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

Протокол SIP. Справочник

Серия: Телекоммуникационные протоколы ВСС РФ
Издательство: БХВ - Санкт-Петербург, 2005 г.
Мягкая обложка, 456 стр.
ISBN 5-8206-0123-8
Тираж: 3000 экз.
Формат: 60x90/8

От издателя

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.

Содержание

Глава 1. Общие принципы и возможности протокола SIP

1.1 Принципы протокола SIP

Протокол инициирования сеансов - Session Initiation Protocol является протоколом прикладного уровня и предназначается для организации, модификации и завершения сеансов связи: мультимедийных конференций, телефонных соединений и распределения мультимедийной информации.

Протокол SIP разработан комитетом IETF, а спецификации протокола представлены в документе RFC 3261 [ ]. В основу протокола заложены следующие принципы:

Персональная мобильность пользователей. Пользователи могут перемещаться без ограничений в пределах сети, поэтому услуги связи должны предоставляться им в любом месте этой сети. Пользователю присваивается уникальный идентификатор, а сеть предоставляет ему услуги связи вне зависимости от того, где он находится. Для этого пользователь с помощью специального сообщения информирует сеть о своих перемещениях.

Масштабируемость сети характеризуется, в первую очередь, возможностью увеличения количества элементов сети при её расширении. Серверная структура сети, построенной на базе протокола SIP, в полной мере отвечает этому требованию.

Расширяемость протокола характеризуется возможностью дополнения протокола новыми функциями при введении новых услуг и его адаптации к работе с различными приложениями. Расширение функций протокола SIP может быть произведено за счет введения новых заголовков и типов сообщений.

Интеграция в стек существующих протоколов Интернет, разработанных IETF. Протокол SIP является частью сложной архитектуры, разработанной комитетом IETF. Эта архитектура включает в себя также протокол резервирования ресурсов (Resource Reservation Protocol, RSVP; RFC 2205), транспортный протокол реального времени (Real-Time Transport Protocol, RTP; RFC 1889), протокол передачи потоков в реальном времени (Real-Time Streaming Protocol, RTSP; RFC 2326), протокол описания параметров связи (Session Description Protocol, SDP; RFC 2327) и прочие. Однако функции протокола SIP не зависят ни от одного из этих протоколов.

Взаимодействие с другими протоколами сигнализации. Протокол SIP может быть

использован совместно с протоколом H.323. Возможно также взаимодействие протокола SIP с системами сигнализации ТфОП - EDSS1 и ОКС №7. Для упрощения такого взаимодействия сигнальные сообщения протокола SIP могут переносить не только SIP-адрес, но и телефонный номер формата E.164 или любого другого формата. Кроме того, протокол SIP, наравне с протоколами H.323 и ISUP, может применяться для обеспечения функционирования устройств управления шлюзами, в этом случае он должен взаимодействовать с протоколом MGCP или MEGACO.

1.2Интеграция протокола SIP с IP-сетями

Важной особенностью протокола SIP является его независимость от транспортных технологий. В качестве транспорта могут использоваться протоколы X.25, Frame Relay, AAL5, IPX и др. Структура сообщений SIP не зависит от выбранной транспортной технологии.

Сигнальные сообщения SIP могут переноситься не только протоколом транспортного уровня UDP, но и протоколом TCP. Протокол UDP позволяет быстрее, чем TCP, доставлять сигнальную информацию (даже с учетом повторной передачи неподтвержденных 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, а также 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 открываться отдельное TCP-соединение. На рисунке 1.1 показано место, занимаемое протоколом SIP в стеке протоколов TCP/IP.

Протокол инициирования сеансов связи (SIP)	Прикладной уровень
Протоколы TCP и UDP	Транспортный уровень
Протоколы IPv4 и IPv6	Сетевой уровень
PPP, AAL5 ATM, Ethernet, V.34.	Уровень звена данных
UTP5, ВОЛС и др.	Физический уровень

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

По сети с маршрутизацией пакетов IP может передаваться пользовательская информация практически любого вида: речь, видео и данные, а также любая их комбинация, называемая мультимедийной информацией. При организации связи между терминалами пользователей необходимо известить встречную сторону, какого рода информация может приниматься (передаваться), алгоритм ее кодирования и адрес, на который следует передавать информацию. Таким образом, одним из обязательных условий организации связи при помощи протокола SIP является обмен между сторонами данными об их функциональных возможностях. Для этой цели чаще всего используется протокол описания сеансов связи - SDP (Session Description Protocol). Поскольку в течение сеанса связи может производиться его модификация, предусмотрена передача сообщений SIP с новыми описаниями сеанса средствами SDP.

Для передачи медиа информации комитет IETF предлагает использовать протокол RTP, но сам протокол SIP не исключает возможность применения для этих целей других протоколов.

В протоколе SIP не реализованы механизмы управления потоками информации и предоставления гарантированного качества обслуживания. Кроме того, протокол SIP не предназначен для передачи пользовательской информации, в его сообщениях может переноситься информация лишь ограниченного объема. При переносе через сеть слишком большого сообщения SIP не исключена его фрагментация на уровне IP, что может повлиять на качество передачи информации.

Также, протокол SIP предусматривает организацию конференций трех видов:

3 в режиме многоадресной рассылки, когда информация передается на один multicast-адрес, а затем доставляется сетью конечным адресатам;

4 при помощи устройства управления конференции, к которому участники конференции передают информацию в режиме точка-точка, а оно, в свою очередь, обрабатывает ее (т.е. смешивает или коммутирует) и рассылает участникам конференции;

5 путем соединения каждого пользователя с каждым в режиме точка-точка.

Протокол SIP дает возможность присоединения новых участников к уже существующему сеансу связи, т.е. двусторонний сеанс может перейти в конференцию.

1.3Адресация 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 доменных имен - Domain Name Service (DNS). Если же во второй части SIP-адреса размещается IP-адрес, то с рабочей станцией можно связаться напрямую.

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: 387-75-47@sip-gateway.ru

Глава 2. Протокол инициализации сессии (SIP)

Структура протокола SIP

SIP представляет собой многоуровневый протокол. Его функционирование описывается комплексом слабо связанных независимых этапов обработки. Если элемент сети SIP содержит некий уровень, это означает, что он поддерживает группу правил, определённых для данного уровня. Однако не каждый элемент, работающий по протоколу SIP, содержит все уровни. Кроме того, элементы, специфицированные для работы в SIP, являются логическими, а не физическими. В действительности физический элемент SIP может выполнять функции различных логических элементов в зависимости от возложенных на него обязанностей.
Нижний уровень SIP отвечает за синтаксис и кодирование. Кодирование определено с использованием расширенной грамматики Backus-Naur Form (BNF). Полное BNF-описание для SIP содержится в RFC 3261, структура сообщений SIP будет рассмотрена в разделе 2.2.
Второй уровень программной реализации протокола является транспортным. Он определяет, как клиент посылает запросы и принимает ответы и как сервер получает запросы и посылает ответы по сети. Транспортный уровень протокола описан в разделе 2.10.
Третий уровень – это уровень транзакций. Транзакция – это запрос, отосланный клиентской стороной с использованием транспортного уровня SIP серверной стороне, вместе со всеми ответами на этот запрос, отосланными серверной стороной клиенту. Уровень транзакций осуществляет повторную передачу сообщений прикладного уровня, определяет соответствие ответов запросу и уведомляет верхний уровень проткокола в случае таймаута. Любая операция, которую выполняет клиент агента пользователя (UAC), реализуется с помощью серии транзакций. Описание работы уровня транзакций приведено в параграфе 2.3.2. Агенты пользователя (UA) и прокси-серверы с сохранением состояний транзакций (stateful прокси-серверы) содержат уровень транзакций. В противоположность им прокси-сервер без сохранения состояний (stateless прокси-сервер) не включает уровня транзакций. 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 определённого типа запроса.
Уровень, находящийся выше уровня транзакций, называется пользователем транзакций (transaction user - TU). Каждый их объектов SIP, кроме stateless прокси-сервера, является пользователем транзакций. Когда TU желает отослать запрос, он создаёт отдельную клиентскую транзакцию и передаёт ей запрос вместе с IP-адресом, портом и типом транспортного протокола для места назначения, которые определяют куда нужно отослать запрос. TU, который создал клиентскую транзакцию, может также отменить её. Когда клиент отменяет транзакцию, он запрашивает, чтобы сервер прекратил дальнейшую обработку запроса, возвратился в исходное состояние и этой передал транзакции ответ с определённым кодом ошибки. Это осуществляется посредством запроса CANCEL, который создаёт свою собственную транзакцию, но выполняет свои функции в отношении отменяемой транзакции (См параграф 2.3.5).
SIP элементы, которыми являются клиент и сервер агента пользователя, stateful и stateless

прокси-серверы и сервер регистрации, содержат программное обеспечение - ядро (core), которое отличает их друг от друга. Ядра, за исключением ядра stateless прокси-сервера, являются пользователями транзакций (TU). Несмотря на то, что функционирование ядер UAC и UAS зависит от типа запроса, существуют некоторые общие правила для всех типов запросов. Эти правила описаны в разделе 2.1. Для UAC эти правила касаются процесса создания запроса, для UAS они касаются обработки запроса и создания ответа. Поскольку регистрация играет важную роль в протоколе SIP, UAS, который может работать с запросом REGISTER, имеет своё название – сервер регистрации (registrar). Раздел 2.3.3 описывает работу ядер UAC и UAS для запроса REGISTER. В разделе 2.3.4. освещается работа UAC и UAS с запросом OPTIONS, используемого для получения информации о функциональных возможностях UA. Остальные запросы, определённые в основной RFC для SIP (RFC 3261), отсылаются в режиме диалога. Диалог представляет собой равноправное взаимодействие двух агентов пользователя по протоколу SIP, которое длится определённое время. Диалог устанавливает последовательность сообщений между UA и обеспечивает верную маршрутизацию запросов. Запрос INVITE является единственным типом запроса, устанавливающим диалог, определённым в рекомендации RFC 3261 (Однако впоследствии расширения протокола определили еще два таких запроса – SUBSCRIBE и REFER). Когда UAC отсылает запрос в режиме диалога, он помимо выполнения общих правил UAC, описанных в разделе 2.1, следует правилам для работы с запросами в ходе диалога. Параграф 2.3.1 даёт понятие о диалогах и описывает процедуры их создания и поддержания в дополнение к процедурам создания запросов в режиме диалога.

Наиболее важный тип запроса в протоколе SIP – это INVITE, который устанавливает сессию между участниками соединения. Сессия – это совокупность участников соединения и медиапотоков между ними, созданных с целью обмена информацией. Параграф 2.3.6 описывает процедуры создания сессий, приводящие к созданию одного или более диалогов. Параграф 2.3.7 повествует о том, как модифицируются параметры сессии путём применения запроса INVITE в режиме диалога. В параграфе 2.3.8 отражено, как разрушается сессия.

2.1. Назначение и функ циональ ность агента поль зователя (UA)

Агент пользователя (UA) - это логический объект, который может выполнять как функции клиента агента пользователя, так и функции сервера агента пользователя. В сети SIP он представляет оконечное оборудование абонента. Соответственно, агент пользователя состоит из клиента агента пользователя (UAC), генерирующего запросы, и сервера агента пользователя (UAS), который формирует ответы. Работа UAC и UAS зависит от типа запроса и от того, происходит ли передача запроса или ответа в процессе диалога. Далее в этом разделе будет рассмотрена работа UAC и UAS вне диалога.

2.1.1. Клиент агента пользователя (UAC)

Клиент агент пользователя – это часть программного обеспечения агента пользователя, которая создает новые запросы, отправляет их и обрабатывает принятые ответы. Запросы генерируются в результате внешних воздействий (нажатия кнопок пользователем, сигнала из

телефонной линии).

2.1.1.1. Создание запроса

Правильный запрос, составленный клиентом агента пользователя, должен включать стартовую строку, содержащую тип запроса, поле Request-URI и версию SIP, и следующий базовый набор полей заголовков: To, From, CSeq, Call-ID, Max-Forwards и Via. Эти заголовки, как и Request-URI, обязательны для всех SIP-запросов. Данные шесть заголовков являются основными частями SIP-сообщения, поскольку совместно обеспечивают большинство требуемых сервисов по маршрутизации сообщений, включающих адресацию сообщений, маршрутизацию ответов, ограничение распространения сообщения, сохранение очередности сообщений и уникальную идентификацию транзакций. В разделе 2.2.2 наряду с основными будут подробно рассмотрены все существующие заголовки.

Ниже будет рассмотрена работа UAC вне диалога. Примерами запросов, отсылаемых вне диалога, является запрос INVITE (См. раздел 2.3.1), устанавливающий сессию, и запрос OPTION для запроса информации о функциональных возможностях (См. раздел 2.3.4.)

Формирование поля Request-URI

Поле Request-URI указывает на пользователя или сервис, к которому адресован запрос. Исходное значение поля Request-URI сообщения устанавливается таким же, как URI в поле To. Исключение составляет тип запроса REGISTER; подробно процесс установки Request-URI для сообщения REGISTER описан в разделе 2.3.3. По причинам обеспечения анонимности (privacy) или из соображений удобства может быть нежелательно устанавливать в полях Request-URI и To одинаковые значения, особенно, если инициирующий UA предвидит, что Request-URI будет изменён в процессе передачи.

В некоторых случаях наличие предустановленного маршрута может повлиять на Request-URI сообщения. Предустановленный маршрут представляет собой упорядоченную последовательность URI, которая идентифицируют последовательность серверов, по которой UAC пошлёт 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 по заполнению полей Request-URI и Route, детализированные в параграфе 2.3.1.2 (даже несмотря на то, что диалога не существует) при использовании желаемого Request-URI в качестве URI удалённого узла.

Формирование заголовка To

Поле заголовка To устанавливает желаемого логического получателя запроса - публичный адрес получателя (address-of-record) или ресурс, на который отправляется запрос. Значение заголовка может как быть, так и не быть конечным получателем 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» (RFC 2806)). 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 адреса 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 «@» SIP URI (например, sip:anton@niits.ru). Использование строки ввода для формирования пользовательской части SIPS адреса подразумевает, что UA желает установить безопасное соединение, имя домена также определяется. Правая часть адреса может указывать домашний домен (home domain) инициатора запроса; в этом случае вызываемый пользователь также находится в данном домене. Тогда SIP элемент, ответственный за пересылку сообщений в данном домене осуществляет обработку исходящего запроса. Это необходимо для таких функций, как «быстрый вызов», которые требуют интерпретации части пользователя в домашнем домене.

Запросы вне диалога не должны содержать параметра «tag» в поле To. Параметр «tag» в заголовке To определяет конкретный терминал вызываемого пользователя (например домашний, рабочий или сотовый телефон) из терминалов 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. Поскольку диалог не был установлен, «tag» в запросе отсутствует.

Пример поля заголовка To:

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

Формирование заголовка From

Поле заголовка From содержит логический идентификатор инициатора сообщения, как правило, публичный адрес вызывающего пользователя. Также 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). UAC должен использовать отображаемое имя "Anonymous", если идентификационная информация пользователя (identity) неизвестна.

Обычно, поле заголовка From запросов, которые создаёт UA, заполняется на основании значения, предварительно определённого пользователем или администратором локального домена пользователя. Если конкретный UA используется несколькими пользователями, он может иметь переключаемые профили, которые содержат URI, соответствующий конкретным пользователям. Получатели запросов могут аутентифицировать инициатора запроса для того, чтобы убедиться, что они те, кого представляют заголовки From их запросов.

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

Примеры:

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

Формирование заголовка Call-ID

Заголовок Call-ID – это уникальный идентификатор, объединяющий группу сообщений. Он должен совпадать для всех запросов и ответов, отправляемых любым из двух UA в процессе диалога.

При создании нового диалога, заголовок Call-ID должен быть выбран UAC как уникальный идентификатор. Все SIP агенты пользователя должны иметь средства, чтобы гарантировать, что Call-ID, созданный ими, не будет случайно генерирован другим UA. Когда запросы отправляются повторно после получения ответа с кодом ошибки, требующего коррекции запроса, (например, запрос на предоставление отклика аутентификации), эти повторные запросы не рассматриваются как новые и не нуждаются в новом значении заголовка Call-ID (См. пункт «Обработка ответов класса 4xx» параграфа 2.1.1.2).

При генерации значений Call-ID рекомендуется использовать случайные криптографические идентификаторы (по RFC 1750), их использование обеспечивает некоторую защиту от взлома сессий 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
```

Формирование заголовка CSeq

Поле заголовка CSeq служит средством для идентификации и упорядочивания транзакций. Оно содержит порядковый номер и тип 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}. Клиент может выбирать любой механизм для создания значений заголовка CSeq.

Пример:

```
CSeq: 4711 INVITE
```

Формирование заголовка Max-Forwards

Поле заголовка Max-Forwards служит для ограничения числа пересылок запроса на пути к месту назначения. Оно состоит из целого числа, которое уменьшается на единицу при каждой пересылке. Если значение этого заголовка достигнет 0 до того, как запрос достигнет места назначения, это сообщение будет отклонено ответом с кодом ошибки 483 (Too Many Hops).

UAC должен вставлять заголовок Max-Forwards в каждый отправляемый запрос. Рекомендуемая величина для его значения: 70. Величина выбрана достаточно большой, чтобы гарантировать, что запрос не будет отброшен сетью SIP при отсутствии петель, но и не слишком большой, 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.

Формирование заголовка Via

Поле заголовка Via указывает один из узлов, используемых для проведения транзакции и идентифицирует местоположение (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 вставить в него поле Via. Также необходимо указать название протокола – SIP, и его версию - 2.0. Поле заголовка Via должно содержать параметр «branch». Этот параметр используется для идентификации транзакции, созданной данным

запросом. Он используется и клиентом, и сервером.

Значение параметра «branch» должно быть уникальным для всех запросов, отправляемых UA. Исключение составляют запрос CANCEL и запрос ACK на ответы, отличные от класса 2xx. Запрос CANCEL будет иметь то же значение параметра «branch», что и запрос, который он отменяет. Запрос ACK на ответ, отличный от класса 2xx также будет иметь тот же параметр «branch», что и INVITE, ответ на который он подтверждает. Уникальность этого параметра облегчает его использование в качестве идентификатора транзакции. Параметр «branch», вставляемый 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 в мировом масштабе.
Другие параметры заголовка Via («maddr», «ttl» и «sent-by») будут установлены во время обработки запроса транспортным уровнем протокола SIP.

Формирование заголовка Contact

Поле заголовка Contact содержит SIP или SIPS URI, который может быть использован для связи с пользователем UA, пославшим сообщение (получатель может использовать этот адрес для своих будущих запросов). Этот заголовок должен присутствовать и содержать только один SIP или SIPS URI в любом запросе, результатом которого может стать установление диалога. Таким запросом, например, является INVITE. Для этих запросов масштаб заголовка Contact должен быть глобальным, т.е. значение поля заголовка Contact содержит URI, на который UA желает получать запросы, и этот URI должен быть действительным, даже если используется в последующих запросах вне диалога.

Если поле Request-URI или первое значение заголовка Route содержит SIPS URI, поле Contact также должно содержать SIPS URI. Заголовок Route служит для принудительной маршрутизации запроса в соответствии со списком прокси-серверов.

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** уникально идентифицирует новые функциональные возможности для SIP в соответствии с расширениями SIP, определёнными в дополнительных RFC. Это делается для того, чтобы предотвратить требование серверов на то, чтобы клиенты реализовывали не стандартизированные, определённые фирмой-производителем функции для того, чтобы получить обслуживание. Расширения, описанные в RFC не имеющих статуса Standard, не используются в заголовке Supported запроса, так как они зачастую используются для определения нестандартизованных расширений фирм-производителей.

Если UAC хочет потребовать, чтобы UAS понял расширение, которое UAC применит к запросу для обработки запроса, он должен вставить 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 вставить в запрос заголовок Proxy-Require, указывающий **option-tag** для этого расширения.

Дополнительные компоненты сообщения

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

SIP-запросы могут содержать закодированное MIME тело сообщения (См. Multipurpose Internet Mail Extensions Part Two: Media Types", RFC 2046). Независимо от типа тела, которое содержит запрос, конкретные заголовки должны быть составлены так, чтобы описывать содержимое тела сообщения (заголовки Content-Disposition, Content-Encoding, Content-Language, Content-Length, Content-Type). См параграф 2.2.2).

Отправка запроса

При отправке 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 RFC 3263: "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. Эти 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 наличие альтернативного набора мест назначения для использования в запросах. 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 упрощённую альтернативу предустановленному маршруту, как способу указания исходящего прокси-сервера. Однако такой подход для конфигурирования исходящего прокси-сервера не рекомендуется, вместо этого лучше использовать предустановленный маршрут с единственным URI. Если запрос

содержит поле заголовка Route, то он будет отправлен 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 отправить запрос по адресу, указанному в первом значении поля заголовка Route, вместо поддержки политики отправки всех сообщений исходящему прокси-серверу. Этим гарантируется, что исходящие прокси-серверы, которые не добавляют поле заголовка Record-Route, выпадут из маршрута последующих запросов. Оконечные точки, которые не могут определить первое значение для поля заголовка Route, делегируют выполнение этой задачи исходящему прокси-серверу.

UAC должен следовать процедурам, определённым в "SIP: Locating SIP Servers", RFC 3263 для stateful SIP элементов, посылая 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.

2.1.1.2. Обработка ответов

Ответы сначала обрабатываются транспортным уровнем SIP, а потом направляются на уровень транзакций. Уровень транзакций производит их обработку и затем передаёт 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.

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 истечения времени (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), то оно должно быть трактовано, как ответ 503 (Service Unavailable). Обычно это означает критическую ICMP-ошибку в протоколе UDP или нарушение соединения в TCP.

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 ответ, 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 и 100, и 183 ответы.

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.

Обработка ответов класса 3xx

При поступлении 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 только один URI - Request-URI оригинального запроса. Если отправитель желает сформировать новые запросы после получения перенаправляющего ответа класса 3xx на предшествующий запрос, 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 не небезопасный URI.

По мере того, как target set растёт, UAC может генерировать новые запросы, используя адреса выбранные из target set в любом порядке. Простейшим механизмом для этого является упорядочивание в соответствии со значением параметра «q» каждого значения заголовка Contact. Параметр «q» определяет приоритеты среди адресов, содержащихся в заголовке Contact путём варьирования 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 определённым значением параметра «q» параллельно. Другой подход предусматривает последовательную обработку в порядке уменьшения значения параметра «q», произвольно выбирая между адресами 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 запросы не должны расцениваться как ошибки.

Когда приходит ошибка при обращении на определённый контактный адрес, клиент должен попытаться отослать запрос на следующий контактный адрес. Это повлечёт за собой создание

новой клиентской транзакции для доставки нового запроса.

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 для 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 разделённый запятыми список значения 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 перезаписано 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 перезаписывается, 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, будет 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 интервала истечения времени действия.

Обработка ответов класса 4xx

Отдельные коды ответов класса 4xx требует особых действий от UA по обработке в не 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 будет выполнять процедуру аутентификации для последующего запроса.

6 Код ответа **413 (Request Entity Too Large)** означает, что запрос содержит тело, слишком большой длины, чтобы UAS мог его принять. При этом UAC заново передает запрос, убирая тело сообщения или уменьшая его.

7 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.

8 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.

9 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. UAC в этом случае должен заново передать запрос, убрав из него все расширения, указанные в поле заголовка Unsupported ответа.

10 **Ответ с кодом 494 (Security Agreement Required)** передаётся сервером при выполнении процедуры выбора механизма обеспечения безопасности. Ответ должен включать заголовок Security-Server со списком механизмов обеспечения безопасности, поддерживаемых сервером. UAC должен выбрать подходящий механизм безопасности и применить его при отсылке 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, который на единицу больше предыдущего.

При получении других ответов класса 4xx передача запроса может производиться заново в зависимости от типа запроса и конкретного случая использования.

2.1.2. Сервер агента пользователя (UAS)

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

2.1.2.1. Процедура обработки запросов

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

Серверы агента пользователя обрабатывают запросы пошагово, как указано в данном разделе (то есть начиная с аутентификации, затем анализа типа запроса, полей заголовков и так далее).

Определение типа зап роса

Когда запрос прошел аутентификацию (или она была пропущена), UAS должен выяснить тип запроса. Если UAS определил, но не поддерживает тип запроса, он должен отправить ответ с кодом 405 (Method Not Allowed); в этом ответе должно присутствовать заголовок Allow, который содержит список типов запросов, поддерживаемых UAS.

В случае поддержки сервером типа запроса, обработка сообщения продолжается.

Определение типа заг оловка

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.

Обработка полей To и Request-URI

В поле заголовка To вызывающий пользователь указывает адрес получателя запроса. UAS может поступать произвольным образом при выработке решения о приёме запроса, заголовок To которого идентифицирует не данный UAS. Однако рекомендуется, чтобы UAS принимал запросы, даже если они содержат неизвестную схему URI (например, схему «tel») в поле To, или если поле To не адресует ни текущего, ни одного из существующих пользователей этого UAS. Если все же, UAS решает отклонить запрос, он должен создать ответ с кодом 403 (Forbidden) и передать его серверной транзакции (понятие серверной транзакции даётся в разделе 2.3.2) для отсылки.

Поле Request-URI идентифицирует UAS, который должен обработать запрос. Если в Request-URI используется схема адресации, неподдерживаемая сервером, запрос должен быть отклонён и отправлен ответ с кодом 416 (Unsupported URI Scheme). Если Request-URI не идентифицирует адрес, для которого UAS готов принять запрос, сообщение отбрасывается и отправляется ответ с кодом 404 (Not Found). Обычно, UA, который использует сообщение REGISTER для связки публичного адреса пользователя (address-of-record) с конкретным

контактным адресом, должен опознавать запросы, Request-URI которых совпадает с его контактным адресом. Другие возможные источники для значения полученного Request-URI включают заголовки Contact запросов и ответов, отправленных UA для установления или обновления параметров диалога.

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 различными путями, вероятнее всего из-за размножения запросов прокси-сервером. Ядро UAS обрабатывает первый такой запрос и отправляет ответ с кодом 482 (Loop Detected) на последующие запросы.

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 должны поддерживаться UAS для правильной обработки запроса. Если UAS не понимает какого-либо идентификатора расширения **option-tag**, указанного в поле Require, он должен отослать ответ с кодом 420 (Bad Extension). UAS должен добавить в ответ заголовок Unsupported со списком непонятных опций, указанных в поле заголовка Require запроса.

Заметим, что заголовки Require и Proxy-Require не должны использоваться в запросе CANCEL, а также в запросе ACK, отправляемом на ответ отличный от класса 2xx. Эти заголовки должны игнорироваться, даже если они имеются в таких запросах.

Запрос ACK на ответ класса 2xx должен содержать только те значения Require и Proxy-Require, которые присутствовали 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`

Эти действия гарантируют, что взаимодействие между клиентом и сервером будет проходить без задержек, когда все опции (options) 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 механизмы согласования. К тому же, это устраняет проблемы, возникающие, когда клиент требует возможностей, которые сервер не поддерживает.

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). Ответ должен содержать заголовок Accept со списком всех типов тел сообщения, которые понимает UAS, в случае наличия в запросе тел сообщения с типами, не поддерживаемыми сервером. Если запрос содержит типы кодеков содержимого, непонятные UAS, ответ должен содержать заголовок Accept-Encoding со списком кодировок, понятных UAS. Также 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.

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 сгенерирован 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, включённом в ответ. Сервер не должен применять расширения, не указанные в заголовке Supported запроса.

После того, как все вышеописанные действия выполнены, дальнейшая обработка запросов зависит от его типа.

2.1.2.2. Создание ответа

Когда UAS создаёт ответ на запрос, он следует общим процедурам, описанным ниже. Также может потребоваться выполнение дополнительных действий, которые зависят от конкретного кода ответа.

После завершения выполнения всех процедур, связанных с созданием ответа UAS возвращает ответ серверной транзакции, от которой был получен запрос.

Отсылка предварительного ответа

Основное правило при генерации ответа, вне зависимости от типа запроса, состоит в том, что серверы UA должны отправлять предварительные ответы только на запросы INVITE. При этом они должны как можно быстрее создавать на запросы кроме INVITE окончательные ответы.

При формировании ответа с кодом 100 (Trying) заголовок Timestamp (указывает, когда UAC отослал сообщение UAS) из запроса должен быть в точности скопировано в него. Это выполняется в целях 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 отправки ответа, выраженную в секундах.

Заголовки и параметры «tag»

Поле From ответа должно совпадать с аналогичным полем запроса, равно как и поля Call-ID, Cseq и Via ответа должно 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», то 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)).

Действие UAS без сохранения состояний

UAS без сохранения состояний (stateless) – это UAS, который не запоминает состояния текущих транзакций. Он нормально отвечает на запросы, но в отличие от UAS с сохранением состояний (stateful) не сохраняет состояния транзакций после отправки 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 напрямую от транспортного уровня SIP и отправляет ответ также напрямую транспортному уровню.

Основная роль stateless UAS состоит в поддержке неаутентифицированных 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 отправлять предварительных (1xx) ответов
- 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 – для одинаковых запросов должна генерироваться одинаковые параметры «tag».

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

2.2. Сообщения протокола SIP

2.2.1. Структура сообщений

Протокол SIP – это текстовый протокол, использующий набор символов ISO 10646 в кодировке UTF-8 (RFC 2279). Сообщения протокола SIP представляют собой либо запрос от клиента серверу, либо ответ сервера клиенту.

Запросы и ответы используют один базовый формат сообщения, одинаковый, несмотря на различия в наборе символов и синтаксисе. Сообщения обоих типов состоят из:

- стартовой строки;

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

Рис 2.1 Структура сообщения протокола SIP

Стартовая строка, каждая строка поля заголовка и пустая строка должны быть завершены символами возврата каретки и перевода строки (CRLF). Пустая строка должна быть 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.

Заголовки сообщений служат для передачи информация 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.

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 не содержит тело сообщения.

2.2.2. Заголовки сообщений

Формат заголовка

Поля заголовков SIP-сообщений похожи на поля заголовков HTTP-сообщений по синтаксису и семантике. В частности, SIP поля заголовков 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 для полей заголовков позволяет использовать любое количество пробелов (SP) для отделения двоеточия. Однако на деле стараются избегать наличия пробелов между названием поля и двоеточием и ограничиваются одним пробелом для отделения двоеточия от значения.

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

В приведенных выше примерах все строки равнозначны, но последняя предпочтительнее.

Поля заголовков могут быть расширены на несколько строк, когда каждая последующая 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 строки 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 для ускорения анализа и обработки. 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, разделённых запятой. Возможно объединить такие заголовки в одну

пару «имя поля: значение поля», не изменяя семантики сообщения, путём добавления каждого последующего 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 также могут присутствовать в сообщении, но объединить их невозможно, поскольку грамматика данных заголовков не придерживается общих правил для SIP-заголовков.

Реализации должны быть способны обработать последовательности заголовков с одинаковым именами со значениями, представленными как в форме последовательности, разделённой запятыми, так и в виде «одно значение на строку».

Примеры последовательностей заголовков, приведённые ниже, правомерны и эквивалентны.

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

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

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

Следующие последовательности заголовков правомерны, но не равнозначны.

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

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

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

Формат значения заголовка зависит от имени заголовка. Это всегда будет последовательность текстовых октетов в UTF-8 кодировке или комбинация символьных фраз (tokens), пустого пространства (whitespace), разделительных знаков и строк, заключённых в кавычки. Большинство существующих полей заголовков придерживаются общего формата для значений, основанного на последовательности пар *имя параметра – значение параметра*, разделённых знаком точка с запятой.

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

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

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

```
Contact: <sip:anton@niits.ru>;expires=3600
```

эквивалентно

```
CONTACT: <sip:anton@niits.ru>;ExPiReS=3600
```

и

```
Content-Disposition: session;handling=optional
```

эквивалентно

```
content-disposition: Session;HANDLING=OPTIONAL
```

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 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 размер для данной сети, что приведет к фрагментации пакета. Для избежания этого используется сжатый формат имен основных заголовков, подобно тому, как это делается в протоколе SDP. Ниже приведен список таких заголовков.

Таблица 2.1 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

Типы заголовков

- **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, кокой 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
```

- **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.

Пример:

```
Accept-Encoding: gzip
```

- **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. В случае если заголовок отсутствует, сервер устанавливает, что клиент поддерживает все языки.

Правило упорядочивания языков в списке по предпочтительности базируется на параметре «q». Пример:

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

- **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. В основном этот заголовок вставляется прокси-сервером для обеспечения функции отличительного звукового сигнала вызова. Пользователь должен иметь возможность отключить эту функцию. Это помогает избежать деструктивных последствий в случае использования данного заголовка SIP элементами, с которыми не были установлены доверительные отношения. Пример:

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

- **Allow**

Заголовок Allow содержит список типов запросов, поддерживаемых агентом пользователя, сформировавшим сообщение. Все типы запросов, понимаемые UA, включая ACK и CANCEL, должны входить в этот список. Отсутствие заголовка Allow не означает, что отсылающий 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
```

- **Allow-Events**

Заголовок Allow-Events в случае присутствия содержит список, состоящий из идентификаторов функциональных возможностей для информирования клиента о событиях определённого типа - **event package**, поддерживаемых клиентом (если посылается в запросе) или сервером (если посылается в ответе). Другими словами, оконечная точка, отсылающая 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 (events), 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 вставляться прокси-серверами. Пример:

```
Allow-Events: refer
```

- **Authentication-Info**

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

```
Authentication-Info: nextnonce="47364c23432d2e131a5fb210812c"
```

- **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. Эти правила описаны 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 содержится в разделе 2.6.3.

```
Authorization: Digest username="Anton", realm="niits.ru",
nonce="84a4cc6f3082121f32b42a2187831a9e",
response="7587245234b3434cc3412213e5f113a5432"
```

- **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 этому идентификатору. Между ними должен стоять символ «@». Значения Call-ID регистрозависимы и могут сравниваться побайтно. Примеры:

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

- **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* даёт 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", RFC 2426) или LDIF ("The LDAP Data Interchange Format (LDIF) – Technical Specification", RFC 2849).

Использование заголовка Call-Info может представлять угрозу безопасности пользователя. Если вызываемый пользователь воспользуется URI, приведёнными

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 вставлен в запрос прокси-сервером. Пример:

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

- **Contact**

Поле заголовка Contact несёт 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, 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 комбинацией символьных фраз или строка, заключённая в кавычки, когда 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 заключённый 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@loniis.ru>
         ;q=0.7; expires=3600,
         "Alexander" <mailto:alexander@loniis.ru> ;q=0.1
```

- **Content-Disposition**

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

В протоколе SIP определено несколько значений для заголовка Content-Disposition. Значение *session* указывает, что часть тела описывает сессию 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 другим образом представлена пользователю. Если заголовок Content-Disposition утерян, 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 для визуального информирования 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, который инициирует диалог; например, это информационное тело может быть представлено в виде вызывного сигнала для телефонного вызова после того, как был отослан предварительный ответ с кодом 180 (Ringing).

Любые MIME тела со значением заголовка Content-Disposition, которое предписывает передачу содержимое пользователю, может обрабатываться только, 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 значение по умолчанию для Content-Disposition. В противном случае выставляется значение *render*. Пример:

```
Content-Disposition: session
```

- **Content-Encoding**

Заголовок Content-Encoding используется в качестве модификатора типов тела сообщения. Когда данный 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.

Первостепенно 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. Причём, UAS должен использовать только типы кодирования, которые были перечисленные в поле заголовка Accept-Encoding запроса.

Пример:

```
Content-Encoding: gzip
```

- **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 конкретном языке, то это будет указано в заголовке. В случае отсутствия заголовка Content-Language в сообщении 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
```

- **Content-Length**

Заголовок Content-Length указывает отображённый в десятичном виде размер тела сообщения, посланного получателю, в байтах. Приложения должны помещать в данное поле размер тела сообщения, подлежащего отправке, не взирая 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. Разрешёнными значениями для Content-Length является любое число, большее или равное нулю. Когда в передаваемом сообщении тело отсутствует, в поле заголовка

Content-Length выставляется ноль.

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

Пример:

```
Content-Length: 349
```

- **Content-Type**

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

Примеры:

```
Content-Type: application/sdp
```

```
c: text/html; charset=ISO-8859-4
```

- **CSeq**

Заголовок CSeq **-** уникальный идентификатор запроса, относящегося к одному соединению. Он служит для корреляции запроса с ответом на него, а также для различия первоначальных запросов с переадресованными. Заголовок состоит из двух частей: натурального числа из диапазона чисел от 1 до 2^{32} и типа запроса. Часть типа запроса является регистрозависимой. Сервер должен проверять значение величины CSeq в каждом принимаемом запросе**,** и считает его новым, если значение больше предыдущего. Этот заголовок копируется из запроса в ответ.

Пример:

```
CSeq: 4711 INVITE
```

- **Date**

Заголовок Date содержит дату и время первой отправки сообщения. В отличие от HTTP/1.1, протокол SIP поддерживает самый новейший, среди описанных в RFC 1123, формат даты, он переводит любой часовой пояс к стандарту GMT, хотя RFC 1123 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 знания клиентами

отклонения своих часовых поясов от GMT.

Пример:

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

- **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 ПК до функции предоставления аудио для оконечных точек, соединённых через шлюзы. Вместо того, чтобы принуждать генерирующий код ошибки сервер делать выбор между посылкой сообщения с кодом ошибки и указанием причины, и воспроизведением звукового сигнала, существует возможность предусмотреть оба варианта в поле заголовка Error-Info. Впоследствии клиент самостоятельно решает, какой тип индикации ошибки предоставить вызывающему пользователю.

UAC обрабатывает SIP или SIPS URI в поле заголовка Error-Info также, как контактный адрес в заголовке Contact перенаправляющего сообщения, и может сгенерировать новое сообщение INVITE, направляемое автоинформатору. Автоинформатор, предоставляющий вызывающему пользователю записанное уведомление, может находиться по адресу, использующему схему адресации, отличную 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>
```

- **Event**

В заголовке должен быть указан ровно один тип события (представленный в виде идентификатора - **event package**), на которое осуществляется подписка (subscription) или передаётся уведомление (notification).

В целях обеспечения соответствия сообщений NOTIFY и SUBSCRIBE значение типа события («event-type») и параметр «id» (в случае 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
```

- **Expires**

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

Пример:

```
Expires: 5
```

- **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 одноимённый заголовок ответа.

Отображаемое имя должно информировать пользователя об инициаторе запроса. В случае если не удаётся установить личность вызывающего пользователя, в качестве display name (отображаемого имени) 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 берутся. Это означает, что отображаемое имя и наличие либо отсутствие угловых скобок не влияют на результат сравнения.

Примеры:

```
From: "Vladimir" <sip:vladimir@protei.ru> ;tag=a48s
```

```
From: sip:+79213434329@gateway.protei.ru;tag=887s
```

- **In-Reply-To**

В поле заголовка In-Replay-To перечисляются уникальные идентификаторы сеансов связи (Call-ID), инициированных отправителями с данным пользователем. Эти идентификаторы по мере поступления буферизируются клиентом и впоследствии включаются в поле заголовка In-Replay-To в ответном вызове. Это позволяет системам автоматического распределения вызовов маршрутизировать ответный вызов инициатору

первого звонка. Также это даёт возможность вызываемым пользователям отфильтровывать вызовы т.е. принимать только ответные вызовы от пользователей, с которыми ранее осуществлялись сеансы связи, инициированные самим вызываемым пользователем. Однако, этот заголовок не является заменой аутентификации в запросах.

Пример:

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

- **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 осуществить сообщению. Это число уменьшается каждым сервером, который пересылает запрос дальше. В качестве первоначального значения рекомендуется брать 70.

Заголовок Max-Forwards должен вставляться теми элементами, которые иначе не могут гарантировать обнаружение петли.

Пример:

```
Max-Forwards: 6
```

- **Min-Expires**

Заголовок Min-Expires несёт 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
```

- **MIME-Version**

Заголовок MIME-Version указывает версию стандарта MIME-информации, помещённой в тело сообщения.

Пример:

```
MIME-Version: 1.0
```

- **Organization**

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

Пример:

```
Organization: Niits
```

- **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 по обнаружению 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 чёткой последовательности: по мере продвижения прокси-серверами каждое новое значение добавляется сверху, смещая вниз все присутствующие значения. Кроме того, также 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 отсылки сообщений регистрации, и может в дальнейшем использовать её для своих задач.

Path отличается от Record-Route тем, что заголовок Path применяется в запросах REGISTER и ответах класса 2xx на него, 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.

Значения заголовка Path придерживаются синтаксиса, определённого 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>
```

- **Priority**

Заголовок Priority указывает на срочность запроса с точки зрения клиента. В нём содержится приоритет SIP-запроса для конечного пользователя или его UA. К примеру, содержимое поля данного заголовка влияет на маршрутизацию вызова и процесс принятия его сервером. Сообщения, не содержащие заголовка Priority, расцениваются, как

сообщение 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), *urgent* (срочное) и *emergency* (экстренное); могут быть определены и дополнительные значения. Рекомендуется, чтобы значение *emergency* выставлялось только в случае экстренных вызовов на соответствующие службы.

Примеры:

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

или

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

- **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, активировавшему функцию анонимности. Агент пользователя должен иметь средства для того, чтобы запросить такие сервисы обеспечения анонимности (privacy services). Для этой цели используется заголовок Privacy. UA помещает в сообщение заголовок Privacy, когда существует необходимость воспользоваться сервисами анонимности, предоставляемыми сетью.

На сегодняшний день 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 сессии (сессий), описание которой, например, содержится в SDP-теле, инициированной этим сообщением. Выполнение этого запроса приведёт к тому, что IP-адрес, с которого будет поступать трафик в ходе сессии, окажется замаскированным. Значение *user* обычно выставляется теми прокси-серверами, у которых есть предварительная договорённость с пользователем для того, чтобы указать, 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 по обеспечению анонимности не требуется. Значение *critical* сообщает, что функции по обеспечению

анонимности сообщения должны быть обязательно выполнены, в противном случае запрос должен быть отклонён.

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

```
Privacy: none
```

- **Proxy-Authenticate**

Заголовок Proxy-Authenticate содержит запрос аутентификации, состоящий из поля, отображающего схему аутентификации, и ряд параметров, необходимый для проведения процедуры аутентификации с данным прокси-сервером для указанного Request-URI. Данный заголовок включается в состав ответа с кодом 407 (Proxy Authentication Required) или с кодом 401 (Unauthorized). Более подробная информация о заголовке содержится в разделе 2.6.3.

Пример:

```
Proxy-Authenticate: Digest realm="niits.ru",
qop="auth", nonce="f84f1cec41e6cbe5aea9c8e88d359",
opaque="", stale=FALSE, algorithm=MD5
```

- **Proxy-Authorization**

Заголовок Proxy-Authorization идентифицирует пользователя прокси-серверу, который требует аутентификации. Значение поля заголовка состоит из отклика аутентификации, содержащего информацию аутентификации агента пользователя для прокси-сервера, обслуживающего область запрашиваемого ресурса. Если запрос идентифицирован и область специфицирована, та же самая идентификационная информация может быть использована для других запросов, направляемых в данную 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 разделе 2.6.3.

Пример:

```
Proxy-Authorization: Digest username="Anton", realm="niits.ru",
nonce="c60f3082ee1212b402a21831ae",
response="245f23415f11432b3434341c022"
```

- **Proxy-Require**

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

Пример:

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

- **P-Asserted-Identity**

Заголовок P-Asserted-Identity используется в сообщениях между логическими элементами SIP, между которыми существуют доверительные отношения (обычно между посредниками), для переноса информации, удостоверяющей пользователя (как правило его публичного адреса). Значение заголовка состоит из URI и опционального отображаемого имени. URI может иметь схему sip, sips, или tel. Также возможно присутствие двух значений в заголовке; 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 с которым 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, отсылающий 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 для своей информации. Пример:

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

- **P-Preferred-Identity**

Заголовок P-Preferred-Identity используется в сообщениях, которые агент пользователя отправляет прокси-серверу, с которым у него установлены доверительные отношения. Заголовок переносит информацию, удостоверяющую пользователя, которую инициатор сообщения желает использовать в качестве значения заголовка P-Asserted-Identity, которое пользующийся доверием элемент вставит в сообщение. Заголовок P-Preferred-Identity может состоять из одного значения и иметь схему sip, sips, или tel. Также возможно присутствие двух значений в заголовке; 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>
```

- **P-Media-Authorization**

Заголовок P-Media-Authorization предназначен для контроля доступа к услуге обеспечения QoS средствами SIP. Использование данного заголовка способствует предотвращению атак типа denial-of-service (отказ в обслуживании). Конечно для такой цели могло быть использовано тело сообщения, но при этом отсутствовала бы возможность сквозного шифрования тела. Заголовок P-Media-Authorization включается в сообщение прокси-сервером, осуществляющим контроль доступа к QoS, во все предварительные ответы (кроме ответа с кодом 100), первый надёжный 1xx или 2xx ответ и во все повторные передачи этого надёжного ответа для UAC или в запрос INVITE, ACK, UPDATE и PRACK - для UAS.

Заголовок P-Media-Authorization содержит один или более идентификаторов для предоставления доступа к QoS. UA использует все идентификаторы из последнего полученного запроса/ответа от прокси-сервера, предоставляющего доступ к QoS, при запросе резервирования ресурсов для медиапотоков, используемых в ходе сессии (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 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).

- **The P-Associated-URI**

Этот заголовок, являясь расширением SIP, позволяет регистрирующему серверу возвращать агенту пользователя список существующих контактных адресов для конкретного зарегистрированного публичного адреса. Заголовок P-Associated-URI содержится в ответе с кодом 200 (OK) на запрос REGISTER и содержит список контактных адресов. Если registrar поддерживает данное расширение, он должен всегда вставлять заголовок P-Associated-URI в ответ 200 (OK) на REGISTER не зависимо от того, какую процедуру осуществлял UA пользователя: регистрацию, изменение регистрации или удаление регистрации и не зависимо от числа зарегистрированных контактных адресов для публичного адреса.

- **P-Called-Party-ID**

Заголовок P-Called-Party-ID помещает прокси-сервер (обычно 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 публичный адрес вызываемого пользователя. Поэтому, изучая содержимое заголовка P-Called-Party-ID, UAS определяет на какой публичный адрес было

послано приглашение (например, пользователь может одновременно использовать личный и рабочий публичные адреса для принятия приглашений к сеансу связи). Сервер может использовать эту информацию, для подачи пользователю различных аудиовизуальных вызывающих сигналов в зависимости от того, на какой из публичных адресов пользователя пришёл вызов. Пример.

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

- **P-Visited-Network-ID**

Сети, построенные в соответствии с 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.

Одним из условий домашней сети для принятия 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 информацию о сети временного пребывания домашней сети. При наличии соглашений с сетью временного пребывания домашняя сеть предоставит «блуждающему» UA требуемые сервисы.

Заголовок P-Visited-Network-ID используется для доставки идентификатора сети, где временно находится UA, для сервера регистрации или домашнего прокси-сервера (home proxy); заголовок включает в запрос прокси-сервер сети временного нахождения. Этот идентификатор должен быть известен и серверу регистрации, и домашнему прокси-серверу домашнего домена, и прокси-серверам в сети временного нахождения UA. Как правило, заголовок P-Visited-Network-ID используется для запроса 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 в глобальном масштабе. Пример.

```
P-Visited-Network-ID: "Visited network number 1"
```

- **P-Access-Network-Info**

Когда UA создаёт SIP запрос или ответ, и знает, что он будет надёжно (с подтверждением) отослан SIP прокси-серверу, который предоставляет услуги данному UA, он вставляет в сообщение заголовок P-Access-Network-Info. Этот заголовок содержит информацию о сети доступа, которую использует UA для осуществления IP-взаимодействия. Как правило, заголовок игнорируется прокси-серверами, находящимися между UA и SIP прокси-сервером, предоставляющим услуги. Прокси-сервер, предоставляющий услуги, может проверить заголовок P-Access-Network-Info и использовать информацию, содержащуюся там, для предоставления услуг определённого типа в зависимости от значения заголовка. Некоторые услуги более или наоборот менее подходят для пользователя в зависимости от типа доступа; также некоторые услуги могут быть предоставлены с бóльшим качеством, когда прокси-сервер владеет детальной информацией о сети доступа (поскольку могут быть адаптированы под конкретного пользователя). Перед тем, как переслать запрос дальше, этот прокси-сервер удаляет заголовок P-Access-Network-Info из сообщения. UA, предоставляющий информацию должен доверять прокси-серверу, который предоставляет сервисы – только в этом случае гарантируется, что прокси-сервер удалит заголовок перед пересылкой за пределы домена, в котором находится прокси-сервер и тем самым не нарушит секретности переданной информации. Такой прокси-сервер обычно находится в домашней сети.

- **P-Charging-Function-Addresses**

В процесс предоставления доступа и обеспечения услуг вовлечено множество элементов распределенной архитектуры сети. Существует необходимость передать каждому SIP прокси-серверу, вовлечённому в транзакцию, информацию об адресе элементов SIP сети, решающих задачи по начислению платы. Это требуется для передачи на них записей сгенерированных прокси-серверами.

В 3GPP архитектуре определено два типа функциональных элементах, 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).

SIP прокси-сервер, который получает SIP запрос, может поместить в него заголовок P-Charging-Function-Addresses перед тем как осуществить пересылку запроса, это возможно только если в запросе не было заголовка с таким названием. Заголовок состоит из списка адресов одного или более элементов сети, ведущих начисление платы CCF или ECF, куда прокси-сервер должен отослать информацию, касающуюся начисления платы. Прокси-сервер, получивший сообщение с таким заголовком также будет знать, по какому адресу передавать записи, содержащие информацию для начисления платы. Агенты пользователя не должны осуществлять работу 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
```

- **P-Charging-Vector**

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

Идентификатор начисления платы определён, как совокупность информации о начислении платы. Информация может быть помещена в идентификатор несколькими сетевыми элементами (включая SIP прокси-серверы) и ими же 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 в глобальном масштабе. Для этого он должен включать две компоненты: локальное уникальное значение и доменное имя или IP-адрес SIP прокси-сервера, сгенерировавшего локальное значение. IOI идентифицирует обе сети, в которых находятся участники диалога или транзакции вне диалога.

Для переноса идентификатора определён заголовок P-Charging-Vector. SIP прокси-сервер в случае отсутствия заголовка P-Charging-Vector может поместить его самостоятельно в начальный запрос или ответ для диалога с теми параметрами, которые ему доступны. SIP прокси-сервер, 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, между которыми установлены доверительные отношения. Пример.

```
P-Charging-Vector: icid-value=1234bc9876e;
                   icid-generated-at=192.0.6.8;
                   orig-ioi=home1protei.ru
```

- **P-DCS-Trace-Party-ID**

В сети, предоставляющей услуги по передаче речевой информации, может быть предоставлена услуга трассировки пришедшего запроса - Customer originated trace, которая обеспечивают вызываемой стороне возможность обратиться к правоохранительным органам в случае злонамеренного телефонного вызова. В SIP эта услуга может обеспечиваться независимо от идентификатора вызывающего пользователя и работать даже, когда пользователь запрашивает услугу анонимности. Для возможности инициирования услуги трассировки, используемой для определения информации, идентифицирующей пользователя не пользующегося доверием UAC, в запрос INVITE помещается дополнительный заголовок P-DCS-Trace-Party-ID. Заголовок P-DCS-Trace-Party-ID не присутствует в других запросах и ответах.

Элемент, адресованный значением поля Request-URI, выполняет определённые 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, не использует заголовок P-DCS-Trace-Party-ID.

- **P-DCS-OSPS**

Некоторые вызовы имеют особые требования к обработке, которые не могут быть удовлетворены обычными агентами пользователя. Например, когда пользователь занят вызовом, а в это время приходит другой вызов, то такой вызов может быть отклонён с указанием занятости. Однако, некоторые виды услуг, предоставляемые операторами ТфОП, требуют особой обработки вызовов. В том числе сервисы «Проверка занятости линии» (BLV) и «Вмешательство в разговор в случае необходимости» (EI), инициированные оператором с Operator Services Position System (OSPS) на ТфОП. Для того, чтобы проинформировать SIP агента пользователя, что входящему вызову должно быть придано особое значение, используется заголовок P-DCS-OSPS. Его значение указывает на то, что запрашивается определённый способ обработки вызова.

На данное время существует три значения для данного заголовка: *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 передаёт 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 послан в запросе от UAC, не пользующегося доверием. Как правило заголовок вставляется контроллером медиа-шлюза (Media Gateway Controller). Пример.

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

- **P-DCS-BILLING-INFO**

В процессе предоставления услуг требуется производить сбор биллинговой информации и её доставку биллинговой системе. Эту функцию может выполнить SIP прокси-сервер. Более того, прокси-серверы задействованные в вызове, между которыми существуют доверительные отношения, могут обмениваться биллинговой информацией, относящейся к участникам вызова. Для записей об изменении состояния баланса (Accounting records) необходимо иметь идентификатор, который связывает все записи об услугах, предоставленных в процессе конкретного вызова. Заголовок P-DCS-BILLING-INFO содержит идентификатор, который может быть использован устройством, генерирующим учётные записи, для ассоциирования записей различного назначения, возможно от различных источников, с информацией, касающейся снятия платы со счёта (billable account). Также заголовок содержит информацию о состоянии счёта подписчика и другую информацию, необходимую для осуществления правильного биллинга обслуживания вызова. Этот заголовок используется только между прокси-серверами и пользующимися доверием UA – применим только в сегменте сети с доверительными отношениями и должен быть удалён при выходе за пределы сегмента.

- **P-DCS-LAES и P-DCS-REDIRECT**

Заголовок P-DCS-LAES содержит информацию, необходимую для выполнения легального электронного наблюдения (Lawfully Authorized Electronic Surveillance). Информация представляет собой адрес и порт устройства, занимающегося электронным наблюдением, для доставки двустороннего потока сигнальных сообщений, относящихся к этому вызову. Заголовок может также включать дополнительный адрес и порт для доставки элементу сети SIP, занимающемуся наблюдением, медиа-информации, передающейся в ходе вызова. Этот заголовок используется только между прокси-серверами и агентами пользователя, между которыми установлены доверительные отношения.

Заголовок P-DCS-Redirect содержит идентифицирующую вызов информацию, необходимую для поддержки требований легального электронного наблюдения для перенаправленных вызовов. Содержит первоначально указанный адрес, адрес перенаправленного запроса и количество предпринятых перенаправлений. Этот заголовок также используется только между прокси-серверами и агентами пользователя, между которыми установлены доверительные отношения.

- **RAck**

Заголовок RAck помещается в запрос PRACK в целях обеспечения надежной доставки предварительных ответов. Он включает два численных значения и поле типа запроса. Первое значение - 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 INVITE
```

- **Reason**

Один и тот же SIP запрос может быть передан по различным причинам. Например, запрос CANCEL может отсылаться прокси-сервером, размножающим запросы, если вызов состоялся на другой ветви или был прерван клиентом агентом пользователя до ответа вызываемой стороны. Несмотря на то, что протокол и поведение системы одинаково в обоих случаях, смысл передаваемого сообщения может существенно различаться. В реализациях, которых 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 также, если вызов будет принят на другом терминале пользователя.

Для создания сервисов, часто необходимо знать причину отправки SIP-запроса. Такую информацию обеспечивает заголовок Reason. Заголовок Reason также предназначен для инкапсуляции кода окончательного ответа в предварительный ответ. Это функция способствует решению проблемы, обозначаемой аббревиатурой HERFP (Heterogeneous Error Response Forking Problem) - ситуации, когда элемент, размножающий запросы, не

может отправить отклоняющий окончательный ответ, если не получил отклоняющий ответ с каждого из направлений, по которому отправил запрос.

Заголовок Reason может содержаться в любом запросе, передаваемом в режиме диалога, в любом запросе CANCEL и в любом ответе, чей код разрешает присутствие этого заголовка. Значение поля заголовка может быть *SIP* с параметром «cause», указывающим код SIP-ответа. Также значение может быть – «*Q.850»* с параметром «cause», информирующем о коде причины по которой было отослано сообщение в десятичном представлении в соответствии с рекомендацией ITU-T Q.850 для систем сигнализации DSS1 и ОКС№7 (подсистема ISUP). SIP-сообщение может содержать более одного значения поля заголовка Reason, однако в этом случае значения должны быть различны (например, одно *SIP* и другое *Q.850*). Реализации игнорируют значения заголовка Reason, которые им не понятны.

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

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

- **Record-Route**

Заголовок Record-Route вставляется прокси-серверами в запросы для того, чтобы последующие запросы в процессе диалога маршрутизировались через данные прокси-серверы.

Пример:

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

- **Refer-To**

Заголовок Refer-To применяется только в запросе REFER. Он указывает URI для передачи 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 он является важной частью механизма Route/Record-Route, описанного в разделе 2.4.1, и не может указывать адрес места назначения для пересылки вызова. Пример:

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

- **Reply-To**

Заголовок Reply-To содержит логический обратный URI, который может отличаться от адреса в поле заголовка From. Например, такой URI может использоваться для ответного звонка пользователя при упущенных вызовах и вызовах, не завершившихся установлением связи. В случае, если пользователь желает оставаться анонимным, заголовок Reply-To либо должен быть исключён из запроса, либо помещён таким образом, чтобы не разглашать личные сведения. 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>
```

- **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
```

- **Retry-After**

Заголовок Retry-After применяется в ответах с кодом 500 (Server Internal Error) или 503 (Service Unavailable), чтобы проинформировать 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) или 603 (Decline), чтобы указать, когда вызываемый пользователь будет снова доступен для вызова. 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 данный параметр отсутствует, подразумевается, что обслуживание будет предоставлено без ограничения времени. Для указания времени предполагаемого обратного звонка может быть использован опциональный комментарий.

Примеры:

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

- **Route**

Заголовок Route служит для принудительной маршрутизации запроса в соответствии со списком прокси-серверов. Процедуры, связанные с заголовком Route подробно рассмотрены в разделе 2.4.1.

Примеры:

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

- **RSeq**

Заголовок RSeq используется в предварительных ответах для их надёжной транспортировки. Он содержит численное значение в интервале от 1 до (2^{32}-1). Каждый надёжно передаваемый предварительный ответ (информационный ответ с подтверждением) содержит RSeq с порядковым номером ответа. Значение заголовка RSeq в каждом последующем надёжном предварительном ответе будет на единицу больше. Пример:

```
RSeq: 988789
```

- **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, находящемуся на расстоянии одной пересылки. Заголовок будет содержать список всех механизмов безопасности, поддерживаемые 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, отсылаемые 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 данное время определены следующие значения для заголовков: *digest*, *tls*, *ipsec-ike*, *ipsec-man*. Пример.

```
Security-Client: digest

Security-Server: tls;q=0.2

Security-Verify: ipsec-ike;q=0.1
```

- **Server**

Поле заголовка Server содержит информацию о программном обеспечении, которое используется сервером для обработки запросов. Раскрытие конкретной версии программного обеспечения сервера может облегчить атаки против программных продуктов, для которых известны уязвимые места. Разработчикам серверов рекомендуется сделать включение этого поля конфигурируемой опцией.

Пример:

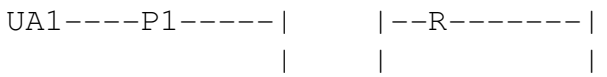
```
Server: HomeServer v2
```

- **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 проинформировать агента пользователя о так называемом «сервисном маршруте» (service route) т.е. маршруте, указывающем путь к серверам, предоставляющим услуги. Впоследствии UA может использовать эту информацию при исходящем вызове для того, чтобы запросить предоставление определённых услуг доменом, где находится используемый сервер регистрации.

Заголовок Service-Route направляет запросы через конкретную последовательность прокси-серверов. При использовании этого сервисного маршрута UA сможет воспользоваться услугами, предоставляемыми прокси-серверами, связанными с сервером регистрации. Значения заголовка Service-Route должны придерживаться синтаксиса, определённого для заголовка Route. В частности, значения должны содержать параметр «lr».

В приведённом ниже примере представлена домашняя сеть, состоящая из регистрирующего сервера (R), прокси-сервера, предоставляющего услуги HSP (home service proxy), базы данных (DBMS) и граничного прокси-сервера поставщика услуг, занимающегося маршрутизацией сообщений (P2). UA1 через исходящий прокси-сервер P1 посылает 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 отправке INVITE пользователю UA2 скопирует содержимое Service-Route в заголовок Route. Пример.



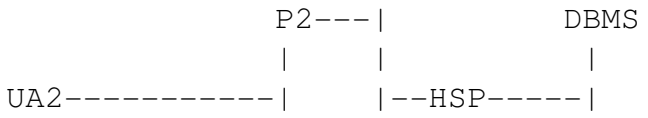


Рис 2.2. Пример сети, использующей заголовок Service-Route

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

- **Subject**

Заголовок Subject содержит дополнительную информацию о типе и характере сеанса связи, позволяя осуществлять фильтрацию вызов без анализа описания сессии.

Примеры:

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

- **Subscription-State**

Заголовок Subscription-State содержится в сообщении NOTIFY и указывает статус подписки (subscription). Значение заголовка может быть *active*, *pending* либо *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 так же включает параметр «expires», указывающий действительную продолжительность подписки; при этом подписчик должен скорректировать время действия подписки в соответствии с полученным значением.

Когда значение заголовка – *terminated*, в заголовке также присутствует параметры «reason» и «retry-after», информирующие о причине окончания подписки и времени, когда подписчик может попытаться создать новую подписку. Пример:

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

- **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 данный 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 вызовом по протоколу SIP и таймера сеанса.

- **Timestamp**

Заголовок Timestamp указывает, когда UAC отослал сообщение UAS. Заголовок позволяет расширениям или приложениям SIP получать оценку RTT (время двойного пробега).

Пример:

```
Timestamp: 54
```

- **To**

Заголовок To определяет логического адресата запроса. В случае присутствия отображаемого имени, оно должно быть с помощью пользовательского интерфейса предоставлено вызываемому пользователю. 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
```

- **Unsupported**

Заголовок Unsupported содержит перечень функциональных возможностей, неподдерживаемых UAS. Добавляется в ответ с кодом 420 (Bad Extension).

Пример:

```
Unsupported: 100rel
```

- **User-Agent**

Поле заголовка User-Agent несёт информацию о клиенте агента пользователя, инициирующего запрос. Раскрытие конкретной версии программного обеспечения UA может облегчить атаки против программных продуктов, для которых известны уязвимые места. Разработчикам UA рекомендуется сделать включение этого поля настраиваемой опцией.

Пример:

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

- **Via**

Поле заголовка Via содержит список элементов сети SIP, через которые запрос прошёл на данный момент. Список нужен для того, чтобы избежать ситуаций, в которых запрос пойдёт по замкнутому пути, а также для тех случаев, когда необходимо, чтобы запросы и ответы обязательно проходили по одному и тому же пути. В результате заголовок отображается 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, на который является предпочтительным для приёма ответов. Также в поле заголовка могут присутствовать параметры: «maddr», «ttl», «received» и «branch».

Возможно использование следующих транспортных протоколов: UDP, TCP, TLS и SCTP (TLS означает TLS поверх TCP). Когда запрос отсылается на SIPS URI, указывается прикладной протокол – SIP, а транспортный протокол – TLS.

