

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Российский технологический университет»**

**МИРЭА**

Институт кибернетики

**Кафедра информационной безопасности**

ДОКЛАД

по дисциплине

«Криптографические протоколы»

На тему:

**«Протоколы защиты текстовой/голосовой/видео связи»**

**Подготовили**

студенты групп ККСО−01/03−14

С.А. Крюков

К.И. Нарежный

А.С. Першин

**Руководитель работы**

А.П. Никитин

Москва, 2019

Оглавление

[Протоколы защиты текстовой связи 3](#_Toc5216850)

[Signal 3](#_Toc5216851)

[MTProto 4](#_Toc5216852)

[EL337 5](#_Toc5216853)

[Протоколы защиты звуковой связи 6](#_Toc5216854)

[Протоколы защиты видеосвязи 7](#_Toc5216855)

[Протокол Skype 7](#_Toc5216856)

[Протокол Tox 8](#_Toc5216857)

[Протокол SRTP 11](#_Toc5216858)

[Протокол ZRTP 13](#_Toc5216859)

# **Протоколы защиты текстовой связи**

## ***Signal***

Протокол Signal — криптографический протокол, созданный компанией Open Whisper Systems для обеспечения сквозного шифрования голосовых вызовов, видеозвонков и мгновенных сообщений. Протокол был разработан Open Whisper Systems в 2013 году и впервые был представлен в приложении TextSecure с открытым исходным кодом, которое позже было объединено в приложение Signal. Этот протокол используют во многих сторонних приложениях: WhatsApp, Facebook Messenger, Viber, Google Allo, G Data Secure Chat — все они используют оригинальную или слегка модифицированную версию Signal Protocol, иногда давая им собственные названия. Например, у Viber это протокол Proteus — по сути, тот же Signal с другими кpиптографическими примитивами.

Протокол сочетает в себе алгоритм управления ключами Double Ratchet Algorithm и расширенный протокол тройного обмена ключами Диффи-Хеллмана (3-DH) и использует Curve25519 , AES-256 и HMAC-SHA256 в качестве примитивов.

## ***MTProto***

Для достижения этой безопасности Telegram использует собственную разработку — криптографический протокол MTProto.

Прежде чем сообщение (или составное сообщение / сообщение из нескольких частей) будет передано по сети с использованием транспортного протокола, оно зашифровывается определённым образом, и вверху сообщения добавляется внешний заголовок, который представляет собой: 64-битный идентификатор ключа (который уникально идентифицирует ключ авторизации для сервера, а также для юзера) и 128-битный ключ сообщения.

Ключ юзера вместе с ключом сообщения определяет актуальный (текущий) 256-битный ключ и 256-битный вектор инициализации, который шифрует сообщение, используя AES-256 шифрование с расширением неопределённого искажения. Обратите внимание, что часть сообщения, которая должна быть зашифрована, содержит переменные данные (сессию, ID сообщения, порядковый номер, соль сервера), которые явно оказывают влияние на ключ сообщения (и таким образом на ключ AES и iv). Ключ сообщения определяется 128 битами нижнего порядка от SHA1 тела сообщения (включая сессию, ID сообщения, и т. д.) Составные сообщения шифруются как одно сообщение.

Более подробно описание протокола вы можете найти на <https://tlgrm.ru/docs/mtproto/description>

Если говорить о конфиденциальности передаваемой информации через этот сервис, то стоит обратить внимание на документацию. В документации написано “Ключ авторизации это 2048-битный ключ, которым обмениваются девайс клиента и сервер, созданный непосредственно во время регистрации юзера на устройстве клиента, чтобы обмениваться ключами Диффи-Хеллмана, и никогда не передаваемый через сеть. Каждый ключ авторизации существует только для конкретного пользователя. Ничто не мешает юзеру иметь несколько ключей (которые согласовываются с «перманентными сессиями» на разных девайсах), некоторые из них могут быть заблокированы навсегда, если девайс утерян.” Данный алгоритм выработки используется в этом протоколе и должен обеспечивать конфиденциальность сообщений. Остается вопрос каким образом ключи согласовываются с «перманентными сессиями» на разных девайсах и зачем это сделано… Ведь что получается в итоге, Telegram накрутили множество алгоритмов, исключили возможность перехвата и подмены трафика, а про банальный пароль забыли. Злоумышленнику не нужно слушать трафик мессенджера, а всего лишь нужно перехватить смс и доступ получен без проблем.

## ***EL337***

Eleet Private Messenger — мессенджер популярный среди пользователей мобильных устройств, ценящих свою приватность. Регистрация в приложении в отличие от Viber, WhatsApp, Skype, Telegram не требует подтверждения e-mail, привязки к номеру мобильного телефона или социальным аккаунтам. Это гарантирует защищенность персональных данных пользователей и тайну личной переписки. Для поиска собеседника вам нужно знать его ID или отсканировать QR код.

