN/IN Requesting InTernet Service, – обслуживание запросов услуг Интернет со стороны PSTN/IN

SSL, Secure Sockets Layer – уровень защиты сокетов

SSP, Service Switching Point, – узел коммутации услуг (Интеллектуальной сети)

STUN, Simple Traversal of UDP through NAT, – протокол простой передачи UDP через NAT

T

TCP, Transmission Control Protocol, – протокол управления передачей (уровень 4 OSI)

TLS, Transport Layer Security, – средства обеспечения безопасности на транспортном уровне (поверх протокола уровня 3 с созданием виртуального соединения)

TMR, Transmission Medium Requirement, – параметр сообщений ISUP

TNS, Transit Network Selection, – параметр сообщений ISUP

T-SGW, Transport Signaling GateWay, – сигнальный шлюз

TTL, Time-To-Live, – допустимая продолжительность существования (пакета) в сети

TU, Transaction User, – пользователь транзакциями

TURN, Traversal Using Relay NAT, – проход через NAT с помощью сервера-ретранслятора

U

UA, User Agent, – агент пользователя

UAC, User Agent Client, – клиент агента пользователя

UAS, User Agent Server, – сервер агента пользователя

UDP, User Datagram Protocol, – протокол транспортировки дейтаграмм пользователя

UMTS, Universal Mobile Telecommunications System, – универсальная система мобильных телекоммуникаций (один из стандартов 3G, разрабатываемый ETSI)

URI, Universal Resource Identificator, – универсальный идентификатор ресурса

URL, Universal Resource Locator, – универсальный указатель местоположения ресурса

UTP, Unshielded Twisted Pair, – не экранированная витая пара

UTRAN, UMTS Terrestrial Radio Access Network, – наземная сеть радиодоступа UMTS

V

VHE, Virtual Home Environment, – виртуальное домашнее окружение

VPN, Virtual Private Network, – виртуальная частная сеть

Глоссарий

Алгоритм хэширования

Хэширующий алгоритм MD5 используется, в частности, для создания цифровых подписей, позволяющих однозначно идентифицировать отправителя. MD5 является алгоритмом «одностороннего» хэширования; это означает, что данные, хэшируемые функцией шифрования, восстановить уже невозможно.

Алгоритм MD5 применяется при проверке паролей. Поскольку теоретически невозможно восстановить исходную строку, обработанную алгоритмом хэширования, можно хэшировать пароль функцией md5 и затем сравнить зашифрованный пароль с результатом обработки пароля, введенного пользователем при попытке получить доступ.

Диалог

Диалог представляет собой равноправное взаимодействие двух агентов пользователя по протоколу SIP в течение некоторого времени. Диалог устанавливает последовательность сообщений между UA и обеспечивает верную маршрутизацию запросов.

Диалог на ранней стадии

Равноправное взаимодействие двух агентов пользователя на стадии организации диалога, устанавливаемое в результате передачи предварительного ответа. Прекращается после поступления окончательного ответа.

Информация answer

Ответ на предложение с описанием сеанса связи в формате SDP, передаваемый в теле сообщения SIP и содержащий принятые параметры сеанса связи.

Информация offer

Предложение с описанием сеанса связи в формате SDP, передаваемое в теле сообщения SIP одним из участников организуемого сеанса связи.

Параметр «q»

Параметр «q» определяет приоритеты среди значений, содержащихся в заголовке, путем присвоения этому параметру значения в пределах от 0 до 1.

Сеанс связи

Совокупность участников связи и медиа-потоков, обеспечивающих обмен информацией между ними.

Event package

Идентификатор функциональных возможностей, которые позволяют информировать клиента (подписчика) о событиях, происходящих в указанном ресурсе.

Interworking indicator

Индикатор наличия взаимодействия, поле параметра FCI в сообщениях ISUP.

Option-tag

Уникальный идентификатор новых функциональных возможностей SIP в соответствии с его расширениями, определенными в дополнительных RFC.