```
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
```

В этом примере 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 обеим сторонам «:» 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 (например, в указанном выше примере это – `SIP/2.0/UDP serv3.niits.ru:4000`) - одинаковы, имеют один и тот же набор параметров, и значения параметров равны.

- **Warning**

Заголовок Warning содержит дополнительную информацию как правило связанную с проблемами обработки запроса сервером. Значения поля заголовка Warning отправляются вместе с ответами и содержат трёхзначный код, имя хоста и предупреждающий текст. Предупреждающий текст должен быть написан на национальном языке, наиболее лёгком для понимания конечного пользователя. Это решение может базироваться на любой доступной информации, такой как местоположение пользователя, поле заголовка Accept-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 сессии. Все предупреждающие коды начинаются с цифры «3», это указывает на предупреждения, характерные для SIP. Предупреждения с 300 по 329 предназначены для проблем с ключевыми словами (keywords) в описании сессии, с 330 по 339 относятся к основным сетевым сервисам, запрашиваемым в описании сессии, с 370 по 379 относятся к количественным параметрам качества обслуживания, также запрашиваемым в описании сессии, с 390 по 399 – разнообразные предупреждения, не подпадающие ни под одну из выше перечисленных категорий.

Таблица 2.2 Список предупреждений

Код предупреждения	Текст предупреждения	Расшифровка
300	Несовместимый сетевой протокол.	Один или несколько сетевых протоколов, указанных в описании сессии, не поддерживаются.
301	Несовместимые форматы сетевого адреса.	Один или несколько форматов сетевых адресов, указанных в описании сессии, не поддерживаются.
302	Несовместимый транспортный протокол.	Один или несколько транспортных протоколов, указанных в описании сессии, не поддерживаются.
303	Несовместимые единицы измерения пропускной способности.	Одна или несколько единиц измерения пропускной способности, указанных в описании сессии, были не поняты.
304	Тип медиа-информации не доступен.	Один или несколько типов медиа-информации, указанных в описании

		сессии, не доступны.
305	Несовместимый формат медиа-информации.	Один или несколько форматов медиа-информации, указанных в описании сессии, не доступны.
306	Атрибут не понят.	Один или несколько атрибутов медиа-информации, указанных в описании сессии, не поддерживаются.
307	Параметр описания сессии 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 сессии, 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 осуществлять никаких автоматических действий.

Дополнительные коды могут быть определены с согласия организации IANA.
Примеры:

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

- **WWW-Authenticate**

Заголовок WWW-Authenticate содержит запрос аутентификации, состоящий из поля, отображающего схему аутентификации, и ряд параметров, необходимый для проведения процедуры аутентификации с данным прокси-сервером для указанного Request-URI.

Данный заголовок включается в состав ответа с кодом 407 (Proxy Authentication Required) или с кодом 401 (Unauthorized). Более подробная информация о заголовке содержится в разделе 2.6.3.

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 SIPv2.0 и обработке их прокси-серверами отображена в таблице 2.3.

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 шесть колонок устанавливают присутствие заголовков в сообщениях различного 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.

- **T**

Заголовок должен вставляться при отсылке, но клиенты/серверы должны быть готовы принять сообщение без данного заголовка.

Если в качестве транспортного протокола используется потоко-ориентированный протокол, то заголовок должен быть вставлен в сообщение при отсылке.

- ***

Требуется наличие заголовка в случае, когда тело сообщения не пустое.

- **F**

Присутствие заголовка запрещено.

- **N/A**

Присутствие заголовка не определено.

Таблица 2.3 Связь заголовков с запросами и ответами протокола SIPv2.0.

Название заголовка	**Место использования заголовка**	**Действия прокси-сервера**	**ACK**	**BYE**	**CAN**	**INV**	**OPT**	**REG**	**IFO**	**SU**
Accept	**Заголовок в запросах**	**-**	**F**	**O**	**F**	**O**	**M***	**O**	**O**	**O**
Accept	**Заголовок в ответах 2xx**	**-**	**F**	**F**	**F**	**O**	**M***	**O**	**N/A**	**F**
Accept	**Заголовок в ответе 415**	**-**	**F**	**C**	**F**	**C**	**C**	**C**	**N/A**	**O**
Accept-Encoding	**Заголовок в запросах**	**-**	**F**	**O**	**F**	**O**	**O**	**O**	**O**	**O**
Accept-Encoding	**Заголовок в ответах 2xx**	**-**	**F**	**F**	**F**	**O**	**M***	**O**	**N/A**	**F**
Accept-Encoding	**Заголовок в ответе 415**	**-**	**F**	**C**	**F**	**C**	**C**	**C**	**N/A**	**O**
Accept-Language	**Заголовок в запросах**	**-**	**F**	**O**	**F**	**O**	**O**	**O**	**O**	**O**

Accept-Language	**Заголовок в ответах 2xx**	**-**	**F**	**F**	**F**	**O**	**M***	**O**	**N/A**	**F**
Accept-Language	**Заголовок в ответе 415**	**-**	**F**	**C**	**F**	**C**	**C**	**C**	**N/A**	**O**
Alert-Info	**Заголовок в запросах**	**a, r**	**F**	**F**	**F**	**O**	**F**	**F**	**N/A**	**F**
Alert-Info	**Заголовок в ответе 180**	**a, r**	**F**	**F**	**F**	**O**	**F**	**F**	**N/A**	**F**
Allow	**Заголовок в запросах**	**-**	**F**	**O**	**F**	**O**	**O**	**O**	**N/A**	**O**
Allow	**Заголовок в ответах 2xx**	**-**	**F**	**O**	**F**	**M***	**M***	**O**	**F**	**O**
Allow	**Заголовок в ответах**	**-**	**F**	**O**	**F**	**O**	**O**	**O**	**N/A**	**O**
Allow	**Заголовок в ответе 405**	**-**	**F**	**M**	**F**	**M**	**M**	**M**	**O**	**M**
Allow-Events	**Заголовок в запросах**	**-**	**O**	**O**	**F**	**O**	**O**	**O**	**N/A**	**O**
Allow-Events	**Заголовок в ответах 2xx**	**-**	**F**	**O**	**F**	**O**	**O**	**O**	**N/A**	**O**
Allow-Events	**Заголовок в ответе 489**	**-**	**F**	**F**	**F**	**F**	**F**	**F**	**F**	**M**
Authentication-Info	**Заголовок в ответах 2xx**	**-**	**F**	**O**	**F**	**O**	**O**	**O**	**N/A**	**O**
Authorization	**Заголовок в запросах**	**-**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**
Call-ID	**Общий заголовок/копируется из запросов в ответы**	**R**	**M**	**M**	**M**	**M**	**M**	**M**	**M**	**M**
Call-Info	**Общий заголовок**	**a, r**	**F**	**F**	**F**	**O**	**O**	**O**	**N/A**	**N/**
Contact	**Заголовок в запросах**	**-**	**O**	**F**	**F**	**M**	**O**	**O**	**O**	**M**
Contact	**Заголовок в ответах 1xx**	**-**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**O**
Contact	**Заголовок в ответах 2xx**	**-**	**F**	**F**	**F**	**M**	**O**	**O**	**F**	**M**
Contact	**Заголовок в ответах**	**D**	**F**	**O**	**F**	**O**	**O**	**O**	**F**	**M**

	3xx									
Contact	Заголовок в ответе 485	-	F	O	F	O	O	O	F	O
Contact	Заголовок в ответе 4xx-6xx	-	F	F	F	F	F	F	F	F
Content-Disposition	Заголовки содержания	-	O	O	F	O	O	O	N/A	O
Content-Encoding	Заголовки содержания	-	O	O	F	O	O	O	O	O
Content-Language	Заголовки содержания	-	O	O	F	O	O	O	N/A	O
Content-Length	Заголовки содержания	A, r	T	T	T	T	T	T	O	T
Content-Type	Заголовки содержания	-	*	*	F	*	*	*	*	*
Cseq	Общий заголовок/копируется из запросов в ответы	R	M	M	M	M	M	M	M	M
Date	Общий заголовок	A	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
Event	Заголовок в запросах	-	F	F	F	F	F	F	F	M
Expires	Общий заголовок	-	F	F	F	O	F	O	O	O
Expires	Заголовок в ответе 2xx	-	F	F	F	O	F	O	O	M
From	Общий заголовок/ копируется из запросов в ответы	R	M	M	M	M	M	M	M	M
In-Reply-To	Заголовок в запросах	-	F	F	F	O	F	F	N/A	F
Max-Forwards	Заголовок в запросах	a, m, r	M	M	M	M	M	M	O	M

Min-Expires	**Заголовок в ответе 423**	**-**	**F**	**F**	**F**	**F**	**F**	**M**	**N/A**	**M**
MIME-Version	**Общий заголовок**	**-**	**O**	**O**	**F**	**O**	**O**	**O**	**N/A**	**O**
Organization	**Общий заголовок**	**A, r**	**F**	**F**	**F**	**O**	**O**	**O**	**O**	**O**
Path	**Заголовок в запросах**	**a, r**	**F**	**F**	**F**	**F**	**F**	**O**	**F**	**F**
Path	**Заголовок в 2xx ответах**	**-**	**F**	**F**	**F**	**F**	**F**	**O**	**F**	**F**
Priority	**Заголовок в запросах**	**a, r**	**F**	**F**	**F**	**O**	**F**	**F**	**O**	**O**
Privacy	**Общий заголовок**	**a, m, r, d**	**O**	**O**	**O**	**O**	**O**	**O**	**O**	**O**
Proxy-Authenticate	**Заголовок в ответе 407**	**a, r**	**F**	**M**	**F**	**M**	**M**	**M**	**O**	**M**
Proxy-Authenticate	**Заголовок в ответе 401**	**a, r**	**F**	**O**	**O**	**O**	**O**	**O**	**N/A**	**N/**
Proxy-Authorization	**Заголовок в запросах**	**d, r**	**O**	**O**	**F**	**O**	**O**	**O**	**O**	**O**
Proxy-Require	**Заголовок в запросах**	**a, r**	**F**	**O**	**F**	**O**	**O**	**O**	**O**	**O**
P-Access-Network-Info	**Общий заголовок**	**D, r**	**F**	**O**	**F**	**O**	**O**	**O**	**O**	**O**
P-Asserted-Identity	**Общий заголовок**	**A, d, r**	**F**	**O**	**F**	**O**	**O**	**F**	**F**	**O**
P-Associated-URI	**Заголовок в ответах 2xx**	**-**	**F**	**F**	**F**	**F**	**F**	**O**	**F**	**F**
P-Called-Party-ID	**Заголовок в запросах**	**A, m, r**	**F**	**F**	**F**	**O**	**O**	**F**	**F**	**O**
P-Charging-Vector	**Общий заголовок**	**A, d,m,r**	**F**	**O**	**F**	**O**	**O**	**O**	**O**	**O**

P-Charging-Function-Addresses	**Общий заголовок**	**A, d, r**	**F**	**O**	**F**	**O**	**O**	**O**	**O**	**O**
P-DCS-Billing-Info	**Общий заголовок**	**A, d, m, r**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-DCS-LAES	**Общий заголовок**	**A, d, r**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-DCS-Redirect	**Общий заголовок**	**A, d, r**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-DCS-OSPS	**Заголовок в запросах**	**D, r**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-DCS-Trace-Party-ID	**Заголовок в запросах**	**D, r**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-Media-Authorization	**Заголовок в запросах**	**A, d**	**O**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-Media-Authorization	**Заголовок в ответах 2xx**	**A, d**	**F**	**F**	**F**	**O**	**F**	**F**	**F**	**F**
P-Media-Authorization	**Заголовок в ответах 101-199**	**A, d**	**F**	**F**	**F**	**O**	**F**	**F**	**N/A**	**N/**
P-Preffered-Identity	**Общий заголовок**	**A, d, r**	**F**	**O**	**F**	**O**	**O**	**F**	**F**	**O**
P-Visited-Network-ID	**Заголовок в запросах**	**A, d**	**F**	**F**	**F**	**O**	**O**	**O**	**F**	**O**
RAck	**Заголовок в запросах**	**-**	**F**	**F**	**F**	**F**	**F**	**F**	**F**	**F**
Reason	**Заголовок в запросах и 1xx ответах**	**-**	**O**	**O**	**O**	**O**	**F**	**F**	**O**	**O**
Record-Route	**Заголовок в запросах**	**A, r**	**O**	**O**	**O**	**O**	**O**	**F**	**N/A**	**O**
Record-Route	**Заголовок в ответах 18x**	**M, r**	**F**	**O**	**O**	**O**	**O**	**F**	**N/A**	**F**
Record-Route	**Заголовок в ответах 2xx**	**M, r**	**F**	**O**	**O**	**O**	**O**	**F**	**N/A**	**O**
Record-Route	**Заголовок в ответах**	**M, r**	**F**	**F**	**F**	**F**	**F**	**F**	**F**	**O**

	401, 484									
Refer-To	Заголовок в запросах	-	F	F	F	F	F	F	F	F
Reply-To	Общий заголовок	-	F	F	F	O	F	F	N/A	F
Require	Заголовок в запросах	a, r	F	C	F	C	C	C	O	O
Retry-After	Заголовок в ответах 404, 413, 480, 486, 500, 503, 600 и 603	-	F	O	O	O	O	O	O	O
Route	Заголовок в запросах	a, d, r	C	C	C	C	C	C	O	C
RSeq	Заголовок в ответе 1xx	-	F	F	F	O	F	F	F	O
Security-Client	Заголовок в запросах	A, r, d	F	O	F	O	O	O	O	O
Security-Server	Заголовок в ответах 421, 494	-	F	O	F	O	O	O	O	O
Security-Verify	Заголовок в запросах	A, r, d	F	O	F	O	O	O	O	O
Server	Заголовок в ответах	-	F	O	O	O	O	O	O	O
Subscription-State	Заголовок в запросах	-	F	F	F	F	F	F	F	F
Subject	Заголовок в запросах	-	F	F	F	O	F	F	O	F
Supported	Заголовок в запросах	-	F	O	O	M*	O	O	N/A	O
Supported	Заголовок в ответе 2xx	-	F	O	O	M*	M*	O	N/A	O
Timestamp	Общий заголовок	-	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
Unsupported	Заголовок в ответе 420	-	F	M	F	M	M	M	O	O
User-Agent	Общий заголовок	-	O	O	O	O	O	O	O	O

Via	**Заголовок в запросах**	**a, m, r**	**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**
Warning	**Заголовок в запросах**	**-**	**F**	**O**	**O**	**O**	**O**	**O**	**O**	**F**
Warning	**Заголовок в ответах**	**-**	**F**	**O**	**O**	**O**	**O**	**O**	**O**	**O**
WWW-Authenticate	**Заголовок в ответе 401**	**a, r**	**F**	**M**	**F**	**M**	**M**	**M**	**O**	**M**
WWW-Authenticate	**Заголовок в ответе 407**	**a, r**	**F**	**O**	**F**	**O**	**O**	**O**	**N/A**	**N/**

2.2.3. Назначение и формат запросов

SIP-запросы характеризуются наличием Request-Line в стартовой строке. Request-Line состоит из названия типа запроса, Request-URI и версии протокола, разделённых 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. См. RFC 822) не допускается.

Тип запроса	**Пробел**	**Request-URI**	**Пробел**	**Версия протокола**	**CRLF**

Рис 2.3 Структура Request-Line

- **Тип запроса**

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

- **Request-URI**

Request-URI – это SIP или SIPS URI. Он указывает на пользователя или сервис, к которому адресован данный запрос. Поле Request-URI не должно содержать пробелов и управляющих символов, а также быть заключённым 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 указан публичный адрес получателя, а в Request-URI – адрес прокси-сервера, через который проходит запрос.

- **Версия протокола**

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

Запрос INVITE - приглашает вызываемого пользователя принять участие в сеансе связи. Сообщение обычно содержит описание сессии, в котором передается вид принимаемой информации и параметры (список возможных вариантов параметров), необходимые для приема информации, также может указываться вид информации, который вызываемый пользователь желает передавать. В ответе на запрос INVITE указывается вид информации, которая будет приниматься вызываемым пользователем, кроме того, может указываться вид информации, которую вызываемый пользователь собирается передавать (возможные параметры передачи информации).
В этом сообщении могут содержаться также данные необходимые для аутентификации абонента, и, следовательно, доступа клиентов к SIP-серверу. В случае необходимости изменения характеристик уже установленных каналов посылается запрос INVITE с новым

описанием сессии. Для приглашения нового участника к уже установленному соединению также используется сообщение INVITE, которое отсылается его агенту пользователя.
Запрос 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 сессии, передаваемое вызывающим пользователем.
Запрос 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 пользователей по нескольким направлениям и по одному из них его находит. Обработку запросов, разосланных по всем остальным направлениям, сервер отменяет при помощи команды CANCEL.
Сообщением BYE оборудование вызываемого или вызывающего пользователей завершает соединение. Сторона, получившая запрос BYE, должна прекратить передачу речевой (мультимедийной) информации и подтвердить ее выполнение ответом 200 (ОК).
При помощи команды 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 посылаться все дальнейшие запросы **INVITE**. Если в команде **REGISTER** поле **Contact** отсутствует, то регистрация остается неизменной. В случае отмены регистрации здесь размещается символ “*”.
- **Expires** указывается время в секундах, по истечении которого регистрация заканчивается. Если данное поле отсутствует, то по умолчанию назначается время – 1 час, после которого регистрация завершается. Регистрацию можно также отменить посылкой сообщения **REGISTER** с полем **Expires**, которому присвоено значение 0 и соответствующим полем **Contact.**

Сообщением **OPTIONS** вызываемый пользователь запрашивает информацию о возможностях терминального оборудования вызываемого пользователя. В ответ на запрос оборудование вызываемого пользователя возвращает требуемую информацию. Применение запроса регламентировано теми случаями, когда существует необходимость узнать о поддерживаемых возможностях оборудования до установления соединения. Запрос не используется для установления соединения.
После завершения описания запросов протокола SIP, в качестве примера приведем типичный запрос INVITE (рис. 2.4.).

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

         v=0
         o=bell 53655765 2353687637 IN IP4 128.3.4.5
         C=IN IP4 kton.loniis.ru
         m=audio 3456 RTP/AVP 0 3 4
```

Рис 2.4. Пример запроса INVITE

В этом примере пользователь Anton (anton@loniis.ru) вызывает пользователя Alexander (alexander@loniis.ru). Запрос передается на прокси-сервер (serv1.loniis.ru). В полях To и From перед адресом стоит надпись, которую вызывающий пользователь желает вывести на дисплее у вызываемого пользователя. В теле сообщения оборудование вызывающего пользователя указывает в формате протокола SDP, что оно может принимать в порту 3456 речевую информацию, упакованную в пакеты RTP речевую информацию, закодированную по одному из алгоритмов кодирования: 0 – PCMU, 3 – GSM и 4 – G.723.

После апробирования протокола SIP на реальных сетях оказалось, что для решения ряда задач вышеуказанных шести запросов недостаточно. Поэтому организацией IETF вводятся новые сообщения протокола. Ими являются: INFO, PRACK, UPDATE, SUBSCRIBE, NOTIFY, REFER, MESSAGE.

INFO

Тип запроса INFO предназначен для обмена сигнальной информацией по SIP сигнальному тракту в процессе установления и поддержания соединения. Запрос INFO не изменяет состояния SIP вызовов, также как не изменяет состояния сессий, 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 данной информации.
Сигнальным трактом для запроса INFO является сигнальное соединение, созданное в результате выполнения процедуры установления соединения для конкретного вызова. Им может быть тракт сигнализации непосредственно между вызывающим и вызываемым агентами пользователя или сигнальный тракт с участием SIP прокси-серверов, которые были вовлечены в процедуру установления соединения и которые поместили своё значение заголовка Record-Route в начальное сообщение INVITE.
В ходе сессии информация может быть передана или в заголовке сообщения INFO, или в части тела сообщения.
Возможными применениями для типа запроса INFO являются:

- Перенос текущих сигнальных сообщений ТфОП между шлюзами ТфОП в течении разговорной сессии
- Перенос DTMF сигналов, сгенерированных в ходе сессии
- Перенос защищённой информации сигналов беспроводных систем для поддержки беспроводной мобильности приложений
- Перенос информации об остатке на счёте (билинговой информации)
- Перенос изображений и другой не потоковой информации между участниками сессии

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 отослать окончательный ответ. Если запрос в не зависимости от наличия тела сообщения был успешно принят сервером для существующего вызова, то должен быть отправлен ответ с кодом 200 (OK). Когда формат тела сообщения неизвестен серверу, передаётся ответ с кодом 415 (Unsupported Media Type).

Запрос INFO может быть отменён. UAS, получивший сообщение CANCEL для отмены INFO, должен возвратить ответ с кодом 487 (Request Cancelled), если до этого времени 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 RFC 2976.

PRACK

Протокол SIP определяет два типа ответов на запрос, инициирующий соединение – предварительные и окончательные. Окончательные ответы передают результат обработки запроса и отсылаются надёжно (с подтверждением). Предварительные ответы обеспечивают информацию о текущей стадии обработки запроса, но отсылаются ненадёжно. Однако, в некоторых случаях, включающих взаимодействие с ТфОП, необходим механизм обеспечения

надёжности передачи предварительных ответов. В целях поддержки функции надёжной транспортировки предварительных ответов требуется наличие у элемента SIP соответствующей опции (идентифицирующейся с помощью **option tag** «100rel»).

Механизм надёжности работает по схеме, сходной с существующими механизмами надёжности для окончательных ответов класса 2xx на запрос INVITE. Надёжные предварительные ответы повторно передаются TU (пользователем транзакций) с интервалом, возрастающим по экспоненциальному закону. Повторные передачи прекращаются, когда приходит сообщение PRACK. Тип запроса PRACK играет ту же роль, что и ACK, но для предварительных ответов. Однако между ними имеется принципиальное отличие – PRACK является обычным SIP-сообщением, как запрос BYE. Т.е. его надёжность обеспечивается последовательно (hop-by-hop) при прохождении через каждый stateful прокси-сервер. Также как BYE (но в отличие от ACK) PRACK требует отправки ответа при его получении.

На устанавливающий диалог запрос UAS может послать ответ класса 1xx (кроме 100) надёжно при условии, что запрос содержит заголовок Supported с **option tag** «100rel». Это означает, что UAC поддерживает расширение для надёжной передачи предварительных ответов и может принимать и обрабатывать такие ответы. В то же время UAC при создании запроса может потребовать надёжной доставки предварительных ответов на него. Для этого UAC вставляет 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 передаёт его серверной транзакции для отсылки. Предварительный ответ устанавливает диалог, если до сих пор он не был установлен.

При получении надёжного ответа UAC создаёт запрос PRACK и передаёт его в диалоге, находящемся на ранней стадии, который связан с предварительным ответом. Сообщение PRACK включает заголовок RAck, который указывает порядковый номер подтверждаемого предварительного ответа. Повторные передачи PRACK не зависят от получения ответов класса 1xx, т.е. UAC не должен повторно передавать запрос PRACK, когда получает повторную передачу уже подтверждённого предварительного ответа.

Повторные передачи надёжного предварительного ответа прекращаются при получении ядром UAS запроса PRACK. Если PRACK соответствует неподтверждённому предварительному ответу, то отсылается ответ класса 2xx.

После того, как надёжный предварительный ответ был подтверждён, UAS может передавать дополнительные надёжные предварительные ответы. Значение заголовка RSeq в каждом последующем предварительном ответе будет на единицу больше.

Дополнительные сведения о запросе PRACK содержатся в RFC 3262. Пример.

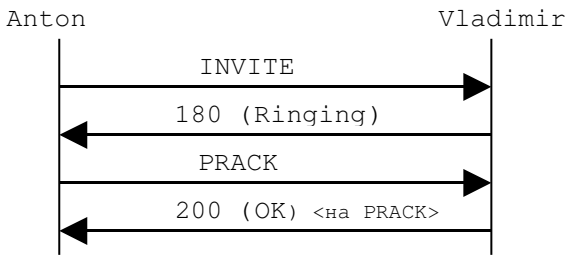


Рис. 2.5. Пример использования запроса PRACK

UPDATE

Часто возникают случаи, когда необходимо изменить некоторые параметры сессии (например, кодеки) до прихода окончательного ответа на начальное сообщение INVITE. Например, предответное состояние (early media) - ситуация, когда сессия устанавливается в целях передачи информации о текущем состоянии вызова до того, как на запрос INVITE будет отправлен сервером ответ. Важно, чтобы и вызывающая, и вызываемая стороны могли изменять параметры сессии до того как поступит ответ на вызов. Запрос re-INVITE (INVITE, передаваемый в ходе сессии См. раздел 2.3.7) не может быть использован для этих целей, поскольку re-INVITE оказывает влияние не только на состояние сессии, но и на состояние диалога. В противоположность ему запрос UPDATE может быть отослан агентом пользователя в режиме диалога (находящегося на ранней стадии или установленного) для обновления параметров сессии без воздействия на состояние диалога. Запрос UPDATE используется следующим образом. Вызывающая сторона создаёт начальную INVITE-транзакцию. В поле заголовка Allow запроса INVITE среди прочих типов запроса указывается UPDATE для того, чтобы указать 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 нём значением *UPDATE*. После установления 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), предназначенное для обновления параметров сессии. 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 отослать запрос UPDATE с информацией **offer**, а вызывающая сторона поместит answer в ответ класса 2xx на UPDATE. Если UA получает не-2xx окончательный ответ на UPDATE, параметры сессии должны оставаться неизменными.

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

Запрос UPDATE является запросом, обновляющим текущий адрес удалённого пользователя (target refresh request).

Дополнительные сведения о запросе UPDATE содержатся в RFC 3311. Пример:

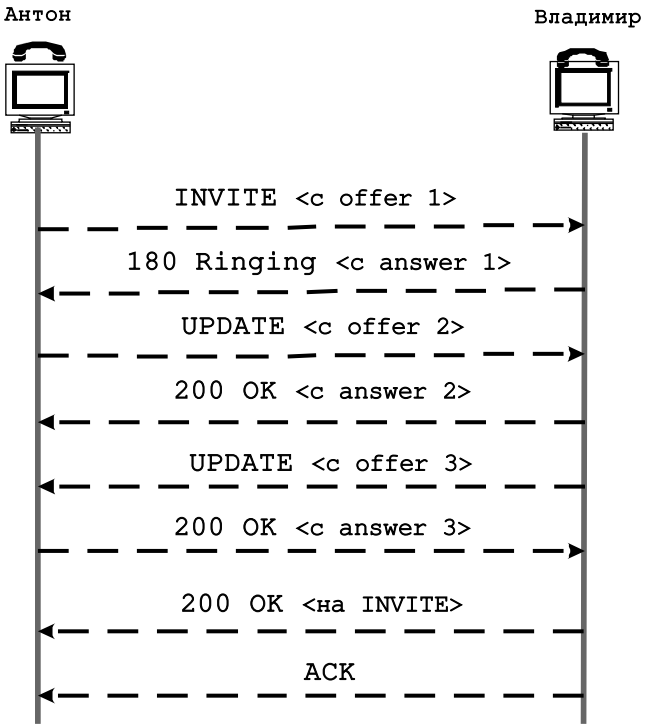


Рис. 2.6 Пример использования запроса UPDATE

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: услуги автоматического обратного вызова (связанные с событиями изменения состояния терминала пользователя), интерактивные списки контактов «buddy lists» (связанные с событиями присутствия пользователя), оповещение об ожидающем сообщении (связанные с событиями изменения состояния почтового ящика) и передача информации о состоянии вызова при взаимодействии сетей Internet и ТфОП.

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

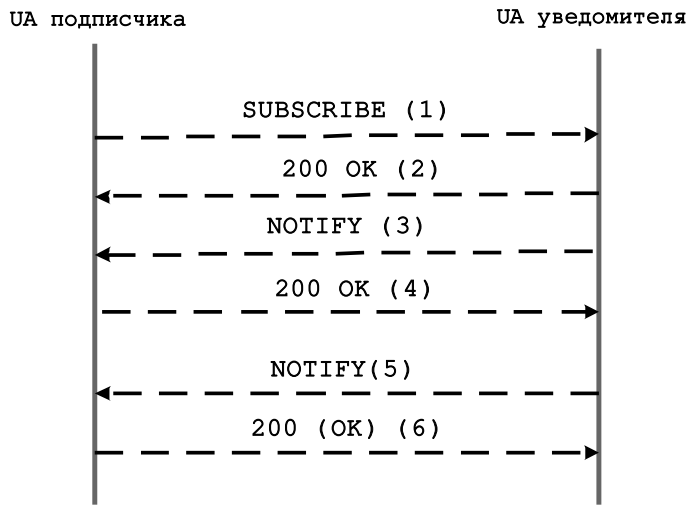


Рис. 2.7 Процедура подписки на предоставление информации

(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

Запрос SUBSCRIBE используется для запроса информации о текущем состоянии и информации об обновлениях состояния удалённого ресурса. SUBSCRIBE – это запрос, создающий диалог. Когда подписчик (subscriber – агент пользователя, запрашивающий и получающий информацию о состоянии ресурса) желает подписаться на получение информации о конкретном состоянии ресурса, он формирует сообщение SUBSCRIBE. Если начальное сообщение SUBSCRIBE представляет собой запрос вне диалога, как зачастую и бывает, его формирование проходит с привлечение процедур по созданию запроса вне диалога для UAC.

Идентификация событий, на которые осуществляется подписка, обеспечивается с помощью трёх компонентов запроса SUBSCRIBE: поля Request URI, заголовка Event и опционально тела сообщения.

- Request URI содержит исчерпывающую информацию для идентификации ресурса, для 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 состояния присутствия пользователя (user presence state), так и для подписки на информацию состояния ящика голосовой почты пользователя).

- В запросе SUBSCRIBE должен присутствовать ровно один заголовок Event, указывающий событие или класс событий, на которое осуществляется 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 для соответствующей подписки путём отсылки ещё одного сообщения SUBSCRIBE с таким же значением параметра «id» заголовка Event в том же диалоге, где существует начальная подписка. Также пользователь может отказаться от подписки. Процедура отказа протекает так же, как и процедура обновления подписки с единственным отличием, заключающимся в том, что значение поля заголовка Expires устанавливается в нулевое состояние.

Запрос SUBSCRIBE должен быть подтверждён окончательным ответом. При этом уведомитель (notifier – агент пользователя, информирующий пользователя о состоянии ресурса) не должен ожидать подтверждения пользователя, прежде чем возвратить окончательный ответ. Если уведомитель способен незамедлительно определить, что он поддерживает **event package**, что аутентифицированный подписчик (subscriber) авторизован на подписку и что не существует других препятствий, чтобы создать подписку, он создаёт подписку и в случае необходимости диалог а затем возвращает ответ с кодом 200 (OK). Также может быть отослан ответ с кодом 202 (Accepted). Такой ответ означает, что запрос был получен и понят, но подписка может быть ещё не авторизована.

После того, как подписка была успешно создана или обновлена, уведомитель должен незамедлительно отослать сообщение NOTIFY, чтобы сообщить подписчику текущее состояние ресурса. Запрос NOTIFY посылается в том же диалоге, который был создан ответом на запрос SUBSCRIBE (если не существовало заранее установленного диалога). Когда происходит изменение в состоянии, на которое была открыта подписка, подписчику также направляется 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 также может содержать подробности, связанные с данным 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 обеспечивают дополнительную информацию о типе произошедшего события и новом состоянии ресурса.

Сообщение NOTIFY должно содержать заголовок Subscription-State со значением либо *active*, либо *pending*, либо *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.

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 принимает уведомление, подписчик должен возвратить ответ с кодом 200 (OK). Дополнительные сведения о запросах SUBSCRIBE и NOTIFY даны в RFC 3265.

REFER

Запрос REFER, посылаемый 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 передачу вызова (Call Transfer). Например, если Anton находится в состоянии вызова с пользователем Vladimir и решает, что Vladimir должен поговорить с пользователем Alexander, Anton может поручить своему SIP UA отправить запрос REFER SIP агенту пользователя Vladimir, снабдив его контактной информацией пользователя Alexander. Если Vladimir даст своё согласие, его UA попытается связаться с Alexander, используя контактный адрес. После этого UA пользователя Vladimir уведомит UA пользователя Anton, насколько успешно прошла попытка установления связи с пользователем Alexander.

Если не определено иначе, правила для отсылки 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 возвратить ответ с кодом 202 (Accepted) до того, как истечёт время действия REFER-транзакции. Вслед за этим UA получателя создаёт подписку. Подписка, создаваемая запросом REFER по своей сути является такой же, как подписка, создаваемая запросом SUBSCRIBE. REFER – это только механизм, который создаёт подписку на событие передачи вызова - «refer». Создание подписки влечёт за собой незамедлительную отправку запроса NOTIFY. Механизм отправки сообщений NOTIFY используется для информирования UA, передавшего REFER, о статусе пересланного вызова. Идентифицирующие диалог значения заголовков To, From и Call-ID для каждого сообщения NOTIFY должны соответствовать аналогичным заголовкам в запросе REFER так, как если бы вместо REFER был запрос SUBSCRIBE. Каждое сообщение NOTIFY должно содержать заголовок Event со значением *refer*. Также NOTIFY содержит тело сообщения типа message/sipfrag (тип тела сообщения message/sipfrag подробно описан в RFC 3420), содержащее исчерпывающую информацию о состоянии действия по пересылке вызова; также в теле сообщения может содержаться дополнительная информация о результате действия.

Запрос NOTIFY может быть создан всякий раз, когда появляется новая информация о статусе события пересылки вызова. Уведомления на событие «refer» не могут отсылаться чаще, чем один раз в секунду. Также возможен такой вариант, когда за время подписки передаётся ровно два сообщения NOTIFY – безотлагательный запрос NOTIFY и NOTIFY,

содержащий окончательный результат пересылки вызова. Окончательный запрос NOTIFY содержит заголовок Subscription-State со значением *terminated* (подписка окончена) и со значением параметра «reason» - *noresource*. Ниже представлен пример обмена сообщениями между Agent A и Agent B при успешной пересылке вызова, где Agent B пересылает вызов кому-либо. Дополнительные сведения о запросе REFER даны в RFC 3515.

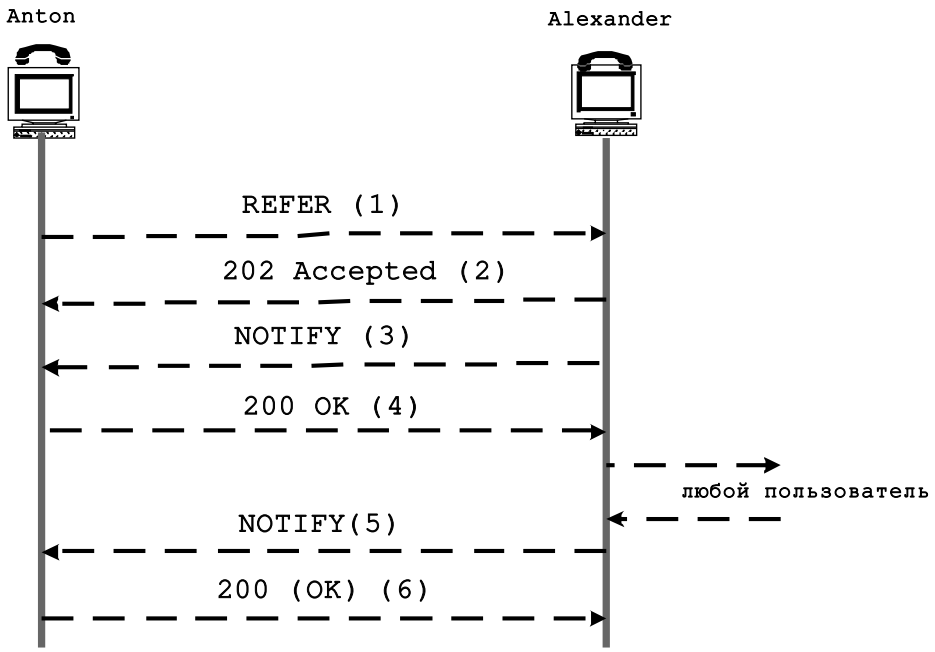


Рис. 2.8 Пример использования запроса REFER

Содержание сообщений F1 и F5

```
Сообщение 1 (F1)

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



Сообщение 5 (F5)

NOTIFY sip:anton@loniis.ru SIP/2.0
Via: SIP/2.0/UDP alx.loniis.ru;branch=z9hG4bK9323394234
To: <sip:anton@loniis.ru>;tag=193402342
From: <sip:alexander@loniis.ru>;tag=4992881234
Call-ID: 898234234@ant.loniis.ru
```

```
CSeq: 1993403 NOTIFY
Max-Forwards: 70
Event: refer
Subscription-State: terminated;reason=noresource
Contact: sip:alexander@loniis.ru
Content-Type: message/sipfrag;version=2.0
Content-Length: 16
```

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 IM-сообщения отличается тем, что они обычно связаны с короткими сеансами взаимодействия (группируются вокруг них), которые состоят из большого числа коротких сообщений, отсылаемых в обе стороны.

В SIP запрос типа MESSAGE предназначен для передачи мгновенных текстовых сообщений (instant massages), используя модель, походящую на функционирование двустороннего пейджера или работу телефона при отправке SMS. То есть не существует прямой связи между сообщениями. Каждое текущее сообщение (IM) независимо – информация о том, что происходит взаимодействие, переговоры между участниками, может быть отражена только в пользовательском интерфейсе клиента или возможно в воображении самого пользователя. Такой подход полностью противоположен сессионной модели, где происходит однозначное взаимодействие участников с чётко определённым началом и концом.

Когда один пользователь решает послать другому пользователю текущее текстовое сообщение (IM), отправитель формирует запрос типа MESSAGE и передаёт его. Поле Request-URI запроса как обычно будет указывать публичный 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 совмещён с системой присутствия, которая при вводе публичного адреса выдаёт новейшую контактную информацию устройства, где в данный момент находится пользователь. Тело сообщения будет включать 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 тоже 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 пройдёт через группу прокси-серверов и будет доставлен получателю. Получив запрос, UA получателя перейдёт к его обработке и в случае успеха в итоге отошлёт окончательный ответ с кодом 200 (OK); это означает, что текстовое сообщение было доставлено пользователю, но указывает на то, что пользователь с ним ознакомился.

Запросы MASSAGE не устанавливают диалога. UAC может передавать запрос

MESSAGE в существующем диалоге. В этом случае запрос также происходит в рамках сессии или сессий, связанных с этим диалогом.

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 конкретный URI в зависимости от протокола (в данном случае SIP URI). То есть если в качестве адреса UA-получателя сообщения представлен IM URI, он должен быть переведён в SIP URI и помещён в поле Request-URI запроса MESSAGE перед отправкой.

Ниже представлен пример обмена сообщениями. В примере Anton передаёт 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 RFC 3428.

Рис. 2.9 Пример использования запроса MESSAGE

Содержание сообщения F1:

```
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: 18

Александр, иди сюда.
```

В заключении все запросы с их кратким описанием сведены в таблицу 2.4.

Таблица 2.4 Типы запросов SIP.

Название запроса	Описание запроса
INVITE	Приглашает пользователя к сеансу связи.
ACK	Подтверждает прием окончательного ответа на запрос INVITE
BYE	Завершает вызов. Посылаться любой из сторон, участвующих в соединении.
CANCEL	Отменяет обработку запросы 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 управляющую и др. информацию для разговорной сессии по сигнальному тракту
PRACK	Используется для надёжной транспортировки предварительных ответов
UPDATE	Служит для изменения параметров сессии до прихода окончательного ответа (без воздействия на состояние диалога).
SUBSCRIBE	Запрашивает информацию о текущем состоянии и информацию об обновлении состояния удалённого ресурса
NOTIFY	Сообщает подписчику текущее состояние ресурса и уведомляет о том, что запрашиваемое событие произошло
REFER	Предписывает получателю связаться с третьей стороной, используя контактную информацию, которая содержится в сообщении
MESSAGE	Переносит мгновенные текстовые сообщения (instant massages)

2.2.4. Назначение и формат ответов на запросы

Характерное отличие SIP-ответов от запросов – это наличие строки состояния Status-

Line в стартовой строке. Status-Line составляют: версия протокола и код ответа (Status-Code) со связанной с ним текстовой расшифровкой (Reason-Phrase), разделённые пробелом (SP). Символы возврата каретки (CR) и перевода строки (LF) могут использоваться только совместно в завершающей строку последовательности CRLF.

Status-Line	**Версия протокола**	**Пробел**	**Status-Code**	**Пробел**	**Reason-Phrase**	**CRLF**

Рис. 2.10 Структура Status-Line

Код ответа – это целое трёхзначное число, отражающее результат обработки запроса сервером. Reason-Phrase даёт 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 в праве 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 упорядочивания кодов в пределах категории. В некоторых случаях оборудование даже необязательно знать все коды ответов, но оно обязательно должно интерпретировать первую цифру ответа.

После принятия и интерпретации запроса, адресат (прокси-сервер) 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 различную 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.

Информационные или предварительные ответы (1xx)

Информационные или предварительные ответы содержат информацию о том, что запрашиваемый сервер находится на стадии выполнения действий по обработке запроса и не может в данный момент выдать окончательный ответ. Сервер посылает ответ класса 1xx, если он предполагает, что формирование окончательного запроса займёт более 200 мс.

Предварительные ответы не требуют от клиента посылки 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, содержащие описание сессий.

Таблица 2.5. Информационные ответы протокола SIP

КОД	НАЗНАЧЕНИЕ
100	**Trying.** Запрос обрабатывается. Например, сервер обращается к базе данных, но местоположение вызываемого пользователя в настоящий момент не определено. Этот ответ так же, как и прочие предварительные ответы, пресекает повторные посылки сообщения INVITE клиентом. От остальных 1xx ответов, его отличает то, что он не может быть пропущен stateful прокси-сервером. Будет, вероятнее всего, передаваться от прокси-сервера или другого промежуточного SIP-сервера в сигнальном тракте соединения.
180	**Ringing.** Местоположение вызываемого пользователя определено. Вызываемый пользователь получает сигнал о входящем вызове от своего UA.
181	**Call Is Being Forwarded.** Прокси-сервер переадресует вызов к другому пользователю. Если прокси-сервер передает этот ответ, он может также указывать в теле сообщения, к какому пользователю он переадресует вызов.
182	**Queued.** Вызываемый пользователь временно не доступен, но входящий вызов поставлен в очередь. Когда вызываемый пользователь станет доступен, он передаст окончательный ответ. Reason-Phrase может дать более детальное описание текущего состояния вызова, например, «В очереди находится 5 вызовов. Предполагаемое время ожидания 15 минут». Сервер периодически будет посылать ответы 182 (Queued) для информирования вызывающего пользователя о статусе вызова, находящегося в очереди.
183	**Session Progress.** Этот ответ используется для того, чтобы заранее получить описание сессии информационного обмена от шлюзов на пути к вызываемому пользователю таким образом, чтобы мог быть проключен речевой тракт в предответном состоянии ещё 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 перед установлением соединения.

Ответы успешной обработки запроса (2xx)

Ответы класса **2xx** означают, что запрос был успешно обработан.

Таблица 2.6 SIP-Ответы успешной обработки 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 вызова, 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 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 отклонён. Значение ответа 202 намеренно неопределённое. 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.

Ответы перенаправления вызова (3xx)

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

Таблица 2.7. 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 сообщение, включающее список доступных ресурсов с основными характеристиками и их местоположение. 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 переадресовывает вызов (а предлагает только перенаправление). Поле Request-URI переадресованного запроса имеет то же значение, что и в 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 напрямую, 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 передаёт только UAS.
380	**Alternative Service.** Запрошенная услуга недоступна, но доступны альтернативные варианты обслуживания, которые описаны в теле сообщения ответа.

Ответы ошибки в запросе (4xx)

Ответы класса **4xx** информируют о том, что в запросе обнаружена ошибка. После получения такого ответа пользователь не должен передавать тот же самый запрос, на который получен ответ **4xx**, без его модификации.

Таблица 2.8 SIP-ответы ошибки в запросе

КОД	НАЗНАЧЕНИЕ
400	**Bad Request.** В запросе обнаружена синтаксическая ошибка. Это означает, что запрос не понят на дальнем конце. 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 реализации.

401	**Unauthorized.** Запрос требует проведения процедуры аутентификации пользователя. Этот ответ посылает UAS или registrar. Когда получен этот ответ, к форматированию сообщений применяются специальные правила.
402	**Payment Required.** Требуется предварительная оплата услуг. Ответ с данным кодом зарезервирован 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 возможных типов запросов 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 выдавать занятые 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 передаётся 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, то он посылает ответ с кодом 404 (Not Found).
413	**Request Entity Too Large.** Размер запроса слишком велик для обработки на сервере. Сервер может завершить соединение, чтобы прекратить приём 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 большой.
415	**Unsupported Media Type.** Сервер не может принять запрос из-за того, что формат содержимого тела сообщения не поддерживается сервером для данного типа запроса. 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 не понятна.

420	**Bad Extension.** Сервер не понимает расширение протокола SIP, которое содержится в поле заголовка Proxy-Require или Require. Сервер должен поместить список неподдерживаемых расширений в заголовок Unsupported ответа.
421	**Extension Required.** Для правильной обработки запроса UAS вынужден применить определённое расширение, однако оно не указано в поле заголовка Supported запроса. Ответы с этим кодом могут содержать заголовок Require со списком требуемых расширений. Сервер посылает 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 отсылки 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, обновлённого запросом, слишком короткое. Данный ответ использует registrar, чтобы отклонить сообщение регистрации, для заголовка Contact которого время действия очень мало.
480	**Temporarily Unavailable.** Соединение с оконечной системой было установлено успешно, но пользователь в данное время не доступен (к примеру, находится вне системы или находится в системе, но в состоянии, препятствующем установлению соединения с вызывающим абонентом, или активировал 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 очень полезно 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, то 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 хорошая защита от длинных и нестабильных маршрутов.
484	**Address Incomplete.** Принят запрос с неполным адресом в поле Request-URI. Дополнительная информация представлена в тестовом 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 данный запрос. Обычно вызывающему пользователю предоставляется несколько адресов на выбор. Использование списка адресов может повлечь за собой нежелательные последствия – раскрытие частной информации о местонахождении конкретного пользователя или организации. Должна существовать возможность сконфигурировать сервер таким образом, чтобы он посылал ответ с кодом 404 (Not Found) по получении неоднозначного запроса или изымал список адресов в поле Contact. Ниже приведён 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 сервисом голосовой почты.
487	**Request Terminated.** Запрос был сброшен 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 не допустимы. Существует возможность связаться с пользователем по другому адресу или используя прочие средства.
489	**Bad Event.** Данный ответ используется, чтобы указать, что сервер не понял типа **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 соответсвующий открытый ключ шифрования; при использовании этого ключа клиентом запросы будут без затруднений обрабатываться UA.
494	**Security Agreement Required.** Данный ответ передаётся сервером при выполнении процедуры выбора механизма обеспечения безопасности. Если запрос клиента помимо заголовка Security-Client со списком механизмом содержит идентификатор **option-tag** «sec-agree» в заголовке Supported, сервер отсылает ответ с кодом 494. Ответ должен включать заголовок Security-Server со списком механизмов обеспечения безопасности, поддерживаемых сервером. Клиент должен выбрать подходящий механизм безопасности и применить его при отсылке запроса.

Ответы отказа сервера (5xx)

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

сервера.

Таблица 2.9 SIP-ответы отказа 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 сервере 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 мероприятий по техническому обслуживанию. Сервер может указать в поле заголовка Retry-After время, по истечении которого следует повторить отправку запроса. UAC или прокси-сервер, получивший ответ 503, должен попытаться направить запрос по иному пути через другой сервер. Не исключены случаи, когда сервер вместо отсылки 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, 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. Этот ответ могут посылать более 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 сессии - информации **offer**, он должен отклонить запрос, передав ответ с кодом 580 – при этом информация answer не передаётся. Данный ответ используется для отказа на предложение установления сеанса связи (**offer**), содержащееся в запросе INVITE или UPDATE. В ответе содержится SDP-описание, основанное на последней принятой от удалённого 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 сессии.

Ответы полной невозможности установления соединения (6xx)

Передаваемый пользователем запрос не может обслужить ни один сервер. Соединение с вызываемым пользователем установить невозможно.

Таблица 2.10 SIP-ответы полной невозможности установления соединения

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 установления сеанса связи.

2.3. Процедуры управления соединением

2.3.1 Диалоги

Диалог представляет собой равноправное взаимодействие двух агентов пользователя по протоколу SIP, которое длится определённое 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 удалённой метки (remote tag). У участвующих в диалоге сторон идентификатор диалога имеет свои отличия. А именно, локальная метка одного UA идентична удалённой метке другого и наоборот. Идентификатор диалога напрямую 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 идентификатора устанавливается по заголовку Call-ID сообщения, удалённая метка – по параметру «tag» в поле заголовка To, а локальная метка по параметру «tag» в поле заголовка From. Для UAS соответственно значение Call-ID идентификатора диалога копируется из заголовка Call-ID сообщения, удалённая метка определяется по параметру «tag» в заголовке From, а локальная метка – по параметру «tag» в заголовке To.

Диалог включает ряд компонентов, которые описывают состояние диалога и используются для передачи сообщений в ходе диалога. Это идентификатор диалога, локальный порядковый номер (для упорядочивания запросов UA, направляемых своему собеседнику), удалённый порядковый номер (для упорядочивания запросов от собеседника к UA), локальный URI, удалённый URI, текущий адрес удалённого пользователя (**remote target**) , булев флаг «secure» и установленный маршрут (**route set**), представляющий из себя упорядоченный список URI. Установленный маршрут – 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 отсылки предварительного ответа; только после получения клиентом окончательного 2xx ответа, диалог переходит в установленное состояние. Если помимо предварительных не приходит никаких ответов, то диалог, находящийся на «ранней стадии» завершается.

2.3.1.1 Процедура создания диалога

Диалоги создаются путём возврата ответов, не информирующих об ошибках, на запросы определённых типов. В данной версии протокола установить диалог возможно только ответами 101-199 и 2xx с параметром «tag» в поле заголовка To на запрос INVITE. Диалог, установленный предварительным ответом на запрос, называется диалогом «на ранней стадии» (early dialog).

Работа UAS

Когда UAS после получения запроса отправляет ответ, инициирующий создание диалога (как, например, ответ класса 2xx на запрос INVITE), UAS должен скопировать содержимое заголовка Record-Route запроса в ответ (включая адреса, параметры адресов и все параметры заголовка вне зависимости известны они серверу или нет) с сохранением порядка следования этих величин. Также UAS должен поместить в ответ заголовок Contact; в этом заголовке указывается предпочтительный адрес для последующих запросов диалога. Обычно, компонента URI, идентифицирующая 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 иметь глобальный масштаб 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 отправки сообщений вызывающему пользователю.

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

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

Установленный маршрут (**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, установленный маршрут устанавливается в «пустое» состояние. Даже если установленный маршрут пуст, он аннулирует предыдущие установленные маршруты для последующих запросов текущего диалога.

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

Удалённый 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 удалённая 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.

Удалённый URI переносится из заголовка From, а локальный URI – из заголовка To.

Работа UAC

Когда UAC посылает запрос, который может привести к созданию диалога (например, INVITE), он должен обеспечить наличие SIP или SIPS URI глобального масштаба в поле заголовка Contact запроса. Если в запросе значение поля Request-URI или верхнее значение заголовка Route – 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 отправлен через TLS, и поле Request-URI содержит SIPS URI, флаг «secure» устанавливается в состояние «TRUE».

Установленный маршрут 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, установленный маршрут переходит в «пустое» состояние. Даже если установленный маршрут пуст, он аннулирует предыдущие установленные маршруты для последующих запросов текущего диалога.

Текущий адрес удалённого пользователя (**remote target**) определяется по URI в поле заголовка Contact ответа. Локальный порядковый номер устанавливается в соответствии с порядковым номером в заголовке CSeq ответа. Удалённый порядковый 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 удалённая 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.

Удалённый URI переносится из заголовка To, а локальный URI – из заголовка From.

2.3.1.2 Процедура посылки и приёма 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, отсылающий запросы будет выполнять роль клиента в транзакции, 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 заголовки Record-Route и Contact. Эти запросы не могут привести к изменению установленного маршрута (**route set**), однако в состоянии модифицировать текущий адрес удалённого пользователя (**remote target**). Для изменения **remote target** применяются определённые типы запросов, которые используют для этой цели вышеупомянутые заголовки. Они носят название запросов, обновляющих текущий адрес удалённого пользователя (target refresh requests). Такими запросами являются re-INVITE и UPDATE.

Работа UAC

- **Создание запроса**

Запрос в состоянии диалога формируется с использованием большинства компонент (см. раздел 2.3.1), описывающих состояние диалога. URI в заголовке To запроса устанавливается в соответствии с удалённым URI диалога, параметр «tag» в заголовке To – по удалённой 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 удалённая или локальная метка имеет нулевое значение, параметр «tag» должен быть исключён из состава заголовка To или From соответственно.

Заголовок Call-ID запроса должен быть составлен в соответствии с Call-ID диалога. Запросы в режиме диалога включают строго 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 запросов. Таким образом, если локальный порядковый номер присутствует, его значение увеличивается на единицу, и результат операции помещается в заголовок CSeq генерируемого запроса. Если же локальный порядковый номер отсутствует, то выбирается начальный порядковый номер.

Содержимое поля типа запроса заголовка CSeq должно соответствовать типу запроса.

Верхняя граница величины порядкового номера - 2^{32}, этого хватит на то, чтобы на протяжении 136 лет клиент в рамках одного вызова генерировал запросы со скоростью один запрос в секунду. Величина начального порядкового номера выбирается таким образом, чтобы порядковые номера последующих запросов текущего вызова не перекрылись с данным. Ненулевое начальное значение порядкового номера даёт возможность клиенту привязать значение к текущему времени. Например, клиент может выбрать значение, отражающее 31 наиболее значимых бит 32-битного секундного счётчика времени в качестве начального порядкового номера.

UAC использует **remote target** и **route set** для формирования поля Request-URI и заголовка Route запроса. В случае отсутствия установленного маршрута UAC копирует **remote target** в поле Request-URI, заголовок Route не добавляется. Если же установленный маршрут 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 установленный маршрут в прямом порядке и со всеми параметрами. Когда установленный маршрут содержит список URI и первый из них не имеет параметра «lr», UAC помещает первый URI установленного маршрута в поле Request-URI, исключая любые запрещённые для этого поля параметры. Клиент должен добавить заголовок Route, содержащий оставшуюся часть установленного 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 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 установленного маршрута 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 удалённого пользователя, в режиме диалога в случае необходимости сменить значение **remote target** – таким образом UA может сменить 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 севера и происходит отправка запроса с использованием тех же процедур, что и при работе вне диалога (См. раздел 2.1.1.1). Итогом выполнения этих процедур обычно является отсылка запроса на адрес, обозначенный первым в заголовке Route или адрес, указанный в поле Request-URI, если заголовок Route отсутствует.

- **Обработка 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 3xx ответов в режиме диалога то же, что и вне диалога (См раздел 2.1.1.2). Когда UAC получает ответ класса 2xx на запрос, обновляющий **remote target**, он должен заменить свой **remote target** на 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.

Работа 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 применяет основные правила обработки запросов, которые применяются как при работе в диалоге, так и вне диалога (см. раздел 2.1.2.1). 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 – это 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 надёжность, 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 удалённого порядкового номера, то он должен быть выставлен в

соответствии с порядковым номером в поле заголовка CSeq запроса. Если же удалённый порядковый номер имеется, но его величина больше порядкового номера в запросе, это значит, что в запросе содержится ошибка; запрос отклоняется и посылается ответ с кодом 500 (Server Internal Error). При величине удалённого 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 удалённый 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 посылается снова. У этого запроса будет уже другой номер в CSeq. UAS никогда не узнает о первом запросе и поэтому обнаружит разрыв в последовательности номеров. Данный разрыв не расценивается как ошибка.

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

2.3.1.3 Процедура завершения диалога

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

Для разрушения установленных диалогов используется запрос BYE (см. раздел 2.8.3) .

2.3.2 Транзакции

SIP – это протокол, ориентированный на транзакции: взаимодействие между элементами сети осуществляется с помощью периодических обменов сообщениями. Транзакция состоит из запроса и любого количества ответов на него (ни одного 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 выступает важность доставки UAC всех ответов с кодом 200 (OK) на запрос INVITE. Чтобы доставить клиенту все ответы с кодом 200, ядро UAS берёт на себя ответственность за их повторную передачу, а ядро UAC соответственно берёт 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 отправкой запросов, а серверная транзакция – отправкой ответов. Они создаются в агентах пользователя и прокси-серверах с сохранением состояний (stateful). На рисунке 2.11, представленном ниже, UAC выполняет клиентскую транзакцию, а его исходящий прокси-сервер выполняет серверную транзакцию. Исходящий сервер также осуществляет клиентскую транзакцию, которая отправляет запрос серверной транзакции входящего прокси-сервера. Входящий прокси-сервер выполняет клиентскую транзакцию, которая в свою очередь отправляет запрос серверной транзакции UAS.

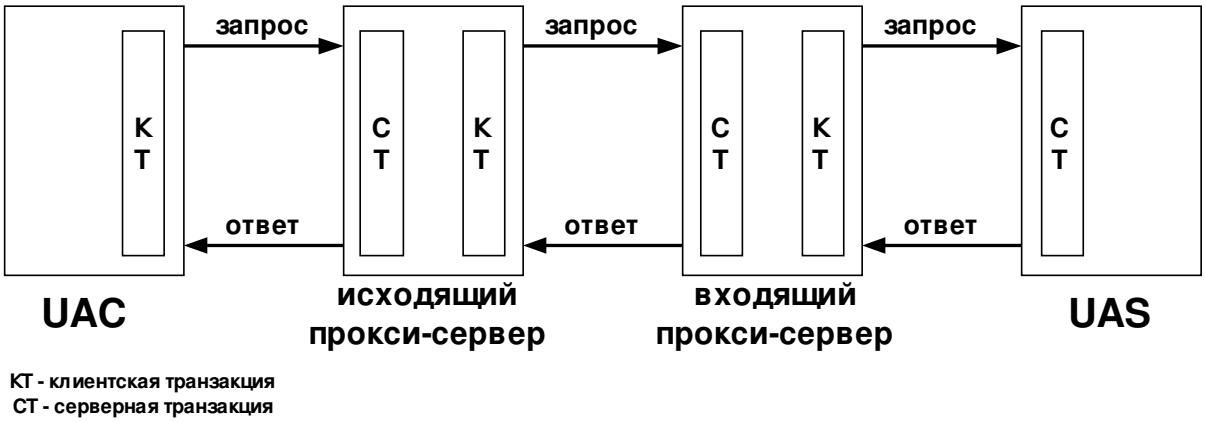


Рис 2.11 Взаимодействие клиентских и серверных транзакций.

Прокси-сервер без сохранения состояний не создаёт 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 надёжно доставить запрос серверной транзакции. Клиентская транзакция также отвечает за получение ответов и доставку их TU, отфильтровывая ответы, переданные повторно, и запрещённые ответы (такие, как ответ на запрос ACK). Кроме того, в случае с запросом INVITE в полномочия клиентской транзакции входит отправка запроса ACK на 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 повторно переданные запросы из сети. Серверная транзакция принимает ответы от TU и передаёт 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. 2xx ответ 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 (из-за требования выполнения этих специфических действия) повторная отсылка 2xx ответов так же, как генерация запроса ACK, это прерогатива ядра UA, а не уровня транзакций. Каждый прокси-сервер на пути следования только пересылает каждый ответ класса 2xx на запрос INVITE и соответствующий ему ACK.

2.3.2.1 Процедуры функционирования клиентских транзакций

Пользователь транзакции (TU) взаимодействует с клиентской транзакцией следующим образом. Когда TU нужно инициировать новую транзакцию, он создает клиентскую транзакцию и передаёт 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 передаёт TU. Один из них поддерживает клиентские транзакции для запросов INVITE (клиентская INVITE-транзакция). Второй тип поддерживает клиентские транзакции всех типов запросов, 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 напрямую транспортному уровню SIP для доставки по сети.

INVITE-транзакция отличается от транзакций других типов запросов из-за своей большой продолжительности: в случае успешного установления сеанса окончательный ответ (класса 2xx) может быть передан только после приёма вызова пользователем. Длительные задержки в процессе отсылки ответа говорят о протекающей процедуре трёхэтапного согласования (three-way handshake). Запросы других типов обрабатываются быстро. TU незамедлительно отвечают на не-INVITE запросы, поскольку существует твёрдая уверенность в том, что согласование должно быть двухэтапным.

Клиентская INVITE-транзакция

INVITE-транзакция реализуется в процессе трёхэтапного согласования. Клиентская транзакция посылает запрос, серверная транзакция отравляет ответ, а затем клиентская транзакция отсылает подтверждение ACK. В случае использования ненадёжного транспортного протокола (такого, как UDP) клиентская транзакция повторно отсылает запросы через отрезок времени T1, который удваивается после каждой повторной передачи. T1 – это оценка периода кругового обращения т.е. времени на передачу и 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 надёжном транспортном протоколе запрос не требует повторной отсылки. После получения

информационного (класса 1xx) ответа все повторные отсылки прекращаются, клиент пребывает в ожидании дальнейших ответов. Серверная транзакция может послать дополнительные 1xx ответы, которые передаются ненадёжно (без подтверждения). В итоге серверной транзакцией посылается окончательный ответ. При использовании надёжного транспортного протокола он передаётся однократно, при использовании ненадёжного – периодически повторяется. Для каждого полученного окончательного ответа клиентская транзакция посылает ACK, назначение которого – прекратить повторную передачу ответа.

Конечный автомат клиентской INVITE транзакции

Конечный автомат (машина состояний) клиентской INVITE транзакции показан на рис. 2.12. Начальное состояние «Calling» наступает, когда TU создаёт новую клиентскую транзакцию по запросу INVITE. Клиентская транзакция должна осуществить передачу запроса на транспортный уровень SIP для дальнейшей транспортировки по сети. Если при этом задействован ненадёжный транспортный протокол, клиентская транзакция запускает таймер A со значением T1. В противном случае таймер A не используется (Таймер A предназначен для контроля времени повторной передачи запросов). Также должен вступить в действие таймер B со значением 64 T1 вне зависимости от используемого транспортного протокола (Таймер B 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 отсылается повторно через интервал времени, удваивающийся после каждой передачи. Повторные передачи 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 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, клиентская транзакция должна проинформировать TU об истечении времени ожидания. Значение 64 T1 выбрано с учётом возможности отправки семи запросов за этот интервал времени при использовании ненадёжного 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 запрос должен быть направлен TU. Ему отправляются все предварительные запросы, пришедшие за время нахождения в состоянии «Proceeding».

Приём 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 отсылает 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 ненадёжного транспортного протокола и 0 для надёжного. Он отражает время, которое серверная транзакция может оставаться в состоянии «Completed», когда используется ненадёжный транспортный протокол. Таймер D идентичен таймеру H в INVITE серверной транзакции, чьё значение по умолчанию 64 T1. Однако клиентская транзакция не располагает сведениями о значении T1, используемом серверной транзакцией, поэтому величина таймера D принята равной 32 с.

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

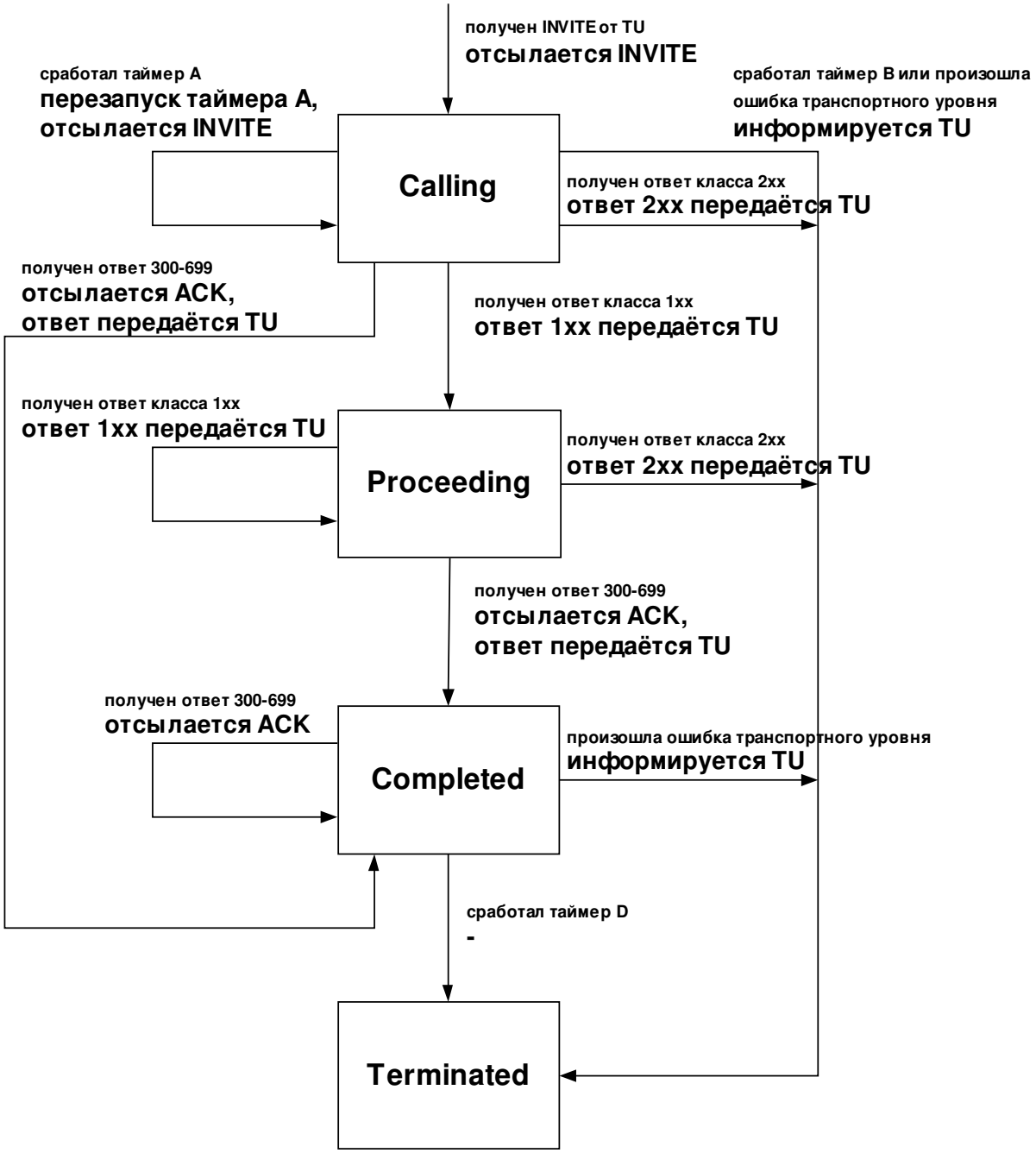


Рис. 2.12 Клиентская INVITE-транзакция.

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) вверх по сети.

Клиентская транзакция должна быть разрушена в момент перехода в состояние «Terminated». Это необходимо для того, чтобы гарантировать правильность выполнения

работы. Причиной тому является то, что 2xx ответы 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 подтверждённым). 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 напрямую к TU. А так как клиентская транзакция разрушается первым ответом класса 2xx, последующие 2xx ответы будут напрямую передаваться ядру.