Все сообщения в приложении Eleet Private Messenger передаются в зашифрованном виде и не хранятся на сервере. В приложении применяются технологии шифрования от устройства к устройству, которые не оставляют следов на серверах и умеют самоликвидироваться в назначенное пользователем время. Теперь вы можете свободно общаться с людьми в любой точке земного шара, не опасаясь утечки информации. Все сообщения в приложении передаются в зашифрованном виде и не хранятся на сервере. Для защиты данных пользователей применяется протокол EL337, это усовершенствованный вариант протокола TextSecure. Во время каждого сеанса использования приложения у пользователя генерируется несколько пар ключей, один из которых открытый, другой закрытый. Открытые ключи можно пересылать другим пользователям, закрытый всегда остается на устройстве пользователя. При общении с собеседником благодаря специальному алгоритму создается общий ключ шифрования. Чтобы исключить подмену ключей, пользователь имеет возможность сравнить цифровой отпечаток ключа с тем, что ему передал собеседник.

Для такой безопасности используют асимметричную криптографию на эллиптических кривых Curve25519 и Ed25519, поточный шифр XSalsa20 аутентифицированный алгоритмом Poly1305 и хеш Blake2b, который быстрее SHA3 и при этом такой же безопасный. К тому же реализация не восприимчива к timing атакам, а значит, ключи нельзя узнать даже с помощью специальных средств.

# **Протоколы защиты звуковой связи**

# **Протоколы защиты видеосвязи**

## ***Протокол Skype***

Протокол Skype – это закрытый протокол интернет–телефонии, используемый продуктом Skype. Спецификации данного протокола никогда не были опубликованы Skype, а официальные приложения, использующие протокол, являются закрытыми.

Сеть Skype не совместима с большинством других сетей VoIP без надлежащего лицензирования от Skype (например, комерческое решение от Skype: SkyStone – выделенный шлюз на несколько каналов). Были предприняты многочисленные попытки изучить и/или перестроить протокол, чтобы выявить принцип его работы, а также исследовать на безопасность.

Видео зашифровано с помощью поточного шифра RC4. Голосовые данные шифруются с помощью AES.

Skype был первой одноранговой IP – телефонной сетью. Раньше, любой клиент имел хорошую пропускную способность видеосвязи за счет технологии p2p. В 2012 году Microsoft изменила дизайн сети и передала все суперноды под свой контроль в качестве размещенных серверов в центрах обработки данных. Microsoft, в то время, защищал свой принцип, считая, что такой подход имеет не менее быструю пропускную способность, масштабируемость и преимущества доступности для сотен миллионов пользователей, которые составляют сообщество продукта Skype". В то время среди узкого круга исследователей была некоторая озабоченность относительно последствий изменения конфиденциальности из–за централизации супернодов. Позже это было доказано публикацией в открытых источниках деятельности программы наблюдения PRISM в июне 2013.

## ***Протокол Tox***

Tox – это одноранговый протокол прикладного уровня, созданный для обмена мгновенными сообщениями и видеосвязи, который обеспечивает сквозное шифрование. Заявленная цель проекта – обеспечить безопасное, но легкодоступное общение для всех. Эталонная реализация протокола публикуется как свободное программное обеспечение с открытым исходным кодом.

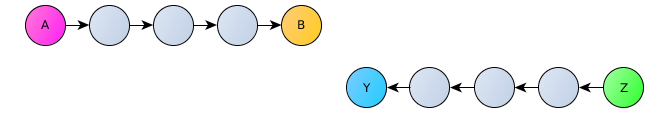
Взаимодействие участников Tox реализовано с использованием пирингового обмена информацией для улучшения пропускной способности, но в отличие от Skype, не требует регистрации для использования, а идентификатор пользователя создаётся локально. После установки Tox автоматически создаётся пара ключей. Публичный ключ можно передавать кому угодно – он служит как уникальный идентификатор для поиска собеседника. Секретный ключ хранится только у владельца и подтверждает его подлинность, не раскрывая персональные данные. Центральный сервер отсутствует, поиск собеседников происходит через DHT сеть.

DHT – это класс децентрализованных распределённых систем поисковой службы, работающей подобно хеш–таблице. Как структура данных, хеш–таблица может представлять собой ассоциативный массив, содержащий пары (ключ–значение). Также, с термином DHT связан ряд принципов и алгоритмов, позволяющих записывать данные, распределяя информацию среди некоторого набора узлов–хранителей, и восстанавливать их путём распределённого поиска по ключу. Особенностью распределённой таблицы является возможность распределить информацию среди некоторого набора узлов–хранителей таким образом, что каждый участвующий узел смог бы найти значение, ассоциированное с данным ключом. Ответственность за поддержание связи между именем и значением распределяется между узлами, в силу чего изменение набора участников является причиной минимального количества разрывов. Это позволяет легко масштабировать DHT, а также постоянно отслеживать добавление и удаление узлов и ошибки в их работе.

