ПРОТОКОЛЫ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ IP-ТЕЛЕФОНИИ

М.Ковцур
maxkovzur@mail.ru

Актуальность защищенной передачи информации в сетях VoIP возрастает по мере развития IP-телефонии. Для обеспечения безопасности IP-телефонии разработан ряд методов, которые рассмотрены в предлагаемой статье. Особое внимание автор уделяет протоколу распространения ключей ZRTP как одному из наиболее перспективных и эффективных.

Защита информации в VoIP-сетях

VoIP-сервисы все шире применяются в сетях передачи данных. Оборудование для передачи голосовой информации по IP-сетям производит ряд широко известных компаний, таких как Cisco, LinkSys (Cisco), AddPac, GrandStream, D-Link и др. Для установления и поддержания соединения в IP-телефонии используются протоколы сигнализации SIP, H.323, MGCP, H.248.

Наиболее распространенный сегодня протокол SIP описан в рекомендации RFC 3261 [1]. Он отличается простотой реализации, гибкостью и расширяемостью. Изначально в SIP было определено всего шесть сообщений, которых было достаточно для установления и поддержания соединения. В процессе развития их число выросло до 14.

Голос в сетях IP-телефонии передается при помощи протоколов RTP/RTCP (Real-time Transport Protocol и Real-Time Transport Control Protocol), описанных в рекомендации RFC 3550 [2]. Протокол RTP используется для инкапсуляции в IP-пакеты голосовой и медиаинформации, а RTCP служит для передачи управляющей информации, контроля качества передачи и обеспечения обратной связи для RTP-потока. При звонке одного корреспондента другому вначале отрабатывает протокол SIP, позволяющий установить соединение между корреспондентами. Как только один из корреспондентов снимает трубку, начинает работу протокол RTP/RTCP.

В силу общедоступности используемых каналов передачи голосовой информации в IP-сетях особую актуальность приобретает конфиденциальность VoIP-сервисов. Для этого возможны два различных подхода: формирование прямого защищенного канала между корреспондентами (например, VPN-туннель) и применение специальных протоколов обеспечения безопасности для IP-сервисов. Первый способ широко распространен при построении виртуальных корпоративных сетей. Однако для его реализации корреспонденты должны поддерживать VPN-протокол, что свойственно далеко не всем VoIP-устройствам (табл.1).

Специальные протоколы обеспечения безопасности IP-телефонии можно разделить на три категории:

- протоколы защиты сигнализации (Secured SIP);

Таблица 1. Оборудование защищенной IP-телефонии

Производитель	Продукт	Реализация	Протоколы			Поддержка VPN
			защиты установки соединения	защиты медиатрафика	распределения ключей	
D-Link	DVG-5008S	Аппаратная	SIP/TLS	Нет данных	SDES	PPTP
AddPack	AP200	Аппаратная	SIP/TLS	SRTP	SDES	Нет данных
LinkSys	Cisco SPA112	Аппаратная	SIP/TLS	SRTP	SDES	Нет
Grandstream	GXW400x	Аппаратная	SIP/TLS	SRTP	SDES	
UM-Labs	RC-2100 SIP Security Controller	Аппаратная	SIP/TLS	SRTP	ZRTP/SDES	
CounterPath	Eye-beam	Программная	SIP/TLS	SRTP	SDES	
3XC	3CX softphone	Программная	SIP/TLS	SRTP	Нет данных	
Asterisk	IP PBX	Программная	SIP/TLS	SRTP	SDES/ ZRTP	
FreeSwitch	IP PBX	Программная	SIP/TLS	SRTP	SDES / ZRTP	
Phoner	Phoner softphone	Программная	SIP/TLS	SRTP	ZRTP	

- протоколы защиты медиаинформации (SRTP);
- протоколы генерации/распределения ключей для протоколов защиты медиаинформации (MIKEY, SDES, ZRTP, DTLS).

Протоколы защиты сигнализации предназначены для защиты информации о телефонных номерах звонящего и вызываемого абонентов, а также о поддерживаемых кодеках. Для этого используется протокол Secured SIP (SSIP, SIP/TLS). Он работает по аналогии с протоколом HTTPS, когда между корреспондентом и сервером организовывается SSL-туннель с использованием сертификатов и открытого ключа. Все SIP-сообщения (сигнализация) передаются по этому туннелю.