Формирование запроса ACK

Далее рассматривается конструкция запросов ACK, отсылаемых в ходе клиентской транзакции. Ядро UAC, генерирующее ACK на ответ класса 2xx, должно следовать правилам, определённым для данного типа запроса в разделе 2.3.1. Запрос ACK, формируемый клиентской транзакцией, должен содержать значения заголовков Call-ID, From, и поля Request-URI, идентичные тем, что были отосланы 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 в оригинальном запросе, но в поле типа запроса выставляется *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 отклонён в случае, если тело сообщение не понято. Поэтому не рекомендуется помещать тело сообщения в подтверждение ACK на 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 пришёл 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 на окончательный ответ, отличный 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
```

Клиентская не-INVITE транзакция

Клиентские не-INVITE транзакции не используют запрос ACK. Они строятся по схеме запрос-ответ. При использовании ненадёжного 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 отосланный предварительный/окончательный ответ только после получения повторно переданного запроса. Поэтому необходимо продолжать повторную передачу запроса после получения предварительного ответа, это гарантирует надёжную доставку окончательного ответа.

Конечный автомат клиентской не-INVITE транзакции

Конечный автомат (машина состояний) клиентской не-INVITE транзакции показан на рис. 2.13. 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. При использовании ненадёжного транспортного протокола запускает 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 повторные передачи происходят с экспоненциально увеличивающимся интервалом времени; возрастание интервала прекращается при достижении величины T2. Значение по умолчанию для T2 – 4 секунды. В течение этого времени серверная не-INVITE транзакция может формировать ответ на запрос, если ответ не был отослан немедленно. Для значений по умолчанию величин T1 и T2 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 проинформировать TU об истечении времени ожидания и перейти в состояние «Terminated». При получении предварительного ответа в состоянии «Trying», он передаётся TU, а затем клиентская транзакция переходит в состояние «Proceeding». При получении окончательного ответа (ответы 200-699) в состоянии «Trying», он также передаётся TU, а клиентская транзакция переходит в состояние «Completed».

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

При переходе клиентской транзакции в состояние «Completed», запускается таймер K со значением T4 при использовании ненадёжного транспортного протокола и 0 – для надёжного. Состояние «Completed» предназначено для буферизации дополнительных повторных ответов, которые могут придти – поэтому это состояние используется только при ненадёжном транспортном протоколе. T4 представляет интервал времени, необходимый сети для завершения передачи сообщений между клиентской и серверной транзакциями. Значение по умолчанию для T4 – 5 секунд. Ответ будет расцениваться повторно переданным, если он относится к той данной транзакции. Когда срабатывает таймер K, клиентская транзакция переходит в состояние «Terminated». Как только клиентская транзакция оказывается в этом состоянии, она тут же разрушается.

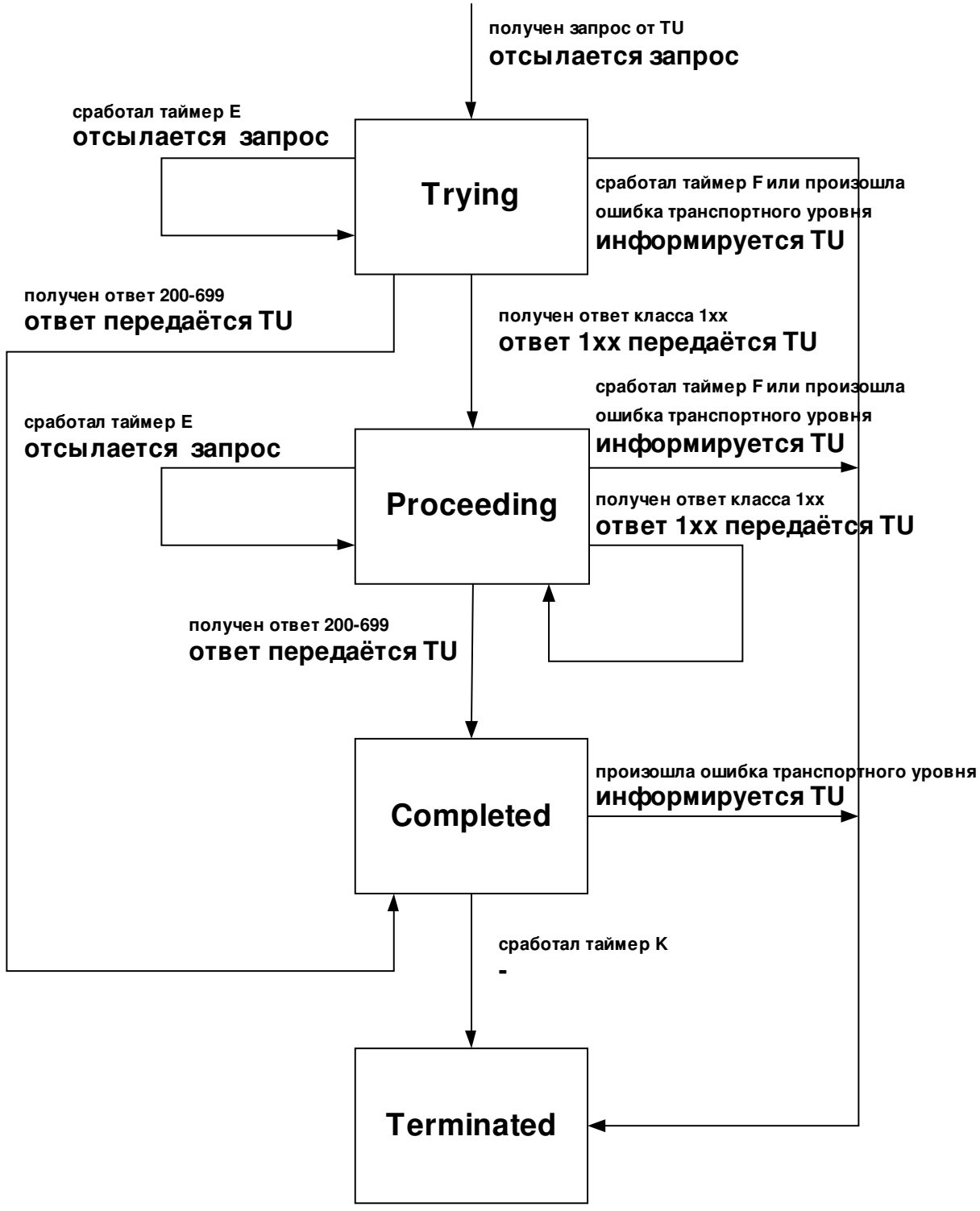


Рис. 2.13 Клиентская не-INVITE транзакция

Соответствие ответов клиентским транзакциям

Когда транспортный уровень клиента принимает ответ, он должен выяснить, какой

клиентской транзакции он 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:

1. Верхний заголовок Via ответа имеет то же значение параметра «branch», что и аналогичный верхний заголовок запроса, инициировавшего транзакцию.

2. Поле типа запроса в заголовке CSeq соответствует типу запроса, создавшему 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 посылается в режиме многоадресной рассылки то возможна доставка нескольких ответов с различных серверов. Все эти ответы будут иметь одинаковый параметр «branch» в верхнем заголовке Via, но будут отличаться параметром «tag» в заголовке To. Первый из них, который будет получен, подвергнется обработке, а остальные будут рассматриваться, как повторные передачи. Когда клиентская транзакция передаёт запрос транспортному уровню SIP для доставки, и если транспортный уровень SIP сообщает при этом об ошибке, то должны выполняться следующие процедуры. Клиентская транзакция информирует TU о транспортной ошибке и переходит в состояние «Terminated».

2.3.2.2 Процедуры функционирования серверных транзакций

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

Серверная INVITE транзакция

Конечный автомат (машина состояний) серверной INVITE транзакции показан на рис. 2.14. Когда приходит запрос, клиентская транзакция создаётся и переходит в состояние «Proceeding». Серверная транзакция должна передать ответ с кодом 100 (Trying), если не обладает сведениями, что TU сформирует предварительный или окончательный ответ в течение 200 мс. Этот предварительный ответ нужен для того, чтобы быстро прекратить повторные передачи запросов и тем самым предотвращать перегрузки сети. Полученный

запрос должен быть передан TU.

TU может передать любое число предварительных ответов серверной транзакции. Пока серверная транзакция находится в состоянии «Proceeding», она должна направлять все эти ответы на транспортный уровень SIP. Т.е. серверная транзакция не занимается повторной отправкой предварительных ответов, они передаются ненадёжно без участия серверной транзакции. Поэтому их отправка 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 его отправить опять же на транспортный уровень SIP для передачи по сети. Серверная транзакция не занимается повторной передачей 2xx ответов, это прерогатива TU. После этого серверная транзакция переходит в состояние «Terminated». При получении ответа от TU с кодом 300-699 серверная транзакция передаёт его транспортному уровню SIP, и переходит в состояние «Completed». При использовании ненадёжного транспорта запускается таймер G со значением T1.

После того, как серверная транзакция перешла в состояние «Completed», включается таймер H со значением 64·T1 для любого транспорта. Таймер H определяет, когда серверная транзакция должна прекратить повторную передачу ответов. Его величина выбирается равной значению таймера B, определяющего время, в течение которого клиентская транзакция будет повторно отсылать запросы. Если срабатывает таймер G, ответ отсылается 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 устанавливается равным T2. Кроме того, если в состоянии «Completed» приходит повторный запрос, серверу следует передать 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, повторный передачи будут прекращены.

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 на повторно переданные окончательные ответы. Как только это состояние наступает, таймер I запускается со значением T4 для случая с ненадёжным транспортным протоколом, и значением 0 для случая с надёжным транспортом. При срабатывании таймера I серверная транзакция переходит в состояние «Terminated».

В состоянии «Terminated» серверная транзакция немедленно разрушается. Как и для клиентских транзакций это нужно для того, чтобы обеспечить надёжность передачи ответов

класса 2xx на запрос INVITE.

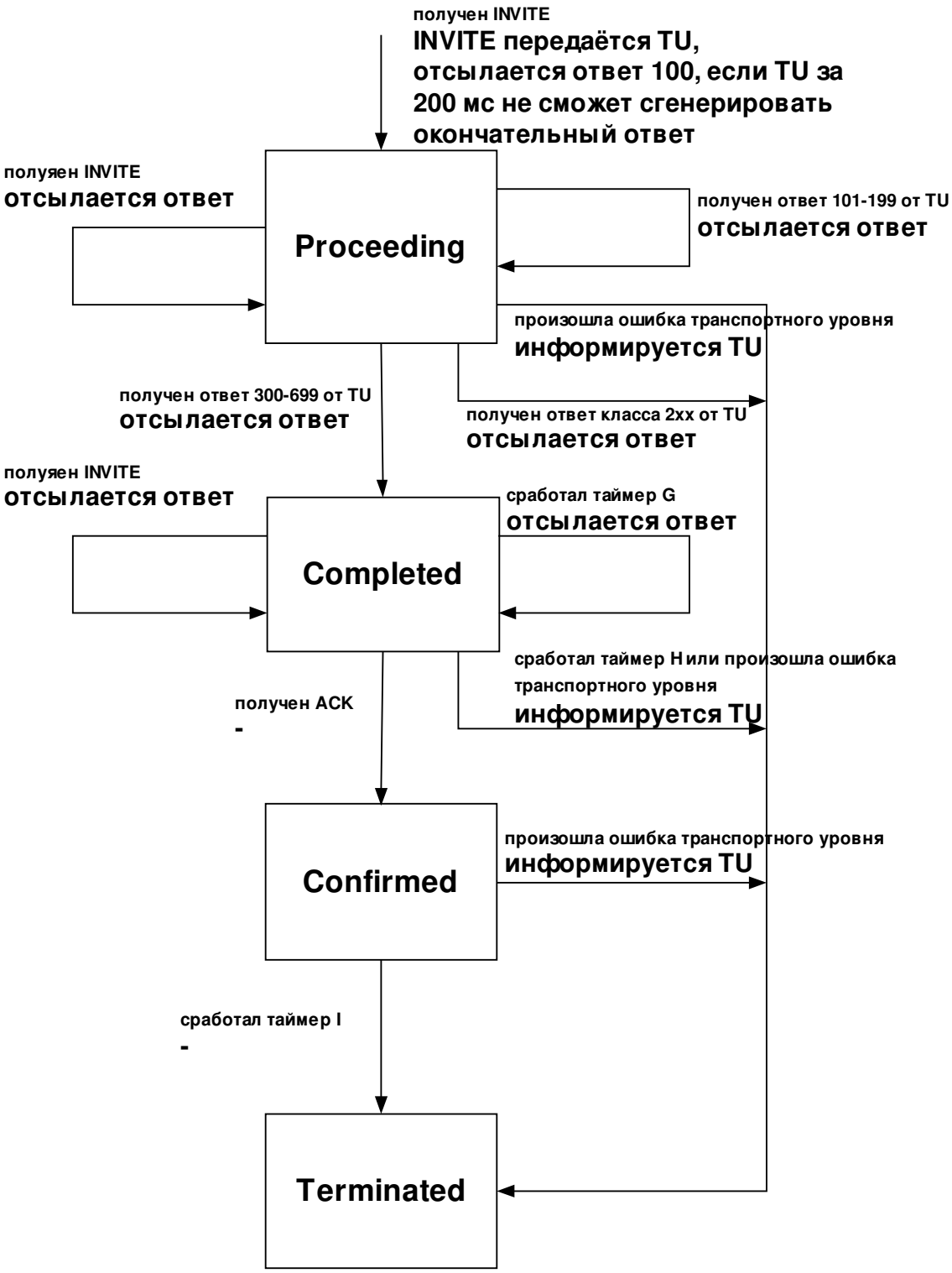


Рис. 2.14 Серверная INVITE транзакция

Не-INVITE серверная транзакция

Конечный автомат (машина состояний) серверной не-INVITE транзакции показан на рис. 2.15. При поступлении запроса (исключая INVITE и ACK) серверная транзакция входит в состоянии «Trying». Запрос передаётся TU, а все дальнейшие повторные запросы отбрасываются (запрос считается повторным, если относится к той же серверной транзакции). Если TU передаёт серверной транзакции предварительный ответ, то 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 передаёт на транспортный уровень SIP предварительный ответ, отосланный последним. Транзакция переходит в состояние «Completed» при поступлении от TU окончательного ответа (ответы 200-699), ответ передаётся на транспортный уровень SIP.

При переходе в состояние «Completed» запускается таймер J со значением 64·T1 секунд для ненадёжного транспортного протокола и 0 секунд – для надёжного. В этом состоянии серверная транзакция передаёт окончательный ответ на транспортный уровень SIP для повторной передачи, когда бы ни пришёл повторный запрос. Любые другие окончательные ответы, поступающие от TU в состоянии «Completed», отклоняются серверной транзакцией. Транзакция остаётся в данном состоянии до тех пор, пока не сработает таймер J; после этого серверная транзакция перемещается в состояние «Terminated». В этом состоянии серверная транзакция прекращает своё существование.

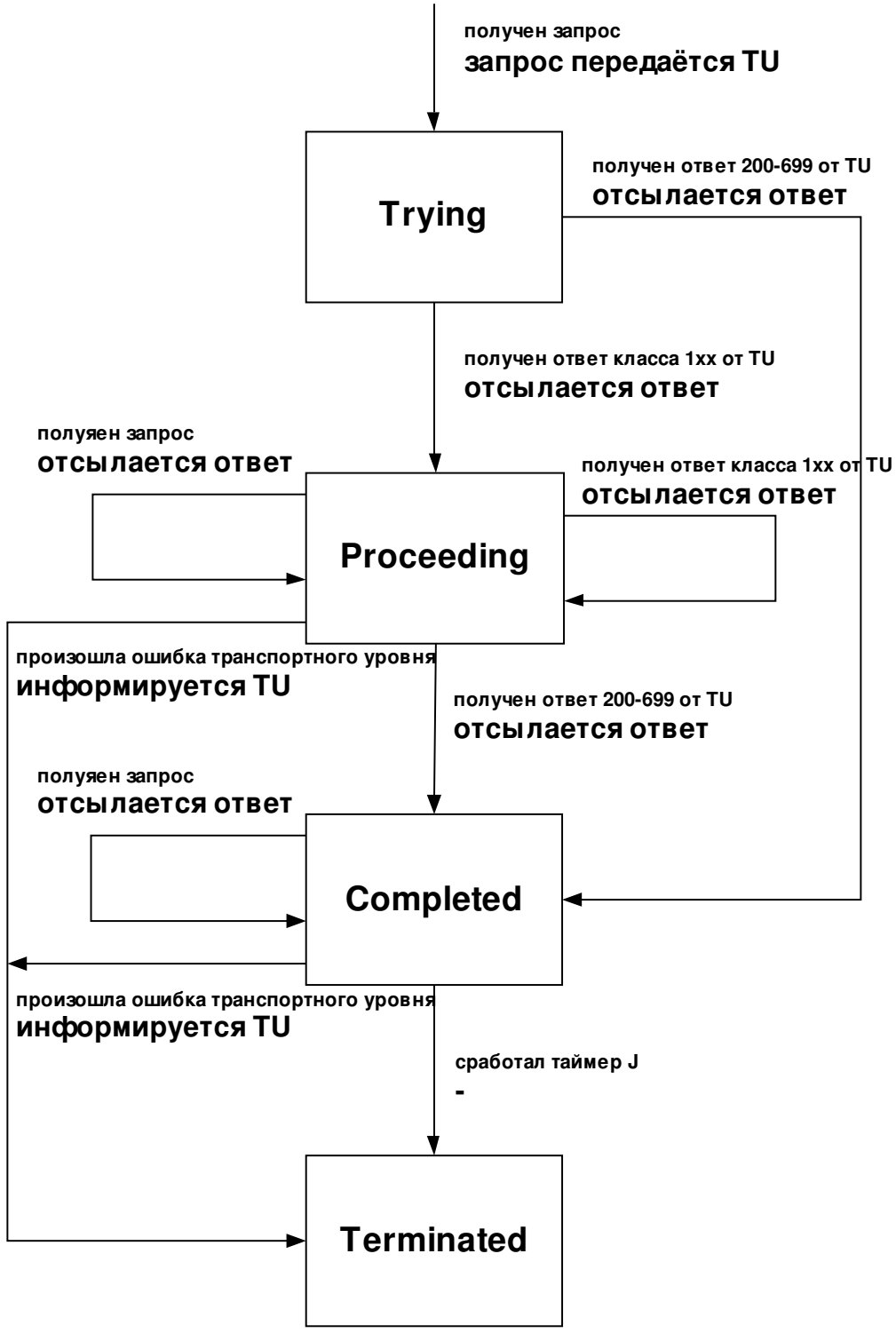


Рис. 2.15 Серверная не-INVITE транзакция

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 SIPv2.0. Следовательно, параметр «branch» уникален для каждой транзакции, в которой принимает участие клиент.

Запрос соответствует транзакции, когда:

1. параметр «branch» запроса совпадает с аналогичным параметром в первом заголовке Via запроса, создавшего транзакцию.

2. имя хоста и номер порта в первом значении заголовка Via совпадают с аналогичными в первом значении заголовка Via запроса, создавшего транзакцию.

3. 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 одинаков с параметром «tag» заголовка To в ответе, отосланном серверной транзакцией. Добавление параметра «tag» заголовка To в процесс сопоставления помогает прокси-серверу отличить ACK на ответ класса 2xx от подтверждения ACK на другие ответы.

Для всех остальных типов сообщений, 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.

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

В этом случае имеет место повторная передача не-INVITE запроса.

Таблица 2.11 Таймеры в протоколе SIP

2.3.2.3 SDL-диаграммы для конечных автоматов транзакций

Клиентская INVITE транзакция

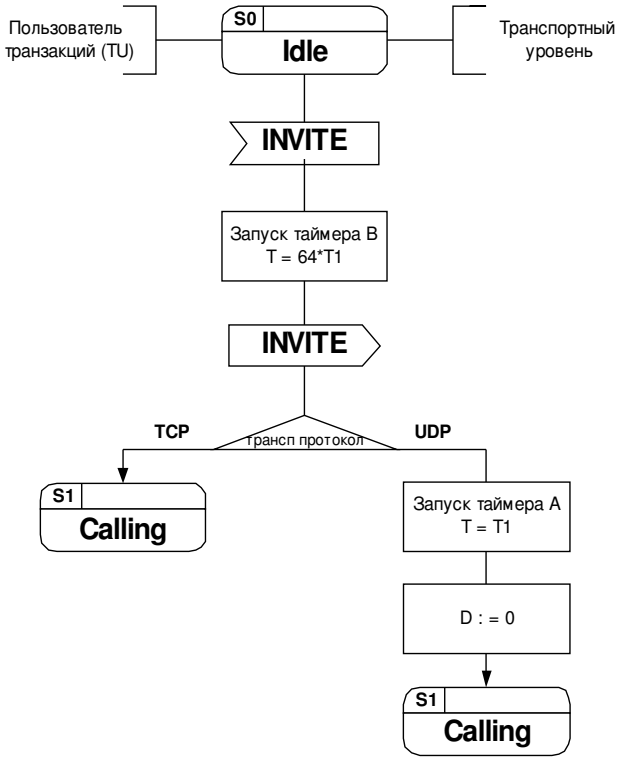


Рис.2.16. Переход в состояние Calling.

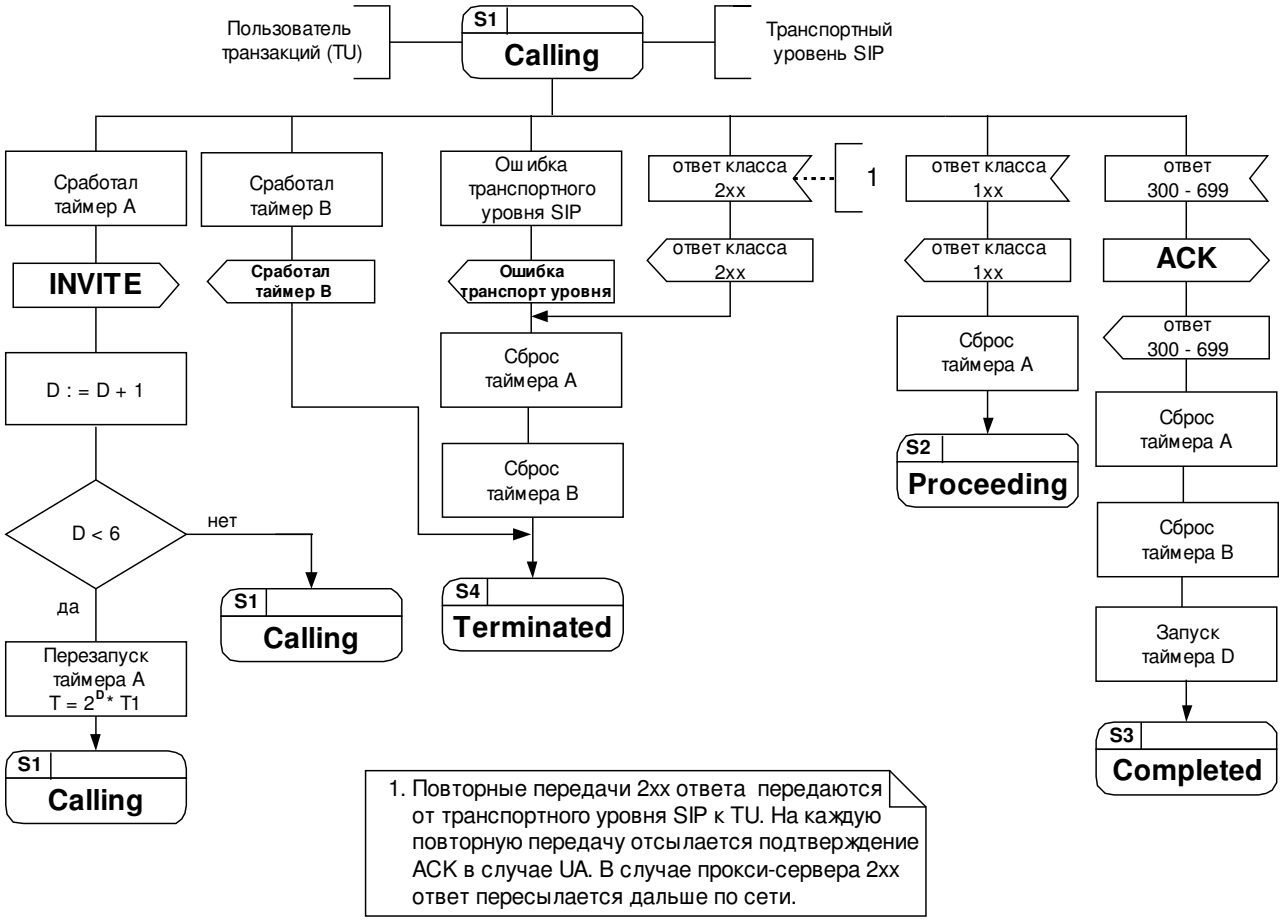


Рис.2.17. Переход в состояние Proceeding.

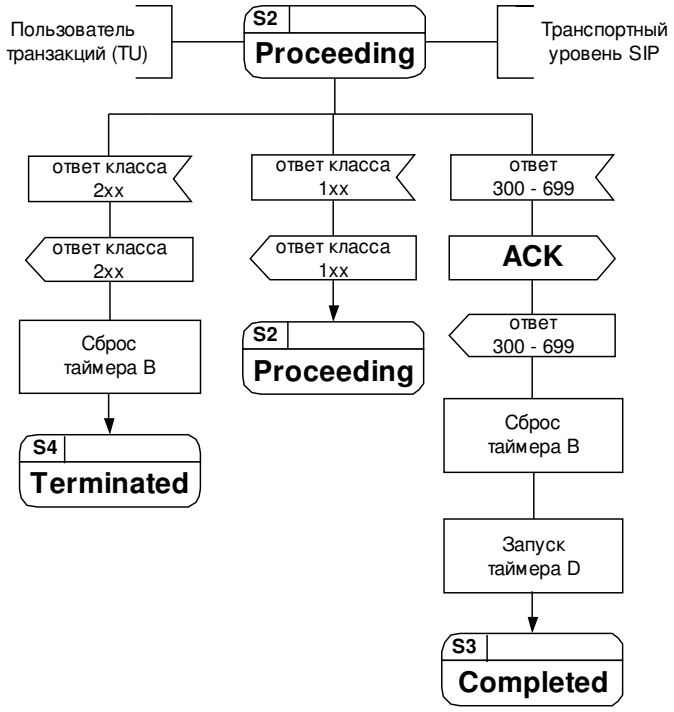


Рис.2.18. Переход в состояние Completed.

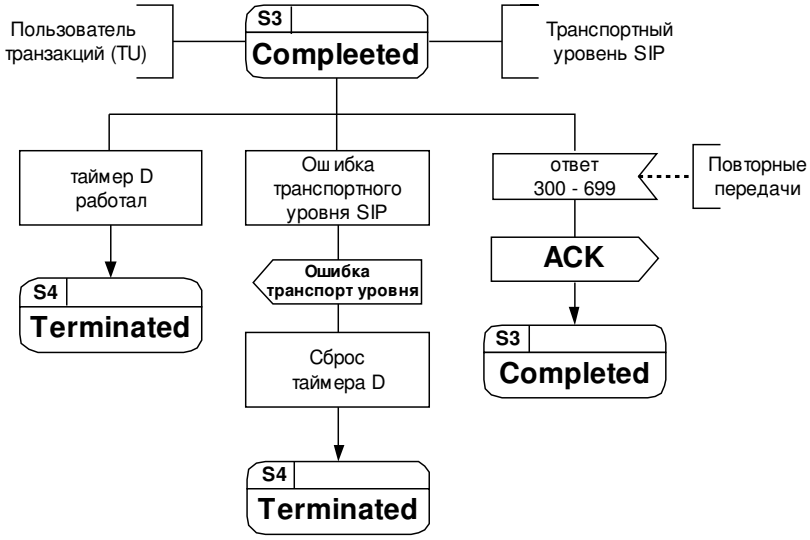


Рис.2.19. Переход в состояние Terminated.

Состояния:

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

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

S2 – «Proceeding» - состояние приёма ответов – получен 1xx ответ, повторные передачи INVITE прекращены, ожидание окончательного ответа.

S3 – «Completed» - состояние окончания обработки сообщений - ожидание повторных передач окончательного ответа 300-699, отсылка подтверждений ACK.

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

Клиентская не-INVITE транзакция

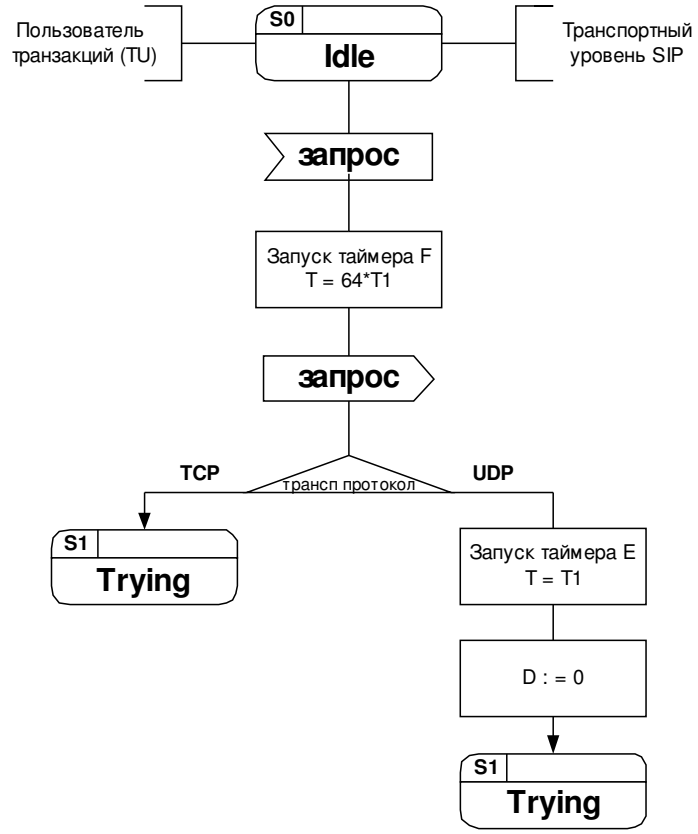


Рис.2.20. Переход в состояние Trying.

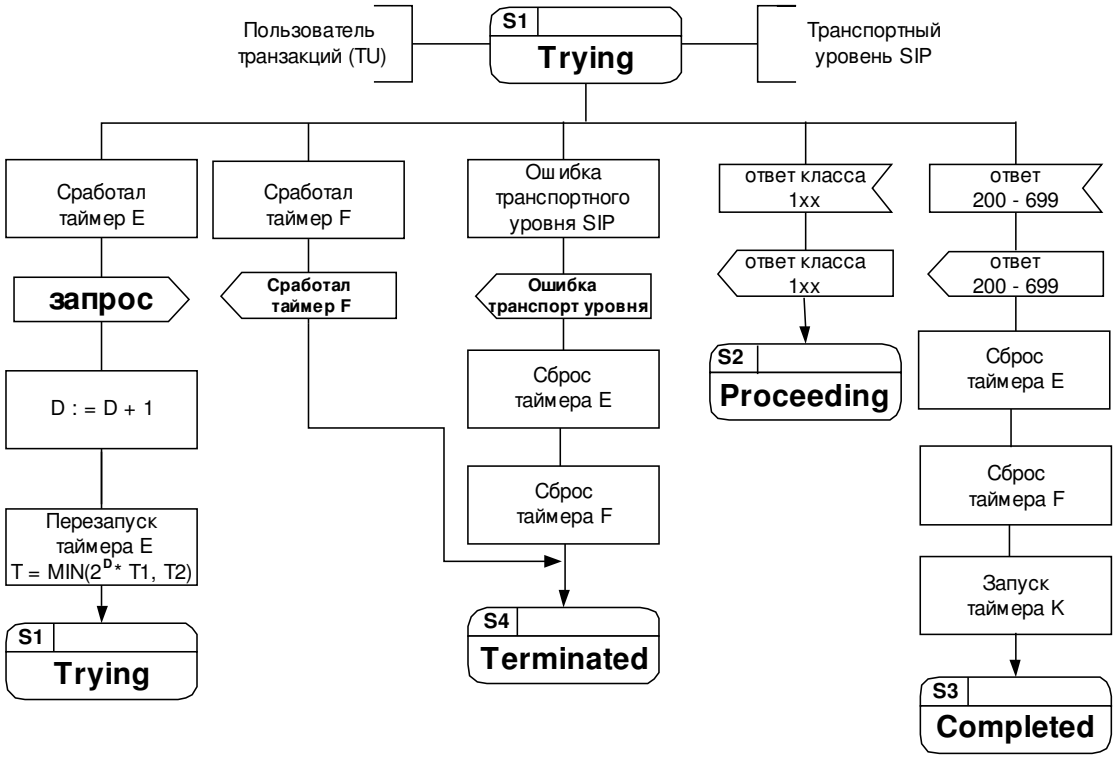


Рис.2.21. Переход в состояние Proceeding.

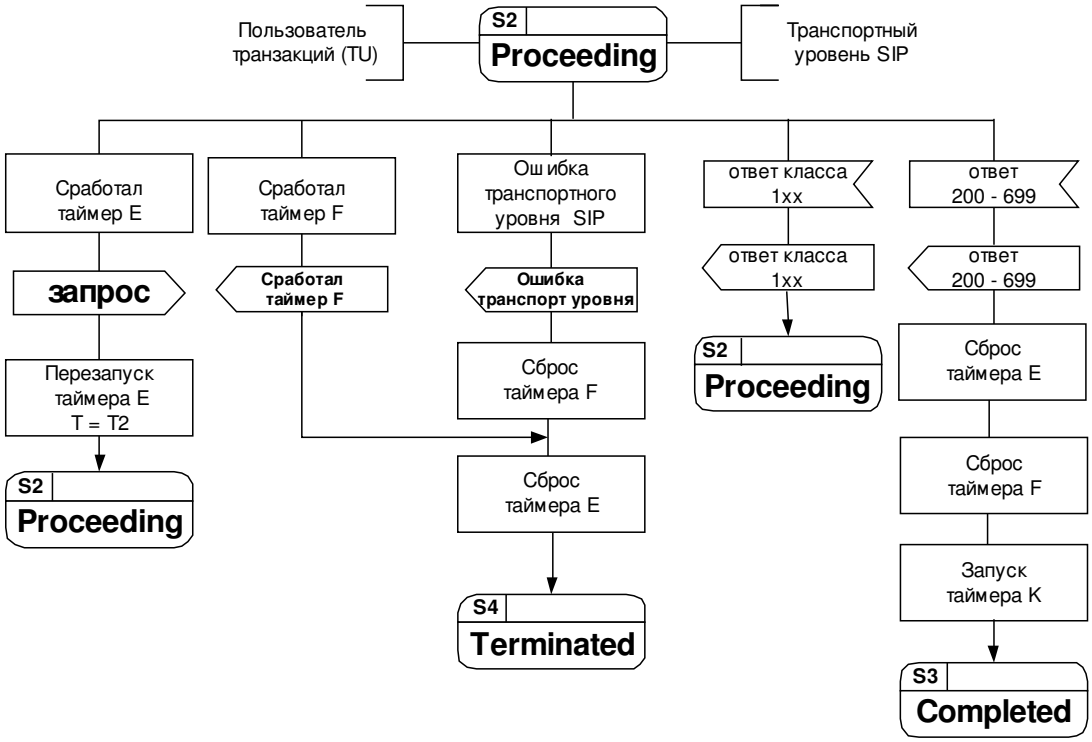


Рис.2.22. Переход в состояние Completed.

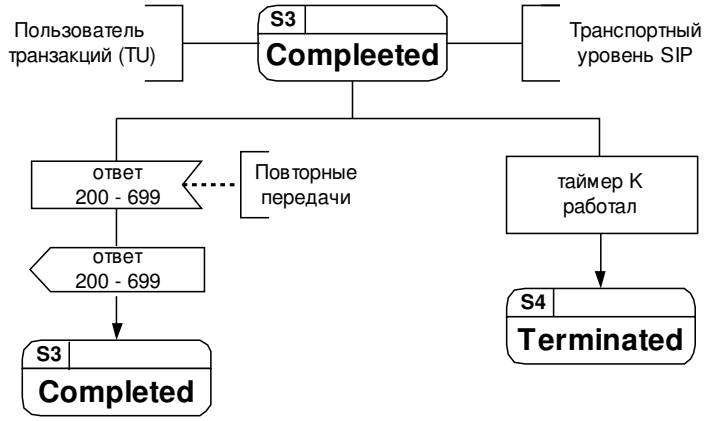


Рис.2.23. Переход в состояние Terminated.

Состояния:

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

S1 – «Trying» - состояние отсылки запроса - периодически отсылается запрос, ожидание ответов.

S2 – «Proceeding» - состояние приёма ответов – получен 1xx ответ, продолжается повторная отсылка запроса, ожидание окончательного ответа.

S3 – «Completed» - состояние окончания обработки сообщений - ожидание повторных передач окончательного ответа 200-699.

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

Серверная INVITE транзакция

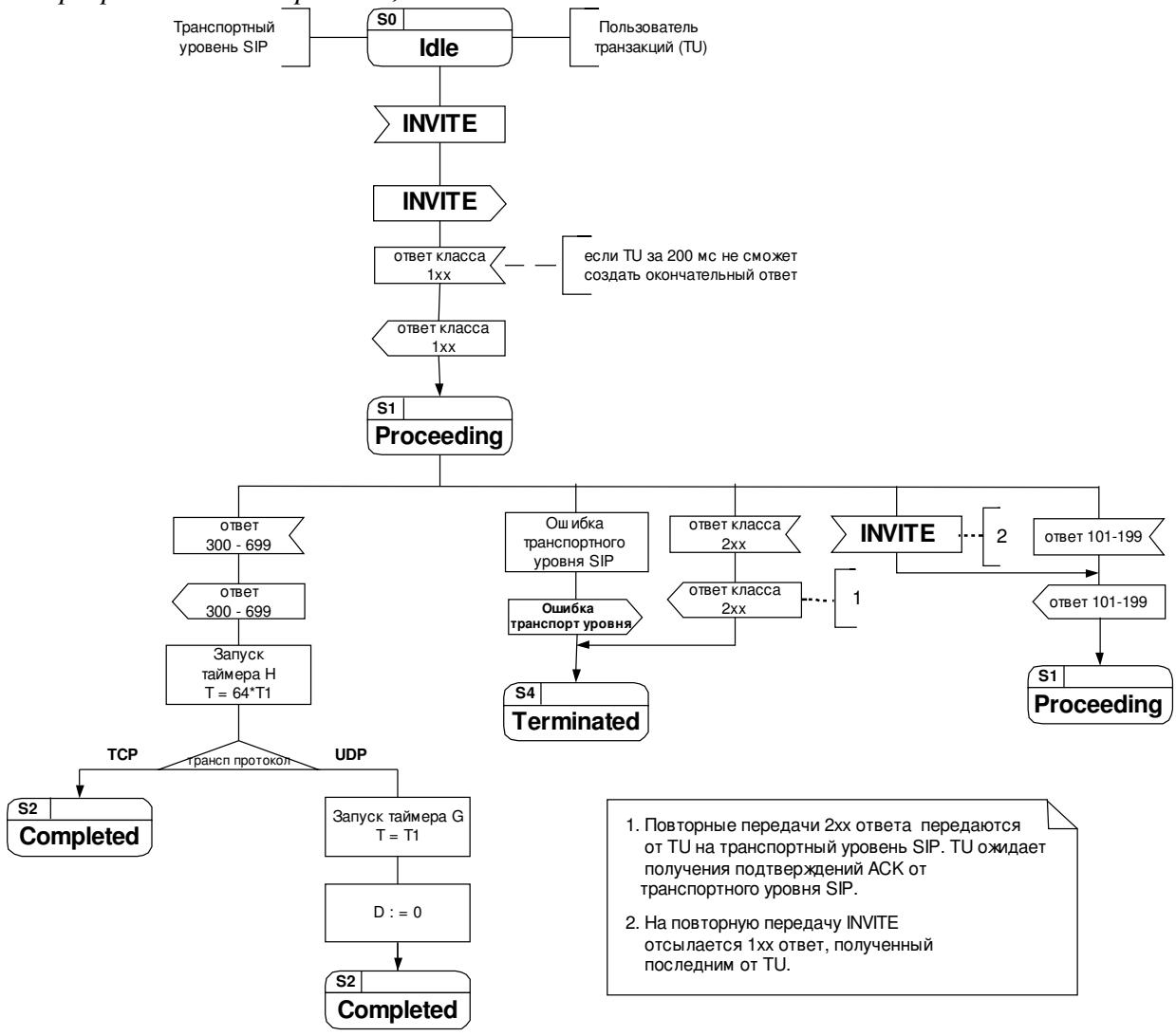


Рис.2.24. Переход в состояние Proceeding.

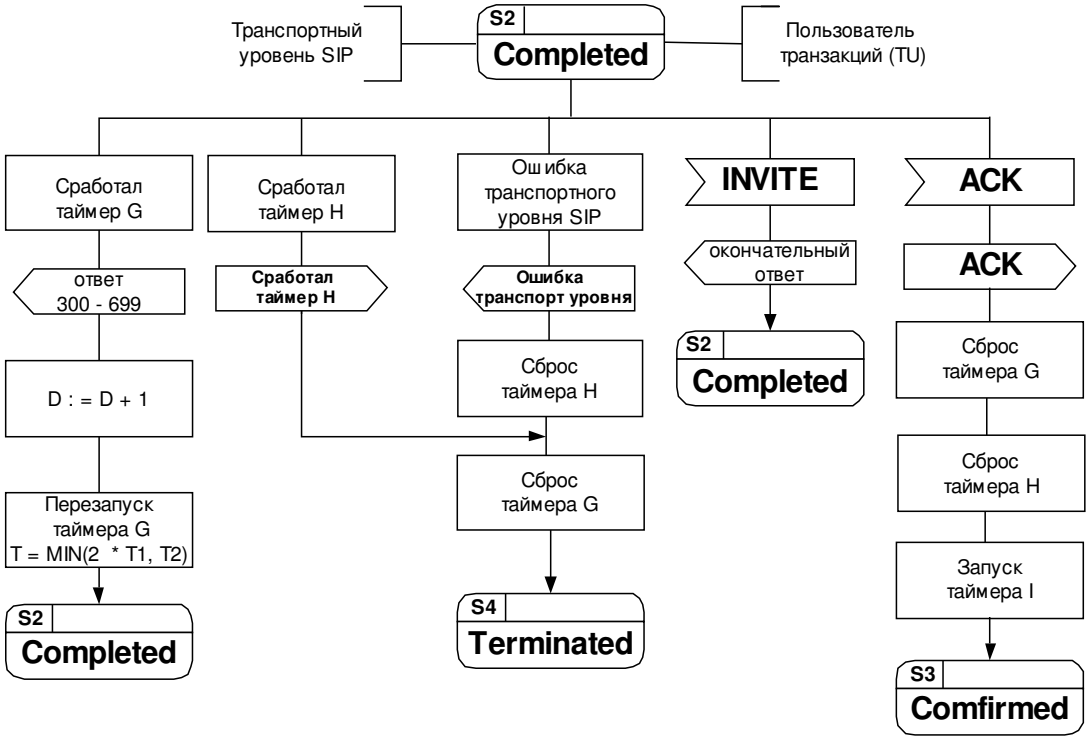


Рис.2.25. Переход в состояние Completed.

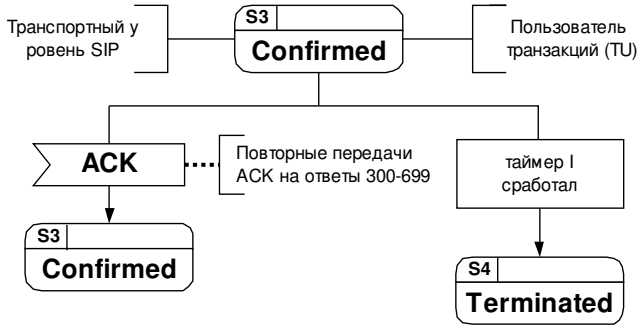


Рис.2.26. Переход в состояние Confiimed.

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

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

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

S3 – «Confirmed» - подтверждённое состояние – приём повторных передач ACK на ответы 300-699.

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

Серверная не-INVITE транзакция

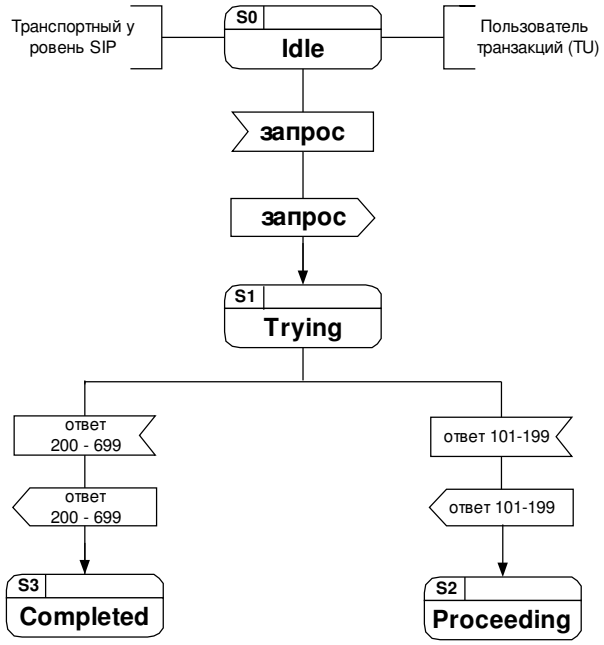


Рис.2.27. Переход в состояние Proceeding.

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

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

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

S3 – «Completed» - состояние окончания обработки сообщений – приём повторных передач запроса, отсылка ответов 200-699 на них.

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

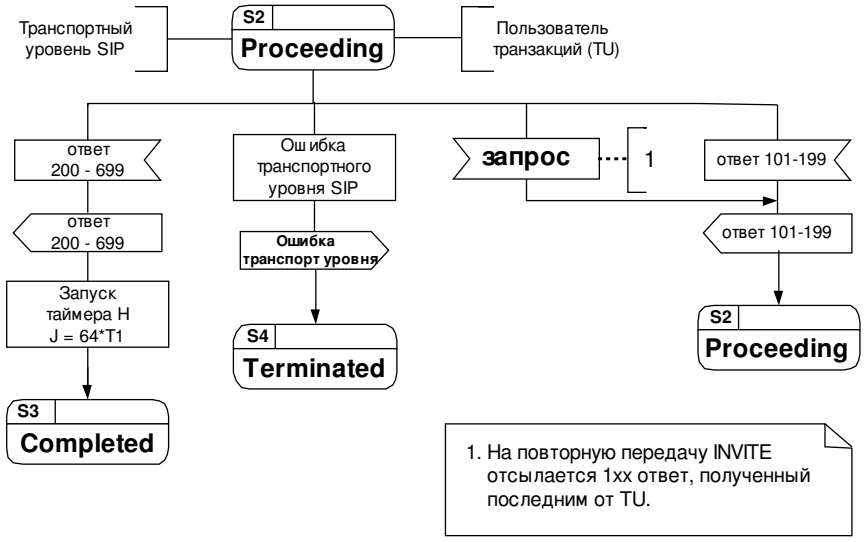


Рис.2.28. Переход в состояние Completed.

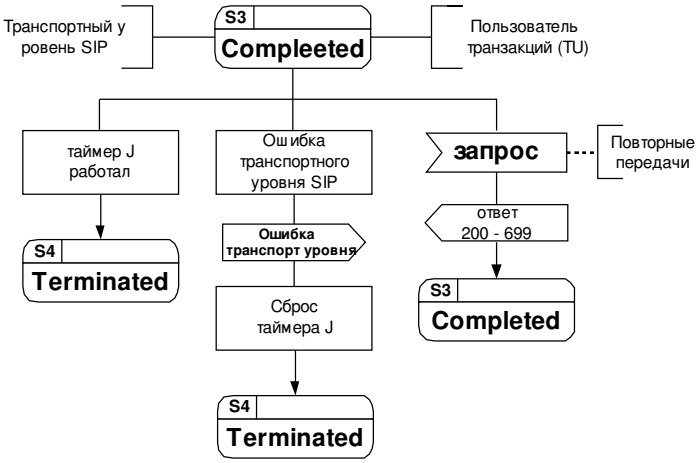


Рис.2.29. Переход в состояние Terminated.

2.3.3 Процедура регистрации

Если пользователь желает 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-серверами и серверами перенаправления, которые ответственны за получение запроса, определение его адресата на основании информации о местонахождении пользователя и его отсылку на установленный адрес. Для этого элементы сети обращаются к серверу определения местоположения (location server), который обеспечивает предоставление информации 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 счёте, прокси-сервер с помощью сервиса определения местоположения узнает адрес агента (агентов) пользователя, где в текущее время пребывает вызываемый пользователь.

Процедура регистрации вносит изменения в базу данных сервиса определения местоположения конкретного домена, где каждая запись представляет собой связку (binding) - сочетание публичного URI (address-of-record) и сопоставленных ему одному или нескольких контактных адресов. Таким образом, когда прокси-сервер этого домена получает запрос, значение поля Request-URI которого совпадает с каким либо публичным адресом, зарегистрированном в данном домене, он направляет его по контактным адресам, зарегистрированным для этого публичного адреса. Необходимость регистрировать публичный адрес у доменного 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 публичного 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 отсылку сообщения REGISTER серверу определённого типа, который называется сервером регистрации (registrar). Он принимает запросы REGISTER и предоставляет информацию из них серверу определения местоположения домена, который он контролирует. Впоследствии информацией, сохранённой на сервере определения местоположения, пользуется прокси-сервер, ответственный за доставку запросов в данный домен.

Процесс регистрации показан полностью на рис. 2.16. Заметим, что функции сервера регистрации и прокси-сервера в сети может выполнять одно устройство. На данном рисунке они изображены раздельно в целях облегчения понимания. Также заметим, что UAS может отсылать запросы для сервера регистрации через прокси-сервер, когда они являются отдельными элементами.

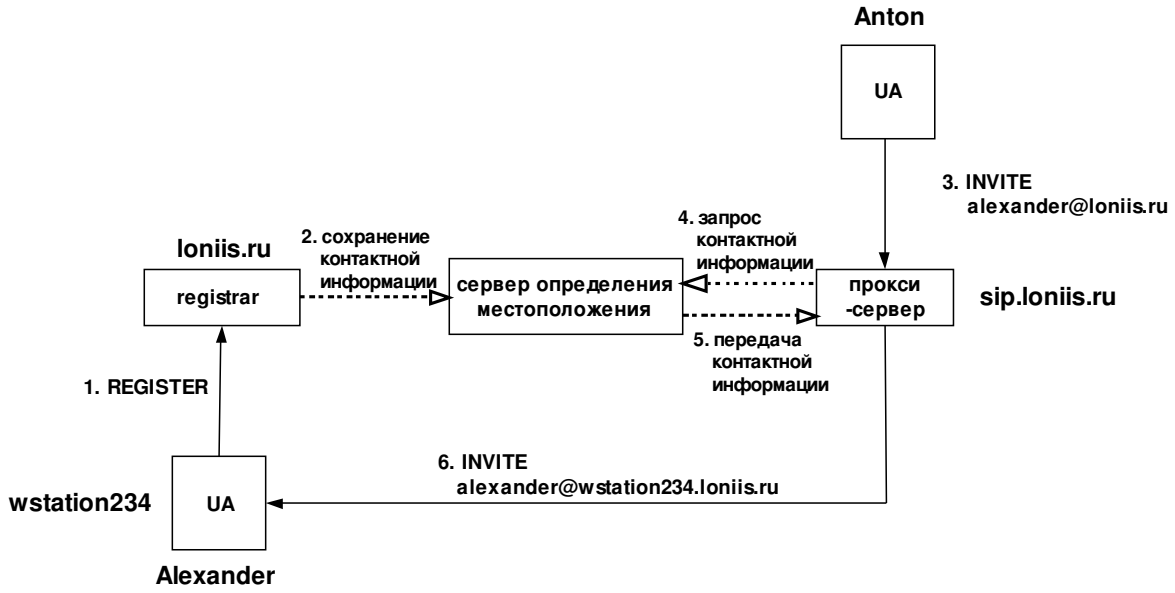


Рис. 2.30 Использование регистрации для сервиса определения местоположения.

Протокол SIP не предписывает выполнение определённых механизмов для реализации сервиса определения местоположения. Единственное требование к серверу регистрации – возможность читать и записывать информацию в базу данных, а к прокси-серверу и серверу перенаправления домена – способность чтения этой информации. Registrar может быть физически совмещён с SIP прокси-сервером одного и того же домена.

2.3.3.1 Процедура фор мирования запроса REGISTER

Запросы REGISTER добавляют, удаляют и изменяют связки в базе данных сервера определения местоположения. Запрос REGISTER может также ввести новое соответствие между публичным адресом и одним или несколькими контактными (создать новые связки).

Регистрация может быть произведена независимой третьей авторизованной стороной (которая не является ни владельцем публичного адреса, ни пользователем, инициирующим вызов на этот адрес).

За исключением моментов, описанных в данном пункте, процесс формирования запроса REGISTER и поведение клиента, отсылающего 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 появления должен быть проигнорирован. В частности, UAC не должен создавать новый установленный 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 публичный адрес, в отношении которого проводится процедура регистрации. Заголовок To и поле Request-URI различаются, т.к. первый содержит имя пользователя. Публичный адрес может быть как SIP, так и SIPS URI.

- **From**

Заголовок From содержит публичный 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 для заголовка Call-ID при отсылке сообщения на соответствующий registrar.

- **CSeq**

Заголовок CSeq гарантирует правильную очерёдность запросов REGISTER. Агент пользователя увеличивает на единицу значение CSeq для каждого запроса REGISTER с

одинаковым Call-ID.

- **Contact**

Запрос REGISTER может включать заголовок Contact, содержащий ноль и более контактных адресов.

Агент пользователя не должен отсылать новых сообщений регистрации (запросы, содержащие новые значения в заголовке Contact), пока не получит окончательный ответ от сервера регистрации на предыдущий запрос или пока не истечёт время ожидания 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 помощью него 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»

Данный параметр определяет время действия связки – поставленные в соответствие публичный 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*.

Создание связок

Запрос REGISTER, отправляемый серверу регистрации, включает контактный адрес (адреса), на который должны направляться SIP-запросы с соответствующим публичным адресом. Публичный 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 определённого публичного адреса.

К примеру, пользователь Vladimir с публичным 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 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 публичного адреса на данном сервере регистрации.

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

- ***Установление времени действия контактного адреса***

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

- ***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 с публичным адресом в заголовке To. В списке контактных адресов в заголовке Contact приоритеты выставляются с помощью параметра «q». Параметр «q» указывает относительный приоритет одного значения в поле заголовка Contact перед другими для соответствующего

публичного адреса.

Удаление связок

Регистрации носят временный характер и теряют свою силу по истечении времени действия, однако они также могут быть удалены пользователем. Клиент может воздействовать на время действия регистрации, установленной сервером регистрации, как описано выше. Поэтому UA запрашивает немедленное удаление связки в базе данных сервера определения местоположения, определяя период времени 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 времени её 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 определённому публичному адресу, не зная точных значений контактных адресов.

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 публичному адресу. UA изучает список на предмет наличия там требуемого контактного адреса. Если контактный адрес присутствует, то UA обновляет значение времени действия контактного адреса в соответствии со значением параметра «expires» или в случае его отсутствия - заголовка Expires. UA отсылает запрос REGISTER для каждой своей связки, пока время действия не истекло; несколько обновлений может быть совмещено в одном запросе REGISTER.

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.

Определение адреса registrar

У UA есть три способа для определения адреса, на который следует отослать

регистрационное сообщение: адрес определяется из конфигурации, адрес определяется из публичного адреса, адрес определяется с помощью использования многоадресной рассылки. В конфигурацию UA может быть внесён адрес конкретного registrar. Если конфигурация UA не содержит адреса registrar, UA использует значение, находящееся справа от «@» в публичном адресе; он помещает его в поле Request-URI запроса и отсылает, используя обычные SIP-механизмы определения местоположения сервера, описанные в "SIP: Locating SIP Servers", RFC 3263. К примеру, UA для пользователя `sip:vladimir@protei.ru` адресует запрос REGISTER на адрес `sip:protei.ru`.

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

Отправка запроса

После того, как сообщение REGISTER сформировано, и определен его адрес 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 отсылать 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 отсылке 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 время действия для каждого контактного адреса в запросе REGISTER равным или большим величины в поле заголовка Min-Expires ответа.

2.3.3.2 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 над 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:

1. 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 адресованный домен по всем правилам работы прокси-сервера.

2. Следуя процедурам функционирования UAS вне диалога, registrar должен обработать значения в поле заголовка Require для того, чтобы гарантировать поддержку всех необходимых расширений.

3. Registrar должен провести аутентификацию UAC. Механизм аутентификации SIP агентов пользователя раскрыт в разделе 2.6. 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.

4. Registrar должен установить, уполномочен ли аутентифицированный пользователь модифицировать регистрации для данного публичного адреса. К примеру, сервер регистрации может обратиться к базе данных авторизации, где отображается список всех публичных адресов, для которых данный пользователь имеет право изменять связки. Если пользователь не в праве изменять регистрации, registrar должен отослать ответ с кодом 403 (Forbidden) и пропустить все оставшиеся шаги.

5. Публичный адрес registrar получает из заголовка To запроса. Если он не согласуется с доменом, указанным в Request-URI, registrar должен отослать 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.

6. Registrar проверяет запрос на наличие в нём 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 время действия отлично от нуля, запрос считается неверным, сервер отсылает ответ с кодом 400 (Invalid Request) и все оставшиеся шаги пропускаются. В противном случае registrar проверяет совпадает ли Call-ID со значением, сохранённым для каждой связки. Если совпадения нет, связка удаляется; если есть - связка удаляется только в том случае, когда значение CSeq в запросе больше значения, сохранённого для данной связки. В противном случае обновление должно быть отменено и запрос отбрасывается.

7. Registrar по очереди обрабатывает каждый контактный адрес в заголовке Contact. Для каждого адреса он определяет интервал времени действия, исходя из следующего:

   - если значение заголовка имеет параметр «expires», величина должна быть выбрана равной запрашиваемой для конкретного контактного адреса.

   - если параметр «expires» отсутствует, но запрос содержит заголовок Expired, величина должна быть выбрана равной запрашиваемой для всех контактных адресов.

   - При отсутствии и параметра «expires», и заголовка Expires сервер регистрации выставляет значение по умолчанию, исходя из своей конфигурации.

   Registrar может выбрать интервал времени действия регистрации меньше запрашиваемого. 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 заголовок Min-Expires, указывающий минимально возможный интервал. После этого оставшиеся шаги пропускаются.

   Интервал времени действия регистрации зачастую используется при создании сервисов. Пример тому - follow-me service, где пользователь может быть доступен на определённом терминале на короткий период времени. Поэтому registrar должен принимать регистрации на короткий период времени; запрос отклоняется только в том случае, если интервал настолько короткий, что 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. Если значения Call-ID в существующей связке и в запросе отличаются, связка должна быть удалена, когда время действия равно нулю, и обновлена, если не равно. 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 учётная 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.

8. Registrar возвращает ответ 200 (OK). Ответ должен содержать заголовок Contact со списком всех активных на данный момент контактных адресов. Каждое значение должно иметь параметр «expires», отображающее время действия, которое выбрал registrar. В ответ также должен входить заголовок Date.

Пример процедуры регистрации

Пользователь Vladimir регистрируется при запуске терминального оборудования. Обмен сообщениями показан на рис. 2.17. Заметим, что процедура аутентификации, необходимая для проведения регистрации, здесь не показана в целях упрощения.

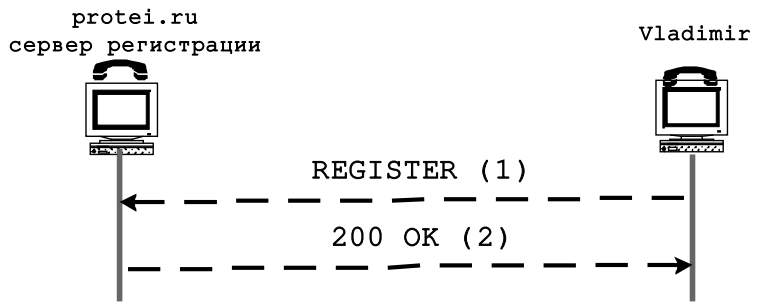


Рис. 2.31 Процедура регистрации

Запрос REGISTER (1)

```
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: 84381763768423O@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:vladimir@192.0.2.4>
Expires: 7200
Content-Length: 0
```

Время действия регистрации истекает через два часа. Registrar посылает ответ 200 (OK):

Ответ 200 (OK) (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: 84381763768423O@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:vladimir@192.0.2.4>
Expires: 7200
Content-Length: 0
```

2.3.4 Процедура запроса информации о функциональных возможностях

SIP-запрос OPTIONS позволяет агенту пользователя запрашивать у другого UA или прокси-сервера информацию о функциональных возможностях. С помощью этой процедуры клиент имеет возможность, 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** определённой опции, клиент может проверить поддержку данной опции сервером UA адресата, отослав запрос OPTIONS. Если в заголовке Supported ответа будет содержаться соответствующий **option-tag**, то UAS поддерживает требуемую опцию. Запрос OPTIONS должны поддерживать все UA.

Адресат для 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 отсылке запроса REGISTER.

В случае, если какой-либо сервер получает запрос OPTIONS со значением заголовка Max-Forwards равным 0, он может ответить на данный запрос, не смотря на то, что поле Request-URI идентифицирует другой сервер. Такое поведение системы заимствовано у протокола HTTP/1.1; используя функцию трассировки маршрута (утилита «traceroute» для HTTP) UA может выяснить функциональные возможности отдельных серверов путём отсылки последовательности запросов OPTIONS с заголовком Max-Forwards, значение которого каждый раз уменьшается на единицу.

Когда истекает время ожидания ответа на отправленный запрос OPTIONS, уровень транзакций может возвратить TU уведомление о таймауте – окончании времени ожидания ответа. Это может означать, что запрос не может быть доставлен по месту назначения и соответственно обработан адресатом.

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

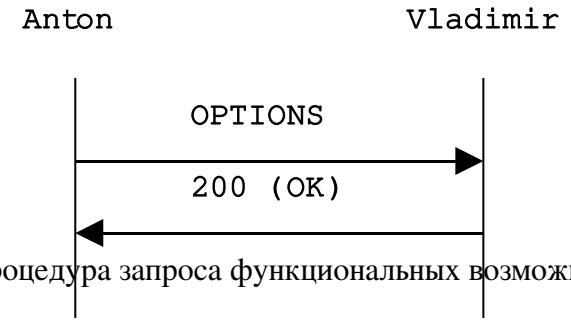


Рис. 2.32 Процедура запроса функциональных возможностей

2.3.4.1 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 помещён заголовок Accept, чтобы указать тип тела сообщения, который UAC предпочёл бы видеть в ответе. Как правило это тип *application/sdp*, описывающий характеристики медиапотока.

Ответ на запрос функциональных возможностей 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, когда OPTIONS отсылается в ходе уже установленного диалога. Ниже представлен пример запроса OPTIONS:

```
OPTIONS sip:alexander@loniis.ru SIP/2.0
Via: SIP/2.0/UDP pc33.niits.tu;branch=z9hG4bKhjhs8ass877
```

```
Max-Forwards: 70
To: <sip:alexander@loniis.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
```

2.3.4.2 Обработка запроса OPTIONS

Ответ на запрос OPTIONS формируется с использованием стандартных правил для UAS по формированию 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, отсылаемому вне диалога, и не оказывает влияния на состояние диалога.

Не смотря на то, что также 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 придти несколько – по одному с каждого направления). Это происходит потому, что только запрос INVITE обладает особым статусом при обработке прокси-серверами, остальные типы запросов обрабатываются по общему механизму.

Если запрос OPTIONS предназначен прокси-серверу, то он создаёт на него 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 создаётся 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 список адресов, по которым предположительно можно связаться с требуемым пользователем. Также может 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 (в примере не показано тело сообщения).

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP pc33.niits.ru;branch=z9hG4bKhjhs8ass877
 ;received=192.0.2.4
```