Path MTU

Path Maximum Transfer Unit. Максимально допустимый размер пакета, для доставки которого от источника к получателю ни на одном участке пути не потребуется его фрагментация. Path MTU имеет значение наименьшей из MTU на всех участках пути.

Registrar

Регистрирующий сервер в сети SIP. Предназначен для создания, изменения и удаления из базы данных информации, указывающей текущее местоположение пользователя.

Remote target

Текущий адрес удаленного пользователя. Является одним из компонентов, описывающих состояние диалога.

Response context

Буфер ответов SIP. Предназначен для сбора, хранения и последующего выбора и передачи наиболее подходящего ответа.

Route set

Конфигурация маршрута – перечень адресов серверов, через которые должен пройти запрос, переданный второму участнику диалога. Является одним из компонентов, описывающих состояние диалога.

Stateful прокси-сервер

Прокси-сервер с сохранением состояний. Хранит информацию (состояние транзакции) о каждом поступившем сообщении и о каждом сообщении, переданном в результате обработки входящего запроса.

Stateless прокси-сервер

Прокси-сервер без сохранения состояний. Работает как ретранслирующий узел сети. Он пересылает каждый запрос следующему элементу, принимая решение о маршрутизации на основе информации, содержащейся в запросе. Полученные ответы он просто передает обратно. Прокси-сервер без сохранения состояний удаляет информацию о поступившем сообщении, как только оно было ретранслировано.

Strict router

Прокси-сервер, стирающий содержимое поля Request-URI при наличии в запросе заголовка Route.

Литература

1. Arkko J., Torvinen V., Camarillo G., Niemi A., Haukka T. Security Mechanism Agreement for SIP. RFC 3329, January 2003.
2. Arlein R., Gurbani V. An Extensible Framework for Constructing Session Initiation Protocol User Agents. Bell Labs Tech. J., 9:3 (2004), 87-100.
3. Braden R. Requirements for Internet Hosts – Application and Support. RFC 1123, October 1989.
4. Braden R., Zhang L., Berson S., Herzog S., Jamin S. Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification. RFC 2205, September 1997.
5. Camarillo G., Marshall W., Rosenberg J. Integration of Resource Management and SIP. RFC 3312, October 2002.
6. Camarillo G., Peterson J., Ong L., Roach A.B. Integrated Services Digital Network (ISDN) User Part (ISUP) to Session Initiation Protocol (SIP) Mapping. RFC 3398, December 2002.
7. Campbell B., Rosenberg J., Schulzrinne H., Huitema C., Gurle D. SIP Extension for Instant Messaging. RFC 3428, December 2002.
8. Crocker D. Standard for the format of ARPA internet text messages. RFC 822, August 1982.
9. Cuervo F., Greene N., Huitema C., Rayhan A., Rosen B., Segers J. Megaco Protocol version 0.8. RFC 2885, August 2000.
10. Dawson F., Howes T. vCard MIME Directory Profile. RFC 2426, September 1998.
11. Donovan S. The SIP INFO Method. IETF RFC 2976, October 2000.
12. Eastlake D., Crocker S., Schiller J. Randomness Recommendations for Security. RFC 1750, December 1994.
13. Egevang K., Francis P. The IP Network Address Translator (NAT). RFC 1631, May 1994.
14. Faltstrom P. E.164 number and DNS. RFC 2916, September 2000.
15. Foster M., McGarry T., Yu J. Number Portability in the Global Switched Telephone Network (GSTN): An Overview. RFC 3482, February 2003.
16. Franks J., Hallam-Baker P., Hostetler J., Lawrence S., Leach P., Luotonen A., Stewart L. HTTP Authentication: Basic and Digest Access Authentication. RFC 2617, June 1999.
17. Franks J., Hallam-Baker P., Hostetler J., Leach P., Luotonen A., Sink E., Stewart L. An Extension to HTTP : Digest Access Authentication. RFC 2069, January 1997.

18. Freed N., Borenstein N. Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. RFC 2046, November 1996.

19. Galvin J., Murphy S., Crocker S., Freed N. Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted. RFC 1847, October 1995.