Для безопасной передачи речи и медиаданных широко используется защищенный

протокол реального времени – Secure Real-time Transport Protocol (SRTP) [3]. Он реализует функции криптографической защиты (шифрование и аутентификацию) речевых сообщений на основе алгоритма шифрования AES. Криптографическая защита пакетов голосовой информации выполняется в режиме реального времени и не вносит изменений в вероятностно-временные характеристики протокола RTP. Но для работы этого алгоритма необходимо предварительно сформировать криптографические ключи. Эту задачу решает протокол распределения ключей.

Основные задачи протокола SRTP:

- шифрование передаваемых голосовых данных;
- аутентификация передаваемых сообщений;
- защита от передачи повторных пакетов;
- сохранение полосы пропускания, сжатие RTP-заголовков.

Протокол SRTP содержит две составляющие – собственно протокол SRTP для переноса и криптозащиты медиаданных и протокол SRTCP (Secure Real-time Transport Control Protocol) для управления медиасессией.

Алгоритмы аутентификации и шифрования могут выполняться независимо друг от друга. Поэтому возможен вариант, когда шифрование выключено и SRTP применяется только для аутентификации. Однако аутентификация сообщения в SRTP обязательна и не может быть отключена.

Протоколы третьей группы предназначены для генерации и распределения ключей шифрования медиаинформации между корреспондентами. Для решения этой задачи могут использоваться протоколы MIKEY, SDES, ZRTP, DTLS.

Протокол обмена ключами MIKEY описан в рекомендациях RFC3830 [4] и RFC6043. MIKEY имеет несколько режимов работы, определяющих способ формирования секретного ключа сессии SRTP, – режим предустановленного ключа, режим открытого ключа и режим Диффи–Хелмана. Причем второй и третий режимы не защищают от атаки вторжения в середину (MiTM, Man In the Middle) и требуют реализации механизма аутентификации сообщений. Транспортом для переноса сообщений протокола может выступать как SIP/SDP-сообщения, так и протокол RTSP (Real Time Streaming Protocol).

Протокол SDES описан в RFC4568 [5]. Его суть: один из корреспондентов передает ключ в SIP-сообщении по сигнальному каналу. Корреспондент получает его и использует для шифрования. Однако при этом обмен сигнальными сообщениями должен быть защищен от злоумышленника. По этой причине SDES можно использовать только при наличии SIP/TLS защищенного соединения с цифровым сертификатом сервера. Также данный способ не обеспечивает безопасности "из конца в конец". Это означает, что если соединение устанавливается, например, IP-АТС, SDES будет распределять ключи между корреспондентом А и IP-АТС и между корреспондентом Б и IP-АТС, но не напрямую между корреспондентами А и Б.

Протокол DTLS для SRTP специфицирован в RFC 5764 [6]. Он описывает формирование медиасессий "точка-точка" с жесткой фиксацией портов UDP корреспондента/респондента. Сообщения протокола передаются совместно с RTP-пакетами. Для организации сессии (DTLS-ассоциации) корреспонденты обмениваются сообщениями. Так как в основе протокола DTLS лежит TLS (Transport Layer Security), использующий инфраструктуру открытых ключей (PKI), то применение DTLS возможно тоже только при наличии PKI.

Одним из наиболее перспективных протоколов генерации ключей является ZRTP [7, 8]. Он применяется в приложении CsipSimple для ОС Android, в программных телефонах Jitsi, Phoner, в программных АТС FreeSwitch и Asterisk, в аппаратных VoIP-шлюзах компании UM-Labs. Отличительная особенность протокола ZRTP – возможность обеспечить безопасность "из конца в конец", от одного корреспондента до другого.