```
To: <sip:alexander@loniis.ru>;tag=93810874
From: Alice <sip:anton@niits.ru>;tag=1928301774
Call-ID: a84b4c76e66710
CSeq: 63104 OPTIONS
Contact: <sip:alexander@loniis.ru>
Contact: <mailto:alexander@loniis.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
```

2.3.5 Процедура отмены запроса

Запрос CANCEL, исходя из названия, используется для отмены предшествующего запроса, отосланного клиентом. Точнее он запрашивает UAS прекратить обработку запроса и создать ответ с определенным кодом на тот запрос. Запрос CANCEL не оказывает воздействия на запрос, на который UAS уже выдал 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 ещё не отослал окончательного ответа, прекращает передавать пользователю сигнал вызова и посылает ответ на INVITE с определённым кодом ошибки (487).
Запросы CANCEL могут быть сформированы и отосланы и прокси-серверами, и клиентами агента пользователя. Раздел 2.3.2. описывает при каких условиях UAC отменит запрос INVITE, в разделе 2.4.1. дано описание использования запроса CANCEL прокси-сервером.

2.3.5.1. Работа клиента

Запросы CANCEL отсылаются для отмены запросов INVITE. Поскольку на прочие запросы ответы приходят незамедлительно, отсылка 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 с отменяемым запросом. Однако, поле типа запроса в заголовке CSeq должно иметь значение *CANCEL*. Это приводит к тому, что сообщение будет идентифицировано и обработано, как отдельная транзакция.

Если отменяемый запрос содержит заголовок Route, CANCEL также должен содержать значения этого заголовка. Это нужно для того, чтобы stateless прокси-серверы могли правильно маршрутизировать запросы CANCEL.
Запрос CANCEL не должен включать заголовков Require и Proxy-Require.

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 отсылаться; точнее, клиент должен ждать прихода предварительного ответа прежде, чем отослать запрос CANCEL. Если на оригинальный запрос пришёл окончательный ответ, нет смысла передавать CANCEL, так как он не оказывает влияния на запросы, на которые уже создан окончательный ответ. Когда клиент решает отослать запрос CANCEL, он создаёт для него клиентскую транзакцию и передаёт 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, что использовались при отсылке оригинального запроса. Если бы было разрешено посылать 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 отменённой и разрушает клиентскую транзакцию оригинального запроса.

2.3.5.2 Работа сервера

Сообщение CANCEL запрашивает TU на стороне сервера отмену незавершённой транзакции. Чтобы определить транзакцию, которую требуется отменить TU принимает запрос CANCEL, и затем применяет процедуры сопоставления транзакций, описанные в параграфе 2.3.2.2 Отменена может быть только единственная соответствующая транзакция.
Обработка запроса CANCEL сервером зависит от типа сервера. Stateless прокси-сервер его перешлёт, 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 приводится в параграфе 2.4.1.8.

Сперва UAS производит обработку CANCEL в соответствии с общими правилами обработки для UAS, описанными в разделе 2.1.2. 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). Также заметим, что запросы CANCEL не содержат заголовка Require.

Если UAS, выполняя процедуры описанные выше, не нашёл соответствующую транзакцию для CANCEL, он отсылает ответ с кодом 481 (Call Leg/Transaction Does Not

Exist) на запрос CANCEL. Если транзакция для оригинального запроса всё ещё существует, действия UAS при получении запроса CANCEL зависят от того, отослал ли он уже окончательный ответ на оригинальный запрос или нет. Если отослал, то CANCEL не оказывает влияния на обработку оригинального запроса, не оказывает влияния на состояние сессии и не оказывает влияния на ответы, сгенерированные 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 посылает 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 передаёт ответ с кодом 200 (OK). Этот ответ составлен с использованием правил, описанных в параграфе 2.1.2.2, учитывая то, что «tag» в заголовке To ответа на CANCEL и «tag» в заголовке To ответа на оригинальный ответ должны быть одинаковыми. Ответ на CANCEL передаётся серверной транзакции для передачи по сети. На рис. 2.19 представлен пример отмены запроса с использованием запроса CANCEL.

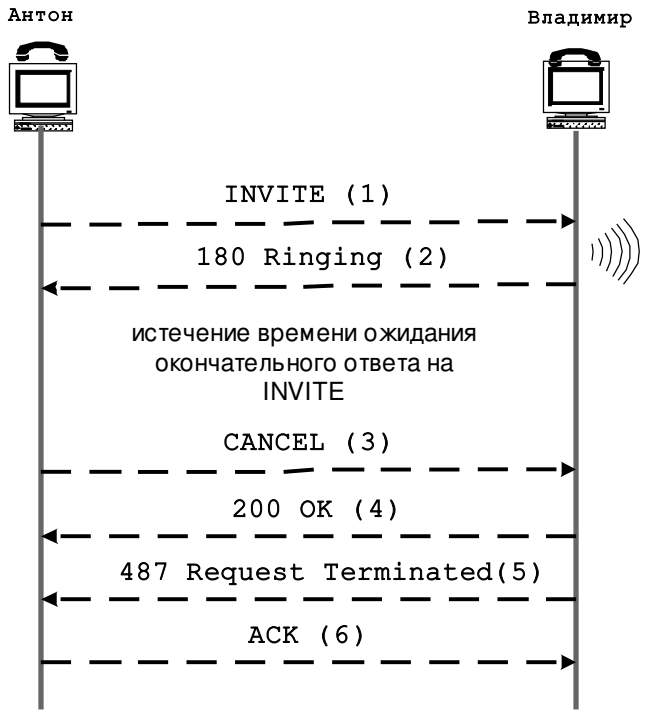


Рис 2.33 Процедура отмены запроса

2.3.6. Инициализация сессий

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

пользователя подтверждения приёма вызова. По прошествии некоторого времени UAS может принять предложение путём отсылки ответа 2xx (OK); после этого сеанс связи считается установленным. Если предложение не принято, посылаются ответы с кодами 3xx, 4xx, 5xx или 6xx в зависимости от причины отказа. Перед отсылкой окончательного ответа UAS также может передать предварительные ответы (1xx) для того, чтобы уведомлять UAC о состоянии вызова со стороны вызываемого пользователя.

После возможного получения одного или нескольких предварительных ответов UAC получит один или более 2xx ответов или один не-2xx окончательный ответ. Из-за того, что ожидание окончательного ответа на запрос INVITE может занять длительное время, механизмы обеспечения надёжности для INVITE-транзакций отличается от механизмов для других запросов (например, OPTIONS). После того, как UAC получает окончательный ответ, он должен посылать ACK на каждый принятую повторную передачу окончательного ответа. Процедура отсылки 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, описанным в параграфе 2.3.2. Для ответов класса 2xx работу с запросом ACK осуществляет ядро UAC.

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

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

2.3.6.1. Процедуры функционирования UAC

Создание первоначального запроса INVITE

Формирование начального запроса INVITE происходит вне диалога с использованием соответствующих стандартных процедур, описанных в разделе 2.1.1.1. 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. Также в запросе скорее всего будет 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 сессий. UAC также может добавить заголовок Expires, чтобы ограничить время действия приглашения к установлению соединения. Если время, указанное в Expires истечёт до получения окончательного ответа на INVITE, UAC отправит 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 отсылка запроса INVITE приводит к созданию нескольких диалогов, обмен происходит отдельно для каждого. Правила использования модели **offer/answer** для начальной INVITE-транзакции представлены ниже:

- Начальное **offer** с описанием сеанса должно содержаться в сообщении INVITE или в первом надёжном сообщении от UAS, не информирующем об ошибке (ответе класса 2xx).

- Если **offer** содержится в сообщении INVITE, то **answer** должен присутствовать в надёжном ответе на INVITE от UAS. Точно такой же ответ с описанием сеанса (**answer**) может быть помещён в предварительные ответы, предшествующие окончательному. UAC трактует первый пришедший **answer**, как описание сеанса связи и игнорирует описания сеанса во всех последующих ответах на начальное сообщение INVITE.

- Если **offer** содержится в первом надёжном сообщении от UAS, не информирующем об ошибке, **answer** должен быть в подтверждении ACK на ответ класса 2xx.

- После отсылки или получения **answer** на первый **offer** UAC может помещать следующие **offer** в запросы, следуя правилам для данного типа запроса, но только если он получил **answer** на предыдущие **offer** и не отослал **offer**, на которые **answer** ещё не получены.

- После того, как UAS отослал или получил **answer** на начальный **offer**, он не должен

включать новых **offer** в состав ответов на начальное сообщение INVITE. Это означает то, что UAS никогда не сможет в одиночку создать новые **offer** до завершения начальной транзакции.

Приведённые выше правила определяют два вида обмена для агентов пользователя:

1. **Offer** в запросе INVITE и **answer** в ответе класса 2xx (и возможно в ответе класса 1xx с тем же значением).

2. **Offer** в ответе класса 2xx и **answer** в подтверждении ACK.

Все агенты пользователя, поддерживающие запрос INVITE, должны поддерживать оба вида обмена.

Протокол описания сеансов связи Session Description Protocol (SDP) (RFC 2327), предназначенный для описания сессий, должен поддерживаться любыми UA.

Указанные выше ограничения для модели **offer/answer** применимы лишь к тем телам сообщения, чей 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* по умолчанию выставляется значение заголовка Content-Disposition - *session*, в противном случае выставляется значение *render*.

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

Обработка ответов на запрос INVITE

Как только UAC передаёт 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).

- **1xx ответы**

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 послать запрос серверу до завершения начальной INVITE-транзакции. Заголовки, содержащиеся в предварительном ответе, действительны только во время нахождения диалога на «ранней стадии» (например, заголовок Allow в предварительном ответе содержит типы запросов, которые могут быть использованы в процессе диалога, пока тот находится на «ранней стадии»).

- **3xx ответы**

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

- **4xx, 5xx и 6xx ответы**

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

- **2xx ответы**

UAC может получить несколько ответов класса 2xx на один отосланный 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 2xx ответе 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 в целях обеспечения обратной совместимости с предыдущей версией рекомендаций).

Ядро UAC должно создавать запрос ACK на каждый полученный ответ класса 2xx. Заголовки запроса ACK формируются так же, как и заголовки любых других сообщений, отсылаемых 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 сессии) в соответствии с правилами, приведёнными выше, в теле сообщения ACK должен находится **answer** (ответ с описанием сессии). Если **offer** в 2xx ответе не может быть принято, ядро UAC формирует **answer** в подтверждение ACK и затем незамедлительно посылает сообщение BYE.

После формирования подтверждения ACK определяются адрес места назначения, номер порта и тип транспортного протокола. Запрос направляется напрямую на транспортный уровень SIP, минуя клиентскую транзакцию. Это происходит потому, что за повторные отсылки запроса 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 2xx ответа на INVITE, UAC не желает продолжать текущий диалог, он должен разорвать диалог посылкой сообщения BYE.

2.3.6.2. Процедуры функционирования UAS

Обработка запроса INVITE

Ядро UAS получает запросы INVITE от уровня транзакций. Вначале происходит выполнение общих процедур 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:

1. Если поступает запрос INVITE, содержащий заголовок Expires, ядро UAS устанавливает таймер со значением, указанным в значении этого заголовка. Срабатывание таймера сигнализирует об истечении времени действия приглашения к установлению сеанса связи. Если это происходит до того, как UAS сформировал окончательный ответ, отправляется ответ с кодом 487 (Request Terminated).

2. Если запрос приходит в ходе диалога то первоначально производятся процедуры обработки запроса, не зависящие от его типа как описано в разделе 2.3.1.2 Также может произойти модификация сессии (см. раздел 2.3.7).

3. 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 предусматривает, что передача запроса INVITE происходит вне диалога, другими словами, запрос направлен на установление новой сессии.

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

Если запрос INVITE не содержит описания сеанса, это значит, что UAC обращается к UAS с просьбой принять участие в сессии и передать информацию **offer**. UAS должен предоставить **offer** в своём первом надёжном 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.

Текущая стадия обработки

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

Если UAS нуждается в отсрочке для того, чтобы ответить 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 трёх минут. Чтобы не допустить этого, UAS должен отсылать предварительный ответ (за исключением ответа с кодом 100) каждую минуту с учётом того, что предварительные ответы могут быть утеряны (т.к. по умолчанию предварительные ответы передаются ненадёжно). INVITE-транзакция может продолжать своё 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).

Перенаправление запроса INVITE

Если UAS решает перенаправить вызов, то посылает 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 передаётся серверной INVITE транзакции, которая осуществляет его повторные передачи.

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 отсылается ответ с кодом 486 (Busy Here). Если UAS знает о том, что никакой другой терминал не в состоянии принять этот вызов, отправляется ответ с кодом 600 (Busy Everywhere). Так как маловероятно, что UAS может располагать такими широкими сведениями, поэтому ответ с кодом 600 обычно не используется. Ответ передаётся серверной INVITE транзакции, которая занимается его повторными передачами.

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

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 вызова, без дополнительных проб. Если запрос INVITE содержит **offer**, и UAS ещё не отослал **answer**, ответ класса 2xx должен включать **answer**. Если в запросе INVITE нет **offer**, ответ класса 2xx должен содержать **offer** в случае, если UAS еще не отослало **offer**.

Как только ответ сформирован, он отсылается 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 посылать ответы непосредственно на транспортный уровень SIP, пока не придёт подтверждение ACK. Ответ класса 2xx передаётся на транспортный уровень SIP с интервалом, начинающимся с T1 секунд и удваивающимся после каждой повторной передаче, пока не достигнет T2 (величины T1 и T2 указаны в разделе 2.3.2).

Повторные отсылки ответа прекращаются, как только приходит подтверждение ACK. Это не зависит от того, какие транспортные протоколы используются для доставки ответов. Повторные передачи ответа класса 2xx передаются из конца в конец, в тоже время не исключается возможность наличия в сети звеньев, использующих протокол UDP для передачи ответов. Чтобы гарантировать надёжную доставку на этих участках, осуществляется повторная передача ответа класса 2xx даже при использовании UAS надёжного транспортного протокола такого, как TCP.

Если за время повторной передачи ответа класса 2xx, равное 64·T1, подтверждение ACK не приходит, диалог устанавливается, но сессия должна быть разрушена. Это происходит путём отсылки сообщения BYE, как описано в разделе 2.3.8.

2.3.7. Процедуры модификации сессий

Успешный запрос INVITE устанавливает диалог между двумя агентами пользователя и сессию, используя модель **offer/answer**. В разделе 2.3.1. описывалось, как модифицировать существующий диалог, используя запрос, обновляющий текущий адрес удалённого пользователя (**remote target**). В этом параграфе рассматривается вопрос модификации действующей сессии. Такая модификация может затрагивать изменение адресов или портов, добавление или удаление медиапотока и т.д. Это выполняется путём отправки запроса INVITE в рамках того же диалога, что установил сеанс связи. Запрос INVITE, который отправлен в рамках существующего диалога, называется re-INVITE.

Заметим, что одно сообщение re-INVITE может в одно и тоже время модифицировать и диалог, и параметры сессии. Модифицировать сессию способен как вызывающий, так и вызываемый пользователь.

Работа UAC

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

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

Если формат описания сессии предусматривает возможность нумерации версий - присвоения порядкового номера SDP-описанию, предложенному в ходе вызова, инициатор модификации сессии должен указать, что версия описания сессии 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 заголовок Alert-Info или тело сообщения со значением *alert* в заголовке Content-Disposition в запрос re-INVITE, поскольку UAS обычно не оповещает пользователя при получении re-INVITE.

В отличие от INVITE, re-INVITE не может быть размножен во время прохождения 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 публичный адрес.

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

1. Если существует действующая клиентская INVITE транзакция, TU должен подождать, пока она перейдёт в состояние «Completed» или «Terminated», для того, чтобы инициировать новый запрос INVITE.

2. Если существует действующая серверная INVITE транзакция, TU должен подождать, пока она перейдёт в состояние «Confirmed» или «Terminated», для того, чтобы инициировать новый запрос INVITE.

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

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

1. 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.

2. 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 ещё одну попытку отсылки re-INVITE, если до этих пор сохранилась необходимость модификации сессии. Например, в случае если произошло завершение вызова, сопровождающееся оправкой сообщения BYE, re-INVITE не будет отправлен.

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

Поведение UAS

UAS, получающий второй запрос INVITE до того, как отправлен окончательный ответ на первый запрос INVITE с меньшим порядковым номером в заголовке CSeq в том же диалоге, возвращает ответ с кодом 500 (Server Internal Error) на второй INVITE 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 данным UAS, отсылает ответ с кодом 491 (Request Pending) на полученный второй запрос.

Если UAS получает re-INVITE для существующего диалога, он должен проверить все идентификаторы версии в описании сессии, а при их отсутствии – содержимое описания сессии, чтобы определить, изменилось оно или нет. Если описание сеанса изменилось, UAS должен настроить соответствующие параметры сессии, возможно после запроса подтверждения пользователя.

Если новое описание сессии неприемлемо, UAS может отклонить его, возвратив ответ с кодом 488 (Not Acceptable Here) на re-INVITE. Такой запрос должен содержать заголовок

Warning.

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

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

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 отсылки информации **offer**, обновляющей существующую сессию, как описано в "An Offer/Answer Model with SDP", RFC 3264 в случае использования протокола SDP. T.e. это значит, что информация **offer** должна включать столько форматов и типов медиа-информации, сколько поддерживает UA. UAS должен гарантировать, что описание сессии частично совпадает с предыдущим описанием – форматами медиа-информации, типом транспортного протокола, или другими параметрами, которые поддерживаются UAC. Это делается для того, чтобы избежать возможного отклонения описания сессии UAC. Если всё же это описание сессии не приемлемо для UAC, он должен сформировать **answer** с надлежащим описанием сеанса и затем отослать BYE для разрушения сессии.

2.3.8. Процедуры разрушения сессий

Этот параграф описывает процедуры прерывания сессий связи, установленных по протоколу SIP. Состояние сессии и состояние диалога очень тесно связаны. Когда сессия инициируется сообщением INVITE, каждый 1xx или 2xx ответ от отдельного UAS создаёт диалог, и, если ответ завершает обмен **offer/answer**, создаёт сессию. В итоге, каждая сессия связывается с одним диалогом. Если на начальный запрос INVITE придёт не-2xx окончательный ответ, он разрушит все сессии и диалоги (если они существовали), созданные посредством ответов на запрос. Завершая транзакцию, не-2xx окончательный ответ также предотвращает создание сессий как результата отправки запроса INVITE.

Запрос BYE используется для завершения отдельных сессий. В этом случае отдельная сессия – это сессия с другим UA, участником диалога. Когда участнику диалога приходит сообщение BYE, все сессии, связанные с данным диалогом должны быть разрушены. UA не может отослать BYE вне диалога. UA вызывающего пользователя может послать BYE как в установленном диалоге, так и в диалогах, находящихся на «ранней стадии», а UA вызываемого пользователя может отправить BYE только в установленном диалоге. Однако UA вызываемого пользователя не должен отправлять BYE до тех пор, пока не получил подтверждения ACK на свой ответ класса 2xx. Если нет расширений SIP, определяющих другие состояния прикладного уровня, связанные с диалогом, сообщение BYE также

разрушает диалог.

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

Обычно если пользователь вешает трубку, это указывает на то, что он хочет завершить попытку установления сеанса связи и разрушить все существующие сеансы связи. Для UA вызывающего пользователя это будет подразумевать отсылку сообщения CANCEL, если 2xx ответ на первоначальный INVITE ещё не пришёл, или отсылку сообщения BYE, если окончательный ответ был принят. Для UA вызываемого пользователя обычно это подразумевает отправку сообщения BYE. Однако вызываемый пользователь может повесть трубку до прихода подтверждения ACK на ответ класса 2xx – тогда оправка сообщения BYE будет удержана программным модулем SIP до прихода ACK в целях правильного завершения соединения. Если определённый интерфейс пользователя позволяет пользователю отклонить вызов, не отвечая на него, то это выполняется отправкой ответа с кодом 403 (Forbidden) или 486 (Busy Here), сообщение BYE в соответствии с приведёнными выше правилами отослано быть не может.

Разрушение сессии с помощью запроса BYE

Работа UAC

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

Работа UAS

Сначала UAS обрабатывает запрос BYE с использованием общих процедур обработки

UAS. Ядро UAS, получившее запрос BYE, проверяет его на соответствие существующему диалогу. Если запрос не соответствует диалогу, ядро UAS должно сформировать ответ с кодом 481 (Call/Transaction Does Not Exist) и отослать его серверной транзакции. Это означает что, если UAC послал запрос BYE без параметров «tag» в соответствующих заголовках, UAS его отклоняет.

Ядро UAS, получившее запрос BYE для существующего диалога, должно выполнить процедуры по обработке запроса в диалоговом режиме. После этого UAS разрушает сессию и заканчивает передавать и ожидать прихода медиа-информации. Ядро UAS формирует ответ класса 2xx и передаёт его серверной транзакции для дальнейшей доставки по сети.

2.4. Назначение и функ циональ ность прокси-сервера

SIP прокси-серверы – это элементы сети SIP, которые маршрутизируют SIP-запросы серверам агента пользователя и SIP-ответы – клиентам агента пользователя. Запрос может проходить несколько прокси-серверов на пути к UAS. Каждый из них будет принимать решения 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 непосредственного выполнения функций прокси-сервера. Элемент может отослать ответ с подходящим кодом ошибки. Отвечая напрямую на запрос, 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 (stateless) состояний для каждого нового запроса. Сервер без сохранения состояний работает как ретранслирующий узел сети. Он пересылает каждый запрос следующему элементу, принимая решения о маршрутизации исходя из информации, содержащейся в запросе. Полученные ответы он просто пересылает обратно. Прокси-серверы без сохранения состояний удаляют информацию о прошедшем сообщении, как только сообщение было ретранслировано. Прокси-серверы с сохранением состояний хранят информацию (состояние транзакции) о каждом входящем запросе и о каждом отосланном запросе, возникающем вследствие обработки входящего запроса. Stateful прокси-сервер может принять решение о размножении запроса, передавая его на несколько различных 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 задействуя

сохранение транзакционных состояний. Например, прокси-сервер может передавать запрос через определенное TCP соединение без сохранения транзакционных состояний до тех пор, пока он способен поместить достаточный объём информации в сообщение для того, чтобы ответ мог быть передан обратно тому соединению, по которому пришёл запрос. Запросы, передающиеся со сменой транспортного протокола, когда TU прокси-сервера играет ведущую роль в обеспечении надёжной 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 stateless и stateful режиме идентичны. Следующий раздел описывает действия прокси-сервера в stateful режиме. В разделе, описывающем stateless режим, будут указаны изменения по сравнению со stateful режимом.

2.4.1. Функциональность прокси-сервера с сохранением состояний

В режиме с сохранением состояний прокси-сервер действует как механизм обработки SIP-транзакций. При функционировании прокси-сервер задействует 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. 2.20). Входящие запросы обрабатываются серверной транзакцией. От серверной транзакции запросы направляются ядру прокси-сервера. Ядро прокси-сервера определяет, куда маршрутизировать запрос, выбирая один или несколько мест назначения для следующей пересылки запроса.

Рис. 2.34 Модель stateful прокси-сервера.

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

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

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

1. Проверка правильности составления запроса
2. Предварительная обработка маршрутной информации
3. Определение адреса (адресов) для пересылки
4. Пересылка запроса
5. Обработка всех ответов

2.4.1.1 Проверка правильности составления запроса

Перед тем, как прокси-сервер начнёт работать с запросом, он должен проверить правильность составления запроса. Сообщение должно пройти следующие проверки:

1. Проверка корректности синтаксиса
2. Проверка схемы URI
3. Проверка заголовка Max-Forwards

4. Проверка на наличие замкнутого пути (опционально)

5. Проверка заголовка Proxy-Require

6. Проверка заголовка Proxy-Authorization

Если хотя бы одна из проверок выявляет ошибку, элемент должен действовать как сервер агента пользователя и отослать 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 (см. параграф 2.1.2.1)

1. Проверка корректности синтаксиса

Запрос должен быть правильно составлен, чтобы быть поддержанным серверной транзакцией. Все компоненты запроса, задействованные в функциях «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 нём содержится неверно составленный заголовок Date. Более того, прокси-сервер не будет удалять данный заголовок при пересылке запроса.

Существующая версия протокола SIP предусматривает дальнейшее расширение. Новые расширения могут определять новые типы запросов и заголовки. Прокси-сервер не должен отклонять запросы из-за того, что ему не известны определённые типы запросов или заголовки.

2. Проверка схемы URI

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

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 возвращает ответ с кодом 483 (Too many hops).

4. Проверка на наличие замкнутого пути

Прокси-сервер может проверить запрос на наличие замкнутого пути (петли) перед продвижением. Если запрос содержит заголовок Via, и в значении этого заголовка имя хоста и номер порта совпадают со значениями, помещенным данным прокси-сервером в предыдущие запросы, то запрос проходил через этот прокси-сервер ранее и либо движется 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 изменённым. Различие в поле Request-URI определит принятие прокси-сервером решения о маршрутизации к другому месту назначения. Распознать замкнутый путь можно путём вычисления параметра «branch» данного сообщения, (процедура вычисления будет описана позже), и сравнения его с аналогичным параметром в том 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 возвратить ответ с кодом 482 (Loop Detected).

5. Проверка заголовка Proxy-Require

Будущие расширения протокола SIP могут представлять 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 не понятны.

Если запрос содержит заголовок Proxy-Require с одним или несколькими значениями, непонятными прокси-серверу, он должен вернуть ответ с кодом 420 (Bad Extension). Ответ должен включать заголовок Unsupported, перечисляющий те идентификаторы **option-tag**, которые не были распознаны прокси-сервером.

6. Проверка заголовка Proxy-Authorization

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

2.4.1.2 Предварительная обработка маршрутной информации

Прокси-сервер должен изучить поле Request-URI запроса. Если поле содержит значение, идентифицирующее данный прокси-сервер, он осуществляет замену значения поля Request-URI на последнее значение заголовка Route с последующим удалением последнего значения заголовка Route (см. пример в параграфе 2.4.4.2). Затем прокси-сервер продолжает работать так, как если бы он получил этот изменённый запрос. Это возможно если SIP-элементом, отсылающим запрос прокси-серверу, является **strict-router** (см. глоссарий). Кроме того, механизм перезаписи позволяет сохранить значение поля Request-URI при прохождении 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, выставленные не по умолчанию, и продолжает обработку так, как будто эти значения отсутствовали. Запрос, содержащий значение параметра «maddr», соответствующее адресу прокси-сервера, может придти на порт или 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.

2.4.1.3 Определение адресов пересылки

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

Если Request-URI в запросе содержит параметр «maddr», Request-URI должен быть помещён 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 несёт 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 HTTP прокси-сервер, 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 отослать 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 выполнения алгоритмической подстановки Request-URI. При обращении к серверу определения местоположения Request-URI должен быть первоначально приведён к каноническому виду для дальнейшего поиска по 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 возвращает ответ с кодом 485 (Ambiguous). Ответ должен содержать заголовок, содержащий URI новых адресов для последующих попыток. Например, запрос INVITE, адресованный на sip:vladimir@protei.ru может быть неясным прокси-серверу, поскольку в базе данных сервиса 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, времени суток поступления запроса, интерфейса, на который пришёл запрос, ошибки предыдущего запроса и даже от текущего уровня загруженности прокси-сервера. Если перечисленные причины каким-либо образом обуславливают наличие определённых адресов, то эти адреса добавляются в перечень. URI могут быть помещены в перечень единожды.

Прокси-сервер не должен помещать дополнительные URI в перечень, 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. Он может использовать любую информацию, полученную при обработке для определения новых адресов. Например, прокси-сервер может добавлять в перечень адресов контактные адреса, полученные в ответе перенаправления класса 3xx. Если прокси-сервер использует динамический источник информации во время создания перечня адресов (к примеру, если он обращается к SIP серверу регистрации), он должен следить за изменениями в источнике на протяжении всего времени обработки запроса. Поэтому, как только у пользователя появляется новый контактный адрес, он сразу должен быть включён в перечень

адресов.

Если поле Request-URI указывает ресурс на прокси-сервере, который не существует, он должен возвратить ответ с кодом 404 (Not Found).

Если перечень адресов остаётся пуст после проведения вышеперечисленных операций, прокси-сервер должен отправить ответ с кодом ошибки 480 (Temporarily Unavailable).

2.4.1.4 Пересылка запроса

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

Простейшим механизмом упорядочивания является механизм, основанный на использовании для целей значения параметра «q», который получен из поля заголовка Contact. Адреса обрабатываются в очерёдности 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 создаётся ядром прокси-сервера перед пересылкой первого запроса.

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

1. Создаёт копию полученного запроса.
2. Обновляет содержимое поля Request-URI.
3. Обновляет содержимое заголовка Max-Forwards.
4. Опционально добавляет значение заголовка Record-route.
5. Опционально добавляет дополнительные заголовки.
6. Выполняет заключительную обработку маршрутной информации.

7. Определяет адрес, номер порта и тип транспортного протокола для пересылки.

8. Добавляет значение в заголовок Via.

9. Добавляет заголовок Content-Length в случае необходимости.

10. Пересылает новый запрос.

11. Устанавливает таймер C.

1. Копирование запроса

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

2. **Обновление содержимого поля Request-URI**

Значение поля Request-URI в стартовой строке копии должно быть заменено на URI из перечня адресов. Если URI содержит параметры, запрещённые для Request-URI, они должны быть удалены. При некоторых обстоятельствах полученный Request-URI помещается в перечень адресов, не будучи изменённым. Для такого URI указанная замена ничего не меняет.

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.

4. Добавление значения Record-Route (опционально)

Если прокси-сервер требует, чтобы путь следования будущих запросов диалога, который будет создан данным запросом, проходил через него, он должен поместить в копию свое значение заголовка Record-Route перед существующими значениями заголовка, даже если уже присутствует заголовок Route. Запросы, устанавливающие диалог, могут содержать предустановленный заголовок Route.

Если запрос передаётся в режиме диалога, прокси-сервер также может вставлять свое значение заголовка Record-Route, если он желает, чтобы последующие запросы этого диалога прошли через него. Значения поля этого заголовка не окажут никакого влияния на установленные 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 удалён из пути, когда конечная точка после возможного сбоя восстановит диалог.

Прокси-сервер может поместить значение заголовка Record-Route в любой запрос. Если запрос не инициирует диалог, терминалы проигнорируют 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 своё значение.

URI, помещённый в качестве значения в заголовок Record-Route, должен быть SIP или SIPS URI. Этот URI должен содержать параметр «lr», обозначающий, что данный прокси-сервер использует поддерживает стандартные механизмы маршрутизации, определённые в RFC 3261. Адрес прокси-сервера может быть различным 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 принятия решений 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 для SIP-реализации: SIP URI и транспортный протокол – TCP или UDP. URI, помещаемый в поле Record-Route, должен однозначно определять SIP прокси-сервер, поместивший его, для того чтобы последующие запросы прошли через этот прокси-сервер.

Если Request-URI содержит SIPS URI или верхнее значение заголовка Route (после выполнения заключительной обработки, описанной в п.6) содержит SIPS URI, то адрес, помещённый 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 создаёт запрос, не содержащий SIPS URI в поле Request-URI или верхнем значении заголовка Route (после выполнения заключительной обработки, описанной в пункте 6), 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 приёме 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; 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 2.3.1 дана более детальная информация, о том, как конечные точки используют значения заголовка Record-Route.

Запись маршрута посредством заголовка Record-Route может быть востребована некоторыми сервисами, где 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 определённого сервиса.

5. Добавление дополнительных заголовков

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

6. Заключительная обработка маршрутной информации

Прокси-сервер может вести внутреннюю политику, которая предусматривает, чтобы запрос прошёл через группу прокси-серверов перед тем, как быть доставленным к месту назначения. Прокси-сервер должен гарантировать, что все эти серверы придерживаются стандартных процедур по обработке 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 помещён в заголовок Route копии над существующими значениями в случае их присутствия. Если заголовок Route отсутствует, он должен быть добавлен с содержащимся в нём перечнем URI.

Если прокси-сервер ведёт внутреннюю политику, которая предписывает, чтобы запрос прошёл через один определённый прокси-сервер, альтернативным вариантом помещения значения в заголовок Route является обход логики пересылки, описанной в пункте 10, путём лишь отсылки запроса на адрес, порт по транспортному протоколу, определённому прокси-серверу. Если запрос уже содержит заголовок Route, этот альтернативный вариант может быть использован, только если известно, что следующий прокси-сервер придерживается стандартных процедур по обработке заголовка Route, как описано выше. Однако механизм

вставки значений в заголовок Route предпочтительнее данного подхода из-за своей надёжности, гибкости, универсальности и последовательности действия. Более того, если поле Request-URI содержит SIPS URI должен быть использован протокол TLS при взаимодействии с этим прокси-сервером.

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:

- прокси-сервер помещает содержимое поля Request-URI в качестве последнего значения в поле заголовка Route.

- 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 для 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).

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

Внутренняя политика прокси-сервера может предписывать отсылать запрос на определённый IP-адрес, порт транспортного протокола 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 по обработке заголовка Route. Однако такой механизм пересылки запроса на определённом 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 по поиску SIP сервера, которому следует отослать запрос («Locating SIP Servers» RFC 3263). Если прокси-сервер модифицировал запрос для его отсылки на **strict-router** в соответствии с параграфом 2.4.1.2, процедуры должны быть применены к значению в поле Request-URI. Если запрос не был изменён, процедуры применяются к первому значению заголовка Route. Выполнение процедур DNS-поиска даст упорядоченные наборы взаимосвязанных величин

(адрес, порт, транспортный протокол). Не зависимо 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, и 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 отсылает запрос, используя новую клиентскую транзакцию, как показано в следующих пунктах 8 – 10. Поскольку каждая попытка приводит к созданию новой транзакции, она представляет новую ветвь. Таким образом, параметр «branch», обеспечиваемый в поле заголовка Via и вставляемый на следующем, 8-ом этапе, будет различным для каждой попытки.

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 вернул окончательный ответ с кодом 408 (Request Timeout).

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 в глобальном масштабе и будет содержать необходимую комбинацию «magic cookie».

Прокси-серверы, сконфигурированные таким образом, чтобы обнаруживать петли, имеют дополнительное ограничение для значения, которое они используют для формирования параметра «branch». Такой прокси-сервер в реализации должен создавать параметр «branch», разделённый на две части. Первая часть должна удовлетворять требованиям, описанным в пункте 2.1.1.1., вторая – служит для отличия спиралей от петель.

Обнаружение петель осуществляется путём проверки на изменение полей, влияющих на обработку, вернувшегося на прокси-сервер запроса (если изменений нет, то замкнутый маршрут присутствует). Значение, помещённое во вторую часть параметра «branch», должно отражать все эти поля (включая поля заголовков Route, Proxy-Require и Proxy-Authorization). Это гарантирует то, что если запрос был маршрутизирован обратно на проки-сервер и одно из полей изменилось, он расценивается как спираль, а не петля. Общепринятый способ создания этого значения – это подсчёт 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. Также при подсчёте контрольной суммы могут учитываться значения заголовков Proxy-Require и Proxy-Authorization в случае их присутствия. Алгоритм хэширования, используемый для подсчёта контрольной суммы, зависит от конкретной

реализации, но предпочтительным является алгоритм MD5 (RFC 1321), 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 приём запроса или маршрутизацию. Это необходимо для того, чтобы отличить запросы с петлёй от запросов, чьи параметры маршрутизации изменились перед возвращением на этот сервер (запросы, проходящие по спирали).

Тип запроса не должен использоваться при подсчёте 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.

9. Добавление заголовка Content-Length в случае необходимости

Если запрос будет отослан 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.

10. Пересылка запроса

Stateful прокси-сервер создаёт новую клиентскую транзакцию для этого запроса и указывает транзакции отправить запрос, используя адрес, порт и тип транспортного протокола, определённые на шаге 7.

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 2.4.1.5 описывает, как значение этого таймера обновляется с помощью предварительных ответов, а параграф 2.4.1.6 описывает обработку при его срабатывании.

2.4.1.5 Обработка ответов

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

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

1. Обнаружение буфера ответов

2. Перезапуск таймера C при помощи предварительных ответов.

3. Удаление верхнего значения заголовка Via.

4. Добавление ответа в буфер ответов.

5. Проверка на необходимость немедленной отправки ответа.

6. При необходимости выбор наилучшего окончательного ответа из буфера ответов.

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 над каждым пересылаемым ответом.

7. Объединение значений в заголовке Authorization (при необходимости).

8. Перезапись значений заголовка Record-Route (опционально).

9. Пересылка ответа.

10. Создание необходимых запросов CANCEL.

1. Обнаружение буфера ответов

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

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 для данной клиентской транзакции. Таймер может быть перезапущен с любым значением, которое больше 3 минут.

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 для этого элемента и не должен пересылаться. Дальнейшая обработка, описанная далее, не производится над этим сообщением, вместо этого выполняются стандартные процедуры UAC по обработке ответов вне диалога (обработка транспортного уровня SIP уже выполнена).

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 данным буфером ответов. Даже если ответ не был выбран, отдельная информация из него может потребоваться для формирования «наилучшего» ответа.

Если прокси-сервер решит послать запрос на часть адресов, указанных в заголовке Contact полученного ответа класса 3xx, путём добавления их в перечень адресов (target set), он должен удалить эти контактные адреса из 3xx ответа перед сохранением ответа 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 посылать запрос на все контактные адреса, полученные из ответа класса 3xx, он не должен добавлять в буфер данный 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 послать 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 отосланного запроса, но преобразованный под схему «sip». В случае использования схемы «tel» это выполняется путём перемещения части номера телефона в tel URL в пользовательскую часть SIP URI и указания в части хоста SIP URI имени домена, куда был отослан предыдущий запрос.

Также как и 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. Проверка ответа на необходимость немедленной пересылки

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 незавершённые клиентские транзакции и не должен создавать новые ветви для данного вызова. Для отмены клиентских транзакций прокси-сервер отправляет по соответствующим направлениям запрос CANCEL, что как правило приводит к получению ответов с кодом 487 от всех незавершённых клиентских транзакций, и только после этого далее передаётся 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 пункте 4 параграфа 2.4.1.5. для последующего выбора и передачи «наилучшего» ответа.

Любой ответ, требующий немедленной пересылки, должен быть обработан, как описано в пунктах 7 и 8.

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

6. Выбор «наилучшего» ответа из буфе ра ответов

Stateful прокси-сервер должен передать окончательный ответ на серверную транзакцию, которой соответствует данный буфер ответов, если ни один окончательный ответ до этого момента не был передан, и все клиентские транзакции, связанные с данным 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 отослать серверной транзакции ответ с кодом 408 (Request Timeout). В противном случае прокси-сервер должен передать один из ответов, сохранённых 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 обеспечивают информацию, влияющую на очередную передачу запроса, таким как 401, 407, 415, 420 и 484, если выбран 4xx класс.

Прокси-сервер, получивший ответ с кодом 503 (Service Unavailable), не должен пересылать далее до тех пор, пока не сможет определить, что любые последующие 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 на адрес, указанный 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 пунктах 7 и 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 от разных терминалов на запрос, инициирующий диалог. Прокси-сервер не должен вставлять параметр «tag» в заголовок To 1xx или 2xx ответа, если запрос его не содержал. Также прокси-сервер не должен модифицировать параметр «tag» в заголовке To 1xx или 2xx ответа.

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 100 (Trying) (ответы 101-199 должны содержать параметр «tag» в заголовке To). Однако прокси-сервер может переслать запрос серверу агента пользователя. Этот UAS отправляет предварительные ответы, входя при этом в режим диалога на «ранней стадии» с инициатором запроса.

Ответы класса с 3xx по 6xx доставляются последовательно, путём ретрансляции (hop-by-hop). Во время передачи такого ответа прокси-сервер работает как UAS, передающий ответ; он формирует свой собственный ответ на основе ответов, полученных от элементов, расположенных ниже по сети. Прокси-сервер не должен изменять «tag» в заголовке To при пересылке 3-6xx ответа 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 SIP элементов, расположенных выше по сети, не имеет значения заменяет ли прокси-сервер «tag» в заголовке To пересылаемого 3xx-6xx ответа, 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 объединении значений заголовков Contact из незакодированных и неаутентифицированных ответов класса 3xx.

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 (Unauthorized) и 407 (Proxy Authentication Required), полученных до настоящего времени в буфер ответов, и перед пересылкой добавить их без изменений в этот ответ. Итоговый 401 или 407 ответ 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 отправке запроса снабдить его требуемой информацией.

8. Перезапись значений заголовка Record-Route

Если выбранный ответ содержит значение заголовка Record-Route, помещённое этим прокси-сервером при прохождении запроса, прокси-сервер может изменить значение перед пересылкой ответа. Это позволяет прокси-серверу иметь два различных адреса - для SIP-

элемента, находящегося 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 отослал его через TLS, прокси-сервер должен перезаписать URI в Record-Route ответа так, чтобы он стал SIP URI.

Новый URI, помещённый в заголовок прокси-сервером, должен удовлетворять ограничениям, наложенным на значения заголовка Record-Route запросов (пункт 4 параграф 2.4.1.4) со следующими изменениями: URI не должен содержать параметра «transport» в случае, если прокси-сервер не знает, что следующий элемент сети SIP, находящийся на пути последующих 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, чей идентификатор в пользовательской части URI соответствует этому прокси-серверу.

Запись значения заголовка Record-Route в запрос прокси-сервером не обязательно приведёт к тому, что ответ будет включать заголовок Record-Route. Если ответ содержит заголовок Record-Route, он будет содержать значение, добавленное прокси-сервером.

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 пункте 3 параграфа 2.4.1.5. 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 приведёт к отсылке 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 отправке 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.

10. **Создание запросов CANCEL**

Если пересланный ответ был окончательным, прокси-сервер должен создать запрос CANCEL для всех незавершённых клиентских транзакций, связанных с этим буфером ответов. Аналогичные действия прокси-сервер предпринимает, когда получает ответ класса 6xx. Незавершённая клиентская транзакция – это транзакция, получившая предварительный ответ, но не получившая окончательного ответа (находится в состоянии «Proceeding») и не предавшая созданный для неё запрос CANCEL. Создание запроса CANCEL описано в разделе 2.3.5.

Отмена незавершённых клиентских транзакций при пересылке окончательного ответа не гарантирует, что оконечная точка не получит ответы с кодом 200 (OK) на запрос INVITE. Ответы 200 (OK) могут быть созданы более чем на одной ветви до того, как запрос CANCEL будет отослан и обработан.

2.4.1.6 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 создаёт запрос CANCEL, соответствующий данной 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, чрезмерной загрузки.

2.4.1.7 Обработка ошибок транспортного уровня SIP

Если транспортный уровень SIP уведомляет прокси-сервер об ошибке, когда тот пытается переслать запрос (см. параграф 2.10.4), прокси-сервер должен вести себя так, как если бы на пересылаемый запрос пришёл ответ с кодом 503 (Service Unavailable). Если прокси-сервер извещается об ошибке при попытке переслать ответ, он отбрасывает ответ. Прокси-сервер не должен отменять незавершённые клиентские транзакции, связанные с данным буфером ответов, после получения извещения.

2.4.1.8 Обработка запроса CANCEL

Stateful прокси-сервер может сформировать запрос CANCEL на любой другой запрос, созданный им ранее. Прокси-сервер должен отменить все незавершённые клиентские транзакции, связанные с буфером ответов, когда получает соответствующий запрос CANCEL.

Stateful прокси-сервер может создать запросы CANCEL для незавершённой INVITE-клиентской транзакции на основании истечения интервала времени, определённого в заголовке Expires запроса INVITE. Однако, это часто не требуется, поскольку оконечные точки, участвующие в соединении, проинформируют 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 создаётся. 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 возвратить ответ с кодом 200 (OK) на запрос CANCEL. В этом случае прокси-сервер выполняет функции в соответствии с правилами для UAS, описанными в разделе 2.1.2. Более того, прокси-сервер должен создать запросы CANCEL для всех незавершённых клиентских транзакций, связанных с данным буфером ответов, как описано в параграфе 2.4.1.5, пункт 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 не был найден буфер ответов, может являться то, что прокси-сервер передал отменяемый запрос без сохранения состояний).

2.4.2. Функциональность 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 – напрямую от транспортного уровня протокола SIP (См раздел 2.10.). В результате они не могут передавать повторные сообщения, созданные самостоятельно. Они только пересылают все повторные сообщения, которые получают; повторные передачи пересылаются также 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 параграфе 2.4.1.1.
Stateless прокси-сервер должен выполнять этапы обработки, описанные в параграфах 2.4.1.2 и 2.4.1.3. со следующими исключениями:

- Stateless прокси-сервер выбирает только один адрес из перечня адресов target set. Этот выбор должен зависеть только от информации, заложенной в сообщении и статичных свойств сервера. В частности, повторно переданный запрос должен отсылаться по одному и тому же направлению каждый раз при обработке. Кроме того, запрос ACK, не содержащий заголовка Route, и запрос CANCEL должны быть отосланы по тому же направлению, что и связанный с ними запрос INVITE.

Stateless прокси-сервер должен выполнять этапы обработки, описанные в параграфе 2.4.1.4 со следующими исключениями:

- Требования для уникальных идентификаторов, содержащихся в параметре «branch» также применимы 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 пункте 8 параграфа 2.4.1.4. Это происходит из-за того, что повторные передачи запроса должны иметь тоже значение параметра, а stateless прокси-сервер не в силах отличить повторную передачу от оригинального запроса. 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 параметров сообщения, которые неизменны при повторных передачах.

  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 путём хэширования значения полученного параметра «branch». В противном случае первый компонент значения параметра «branch» подсчитывается путём 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, определённые в параграфе 2.4.1.4, 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 передачи запроса. Для параметра «branch» заголовка Via это подразумевает, что преобразования должны основываться на статичной конфигурации прокси-сервера или свойствах запроса, неизменных при повторной передаче.

- Stateless прокси-сервер определяет, куда переслать запрос так же, как это описано для stateful прокси-сервера в пункте 10 параграфа 2.4.1.4. Запрос отсылается напрямую транспортному уровню SIP вместо того, чтобы проходить через клиентскую транзакцию.

  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 подсчётов только информацию из самого сообщения и данные из статичной 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 величине таймаута 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 параграфе 2.4.1.5., не осуществляется 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 этот 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.

2.4.3 Работа с заголовком Route и полем Request-URI

Действия, которые выполняет прокси-сервер над запросом, содержащим заголовок Route, могут быть разделены на следующие шаги.

1. Прокси-сервер анализирует поле Request-URI. Если оно указывает ресурс, обслуживаемый данным прокси-сервером, он заменяет его на URI, полученные в результате обращения к серверу определения местоположения. В противном случае прокси-сервер не изменит значение в поле Request-URI.

2. Прокси-сервер анализирует URI в верхнем значении заголовка Route. Если URI определяет данный прокси-сервер, он удаляет это значение из заголовка Route.

3. Прокси-сервер отошлёт 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 («Locating SIP Servers» RFC 3263).

Если на пути запроса не встречается **strict-router**, поле Request-URI всегда будет отражать URI адресата запроса.

2.4.4 Примеры

2.4.4.1 Взаимодействие через исходящий и входящий прокси-серверы

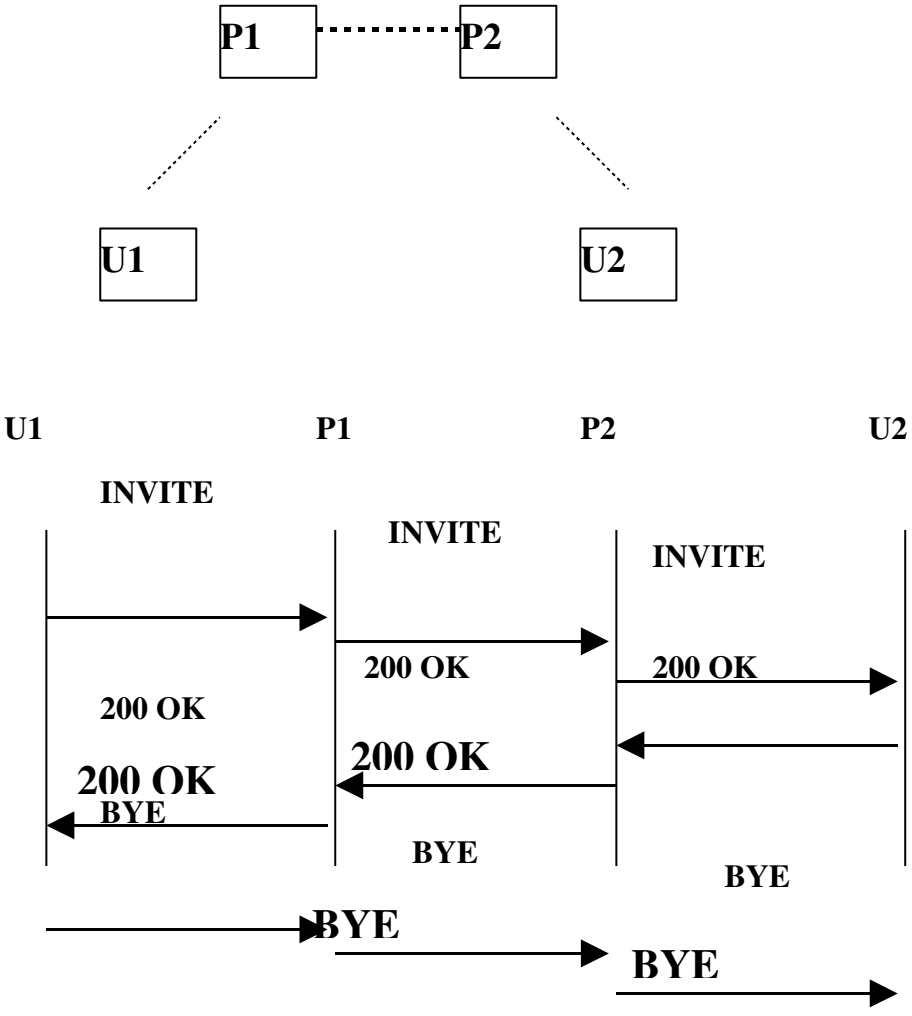


Рис. 2.35. Взаимодействие через исходящий и входящий прокси-серверы

Данный сценарий является примером обмена сообщениями через два прокси-сервера: исходящий и входящий - U1 -> P1 -> P2 -> U2, в котором прокси-серверы записывают свои значения в заголовок Record-Route.

U1 (Агент пользователя 1) посылает P1:

```
INVITE sip:vladimir@protei.ru SIP/2.0
Contact: sip:anton@u1.niits.ru
```

P1 – исходящий прокси-сервер. P1 не отвечает за protei.ru, поэтому он отыскивает адрес прокси-сервера, обслуживающего protei.ru с помощью DNS-процедур и отсылает запрос по полученному адресу. Также он добавляет 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. Он обслуживает домен protei.ru, поэтому он обращается к серверу определения местонахождения и перезаписывает значение в поле Request-URI. Также он добавляет значение заголовка Record-Route. Заголовок Route отсутствует, поэтому новое значение Request-URI определяет, куда отсылать 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 передаёт 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 устанавливает компонент состояния диалога **remote target** в значение `sip:anton@u1.niits.ru`, и другой компонент - **route set** в значение `<sip:p2.protei.ru;lr>,<sip:p1.niits.ru;lr>`.

Ответ пересылается от прокси-сервера 2 к прокси-серверу 1 и Агенту пользователя 1. Теперь UA устанавливает своё 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 отсылать запрос. Запрос отсылается 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:

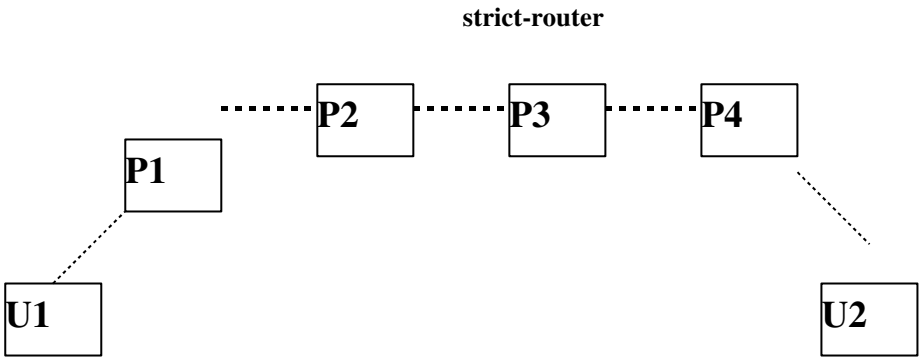
```
BYE sip:vladimir@u2.protei.ru SIP/2.0
Route: <sip:p2.protei.ru;lr>
```

P2 также определяет, что 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
```

2.4.4.2 Прохождение сообщений через strict-router

В этом сценарии диалог устанавливается через 4 прокси-сервера, каждый из которых добавляет своё значение в заголовок Record-Route. Третий прокси-сервер - **strict-router.**



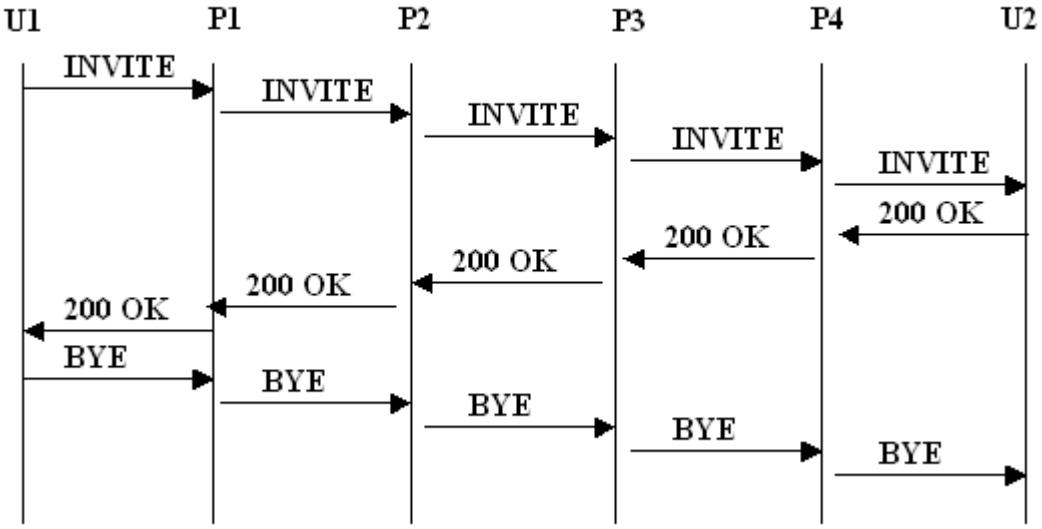


Рис. 2.36 Прохождение сообщений через strict-router

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 посылает на него ответ с кодом 200 (OK). Позже U2 посылает 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 – он удаляет это значение. Затем он готовится отправить запрос на основании нового первого значения заголовка Route - `sip:p3.loniis.ru`, но обнаруживает, что этот URI не содержит параметра «lr». Поэтому перед отправкой 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 отсылает запрос P1 на основании первого значения в заголовке Route.

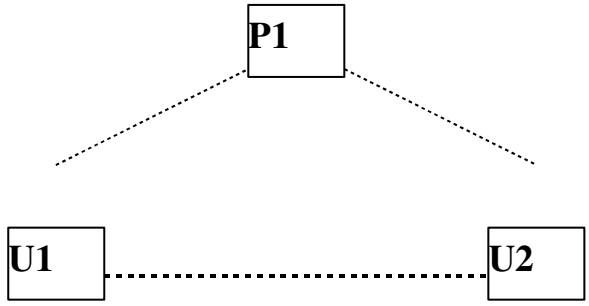
P1 обнаруживает свой адрес в первом значении заголовка Route, поэтому он удаляет это значение, получая:

```
BYE sip:anton@u1.niits.ru SIP/2.0
```

Так как P1 не отвечает за `u1.niits.ru` и заголовок Route отсутствует, P1 перешлёт запрос на `u1.niits.ru` на основании Request-URI.

2.4.4.3 Прохождение сообщений через прокси-сервер с перезаписью значения в заголовке Record-Route

В этом сценарии U1 и U2 находятся в разных частных пространствах имён; они устанавливают диалог через прокси-сервер P1, который работает, как шлюз между пространствами имён.



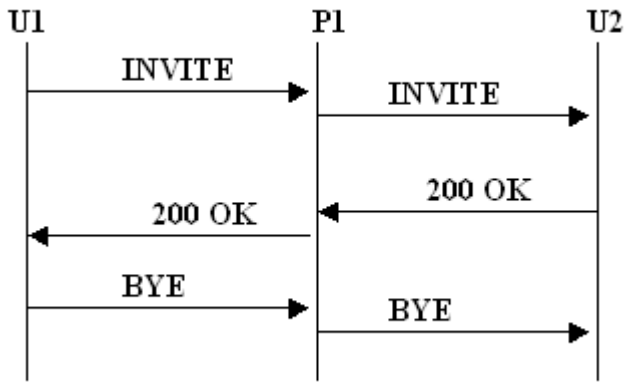


Рис. 2.37 Прохождение сообщений через прокси-сервер с перезаписью значения в заголовке Record-Route

U1 отправляет:

```
INVITE sip:vladimir@gateway.niits.ru SIP/2.0
Contact: <sip:anton@u1.niits.ru>
```

P1 обращается к серверу определения местоположения и отсылает U2 следующее:

```
INVITE sip:vladimir@protei.ru SIP/2.0
Contact: <sip:anton@u1.niits.ru>
Record-Route: <sip:gateway.protei.ru;lr>
```

U2 отправляет 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 своё значение в заголовке Record-Route, чтобы обеспечить значение, которое будет полезно для U1, и отсылает следующее U1:

```
SIP/2.0 200 OK
Contact: <sip:vladimir@u2.protei.ru>
Record-Route: <sip:gateway.niits.ru;lr>
```

Позже U1 отсылает P1 следующий запрос BYE:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
Route: <sip:gateway.niits.ru;lr>
```

который P1 отсылает U2 в виде:

```
BYE sip:vladimir@u2.protei.ru SIP/2.0
```

2.5. Назначение и функ циональ ность сервера перенаправ ления

Для некоторых сетевых архитектур может оказаться полезным понизить загрузку прокси-серверов, которые отвечают за маршрутизацию запросов, и осуществить доставку начального сообщения, используя сервер перенаправления. Сервер перенаправления помещает информацию маршрутизации для поступившего запроса в ответ, предназначенный клиенту. Клиент, получивший перенаправляющий ответ от данного сервера, снова отсылает запрос, но 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 передаёт ему список альтернативных адресов; затем он возвращает клиенту ответ класса 3xx. Для правильно составленных запросов CANCEL он возвращает ответ класса 2xx. Окончательный ответ завершает эту SIP-транзакцию. На протяжении всей транзакции сервер перенаправления сохраняет её 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 возможные местоположения адресуемого пользователя или просто дополняют исходный адрес дополнительными транспортными параметрами. Например, ответ с кодом 301 (Moved Permanently) или 302 (Moved Temporarily) может возвратить адрес, совпадающий по местоположению с начальным, но доопределённый транспортными параметрами, указывающими другое имя сервера, или multicast-адрес, которые должны быть использованы в новом запросе, или смену транспортного протокола с UDP на 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 осуществить пересылку запроса по месту назначения в случае, если он совмещён с исходящим прокси-сервером.

Заголовок Contact содержит URI, указывающие на возможное текущее местоположение вызываемого пользователя, причём это могут быть не только SIP-адреса. Заголовок может включать URI для телефона, факса, электронной почты. Значение заголовка Contact может направлять на ресурс, не имеющей связи с запрашиваемым. Например, для SIP вызовов, связанных со шлюзом ТфОП, может оказаться необходимым доставить определённое информационное уведомление такое, как «Номер изменился».

Параметр «expires» значения 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 выставляются равными *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 по перенаправлению.

2.6. Процедуры функционирования HTTP аутентификации

Протокол SIP обеспечивает для аутентификации stateless механизм на основе запроса аутентификации, который базируется на принципах аутентификации протокола HTTP. Всегда, когда прокси-сервер или UA получает запрос (кроме исключений, описанных ниже), он может потребовать от инициатора запроса удостоверить свою подлинность.

Для удостоверения подлинности в протоколе SIP определён только один механизм – аутентификация на базе схемы «Digest». В соответствии с последней рекомендацией для протокола SIP – RFC 3261 схема «Basic» ввиду своей малой 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 чётко определена область аутентификации, т.е. та область, для доступа к которой клиент должен передать отклик аутентификации. Область защиты указывается в поле realm запроса аутентификации, переданного сервером клиенту. Для удобства к значению поля realm предъявляются следующие требования:

- Значение поля realm должно быть уникально в глобальном масштабе. Поле realm должна содержать имя хоста или имя домена.

- Значение realm должно представлять собой удобный для восприятия человеком идентификатор, который может быть отображён пользователю.

Например:

```
INVITE sip:vladimir@protei.ru SIP/2.0
```

```
Authorization: Digest realm="protei.ru", <...>
```

Каждая область аутентификации имеет свой список имён пользователя (usernames) и паролей (passwords). Если сервер не требует аутентификации для определённого 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 своё собственное, зависящее от устройства имя пользователя и пароль для их области.

Когда UAC получает запрос на подтверждение подлинности, он должен отобразить пользователю содержимое поля realm, содержащегося в нём, если программному модулю UAC не известно значение отклика аутентификации для области, указанной в запросе аутентификации.

Несмотря на то, что сервер может запрашивать проведения 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 проведения 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 поступлении 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 является не действительным, или сервер, требующий подтверждения подлинности, не принимает значение отклика по какой-либо другой причине, то сервер может повторить свой запрос или отослать ответ с кодом 403 (Forbidden). UAC не должен повторно пытаться отсылать запрос со значением отклика аутентификации, которое было отклонено.

2.6.1 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 путём отклонения запроса и отсылки ответа с кодом 401 (Unauthorized).

В ответное сообщение 401 (Unauthorized) должен быть включён заголовок WWW-Authenticate. Значение поля состоит из как минимум одного запроса аутентификации, который включает 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="protei.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 отправить запрос, снабдив его надлежащим значением отклика аутентификации. UAC может потребовать от пользователя введения имени пользователя и пароля. Как только будет обеспечено значение отклика аутентификации (или напрямую пользователем, или обнаружено во внутреннем наборе ключей, UA должен занести в кэш-память значение отклика аутентификации, соответствующее адресу в заголовке To и полю realm и впоследствии использования это значение для следующих запросов.

Если для данной области не было найдено значения отклика, UAC может попытаться отослать 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)), может это сделать, добавив в запрос заголовок Authorization.

Значение заголовка Authorization содержит значение отклика аутентификации, включающее информацию аутентификации агента пользователя для области, в которой находится запрашиваемый ресурс, а также параметры, требуемые 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 отсылает запрос, снабжённый значением отклика аутентификации, также он должен увеличить значение в поле заголовка CSeq.

2.6.2 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 путём отклонения запроса и отсылки ответа с кодом 407 (Proxy Authentication Required). В ответное сообщение прокси-сервер должен поместить заголовок Proxy-Authenticate с запросом аутентификации, включающим 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 отправить запрос, снабдив его надлежащим значением отклика аутентификации. Он должен выполнять те же процедуры по отображению пользователю поля realm так же, как указано в параграфе 2.6.1 при получении ответа с кодом 401.

Если значение отклика аутентификации для данной области не найдено, UAC может предпринять ещё одну попытку отправки запроса 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 удаляться 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, отсылаемый размножающим запросы прокси-сервером

агенту пользователя. Очерёдность следования значений этих заголовков не имеет значения.

Прокси-сервер перешлёт 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 отклика аутентификации для каждого запроса аутентификации, запрос не дойдёт 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 отсылки запроса 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.

2.6.3 Схема аутентификации «Digest»

Этот параграф описывает схему аутентификации HTTP Digest, адаптированную для использования в SIP. Использование этой схемы для SIP практически полностью совпадает с применением для протокола HTTP. Правила работы Digest аутентификации определены в RFC 2617 с единственной заменой "HTTP/1.1" на "SIP/2.0" за исключением некоторых различий: URI может иметь как схему SIP, так и SIPS; при формировании значения nonce не может использоваться значение HTTP-заголовка Etag; не возможно применение операций буферизации, специфицированных для HTTP.

Рассмотрим пример проведения процедуры аутентификации «пользователь-пользователь». Вызывающий пользователь Anton желает установить с пользователем Vladimir сеанс связи и отправляет ему запрос INVITE. UAS, конфигурация которого предусматривает обеспечение контроля доступа, не находит в сообщении заголовка Authorization и поэтому отправляет 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 – protei.ru и уникальное значение в поле nonce. UAC формирует комбинацию, включающую имя

пользователя, пароль, полученное значение nonce и другие компоненты, и преобразует её с помощью хэш функции в отклик аутентификации. Значение отклика помещается в заголовок Authorization запроса INVITE и заново отсылается UAS. Сервер примет запрос, самостоятельно подсчитает значение отклика и сравнит его с указанным сообщении. Если значение отклика окажется действительным, то UAS приступит к обработке запроса.

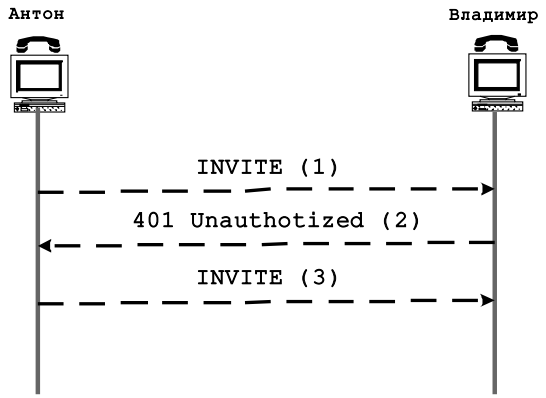


Рис. 2.38 Процедура аутентификации

Рассмотрим состав заголовков, участвующих в процедуре аутентификации более подробно.

В поле заголовка WWW-Authenticate содержится запрос на аутентификацию (challenge). Запрос аутентификации включает наименование схемы (Digest) и компоненты, необходимые для проведения процедуры аутентификации: realm, domain, nonce, opaque, stale, algorithm, qop. При этом необходимым элементами 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**

Заключённый в кавычки список URI, который определяет область защиты. Список сообщает клиенту о группе URI, для которых может быть отослана одинаковая информация аутентификации. Если поле domain отсутствует или существует, но не содержит значения, клиент должен подразумевать, что область защиты состоит только из адресов отвечающего сервера. Поле domain не имеет смысла для заголовка 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 конкретной 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 временная метка (time-stamp), то сервер имеет возможность ограничить время действия значения nonce. Значения nonce не прозрачно для клиента.

- **opaque**

Строка данных, сформированная сервером, которая должна быть возвращена клиентом неизменной в заголовке Authorization последующих запросов, адресованных на URI, находящихся в той же области защиты. Рекомендуется, чтобы строка была в шестнадцатеричном представлении или закодирована с помощью base64.

- **stale**

Флаг, указывающий, что предыдущий запрос клиента был отклонён, поскольку значение nonce было просрочено. Если флаг установлен в состояние «TRUE», клиент может повторить попытку отсылки сообщения без запроса у пользователя нового имени пользователя 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 – 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, данный запрос аутентификации должен быть проигнорирован (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 RFC 2069. Qop – строка, заключённая 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. Отклик аутентификации включает наименование схемы аутентификации (Digest) и компоненты, необходимые для проведения процедуры аутентификации: username, realm, nonce, uri, response, algorithm, cnonce, opaque, qop, nc. При этом необходимым элементами являются username, realm, nonce, uri, response. Значения полей opaque и algorithm должны быть скопированы из заголовка WWW-Authenticate соответствующего ответа. Значение realm, nonce, algorithm, opaque описано выше.

- **username**

Имя пользователя в указанной области аутентификации.

- **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 различным пользователям в зависимости от действий определённого прокси-сервера при продвижении запроса.

- **response**

Строка, состоящая из 32 шестнадцатеричных разрядов, удостоверяющая, что пользователю известен пароль. Формируется с помощью применения функции хэширования к следующим 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 подсчёт отклика аутентификации.

- **cnonce**

Значением поля cnonce является строка, заключённая в кавычки, которая создаётся клиентом и используется клиентом и сервером для предотвращения атак, обеспечения взаимной аутентификации и обеспечения 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 запрос) в шестнадцатеричном выражении, которые были отосланы клиентом с использованием значения nonce, применяемом в данном запросе. Например, в первом запросе отсылаемом на ответ 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 эту 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 требуемое поле отсутствует, передаётся ответ с кодом 400 (Bad Request). Если значение поля response неверно, указание об ошибке должно быть занесено в журнал регистрации, поскольку повторные ошибки такого

плана, произошедшие с одним и тем же клиентом, могут говорить о том, что злоумышленник пытается подобрать пароль.

В целях Digest-аутентификации применяется заголовок Authentication-Info. Заголовок Authentication-Info используется сервером в ответе для того, чтобы сообщить некоторую информацию, касательную процедуры 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 своём следующем отклике аутентификации. По сути поле nextnonce является средством, реализующим выбор между использованием старого значения nonce или нового – изменённого значения. Если клиент пренебрегает требованием сервера и не использует значение из поля nextnonce при создании содержимого поля заголовка Authorization, от сервера может придти запрос на повторную аутентификацию с флагом "stale=TRUE".

- **qop**

Указывает уровень защиты, применённый сервером к ответу. Сервер должен использовать тоже значение, что было послано 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 также обеспечивает определённый уровень защиты целостности 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 также может использоваться 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.

2.7. Защита содержимого тела сообщения средствами S/MIME

SIP-сообщения могут содержать тела сообщения в кодировке MIME, стандарт MIME в свою очередь включает механизмы защиты MIME-содержимого для гарантии целостности и конфиденциальности (включая multipart/signed и application/pkcs7-mime MIME-типы подробно описанные в RFC 1847, RFC 2630, RFC 2633). Однако разработчики должны учитывать, что иногда могут существовать сетевые посредники (но не типичные для протокола SIP прокси-серверы), в обязанности которых входит анализ и модификация тел SIP сообщений (особенно, содержащих описание сессии в формате SDP); поэтому защита MIME-тел может помешать функционированию посредников этого типа, например, некоторым типам межсетевых экранов (firewalls).

В старой рекомендации протокола SIP RFC 2543 для шифрования заголовков и тел сообщений применялся механизм PGP. Этот механизм более не используется.

2.7.1 S/MIME сертификаты

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 определённого получателя. Отправители должны заранее обладать открытыми ключами получателей для шифрования тел сообщения. Открытые ключи могут храниться в UA в наборе ключей.

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 тех, которые используются серверами тем, что информация, удостоверяющая пользователя является не конкретным именем хоста, а адресом пользователя. Этот адрес формируется путём связывания частей SIP или SIPS URI: «userinfo» (информация пользователя), «@» и «domainname» (имя домена) (например, anton@niits.ru); в большинстве случаев он соответствует публичному адресу пользователя.

Каждый агент пользователя, который поддерживает S/MIME, должен иметь в распоряжении набор ключей, содержащий сертификаты конечных пользователей. Этот набор ключей устанавливает соответствие между публичными адресами и соответствующими им сертификатами. Каждый раз, когда пользователи используют в качестве URI инициатора вызова (находящегося в поле заголовка From) один и тот же публичный адрес, они должны использовать один и тот же сертификат.

Любые механизмы, зависящие от наличия сертификатов конечного пользователя, серьёзно ограничены тем, что сегодня не существует единого сертификационного центра, который бы обеспечивал 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 существуют уже установленные 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 по импортированию (вручную или автоматически) найденных в публичных каталогах сертификатов, которые соответствуют адресу места назначения SIP запроса.

2.7.2 Обмен ключами S/MIME

Открытые ключи также могут быть переданы средствами протокола SIP. Средствами S/MIME для этой цели в тело сообщения помещается цифровая подпись, содержащая сертификат, переносящий открытый ключ пользователя, необходимый для верификации подписи.

Когда UAC посылает запрос вне диалога, 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 включает сертификат, UAS первоначально должен проверить достоверность сертификата. Затем UAS обнаруживает информацию, удостоверяющую владельца сертификата, и сравнивает это значение со значением заголовка From запроса. Если сертификат не может быть верифицирован, то UAS должен уведомить своего пользователя о данном 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 публичный адрес владельца сертификата.

Когда UAS на запрос, переданный вне диалога, отсылает ответ, содержащий тело сообщения, подписанное цифровой подписью S/MIME, он также 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 содержащий тело сообщения, подписанное цифровой подписью S/MIME, включающей сертификат, UAC первоначально должен проверить достоверность сертификата. UAC также должен 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 ещё не передал UAS своего собственного сертификата в предыдущих транзакциях, он должен в своё следующее сообщение поместить тело, подписанное цифровой подписью S/MIME, включающей 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 всё же принимает такой сертификат, он должен быть добавлен в набор ключей параллельно предыдущим значениям для этого публичного адреса.

Механизм обмена ключами не гарантирует безопасный обмен ключами 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, публичному адресу, указанному в заголовке To отклонённого запроса); сертификат находится в теле сообщения, для которого значение параметра «smime-type» заголовка Content-Type - *certs-only*. Отсылка 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 защищённое средствами S/MIME тело, для которого значение параметра «handling» в заголовке Content-Disposition - *required*, должен отклонит запрос, отослав 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 удалённое устройство не поддерживает S/MIME. Затем пользователь может заново отослать запрос без использования S/MIME. Однако следует иметь в виду, что ответ с кодом 415 может представлять собой атаку, направленную на понижение уровня безопасности.

Если агент пользователя на запрос, содержащий защищённое средствами S/MIME тело,

получает ответ, с не защищённым телом сообщения, UAC должен уведомить своего пользователя о том, что в ходе данной сессии S/MIME не может быть использован. Подобным образом, если UA, который поддерживает S/MIME, получает запрос с незащищённым телом, он не должен отправлять ответ с защищённым телом и должен уведомить своего пользователя, что в ходе данной сессии S/MIME не может быть использован.

Все описанные условия требуют уведомления пользователя при возникновении аномальных ситуаций, связанных с сертификатами. При этом пользователь должен осознавать, что неожиданная изменения в сертификате или отсутствие защиты, когда она ожидается, являются основанием для предупреждения, но не обязательно говорят о предпринятом вмешательстве злоумышленника. Пользователи могут прервать любую попытку соединения или отказать на полученный запрос установления соединения; на языке телефонии это означает, что они могут повесить трубку и осуществить обратный вызов (call back). Заметим, что пользователи иногда вынуждены изменить их сертификаты, например когда они подозревают, что секретность их закрытого ключа скомпрометирована. Когда их закрытый ключ теряет свойство секретности, пользователи должны создать новый 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 сертификат в теле сообщения, подписанном цифровой подписью, который не соответствует используемому сертификату, UA должен уведомить своего пользователя об этом, чтобы указать, что новый сертификат возможно послан злоумышленником.

2.7.3 Защи та тела сообще ния

В соответствии с S/MIME защищённые 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 защищённая часть тела multipart/signed. Защищённое тело сообщение может быть зашифровано, подписано цифровой подписью, либо цифровая подпись может быть передана в целях 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, обрамлённый звёздочками (*), 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 подписано 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-
```

2.7.4 Защита SIP сообщений средствами S/MIME (SIP-туннелирование)

В целях 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 SIP запросы и ответы 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 в тело сообщение message/sip, он также помещает в ответ тело типа message/sip с инкапсулированным сообщением.

Заметим, что если в сообщении должны быть переданы помимо защищённого незащищённые тела сообщения, то тело типа message/sip может быть отослано в качестве части тела типа multipart/mixed.

2.7.4.1 Обеспечение целостности SIP сообщений

Туннелирование SIP сообщений внутри тела сообщения средствами S/MIME может обеспечить целостность для 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 приёмном конце целостность заголовка должна быть определена путём сопоставления значения заголовка «внутреннего» сообщения со значением аналогичного заголовка во «внешнем» сообщении. Заголовками, которые могут быть допустимо 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 передаче из конца в конец, реализации не должны расценивать это, как нарушение защиты. Изменения в других заголовках сигнализируют о нарушении целостности, пользователи должны быть об этом извещены своим агентом пользователя. 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), агент пользователя должен предупредить пользователя о возможном нарушении защиты.

Если получателем обнаружено нарушение целостности сообщения, сообщение должно быть отклонено и отправлен 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-
```

2.7.4.2 Шифрование при туннелировании

Существует возможность применить шифрование информации, находящейся в теле сообщения типа message/sip, на практике большинство заголовков требуются при прохождении 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 применений 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 публичный адрес инициатора, может быть зашифровано 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 часть тела, содержащее 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, обрамлённый звёздочками (*), 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-
```

2.8 Процедуры обеспечения безопасности

Использование посредников, сложность доверительных отношений, использование протокола SIP между узлами при отсутствии доверительных отношений и работа в режиме пользователь-пользователь сильно усложняет процедуры обеспечения безопасности. Для того, чтобы обеспечить безопасность с учетом всех особенностей SIP, требуются несколько отдельных механизмов, применимых к различным аспектам протокола.

Обеспечение безопасности сигнализации SIP не опирается на безопасность протоколов, используемых совместно с SIP, таких как RTP, и не происходит при участии процедур обеспечения безопасности отдельных тел сообщений, что поддерживается в SIP. Ниже рассматриваются традиционные типы возможных угроз, исходя из которых определяются меры по безопасности для SIP. Затем, в соответствии с несколькими механизмами обеспечения безопасности, детализируется комплекс процедур, требуемых для устранения данных угроз. Далее, перечисляются требования для разработчиков оборудования вместе с типичными примерами реализаций, в которых эти механизмы могут быть использованы для улучшения безопасности SIP.

2.8.1 Типы угроз

В текущем 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 конкретных сервисов обеспечения безопасности, которые потенциально могут предотвратить все категории угроз.

Эти атаки 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 также могут модифицировать пакеты (возможно на скомпрометированном посреднике). Злоумышленники могут заниматься кражей услуг, подслушиванием, разрушать сессии.

2.8.1.1 Злоумышленная регистрация

Механизм регистрации SIP позволяет агенту пользователя идентифицировать себя серверу регистрации, как терминал, который в данный момент задействован пользователем (с присвоенным публичным адресом). Registrar идентифицирует пользователя по информации, указанной в заголовке From сообщения REGISTER, чтобы определить может ли этот запрос изменять контактные адреса, связанные с публичным адресом, приведённом в заголовке To. Несмотря на то, что зачастую эти два поля одинаковы, существуют реализации SIP, когда третья сторона может зарегистрировать контакты от имени пользователя.