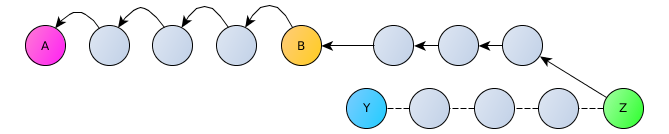
В отличии от DHT–нод, вся информация о которых известна или может быть получена любым клиентом DHT–сети Tox, клиентские приложения скрыты от стороннего наблюдателя – простого знания ToxID контакта (содержащего его публичный ключ) недостаточно для того, чтобы установить местонахождение этого контакта. Для соединения двух приложений Tox используется механизм луковой маршрутизации («onion routing»).

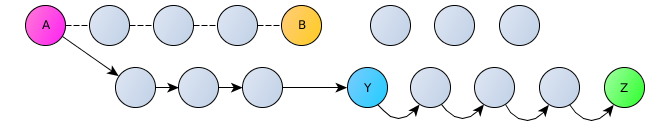
Установка связи

Механизм установления связи между двумя клиентами можно представить следующим образом. Два клиента (A и Z) анонсируют свой публичный ключ на ближайших (для своего публичного ключа) нодах через три случайные промежуточные DHT–ноды, каждая из которых знает только своих соседей по маршрутизации пакета, но не может прочитать содержимое пакета.

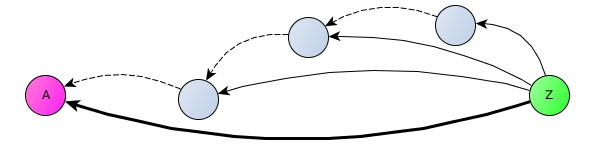


Второй клиент (Z), желающий соединиться с первым (A), так же через три случайные DHT–ноды делает запрос на установку соединения на ближайшую к искомому ключу (A) ноду, которая знает об анонсе первого клиента и маршруте, по которому требуется передать запрос от второго клиента.

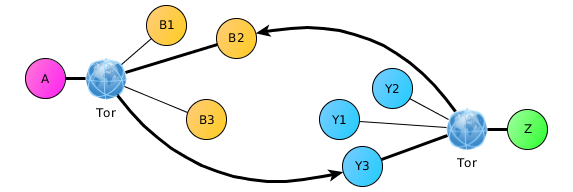


Первый клиент (A) в случае принятия запроса на установку соединения, проделывает обратную операцию к ближайшей DHT–ноде второго клиента (Z).  


После обмена информацией о метоположении друг–друга и временных ключах клиенты могут соединиться напрямую.



Если клиенты не желают делиться информацией о своем местоположении даже со своими контактами, они могут использовать ноды поддерживающие TCP–relay через прокси типа Tor.



Tox использует криптографические примитивы, присутствующие в криптобиблиотеке NaCl. В частности, Tox использует Curve25519 для обмена ключами (ECDH), алгоритм xsalsa20 для поточного шифрования и алгоритм Poly1305 для подсчета MAC (massage access code – режима имитовставки).

Данный протокол не проходил аудит на безопасность специалистами по защите информации. Ведутся его активные разработки, например, на безопасном языке Rust

## ***Протокол SRTP***

Secure Real–time Transport Protocol – Безопасный протокол передачи данных в реальном времени (или SRTP) определяет профиль RTP (транспортный протокол в реальном времени) и предназначен для шифрования, установления подлинности сообщения, целостности, защиту от замены данных RTP в однонаправленных и multicast передачах медиа и приложениях. Он был разработан небольшой командой криптоэкспертов IP протоколов компаний Cisco и Ericsson в составе David Oran, David McGrew, Mark Baugher, Mats Naslund, Elisabetta Carrara, Karl Norman, и Rolf Blom. Был впервые опубликован в IETF в марте 2004 как RFC 3711.