Особенности протокола ZRTP

Для работы протокола SRTP корреспондентам, участвующим в обмене, необходим ключевой материал. Спецификация SRTP описывает получение ключевого материала на основе мастер-ключа и дополнительного ключа сессии, но не описывает обмен ключами между пользователями. Для этого предлагается использовать отдельно разработанные протоколы. Одним из таких протоколов, позволяющих выполнить обмен ключами для работы SRTP между корреспондентами по открытым каналам связи, является протокол ZRTP. В задачу этого протокола входят:

- генерация ключевых параметров SRTP-сессии;

- обеспечение конфиденциальности сообщений протокола;
- аутентификация корреспондентов;
- защита от атаки вторжения посередине как с использованием, так и без использования инфраструктуры открытых ключей.

При установлении защищенного VoIP-соединения протокол ZRTP работает сразу после завершения работы протокола SIP. И только после успешного завершения работы протокола ZRTP начинается передача медиатрафика по протоколу SRTP.

Протокол ZRTP предусматривает работу корреспондентов по схеме "точка-точка", при этом отдельно выделена возможность применения протокола при многопоточном режиме, когда необходимо организовать несколько защищенных медиапотоков (например, голос и видео). Также предусмотрен вариант работы с легитимным посредником, которым может выступать, например, корпоративная телефонная станция. Для работы по протоколу ZRTP каждый из корреспондентов должен иметь предустановленный уникальный идентификатор (ZID), а оборудование – поддерживать одинаковые наборы криптографических алгоритмов (табл.2).

В основе протокола ZRTP – обмен ключами по алгоритму Диффии-Хелмана во время установления соединения. Особенность ZRTP – все данные передаются внутри RTP-подобных пакетов (рис.1). Благодаря этому ZRTP несовместимые устройства просто отклоняют ZRTP-пакеты, поэтому они не влияют на установленное соединение.

Для аутентификации корреспондентов, а также исключения атаки вторжения в середину (MiTM, Man in The Middle) протокол ZRTP предусматривает использование короткой аутентификационной строки (SAS, Short Authentication String). SAS рассчитывается обоими

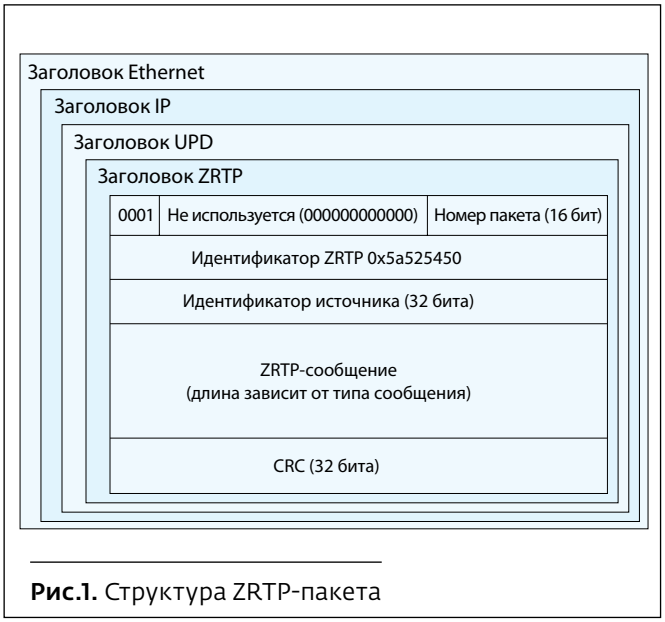

Рис.1. Структура ZRTP-пакета

корреспондентами по специальному алгоритму и может быть использована для вербального сравнения непосредственно в ходе соединения (увидел-произнес-сравнили). Для контроля целостности каждое передаваемое сообщение ZRTP содержит проверочный код CRC, а также код аутентификации сообщения MAC (Message Authentication Code), вычисляемый посредством хеш-функций. В качестве аргумента хеш-функции используется само защищаемое сообщение, ключом служит специальный параметр, передаваемый (для большинства сообщений) в следующем сообщении. Алгоритм хеширования согласовывается на первой фазе протокола (см. табл.2). Ошибка в хеш-сообщении, как правило, означает обнаружение атаки MiTM, так как искажения за счет канальных ошибок проявляются при проверке контрольной суммы CRC пакета ZRTP.