Однако, заголовок From SIP-запроса может быть произвольно изменён владельцем UA, и это открывает дверь злонамеренным регистрациям. Злонамеренный пользователь, который успешно имитирует сторону, уполномоченную на изменение контактных адресов, связанных с публичным адресом, например, удаляет из регистрации все существующие контактные адреса и затем регистрирует своё собственное устройство в качестве надлежащего контактного адреса, и тем самым направляя все запросы, адресованные пользователю, на своё устройство.

Эта угроза принадлежит к 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 осуществлять контроль доступа к своим ресурсам путём аутентификации получаемых запросов. Даже UA конечного пользователя, например SIP-телефоны, заинтересованы в верификации инициатора запроса.

Этот тип угрозы демонстрирует необходимость процедур обеспечения безопасности, которые дают возможность устройствам SIP аутентифицировать инициаторов запросов.

2.8.1.2. Имитация сервера

Доменное имя, на которое направляется 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, что злоумышленный пользователь может сымитировать удалённый 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 посылает

запрос в protei.ru, но сервер перенаправления в loniis.ru передаёт фальшивый ответ, позаимствовав SIP-заголовки для ответа у protei.ru. Ложные контактные адреса в ответе перенаправления могут направить инициировавшего запрос UA на несоответствующие или ненадёжные ресурсы или просто не допускать попадание запросов в protei.ru.

Это семейство угроз обширно, а многие из них критичны. В противоположность угрозе злонамеренной регистрации, рассмотрим случай, когда сообщение регистрации, отправленное в домен protei.ru, перехватывает сетевым элементом в loniis.ru, и отправляет на неё ложный ответ с кодом 301 (Moved Permanently). Этот перенаправляющий ответ, который, как кажется, пришёл 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 отсылают запросы.

2.8.1.3. 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 или изменять тела, содержащиеся в запросе.

Рассмотрим UA, который использует тела SIP сообщения для того, чтобы передать ключи шифрования для мультимедийной сессии. Несмотря на то, что он доверяет прокси-серверу домена в плане доставки сигнальной информации, для UA может быть не желательно, чтобы администраторы домена были способны расшифровывать все последующие мультимедийные сессии. Ещё хуже, если прокси-сервер будет предпринимать активные действия, например, менять ключ сессии, перехватывая его, либо менять характеристики безопасности, запрашиваемые инициирующим UA.

Это семейство угроз относится не только к ключам сессии, но и ко всем возможным типам тел сообщения, передаваемых из конца в конец 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 разговоров для подслушивания последующих переговоров.

Также заметим, что для некоторых заголовков в SIP чрезвычайно важна сквозная передача из конца в конец, 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 некоторых определённых случаях – для заголовков. Сервисы обеспечения безопасности, требуемые для тел, включают обеспечение конфиденциальности, целостности и

аутентификации. Эти сквозные сервисы должны быть независимы от средств, используемых для обеспечения безопасности при взаимодействии с посредниками такими, как прокси-сервер.

2.8.1.4. Срыв сессий

После установления диалога в результате обмена инициирующими сообщениями, могут отсылаться последующие запросы, изменяющие состояние диалога и/или сессии. Необходимо, чтобы участники сессии 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 сессии («tag» заголовков To и From и так далее) и затем вставляет в сессию запрос BYE. Злонамеренный пользователь подделает запрос так, чтобы существовала видимость, что он пришёл от второго участника сессии. Как только BYE достигнет места назначения, сессия будет преждевременно разорвана.

Подобные угрозы, возникающие во время сессии, включают передачу ложных запросов re-INVITE, видоизменяющих сессию (возможно для снижения безопасности сессии или для перенаправления медиапотоков в рамках атаки по прослушиванию разговоров).

Наиболее эффективная контрмера для этой угрозы – это аутентификация отправителя запроса BYE. В рассматриваемом случае получатель должен знать только, что BYE пришёл от той же стороны, с которой был установлен соответствующий диалог, что не требует знать полную информацию, идентифицирующую отправителя. Также, если злонамеренный пользователь не в состоянии узнать параметры сессии из-за предпринятых мер по обеспечению конфиденциальности, подделать запрос BYE будет не возможно. Однако некоторым посредникам (таким как прокси-серверы) требуется просматривать параметры установленной сессии.

2.8.1.5 Отказ в обслуживании (DoS-атаки)

Атаки, вызывающие отказ в обслуживании (Denial-of-service attacks), сосредоточены на приведении сетевого элемента в недоступное состояние как правило путём направления чрезмерного количества сетевого трафика на его интерфейсы. Распределённые атаки, вызывающие отказ в обслуживании, позволяют одному пользователю сети оказать влияние на несколько сетевых хостов так, чтобы они «затопили» целевой хост большим количеством сетевого трафика.

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 создаёт возможности проведения распределённых атак, вызывающих отказ в обслуживании, которые должны быть учтены разработчиками и операторами SIP-систем.

Злонамеренные пользователи могут создавать поддельные запросы, содержащие IP-адрес фальсифицированного источника и соответствующий заголовок Via, который

идентифицирует хост, на которого направлена атаки, как инициатора запроса, и затем отсылать этот запрос большому числу сетевых SIP-элементов, тем самым используя SIP агенты пользователя или прокси-серверы для создания чрезмерного количества трафика, направляемого на определённый узел.

Подобным образом злоумышленники могут использовать фальсифицированные значения заголовка Route запроса, которые идентифицируют целевой хост, и затем посылают такие сообщения на прокси-серверы, размножающие запросы, которые увеличивают поток сообщений, направляемых по указанному адресу. Для подобного результата может использоваться заголовок Record-Route, когда злонамеренный пользователь уверен, что инициированный запросом диалог выльется в множественные транзакции, созданные в обратном направлении.

Часть denial-of-service атак становится возможной, если запросы REGISTER не аутентифицируются и не авторизируются должным образом серверами регистрации. Злонамеренные пользователи могут удалить регистрации некоторых или всех пользователей в административном домене, что повлечёт за собой отсутствие возможности пользователей быть приглашёнными к установлению новых сессий. Они также могут зарегистрировать большое число контактных адресов, определяющих один и тот же хост для данного публичного адреса. Это делается для того, чтобы использовать registrar и связанные с ним прокси-серверы, как средства увеличения объёма сетевого трафика при данном типе атак. Также злонамеренные пользователи могут попытаться исчерпать доступный объем памяти и ресурсы диска сервера регистрации путём регистрации огромного количества связок.

Эти проблемы демонстрируют общую необходимость создания сетевых архитектур, минимизирующих риск возникновения denial-of-service угроз.

2.8.2 Механизмы обеспечения безопасности

Исходя из типов угроз, описанных выше, для протокола SIP требуются следующие основные сервисы обеспечения безопасности: сохранение конфиденциальности и целостности при обмене сообщениями, предотвращение атак воспроизведения (replay-attacks) или получения доступа к сообщению путём фальсификации (message spoofing), обеспечение аутентификации и анонимности участников сессии, предотвращение denial-of-service атак. Тела, находящиеся внутри SIP-сообщений отдельно требуют применения сервисов безопасности, обеспечивающих конфиденциальность, целостность и аутентификацию.

Вместо того, чтобы определять новые, специфичные для SIP механизмы безопасности, SIP по возможности использует существующие модели обеспечения безопасности, заимствованные из протоколов HTTP и SMTP.

Полное шифрование сообщений обеспечивает наилучшее средство для сохранения конфиденциальности сигнальной информации, оно также может гарантировать, что сообщения не были модифицированы злонамеренными посредниками. Однако, SIP запросы и ответы 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 для того, чтобы SIP-запросы правильно маршрутизировались.

SIP прокси-серверы также должны 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 посылают запросы.

Логические объекты SIP также требуют взаимной идентификации в безопасной форме. Когда SIP терминал передаёт 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 сообщения SIP обеспечивает альтернативный способ сквозной взаимной аутентификации, а также обеспечивает предельную степень доверия посредникам агентов пользователя.

2.8.2.1. 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 распространённых 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 в качестве дополнения традиционного протокола IP в области обеспечения безопасности. В большинстве случаев IPSec используется в архитектурах, где совокупность 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, который он получает с определённого интерфейса (как в архитектуре VPN). IPSec также может быть 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, находящемся на расстоянии одной пересылки, также хорошо подходят для использования IPSec.

TLS обеспечивает безопасность на транспортном уровне поверх протоколов, ориентированных на соединение (например, TCP); «tls» (означающее TLS по TCP) может быть определён, как желательный транспортный протокол в значение заголовка Via или в SIP URI. TLS лучше всего подходит для архитектур, где необходимо обеспечить безопасность при последовательной передаче между хостами, где отсутствуют доверительные отношения.

Например, UA пользователя Anton доверяет локальному прокси-серверу в своём 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.

TLS должен быть жёстко связан с SIP приложением.

Заметим, что транспортные механизмы для протокола SIP реализуются на основе последовательной передачи через ретрансляционные участки; поэтому UA, который отсылает запросы первому прокси-серверу с использованием TLS, не имеет гарантии, что TLS будет использоваться на протяжении всего пути сообщения.

2.8.2.2. Схема SIPS URI

Схема SIPS позволяет указывать для ресурсов, что они достижимы только при условии выполнении процедур по обеспечению 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 публичный адрес для конкретного пользователя – т.е. быть адресом, который закреплён 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 публичного адреса вызываемого пользователя, схема SIPS предписывает, чтобы весь путь запроса до прокси-сервера, обслуживающего требуемый ресурс, был защищён.

Адреса со схемой SIPS могут быть использованы, помимо поля Request-URI запроса, во многих других случаях, - в публичных адресах, 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).

2.8.2.3 HTTP аутентификация

Протокол SIP обеспечивает возможность посылки запроса подтверждения подлинности, базирующуюся на HTTP-аутентификации. Это выражается в передаче ответов с кодом 401 и 407, включающих заголовки, которые используются для переноса запросов аутентификации, с последующей передачей сообщений-запросов, заголовки которых содержат отклики аутентификации. Использование 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 описывается в разделе 2.6.

2.8.2.4. S/MIME

Как сказано выше, полное сквозное шифрование SIP сообщений в целях конфиденциальности не приемлемо, поскольку сетевые посредники (такие, как прокси-серверы) должны иметь доступ к определённым заголовкам для правильной маршрутизации сообщений; в противном случае SIP сообщения будут немаршрутизируемы.

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 для тел сообщения, равно как и взаимную аутентификацию. Также возможно использовать S/MIME для обеспечения целостности и конфиденциальности SIP заголовков путём туннелирования сообщений SIP. Использование S/MIME в протоколе SIP описано в разделе 2.7.

2.8.3 Реализация механизмов обеспечения безопасности

2.8.3.1 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 также должны иметь возможность принимать сообщения, переданные с использованием TLS. Прокси-серверы, серверы перенаправления и регистрации должны располагать сертификатом, 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 сертификатов 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.

2.8.3.2 Решения по обеспечению безопасности

Совместное функционирование механизмов обеспечения безопасности может в определённой степени придерживаться существующих Web и E-Mail моделей безопасности. На верхнем уровне агенты пользователя аутентифицируют себя серверам (таким, как прокси-сервер, серверам перенаправления и серверам 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.

Регистрация

При регистрации UA в своём локальном административном домене, он должен установить TLS соединение со своим сервером регистрации. Registrar передаёт сертификат агенту пользователя. Узел, определяемый сертификатом должен находится в пределах домена, в котором UA намеревается зарегистрироваться. Например, если UA намеревается выполнить регистрацию для публичного адреса «anton@niits.ru», сертификат узла должен идентифицировать хост, находящийся в домене niits.ru (например 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 неподходящий узел, UA не должен отправлять сообщение REGISTER и следовательно продолжать процесс регистрации.

Когда registrar выдаёт 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 создаёт запрос REGISTER, содержащий в поле Request-URI адрес, соответствующий сертификату узла, полученному от сервера регистрации. После того, как UA отослал запрос REGISTER по существующему TLS соединению, registrar должен запросить подтверждение подлинности запроса, отослав ответ с кодом 401 (Proxy Authentication Required). Значение поля realm в заголовке Proxy-Authenticate ответа должно соответствовать домену, определённому из сертификата узла. Когда UAC получает запрос аутентификации, он должен либо обратиться к пользователю за значением отклика аутентификации, либо выбрать значение отклика в наборе ключей, 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 передаётся серверу регистрации.

Так как registrar требует от агента пользователя аутентифицировать себя, для злоумышленника будет представлять трудность подделать запросы REGISTER для конкретного публичного адреса пользователя. Также, поскольку сообщение REGISTER

посылается по конфиденциальному TLS соединению, у злонамеренных пользователей не будет возможности перехватить запрос REGISTER, чтобы записать значение отклика аутентификации для осуществления атак воспроизведения (replay-атак).

Далее, после того, как registrar осуществит регистрацию, UA должен оставить это соединение открытым при условии, что registrar также функционирует как прокси-сервер, которому посылаются запросы, предназначенные для пользователей, находящимся 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 – злоумышленники не могут создать поддельные запросы, которые могли бы быть отосланы через этот прокси-сервер.

Междоменные запросы

Теперь предположим, что агент пользователя Anton желает инициировать сессию с пользователем в удалённом административном домене, имеющим публичный адрес vladimir@protei.ru. При этом локальный административный домен вызывающего пользователя (niits.ru) имеет локальный исходящий прокси-сервер.

Прокси-сервер, выполняющий приём входящих запросов для административного домена, также может работать, как исходящий прокси-сервер; это допускается для niits.ru с целью обеспечения 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 тоже TLS соединение с локальным прокси-сервером при отсылке запроса INVITE другому пользователю. UA использует отклик аутентификации, сохранённый в кэш-памяти, для сообщения INVITE, 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 маршрутизироваться. 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 возвращает 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 удалённый домен. Следовательно, локальный исходящий прокси-сервер в niits.ru должен установить TLS соединение с удалённым 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 удалённой 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 пересылке 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 жёсткую политику безопасности, которая требует отклонять запросы, которые не соответствуют административному домену, откуда они были пересланы. Такая политика безопасности может применяться, например, для предотвращения получения спама.

Однако, такая политика гарантирует только то, что запрос пришёл от домена, указанного в нём; это не позволяет домену protei.ru узнать, как niits.ru аутентифицировал пользователя Anton. Только если protei.ru будет с помощью других средств иметь доступ к информации аутентификации домена niits.ru, он возможно сможет узнать, как Anton себя идентифицировал. В определённых случаях в домене protei.ru может быть установлена ещё более жёсткая политика, которая запрещает приём запросов, пришедших 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 существует таковой, связанный с пользователем, адресуемым в запросе (в данном случае vladimir@protei.ru). INVITE должен быть переслан по этому каналу пользователю Vladimir. Поскольку запрос приходит по TLS соединению, которое было ранее аутентифицировано, как соединение к 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 сервисы безопасности на протяжении времени жизни всего диалога. Если прокси-серверы не добавляют свои значения заголовка Record-Route, будущие сообщения будут передаваться из конца в конец - непосредственно между Anton и Vladimir без применения каких-либо сервисов обеспечения безопасности (если действующие стороны не договорились об использовании других средств обеспечения сквозной безопасности, таких как S/MIME).

Злоумышленник, столкнувшись с такой архитектурой, например, не сможет подделать запрос BYE и поместить его в поток сигнальной информации между Vladimir и Anton, поскольку не имеет возможности узнать параметры сессии, а также потому, что механизмы обеспечения целостности защищают трафик между Anton и Vladimir.

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 отправить запрос INVITE по адресу vladimir@protei.ru, его UA должен инициировать TLS соединение непосредственно с прокси-сервером protei.ru (используя механизмы, описанные в "SIP: Locating SIP Servers", RFC 3263). Когда UA пользователя Alexander получает сертификат от прокси-сервера домена protei.ru, он должен быть проверен перед тем, 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 2.7.4) в сообщении INVITE. Прокси-сервер домена protei.ru также может иметь жёсткую политику, касающуюся запросов, которые не содержат protei.ru в доменной части адреса в заголовке From. На такие запросы прокси-сервер может даже не отправлять запросы аутентификации – он расценивает этих пользователей как неаутентифицированных.

В отношении пользователя Vladimir прокси-сервер в домене protei.ru имеет следующую политику: все неаутентифицированные запросы перенаправляются на контактный адрес <sip:vladimir@192.0.2.4>, зарегистрированный для публичного адреса vladimir@protei.ru. Alexander получает ответ перенаправления вызова по TLS соединению, которое он установила с прокси-сервером домена protei.ru; поэтому он доверяет достоверности контактного адреса.

Затем Alexander должен установить TCP соединение по назначенному адресу и послать новое сообщение INVITE с полем Request-URI, содержащим полученный контактный адрес (пересчитав подпись, заверяющую тело сообщения). Vladimir получает этот INVITE на интерфейс, обозначенный как «небезопасный», но его UA анализирует и, в данном случае, опознаёт содержимое заголовка From запроса и затем сопоставляет локально буферизированный сертификат с сертификатом, присутствующим в подписи тела сообщения INVITE. Vladimir создаёт ответное сообщение, аутентифицируя себя подобным образом, и отправляет его пользователю Alexander; после этого безопасный диалог вступает в силу.

Иногда межсетевые экран или NAT (Network Address Translation) в административном домене могут препятствовать установлению прямого TCP соединения с UA. В этих случаях прокси-серверы также могут пересылать запросы агентам пользователя, но уже без вовлечения доверительных отношений, например, путём отказа от существующего TLS соединения и пересылки запроса по не защищенному TCP-соединению, как предписывает внутренняя политика.

Защита от отказа в обслуживании (DoS-атак)

Для минимизации риска атак, направленных на отказ в обслуживании сервера, требуется выполнение следующих правил.

Когда узел, на котором функционирует SIP прокси-сервер, 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).

Также возможно 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.

В не зависимости от того, какие решения обеспечения безопасности реализуются, чрезмерные потоки сообщений, направленные на прокси-серверы, могут привести к закрытию ресурсов прокси-сервера; соответственно это предотвратит достижение трафика своего места 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 флудингу, чем stateless прокси-серверы.

Агенты пользователя и прокси-серверы должны запрашивать аутентификацию отправителей запросов путём 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 неаутентифицированным 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 заключении подчеркнём, что взаимная аутентификация прокси-серверов с помощью таких механизмов как TLS значительно понижает потенциальные возможности злонамеренного пользователя по введению поддельных запросов или ответов, которые могут привести к отказу из-за перегрузки.

2.9 Алгоритмы установления соединения

2.9.1 Установление соединения с участием прокси-сервера

В этом сценарии Anton вызывает пользователя Vladimir с использованием двух прокси-серверов Proxy 1 и Proxy 2. Первоначальный запрос INVITE (F1) включает предустановленный маршрут в заголовке Route с адресом Proxy 1. Proxy 1 сконфигурирован как исходящий прокси-сервер для Anton. Запрос не содержит информации аутентификации, поэтому Proxy 1 возвращает ответ с кодом 407 (Proxy Authorization), содержащий запрос подтверждения подлинности.

Затем отсылается новый запрос INVITE (F4), содержащий надлежащий отклик аутентификации и происходит установление соединения. Вызов разрушается, когда Vladimir отсоединяется и посылает сообщение BYE.

Прокси-сервер Proxy 1 вставляет заголовок Record-Route в сообщение INVITE, 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.

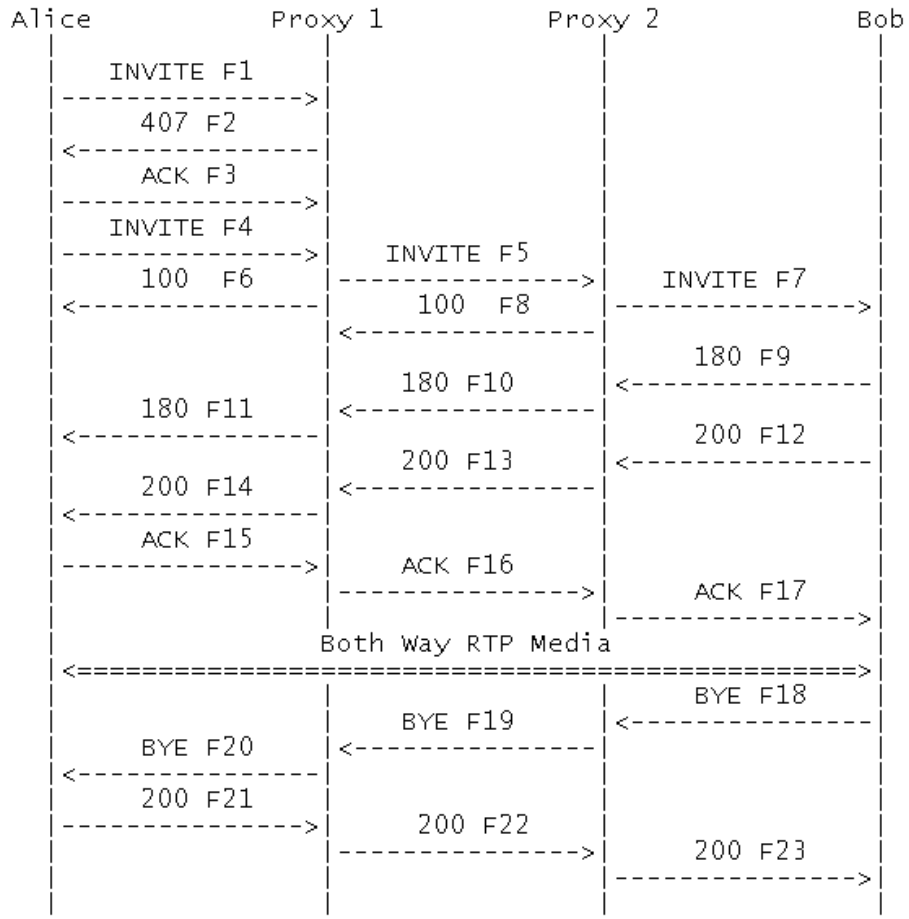


Рис. 2.39 Установление соединения с участием прокси-серверов

Содержание сообщений

F1 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
```

Прокси-сервер Proxy 1 запрашивает аутентификацию.

F2 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
```

F3 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
```

Anton предпринял новую попытку отсылки запроса INVITE, содержащего отклик аутентификации.

F4 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 пользовательскую информацию на порт 49172 из сети.

F5 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
```

F6 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
```

F7 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
```

F8 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
```

F9 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
```

F10 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
```

F11 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
```

F12 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
```

F13 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
```

F14 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
```

F15 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
```

F16 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
```

F17 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 вешает трубку. Заметим, что значение CSeq не равно 3. Терминалы пользователей Anton и Vladimir поддерживают свой собственный раздельный порядок счёта CSeq.

F18 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
```

F19 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
```

F20 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
```

F21 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
```

F22 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
```

F23 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
```

2.9.2 Установление соединения с участием сервера перенаправления

В этом сценарии Anton осуществляет вызов пользователя Vladimir с помощью сервера перенаправления. Первоначально сообщение INVITE отсылается серверу перенаправления. Он возвращает ответ с кодом 302 (Moved Temporarily), содержащий заголовок Contact с текущим SIP-адресом пользователя Vladimir. Затем Anton создаёт новый запрос INVITE и отсылает его пользователю Vladimir через прокси-сервер и 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 отправляет сообщение BYE.

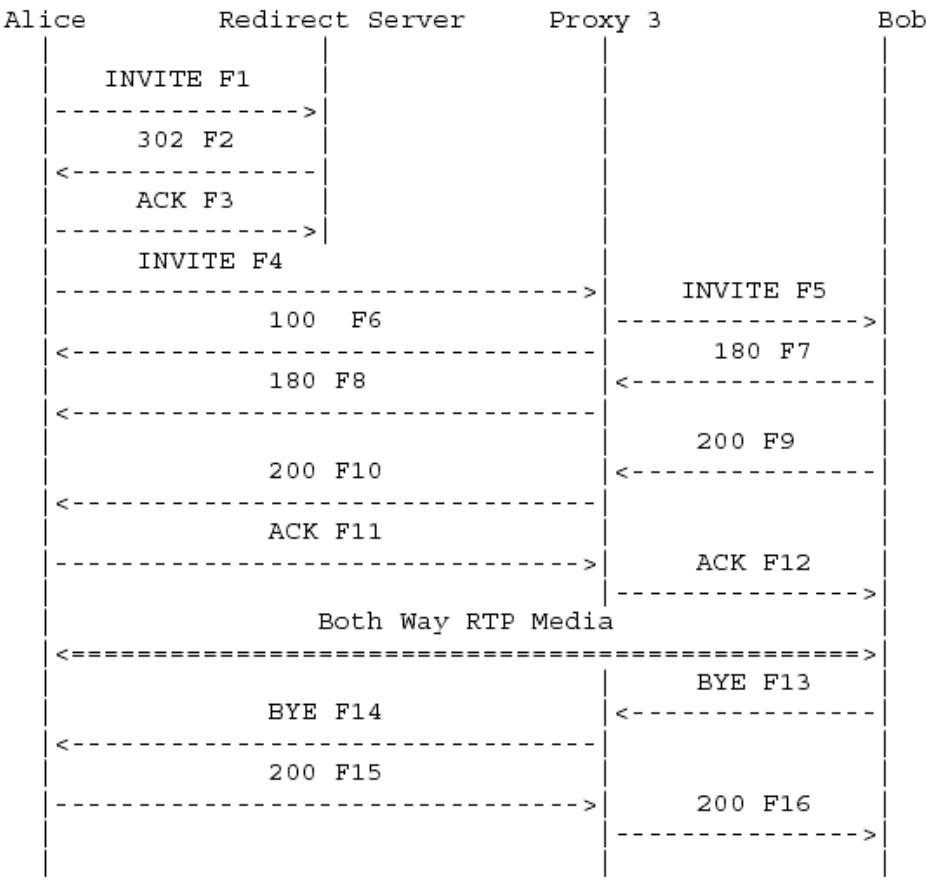


Рис. 2.40 Установление соединения с участием сервера перенаправления

Содержание сообщений

F1 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
```

F2 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
```

F3 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
```

F4 INVITE Anton -> Proxy 3

```
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
```

F5 INVITE Proxy 3 -> Vladimir

```
INVITE 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=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
Record-Route:
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
```

F6 100 (Trying) Proxy 3 -> 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
```

F7 180 (Ringing) Vladimir -> Proxy 3

```
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
```

F8 180 (Ringing) Proxy 3 -> 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
```

F9 200 (OK) Vladimir -> Proxy 3

```
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
```

F10 200 (OK) Proxy -> 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 сессии агента пользователя Anton, так как оно отсутствует в запросе INVITE.

F11 ACK Anton -> Proxy 3

```
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
```

F12 ACK Proxy 3 -> 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 вешает трубку.

F13 BYE Vladimir -> Proxy 3

```
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
```

F14 BYE Proxy 3 -> 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
```

F15 200 (OK) Anton -> Proxy 3

```
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
```

F16 200 (OK) Proxy 3 -> 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
```

2.10. Транспортный уровень протокола SIP

Транспортный уровень протокола SIP отвечает за фактический перенос запросов и ответов через сеть с использованием её транспортных протоколов. Кроме того в случае применения ориентированных на соединение транспортных протоколов транспортный уровень SIP осуществляет выбор используемого соединения 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 открытии соединения оно контролируется транспортными функциями и клиента и сервера. Соединения идентифицируются указателем, состоящим из адреса, порта и транспортного протокола на удалённом конце соединения. Когда соединение открывается транспортным уровнем SIP, этот указатель принимает значение IP-адреса, номера порта и транспортного протокола места назначения. Когда соединение принимается транспортным уровнем SIP, указатель принимает значение IP-адреса, номера порта и транспортного протокола источника.
Заметим, номер порта может являться временным (и не существует возможности узнать, является он временным или выбран с помощью DNS-процедур, описанных в «Locating SIP servers», RFC 3263), 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 каждом направлении.
Соединения должны оставаться открытыми некоторый интервал времени, определённый конкретной реализацией, после того как последнее сообщение было отправлено или получено через это соединение. Интервал времени должен как минимум быть равным наибольшему времени, требующемуся элементу для того, чтобы перевести транзакцию в завершённое состояние («Terminated»). Это нужно для того, чтобы сделать возможным выполнение транзакционных функций на том же соединении, на 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 большую величину для таймера C (см. пункт 11 параграфа 2.4.1.4.)
Все узлы SIP должны уметь работать с протоколами UDP и TCP. Могут применяться также и другие протоколы.

2.10.1 Работа клиента

2.10.1.1 Отсылка запросов

Клиентская сторона транспортного уровня SIP отвечает за передачу запросов и приём ответов. Пользователь транспортного уровня SIP, которым может являться либо транзакция, либо ядро SIP-приложения, передаёт клиентской стороне транспортного уровня SIP запрос, IP-адрес, номер порта, название транспортного протокола и возможно TTL при многоадресной рассылке.

Если запрос находится в пределах 200 байт path MTU(максимальная единица передачи на пути) или если он больше 1300 байт, а path MTU не известна, запрос должен передаваться с использованием транспортного протокола, осуществляющего 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 предотвращает возможную фрагментацию сообщений, передающихся по протоколу UDP, и обеспечивает 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 байт включая IP и UDP заголовки.

Существование 200-байтного «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 байт выбирается, когда path MTU не известна, исходя из допущения, что Ethernet MTU составляет 1500 байт.

Иногда элемент сети SIP вынужден послать запрос по протоколу TCP из-за большого размера сообщения (однако при этом существует возможность отправить сообщение и по UDP). Тогда если в результате попытки установления TCP-соединения произошёл сбой или выяснилось, что протокол ICMP не поддерживается, элемент сети SIP должен совершить повторную попытку отправки запроса, 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 последующих 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 отсылается запрос, клиентская сторона транспортного уровня SIP должна поместить значение, состоящее из IP-адреса/имени хоста и номера порта, в заголовок Via. Если номер порта отсутствует, значение по умолчанию для него выставляется в зависимости от используемого транспортного протокола. Оно равно 5060 для UDP, TCP и SCTP, и 5061 для TLS.

При применении надёжных транспортных протоколов ответ отсылается на соединение, по которому был получен запрос. Поэтому клиентская сторона транспортного уровня SIP должна быть готова получить ответ на тоже соединение, которое использовалось в процессе

отсылки запроса. В случае возникновения ошибки сервер может попытаться открыть новое соединение для отсылки ответа. Чтобы предусмотреть этот случай, транспортный уровень SIP также должен быть готов принять входящее соединение на IP-адрес источника, с которого был отослан запрос, и номер порта, указанный в значении заголовка Via. Транспортный уровень SIP должен также быть готов получить входящее соединение на любой адрес и порт, которые будут выбраны сервером с использование процедур, описанных в Section 5, «Locating SIP servers», RFC 3263.

При использовании ненадёжных транспортных протоколов клиентская сторона транспортного уровня должна быть готова принять ответы на IP-адрес источника, с которого отослан запрос (поскольку ответы отсылаются обратно на адрес ресурса) и номер порта, указанный в значении заголовка Via. Более того, также как и для надёжных транспортных протоколов, в определённых случаях ответ может быть отослан по другому адресу. Клиент должен быть готов принять ответы на любой адрес и порт, которые будут определены сервером при использовании процедур, описанных в Section 5, «Locating SIP servers», RFC 3263.

При осуществлении многоадресной рассылки клиентская сторона транспортного уровня SIP должна быть готова получить ответы на ту же multicast-группу и порт, на которые посылается запрос (то есть для того, чтобы получить ответ, требуется быть членом той multicast-группы, которой был отослан запрос).

Если запрос предназначен для отсылки на IP-адрес, порт и транспортный протокол, для которых уже есть открытое соединение, он отсылается по этому соединению, однако не исключается возможность создания другого соединения.

Если при отсылке запроса применяется многоадресная рассылка, он отсылается с использованием группового адреса, номера порта и TTL, предоставленных пользователем транспортного уровня SIP. Если запрос отсылается с использованием ненадёжного транспортного протокола, он передаётся на IP-адрес и порт, полученные от пользователя транспортного уровня.

Получение ответов

Когда получен ответ, клиентская сторона транспортного уровня SIP проверяет верхнее значение заголовка Via. Если в значении адрес и порт не соответствует значению, на которое была сконфигурирована клиентская сторона транспортного уровня, по умолчанию ответ отклоняется.

Если существуют действующие клиентские транзакции, то клиентская сторона транспортного уровня SIP использует процедуры, описанные в параграфе 2.3.2.1, чтобы сопоставить пришедший ответ с существующей транзакции. Если обнаруживается соответствие, ответ должен быть передан этой транзакции. В противном случае ответ должен быть передан ядру (в не зависимости от того, будет ли это stateless прокси-сервер, stateful прокси-сервер или UA) для дальнейшей обработки. Обработка ответов, не соответствующих ни одной из транзакций, зависит от типа ядра (например, ядро прокси-сервера перешлёт их, в то время, как ядро UA их отклонит).

2.10.2 Работа сервера

Получение запросов

Сервер должен быть готов получать запросы на любую комбинацию IP-адреса, порта и транспортного протокола, на которую клиент может отправить запрос после осуществления 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 также должен ожидать поступления TCP пакетов. Это происходит потому, что если сообщение слишком длинное, оно может требовать оправки по протоколу TCP, а не UDP. Обратное не верно. Серверу нет смысла ожидать поступления UDP пакетов на определённый адрес и порт, если он ожидает поступления на этот адрес и порт TCP пакетов. Однако могут возникнуть определённые причины, когда при этом сервер будет ожидать UDP пакеты.

Когда серверная сторона транспортного уровня SIP получает запрос, пришедший по любому транспортному протоколу, он должен проанализировать адрес и порт в верхнем значении заголовка Via. Если часть URI, отражающая имя узла, содержит доменное имя или содержит IP-адрес, который отличается от адреса источника пакетов, сервер должен добавить параметр «received» в данное значение заголовка Via. Этот параметр должен содержать адрес источника, от которого был получен пакет. Параметр будет использован серверной стороной транспортного уровня SIP при отсылке ответа, который должен быть передан на IP-адрес источника, с 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 с 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 серверной транзакции. Для этого используются процедуры, описанные в параграфе 2.3.2.2. Если соответствующая серверная транзакция найдена, запрос передаётся этой транзакции для обработки. Если таковой не найдено, запрос передаётся ядру, которое может принять решение о формировании новой серверной транзакции для данного запроса. Заметим, что когда ядро UAS отсылает 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.

Отсылка 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 надёжный транспортный протокол такой, как TCP, SCTP или TLS (по TCP или SCTP), ответ должен быть отослан по существующему соединению источнику оригинального запроса, который создал транзакцию, если это соединение всё ещё открыто. Для этого серверная сторона транспортного уровня SIP должна удерживать связь между серверными транзакциями и транспортными соединениями. Если соединение закрылось, сервер должен открыть 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 не определено в значении заголовка Via). Если попытка открытия соединения терпит неудачу, сервер должен найти адрес и номер порта с помощью выполнения процедур, специфицированных в RFC 3263 «SIP: Locating SIP servers», чтобы открыть соединение и отправить по нему ответ.

- В случае, когда значение заголовка Via содержит параметр «maddr», ответ должен быть переслан на адрес, указанный в нём, с использованием порта, обозначенного в значении заголовка или порта 5060, в случае его отсутствия. Если адрес является адресом многоадресной рассылки, ответ отсылается 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 ненадёжных unicast транспортных протоколов), ответ должен быть отослан на адрес, указанный в данном параметре с использованием порта, определённого в значении заголовка, или порта 5060, если он отсутствует. Если это не удаётся, например, приводит к получению ответа протокола ICMP – «port unreachable» (порт недоступен), должны быть применены процедуры, описанные в Section 5, «Locating SIP servers», RFC 3263, для того, чтобы определить, куда отослать ответ.

- В противном случае ответ должен быть отослан на адрес, указанный в значении заголовка, с использованием процедур, описанных в Section 5, «Locating SIP servers», RFC 3263.

.

2.10.3 Длина тела сообщения

В случае использования 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, помещённые 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 создаёт ответ с кодом 400 (Bad Request).
В случае потоко-ориентированных протоколов, таких как TCP, заголовок Content-Length, указывающий размер тела, используется всегда.

2.10.4. Обработка ошибок

Обработка ошибок не зависит от того, чем является сообщение – запросом или ответом. Если пользователь транспортного уровня SIP, например, уровень транзакций или ядро запрашивает, чтобы сообщение было отослано по ненадёжному транспортному протоколу, и в результате происходит ICMP-ошибка, поведение 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 проинформировать пользователя транспортного уровня о неудаче при отсылке. Ошибки протокола ICMP «Source quench» и «TTL exceeded» должны игнорироваться.

Если пользователь транспортного уровня SIP запрашивает, чтобы сообщение было отослано по надёжному транспортному протоколу, и в результате происходит ошибка соединения, транспортный уровень SIP должен проинформировать пользователя транспортного уровня об ошибке.

3. Протокол SIP для телефонии (SIP-T)

3.1. Назначение и особенности протокола SIP-T

В предыдущей главе был подробно описан один из ключевых протоколов IP-телефонии, протокол SIP. Но сегодня сети SIP не могут существовать изолированно от традиционных телефонных сетей, поэтому необходимо иметь легкий для понимания и простой в управлении интерфейс взаимодействия между ними. Для решения этой задачи было разработано расширение спецификаций протокола SIP – SIP–T.

Протокол SIP–T (SIP для телефонии) предназначен для прозрачной передачи сигнализации ТфОП (например, ОКС №7 или DSS) по сети SIP. SIP–T использует два механизма переноса сигнализации, известных как «инкапсуляция» и «трансляция». В шлюзах SIP–ISUP, сообщение ISUP ОКС №7 инкапсулируется в сообщение протокола SIP, при этом сохраняется только информация, необходимая для обслуживания. Однако, некоторые посредники при передаче, например, прокси-серверы, принимающие решения о продвижении запроса SIP дальше по сети, не могут правильно распознать ISUP. Наряду с инкапсуляцией, важная информация транслируется из сообщения ISUP в заголовоки сообщений SIP, в порядке определяемым тем как дальше будет маршрутизироваться SIP запрос.

Процедуры, необходимые при взаимодействи сети SIP и ТфОП приведены в таблице:

Таблица 3.1

Требования при 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, содержаейся в сообщениях ISUP	Трансляция параметров сообщений ISUP в заголовки запросов SIP
Передача сигнальной информации ISUP во время мультимедийной сессии	Использование запроса INFO

В традиционной телефонной сети используется множество различных протоколов, но в данном справочнике рассматривается взаимодействие только SIP и ISUP (прикладной уровень стека протоколов ОКС №7). ISUP взят за основу как самый используемый в настоящее время.

3.2 Сценарии организации взаимодействия

Далее описаны несколько различных сценариев, имеющих место на реальных сетях связи.

3.2.1 Применение SIP-T при транзите трафика (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 с тело SIP запроса.

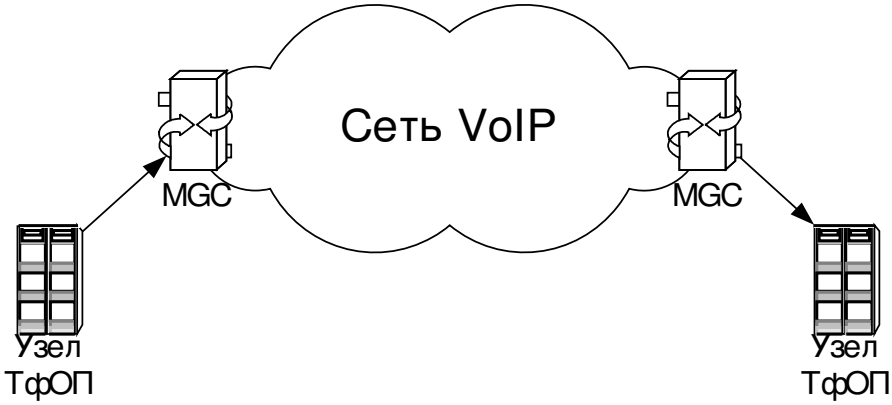


Рисунок 3.1 Сеть в случае транзита трафика ТфОП через сеть VoIP.

На рисунке 3.1 показана схема установления соединения для данного сценария. Когда звонок, предназначенный для сети SIP, идет от абонента ТфОП, то сообщение ISUP будет получено шлюзом сигнализации в MGC (Media Gateway Controller), который является точкой взаимодействия ТфОП и сети SIP. MGC стандартными средствами протокола SIP посылает запрос в сеть. Протокол SIP осуществляет стандартную маршрутизацию внутри сети, чтобы определить нужную точку выхода (в нашем случае также MGC) для вызова. Найдя ее, он начинает диалог установления медиа–соединения между начальной и конечной точками. Оконечный MGC, который является точкой выхода из сети SIP в ТфОП, передает сообщение ISUP в сеть ТфОП, используя любое вложенное сообщение ISUP, пришедшее в запросе SIP, не проверяя его (т.е. просто транслирует сообщение в сеть ТфОП, не анализируя содержимое).

Пример простого соединения для данного сценария приведен на рисунке 3.2

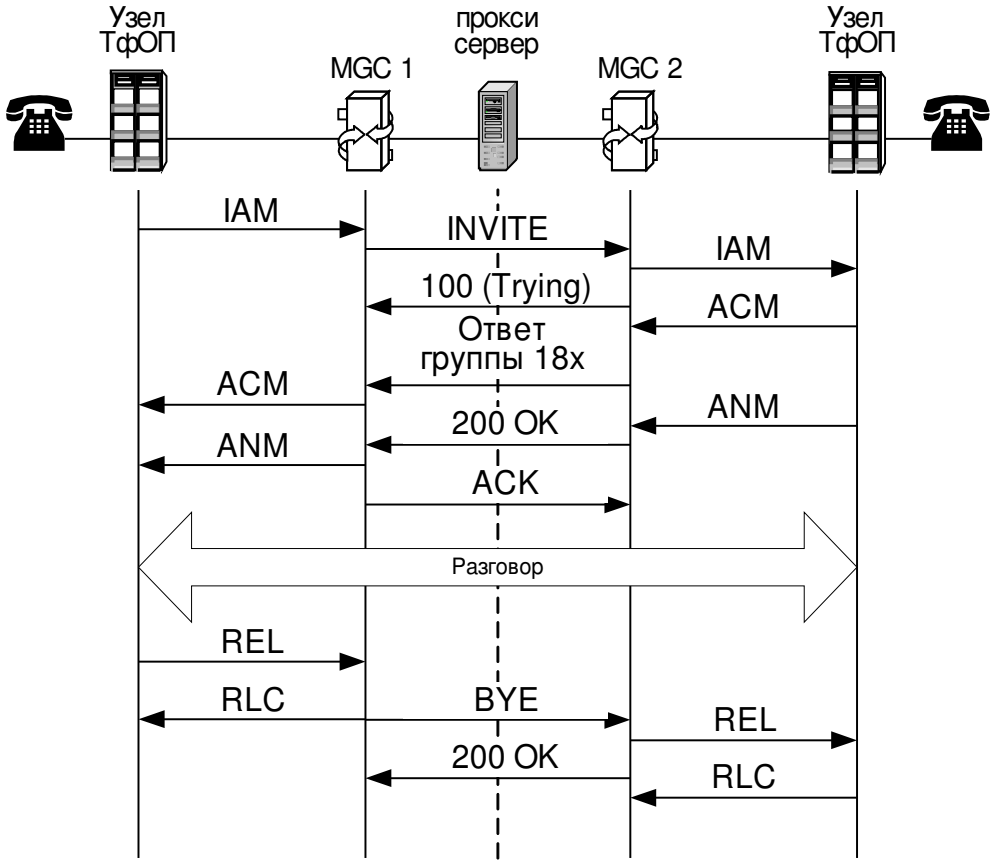


Рисунок 3.2 Пример обмена сообщениями между узлами ТфОП при транзите трафика через сеть VoIP.

3.2.2 Процедуры организация связи из ТфОП в IP-сеть

В данном сценарии начальной точкой является телефон ТфОП, а конечной точкой – терминальное оборудование пользователя сети SIP.

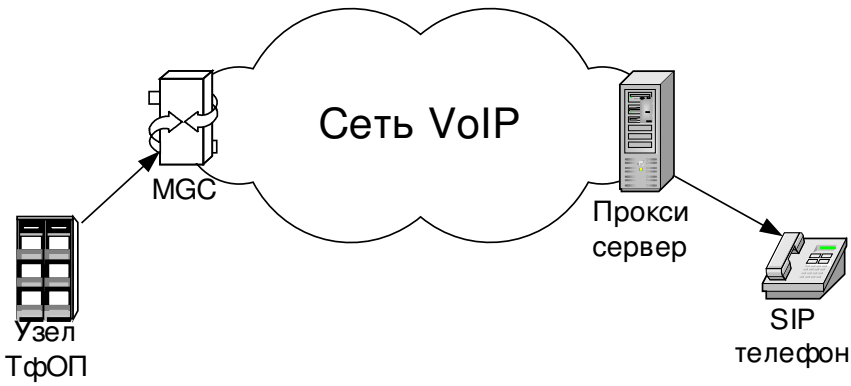


Рисунок 3.3 Сеть в случае установления соединения ТфОП – VoIP.

На рисунке 3.3 приведена схема организации взаимодействия для данного сценария.

Здесь требуется только один MGC, который при получении сообщения ISUP из ТфОП транслирует его в SIP – запрос, который посылается в сеть. Этот запрос обрабатывается в сети SIP стандартным образом и в результате устанавливается медиа–соединение через MGC и прокси-сервер, на телефон в сети SIP.

Порядок обмена сообщениями в данном сценарии приведен на рисунке 3.4.

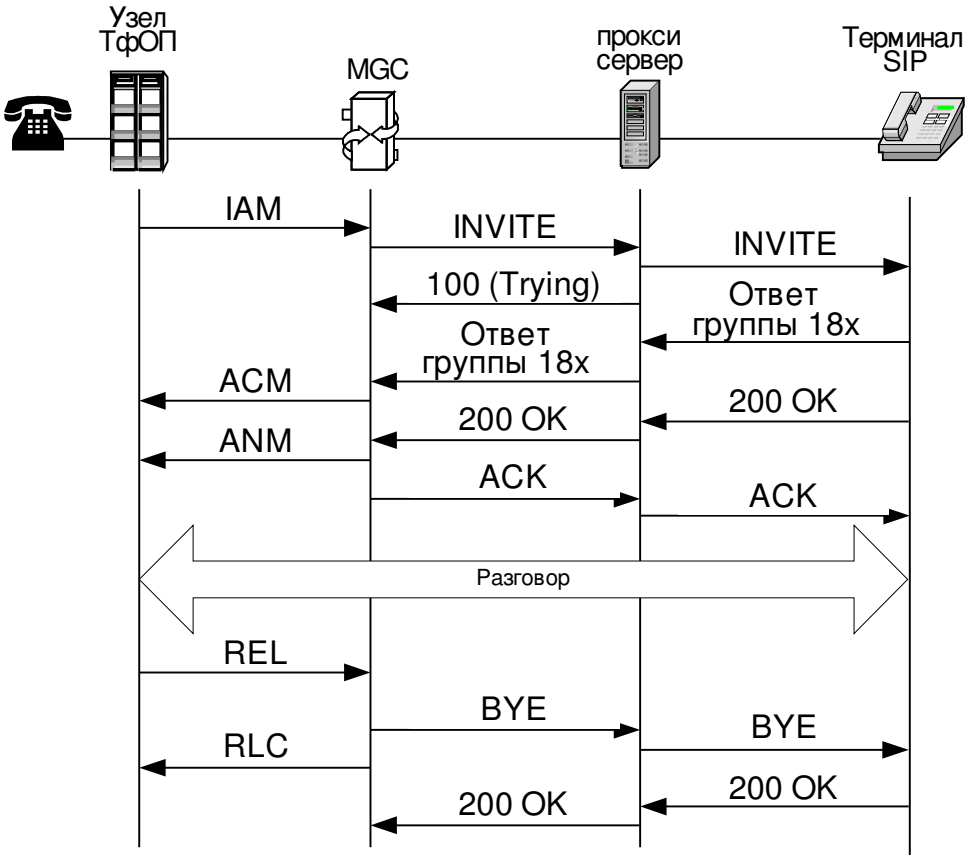


Рисунок 3.4 Пример обмена сообщениями при организации вызова из ТфОП в сеть VoIP.

3.2.3 Процедуры организация связи из IP-сети в ТфОП

В данном сценарии начальной точкой является терминал сети SIP, а конечной точкой является телефон ТфОП.

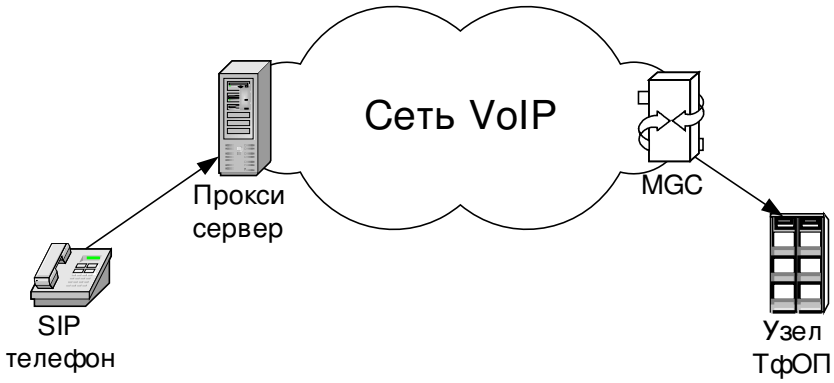


Рисунок 3.5 Сеть в случае установлеия соединения VoIP – ТфОП.

Схема организации взаимодействия для данного сценария показана на рисунке 3.5. В отличие от предыдущих двух случаев, где сообщение ISUP инкапсулировалось в тело запроса SIP и часть этого сообщения транслировалась в заголовок этого запроса, в данном сценарии MGC, являющийся точкой выхода из сети SIP, только транслирует заголовок запроса SIP в соответствующие параметры сообщения ISUP.

Порядок организации взаимодействия в данном сценарии показан на рисунке 3.6.

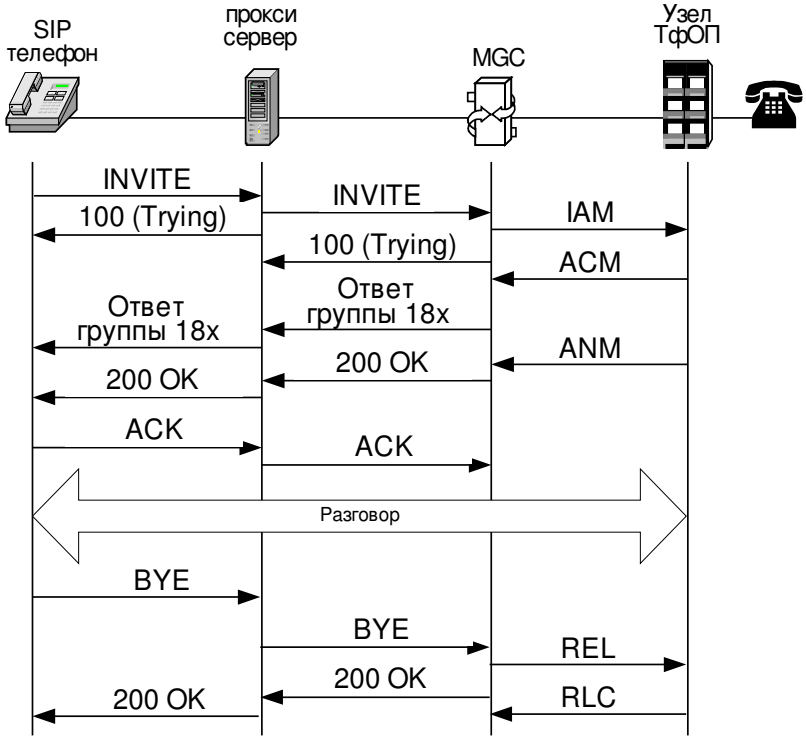


Рисунок 3.6 Пример обмена сообщениями при организации вызова из сети VoIP в ТфОП.

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 ISUP и посылает в ТфОП запрос IAM.

Пока сигнальная информация анализируется и доставляется до нужного абонента ТфОП, пользователю сети SIP посылается ответ с кодом 100 (Trying).

Как только из сети ТфОП приходит сообщение о приеме всего адреса (ACM)

вызывающему пользователю выдается ответ 180 (Ringing) или 183 (Session Progress).

Далее абонент поднимает трубку, и из ТфОП приходит сообщение ANM на MGC. MGC посылает в сеть ответ с кодом 200 (OK). После этого на MGC приходит окончательное подтверждение ACK.

3.3 Компоненты протокола SIP–T

3.3.1 Использование протокола SIP

SIP–T использует методы и процедуры протокола SIP, описанные в главе 2 данного справочника и [RFC 3261].

3.3.2 Процедуры инкапсуляции сигнальных сообщений

Возможность инкапсуляции сигнализации ТфОП является одним из основных требований к SIP–T. 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 вариантов ISUP и поэтому для определения используемого варианта введен специальный MIME тип - ISUP Media Type, позволяющий удобно получать информацию об используемом варианте ISUP.

ISUP Media Type содержит следующую информацию:

Таблица 3.2

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 послать пользователю сообщение о том, что данный тип ISUP не поддерживается. Данная спецификация не ограничивает значения, которые могут быть использованы в «version»; это оставлено на усмотрение системным администраторам.

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 или ANSI.

Параметр «base» может включатся опционально в некоторые сообщения, если обязательно требуется, чтобы получатель правильно распознал используемый тип ISUP, т.к.

параметр «version» может быть не понят. Таблица 3.3 представляет собой возможные значения параметра «base», поддерживаемых телом сообщения типа application/ISUP.

Таблица 3.3

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-)

Заголовок Content-Disposition может служить для описания процесса обработки вложенного сообщения ISUP, в частности 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.

Полное описание заголовка Content-Disposition находится в главе 2 и [RFC 2046]. Для вложенных ISUP сообщений заголовок Content-Disposition может иметь следующие параметры и значения:

На сегодняшний день существует значение заголовка – *signal*, параметр – «handling», допустимые значения параметра – *optional* и *required* (см. Главу 2, раздел 2.2.2).

Далее представлен 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 протокола SDP и инкапсулированное сообщение ISUP IAM.

Примечание. Части сообщения разделяются специальной строкой, задаваемой параметром boundary (см. [RFC 2046]). В данном 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 9@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

3.3.3 Процедуры трансляции сигнальных сообщений

Трансляция применяется при преобразовании сигнальной информации между протоколами ISUP и SIP. По существу трансляция включает в себя два компонента:

1. 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 между сообщениями ISUP и SIP, каждое сообщение ISUP должно быть транслировано в конкретное сообщение SIP. Например, IAM в INVITE, REL в BYE и т.д.

2. Преобразование параметров сообщения ISUP в заголовок SIP сообщения: Запрос SIP, который используется для установки соединения, должен содержать необходимую для маршрутизации прокси-серверами информацию, например это может быть телефонный номер, набранный вызывающим абонентом.

На практике очень важно стандартизировать процедуры трансляции информации из ISUP в SIP (например, Called Party Number в ISUP IAM должен быть записан в заголовок To и поле Request-URI и т.д.).

Одной из проблем трансляции 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, такое, где параметр заголовка To и поля Request-URI запроса SIP отличаются 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.

3.3.4 Поддержка передачи сигнальных сообщений во время сеанса

Базовые сообщения протокола SIP не могут обеспечить передачу сигнальных сообщений во время сеанса связи. Для этих целей необходимо использовать дополнительное сообщение INFO [RFC 2976]. Однако этот запрос не подходит для передачи сигналов, когда телефонный номер из ТфОП передается по частям (overlap dialing). Также это сообщениене не рекомендуется использовать для передачи сигналов DTMF. Описание механизмов передачи сигналов DTMF представлено в [RFC2833 “RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals”].

3.4 Согласование содержимого сообщений протокола SIP

Шлюз, который отправляет запрос, может включить в сообщение тело, состоящее из нескольких частей различных типов: например, описание сеанса связи SDP и сообщение ISUP. Если получатель не поддерживает разделения тела сообщения на нескольких частей (формат multipart/mixed) и/или пришедший в сообщении тип ISUP MIME (application/ISUP), то он отклоняет пришедший запрос и посылает ответ с кодом 415 (Unsupported Media Type), в котором содержатся поддерживаемые форматы (по умолчанию - application/SDP). Шлюз, который посылал сообщение, должен впоследствии переслать запрос еще раз, предварительно удалив из него часть тела сообщения с сообщением ISUP (т.е. оставить только описание SDP) и данный запрос будет принят.

Это приводит к необходимости наличия механизма, при котором устройство,

отправляющее 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 входит в обязательные, то отклоняется все сообщение. Так, например, для терминала SIP потеря части тела, в которой содержится сообщение ISUP, совершенно не критична.

Примеры реализации данного механизма:

Поддержка ISUP необязательна. UA 2 принимает INVITE, независимо от того может он обработать ISUP или нет.

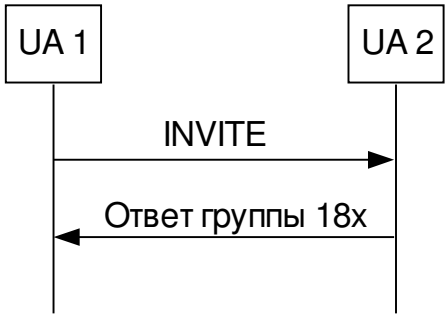


Рисунок 3.6.1 Диаграмма обмена сообщениями в между узлами в сети SIP. Поддержка ISUP необязательна.

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

2. Поддержка ISUP предпочтительна. UA 2 не поддерживает ISUP и отклоняет запрос, посылая сообщение об ошибке 415 (Unsupported Media Type). UA 1 выбрасывает из запроса часть с ISUP и опять посылает сообщение, содержащее только информацию SDP и оно принимается UA 2.

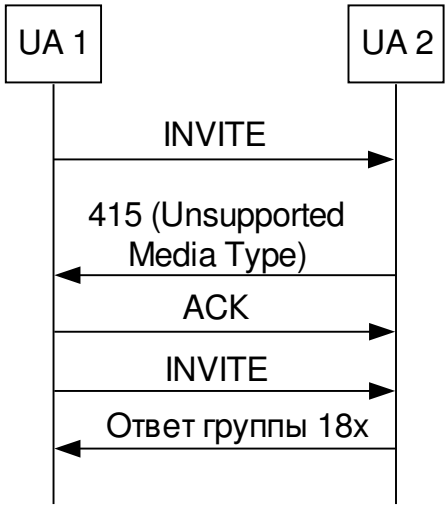


Рисунок 3.6.2 Диаграмма обмена сообщениями в между узлами в сети SIP. Поддержка ISUP предпочтительна.

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-->

INVITE-->
(Content-type: application/sdp)

<--18x

3. Поддержка ISUP критична для входящего звонка. UA2 не поддерживает ISUP и посылает сообщение об ошибке 415 (Unsupported Media Type). Тогда UA1 перенаправляет запрос на UA 3.

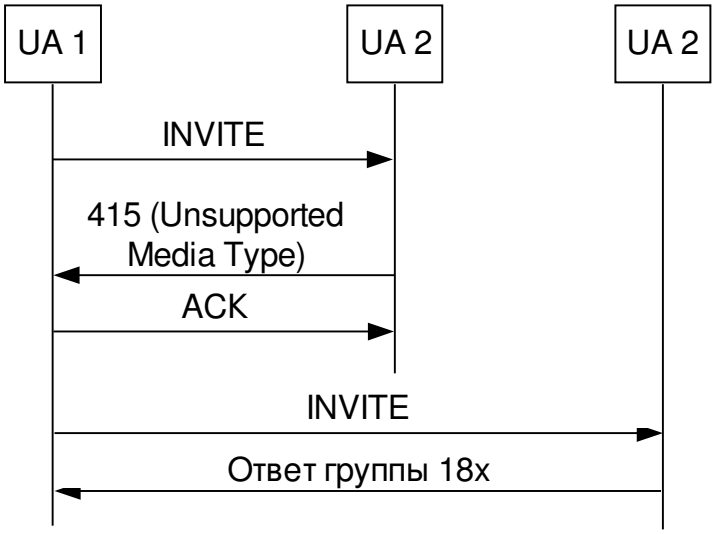


Рисунок 3.6.3 Диаграмма обмена сообщениями в между узлами в сети SIP. Поддержка ISUP обязательна.

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;)

3.5 Процедуры обеспечения безопасности

Протокол SIP–T может работать как внутрисетевой сигнальный механизм, подчиняясь установленным ранее 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 элементы обнаружения и исключения несанкционированных запросов, чтобы гарантировать то, что передача вложенного ISUP не приводит к нарушению безопасности.

Сообщения ISUP, инкапсулированные в запросы SIP, должны быть защищены, желательно использование средств защиты тел сообщения S/MIME [RFC 2976]. Более подробно это описано в главе Security Considerations в основной спецификации протокола SIP [RFC 3261] и главе 2. Процедуры аутентификации S/MIME гарантируют конечному пользователю то, что сообщение ISUP было отправлено зарегистрированным терминалом. Шифрование обеспечивает то, что правильно распознать вложенное ISUP сообщение смогут только операторы, имеющие необходимый ключ.

3.6 Преобразование сигнальных протоколов ISUP и SIP

3.6.1 Общие принципы взаимодействия

Протокол SIP обычно работает поверх протокола IP, разговор пользователей рассматривается как мультимедийный сеанс связи, включающий передачу аудио данных.

ISUP работает в стеке протоколов ОКС №7 поверх подистемы MTP, а также может функционировать на базе технологий IP [RFC 2960]. ISUP предназначен для установления соединений и управления ими, а также для обслуживания сети.

Устройство, содержащее модуль преобразования сообщений между протоколами ISUP и SIP могут называть Media Gateway Controller (MGC), также используется термин «softswitch» или «call agent». MGC имеет логический интерфейс для работы с обоими типами сетей, ISUP и SIP. Преобразованием аудио информации из формата, принятого в сети SIP в формат ТфОП занимается Media Gateway (MG) с магистральным интерфейсом E1/T1 (со стороны ТфОП) и интерфейсом IP (со стороны сети IP). MGC и MG могут быть объединены физически в одно устройство или находится отдельно.

MGC используется как связующее звено между ТфОП и сетями SIP, так как вызовы из телефонной сети могут поступать на IP–телефоны и наоборот, а вызовы из ТфОП могут проходить транзитом через сеть SIP.

Взаимодействие этих двух сетей основано на следующих принципах: инкапсуляции сообщений ISUP в тело запросов SIP и трансляции части информации сообщения ISUP, необходимой для правильной маршрутизации, в заголовок запроса SIP.

Для сообщений ISUP, проходящих через сеть SIP, трансляция позволяет таким элементам сети SIP, как прокси–серверы, которые не умеют работать непосредственно с сообщениями ISUP, правильно передавать сообщение, основываясь на данных, транслированных из сообщения ISUP в заголовок запроса SIP (например, номер вызываемого абонента).

3.6.2 Требования к протоколу SIP при взаимодействии с сетью ТфОП

Для того чтобы преобразование сообщений из формата ISUP в SIP и наоборот проходило правильно и без ошибок, используются несколько различных механизмов,

рассматриваемых ниже. Если SIP UAC/UAS получает сообщение в том формате, которого он не поддерживает, то установление соединения еще возможно, но при этом сценарий обмена сообщениями будет отличаться от стандартного.

3.6.2.1 Процедуры прозрачной передачи сообщений ISUP

Чтобы позволить шлюзам использовать полный объем услуг, предоставляемых существующей телефонной сетью при транзите сообщений ТфОП через сеть SIP (ТфОП-SIP-ТфОП), запрос SIP должен быть способен транспортировать полезную нагрузку в виде инкапсулированного сообщения ISUP от шлюза к шлюзу.

3.6.2.2 Процедуры поддержки формата MIME

В большинстве случаев при взаимодействии с ТфОП, сообщение SIP используется для транспортировки информации для установления медиа-соединения (по протоколу SDP), а также информации ISUP и/или биллинговых данных.

Узлы в сети должны поддерживать тип тела сообщения multipart/mixed, описанный в [RFC 2046]. Поддержка данного формата для клиента осуществляется путем добавления значения *multipart/mixed* в заголовок Accept.

3.6.2.3 Процедуры передачи многочастотного набора DTMF

При взаимодействии сети SIP и сети ISUP часто возникает задача передать сигналы DTMF. Передача таких сигналов через сеть SIP вызывает ряд проблем, однозначного решения которых на данный момент не существует.

Кодек, который используется в сети SIP для сжатия речи не может использоваться для передачи сигналов DTMF, т.к. сигналы DTMF могут сильно искажаться. Поэтому должен применятся символьный метод передачи в полосе. Этот метод описан в [RFC 2833 “RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals”].

3.6.2.4 Процедуры обеспечения проключения голосовых трактов в предответном состоянии

В предответном состоянии пользователю может передаваться аудио информация. Предответным называют состояние вызова до того, как соединение будет установлено полностью, т.е. будет получен окончательный ответ 2xx

При работе с сетью ТфОП желательно проключение такого медиапотока в обратном направлении для того, чтобы пользователю могли проигрываться мелодии и сообщения. Нужно отметить, что сообщение INVITE практически всегда содержит вложение в формате SDP, которого достаточно, для того чтобы проключить голосовой 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 голосовые 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 сессии, не может быть установлен. Поэтому шлюзы должны поддерживать более сложные методы проключения голосовых трактов. Один из таких методов описан в [RFC 3311].

3.6.2.5 Процедуры поддержки обмена сообщениями, не меняющими состояния конечного автомата протокола SIP, во время активной сессии

При взаимодействии с ТфОП могут возникать ситуации, когда необходимо передать сигнальные сообщения, в то время, когда голосовой тракт уже проключен. Для того этих целей используется запрос INFO, так как использование одного из 6 основных запросов SIP приведет к обрыву соединения. Подробное описание INFO содержится в [RFC 2976] или главе 2 раздел 2.2.3. Все шлюзы в сети должны уметь обрабатывать этот запрос.

Шлюзы также должны интерпретировать ответы с кодом 405 (Method Not Allowed) и 501 (Not Implemented), как не критичные ответы на запросы INFO, то есть если второй шлюз не поддерживает полученный запрос INFO, то данное соединения не прерывается.

3.6.2.6 Поддержка механизмов обеспечения конфиденциальности

ISUP имеет средства, с помощью которых пользователь может определить, хочет ли он, чтобы его номер определялся у вызываемого абонента. Когда шлюзы получают запрос с запретом определения 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. Один из таких механизмов, который поддерживает полностью засекреченную передачу информации описан в [RFC 3323] и главе 2.

3.6.2.7 Процедуры обеспечения информации о причинах, вызвавших отсылку 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 посылается для прекращения диалога установления соединения. Однако запрос CANCEL не может содержать вложений, и поэтому в случае транзита трафика сообщение REL не может быть инкапсулировано в запрос CANCEL.

Описанная выше проблема возникает редко, так как на практике REL посылается, если

абонент не дождался ответа и повесил 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 стандартного 16 (Normal clearing). Для таких случаев шлюзы могут поддерживать механизмы передачи соответствующего кода причины. Один из этих механизмов - заголовок Reason описан в [RFC3326.] и главе 2 раздел 2.2.2.

3.6.3 Преобразование сигнальных сообщений протокола ISUP в SIP

3.6.3.1 Процесс обмена сообщениями

Приведенные далее сценарии иллюстрируют порядок следования сообщений для типичного примера успешного установления соединения или незавершенного в результате ошибки в случае, когда соединение инициируется из сети ТфОП. Ответ с кодом 100 (Trying), посылаемый в то время, пока не придет ответ на INVITE, не изображается, так как его наличие не обязательно и зависит от конфигурации сети.

На диаграммах, вся сигнализация (ISUP и SIP) идет через MGC; управление медиапотоком (например, освобождение каналов) осуществляется MG под контролем MGC. Для упрощения диаграмм здесь представлено одно устройство, названное “MGC/MG”.

Установление соединения (быстрый ответ не используется).