Для шифрования медиа потока (в целях конфиденциальности соединения), SRTP стандартизирует использование только единственного шифра, AES, который может использоваться в двух режимах, превращающих изначально блочный шифр AES в потоковый шифр:

1. Сегментированный целочисленный счётчик – типичный режим, который осуществляет произвольный доступ к любым блокам, что является существенным для трафика RTP, передающегося в публичных сетях с непредсказуемым уровнем надежности и возможной потерей пакетов. В общем случае почти любая функция может использоваться в роли «счётчика», предполагая, что эта функция не повторяется для большого числа итераций. Но стандарт для шифрования данных RTP – только обычное целочисленное значение счётчика. AES, работающий в этом режиме, является алгоритмом шифрования по умолчанию, с длиной шифровального ключа по умолчанию в 128 бит и ключом сессии длиной в 112 бит.
2. f8–режим – вариант режима способа обратной связи, расширенного, чтобы быть доступным с изменённой функцией инициализации. Значения по умолчанию для шифровального ключа и ключа сессии – то же, что и в AES в режиме, описаном выше. (AES, работающий в этом режиме, был выбран для использования в мобильных сетях передачи 3G UMTS.)

Хотя технически в SRTP можно легко встраивать новые алгоритмы шифрования, стандарт SRTP заявляет, что новые алгоритмы шифрования, помимо описанных, не могут просто быть добавлены в реализацию протокола SRTP. Единственный юридически легальный способ добавить новый алгоритм шифрования для совместимости со стандартом SRTP, состоит в том, чтобы опубликовать новый стандарт RFC, где должно быть ясно определено использование нового алгоритма.

Чтобы подтвердить подлинность сообщения и защитить его целостность, алгоритм хеширования HMAC–SHA1, определенный в RFC 2104, используется для получения 160–битового хэша, который затем урезается до 80 или до 32 битов, чтобы стать опознавательным признаком, включаемым в пакет. HMAC вычисляется по типу payload пакета и данных в заголовке пакета, включая номер последовательности пакета. Чтобы защитить против встраивания сообщений Человека\_посередине, приёмник поддерживает индексы ранее полученных пакетов, сравнивает их с индексом каждого нового полученного пакета и пропускает новый пакет, только если он не проигрался (то есть был послан), прежде. Такой подход в большой степени полагается на полную защиту целостности (чтобы лишить возможности изменить индексы последовательности пакетов для обмана).

SRTP полагается на внешний протокол обмена ключами, чтобы установить главный начальный ключ. Разработан специальный протокол для защиты видеосвязи с использованием SRTP – ZRTP.

## ***Протокол ZRTP***

ZRTP – криптографический протокол согласования ключей шифрования, используемый в системах передачи голоса/видео по IP–сетям (VoIP). ZRTP описывает метод получения ключей по алгоритму Диффи – Хелмана для организации Secure Real–time Transport Protocol (SRTP). ZRTP осуществляет согласование ключей в том же потоке RTP, по которому установлена аудио/видео связь, то есть не требует отдельного канала связи. Разработан Филипом Циммерманом (Phil Zimmermann, автор Pretty Good Privacy), Джоном Калласом (Jon Callas) и Аланом Джонстоном (Alan Johnston) в 2006 году.

Протокол RTP (англ. Real–time Transport Protocol) работает на прикладном уровне (OSI – 7) и используется при передаче трафика реального времени. Протокол был разработан Audio – Video Transport Working Group в IETF и впервые опубликован в 1996 году как RFC 1889 (RFC 1889 является устаревшим с момента выхода RFC 3550 в 2003 году).

Протокол RTP переносит в своём заголовке данные, необходимые для восстановления аудиоданных или видеоизображения в приёмном узле, а также данные о типе кодирования информации (JPEG, MPEG и т. п.). В заголовке данного протокола, в частности, передаются временная метка и номер пакета. Эти параметры позволяют при минимальных задержках определить порядок и момент декодирования каждого пакета, а также интерполировать потерянные пакеты.

SAS

Сам по себе алгоритм обмена ключами Диффи – Хелмана не может обеспечить защиту от присутствия человека посередине (man in the middle). Для аутентификации ZRTP использует Short Authentication String (SAS) "короткую строку аутентификации" (4 старших байта хеша), являющуюся сокращённым представлением криптографического хеша полученных ключей Диффи – Хелмана. Значения SAS вычисляются на каждой стороне соединения, абоненты передают их друг другу голосом для сверки. Если значения не совпали, то с большой уверенностью можно предположить присутствие Человека\_посередине. Использование алгоритма Диффи – Хелмана даёт потенциальному Человеку\_посередине всего одну попытку сгенерировать правильную SAS при попытке атаки.

Непрерывность ключевого материала

Вторым уровнем защиты ZRTP от атак "человек посередине" является непрерывность ключевого материала. Хеш ключевой информации предыдущего вызова подмешивается к параметрам алгоритма Диффи – Хеллмана при следующем вызове (между этими же абонентами), что придаёт протоколу ZRTP сходство с SSH. Если "человека посередине" не было при первом вызове, он исключается из всех последующих.

Данный протокол является стандартизованным, а значит он проходил аудит безопасности работы.