Протокол выполняется последовательно в четыре фазы: обнаружение, подтверждение,

Таблица 2. Криптографические наборы протокола ZRTP

Криптографические функции	Обязательная поддержка		Опциональная поддержка	
	Алгоритм	Длина ключа, бит	Алгоритм	Длина ключа, бит
Шифрование данных	AES	128	AES	256
			TwoFish	128 192 256
Аутентификация сообщений	HMAC-SHA1 HMAC-SHA1	32 80	Skein-512-MAC	32 64
Хеш-функция	SHA-256 SHA-384	256 384	NIST SHA-3	256 384
Обмен ключами	DH 3k	–	EC 25, EC 38 DH 2k	–
Аутентификация корреспондентов	PGP X.509v3	–	–	–

вычисление ключей и завершение. Рассмотрим процедуру взаимодействия корреспондентов.

Фаза 1 – обнаружение

В фазе 1 протокола корреспонденты обмениваются сообщениями Hello, которые подтверждаются ответным сообщением HelloACK. Сообщения Hello содержат информацию о поддерживаемых партнерами криптографических примитивах, необходимую для определения возможности использования SRTP. Как правило, это поддерживаемые алгоритмы хеширования, алгоритмы шифрования, типы аутентификационных тегов, протоколы согласования ключей, типы SAS. Также передаются информация о версии ZRTP, идентификатор производителя и версия программного обеспечения ZRTP, набор флагов для различных операций.

Для перехода протокола в следующую фазу оба корреспондента должны получить сообщение Hello и хотя бы один из них – сообщение HelloACK.

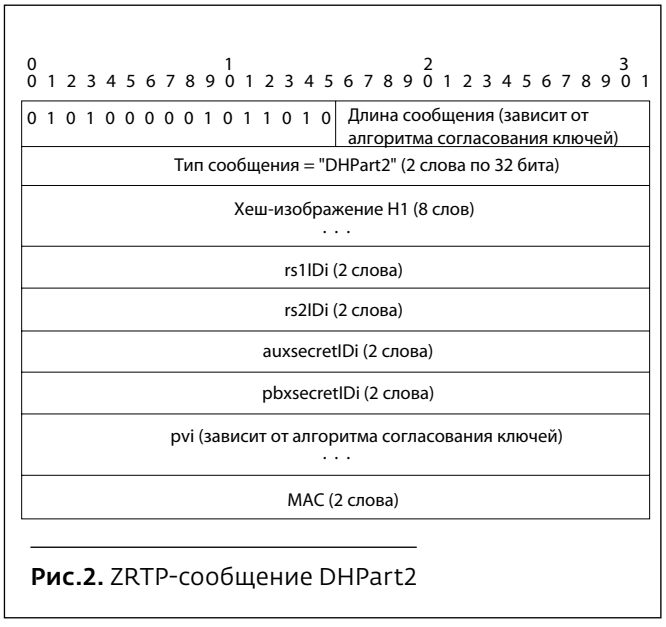

Рис.2. ZRTP-сообщение DHPart2

Фаза 2 – подтверждение

В начале фазы 2 определяется, кто будет инициатором при выполнении алгоритма Диффи-Хелмана, а кто – респондентом (отвечающим). Для этого служит сообщение Commit, которое посылает устройство-инициатор. Сообщение содержит значение hvi (hash value of initiator). В том случае, если оба устройства выбирают роль инициатора и одновременно посылают Commit, значения hvi сравниваются, и тот, у кого hvi оказывается больше, принимает роль инициатора, а другое устройство – респондента.

Для вычисления hvi используется предварительно сформированное сообщение DHPart2 (рис.2). Значение hvi вычисляется, как укороченная до 256 бит хеш-функция от конкатенации сообщения DHPart2 и сообщения Hello респондента:

hvi=hash(DHPart2 || Hello респондента),

где hash – согласованная на первой фазе хеш-функция; || обозначает операцию конкатенации.

Помимо выбора одного из корреспондентов в качестве инициатора, назначение параметра hvi – проверка правильности успешной SAS-аутентификации и проверка правильности DHPart2-сообщения (защита от модификации злоумышленником).

Сообщения DHpart устройства формируют до начала обмена: DHpart1 – от респондента и DHpart2 – от инициатора. В них передаются

необходимые для формирования ключа параметры.