20. Garcia-Martin M., Henrikson E., Mills D. Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the 3rd Generation Partnership Project (3GPP). RFC 3455, January 2003.

21. Gaynor M., Bradner S. Firewall Enhancement Protocol. RFC 3093, April 2001.

22. Гольдштейн Б.С., Ехриель И.М., Рерле Р.Д. Интеллектуальные сети. — М.: Радио и связь, 2000.

23. Гольдштейн А.Б., Саморезов В.В. Методические рекомендации к лабораторным работам по IP-телефонии // СПбГУТ. СПб, 2002.

24. Гольдштейн Б.С., Пинчук А.В., Суховицкий А.Л. IP-телефония. М.: Радио и связь, 2001.

25. Гольдштейн Б.С. Протоколы сети доступа. Том 2. 2-е изд. М.: Радио и связь, 2001.

26. Гольдштейн Б.С. Сигнализация в сетях связи. Том 1. 3-е изд. М.: Радио и связь, 2001.

27. Гольдштейн Б.С., Ехриель И.М., Рерле Р.Д. Стек протоколов ОКС7. Подсистема ISUP. Справочник// СПб.: BHV-2003.

28. Гольдштейн Б.С., Ехриель И.М., Рерле Р.Д. Стек протоколов ОКС7. Подсистема MTP. Справочник// СПб.: BHV-2003.

29. Гольдштейн Б.С., Ехриель И.М., Кадыков В.Б., Рерле Р.Д. Протоколы V5.1 и V5.2. Справочник// СПб.: BHV-2003.

30. Гольдштейн Б.С., Сибирякова Н.Г., Соколов А.В. Сигнализация R1.5. Справочник// СПб.: BHV-2004.

31. Good G. The LDAP Data Interchange Format (LDIF) – Technical Specification. RFC 2849, June 2000.

32. Greene N., Ramalho M., Rosen B. Media Gateway Control Protocol Architecture and Requirements. RFC 2805, April 2000.

33. Groves C., Pantaleo M., Anderson T., Taylor T. Gateway Control Protocol Version 1. RFC 3525, June 2003.

34. Gurbani V., Chiang T. – C., Reid J. The Need for Third-Party Call Control. Bell Labs Tech. J., 7:1 (2002), 41-46.

35. Gurbani V., Liu K. Session Initiation Protocol: Service Residency and Resiliency. Bell Labs Tech. J., 8:1 (2003), 83-94.

36. Hadley M., Schulzrinne H., Scholler E., Rosenberg J. SIP: Session Initiation Protocol. IETF RFC 2543, March 1999.

37. Handley M., Jacobson V. SDP: Session Description Protocol. RFC 2327, April 1998.

38. Hilt V., Hofmann M. Approaches to implementing Services in the SIP Networks. Bell Labs Tech. J., 9:3 (2004), 39-44.

39. Housley R. Cryptographic Message Syntax. RFC 2360, June 1999.

40. ITU-T Recommendation H.248.1, Gateway Control Protocol: Version 2, May 2002.

41. ITU-T Recommendation H.323. Packet-based Multimedia Communications Systems, July 2003.

42. ITU-T Recommendation Q.761–Q.764, Signaling System №7 – ISDN User Part Functional Description, September 1997.

43. Johnston J., Sparks R., Cunningham C., Summers K. SIP Basic Call Flow Examples. RFC 3665, December 2003.

44. Marshall W. Private SIP Extensions for Media Authorization. RFC 3313, January 2003.

45. Marshall W., Andreasen F. Private SIP Proxy-to-Proxy Extensions for Supporting the Packet Cable Distributed Call Signaling Architecture. RFC 3603, October 2003.

46. Modarressi Abdi R., Mohan Seshadri. Control and Management in Next-Generation Networks: Challenges and Opportunities, IEEE Communications, 94–102.

47. Morneault K., Dantu R., Sidebottom G., Bidulock B., Heitz J. Signaling System 7 (SS7) Message Transfer Part 2 (MTP2) – User Adaptation Layer. RFC 3331, September 2002.