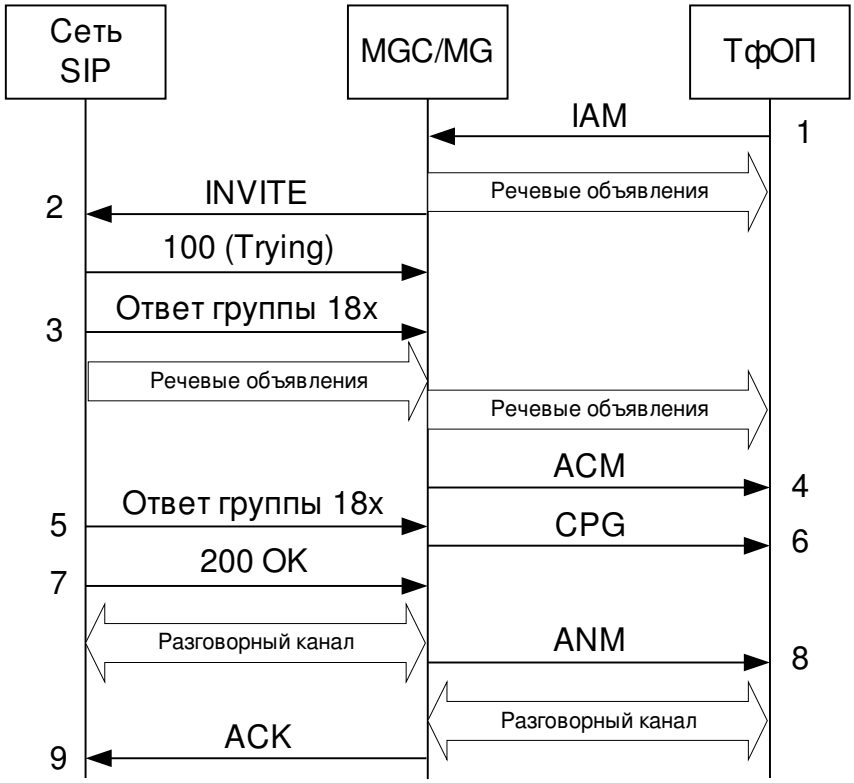


Рисунок 3.7 Диаграмма обмена сообщениями при установлении соединения без быстрого ответа.

1. Когда абонент сети ТфОП вызывает пользователя сети SIP, из ТфОП посылается сообщение IAM в направлении нужного шлюза.
2. После получения IAM шлюз посылает INVITE соответствующему узлу в сети SIP.
3. Если адресной информации достаточно, то узел SIP генерирует предварительный ответ группы 18x.
4. При получении ответа группы 18x шлюз отправляет 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. Узел сети SIP использует предварительные ответы, чтобы индицировать состояние процесса обслуживания вызова.
6. После того, как ACM будет послан, все временные ответы будут транслироваться в сообщения ISUP CPG, как обозначено в параграфе 3.2.3.2.
7. После того как узел в сети SIP ответит на вызов, будет послано сообщение 200 (OK).
8. При получении ответа 200 (OK) шлюз посылает в сеть ТфОП сообщение ANM.
9. Шлюз посылает узлу в сети SIP сообщение ACK с подтверждением параметров переданных в INVITE.

Установление соединения (быстрый ответ включен).

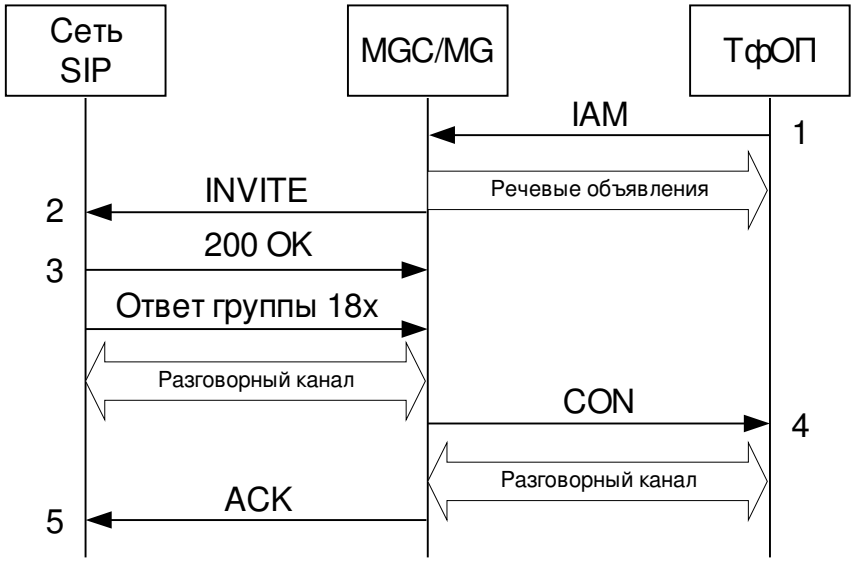


Рисунок 3.8 Диаграмма обмена сообщениями при установлении соединения с использованием быстрого ответа.

1. Когда абонент ТфОП вызывает пользователя сети SIP, то из ТфОП посылается сообщение IAM в направлении нужного шлюза
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и отправляет его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
3. Далее если на узле сети SIP установлен автоматический ответ, то шлюзу посылается ответ 200 (OK).
4. После получения ответа 200 (OK) шлюз посылает сообщение CON тому узлу ISUP, который запрашивал установление соединения.
5. Шлюз посылает узлу в сети SIP сообщение ACK с подтверждением параметров переданных в INVITE.

Таймеры протокола SIP

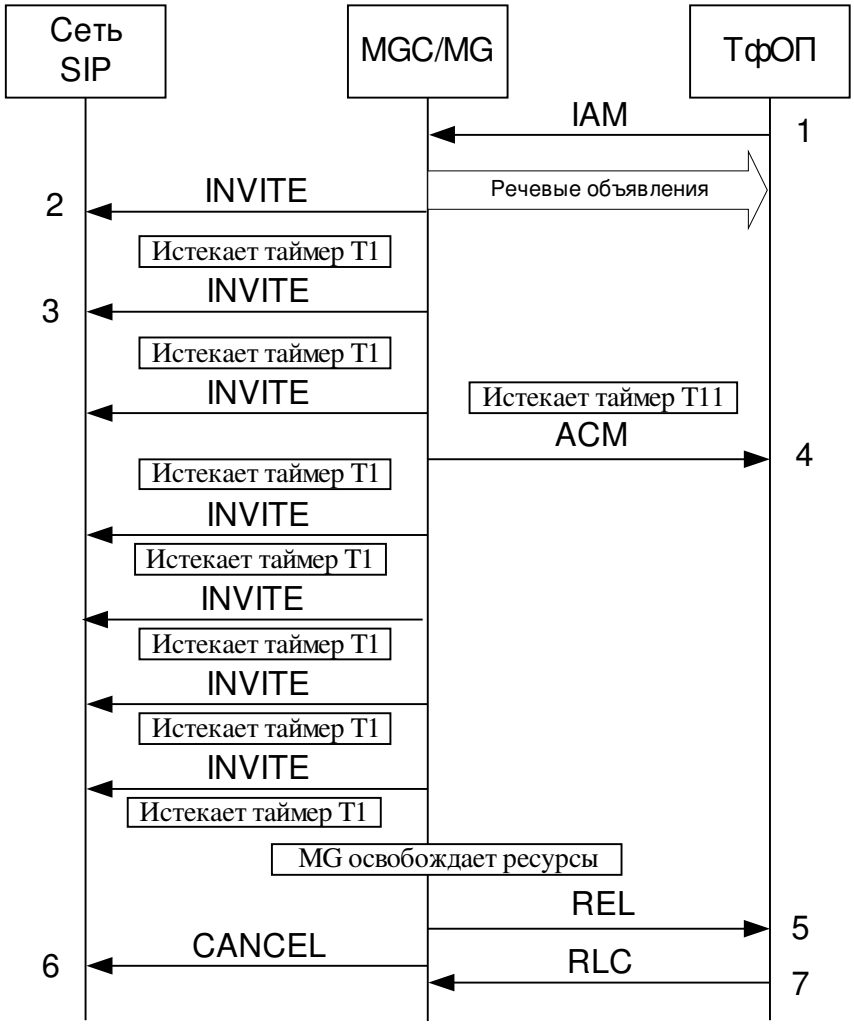


Рисунок 3.9 Диаграмма обмена сообщениями при истечении таймеров в сети SIP.

4. Когда абонент ТфОП вызывает пользователя сети SIP, то в направлении соответствующего шлюза из ТфОП посылается сообщение IAM .
5. После получения сообщения IAM шлюз генерирует сообщение INVITE и отправляет его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера. Включаются таймер ISUP T11 и таймер SIP T1.
6. Каждый раз по истечении таймера T1 сообщение INVITE посылается еще раз. В основной RFC по протоколу SIP – [RFC 3261] определено, что сообщение INVITE может посылаться 7 раз.
7. По истечению таймера T11 узлу ISUP посылается сообщение ACM для начала прерывания устанавливаемого соединения. При этом на удаленном узле ISUP включается таймер T7. В сообщении ACM в поле «Called Party Status» записано значение *no indication* [нет признака].
8. Как только будет послано максимально возможное число сообщений INVITE, шлюз посылает узлу ISUP сообщение REL для прекращения установления соединения.
9. Шлюз также посылает сообщение CANCEL узлу SIP для прекращения любых попыток ответа.

10. После получения сообщения REL узел ISUP посылает шлюзу сообщение RLC для подтверждения.

Истечение таймера ISUP T9

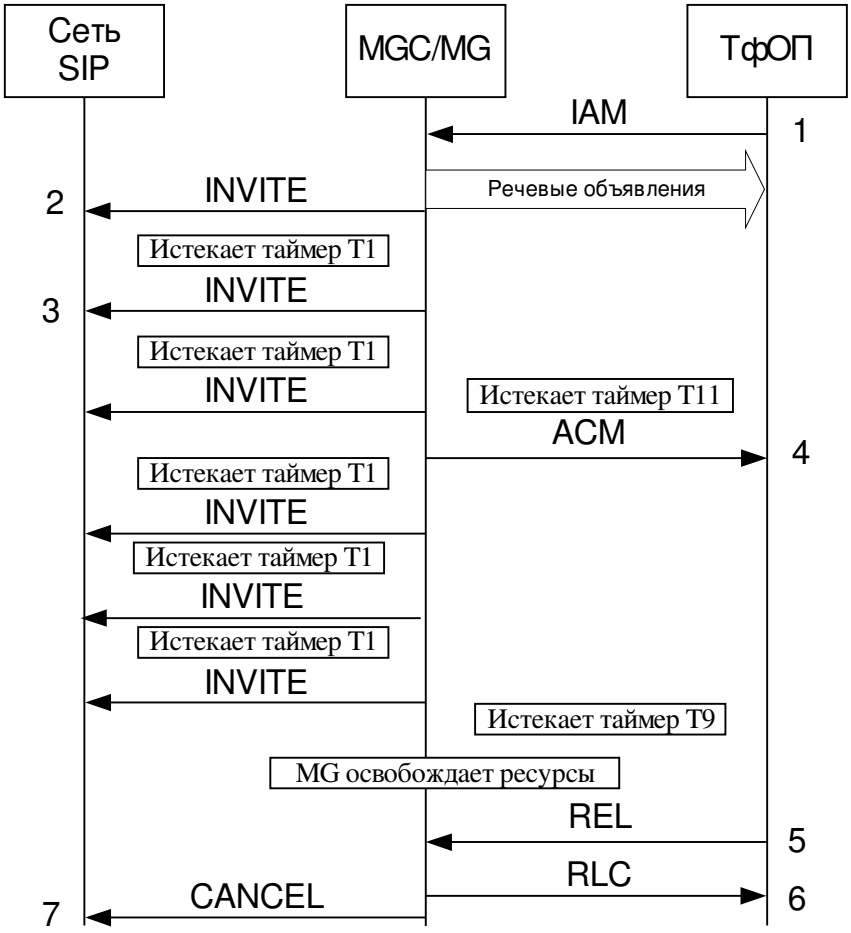


Рисунок 3.10 Диаграмма обмена сообщениями при истечении таймера T9 в сети ТфО П.

1. Когда абонент ТфОП вызывает пользователя сети SIP, то в направлении нужного шлюза из ТфОП посылается сообщение IAM.
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и отправляет 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 по истечении таймера T1 сообщение INVITE посылается еще раз, максимум 7 раз. С момента запуска T1 проходит 0.5 секунды или больше, после чего сообщение посылается еще раз. Если для T1 задана длительность более 500 мс, то возможно, что времени на посылку 7 запросов INVITE потребуется больше, чем время истечения таймеров ISUP T11 + ISUP T9.
4. По истечению таймера T11 узлу ISUP посылается сообщение ACM для начала прерывания устанавливаемого соединения. При этом на удаленном ISUP узле включается таймер T7. В сообщении ACM в поле Called Party Status (статус вызываемой стороны) записано значение no *indication* (нет признака).

5. Когда таймер Т9 на узле в сети ТфОП истекает, то шлюзу посылается сообщение REL.
6. После получения сообщения REL шлюз посылает узлу сети ТфОП RLC для подтверждения.
7. Сообщение REL также указывает шлюзу, что надо послать CANCEL узлу в сети SIP.

Ошибки в сети SIP при установлении соединения

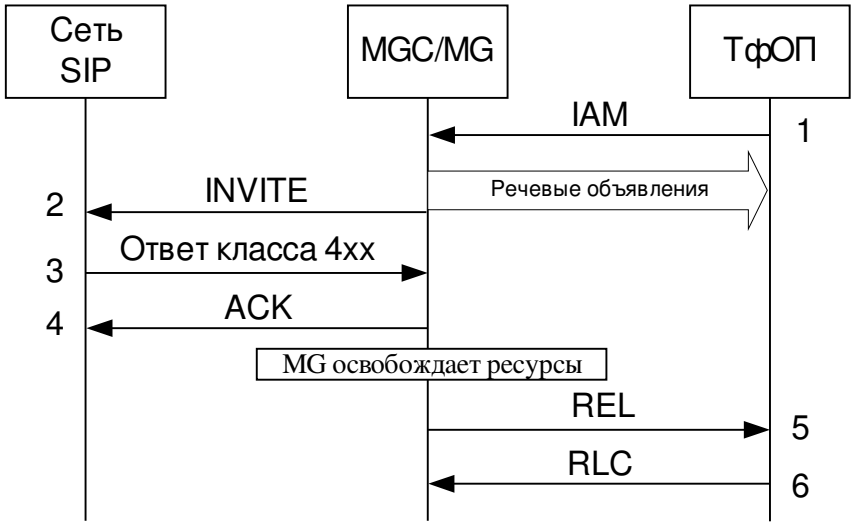


Рисунок 3.11 Диаграмма обмена сообщениями при неуспешном установлении соединения из – за ошибки в сети SIP.

7. Когда абонент ТфОП вызывает пользователя сети SIP, то в направлении нужного шлюза из ТфОП посылается сообщение IAM.
8. После получения сообщения IAM шлюз генерирует сообщение INVITE и отправляет его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
9. Узел в сети SIP индицирует ошибку, посылая ответ с кодом 400 (Bad Request) или выше к шлюзу.
10. Шлюз посылает узлу в сети SIP сообщение ACK для подтверждения параметров, переданных в запросе INVITE.
11. Шлюз генерирует сообщения REL узлу в сети ТфОП с кодом события, соответвтвующему из коду ответа SIP.
12. Узел в сети ТфОП посылает RLC в подтверждение приема REL.

Перенаправление запросов с сети SIP

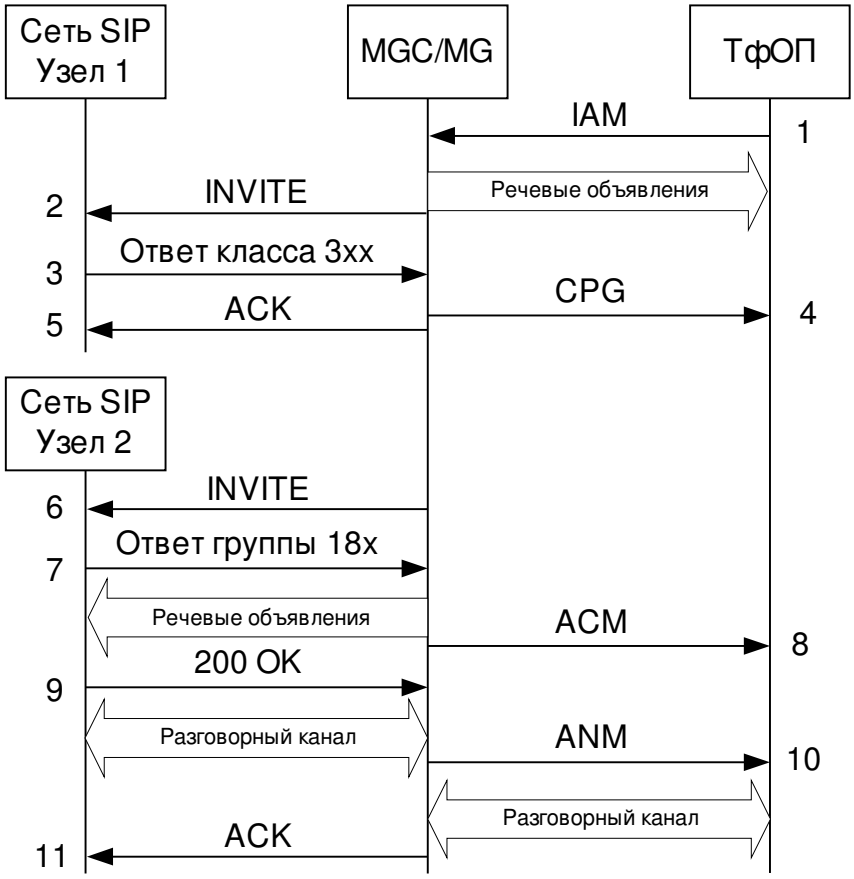


Рисунок 3.12 Диаграмма обмена сообщениями при перенаправлении запросов в сети SIP.

1. Когда абонент ТфОП вызывает пользователя сети SIP, то в направлении нужного шлюза из ТфОП посылается сообщение IAM
2. После получения сообщения IAM шлюз генерирует сообщение INVITE и отправляет его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
3. Посылая шлюзу ответ класса 3xx, узел сети SIP указывает, что пользователь, с которым хочет установить соединение абонент ТфОП, имеет другой адрес. На данном этапе предполагается, что Contact URL является адресом, на который нужно направлять вызов.
4. Шлюз посылает сообщение CPG с содержанием события, которое пришло в ответе класса 3xx. Однако при этом надо следить, сможет ли узел корректно распознать запрос, т.к. не все терминалы ISUP поддерживают передачу CPG до того, как пришло сообщение ACM.
5. Шлюз посылает узлу в сети SIP сообщение ACK для подтверждения параметров, переданных в запросе INVITE.
6. Шлюз пересылает запрос по адресу, указанному в заголовке Contact пришедшего ранее ответа класса 3xx.
7. Когда узел сети SIP примет необходимое количество адресной информации чтобы определить ему пришел запрос или нет, то он посылает предварительный ответ группы 18x, если адрес правильный.
8. После получения ответа с кодом 180 (Ringing) шлюз посылает узлу сети SIP сообщение ACM с кодом этого события.

9. Когда узел сети SIP ответил на звонок, посылается ответ с кодом 200 (OK).
10. После получения 200 (OK) шлюз посылает сообщение ANM в направлении узла в сети ТфОП.
11. Шлюз посылает узлу в сети SIP сообщение ACK для подтверждения параметров, переданных в запросе INVITE.

Установление соединения прерывается со стороны ТфО П

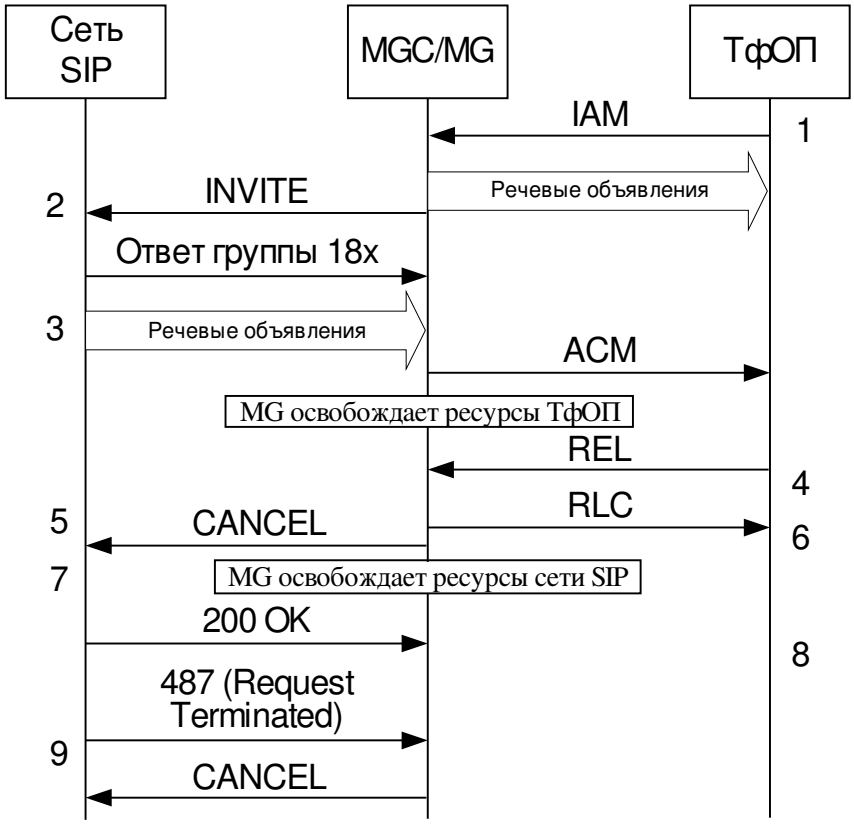


Рисунок 3.13 Диаграмма обмена сообщениями при прерывании установления соединения состороны ТфО П.

9. Когда абонент ТфОП вызывает пользователя сети SIP, то в направлении нужного шлюза из ТфОП посылается сообщение IAM.
10. После получения сообщения IAM шлюз генерирует запрос INVITE и отправляет его соответствующему узлу сети SIP, основываясь на анализе набранного абонентом ТфОП номера.
11. Когда узел в сети SIP примет достаточно адресной информации для подтверждения правильности доставки запроса INVITE, то он посылает шлюзу предварительный ответ с кодом 180 (Ringing) или выше.
12. После получения 180 (Ringing) шлюз посылает в сеть ТфОП сообщение ACM
13. Если вызывающая сторона (т.е. узел сети ТфОП) повесит трубку раньше, чем узел в

сети SIP ответит на звонок, то генерируется сообщение REL.

14. Шлюз освобождает тракт передачи в сети ТфОП и идентифицирует его как доступный для дальнейшего использования, посылая сообщение RLC.
15. При получении сообщения REL до того как был получен ответ на запрос INVITE шлюз посылает узлу в сети SIP запрос CANCEL.
16. При получении CANCEL узел сети SIP отправляет в подтверждение ответ 200 (OK).
17. Узел сети SIP посылает ответ с кодом 487 (Call Cancelled) для подтверждения окончания обработки запроса INVITE.
18. Шлюз посылает узлу в сети SIP сообщение ACK для подтверждения параметров, переданных в запросе INVITE.

3.6.3.2. Конечные автоматы SIP-T при взаимодействии ISUP-SIP

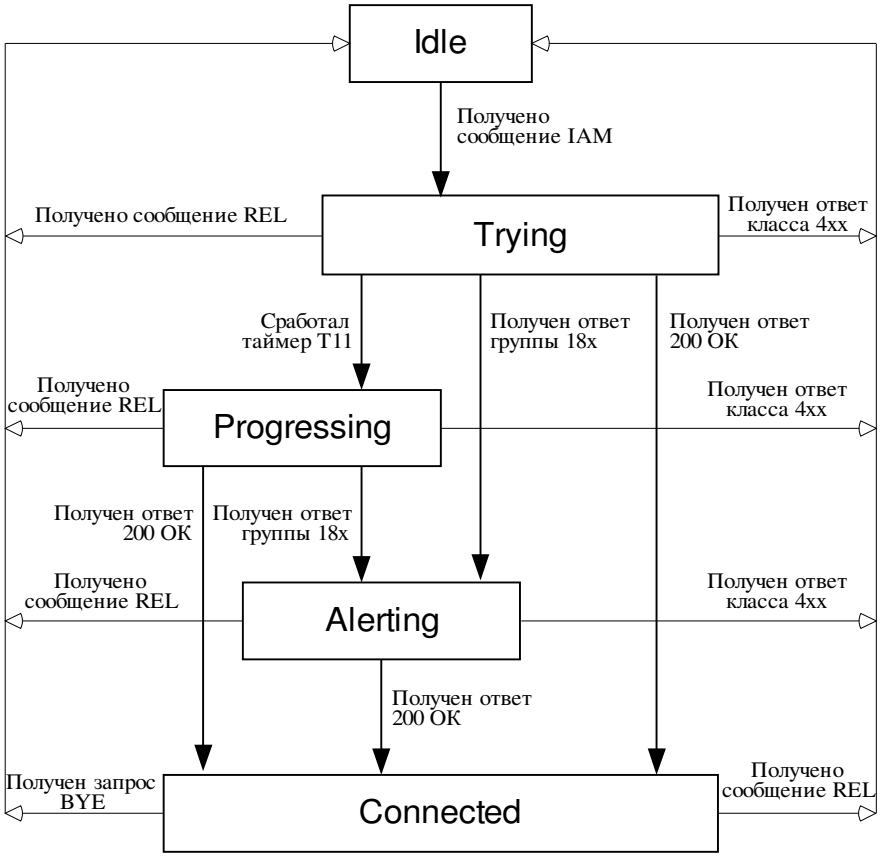


Рисунок 3.14 Конечные автоматы SIP-T при взаимодействии ISUP-SIP.

Начало получения адресной информации

После получения сообщения IAM, шлюз должен зарезервировать подходящий внутренний ресурс, например, цифровой сигнальный процессор DSP и подобные ему устройства, необходимые для управления передачей сообщений в IP части устанавливаемого разговорного тракта.

Процедуры преобразования сообщения IAM в запрос INVITE

Если шлюз получает сообщение IAM, он должен создать запрос INVITE для передачи в сеть SIP. Далее описывается, каким образом шлюз заполняет все поля заголовков сообщения INVITE, основываясь на параметрах пришедшего сообщения IAM.

При создании INVITE первое, что должно быть транслировано из IAM это два универсальных идентификатора ресурса (URI), один – адрес вызывающего абонента, второй – адрес вызываемого пользователя в сети SIP. Первый адрес помещается в заголовок From сообщения INVITE, после того как нужная информация будет конвертирована из формата ISUP, а второй адрес в большинстве случаев транслируется, как в заголовок To, так и в поле Request URI.

Запрошенный адрес вызываемого пользователя должен быть транслирован в tel URL , который используется в поле Request URI сообщения INVITE. Шлюз также может сначала попытается использовать функцию трансляции телефонного номера [Telephone Number Mapping (ENUM)[RFC 2916]] для преобразования телефона вызываемого пользователя в SIP URI. После завершения трансляции к телефонному URI должны быть добавлены некоторые дополнительные поля ISUP, а именно:

- Если сеть поддерживает услугу переносимости номера (опционально), тогда при анализе сообщения шлюз должен обработать еще несколько параметров. Бит 'number translated' параметра FCI показывает, что услуга переносимости 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, тогда содержимое поля номера вызываемого абонента (CPN) (в данном случае Location Routing Number - LRN) должно быт транслировано 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 трансляции в различные поля tel URL. LRN всегда используется на территории одной страны, и, следовательно, поле 'rn=' не должно содержать знака '+'. Подробнее о процедурах услуг переносимости номера см. [Number Portability in the Global Switched Telephone Network (GSTN): An Overview, RFC 3482].
- Поля, которые используются при маршрутизации по идентификатору сети (оператора). Подробнее о поддержке такого типа маршрутизации излагается в [RFC 3398].

В большинстве случаев, адрес, записываемый в поле tel URL, содержится в заголовке To и поле Request-URI. Однако когда в сообщении IAM содержится параметр OCN, то значение заголовка To должно быть образовано трансляцией номера из OCN, и тогда заголовок To и поле Request-URI могут различаться.

Конструкция поля заголовка From зависит то наличия параметра ISUP Calling Party's Number (CIN). Если CIN отсутствует, тогда шлюз должен сформировать поле заголовка From,

содержащим SIP URI, без пользовательской части, содержащим только адрес шлюза (например, sip:gw.protei.ru). Если CIN присутствует, тогда он должен быть транслирован в tel URI, содержащемся в поле заголовка From.

Сообщения 100 (Trying)

При приеме ответа с кодом 100 (Trying) в ТфОП не должно передаваться никаких сообщений; это сервисный ответ, который говорит лишь о том, что надо прекратить посылать INVITE.

Сообщения группы 18x

Если до получения ответа группы 18x, не содержащего вложений ISUP, сообщение ACM еще не было послано, тогда сообщения, посылаемые в сторону абонента ТфОП от шлюза генерируются соответственно следующей таблице. Если сообщение ACM было передано до получения ответа группы 18x, вызов переходит в состояние «Progressing» вместо «Alerting» (см диаграмму конечных автоматов).

Таблица 3.4

Пришедший ответ	Сообщение, посылаемое 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 послано, и ответ группы 18x не содержит вложений ISUP, то сообщения от шлюза генерируются соответственно следующей таблице.

Таблица 3.5

Пришедший ответ	Сообщение, посылаемое 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 группы 18x шлюз должен послать КПВ вызывающему абоненту из ТфОП (за исключение случаев, когда шлюз знает, что КПВ будет передаваться от узла ТфОП).

Шлюз может получать медиа-данные в любой момент после того, как будет передана запрос INVITE с вложением SDP, даже до того как будет получен предварительный ответ

группы 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 с кодом 183 (Session progress) в то время, когда абоненту ТфОП воспроизводились медиа-данные, то шлюз должен транслировать это сообщение, если в нем не содержалось 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 отправлено, 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. Если в принятом сообщении группы 18x содержалось разрешенное и доступное для обработки сообщение ISUP, то шлюз должен повторно установить соответствующие параметры, содержащиеся в теле вложенного сообщения, включая параметр BCI, поскольку он определяет сообщение, которое будет послано в сеть ТфОП. В отсутствии вложенного сообщения параметр BCI должен быть установлен следующим образом:

Таблица 3.6

Message type:	ACM
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

Когда у параметра BCI в поле Interworking indicator (индикатор наличия взаимодействия) установлено значение interworking encountered (взаимодействие на некоторых участках), это означает, что сеть ISDN взаимодействует с сетью, которая не поддерживает большинство сервисов, предоставляемых сетью ISDN, и не может использовать функции, которые обычно применяются.

Сообщения класса 2xx

Таблица 3.7

Пришедший ответ	Сообщение, посылаемое шлюзом
200 (OK)	ANM, ACK

После того, как получен ответ 200 (OK), шлюз должен создать медиапоток в нужном направлении, а также послать сообщение ANM абоненту ТфОП и сообщение ACK пользователю сети SIP.

Если ответ 200 (OK) придет на шлюз до того, как будет послано сообщение ACM, то вместо ANM будет послано сообщение CON, если используемый вариант ISUP поддерживает данное сообщение.

Если в пришедшем ответе 200 (OK) содержалось понятное шлюзу инкапсулированное сообщение ANM, то шлюз должен заполнить все параметры отправляемого абоненту ТфОП сообщения ANM, используя вложенное сообщение.

Обратите внимание, что шлюз может получать ответ 200 (OK) не только в ответ на запрос IVNITE, а также, например, на запрос INFO. Процедуры, описанные в этом разделе, относятся только к тем ответам с кодом 200 (OK), которые были получены в ответ на запрос INVITE. Если 200 (OK) пришел в ответ не на запрос INVITE, то шлюз не должен посылать абоненту ТфОП никаких сообщений.

Сообщения класса 3xx

Когда шлюз получает ответ класса 3xx, он должен определить новое местоположение пользователя, послав один или несколько запросов на установление нового соединения. При этом используются адреса, указанные в заголовке Contact пришедшего ответа. Ответы класса 3xx обычно посылаются сервером переадресации.

Если в заголовке Contact ответа класса 3xx содержится URI, который находиться в ТфОП (т.е. шлюз используется как транзит), то шлюз должен послать новый запрос и работать как обычный коммутатор ТфОП (без применения SIP).

Кроме этого, в ответ на сообщение класса 3xx, шлюз может послать сообщение REL абоненту ТфОП, 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 заголовка Contact.

Пока шлюз обрабатывает список адресов из заголовка Contact (генерирует и посылает новые INVITE), он может послать 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 специфические ситуации, например при биллинге услуг.

Сообщения классов 4xx – 6xx

Если на шлюз поступает сообщение классов 4xx-5xx, то это означает, что отправленный запрос INVITE был отклонен. В этом случае шлюз должен освободить соответствующие ресурсы, послать сообщение REL с нужными параметрами абоненту ТфОП и запрос АСК пользователю сети SIP. В некоторых случаях, описанных ниже, шлюз может попытаться решить проблему путем повторной посылки исправленного запроса. После того, как шлюз посылает сообщение REL абоненту ТфОП, он ожидает обратного ответа RLC с подтверждением освобождения всех ресурсов.

Трансляция кодов ответов сети SIP в коды событий [Cause Code] ISUP

Если в сообщении класса 4xx и выше, пришедшем на шлюз, содержалось вложенное REL, то из него должен быть извлечен параметр Cause Indicator (индикатор события, CAI), а его значение использовано для заполнения этого же параметра сообщении REL, передаваемого далее в ТфОП. Если ответ пришел без вложенного сообщения, то рекомендуется использовать трансляцию из кодов ответов SIP в коды событий ISUP.

Любой код ответа сети SIP, не указанный ниже, должен быть транслирован в код события 31 (Normal, unspecified). Это относится только к ответам. Запросы BYE и CANCEL, которые используются для завершения диалога, в большинстве случаев должны быть транслированы в код 16 (Normal clearing).

По умолчанию, расположение участка сети, связанного с параметром CAI, должно быть указано таким образом, что коды 6xx определяют расположение пользователя, а коды 4xx и 5xx расположение сети. Исключения отмечены ниже.

Точно также, как имеются коды событий ISUP, используемые только в ТфОП и не имеющие аналогов в сети SIP, так и некоторые коды состояния SIP не транслируются в ISUP. Эти коды отмечены знаком (+)

Таблица 3.8

Код полученного ответа	Код события 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 (+)
414 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

(*) В некоторых случаях шлюз SIP может послать отклик аутентификации к UAS сети SIP, который отклонил запрос INVITE, т.к. из-за тербования аутентификации. Если шлюз может сам авторизовать пользователя, то он должен сделать это и продолжить установление соединения. Только в том случае если шлюзу не удается провести авторизацию пользователя должно быть послано сообщение с кодом 21 (Call rejected).

(+) Если это возможно, то шлюз должен реагировать на ошибки протокола, по возможности устраняя их и делая попытку возобновить процесс установления соединения. Сессию следует прерывать только в том случае, если шлюз не сможет корректно обработать ошибку.

Когда в ответах 606 (Not acceptable) и 488 (Not Acceptable here) содержится заголовок Warning, то трансляция в ISUP должна осуществляться с учетом информации в этих

заголовках. В большинстве случаев при трансляции рекомендуется использовать код события 31 (Normal, unspecified). Если код значения заголовка Warning несет информацию о том, что информацию невозможно доставить из-за того, что запрошенные средства передачи не поддерживаются (т.е. технически не реализованы), то для трансляции рекомендуется использовать код события 65 (Bearer Capability Not Implemented).

Получение сообщения REL

Сообщение может передаваться, когда абонент сети ТфОП вешает трубку до того, как был получен ответ на вызов, поэтому шлюз прерывает установление соединения. Пользователю SIP необходимо послать запрос CANCEL, сообщение BYE не используется в данном случае, т.к. не было получено окончательного ответа от пользователя сети SIP. На запрос CANCEL должен придти ответ 200 (OK). После этого на отправленный запрос INVITE ожидается окончательный ответ 487 (Request Terminated).

Шлюз должен сохранять историю обмена сообщениями для этого соединения на некоторое время, начиная с прихода ответа 200 (OK) на отправленный ранее INVITE (даже после приема 200 (OK) в ответ на запрос CANCEL). В этой ситуации, вслед за соответствующим запросом BYE, шлюз должен послать ACK.

В ситуациях, когда разговорная сессия 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), то шлюз может использовать другой метод обработки кодов событий.

Истечение таймера ISUP T11

Чтобы не допустить истечения удаленного таймера ISUP T7 шлюз может иметь свой собственный управляющий таймер. ISUP определяет этот таймер как T11. Длительность таймера T11 тщательно подобрана таким образом, что он всегда меньше таймера T7 любого узла, взаимодействующего со шлюзом в данный момент.

Если шлюз поддерживает использование таймера T11, то по его истечении должно быть послано предварительное сообщение ACM (early ACM) (т.е. параметр called party status (статус вызываемой стороны) устанавливается в значение ‘no indication’), чтобы предотвратить истечение таймера T7 удаленного узла в сети ISUP, если используемый вариант ISUP это поддерживает.

Если после этого приходит ответ 180 (Ringing), то должно быть послано соответствующее сообщение CPG, как это было показано ранее в разделе “Сообщения группы 18x”.

3.6.3.3 SDL-диаграммы приема и передачи сообщений при взаимодействии ТфОП – SIP

рисунок 3.15 Переход в состояние «Trying»

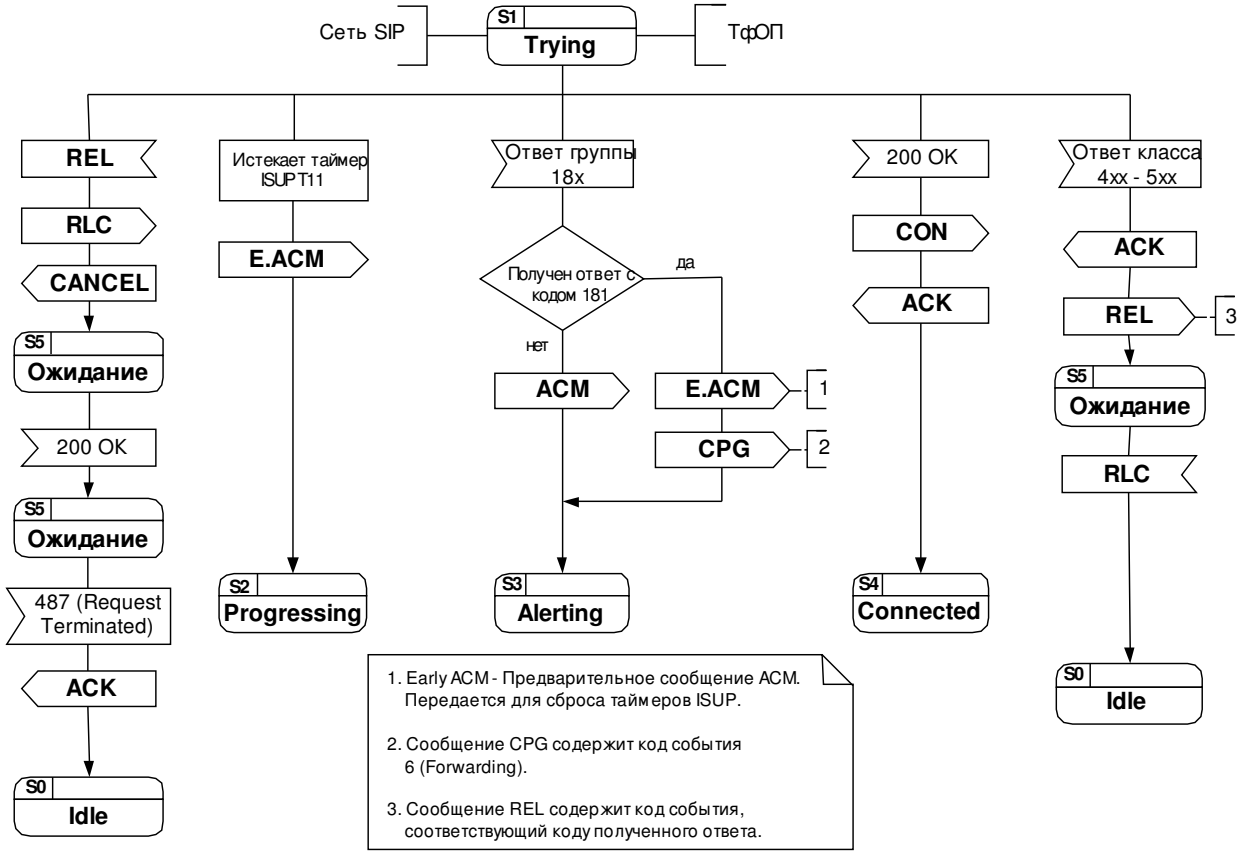


рисунок 3.16 Переходы из состояния «Trying»

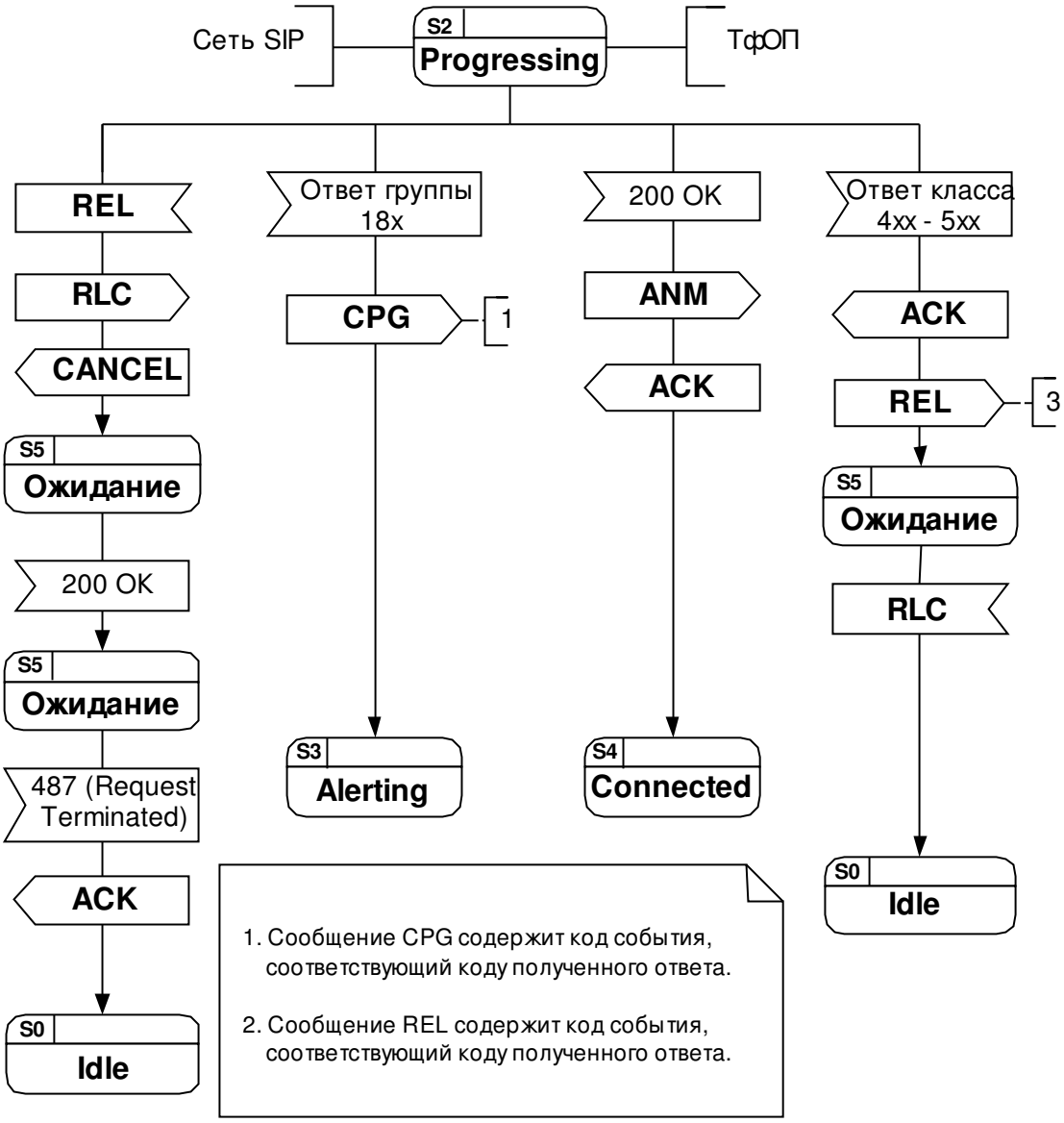
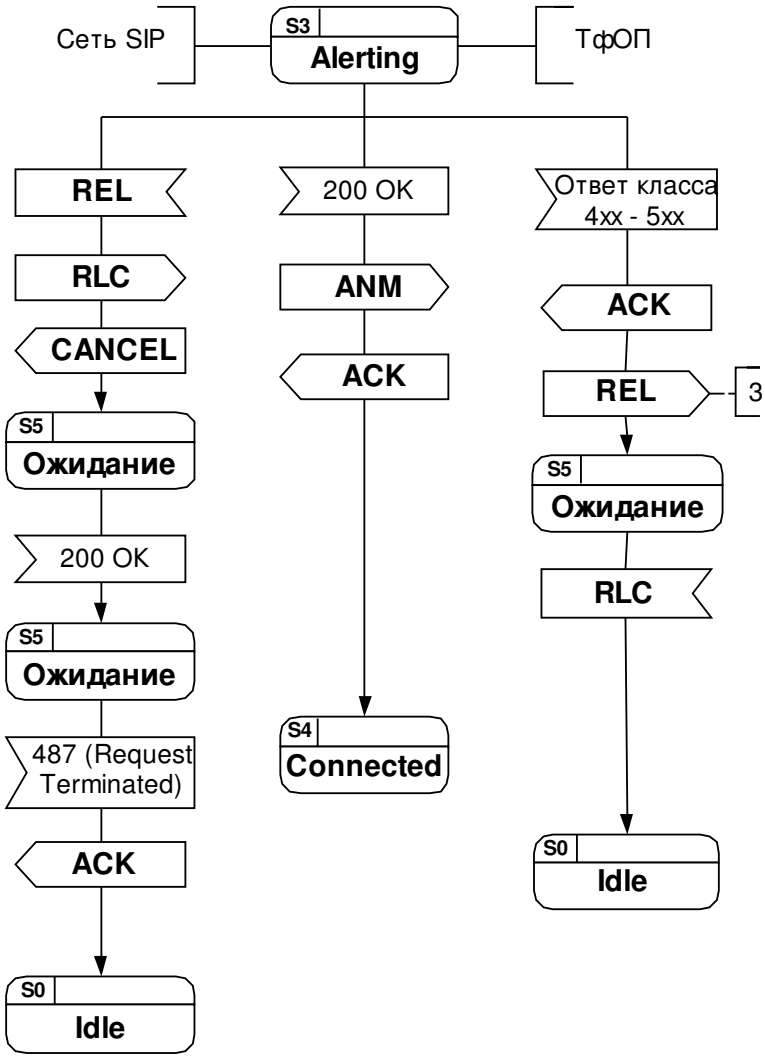


рисунок 3.17 Переходы из состояния «Progressing»



1. Сообщение REL содержит код события, соответствующий коду полученного ответа.

рисунок 3.18 Переходы из состояния «Alerting»

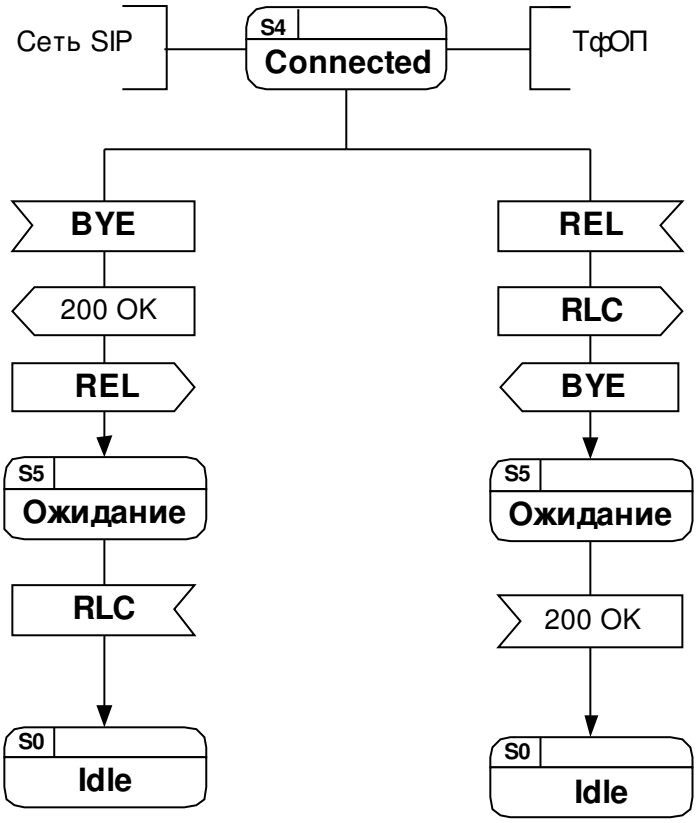


рисунок 3.19 Переходы из состояния «Connected»

Состояния:

S0 – «Idle» - исходное состояние - соединения не существует

S1 – «Trying» - пробное сстояние - периодически отсылается запрос INVITE, ожидание ответов.

S2 – «Progressing» - состояние приёма ответов – получено предварительное ACM, ожидание ответа группы 18x.

S3 – «Alerting» - состояние вызова - окончания обработки сообщений – вызываемый абонент получает ПВ, вызывающий – ПКВ, ожидание окончательного ответа 200 ОК.

S4 – «Connected» - состояние окончания обработки сообщений – соединение установлено.

S5 – «Ожидание» - состояние ожидания ответа.

3.6.3.4 Примеры сценариев для случая соединения ТфОП – SIP

В данном разделе приводятся 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 А 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.

Успешное установление соединения от абонента ТфО П к пользователю сети SIP

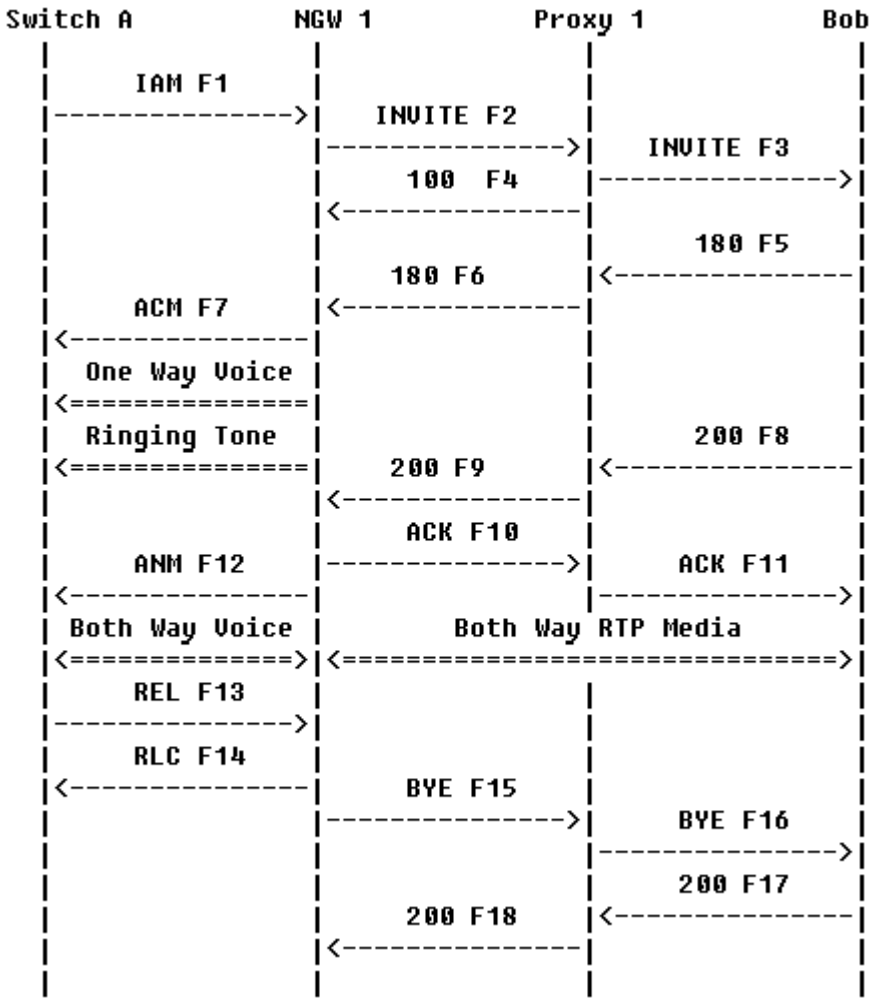


Рисунок 3.20 Диаграмма обмена сообщениями при успешном установлении соединения ТфО П – SIP.

В данном сценарии пользователь Maxim через Network Gateway 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. Разговор прерывается, когда абонент Maxim вешает трубку, АТС посылает сообщение REL на NGW1, где оно преобразуется в BYE для сети SIP.

Содержимое сообщений

F1 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

F2 INVITE Maxim -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения расположения для определения местонахождения пользователя Anton. Анализ месторасположения пользователя проводится на основании номера вызываемого пользователя Anton. NGW1 готовится принимать данные на порт 3456 от терминала абонента Maxim.

F3 INVITE Proxy 1 -> Anton

```
INVITE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
```

Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) Anton -> Proxy 1

SIP/2.0 100 Trying
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 180 (Ringing) Anton -> Proxy 1

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>
Content-Length: 0

F6 180 (Ringing) Proxy 1 -> NGW 1

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>
Content-Length: 0

F7 ACM NGW 1 -> ATC A

Получение сообщения ACM

F8 200 (OK) Anton -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
Contact: <sip:anton@client.b.loniis.ru>
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F9 200 (OK) Proxy 1 -> NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F10 ACK NGW 1 -> Proxy 1

```
ACK sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0
```

F11 ACK Proxy 1 -> Anton

```
ACK sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
```

Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F12 ANM Anton -> NGW 1

Получение сообщения ANM

Между абонентом Maxim и пользователем Anton установлен RTP поток (через GW)

Пользователь Maxim вешает трубку.

F13 REL Maxim -> NGW 1

Получение сообщения REL с CauseCode=16 Normal

F14 RLC NGW 1 -> Maxim

Получение сообщения RLC

F15 BYE NGW 1-> Proxy 1

BYE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F16 BYE Proxy 1 -> Anton

BYE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2

;received=192.0.2.103
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F17 200 (OK) Anton -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F18 200 (OK) Proxy 1 -> NGW 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

Успешное установление соединения от абонента ТфОП к пользователю сети SIP, быстрый ответ

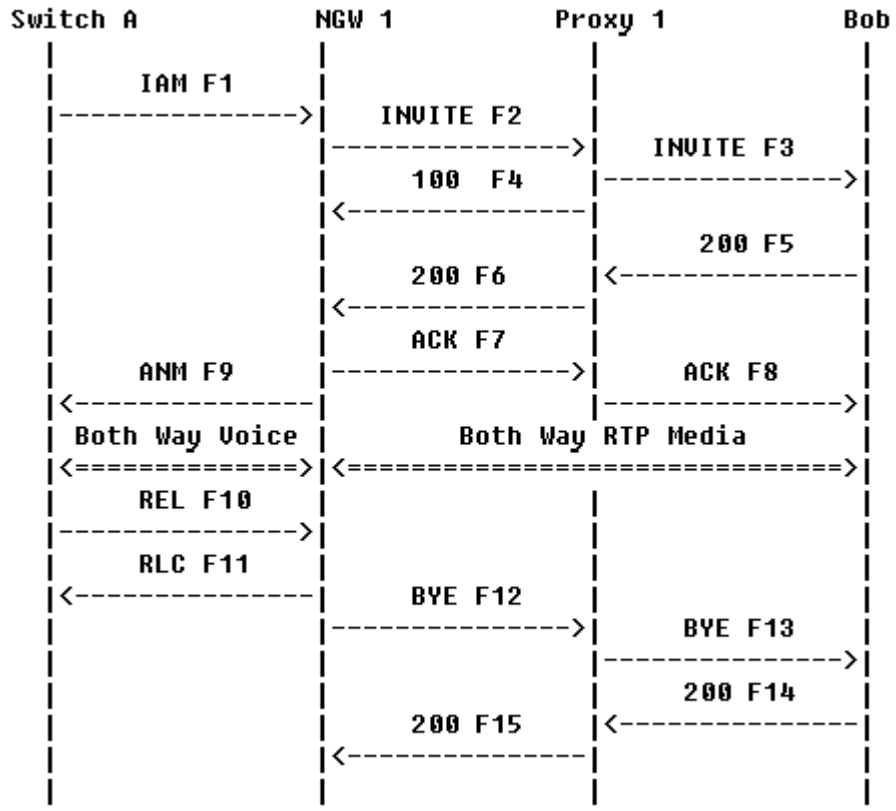


Рисунок 3.21 Диаграмма обмена сообщениями при успешном установлении соединения ТфО П – SIP. Используется быстырй ответ.

Этот сценарий «быстрого ответа» похож на предыдущий. Разница в том, что терминал пользователя Anton отвечает на звонок мгновенно. Т.е. от терминала пользователя Anton сразу поступает ответ с кодом 200 (ОК), без передачи предварительного ответа 180 (Ringing). Шлюз посылает сообщение ANM, пропуская сообщение ACM. Обратите внимание на то, что для протокола ISUP версии ETSI и некоторых других вариантов ISUP вместо ANM должно быть послано сообщение CON.

Содержимое сообщений

F1 IAM ATC A (Maxim) -> NGW 1

Получение сообщения IAM
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-262-5326,NPI=E.164,NOA=National

F2 INVITE NGW 1 -> Proxy 1

INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0

Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

Proxy 1 использует функцию определения расположения для определения местонахождения пользователя Anton. Анализ расположения проводится на основании номера вызываемого пользователя. NGW1 готовится принимать данные на порт 3456 от терминала пользователя Maxim.

F3 INVITE Proxy 1 -> Anton

INVITE anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-

c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) Proxy 1 -> NGW 1

SIP/2.0 100 Trying
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 200 (OK) Anton -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F6 200 (OK) Proxy 1 -> NGW 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F7 ACK NGW 1 -> Proxy 1

```
ACK anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0
```

F8 ACK Proxy 1 -> Anton

```
ACK anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=130.131.132.14
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
```

To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F9 ANM NGW1 -> Maxim

Передается сообщение ANM. Между абонентом Maxim и пользователем Anton установлен RTP поток (через GW).

Пользователь Maxim вешает трубку.

F10 REL Maxim -> NGW 1

Передается сообщение REL с CauseCode=16 Normal.

F11 RLC NGW 1 -> Maxim

Передается сообщение RLC.

F12 BYE NGW 1 -> Proxy 1

BYE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F13 BYE Proxy 1 -> Anton

BYE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F14 200 (OK) Anton -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F15 200 (OK) Proxy 1 -> NGW 1

SIP/2.0 200 OK
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

Успешное установление соединения абонента, являющегося абонентом УПАТС к пользователю сети SIP

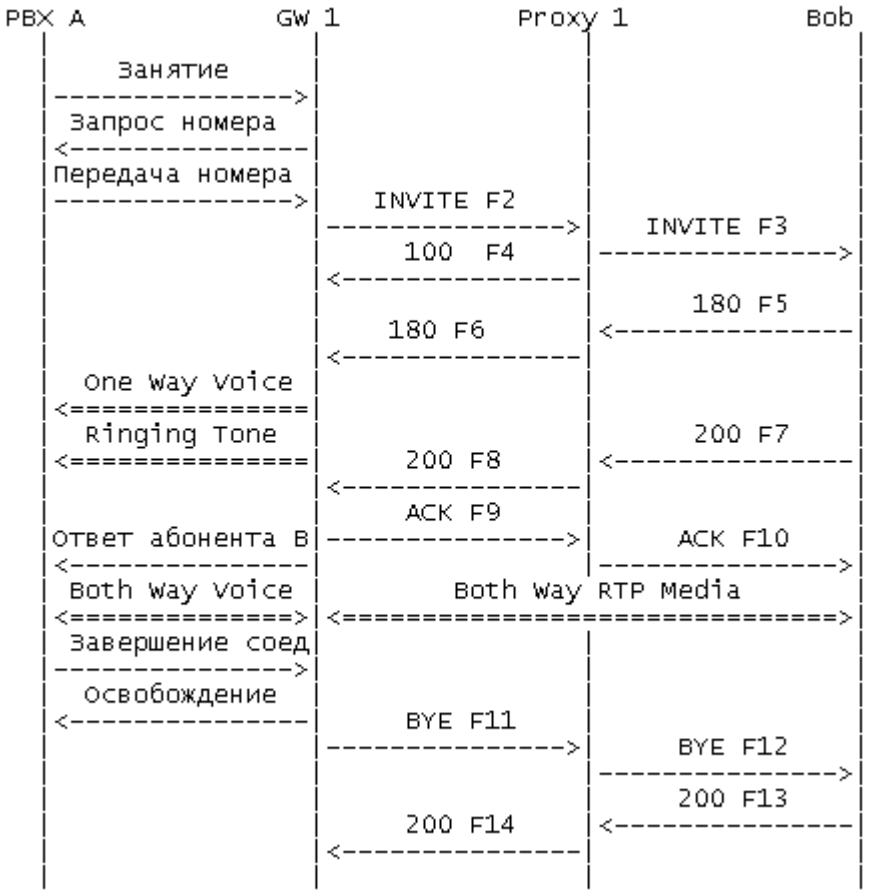


Рисунок 3.22 Диаграмма обмена сообщениями при успешном установлении соединения абонента УПАТС с пользователем сети SIP.

В данном сценарии Maxim вызывает пользователя Anton через УПАТС, GW1 и Proxy 1. УПАТС и шлюз (GW) 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 конкретного протокола, а описаны типы передаваемых сигналов.

Шлюз может опознать только группу линий, по которой пришел звонок и не может идентифицировать линию в УПАТС, по которой пришел вызов. Поэтому в данном примере для идентификации номера вызывающего абонента в SIP URI используется адрес sip:551313@gw1.a.loniis.ru.

Содержимое сообщений

УПАТС -> GW 1

Занятие линии

GW 1 -> PBX (Maxim)

Подтверждение занятия линии

Передача номера вызываемого абонента от УПАТС -> GW 1

F2 INVITE GW 1 -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
Max-Forwards: 70
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:551313@gw1.a.loniis.ru;user=phone>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F3 INVITE Proxy 1 -> Anton

```
INVITE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
 ;received=192.0.2.201
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:551313@gw1.a.loniis.ru;user=phone>
Content-Type: application/sdp
Content-Length: 146

v=0
```

o=GW 2890844527 2890844527 IN IP4 gw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) Proxy 1 -> GW 1

SIP/2.0 100 Trying
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.201
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 180 (Ringing) Anton -> Proxy 1

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.201
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>
Content-Length: 0

F6 180 (Ringing) Proxy 1 -> GW 1

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
 ;received=192.0.2.201
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm

To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>
Content-Length: 0

F7 200 (OK) Anton -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
Contact: <sip:anton@client.b.loniis.ru>
CSeq: 1 INVITE
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F8 200 (OK) Proxy 1 -> GW 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>

Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F9 ACK GW 1 -> Proxy 1

ACK sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F10 ACK Proxy 1 -> Anton

ACK sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Max-Forwards: 69
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F11 BYE GW 1 -> Proxy 1

BYE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65

Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F12 BYE Proxy 1 -> Anton

BYE sip:anton@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
Max-Forwards: 69
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F13 200 (OK) Anton -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F14 200 (OK) Proxy 1 -> GW 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP gw1.a.loniis.ru:5060;branch=z9hG4bKwqwee65
;received=192.0.2.201
From: <sip:551313@gw1.a.loniis.ru;user=phone>;tag=jwdkallkzm
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159

Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

Неуспешное установление соединения из ТфО П в сеть SIP. Пользователь не найден.

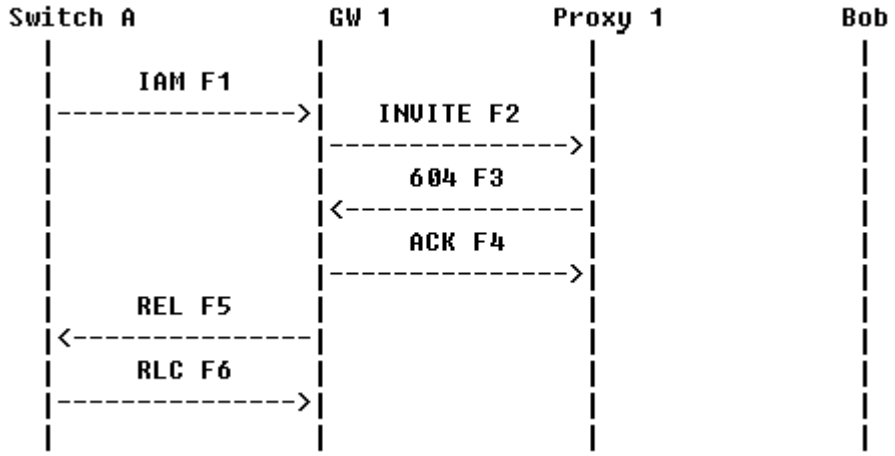


Рисунок 3.23 Диаграмма обмена сообщениями при неуспешном установлении соединения ТфО П – SIP. Пользователь не найден.

Абонент Maxim пытается вызвать пользователя Anton через GW1 и Proxy1. Прокси-сервер не находит пользователя с заданным номером и завершает установление соединения посылая ответ с соответствующим кодом ошибки, который транслируется в сообщение REL для абонента Maxim с соответствующим кодом события.

Содержимое сообщений

F1 IAM ATC A -> GW 1

Получение сообщения IAM
CgPN=095-386-4515,NPI=E.164,NOA=National
CdPN=812-100-2516,NPI=E.164,NOA=National

F2 INVITE Maxim -> Proxy 1

INVITE sip:+78121002516@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@gw1.a.loniis.ru;user=phone>;tag=076342s

To: <sip:+78121002516@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Contact:
<sip:+70953864515@gw1.a.loniis.ru;user=phone;transport=tcp>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

Proxy 1 обращается к серверу местоопределения для получения информации о текущем месте пребывания пользователя с адресом +7-812-100-2516. Пользователь с таким идентификатором отсутствует, поэтому Proxy 1 завершает установление соединения.

F3 604 (Does Not Exist Anywhere) Proxy 1 -> GW 1

SIP/2.0 604 Does Not Exist Anywhere
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@gw1.a.loniis.ru;user=phone>;tag=076342s
To: <sip:+78121002516@ss1.a.loniis.ru;user=phone>;tag=6a34d410
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 INVITE
Error-Info: <sip:does-not-exist@ann.a.loniis.ru>
Content-Length: 0

F4 ACK GW 1 -> Proxy 1

ACK sip:+78121002516@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@gw1.a.loniis.ru;user=phone>;tag=076342s
To: <sip:+78121002516@ss1.a.loniis.ru;user=phone>;tag=6a34d410
Call-ID: 4Fde34wkd11wsGFDs3@gw1.a.loniis.ru
CSeq: 1 ACK

Content-Length: 0

F5 REL GW 1 -> Maxim

Получение сообщения REL с CauseCode=1

F6 RLC Maxim -> GW 1

Получение сообщения RLC

Неуспешное установление соединения из ТфО П в сеть SIP. Линия занята.

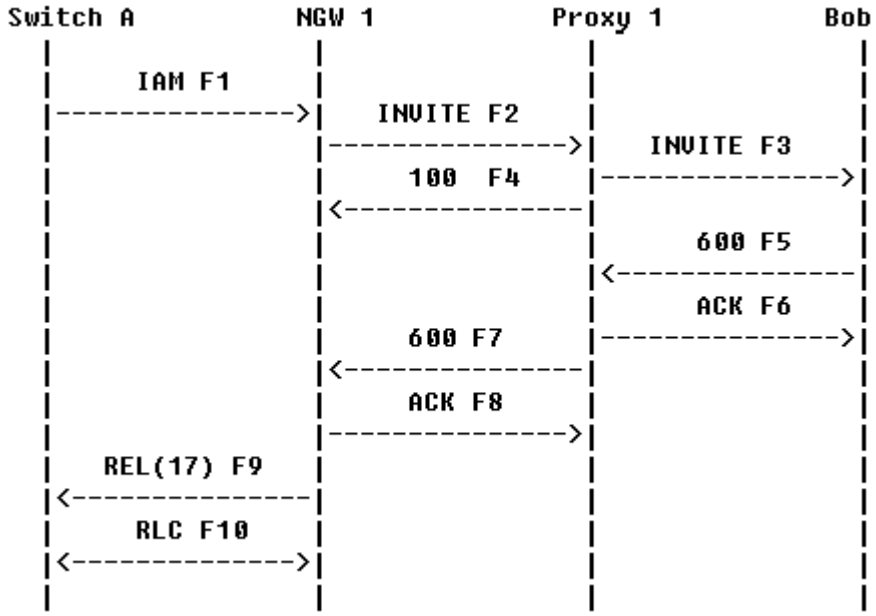


Рисунок 3.24 Диаграмма обмена сообщениями при неуспешном установлении соединения ТфО П – SIP. Линия занята.