Для защиты от атаки MiTM-протокол использует данные, накопленные в предыдущих соединениях. Для этого используется специальная таблица (кеш) в памяти устройств. В таблице располагаются некие значения rs1, rs2, pbxsecret и auxsecret. По окончании каждой сессии они обновляются по определенным правилам. На основании этих локальных значений инициатор посредством хеш-функции рассчитывает параметры rs1IDi, rs2IDi, auxsecretIDi, и pbxsecretIDi, респондент – rs1IDr, rs2IDr, auxsecretIDr и pbxsecretIDr. Например, rs1IDr = MAC (rs1, Responder), где MAC – стандартная хеш-функция, Responder – ее ключ. Значение сокращается до 64 бит.

Эти значения передаются в сообщения DHpart1 (от респондента) и DHpart2 (от инициатора). Вид сообщения DHpart2 представлен на рис.2. Полученные в сообщениях DHpart-значения каждое устройство сравнивает с рассчитанными локально и на основании сравнения по определенным правилам формирует набор параметров s1, s2 и s3:

s1=F(rs1IDi, rs2IDi);

s2=F(auxsecretIDi);

s3=F(pbxsecretIDi),

где F – функция сравнения.

Например, s2=auxsecret, если auxsecretIDi= auxsecretIDr, иначе s2=NULL.

Кроме того, перед фазой 2 вычисление ключей производится по алгоритму Диффи-Хелмана. Для этого каждый из корреспондентов генерирует свое случайное число (svi – инициатор и svr – респондент), после чего вычисляет открытые ключи pvi = q^{svi} mod p и pvr = q^{svr} mod p (здесь q и p – известные всем пользователям протокола числа, mod p – значение по модулю p). Открытые ключи pvr и pvi передаются в сообщениях DHPart1 и DHPart2, соответственно.

Таблица 3. Параметры алгоритмов согласования ключей

Алгоритм согласования ключей	длина DHR, байт	Длина модуля, бит
DH-3072	384	3072
DH-2048	256	2048
ECDH P-256	32	256
ECDH P-384	48	384

Фаза 3 – вычисление ключей

Оба корреспондента используют полученные открытые ключи pvi и pvr для расчета результирующего ключа обмена Диффи-Хелмана. Для этого респондент вычисляет результирующее значение DHR = pvi^{svr} mod p, а инициатор – DHR = pvr^{svi} mod p (очевидно, что результат будет одинаков). Параметр p выбирается исходя из выбранного алгоритма согласования ключей (табл.3).

Также вычисляется параметр total_hash – полная цифровая подпись сообщений, используемых в обмене:

total_hash = hash(Hello of responder || Commit || || DHPart1 || DHPart2).

Секретный ключ вычисляется как

s0=hash(1 || DHR || "ZRTP-HMAC-KDF" || ZIDi || ZIDr || ||total_hash || len(s1) || s1 || len(s2) || s2 || len(s3) || s3),

где len(s1), len(s2), len(s3) – 32-разрядные целые, соответствующие размеру в октетах параметров s1, s2, s3; "ZRTP-HMAC-KDF" – строка фиксированной длины, показывающая, какая часть s0 используется для ключа в алгоритмах

хеширования HMAC и KDF. Секретный ключ s0 используется для вычисления KDF (Key Derivation Function).

KDF – хеш-функция генерации ключей. В ZRTP она использует один из хеш-алгоритмов HMAC, согласованный корреспондентами в первой фазе выполнения протокола ZRTP. Функции KDF для протокола ZRTP не требуется генерировать ключевой материал более 256 или 384 бит. По этой причине достаточно единичного вызова HMAC-функции.

KDF определяется как
KDF(KI, Label, Context, L) = HMAC(KI, c || Label || || 0x00 || Context || L),
где KI – секретный ключ, например s0; c – значение счетчика, в данном случае c = 1, так как HMAC выполняется только один раз; Label – строка ненулевых октетов, определяющая цель генерируемого ключевого материала; 0x00 – разделитель, наличие которого является требованием по NIST SP800 108. Context-поле является конкатенацией ZIDi || ZIDr || info, где info – дополнительный материал, известный обоим корреспондентам. В качестве Context рекомендуется использовать параметр total_hash, вычисляемый после обмена открытыми ключами по Диффи-Хелмана. L – 32-разрядное целое число, не превышающее длину в битах выходного материала HMAC. Выходное значение KDF-функции усекается по крайним левым L битам.