48. Ramsdell B. S/MIME Version 3 Message Specification. RFC 2633, June 1999.

49. Rivest R. The MD5 Message-Digest Algorithm. RFC 1321, April 1992.

50. Roach B. SIP Specific Event Notification. RFC 3265, August 2002.

51. Rosenberg J. SIP UPDATE Method. RFC 3311, September 2002.

52. Rosenberg J., Peterson J., Schuzlrinne H., Camarillo G. Best Current Practices for Third Party Call Control in the Session Initiation Protocol (SIP). IETF RFC 3725, April 2004.

53. Rosenberg J., Schulzrinne H. An Extension to the Session Initiation Protocol (SIP) for Symmetric Response Routing. IETC RFC 3581, August 2003.

54. Rosenberg J., Schulzrinne H. An Offer/Answer Model with SDP, RFC 3264, June 2002.

55. Rosenberg J., Schulzrinne H. Reliability of Provisional Responses in SIP. RFC 3262, June 2002.

56. Rosenberg J., Schulzrinne H. SIP: Locating SIP Servers. RFC 3263, June 2002.

57. Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J., Sparks R., Handley M., Schooler E. SIP: Session Initiation Protocol. RFC 3261, June 2002.

58. Rosenberg J., Weinberger J., Huitema C., Mahy R. STUN—Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). RFC 3489, March 2003.

59. Шнепс-Шнеппе М.А. Интернет-телефония: протокол SIP и его применения. М.: Макс Пресс, 2002.

60. Шнепс М.А. Системы распределения информации. Методы расчета. – М.: Связь, 1979.

61. Schulzrinne H., Casner S., Frederick R., Jacobson V. RTP: A Transport Protocol for Real-Time Applications. RFC 1889, January 1996.

62. Schulzrinne H., Casner S., Frederick R., Jacobson V. RTP: A Transport for Real-Time Applications, RFC 3550, July 2003.

63. Schulzrinne H., Petrack S. RTP payload for DTMF Digits, Telephony Tones and Telephony Signals, RFC 2833, May 2000.

64. Schulzrinne H., Rao A., Lanphier R., Real Time Streaming Protocol (RTSP), RFC 2326, April 1998.

65. Schulzrinne, Oran D., Camarillo G. The Reason Header Field for SIP" RFC 3326, December 2002.

66. Sidebottom G., Morneault K., Pastor-Balbas J. Signaling System 7 (SS7) Message Transfer Part 3 (MTP3) – User Adaptation Layer (M3UA). RFC 3332, September 2002.

67. Sparks R. Internet Media Type message/sipfrag. RFC 3420, November 2002.

68. Sparks R. SIP Refer Method. RFC 3515, April 2003.

69. Stewart R., Xie Q., Morneault K., Sharp C., Schwarzbauer H., Taylor T., Rytina I., Kalla M., Zhang L., Paxson V. Stream Control Transmission Protocol. RFC 2960, October 2000.

70. Зарубин А.А. Современные инфокоммуникационные услуги и концепция Open System Access // СПбГУТ, СПб.: 2003.

71. Technical Specification. 3rd Generation Partnership Project. 3GPP TS 23.002. Network architecture. (Release 5).

72. Technical Specification. 3rd Generation Partnership Project. 3GPP TS 23.228. IP multimedia subsystem, Stage 2.

73. Technical Specification. 3rd Generation Partnership Project. 3GPP TS 24.228. Signalling flows for the IP multimedia call control based,on SIP and SDP; Stage 3 (Release 5).

74. Technical Specification. 3rd Generation Partnership Project. 3GPP TS 24.229. IP multimedia call control protocol based on SIP and SDP.

75. Vaha-Sipila A. URLs for Telephone Calls. RFC 2806, April 2000.

76. Willis D., Hoeneisen B. Session Initiation Protocol (SIP) Extension Header Field for Registering Non-Adjacent Contacts, IETF RFC 3327, December 2002.

77. Yergeau F. UTF-8, a transformation format of ISO 10646. RFC 2279, January 1998.