В данном сценарии абонент Maxim вызывает пользователя Anton через NGW1 и Proxy1. Прокси-сервер находит пользователя Anton и направляет ему вызов. Терминал пользователя Anton отклоняет вызов, возвращая ответ с кодом 600 (Busy Everywhere). Шлюз посылает сообщение REL, содержащее сод события соответствующий пришедшему ответу.

Поскольку в сообщении IAM (F1) не содержалось параметра Interworking, то сигнал «занято» будет генерироваться из локального узла (например, АТС А). В некоторых сценариях, где указано межсетевое взаимодействие, сигнал «занято» подается из шлюза. Подробнее об этом изложено в [RFC 3398].

Содержимое сообщений

F1 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

F2 INVITE Maxim -> Proxy 1

INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F3 INVITE F3 Proxy 1 -> Anton

INVITE anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) Proxy 1 -> NGW 1

SIP/2.0 100 Trying
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 600 (Busy Everywhere) Anton -> Proxy 1

SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F6 ACK Proxy 1 -> Anton

ACK anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159

Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F7 600 (Busy Everywhere) Proxy 1 -> NGW 1

SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F8 ACK NGW 1 -> Proxy 1

ACK anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F9 REL NGW 1 -> Maxim

Получение сообщения REL с CauseCode=17 Busy

F10 RLC Maxim -> NGW 1

Получение сообщения RLC

Неуспешное установление соединения. Линия занята. IAM содержит параметр interworking.

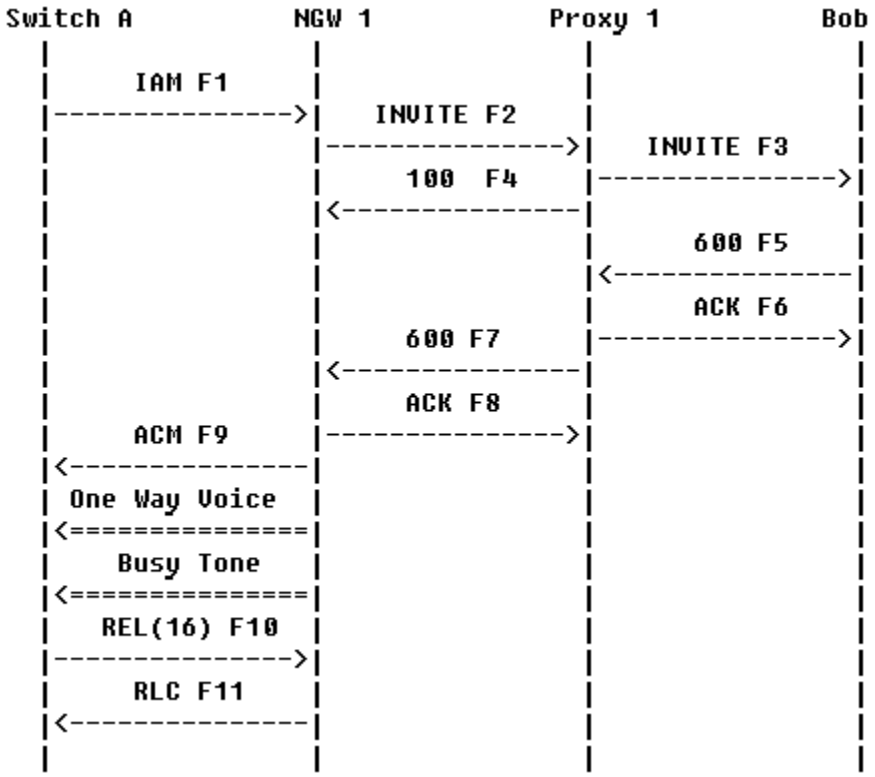


Рисунок 3.25 Диаграмма обмена сообщениями при неуспешном установлении соединения ТфО П – SIP. Линия занята. IAM содержит параметр interworking.

В данном сценарии абонент Maxim вызывает Antona через Network Gateway NGW1 и Proxy 1. Прокси-сервер находит пользователя Anton и направляет к нему вызов. Его терминал отклоняет пришедший вызов, возвращая ответ с кодом ошибки. NGW 1 создает односторонний аудио тракт к абоненту Maxim и выдает ему сигнал «занято». Вызывающий абонент вешает трубку.

Шлюз выдает сигнал «занято», т.к. в сообщении IAM (F1) был указан параметр interworking. В предыдущем сценарии в сообщении REL был указан код сигнала «занято» и он выдавался абоненту ближайшей к нему АТС, т.к. параметр interworking не был указан.

Содержимое сообщений

F1 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
Interworking=encountered

F2 INVITE NGW1 -> Proxy 1

INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2

Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F3 INVITE Proxy 1 -> Anton

INVITE anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) Anton -> Proxy 1

SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 600 (Busy Everywhere) Anton -> Proxy 1

SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F6 ACK Proxy 1 -> Anton

ACK anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F7 600 (Busy Everywhere) Proxy 1 -> NGW 1

SIP/2.0 600 Busy Everywhere
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE

Content-Length: 0

F8 ACK NGW 1 -> Proxy 1

ACK sip:ngw1@a.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F9 ACM NGW 1 -> Maxim

Передается сообщение ACM

NGW1 устанавливает односторонний речевой канал к абоненту Maxim. Соединение разрушается после того, как Maxim вешает трубку.

F10 REL Maxim -> NGW 1

Передается сообщение REL с CauseCode=16

F11 RLC NGW 1 -> Maxim

Передается сообщение RLC

Неуспешное установление соединения. Истекает таймер.

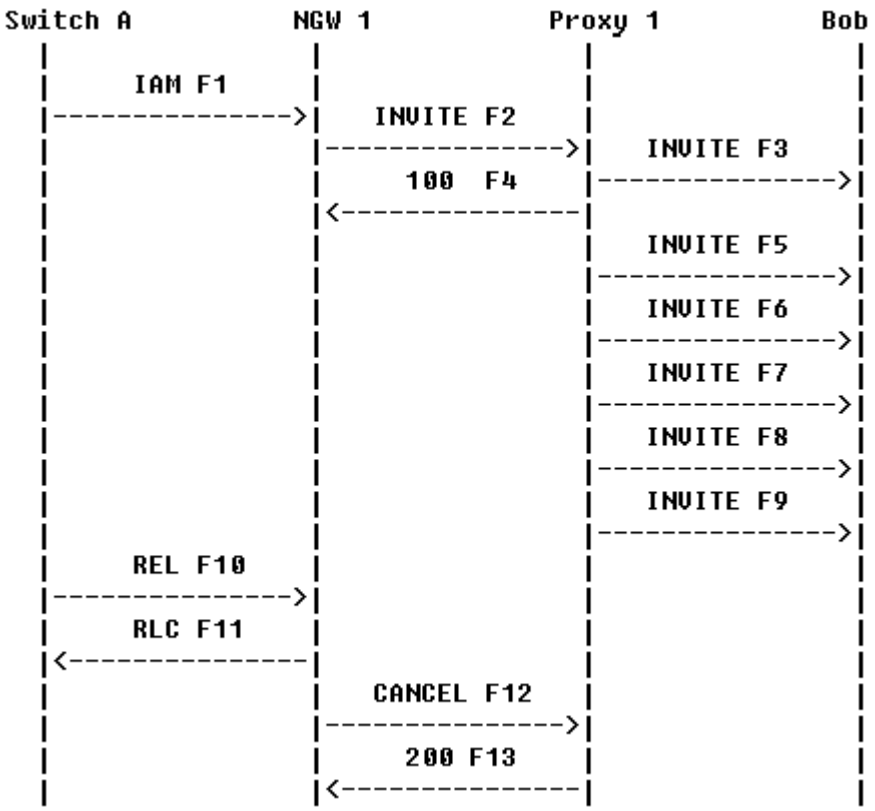


Рисунок 3.26 Диаграмма обмена сообщениями при неуспешном установлении соединения. Истекает таймер.

Абонент Maxim вызывает пользователя Anton через NGW1 и Proxy 1. Сообщение INVITE посылается еще раз, если Anton не ответил на предыдущее до истечения таймера T1. Anton не отвечает на все посланные запросы до истечения таймера в ТфОП. После того, как таймер в ТфОП истекает, то установление соединения прекращается и посылается сообщение REL. Шлюз транслирует это сообщение в CANCEL.

Содержимое сообщений

F1 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

F2 INVITE Maxim -> Proxy 1

INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru

CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F3 INVITE Proxy 1 -> Anton

INVITE sip:anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
c c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) Proxy 1 -> NGW 1

SIP/2.0 100 Trying
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>

Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F6 INVITE Proxy 1 ->Anton

Аналогично сообщению F3

F7 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F8 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F9 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

Истек таймер в ТфОП

F10 REL Maxim -> NGW 1

Передается сообщение REL с CauseCode=16 Normal

F11 RLC NGW 1 -> Maxim

Передается сообщение RLC

F12 CANCEL NGW 1 -> Proxy 1

CANCEL sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals

To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

F13 200 (OK) Proxy 1 -> NGW 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

Неуспешное установление соединения. Истекает таймер. Прокси-сервер работает в режиме без сохранения состояний.

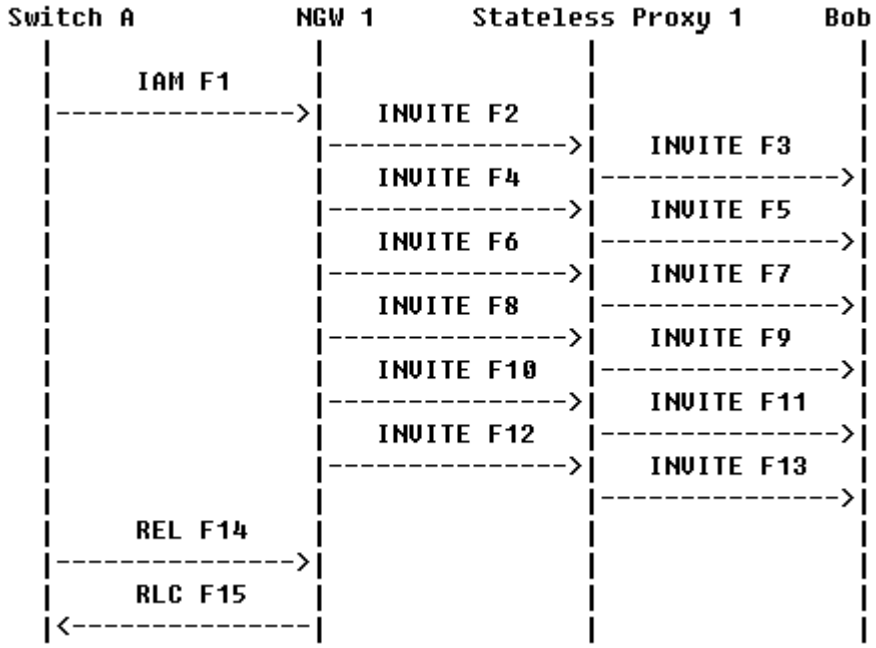


Рисунок 3.27 Диаграмма обмена сообщениями при неуспешном установлении соединения. Истекает таймер. Прокси-сервер режиме без сохранения состояний.

В данном сценарии абонент Maxim вызывает пользователя Anton через NGW1 и Proxy1. Так как прокси-сервер находится в режиме без сохранения состояний, он не использует ответ 100 (Trying). NGW1 повторно посылает INVITE, когда истекает таймер SIP T1. Терминал пользователя Anton не отвечает на запросы. По истечению таймера в сети

ТфОП завершается установление соединения и посылается сообщение REL (CauseCode=102 Timeout).

Содержимое сообщений

F1 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

F2 INVITE NGW 1 -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone  SIP/2.0
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F3 INVITE Proxy 1 -> Anton

```
INVITE sip:anton@b.loniis.ru  SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.201
Max-Forwards: 69
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
```

CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 INVITE NGW 1 -> Proxy 1

Аналогично сообщению F2

F5 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F6 INVITE NGW 1 -> Proxy 1

Аналогично сообщению F2

F7 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F8 INVITE NGW 1 -> Proxy 1

Аналогично сообщению F2

F9 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F10 INVITE NGW 1 -> Proxy 1

Аналогично сообщению F2

F11 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

F12 INVITE NGW 1 -> Proxy 1

Аналогично сообщению F2

F13 INVITE Proxy 1 -> Anton

Аналогично сообщению F3

Истекает таймер в ТфОП

F14 REL Maxim -> NGW 1

Получение сообщения REL с CauseCode=102 Timeout

F15 RLC NGW 1 -> Maxim

Получение сообщения RLC

Неуспешное установление соединения. Вызывающий абонент вешает трубку не дождавшись завершения установления соединения.

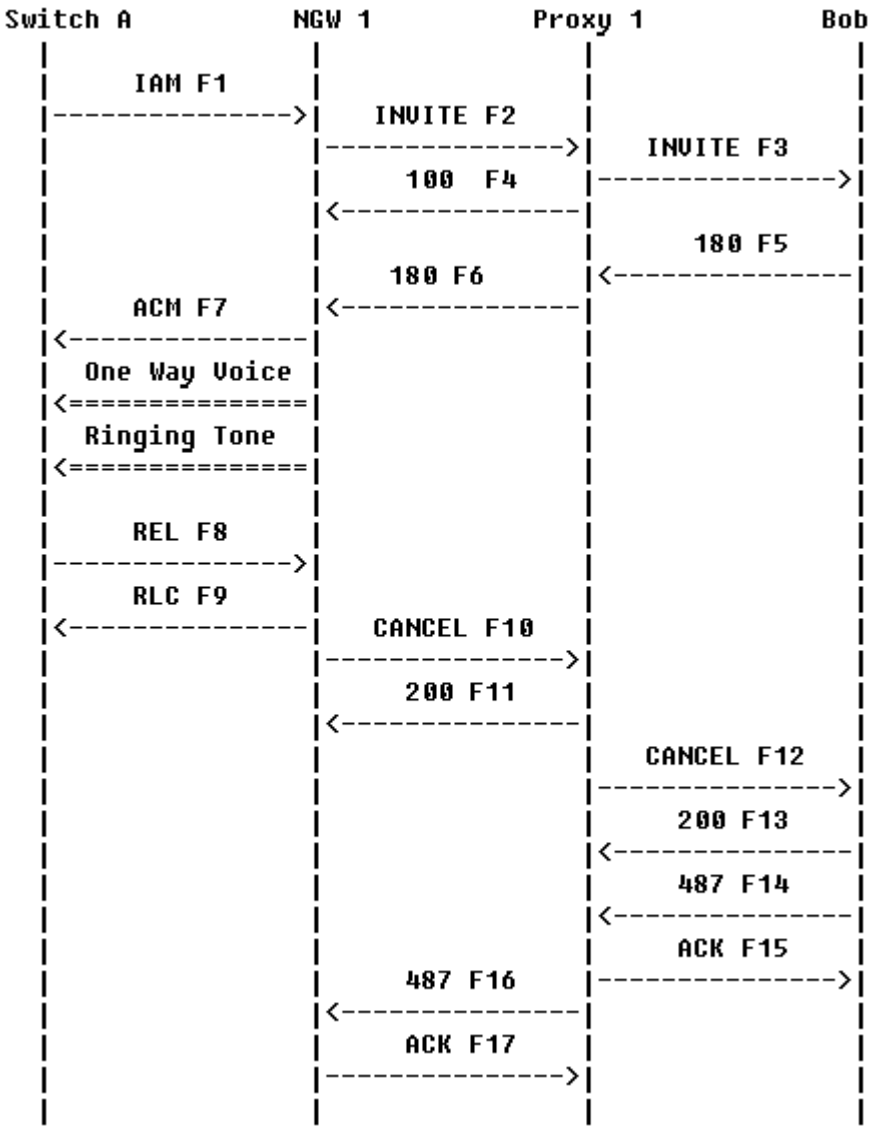


Рисунок 3.28 Диаграмма обмена сообщениями при неуспешном установлении соединения. Вызывающий абонент вешает трубку.

В данном сценарии абонент Maxim посылает вызов пользователю Anton через NGW1 и Proxy1. Вызов достиг вызываемого абонента, но он не снимает трубку. NGW1 проключает односторонний канал и отправляет абоненту Maxim акустический сигнал «контроль посылки вызова», т.к. в сообщении IAM присутствует параметр interworking. Maxim вешает трубку, не дождавшись ответа, после чего отправляет сообщение REL, которое транслируется в сообщение CANCEL. Если терминал пользователя Anton посылает ответ 200 (OK) после того, как пришло сообщение REL, то NGW1 отправит сначала сообщение ACK, а только потом BYE для правильного завершения установления соединения.

Содержимое сообщений

F1 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

F2 INVITE Maxim -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone  SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F3 INVITE Proxy 1 -> Anton

```
INVITE sip:anton@b.loniis.ru  SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
```

```
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F4 100 (Trying) Anton -> Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.201
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0
```

F5 180 (Ringing) Anton -> Proxy 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru;transport=tcp>
Content-Length: 0
```

F6 180 (Ringing) Proxy 1 -> NGW 1

```
SIP/2.0 180 Ringing
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
```

To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:anton@client.b.loniis.ru>
Content-Length: 0

F7 ACM NGW 1 -> Maxim

Передается сообщение ACM

Maxim вешает трубку

F8 REL Maxim -> NGW 1

Передается сообщение REL с CauseCode=16 Normal

F9 RLC NGW 1 -> Maxim

Передается сообщение RLC

F10 CANCEL NGW 1 -> Proxy 1

CANCEL sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

F11 200 (OK) Proxy 1 -> NGW 1

SIP/2.0 200 OK
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

F12 CANCEL Proxy 1 -> Anton

```
CANCEL sip:anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0
```

F13 200 (OK) Anton -> Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0
```

F14 487 (Request Terminated) Anton -> Proxy 1

```
SIP/2.0 487 Request Terminated
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
 ;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0
```

F15 ACK Proxy 1 -> Anton

```
ACK sip:anton@b.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
```

Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F16 487 (Request Terminated) Proxy 1 -> NGW 1

SIP/2.0 487 Request Terminated
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
;received=192.0.2.103
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F17 ACK NGW 1 -> Proxy 1

ACK sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ngw1.a.loniis.ru:5060;branch=z9hG4bKlueha2
Max-Forwards: 70
From: <sip:+70953864515@ngw1.a.loniis.ru;user=phone>;tag=7643kals
To: <sip:+78122625326@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 4Fde34wkd11wsGFDs3@ngw1.a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

3.6.4 Преобразование SIP в ISUP

3.2.4.1 Процесс обмена сообщениями

Следующие примеры обмена сообщениями иллюстрируют типичные случаи установления соединения в сети SIP. На диаграммах не показан ответ на запрос INVITE 100 (Trying), хотя он применяется в большинстве сценариев установления соединения сети SIP.

На диаграммах, представленных ниже, все сигнальные сообщения (SIP и ISUP) проходят через MGC; управление медиапотоком (например, подача аудио данных пользователю, освобождение канала и т.д.) выполняется MG под управлением MGC. Для упрощения на диаграммах они представлены одним узлом, обозначенным MGC/MG.

Установление соединения (быстрый ответ не используется)

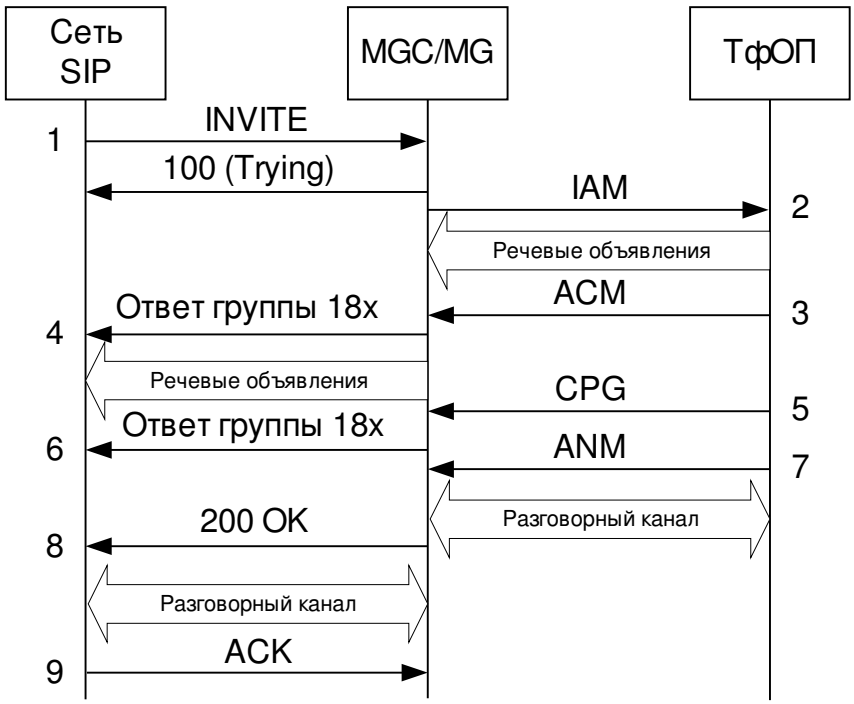


Рисунок 3.29 Диаграмма обмена сообщениями при успешном установлении соединения SIP – ТфОП.

- Когда пользователь SIP хочет начать разговор, терминал SIP посылает запрос INVITE.
- Шлюз получает запрос, переданный терминалом SIP, и преобразует его в сообщение IAM для сети ТфОП, после чего отправляет это сообщение.
- Когда узел ТфОП получает необходимую адресную информацию, он начинает процесс установления соединения с абонентом, а шлюзу отправляется сообщение ACM.
- Параметр called party status (состояние вызываемой стороны) в сообщении ACM преобразуется в один из предварительных ответов SIP и посылается терминалу

пользователя сети. Этот ответ может содержать вложение SDP для установления однонаправленного медиапотока (как показано на диаграмме). Если вложение SDP не содержится, то после шага 8 проключается двухсторонний медиапоток.

- Если используемый вариант ISUP это позволяет, то узел ТфОП может посылать сообщения CGP для индикации того, что вызов пока стоит на обслуживании.
- Если сообщение CGP пришло на шлюз, то он транслирует его в один из предварительных ответов и направляет его к терминалу пользователя.
- Как только абонент ТфОП ответит на вызов (поднимет трубку) шлюзу будет отправлено сообщение ANM.
- После того, как будет получено сообщение ANM, шлюз отправит ответ 200 (OK) пользователю SIP.
- В ответ на это терминал пользователя SIP отправит сообщение ACK как принятие всех параметров и успешного завершения процесса установления соединения.

Установление соединения (используется быстрый ответ)

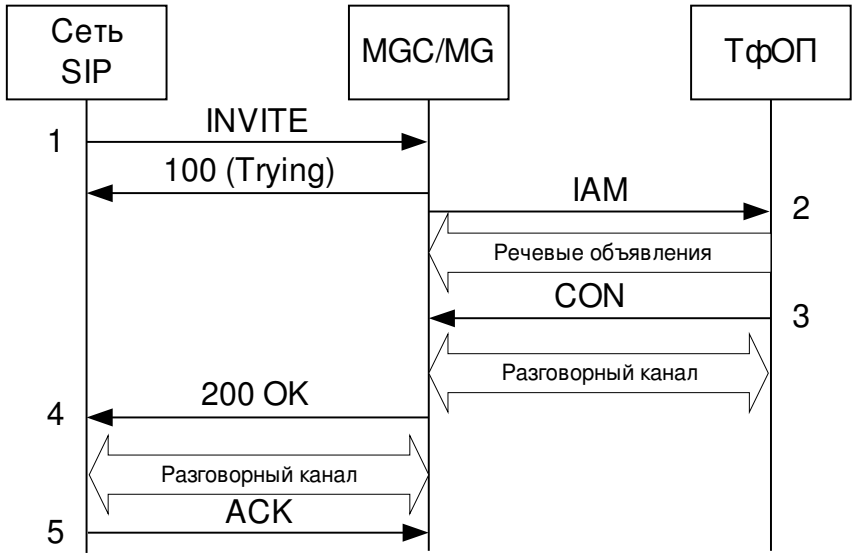


Рисунок 3.30 Диаграмма обмена сообщениями при успешном установлении соединения SIP – ТфОП. Используется быстрый ответ.

Этот сценарий не поддерживается в версии ISUP ANSI

1. Когда пользователь SIP хочет начать соединение с абонентом ТфОП терминал посылает запрос INVITE.
2. Шлюз принимает запрос INVITE и транслирует его в сообщение IAM, которое отправляется в ТфОП.
3. Поскольку удаленный узел в сети ТфОП сконфигурирован на авто-ответ, он посылает сообщение CON в ответ на пришедшее сообщение IAM.
4. После получения сообщения CON шлюз отправляет ответ с кодом 200 терминалу SIP.
5. После получения этого ответа терминал SIP считает соединение установленным и, подтверждая все параметры, посылает сообщение ACK.

Истечение таймера Т7

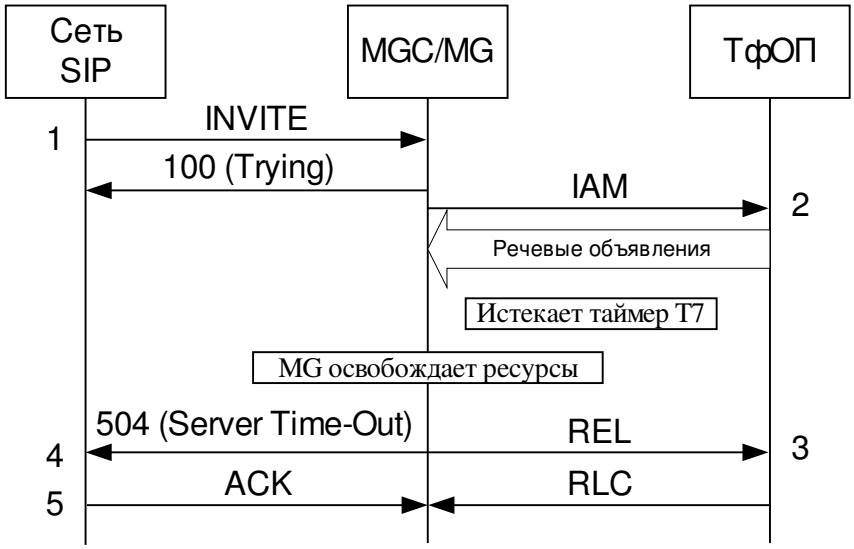


Рисунок 3.31 Диаграмма обмена сообщениями при истечении таймера Т7 сети ТфОП.

- Когда пользователь SIP хочет установить соединение с абонентом ТфОП, его терминал посылает запрос INVITE.
- Шлюз принимает запрос INVITE и транслирует его в сообщение IAM, которое отправляется в ТфОП. С этого момента начинается отсчет таймера Т7.
- Таймер истекает прежде, чем приходит сообщение ACM или CON и поэтому к шлюзу отправляется REL.
- От шлюза к терминалу SIP отправляется ответ 504 (Version Not Supported) с кодом ошибки истечения таймера.
- Получив ответ, терминал SIP отправляет подтверждение приема ACK.

Истечение таймеров SIP

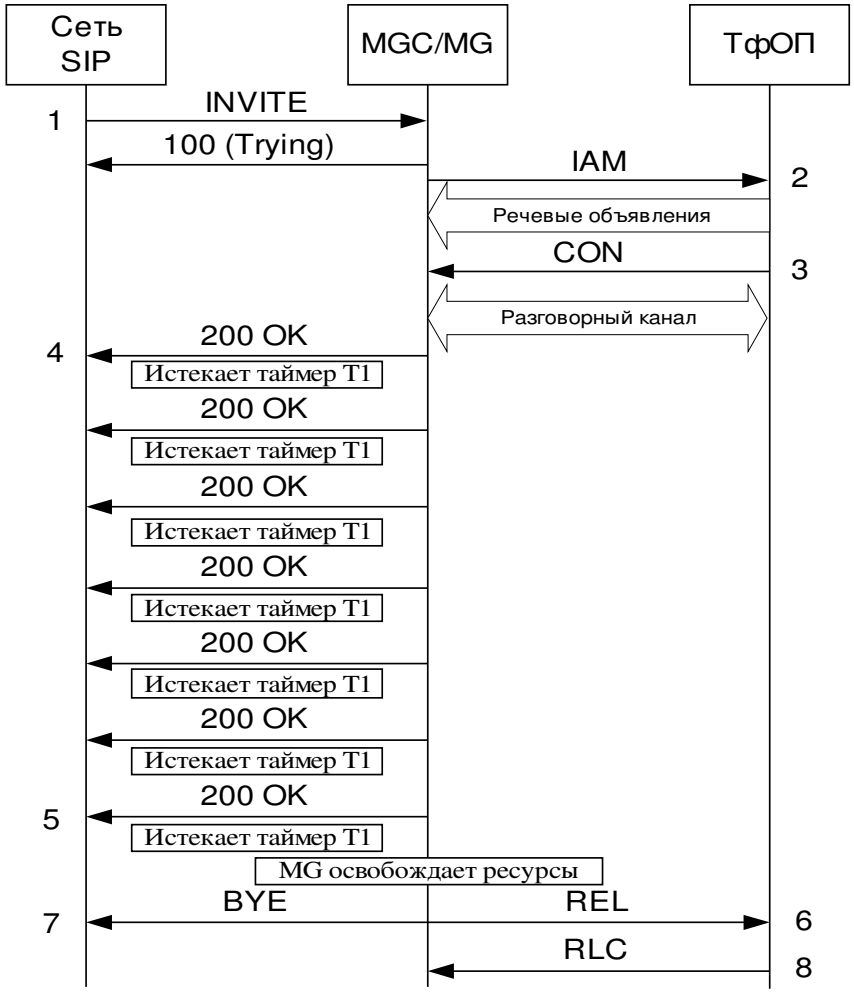


Рисунок 3.32 Диаграмма обмена сообщениями при истечении таймера T1 сети SIP.

2 Когда пользователь SIP хочет установить соединение с абонентом ТфОП, терминал посылает запрос INVITE.

3 Шлюз принимает запрос INVITE и транслирует его в сообщение IAM, которое отправляется в ТфОП.

4 Поскольку узел в сети ТфОП сконфигурирован на быстрый ответ, он посылает сообщение CON, как только получит IAM. В варианте ISUP ANSI вместо CON посылается ANM (без ACM).

5 После получения сообщения CON шлюз посылает терминалу SIP ответ 200 (OK) и начинает отсчет таймера T1.

6 Ответ пересылается повторно всякий раз, кода истекает таймер.

7 После семи повторений установление соединения завершается, узлу ТфОП посылается сообщение REL с кодом события 102 (recover on timer expiry).

8 Терминалу SIP передается сообщение BYE для завершения процедуры установления соединения. Дальнейший процесс завершения соединения не показан, поскольку не известно состояние терминала SIP.

9 На сообщение REL узел ТфОП отвечает сообщением RLC.

Ошибка установления соединения на стороне ТфОП

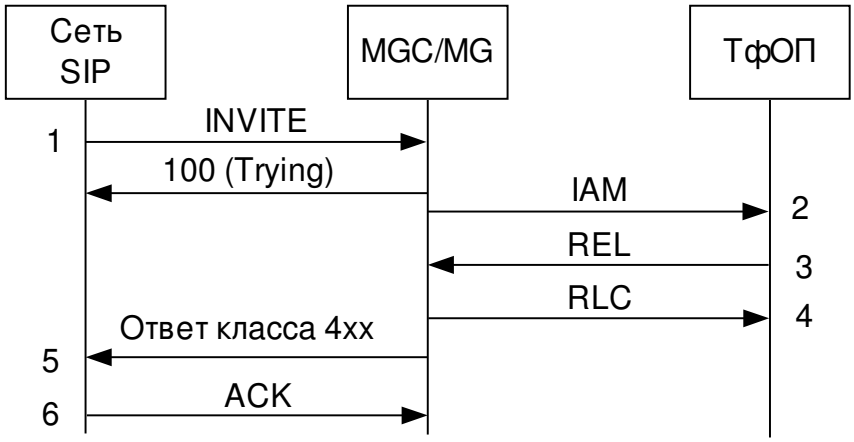


Рисунок 3.33 Диаграмма обмена сообщениями при успешном установлении соединения SIP – ТфОП.

- Когда пользователь SIP хочет начать соединение с абонентом ТфОП терминал посылает запрос INVITE.
- Шлюз принимает запрос INVITE и транслирует его в сообщение IAM, которое отправляется в ТфОП.
- Поскольку узел ТфОП не может установить соединение, он посылает шлюзу сообщение REL.
- Шлюз подтверждает сообщение об отмене установления соединения сообщением RLC и освобождает занятые ресурсы.
- Терминалу SIP направляется ответ с кодом ошибки, соответствующей коду события в сообщении REL.
- Терминал SIP подтверждает завершение процедуры установления соединения сообщением ACK.

В сообщении ACM содержится код события

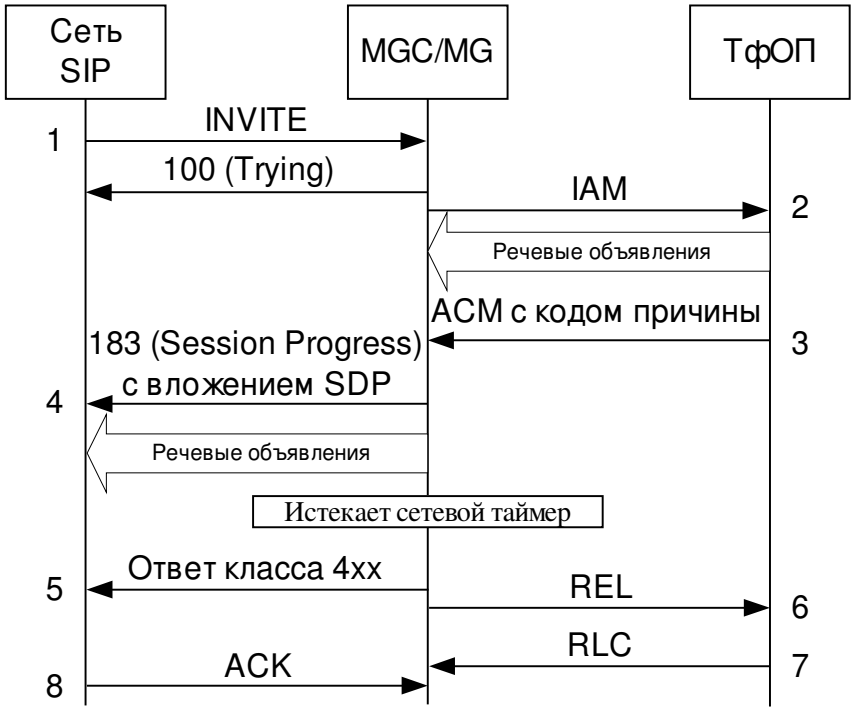


Рисунок 3.34 Диаграмма обмена сообщениями. В сообщении ACM содержиться код события.

- Когда пользователь SIP хочет начать соединение с абонентом ТфОП терминал посылает запрос INVITE.
- Шлюз принимает запрос INVITE и преобразует его в сообщение IAM, которое отправляется в ТфОП.
- Поскольку узел ТфОП не может установить соединение (например, абонент находится вне зоны действия сети), он посылает шлюзу сообщение ACM с соответствующим кодом события. Шлюз запускает таймер.
- После получения ACM с кодом события (содержащимся в параметре CAI) шлюз посылает к терминалу SIP ответ с кодом 183 (Session progress), содержащий вложение SDP для создания одностороннего аудио канала на непродолжительное время.
- Генерируется основанный на коде события заключительный ответ на INVITE и отправляется терминалу SIP для прекращения процедуры установления соединения.
- По истечению таймера узлу ТфОП посылается сообщение REL для прерывания установления соединения. Однако если пользователь SIP прервет установление соединения раньше, чем истечет таймер (терминал пошлет сообщение CANCEL), то процесс обмена сообщениями будет соответствовать следующему сценарию.
- На полученное сообщение REL узел ТфОП отвечает сообщением RLC.
- Терминал SIP посылает подтверждение АСК, завершая процесс обмена сообщениями.

Пользователь SIP прерывает установление соединения

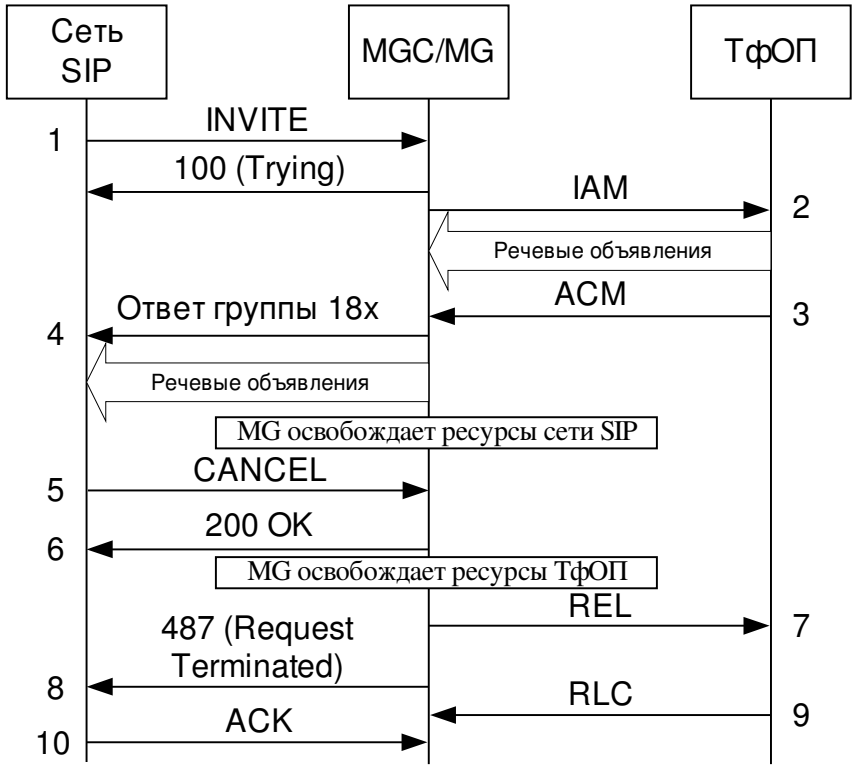


Рисунок 3.35 Диаграмма обмена сообщениями. Вызывающ ий пользователь вешает трубку.

11 Когда пользователь SIP хочет начать соединение с абонентом ТфОП терминал посылает запрос INVITE.

12 Шлюз принимает запрос INVITE и транслирует его в сообщение IAM, которое отправляется в ТфОП.

13 Когда узел ТфОП получит необходимую адресную информацию, он начнет процесс установления соединения с абонентом, а шлюзу отправится сообщение ACM.

14 Параметр called party status (состояние вызываемой стороны) в сообщении ACM преобразуется в один из предварительных ответов SIP и посылается терминалу пользователя сети. Этот ответ может содержать вложение SDP для установления однонаправленного мультимедийного потока.

15 Для обрыва установленного соединения до того, как абонент ТфОП ответит на вызов, терминалом SIP посылается сообщение CANCEL.

16 Это сообщение подтверждается ответом 200 (OK).

17 После получения CANCEL шлюз отправляет узлу ТфОП сообщение REL.

18 Шлюз посылает ответ с кодом 487 (Request Terminated) терминалу SIP, как сигнал о завершении обработки запроса INVITE.

19 На пришедшее сообщение REL узел ТфОП отвечает сообщением RLC.

20 Терминал SIP подтверждает прием ответа 487 (Call Cancelled) сообщением ACK.

3.6.4.2 Конечные автоматы SIP-T при взаимодействии SIP-ISUP

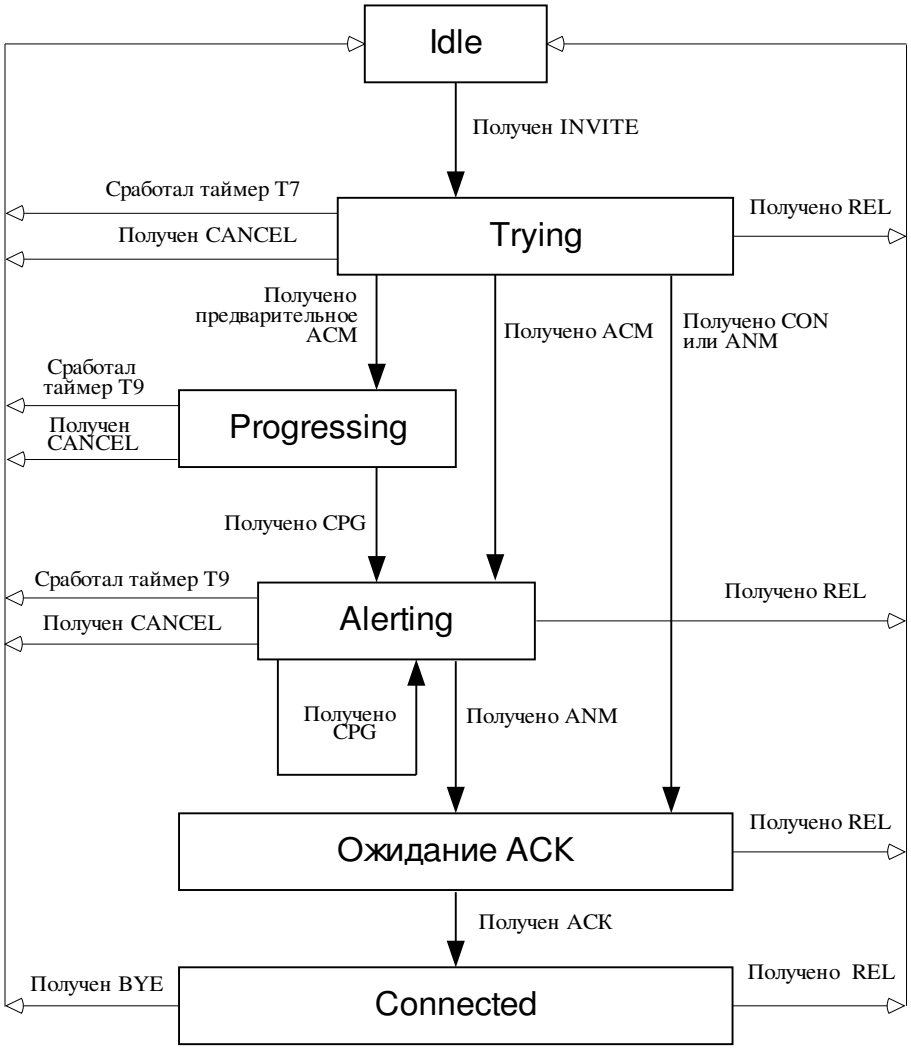


Рисунок 3.36 Конечные автоматы SIP-T при взаимодействии SIP-ISUP.

Получение запроса INVITE

При получении запроса INVITE шлюз может послать терминалу SIP ответ с кодом 100 (Trying), для индикации того, что вызов обслуживается.

Так же при получении запроса шлюз должен зарезервировать ресурсы для создания мультимедийного потока, поскольку сообщение IAM не может быть послано до этого. Обычно ресурсами являются тайм-слот в первичном цифровом потоке E1/T1 и RTP/UDP порт на стороне сети SIP. Для резервирования ресурсов могут использоваться любые процедуры обеспечения заданного качества обслуживания.

После того, как будет послано сообщение IAM шлюз запускает таймер T7.

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 последний параметр, которой указывает на 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 минимум пять этих параметров.

Таким образом, любой шлюз должен уметь транслировать заголовки пришедшего запроса INVITE в значения этих параметров. Значения большинства параметров (таких как NCI и USI) шлюз создает самостоятельно, исходя из параметров устанавливаемого соединения. Остальные, такие как CPN, заполняются основываясь на информации, пришедшей в запросе INVITE.

Так же в сообщении IAM может содержаться ряд необязательных параметров. При трансляции сообщений из SIP в ISUP их рекомендуется использовать, но можно не учитывать. Как было отмечено ранее, трансляция позволяет аппаратуре сети SIP понимать содержимое сообщений узлов ТфОП, если инкапсулированное сообщение ISUP отсутствует или его невозможно извлечь. Параметры, которые важны только для ТфОП при сценарии ТфОП-SIP-ТфОП инкапсулируются в тело запросов SIP, транслировать их значение не обязательно. Из всех необязательных параметров транслироваться могут 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, то он должен использовать для формирования IAM вложенное сообщение ISUP, если оно существует. Если это возможно, то шлюз при создании нового IAM должен попытаться использовать значения параметров из вложенного сообщения. В некоторых случаях шлюз не может использовать инкапсулированное сообщение – например, не может обработать тот вариант ISUP вложенного сообщения и поэтому вынужден его игнорировать. Даже если использование инкапсулированного сообщения возможно, то 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 над значениями параметров инкапсулированного сообщения. Это позволяет вводить множество услуг в сети SIP, например, таких как различные варианты переадресации вызовов.

Например, пришедшее на шлюз сообщение INVITE содержит инкапсулированный IAM с параметром CPN, содержащим телефонный номер +78125332699, а поле Request-URI сообщения INVITE содержит *tel:+70955550110*. Тогда шлюз при создании нового сообщения IAM для ТфОП должен использовать телефонный номер, содержащийся в Request-URI, а не тот, который находиться в поле CPN вложенного IAM. Более подробное описание транслирование заголовков SIP параметры ISUP описано ниже.

Шлюз должен иметь прописанные заранее параметры по умолчанию для обязательных параметров ISUP и использовать их в тех случаях, когда значение параметра не может быть получено трансляцией (из сообщения SIP) (например такие параметры 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 трансляции, если используются механизмы переносимости номера, описанние которых

следует ниже.

Первым шагом трансляции полей и заголовков сообщения INVITE в параметры IAM является проверка поля Request-URI.

Если шлюзом поддерживается услуга переносимости номера, тогда следующие шаги связаны с обработкой параметров «npdi» и «rn» поля Request-URI. Подробнее о процедурах услуг переносимости номера см. [Number Portability in the Global Switched Telephone Network (GSTN): An Overview, RFC 3482].

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 переносимости номера.

Если в дополнение к записи npdi=yes не существует поля «rn», тогда основной телефонный номер в tel URL (цифры номера следуют непосредственно за «tel:») 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», тогда телефонный номер из параметра «rn» должен быть преобразован в формат ISUP и занесен в значение параметра CPN. Основной телефонный номер в tel URL также 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 сепаратором) и комбинированный результат используется как значение параметра CPN.

Все последующие обязательные при трансляции действия выполняются после проверки использования услуги переносимости номера и разбора параметров, связанных с этой функцией.

Если переносимость номера не поддерживается шлюзом, тогда 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). Подробнее о поддержке такого типа маршрутизации можно узнать в [RFC 3398].

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 возвращен ответ с кодом ошибки.

Исключением из этого правила является случай, когда заголовок From не содержит телефонного номера. Такое может быть, если вызов производился с телефона в сети SIP, а тогда адрес отправителя обычно имеет вид user@host. Тогда при создании нового сообщения IAM параметр CIN должен быть опущен. Шлюз также может использовать альтернативные не стандартизированные процедуры для трансляции SIP URI в телефонный номер.

Когда шлюз получает запрос с вложенным проверенным и доступным для обработки сообщением ISUP, то он должен установить в генерируемом IAM индикатор FCI в соответствии с тем, какие значения имеют все биты связанные с взаимодействием во вложенном сообщении ISUP. В большинстве случаев встречается значение индикаторов ‘no interworking’. Если в запросе, пришедшем на шлюз, нет доступного для обработки вложенного сообщения ISUP, при создании нового IAM рекомендуется устанавливать бит Interworking Indicator параметра FCI в значение 'no interworking', а также ISDN User Part Indicator в значение ‘ISUP used all the way’; шлюз также может установить параметр Originating Access Indicator в значение ‘Originating access non-ISDN’.

Когда в параметре FCI установлено значение 'interworking encountered', то это показывает, что сеть ISUP взаимодействует с сетью, которая не поддерживает большинство из услуг, существующих в сетях ISUP. Поэтому сеть ISUP вынужденно ограничивает различные сервисы, которые она обычно использует, включая использование кодов событий [cause code] при неудачном установлении соединения. Если необходимо, производители могут снабдить шлюз настраиваемыми опциями, используемыми по усмотрению сервис провайдерами в том случае, когда в параметре FCI указано взаимодействие, а вложенное сообщение отсутствует или недоступно.

Истечение таймера ISUP T7

Так как узел ТфОП не отвечает в течение установленного времени, MG освобождает свои ресурсы. Терминалу SIP отправляется ответ 504 (Server Timeout). Узлу ТфОП отправляется сообщение REL с кодом события 102 (ошибка протокола, основанная на истечении таймера). Шлюз может ожидать ответа из ТфОП сообщением RLC и ответа от сети SIP сообщением ACK, которые подтверждают освобождение ресурсов.

Получение сообщения CANCEL или BYE

Если запросы CANCEL и BYE были получены до того, как произошло установление соединения, то в сеть SIP должен быть отправлен ответ 200 (ОК) для подтверждения приема запроса CANCEL или BYE; также должен быть послан ответ с кодом 487 (Request Terminated) для прекращения передачи запросов INVITE. Далее все ресурсы освобождаются, и узлу ТфОП отправляется сообщение REL с кодом события 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 транслируется в код события 16 (normal clearing), в некоторых случаях код события в инкапсулированном REL может быть

другим. Поэтому параметр Cause Indicator из вложенного сообщения должен быть использован в сообщении REL, отправляемом узлу ТфОП.

Запросы CANCEL и BYE могут содержать заголовок Reason, значение которого должно транслироваться 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 отправит сообщение REL

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 отправлено сообщение RLC.

Если запрос INVITE, который инициировал сессию, содержал проверенное и доступное для обработки сообщение ISUP, тогда соответствующее вложенное сообщение ISUP должно быть отправлено в ответе на данный INVITE. Следовательно, как только сообщение REL приходит на шлюз его нужно включить в тело ответа SIP. Шлюз не должен возвращать ответ с инкапсулированным сообщением ISUP, если инициатор создания сессии не использовал вложенные сообщения ISUP в запросе INVITE.

Получение некоторых эксплуатационно-управляющих сообщений ISUP в ответе на IAM такие как Blocking Message (BLO), Reset Message (RSC) или их эквиваленты может привести к прекращению установления соединения на данном этапе.

Трансляция кодов событий ISDN в коды ответов сети SIP

Сообщение REL в сети ОКС №7 выполняет множество функций и поэтому имеет громоздкую структуру, создающую трудности при 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 ответов. В сети ISUP сообщение REL может быть послано для прекращения установления соединения (BYE в сети SIP), для отмены посланного ранее запроса, пока его обработка не завершена (CANCEL в сети SIP), или отклонить запрос установки соединения, который был недавно получен (различные типы ответов в сети SIP).

Не все коды событий ISUP нужно транслировать 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 был послан 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. данная ситуация может возникнуть только в шлюзе.

Следовательно, при получении кода состояния ISUP 44 (Request circuit or channel not available) трансляция в SIP производиться не будет.

Если значение кода события отличается от приведенных ниже, то используется ответ по умолчанию 500 (Server internal error).

В дополнение с коду события ISDN используется параметр CAI, содержащий причину location, которая показывает, в какой сети произошло прерывание вызова. В большинстве случаев причина location не влияет на трансляцию в коды состояния SIP; некоторые исключения отмечены ниже. В кодах причины ISDN также может присутствовать поле диагностики, которое содержит дополнительную информацию, имеющую отношение к прерыванию вызова.

Рекомендуются следующие варианты трансляции:

Таблица 3.9

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 события ISUP 16 (normal call clearing) обычно транслируется в запросы BYE или CANCEL

(+) – Если значение причины location является ‘user’, тогда вместо ответа класса 4xx выдается ответ класса 6xx (вместо ответа 403 – ответ 603)

(=) – Процедура ANSI – в сети ANSI коду события 26 присваивается значение misrouted ported number. Предполагается, что в основной сети была использована услуга переносимости номера. В остальных случаях код события 26 обычно не используются процедурами ISUP.

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 данный номер должен быть добавлен в заголовок Contact ответа SIP (301).

Ресурсы недоступны

Этот вид событий указывает на временный отказ в обслуживании, так как все ресурсыв данный момент задействованы. Поэтому при трансляции к ответу SIP может быть добавлен заголовок Retry-After.

Таблица 3.10

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

Сервис или опция недоступны.

Этот вид событий указывает, что имеются временные проблемы при обработке запроса, которые оборудование самостоятельно устранит через какое-то время.

Таблица 3.11

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

Сервис или опция недоступны

Таблица 3.12

Код причины ISUP	Ответ SIP
65 bearer capability not implemented	488 Not Acceptable Here
70 only restricted digital avail	488 Not Acceptable Here
79 service or option not implemented	501 Not implemented

Неверное сообщение

Таблица 3.13

Код причины ISUP	Ответ SIP
87 user not member of CUG	403 Forbidden
88 incompatible destination	503 Service unavailable

Ошибка протокола

Таблица 3.14

Код причины ISUP	Ответ SIP
102 recovery of timer expiry	504 Gateway timeout
111 protocol error	500 Server internal error

Взаимодействие с другими сетям

Таблица 3.15

Код причины ISUP	Ответ SIP
127 interworking unspecified	500 Server internal error

Получение предварительного ответа ACM

Сообщение ACM посылается в тех случаях, когда необходимо показать, что вызов еще обрабатывается и перезапустить таймеры ISUP, при этом вызов не попадает в состояние «Alerting». Такой метод применяется, например, в мобильных сетях, где перемещение пользователя может создавать большие задержки при установлении соединения. Предварительное ACM посылается перед тем, как вызов 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 установлено в 00 (no indication).

После того, как было послано предварительное ACM, сеть ISUP ожидает сообщения CPG, чтобы перевести вызов в следующую стадию.

Когда шлюз получает предварительное ACM, то он должен отправить ответ 183 (Session Progress) в сеть SIP. В сценарии, когда сеть SIP является транзитной предварительное ACM должно быть также включено в тело ответа.

Отправка ответа 183 (Session Progress) до того, как было получено сообщение о приеме всего номера (ACM) создает проблемы при транзите трафика через сеть SIP (SIP – bridging), и, следовательно, не должена производится.

Получение сообщения ACM

В большинстве случаев при получении сообщения ACM шлюз должен послать предварительный ответ (из группы 18x) в сеть SIP. Если запрос INVITE, полученный от инициатора сессии, содержал проверенное и доступное для обработки инкапсулированное сообщение ISUP, то ACM, полученное шлюзом должно быть присоединено к предварительному ответу, отправляемому в сеть SIP.

Если ACM содержит параметр Backward Call Indicators со значением ‘subscriber free’, то шлюз должен оправить ответ 180 (Ringing). Если до этого ответа не проключались разговорные тракты, то все акустические сигналы выдаются агентом пользователя (SIP user

agent) в терминале. Шлюз также может выдавать акустические сигналы.

Если параметр Backward Call Indicators (BCI) в сообщении ACM показывает, что на пути вызова встречается межсетевое взаимодействие (обычно в ACM указывается взаимодействие с более простой сетью, которая не подерживат сигнализацию по выделенному каналу), тогда может быть использована внутриканальная передача служебных сигналов о состоянии вызова (‘занято’ и др). Также, если это возможно, должен быть создан односторонний разговорный канал в обратном направлении. Такой канал также 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 каналы, шлюз должен послать ответ с кодом 183 (Session Progress) в сеть SIP.

При получении сообщения ACM коммутаторы в большинстве сетей ISUP запускают таймер T9, длительность которого обычно варьируется от 90 секунд до 3 минут. В случае, когда созданы односторонние разговорные каналы и передаются речевые объявления или служебные сигналы в предответном состоянии, этот таймер определяет, какое количество времени вызывающий пользователь может прослушивать подаваемую ему информацию (например, КПВ или речевые сообщения) от удаленного узла ISUP без начисления оплаты за соединение. КПВ и голосовые сообщения генерирует ближайшая к пользователю АТС ТфОП. Если необходимо, чтобы голосовые сообщения проигрывались дольше длительности таймера T9, то сеть посылает сообщение ANM, которое инициирует создание двухстороннего канала на неопределенный промежуток времени. Обычно в сетях ISUP процедуры биллинга начинают использовать после получением сообщения ANM. Некоторые сети могут не поддерживать таймер T9.

Получение сообщений CON или ANM

Когда шлюз получает сообщение CON или ANM это означает, что вызываемый абонент ответил на звонок и в сеть SIP должен быть отправлен ответ 200 (ОК). В сценарии, когда сеть SIP используется как транзитная (в данном случае запрос INVITE инициатора сессии содержит разрешенное и доступное вложение) то пришедшее сообщение CON или ANM должно быть инкапсулировано в ответ 200 (ОК), отсылаемый в сеть SIP. После этого должен быть установлен двухсторонний медиапоток, если этого не было сделано заранее.

Когда межсетевое взаимодействие ведется с сетями того же класса, что и сеть 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 отправлен предварительный ответ группы 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).

Истечение таймера T9

Данный таймер может использоваться не во всех сетях. Его истечение показывает, что ANM для этого вызова не был получен в течении длительного времени (с момента передачи ACM). Обычно это означает, что терминал вызываемого абонента получал сигнал ПВ длительное время, но не ответил. Также это может происходить в случае межсетевого взаимодействия, когда сеть передает информацию о статусе абонента (например, абонент находиться вне зоны действия сети) повторяя её несколько раз. Любая причина задержки вызова в этом состоянии до истечения этого таймера приводит к тому, что обработка запроса прекращается. Все ресурсы шлюза, связанные с данным мультимедийным потоком освобождаются. В сеть SIP должен быть отправлен ответ 480 (Temporarily Unavailable), а в сеть ТфОП - сообщение REL с кодом события 19 (no answer from the user). Шлюз ожидает сообщения RLC из сети ТфОП и запроса ACK из сети SIP в подтверждение освобождения ресурсов.

Получение сообщения CPG

CPG – это предварительное сообщение, посылаемое в следующих случаях:

- вызов поставлен на обслуживание;
- передается сигнал КПВ;
- передаются речевые объявления.

Если CPG показывает, что идет передача речевых объявлений, то шлюз должен установить односторонний аудио канал в обратном направлении и начать передачу.

При транзите трафика через сеть SIP, сообщение CPG должно быть инкапсулировано в ответ группы 18x, отправляемый в сеть SIP и определяемый по коду события CPG с помощью следующей таблицы:

Таблица 3.16

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

Если сообщение CPG не показывает, что вызываемому абоненту подается ПВ, то текущее состояние вызова не изменяется.

Получение ACK

В этом состоянии соединение уже полностью установлено и имеет место разговорная фаза. При получении данного запроса в сеть ISUP ничего не отправляется.

3.6.4.3 SDL-диаграммы приема и передачи сообщений при взаимодействии SIP – ТфОП

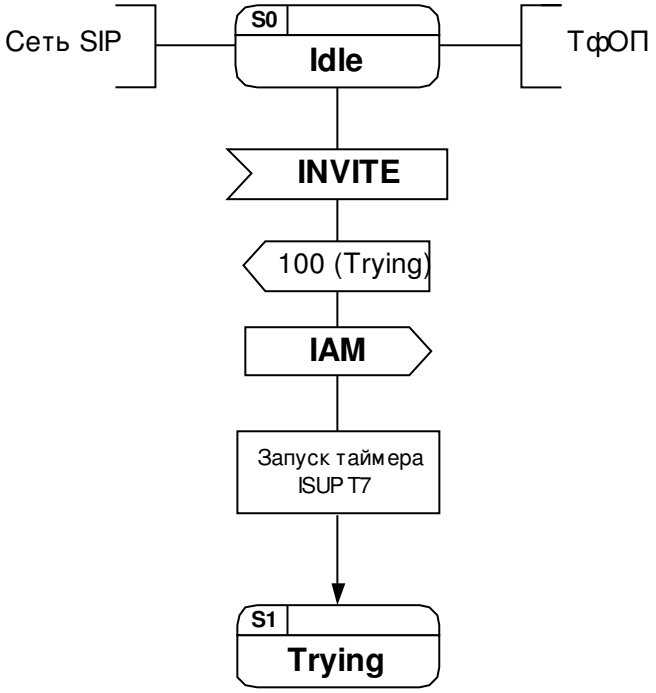


рисунок 3.37 Переход в состояния «Trying»

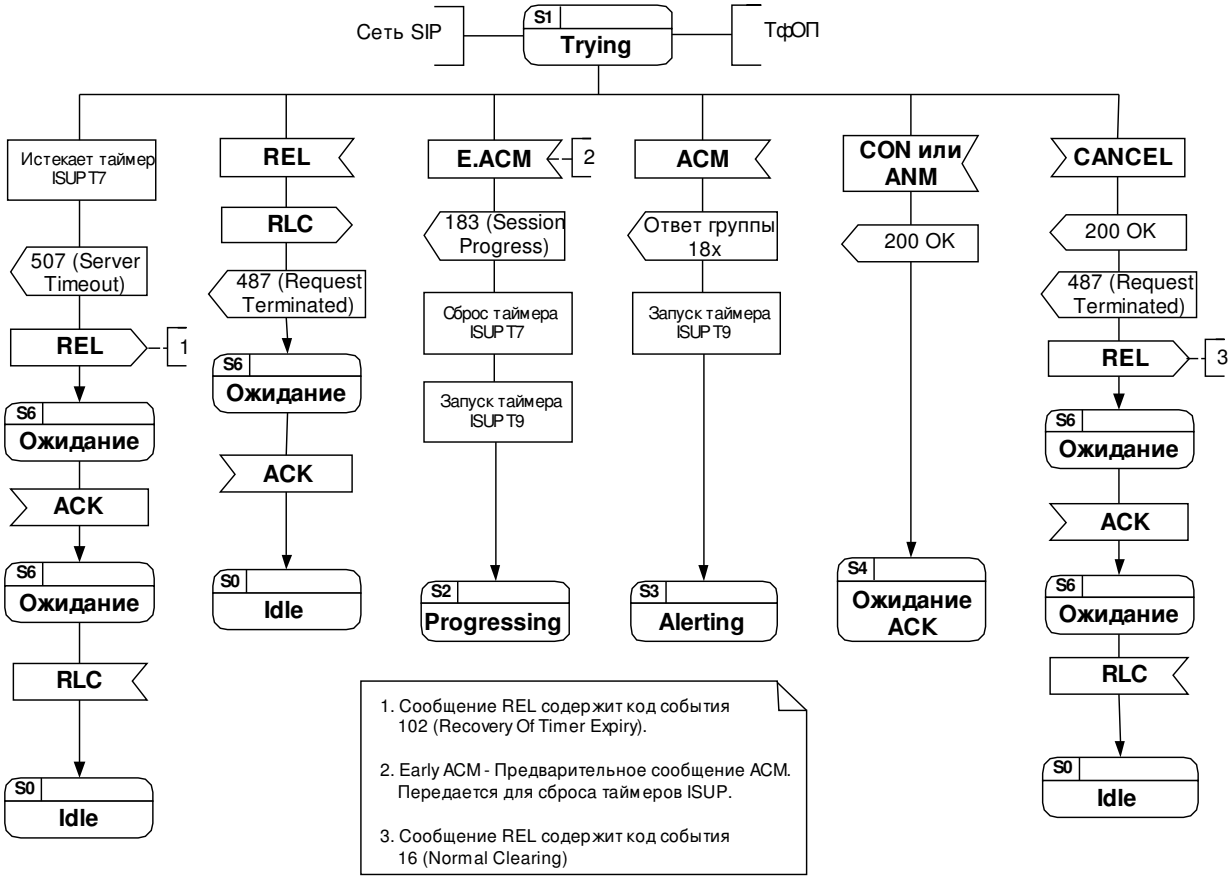


рисунок 3.38 Переходы из состояния «Trying»

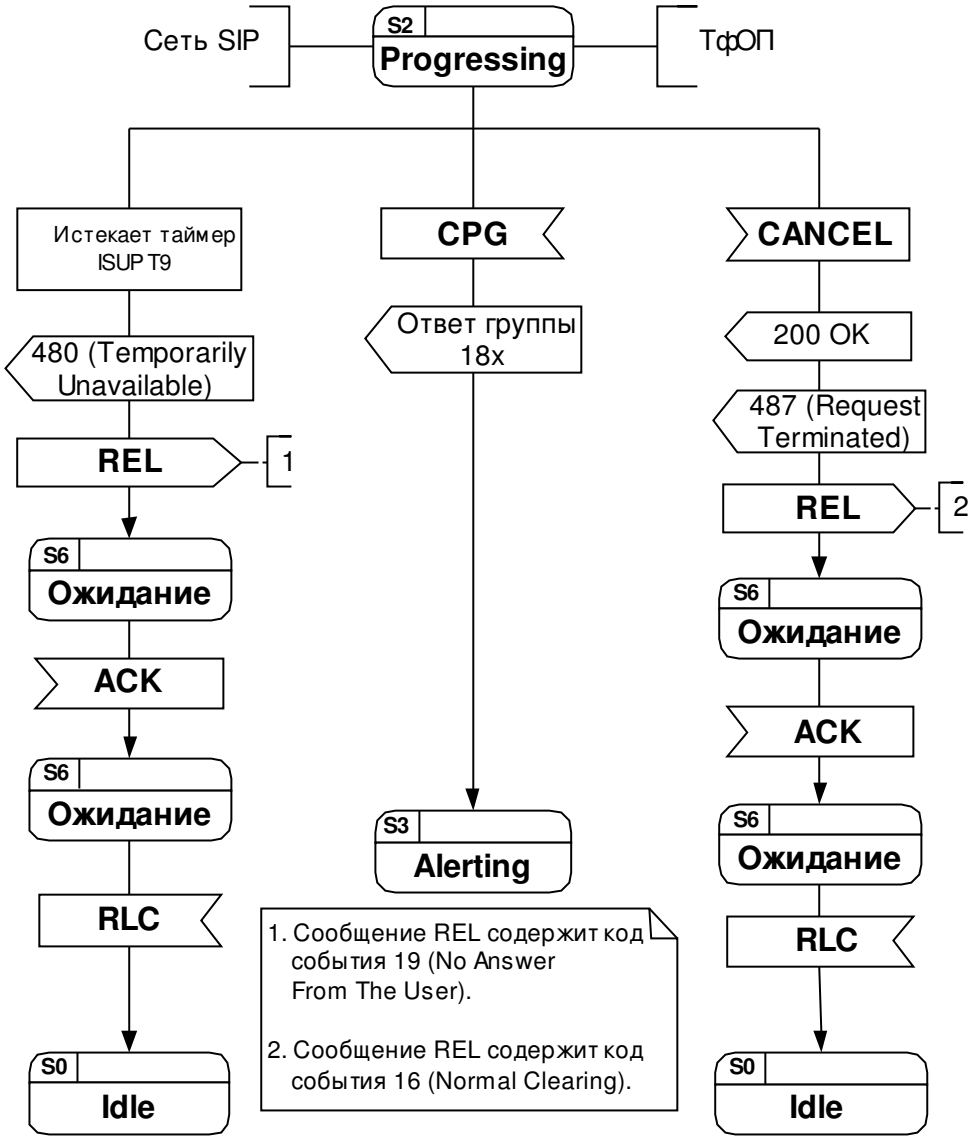


рисунок 3.39 Переходы из состояния «Progressing»

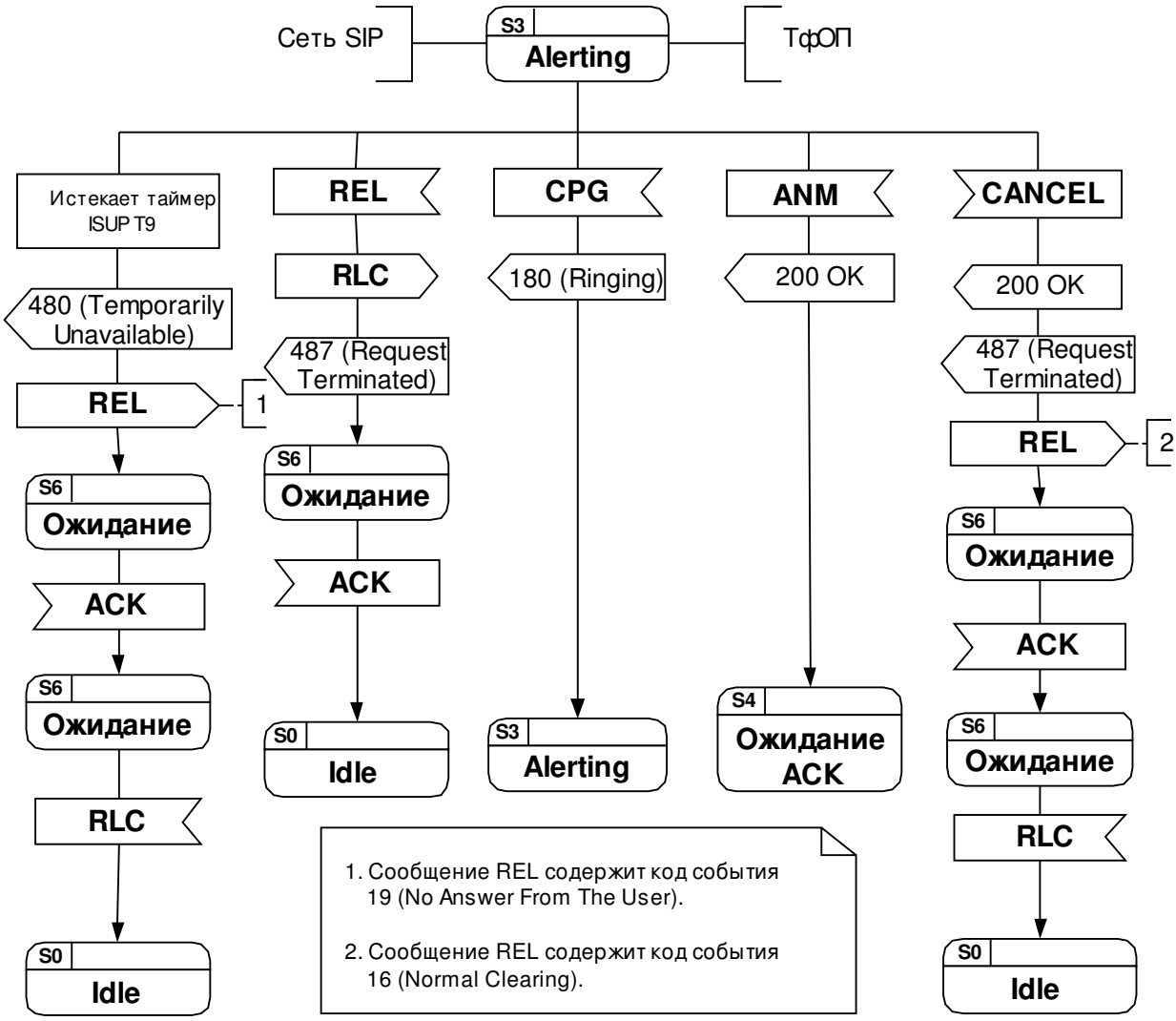


рисунок 3.40 Переходы из состояния «Alerting»

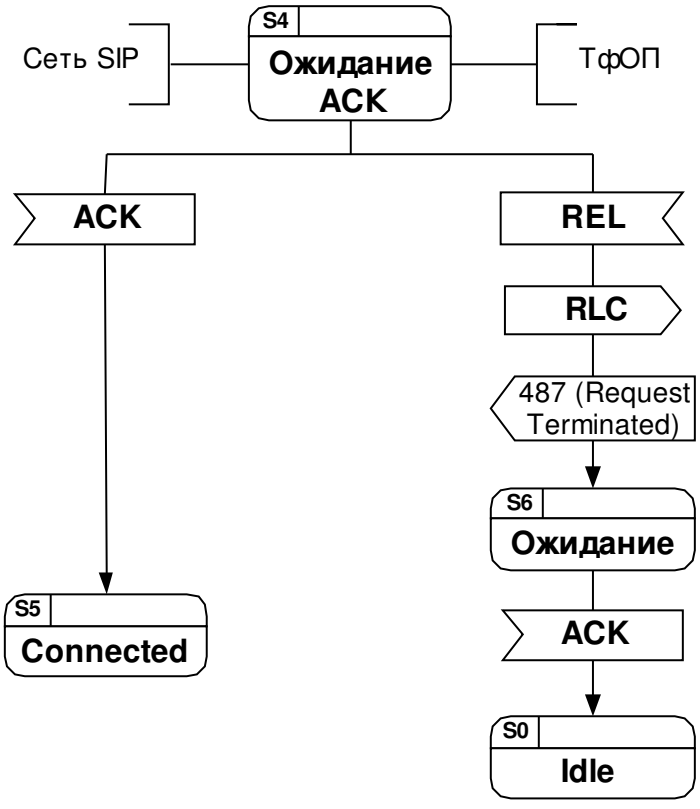


рисунок 3.41 Переходы из состояния «Ожидание АСК»

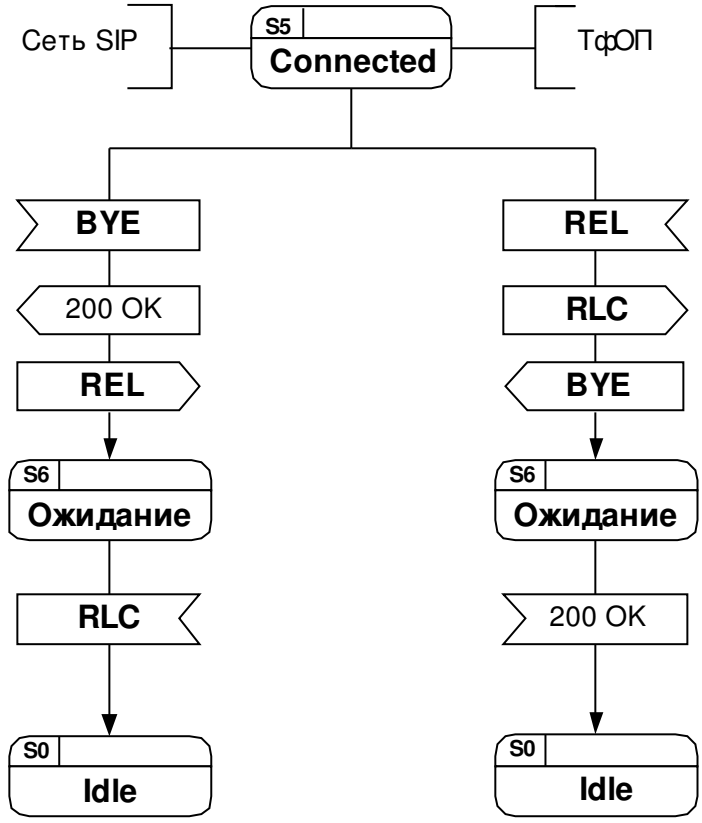


рисунок 3.42 Переходы из состояния «Connected»

Состояния:

S0 – «Idle» - исходное состояние - соединения не существует

S1 – «Trying» - пробное состояние - периодически отсылается запрос INVITE, ожидание ответов.