С помощью KDF вычисляются SRTP-ключи – мастер-ключ и дополнительные ключи сессии.

Для инициатора:
SRTP мастер-ключ сессии
srtpkeyi = KDF(s0, "Initiator SRTP master key", KDF_Context, согласованная длина ключа AES);
SRTP дополнительный ключ сессии
srtpsalti = KDF(s0, "Initiator SRTP master salt", KDF_Context, 112).

Для респондента:
SRTP мастер-ключ сессии
srtpkeyr = KDF(s0, "Responder SRTP master key", KDF_Context, согласованная длина ключа AES);
SRTP дополнительный ключ сессии
srtpsaltr = KDF(s0, "Responder SRTP master salt", KDF_Context, 112),
где KDF_Context = (ZIDi || ZIDr || total_hash) – конкатенация соответствующих переменных.

Фаза 4 – завершение

Обмен завершается отправкой сообщений Confirm1 и Confirm2. Они содержат зашифрованные с помощью AES (с обратной связью по шифротексту) сообщения, включающие несколько флагов и параметров, в том числе время действия нового сгенерированного ключа, а также некоторые служебные флаги и опциональные цифровые подписи. Протокол считается завершенным, когда респондент отправляет сообщение Conf2ACT или первый SRTP-пакет с верным тегом аутентификации. При этом инициатор воспринимает верный SRTP-пакет как эквивалент получения Conf2ACK-сообщения. Таким образом, в процессе выполнения протокола ZRTP передаются несколько сообщений различной длительности (табл.4).

Кроме режима генерации ключей в результате обмена ключами Диффи-Хелмана протокол ZRTP предусматривает возможность работать в режиме предустановленного ключа (общего секрета) и в многопоточном режиме.

Режим предустановленного ключа

В этом режиме корреспонденты пропускают вычисление ключей по алгоритму Диффи-Хелмана и используют значение rs1 из кеш-памяти устройств. Выбор режима с предустановленным ключом осуществляется инициатором в сообщении Commit. В конце выполнения протокола ключ rs1 заменяется на новый.

Инициатор вычисляет preshared_key:
preshared_key = hash(len(rs1) || rs1 || len(auxsecret) || || auxsecret || len(pbxsecret) || pbxsecret).

В сообщении Commit вместо hvi передается случайная последовательность длиной в 16 октетов и keyID = MAC(preshared_key, "Prsh"), укороченный до 64 бит.

Респондент использует полученный keyID для поиска ключевого материала в своей кеш-памяти. Он сам вычисляет значения preshared_keyr на основе сохранившегося у него в памяти ключевого материала:
preshared_keyr1 = hash(len(rs1) || rs1 || len(auxsecret) || || auxsecret || len(pbxsecret) || pbxsecret);
preshared_keyr2 = hash(len(rs1) || rs1 || "0x00000000" || || len(pbxsecret) || pbxsecret);
preshared_keyr3 = hash(len(rs1) || rs1 || len(auxsecret) || || auxsecret || "0x00000000");
preshared_keyr4 = hash(len(rs1) || rs1 || "0x00000000" || || "0x00000000");
keyIDr(1...4) = MAC(preshared_keyr(1...4), "Prsh").

Если какой-либо из ключей keyIDr респондента совпадает с keyID, полученным от инициатора, респондент выбирает соответствующий

Таблица 4. Параметры сообщений протокола ZRTP

Назначение	Обозначение	Длина, бит	Подтверждение	Число передач
Согласование параметров и возможностей корреспондентов	Hello	1392	Да	20
Подтверждение получения Hello-сообщения	HelloACK	560	Нет	1
Согласование хеш-функций hvi	Commit	1392	Да	10
Первое сообщение обмена ключами Диффи-Хелмана	DHPart1	4208	–	
Второе сообщение обмена ключами Диффи–Хелмана	DHPart2	4208	Да	10
Подтверждение обмена с применением сгенерированного общего ключа	Confirm1	1072	–	
Подтверждение обмена с применением сгенерированного общего ключа	Confirm2	1072	Да	10
Подтверждение Confirm2	ConfACK	560	Нет	1

ему preshared_keyr и продолжает работу в режиме предустановленного ключа. Сообщения DHPart1 и DHPart2 в этом режиме не передаются. После сообщения Commit сразу следует вычисление SRTP-ключей и обмен сообщениями Confirm1 и Confirm2.