S2 – «Progressing» - состояние приёма ответов – получено предварительное ACM, ожидание ответа группы 18x.

S3 – «Alerting» - состояние вызова пользователя - окончания обработки сообщений – вызываемый абонент получает ПВ, вызывающий – ПКВ, ожидание окончательного ответа 200 ОК.

S4 – «Ожидание АСК» - состояние ожидания подтверждения параметров соединения – ожидание запроса АСК, подтверждающего параметры соединения.

S5 – «Connected» - состояние окончания обработки сообщений – соединение установлено.

S6 – «Ожидание» - состояние ожидания ответов.

3.6.4.4 Примеры сценариев и сообщений для случая вызова SIP – ТфОП

В следующих сценариях пользователь Maxim (sip:max@loniis.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 проходить через прокси сервер Proxy1, и шлюз Network Gateway. В некоторых сценариях Maxim звонит абоненту Aleksey, 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 необходимого для занесения в параметр calling party number сообщения ISUP.

В сценариях с ошибками соединение не устанавливается. Однако в некоторых случаях медиапотоки все же проключаются. Это бывает в тех случаях, когда ТфОП выдает акустические сигналы о состоянии абонента (“занято” и другие 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. Данные разговорные потоки устанавливаются с использованием ответа 183 (Session Progress), который содержит в себе данные SDP.

Передача служебных объявлений может быть завершена пользователем, или шлюзом после истечения таймера.

В остальных сценариях коды причины ISUP транслируются в ответы SIP. В этих сценариях медиапотоки не устанавливаются, но коды ошибок доводятся до пользователя с помощью SIP UAC.

Успешное установление соединения из сети SIP в ТфОП

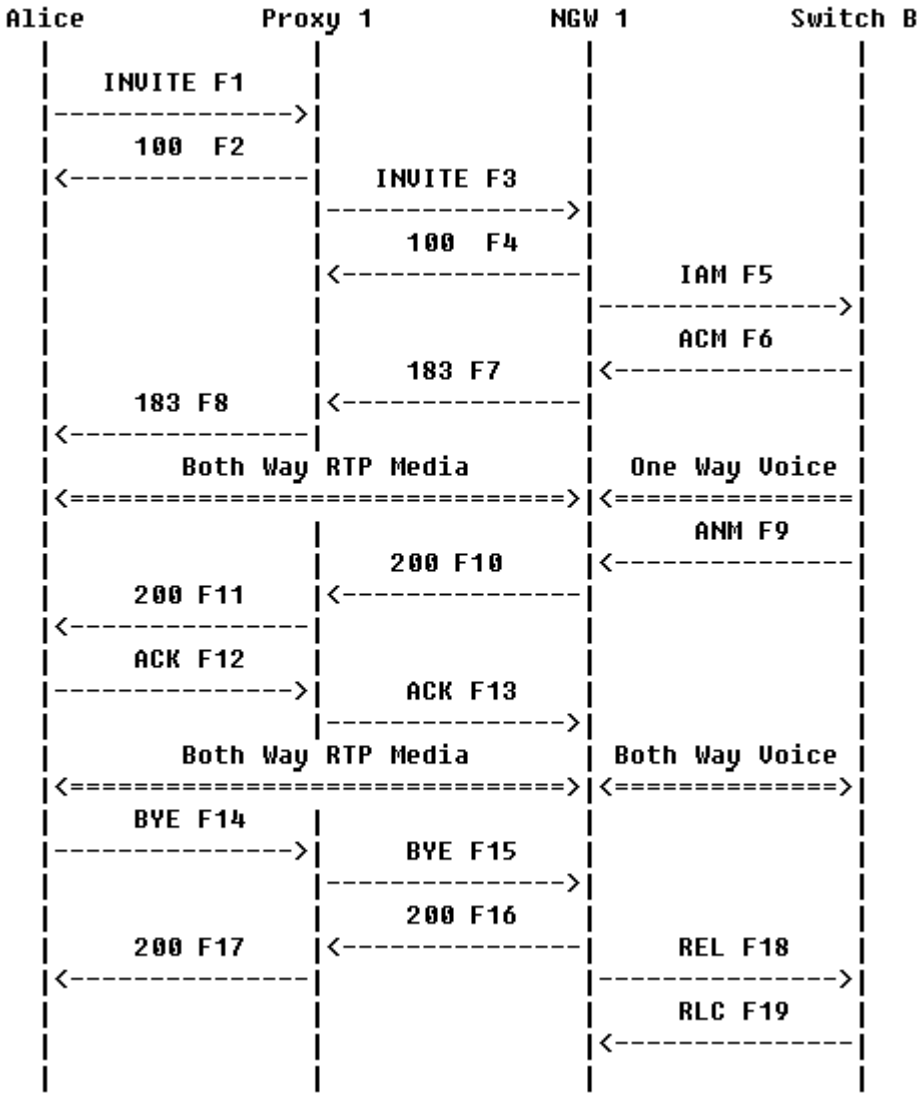


Рисунок 3.43 Диаграмма обмена сообщениями. Успешное установление соединения SIP – ТфО П.

Пользователь Maxim набирает международный номер +78122625326 стандарта E.164 для того, чтобы позвонить абоненту Anton. Особое значение имеют последние 7 цифр (или меньше в зависимости от нумерации на местной сети). Предполагается, что SIP UA может конверовать набранный номер в международный формат и помещает его в SIP URI. Вместо SIP URI может использоваться tel URI.

Maxim может в заголовке From свой SIP адрес (sip:max@loniis.ru) или телефонный номер SIP (sip:+70953864515@ss1.loniis.ru;user=phone). В данном примере используется телефонный номер, также этот номер будет использован при трансляции на NGW1 как номер вызывающей стороны (calling party identification). Чтобы этот номер прошел в сеть ТфОП его корректность обязательно должна быть проверена и подтверждена.

В этом сценарии Anton отвечает на вызов, затем после разговора пользователь Maxim первым вешает трубку.

Также в данном сценарии в сообщениях F7 – F11, возвращаемых шлюзом, в заголовке Contact записано sip:ngw1@a.loniis.ru. Это сделано потому, что NGW1 принимает только запросы, пришедшие от Proxy1 – остальные сигнальные сообщения игнорируются. Так как это

значение заголовка Contact может использоваться не только в этом диалоге и за пределами этого сценария, URI в заголовке Contact для NGW1 должен назначаться Proxy1. Этот URI назначается службой DNS для Proxy1 (sip:ss1.a.loniis.ru), который потом изменяет его на sip:ngw1.a.loniis.ru, что и является адресом NGW1.

В данном сценарии для транспортировки сообщений используется протокол TCP.

Содержимое сообщений:

F1 INVITE Maxim -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru;transport=tcp>
Proxy-Authorization: Digest username="Max", realm="a.loniis.ru",
 nonce="dc3a5ab25302aa931904ba7d88fa1cf5", opaque="",
 uri="sip:+78122625326@ss1.a.loniis.ru;user=phone",
 response="ccdca50cb091d587421457305d097458c"
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F2 100 (Trying) Proxy 1 -> Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
```

To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

Proxy 1 использует функцию определения местоположения для поиска адреса нужного шлюза к которому нужно отправить вызов. Вызов направляется к шлюзу NGW1. Клиент, установленный на терминале пользователя Maxim готовиться принимать на порт 49172 данные из сети.

F3 INVITE Proxy 1 -> NGW 1

INVITE sip:+78122625326@ngw1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) NGW 1 -> Proxy 1

SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>

Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 IAM NGW 1 -> ATC B

Передается сообщение IAM
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National

F6 ACM ATC B -> NGW 1

Передается сообщение ACM

F7 183 Session Progress NGW 1 -> Proxy 1

SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

NGW 1 отправляет аудио данные из ТфОП (КПВ) по каналу RTP Maxim

F8 183 (Session Progress) Proxy 1 -> Maxim

SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F9 ANM ATC B -> NGW 1

Передается сообщение ANM

F10 200 (OK) NGW 1 -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE

Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F11 200 (OK) Proxy 1 -> Maxim

SIP/2.0 200 OK
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F12 ACK Maxim -> Proxy 1

ACK sip:ngw1@a.loniis.ru SIP/2.0
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70

Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F13 ACK Proxy 1 -> NGW 1

ACK sip:ngw1@a.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

Maxim вешает трубку, завершая разговор с пользователем Anton.

F14 BYE Maxim -> Proxy 1

BYE sip:ngw1@a.loniis.ru SIP/2.0
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F15 BYE Proxy 1 -> NGW 1

BYE sip:ngw1@a.loniis.ru SIP/2.0

Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F16 200 (OK) NGW 1 -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F17 200 (OK) Proxy 1 -> Maxim

SIP/2.0 200 OK
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F18 REL NGW 1 -> ATC B

Передается сообщение REL с CauseCode=16 Normal

F19 RLC ATC B -> NGW 1

Передается сообщение RLC

Успешное установление соединения из сети SIP в ТфО П с испльзованием УПАТС.

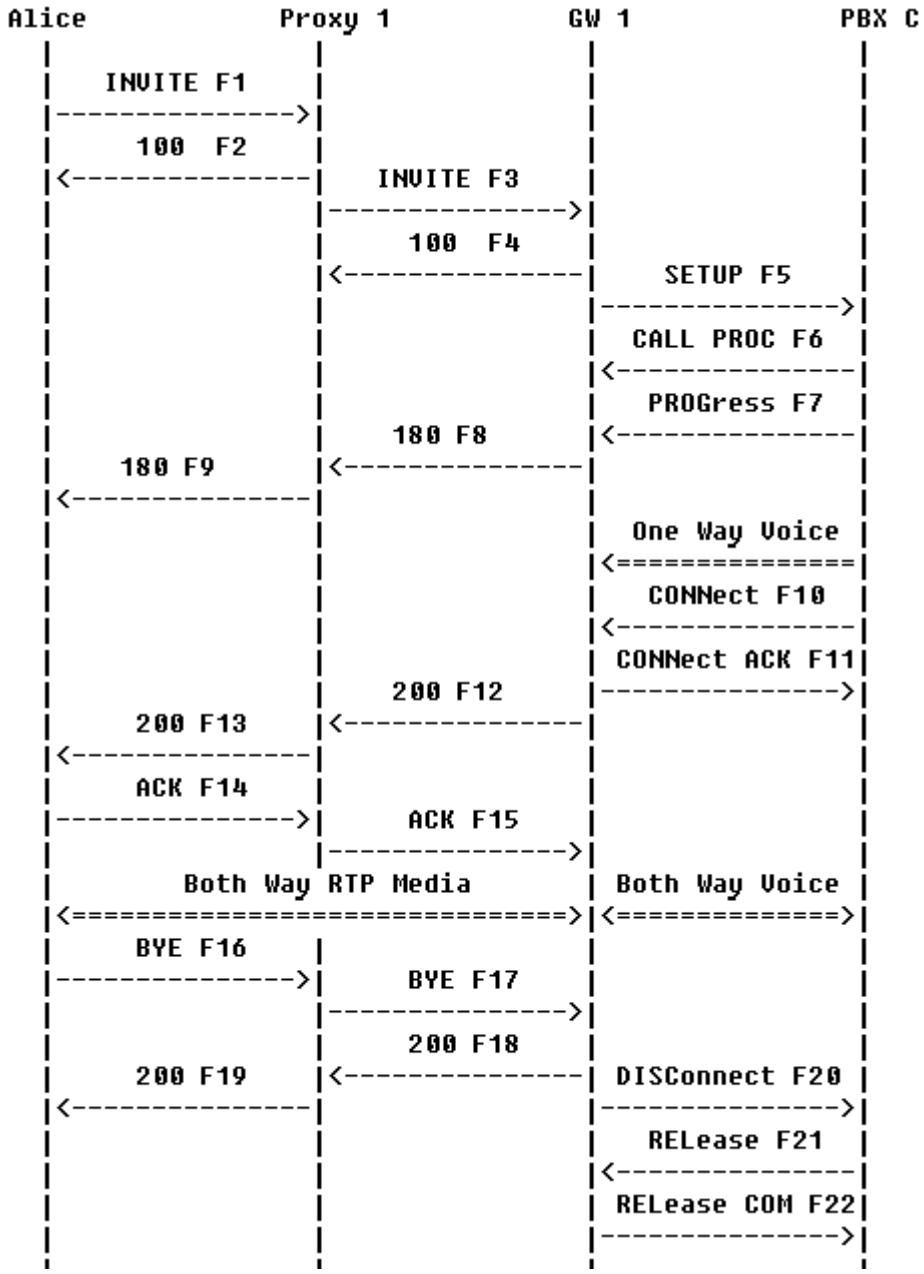


Рисунок 3.44 Диаграмма обмена сообщениями. Успешное установление соединения из сети SIP к абоненту ТфО П, использующиму УПАТС.

Пользователь Maxim находиться в сети SIP, а абонент Alexey подключен к УПАТС. Между шлюзоми УПАТС используется сигнализация DSS. УПАТС подключается через группу каналов ISDN. Maxim набирает международный номер Алексея (+7-812-387-5333),

который помещается в SIP URI.

Часть адреса, находящегося в поле 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 находиться в поле Request-URI запроса, отправляемого шлюзу.

Заметьте, что значение заголовка Contact для GW 1, используемый в сообщениях F8, F9, F12, и F13 - sips:4443333@gw1.a.loniis.ru, который переадресует все вызовы прямо на сервер.

В данном сценарии используются SIPS URI.

Содержимое сообщений

F1 INVITE Maxim -> Proxy 1

```
INVITE sips:+78123875333@ss1.a.loniis.ru;user=phone  SIP/2.0
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sips:max@client.a.loniis.ru>
Proxy-Authorization: Digest username="Max",
 realm="a.loniis.ru", nonce="qo0dc3a5ab22aa931904badfa1cf5j9h",
 opaque="", uri="sips:+78123875333@ss1.a.loniis.ru;user=phone",
 response="6c792f5c9fa360358b93c7fb826bf550"
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F2 100 (Trying) Proxy 1 -> Maxim

SIP/2.0 100 Trying
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Content-Length: 0

F3 INVITE Proxy 1 -> GW 1

INVITE sips:4443333@gw1.a.loniis.ru SIP/2.0
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sips:max@client.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) GW -> Proxy 1

SIP/2.0 100 Trying
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>

;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Content-Length: 0

F5 SETUP GW 1 -> УПАТС С

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

F6 CALL PROCeeding УПАТС С-> GW 1

Protocol discriminator=Q.931
Message type=CALL PROC
Channel identification=Exclusive B-channel

F7 PROGress УПАТС С -> GW 1

Protocol discriminator=Q.931
Message type=PROG
Progress indicator=1

F8 180 (Ringing) GW 1 -> Proxy 1

SIP/2.0 180 Ringing
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>

;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.loniis.ru>
Content-Length: 0

F9 180 (Ringing) Proxy 1 -> Maxim

SIP/2.0 180 Ringing
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.loniis.ru>
Content-Length: 0

F10 CONNect УПАТС С -> GW 1

Protocol discriminator=Q.931
Message type=CONN

F11 CONNect ACK GW 1 -> УПАТС С

Protocol discriminator=Q.931
Message type=CONN ACK

F12 200 (OK) GW 1 -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>

;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F13 200 (OK) Proxy 1 -> Maxim

SIP/2.0 200 OK
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sips:4443333@gw1.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 144

v=0
o=GW 2890844527 2890844527 IN IP4 gw1.a.loniis.ru
s=-
c=IN IP4 gw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F14 ACK Maxim -> Proxy 1

ACK sips:4443333@gw1.a.loniis.ru SIP/2.0
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 ACK
Content-Length: 0

F15 ACK Proxy 1 -> GW 1

ACK sips:4443333@gw1.a.loniis.ru SIP/2.0
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 ACK
Content-Length: 0

Далее Maxim вешает трубку, завершая разговор с пользователем Alexey.

F16 BYE Maxim -> Proxy 1

BYE sips:4443333@gw1.a.loniis.ru SIP/2.0
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <sips:ss1.a.loniis.ru;lr>
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 3 BYE
Content-Length: 0

F17 BYE Proxy 1 -> GW 1

```
BYE sips:4443333@gw1.a.loniis.ru SIP/2.0
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 3 BYE
Content-Length: 0
```

F18 200 (OK) GW 1 -> Proxy 1

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS ss1.a.loniis.ru:5061;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 3 BYE
Content-Length: 0
```

F19 200 (OK) Proxy 1 -> Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/TLS client.a.loniis.ru:5061;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sips:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Carol <sips:+78123875333@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 3 BYE
Content-Length: 0
```

F20 DISConnect GW 1 -> УПАТС С

Protocol discriminator=Q.931
Message type=DISC
Cause=16 (Normal clearing)

F21 RELease УПАТС С -> GW 1

Protocol discriminator=Q.931
Message type=REL

F22 RELease COMplete GW 1 -> УПАТС С

Protocol discriminator=Q.931
Message type=REL COM

Успешное установление соединения из сети SIP в сеть ТфО П в условиях перегрузки шлюза

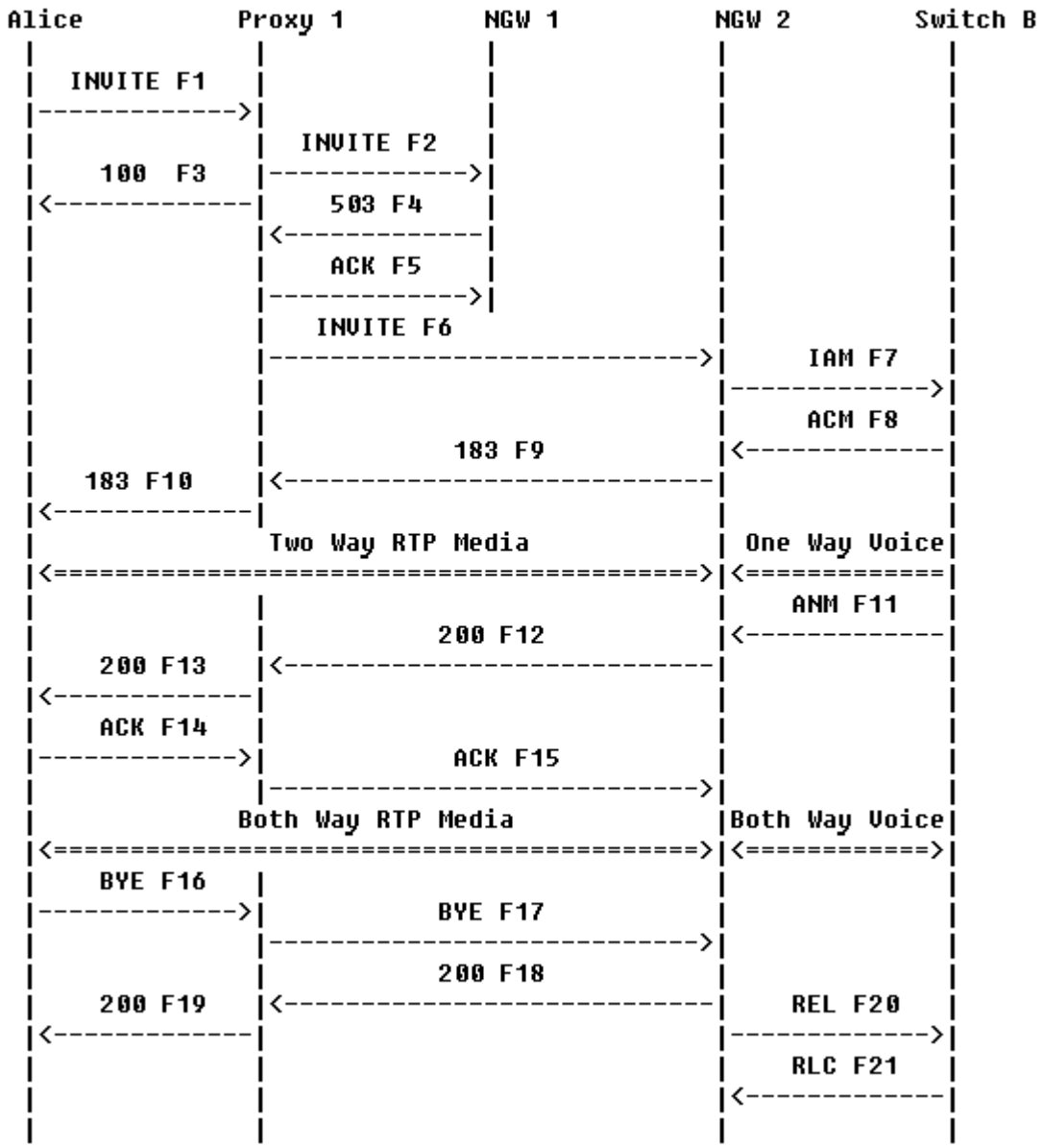


Рисунок 3.45 Диаграмма обмена сообщениями. Успешное установление соединения SIP – ТфО П в условиях перегрузки шлюза.

Пользователь Maxim отправляет вызов абоненту Anton через Proxy 1. Proxy 1 перенаправляет вызов на Network Gateway NGW 1. Данный шлюз в этот момент недоступен (загружен обработкой других вызовов) и отклоняет пришедший запрос, возвращая 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

F1 INVITE Maxim -> Proxy 1

INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9

```
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru>
Proxy-Authorization: Digest username="Max",
 realm="a.loniis.ru", nonce="b59311c3ba05b401cf80b2a2c5ac51b0",
 opaque="", uri="sip:+78122625326@ss1.a.loniis.ru;user=phone",
 response="ba6ab44923fa2614b28e3e3957789ab0"
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения местоположения для нахождения текущего адреса абонента Anton. Proxy 1 получает два варианта расположения абонента основной NGW 1и дополнительный NGW 2. Первый запрос направляется на NGW 1

F2 INVITE Proxy 1 -> NGW 1

```
INVITE sip:+78122625326@ngw1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 154
```

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F3 100 (Trying) Proxy 1 -> Maxim

SIP/2.0 100 Trying
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F4 503 (Service Unavailable) NGW 1 -> Proxy 1

SIP/2.0 503 Service Unavailable
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=123456789
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 ACK Proxy 1 -> NGW 1

ACK sip:ngw1@a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1

Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru>;user=phone>
;tag=123456789
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

Proxy 1 направляет запрос на шлюз NGW 2

F6 INVITE Proxy 1 -> NGW 2

INVITE sip:+78122625326@ngw2.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.2
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F7 IAM NGW 2 -> ATC B

Получение сообщения IAM
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National

F8 ACM ATC B -> NGW 2

Получение сообщения ACM

F9 183 (Session Progress) NGW 2 -> Proxy 1

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.2
 ;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.loniis.ru
s=-
c=IN IP4 ngw2.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Пользователю Maxim от шлюза посылаются RTP пакеты с аудио данными (КПВ).

F10 183 (Session Progress) Proxy 1 -> Maxim

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
```

Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.loniis.ru
s=-
c=IN IP4 ngw2.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F11 ANM ATC B -> NGW 2

Получение сообщения ANM

F12 200 (OK) NGW 2 -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.2
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.loniis.ru
s=-
c=IN IP4 ngw2.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F13 200 (OK) Proxy 1 -> Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw2@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw2.a.loniis.ru
s=-
c=IN IP4 ngw2.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F14 ACK Maxim -> Proxy 1

```
ACK sip:ngw2@a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0
```

F15 ACK Proxy 1 -> NGW 2

```
ACK sip:ngw2@a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.2
```

Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

Между пользователем Maxim и абонентом Anton проключается RTP поток (через GW). Через некоторое время Maxim вешает трубку, завершая разговор с абонентом Anton.

F16 BYE Maxim -> Proxy 1

BYE sip:ngw2@a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
Route: <ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F17 BYE Proxy 1 -> NGW 2

BYE sip:ngw2@a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.2
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Max-Forwards: 69
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F18 200 (OK) NGW 2 -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.2
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F19 200 (OK) Proxy 1 -> Maxim

SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 BYE
Content-Length: 0

F20 REL NGW 2 -> ATC B

Получение сообщения REL с CauseCode=16 Normal

F21 RLC ATC B -> NGW 2

Получение сообщения RLC

Успешное установление соединения внутри сети SIP с использованием ENUM Query

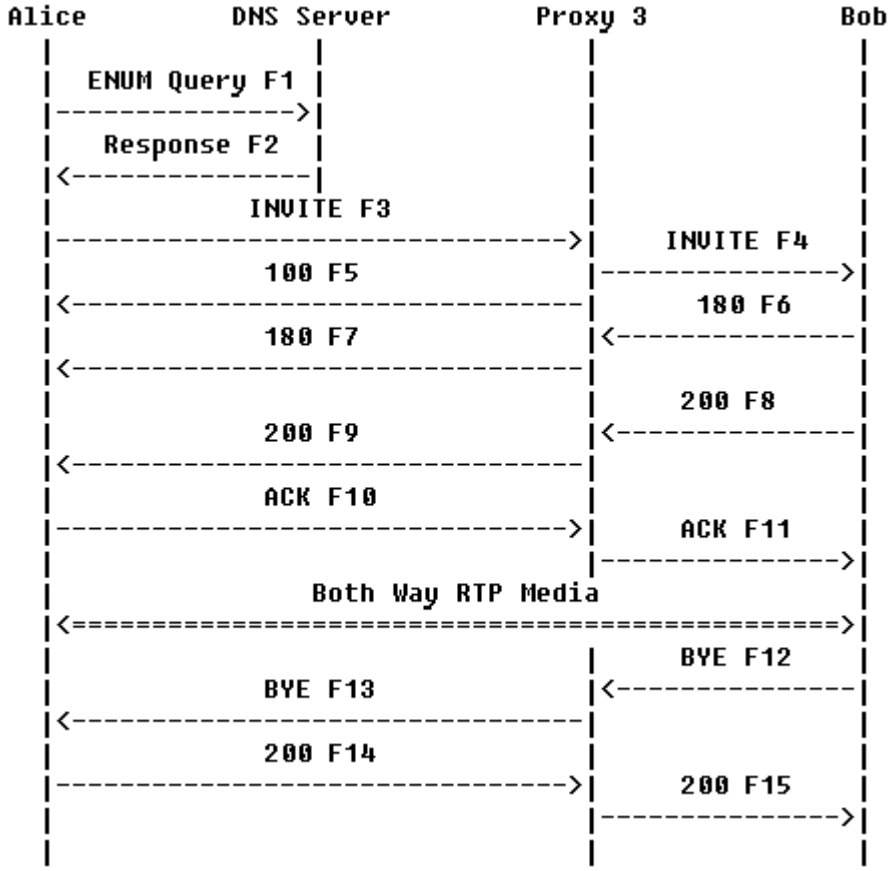


Рисунок 3.46 Диаграмма обмена сообщениями. Успешное установление соединения внутри сети SIP с использованием ENUM Query.

В данном сценарии пользователь Maxim пытается вызвать абонента Anton, набирая его телефонный номер (9722625326). UA пользователя Maxim преобразует введенный телефонный номер в формат номера E.164 (+78122625326), и выполняет процедуру ENUM над 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 возвращается публичный адрес пользователя Anton (sip:+78122625326@b.loniis.ru). Если адрес имеет такой вид это означает, что в данный момент пользователь Anton находиться в сети SIP. Результатом всего этого является то, что UA пользователя Maxim отправляет запрос INVITE и соединение устанавливается в сети SIP, минуя ТфОП. Разговор

заканчивается, когда терминал пользователя Anton посылает сообщение BYE.

Содержимое сообщений.

F1 ENUM Query Maxim -> DNS Server

```
2.2.2.2.5.5.5.2.7.9.1.e164.arpa
```

F2 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.loniis.ru!".
```

F3 INVITE Maxim -> Proxy 3

```
INVITE sip:+78122625326@b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sip:+70953864515@client.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F4 INVITE Proxy 3 -> Anton

```
INVITE sip:+78122625326@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss3.b.loniis.ru:5060;branch=z9hG4bK721e418c4.1
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
```

Record-Route: <sip:ss3.b.loniis.ru;lr>
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sip:+70953864515@client.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=UserA 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F5 100 (Trying) Proxy 3 -> Maxim

SIP/2.0 100 Trying
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Content-Length: 0

F6 180 (Ringing) Anton -> Proxy 3

SIP/2.0 180 Ringing
Via: SIP/2.0/UDP ss3.b.loniis.ru:5060;branch=z9hG4bK721e418c4.1
 ;received=192.0.2.233
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss3.b.loniis.ru;lr>
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.loniis.ru>
Content-Length: 0

F7 180 (Ringing) Proxy 3 -> Maxim

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss3.b.loniis.ru;lr>
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.loniis.ru>
Content-Length: 0
```

F8 200 (OK) Anton -> Proxy 3

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss3.b.loniis.ru:5060;branch=z9hG4bK721e418c4.1
 ;received=192.0.2.233
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss3.b.loniis.ru;lr>
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 client.b.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F9 200 (OK) Proxy -> Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss3.b.loniis.ru;lr>
```

From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 INVITE
Contact: <sip:+78122625326@client.b.loniis.ru>
Content-Type: application/sdp
Content-Length: 151

v=0
o=Anton 2890844527 2890844527 IN IP4 client.b.loniis.ru
s=-
c=IN IP4 192.0.2.100
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F10 ACK Maxim -> Proxy 3

ACK sip:+78122625326@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bq9
Max-Forwards: 70
Route: <sip:ss3.b.loniis.ru;lr>
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 ACK
Content-Length: 0

F11 ACK Proxy 3 -> Anton

ACK sip:+78122625326@client.b.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss3.b.loniis.ru:5060;branch=z9hG4bK721e418c4.1
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bq9
;received=192.0.2.101
Max-Forwards: 69
From: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
To: <tel:+78122625326>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 2 ACK
Content-Type: application/sdp
Content-Length: 0

Между пользователем Maxim и абонентом Anton проключается RTP поток. Затем Anton

вешает трубку, завершая разговор с пользователем Maxim.

F12 BYE Anton -> Proxy 3

```
BYE sip:+70953864515@client.a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP client.b.loniis.ru:5060;branch=z9hG4bKfgaw2
Max-Forwards: 70
Route: <sip:ss3.b.loniis.ru;lr>
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 BYE
Content-Length: 0
```

F13 BYE Proxy 3 -> Maxim

```
BYE sip:+70953864515@client.a.loniis.ru SIP/2.0
Via: SIP/2.0/UDP ss3.b.loniis.ru:5060;branch=z9hG4bK721e418c4.1
 ;received=192.0.2.100
Via: SIP/2.0/UDP client.b.loniis.ru:5060;branch=z9hG4bKfgaw2
Max-Forwards: 69
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 BYE
Content-Length: 0
```

F14 200 (OK) Maxim -> Proxy 3

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP ss3.b.loniis.ru:5060;branch=z9hG4bK721e418c4.1
 ;received=192.0.2.233
Via: SIP/2.0/UDP client.b.loniis.ru:5060;branch=z9hG4bKfgaw2
 ;received=192.0.2.100
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 BYE
Content-Length: 0
```

F15 200 (OK) Proxy 3 -> Anton

SIP/2.0 200 OK
Via: SIP/2.0/UDP client.b.loniis.ru:5060;branch=z9hG4bKfgaw2
;received=192.0.2.100
From: <tel:+78122625326>;tag=314159
To: <sip:+70953864515@a.loniis.ru>;tag=9fxced76sl
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 BYE
Content-Length: 0

Неуспешное установление соединения из сети SIP в сеть ТфОП: Сообщение об ошибке из ТфОП

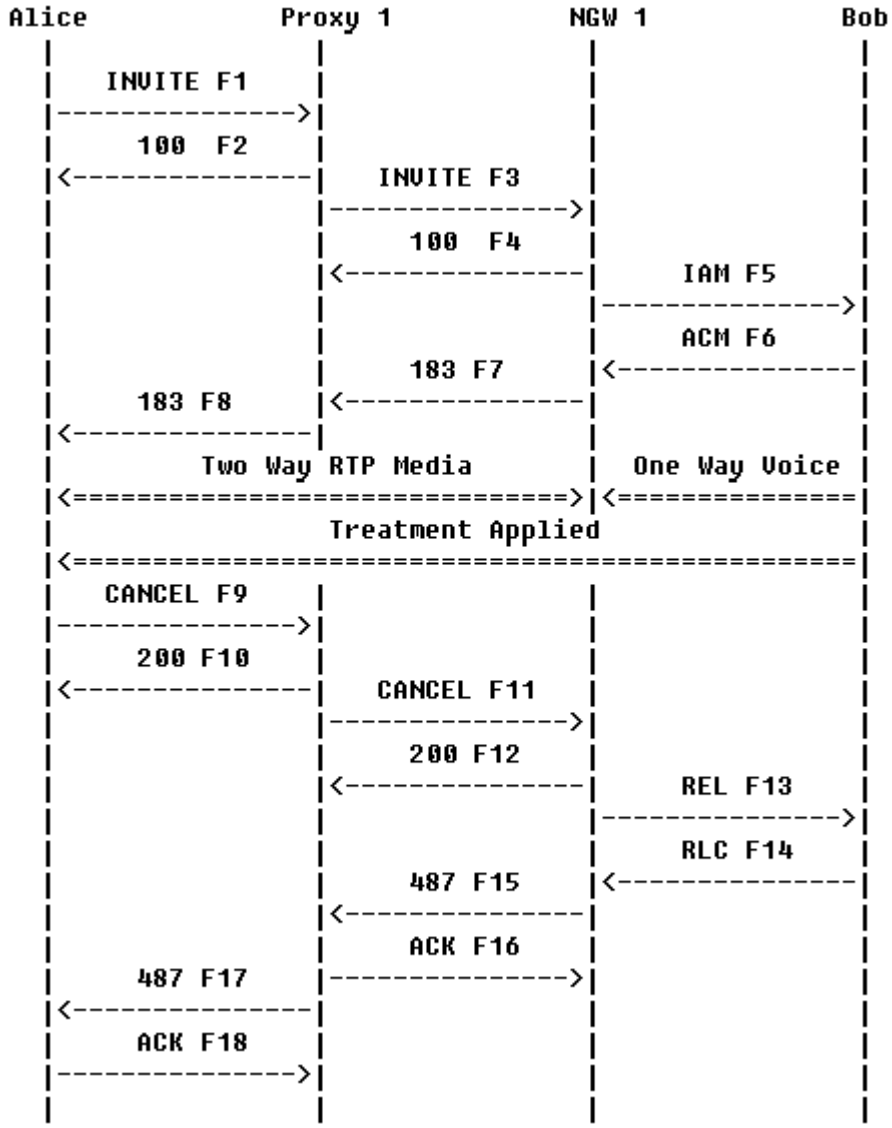


Рисунок 3.47 Диаграмма обмена сообщениями. Неуспешное установление соединения SIP – ТфОП. Ошибка в ТфОП.

Maxim отправляет вызов абоненту Anton через Proxy 1 и Network Gateway NGW Вызов отклоняется АТС В в ТфОП, а терминалу пользователя Maxim выдается звуковое

сообщение об ошибке (или звуковой сигнал). Пользователь Maxim, прослушав звуковое сообщение, вешает трубку, в результате чего терминалом отправляется запрос CANCEL для прекращения обработки вызова. Отправляется CANCEL, а не BYE, т.к. не было получено окончательного ответа на INVITE.

Содержимое сообщений

F1 INVITE Maxim -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru>
Proxy-Authorization: Digest username="Max",
 realm="a.loniis.ru", nonce="01cf8311c3b0b2a2c5ac51bb59a05b40",
 opaque="", uri="sip:+78122625326@ss1.a.loniis.ru;user=phone",
 response="e178fbe430e6680a1690261af8831f40"
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F2 100 (Trying) Proxy 1 -> Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0
```

Proxy 1 использует функцию определения местоположения чтобы найти текущее расположение абонента Anton. На основе этого анализа вызов направляется шлюзу NGW 1. Терминал пользователя Maxim готовиться принимать данные из сети на порт 49172 .

F3 INVITE Proxy 1 -> NGW 1

```
INVITE sip:+78122625326@ngw1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F4 100 (Trying) NGW 1 -> Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0
```

F5 IAM NGW 1 -> ATC B

Отправляется сообщение IAM
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National

F6 ACM ATC B -> NGW 1

Отправляется сообщение ACM

F7 183 (Session Progress) NGW 1 -> Proxy 1

SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F8 183 (Session Progress) Proxy 1 -> Maxim

SIP/2.0 183 Session Progress
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101

```
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Вызывающий абонент прослушивает речевые объявления, после чего вешает трубку.

F9 CANCEL Maxim -> Proxy 1

```
CANCEL sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0
```

F10 200 (OK) Proxy 1 -> Maxim

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
```

Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

F11 CANCEL Proxy 1 -> NGW 1

CANCEL sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

F12 200 (OK) NGW 1 -> Proxy 1

SIP/2.0 200 OK
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 CANCEL
Content-Length: 0

F13 REL NGW 1 -> ATC B

Отправляется сообщение REL с CauseCode=18 No user responding

F14 RLC ATC B -> NGW 1

Отправляется сообщение RLC

F15 487 (Request Terminated) NGW 1 -> Proxy 1

SIP/2.0 487 Request Terminated
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101

From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F16 ACK Proxy 1 -> NGW 1

ACK sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F17 487 (Request Terminated) Proxy 1 -> Maxim

SIP/2.0 487 Request Terminated
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F18 ACK Maxim -> Proxy 1

ACK sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/UDP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159

Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

Неуспешное установление соединения из сети SIP в ТфОП: ТфОП отклоняет вызов, возвращая REL с кодом события

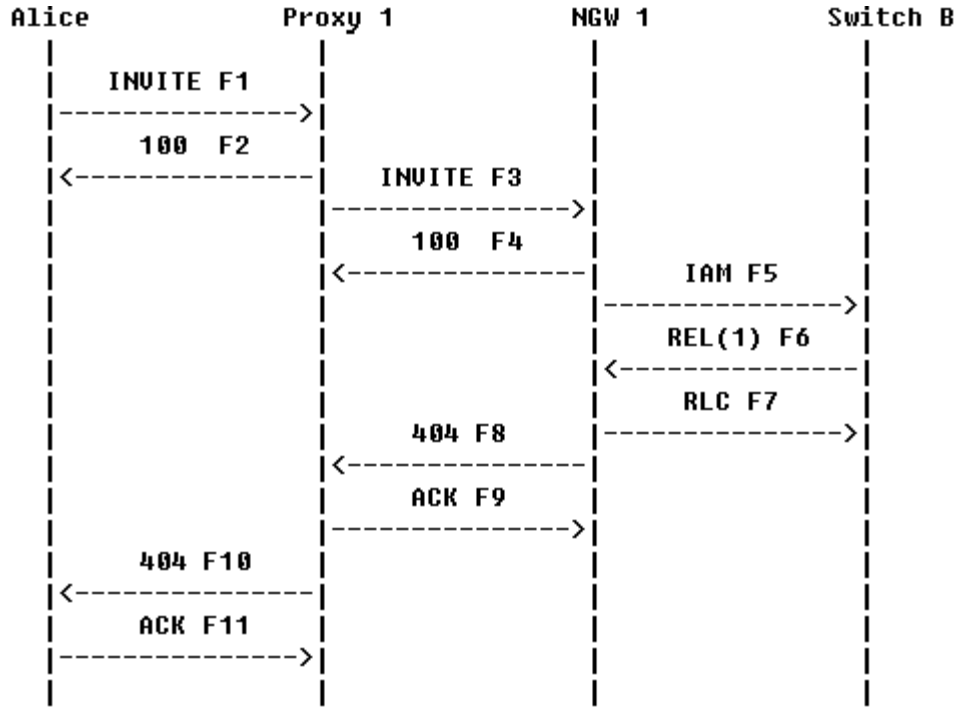


Рисунок 3.48 Диаграмма обмена сообщениями. Неуспешное установление соединения SIP – ТфОП. ТфОП отклоняет вызов, возвращая REL с кодом события.

Maxim отправляет вызов абоненту Anton через Proxy 1 и NGW 1. АТС В в ТфОП отклоняет вызов, возвращая сообщение REL с соответствующим кодом события. Этот код события транслируется в ответ SIP 404 (Not Found), и этот ответ отправляется терминалу пользователя Maxim.

Содержимое сообщений.

F1 INVITE Maxim -> Proxy 1

```
INVITE sip:+44-1234@ss1.a.loniis.ru;user=phone  SIP/2.0
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru;transport=tcp>
Proxy-Authorization: Digest username="Max",
 realm="a.loniis.ru", nonce="j1c3b0b01cf832da2c5ac51bb59a05b40",
 opaque="", uri="sip:+44-1234@ss1.a.loniis.ru;user=phone",
 response="a451358d46b55512863efe1dccaa2f42"
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F2 100 (Trying) Proxy 1 -> Maxim

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0
```

Proxy 1 использует функцию определения местоположения чтобы найти, где в данный момент находиться Anton. Далее вызов направляется шлюзу NGW 1. Клиент терминала Maxim готовиться принимать данные из сети на порт 49172.

F3 INVITE Proxy 1 -> NGW 1

```
INVITE sip:+44-1234@ngw1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

F4 100 (Trying) NGW 1 -> Proxy 1

```
SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0
```

F5 IAM NGW 1 -> ATC B

Отправляется сообщение IAM

CdPN=44-1234,NPI=E.164,NOA=International
CgPN=095-386-4515,NPI=E.164,NOA=National

F6 REL ATC B -> NGW 1

Отправляется сообщение REL с CauseValue=1 Unallocated number

F7 RLC NGW 1 -> ATC B

Отправляется сообщение RLC

Шлюз преобразует CauseValue=1 в ответ SIP 404 (Not Found)

F8 404 (Not Found) NGW 1 -> Proxy 1

SIP/2.0 404 Not Found
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Error-Info: <sip:not-found-ann@ann.a.loniis.ru>
Content-Length: 0

F9 ACK Proxy 1 -> NGW 1

ACK sip:+44-1234@ngw1.a.loniis.ru;user=phone SIP/2.0
Max-Forwards: 70
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F10 404 (Not Found) Proxy 1 -> Maxim

SIP/2.0 404 Not Found
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Error-Info: <sip:not-found-ann@ann.a.loniis.ru>
Content-Length: 0

F11 ACK Maxim -> Proxy 1

ACK sip:+44-1234@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+44-1234@ss1.a.loniis.ru;user=phone>;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

Неуспешное установление соединения из сети SIP в ТфО П: Истекает таймер ожидания сообщения ANM на шлюзе

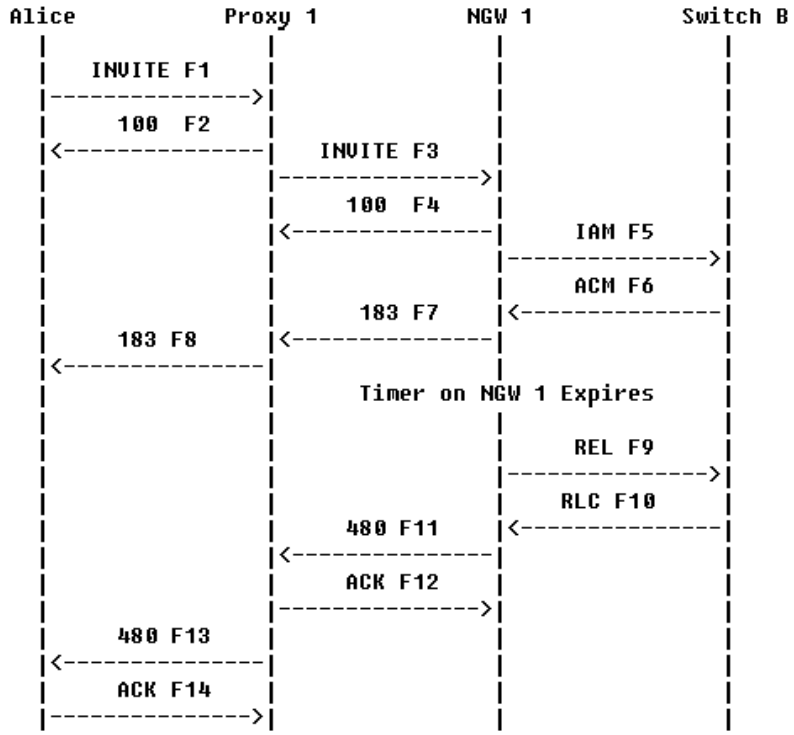


Рисунок 3.49 Диаграмма обмена сообщениями. Неуспешное установление соединения

SIP – ТфО П. Истекает таймер ожидания сообщения ANM.

Пользователь Maxim отправляет вызов абоненту Anton через Proxy 1 и NGW 1. Шлюз прерывает установление соединения после того, как истекает таймер ожидания сообщения ANM от АТС В в ТфОП (т.е. вызываемый абонент в течении определенного времени не отвечает на вызов). Для окончания установления соединения шлюз отправляет сообщение REL в ТфОП и ответ 480 (Temporarily Unavailable) терминалу пользователя Maxim в сеть SIP.

Содержимое сообщений:

F1 INVITE Maxim -> Proxy 1

```
INVITE sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru;transport=tcp>
Proxy-Authorization: Digest username="Max",
 realm="a.loniis.ru", nonce="da2c5ac51bb59a05j1c3b0b01cf832b40",
 opaque="", uri="sip:+78122625326@ss1.a.loniis.ru;user=phone",
 response="579cb9db184cdc25bf816f37cbc03c7d"
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

Proxy 1 использует функцию определения местоположения чтобы найти текущее расположение абонента Anton. На основе данных этого анализа вызов направляется шлюзу NGW 1. Клиент терминала пользователя Maxim готовиться принимать данные из сети на порт 49172 .

F2 100 (Trying) Proxy 1 -> Maxim

SIP/2.0 100 Trying
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F3 INVITE Proxy 1 -> NGW 1

INVITE sip:+78122625326@ngw1.a.loniis.ru;user=phone SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Max-Forwards: 69
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:max@client.a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 154

v=0
o=Max 2890844526 2890844526 IN IP4 client.a.loniis.ru
s=-
c=IN IP4 client.a.loniis.ru
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F4 100 (Trying) NGW 1 -> Proxy 1

SIP/2.0 100 Trying
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101

From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Content-Length: 0

F5 IAM NGW 1 -> ATC B

Отправляется сообщение IAM
CdPN=812-262-5326,NPI=E.164,NOA=National
CgPN=095-386-4515,NPI=E.164,NOA=National

F6 ACM ATC B -> NGW 1

Отправляется сообщение ACM

F7 183 (Session Progress) NGW 1 -> Proxy 1

SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000

F8 183 (Session Progress) Proxy 1 -> Maxim

```
SIP/2.0 183 Session Progress
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
Record-Route: <sip:ss1.a.loniis.ru;lr>
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
 ;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
 ;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Contact: <sip:ngw1@a.loniis.ru;transport=tcp>
Content-Type: application/sdp
Content-Length: 146

v=0
o=GW 2890844527 2890844527 IN IP4 ngw1.a.loniis.ru
s=-
c=IN IP4 ngw1.a.loniis.ru
t=0 0
m=audio 3456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

После того, как истекает время ожидания на шлюзе, он отправляет REL в ТфОП и ответ 480 (Temporarily Unavailable) в сеть SIP.

F9 REL NGW 1 -> ATC B

Отправляется сообщение REL с CauseCode=18 No user responding

F10 RLC ATC B -> NGW 1

Отправляется сообщение RLC

F11 480 (Temporarily Unavailable) NGW 1 -> Proxy 1

```
SIP/2.0 480 Temporarily Unavailable
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
 ;received=192.0.2.111
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
 ;received=192.0.2.101
```

From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Error-Info: <sip:temp-unavail-ann@ann.a.loniis.ru>
Content-Length: 0

F12 ACK Proxy 1 -> NGW 1

ACK sip:ngw1@a.loniis.ru SIP/2.0
Via: SIP/2.0/TCP ss1.a.loniis.ru:5060;branch=z9hG4bK2d4790.1
Max-Forwards: 70
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

F13 480 (Temporarily Unavailable) Proxy 1 -> Maxim

SIP/2.0 480 Temporarily Unavailable
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
;received=192.0.2.101
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl
To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 INVITE
Error-Info: <sip:temp-unavail-ann@ann.a.loniis.ru>
Content-Length: 0

F14 ACK Maxim -> Proxy 1

ACK sip:+78122625326@ss1.a.loniis.ru;user=phone SIP/2.0
Max-Forwards: 70
Via: SIP/2.0/TCP client.a.loniis.ru:5060;branch=z9hG4bK74bf9
From: Max <sip:+70953864515@ss1.a.loniis.ru;user=phone>
;tag=9fxced76sl

To: Anton <sip:+78122625326@ss1.a.loniis.ru;user=phone>
;tag=314159
Call-ID: 2xTb9vxSit55XU7p8@a.loniis.ru
CSeq: 1 ACK
Content-Length: 0

3.6.7 Формирование телефонных URI

Прокси-сервер в сети SIP может маршрутизировать сообщения в сети по любому признаку, заданному администратором сети. Обычно для этих целей используется поле Request-URI. Заголовки To и From так же представляют интерес при маршрутизации. Трансляция SIP–ТфОП предполагает, что прокси-сервер использует как минимум эти три поля, каждое из которых содержит URI.

При трансляции SIP–ТфОП часто требуется из SIP URI получить телефонный номер. Но также может потребоваться из телефонного номера в сообщении ISUP получить SIP URI.

Наиболее часто используемый формат, используемый в SIP, для представления телефонных номеров – это tel URL. При преобразовании между форматами tel URL может целиком занимать поле URI, т.е. tel URL используется как URI, или может быть пользовательской частью адреса в SIP URI. Например, заголовок 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 с соответствующим значением - ‘;user=phone’, например:

To: sip:83000@sip.loniis.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 различных областях или административных доменах. В большинстве сетей, где не используется SIP–T, шлюзы не отвечают за маршрутизацию вызова из конца в конец. Фактически это означает, что шлюз не сможет узнать каком в локальном или 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, набранный вызывающим абонентом. Если шлюз, закрепленный за вызывающим абонентом, не переведет номер в международный формат, то этот номер сможет правильно распознаться удаленными элементами сети.

В сигнализации ISUP формат телефонного номера определяется рядом отдельных параметров, таких, например, как Called Party Number (CPN) и Calling Party's Number (CIN); когда номер вызывающего абонента представляется этими параметрами, то к ним добавляется некоторая дополнительная информация.

В [RFC 3398 “ISUP to SIP Mapping”] CPN определяют как ISUP format, а формат номера вызывающего абонента CIN - ISUP calling format. Формат содержит байт, называемый «Индикатор типа адреса» [Nature of Address (NoA) indicator], он сопровождается другим байтом, который содержит «Индикатор плана нумерации» [Numbering Plan Indicator (NPI)]. Они оба являются префиксом к серии переменной длинны, содержащей цифры телефонного номера в двоичной форме [Binary Coded Decimal (BCD)]. В случае номера вызывающего абонента, поле NPI также содержит 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.

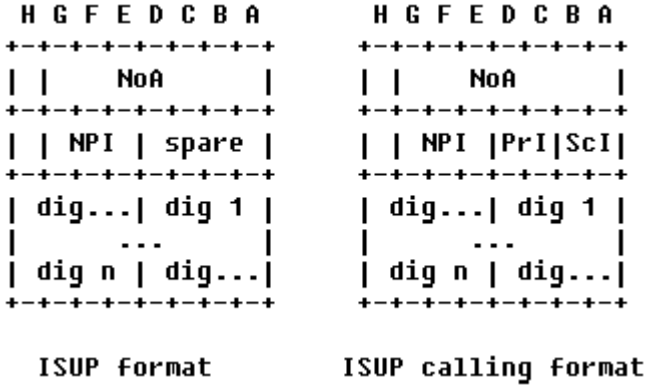


Рисунок 3.50 Форматы телефонных номеров ISUP format и ISUP calling format.

В поле NPI обычно устанавливается значение 'ISDN (Telephony) numbering plan (Recommendation E.164)', но это не означает, что цифры, следующие за этим полем, 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 данного 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 такого типа вызовов выходит за рамки данного справочника.

3.6.7.1 Процедура преобразования формата ISUP в формат tel URL

Основываясь на сказанном ранее, преобразование из формата ISUP в tel URL осуществляется следующим образом. Сначала, при условии, что 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, тогда при преобразовании перед цифрами номера должна быть добавлена строчка ‘tel:+’. Если поле NoA указывает на то, что номер находиться в национальном формате, то до того как номер будет транслирован в tel URL, перед цифрами номера необходимо добавить код страны.

Если в сообщении присутствует параметр, содержащий специфическую информацию для имени вызывающего абонента (например, Generic Name Parameter в ANSI), то, если параметр Presentation Indicator не установлен в 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', то шлюзом должен быть создан специальный URI, который сообщит удаленному терминалу, что идентификация пользователя не проводиться. 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', то шлюз должен обрабатывать сообщение IAM как будто параметр CIN отсутствует. Параметр screening indicators в данном случае не транслируется.

3.6.7.2 Процедура преобразования формата tel URL в формат ISUP

Преобразование из tel URL в формат ISUP выполняется проще. Если URI находиться в международном формате, то шлюз должен проанализировать код страны из URI. Если код страны является локальным для шлюза (шлюз имеет один или больше каналов, через которые можно установить речевое соединение, не выходя за пределы данного кода страны), то поле NoA должно быть установлено в значение ‘national (significant) number’, и код страны должен быть удален из URI перед трансляцией. Если область, определяемая кодом страны, не является локальной для шлюза, то в поле NoA устанавливается значение ‘international number’ и код страны оставляется в URI. В любом случае в поле NPI должно быть установлено значение ‘ISDN numbering plan’.

Если URI находиться не в международном формате, то 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 шлюз не поддерживает данные процедуры обработки, то к вызывающему оборудованию направляется ответ с соответствующим кодом состояния SIP, отражающим, что URI был не понят шлюзом

(если это URI является Request-URI, то отсылается ответ 484 (Address incomplete)).

При преобразовании tel URL в ISUP calling format процедура идентична приведенноой выше, но дополнительно параметр presentation indicator устанавливается в значение ‘presentation allowed’, а параметр screening indicator в значение ‘network provided’, если настройки сети или параметры пользователя не указывают на другие значения.

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, к примеру, md5( ) и затем сравнивать зашифрованный пароль с результатом обработки пароля, введенного пользователем при попытке получения доступа.
диалог	Диалог представляет собой равноправное взаимодействие двух агентов пользователя по протоколу SIP, которое длится определённое время. Диалог устанавливает последовательность сообщений между UA и обеспечивает верную маршрутизацию запросов.
диалог на ранней стадии	Равноправное взаимодействие двух агентов пользователя на стадии организации диалога, созданное в результате отсылки предварительного ответа. Разрушается после прихода окончательного ответа.
инкапсуляция	Процедура вложения сигнальных сообщений ISUP в запросы SIP
информация answer	Ответ на предложение с описанием сеанса связи в формате SDP, передаваемый в теле сообщения SIP и содержащий принятые параметры сессии.
информация offer	Предложение с описанием сеанса связи в формате SDP, передаваемое в теле сообщения SIP одним из участников вызова.
ОКС №7	Стек протоколов ОКС №7
параметр «q»	Параметр «q» определяет приоритеты среди значений, содержащихся в заголовке путём варьирования значения в пределах от 0 до 1.
сессия	Совокупность участников соединения и медиа-потоков между ними, созданных с целью обмена информацией.
трансляция	Процедура перевода значений части параметров сообщения ISUP в значение заголовков запросов SIP
ТфОП	Телефонная сеть общего пользования
Stateless Прокси-сервер	Прокси-сервер без сохранения состояний. Работает как ретранслирующий узел сети. Он пересылает каждый запрос следующему элементу, принимая решения о маршрутизации исходя из информации, содержащейся в запросе. Полученные ответы он

	просто пересылает обратно. Прокси-серверы без сохранения состояний удаляют информацию о прошедшем сообщении, как только сообщение было ретранслировано.
Stateful прокси-сервер	Прокси-сервер с сохранением состояний. Хранит информацию (состояние транзакции) о каждом поступившем сообщении и о каждом отосланном сообщении, возникающем вследствие обработки входящего запроса.
BCI	Backward Call Indicators – обратные индикаторы условий обслуживания вызовов. Параметр сообщений ISUP.
called party status	Статус вызываемой стороны. Параметр сообщения ISUP.
CIC	Carrier Identification Code – идентификатор оператора (сети).
CIN	Calling Party's Number – номер вызывающего абонента. Параметр сообщений ISUP.
CON	Connect Message – сообщение IUSP.
CPG	Call Progress – сообщение ISUP.
CPN	Called Party Number – номер вызываемого абонента. Параметр сообщений ISUP.
event package	Идентификатор функциональных возможностей для информирования клиента (подписчика) о событиях определённого типа, произошедших на указанном ресурсе.
FCI	Forward Call Indicators - прямые индикаторы условий обслуживания вызовов. Параметр сообщения ISUP.
GAP	Generic Address Parameter – параметр сообщения ISUP.
IM	(Instant Messaging) Интерактивный обмен текстовыми сообщениями, происходящий между группой участников в режиме, близком к реальному времени
Interworking indicator	Индикатор наличия взаимодействия, поле параметра BCI
ISUP	Integrated Services Digital Network (ISDN) User Part – протокол пользовательского уровня в стеке ОКС №7
MG	Media Gateway – шлюз для преобразования медиа-информации из формата ТфОП в формат, принятый в сети SIP и наоборот. Может работать как шлюз сигнализации.
MGC	Media Gateway Controller – контроллер шлюзов. Также может выполнять функции шлюза сигнализации.
NAPTR	Naming Authority Pointer – процедура преобразования номера в формат SIP URL на DNS сервере.
NCI	Nature of Connection Indicator – параметр сообщений ISUP.
OCN	Original Called Number – параметр сообщений ISUP.
option-tag	Уникальный идентификатор новых функциональных возможностей для SIP в соответствии с расширениями SIP, определёнными в дополнительных RFC.

path MTU	(path Maximum Transfer Unit) Максимальная единица передачи на пути, другими словами это максимально допустимый размер передаваемого от источника к получателю пакета, который может быть доставлен без фрагментации на любом из участков пути. Path MTU имеет значение наименьшей из MTU участков пути.
registrar	Регистрирующий сервер в сети SIP. Предназначен для создания, изменения и удаления информации в базе данных, указывающей текущее местоположение пользователя.
remote target	Текущий адрес удалённого пользователя. Является одним из компонентов, описывающих состояние диалога.
route set	(Установленный маршрут) – это перечень адресов серверов, через которые должен пройти запрос, отправленный второму участнику диалога. Является одним из компонентов, описывающих состояние диалога.
Session Initiation Protocol for Telephones (SIP-T)	Протокол инициализации сессии для телефонии (SIP - T)
strict router	Прокси-сервер, стирающий содержимое поля Request-URI при наличии в запросе заголовка Route.
TMR	Transmission Medium Requirement – параметр сообщений ISUP.
TNS	Transit Network Selection – параметр сообщения ISUP.

Список литературы

12. Rosenberg J., Schulzrinne H., Camarillo G., Johnston A., Peterson J., Sparks R., Handley M., E. Schooler "SIP: Session Initiation Protocol", RFC 3261

13. M.Handley, H. Schulzrinne, E. Schooler, J. Rosenberg " SIP: Session Initiation Protocol" RFC 2543

14. J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, L. Stewart "HTTP Authentication: Basic and Digest Access Authentication" RFC 2617

15. Donovan S. "The SIP INFO Method", RFC 2976

16. J. Rosenberg, H. Schulzrinne "Reliability of Provisional Responses in SIP" RFC 3262

17. A. B. Roach "SIP - Specific Event Notification" RFC 3265

18. J. Rosenberg "SIP UPDATE Method" RFC 3311

19. R. Sparks "SIP Refer Method" RFC 3515

20. A. Johnston, A. Johnston, R. Sparks, C. Cunningham, K. Summers "SIP Basic Call Flow Examples" RFC 3665

21. J. Rosenberg, H. Schulzrinne "SIP: Locating SIP Servers" RFC 3263

22. J. Peterson " Privacy Mechanism for SIP" RFC 3323

23. H. Schulzrinne, D. Oran, G. Camarillo "The Reason Header Field for SIP" RFC 3326

24. G. Camarillo, W. Marshall, J. Rosenberg "Integration of Resource Management and SIP" RFC 3312

25. W. Marshall "Private SIP Extensions for Media Authorization" RFC 3313

26. C. Jennings, J. Peterson, M. Watson "Private Extensions to SIP for Asserted Identity within Trusted Networks" RFC 3325

27. D. Willis, B. Hoeneisen "SIP Extension Header Field for Registering Non-Adjacent Contacts" RFC 3327

28. J. Arkko, V. Torvinen, G. Camarillo, A. Niemi, T. Haukka "Security Mechanism Agreement for SIP" RFC 3329

29. B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, D. Gurle "SIP Extension for Instant Messaging" RFC 3428

30. M. Garcia-Martin, E. Henrikson, D. Mills "Private Header (P-Header) Extensions to SIP for

the 3rd-Generation Partnership Project (3GPP)" RFC 3455

31. W. Marshall, F. Andreasen "Private SIP Proxy-to-Proxy Extensions for Supporting the PacketCable Distributed Call Signaling Architecture" RFC 3603

32. D. Willis, B. Hoeneisen "SIP Extension Header Field for Service Route Discovery During Registration" RFC 3608

33. Гольдштейн Б.С. Пинчук А.В. Суховицкий А.Л. IP – телефония. М.: Радио и связь, 2001.