Промежуточные параметры вычисляются как

total_hash = hash(Hello of responder || Commit);
KDF_Context = (ZIDi || ZIDr || total_hash);
s0 = KDF(preshared_key, "ZRTP PSK", KDF_Context, согласованная длина хеш-функции).

Далее ключи вычисляются по аналогии с алгоритмом Диффи–Хелмана.

Если совпадений какого-либо из keyIDr респондента с keyID инициатора не найдено, протокол прекращает работу в режиме предустановленного ключа. Респондент должен ответить сообщением DH Commit, что сделает его инициатором, а текущий инициатор станет респондентом, так как сообщение Commit с DH-режимом всегда имеет более высокий приоритет, чем сообщение с режимом предустановленного ключа.

Многопоточный режим

Многопоточный режим используется только в том случае, если между корреспондентами существует

SRTP-медиапоток с ZRTP-ключом. Алгоритм ZRTP позволяет получить несколько SRTP-ключей из одного обмена ключами Диффи–Хелмана.

Многопоточный режим назначает инициатор, о чем он сообщает в Commit. При этом сообщения DHPart1 и DHPart2 не передаются. В отличие от режима с предустановленным ключом кеш-память не используется. Это позволяет выполнять несколько ZRTP-протоколов одновременно между корреспондентами, например, для генерации дополнительных ZRTP-ключей с целью организации видеопотока при существующем голосовом соединении.

Активный ZRTP-ключ вычисляется как
ZRTPSess = KDF(s0, "ZRTP Session Key", KDF_Context, согласованная длина хеш-функции);
s0 = KDF(ZRTPSess, "ZRTP MSK", KDF_Context, согласованная длина хеш-функции).

Далее ключи вычисляются по аналогии с режимом Диффи–Хелмана.

Время выполнения ZRTP

Для повышения вероятности успешного завершения инициализации сессии протокол ZRTP предусматривает многократную передачу сообщений по таймерам T1 и T2. Таймер T1 используется для повторной передачи сообщений типа Hello, T2 – для повторной передачи всех остальных сообщений.

Сообщение Hello может быть повторно отправлено 20 раз. При этом начальное значение T1, равное 50 мс, при повторных передачах удваивается, пока не достигнет 200 мс. Таким образом, максимальное время выполнения первой фазы протокола, без учета времени формирования, переноса и обработки сообщения, будет составлять 50+100+18·200 (мс) = 3,75 с. Повторная передача Hello-сообщения прекращается, только если принято ответное сообщение HelloACK или сообщение Commit. Последующие сообщения – Commit, DHPart2 и Confirm2 дублируются только инициатором, если соответствующий ответ DHPart1, Confirm1 и ConfACK не получен. Повторная передача Confirm2 может быть остановлена, если получен первый SRTP-пакет с верным значением AuthTag.

Все сообщения, помимо Hello, повторяются с интервалом от T2=150 мс с увеличением до 1200 мс. Максимально, без учета времени формирования, переноса и обработки, этот интервал может составить 150+300+600+7·1200 (мс) = 9,45 с. Данное время достаточно велико для телефонии. Согласно приказу №113-2007 "Требования к организационно-техническому обеспечению устойчивого функционирования сети связи общего пользования из нормативной базы отрасли "Связь" РФ" в области качества услуг, выделяют следующие нормы на время установления соединения: для сетей зоновой связи – 2,7 с, местной связи – 6,6 и для сетей междугородной и международной связи – 5,4 с.

***

Можно констатировать, что ZRTP является надежным протоколом распределения ключей для SRTP-протокола. Он позволяет обеспечивать гибкий подход к генерации ключей для протокола обеспечения конфиденциальности при передаче голосовой и медиаинформации (SRTP). Протокол обладает некоторой избыточностью, так как часть параметров введена для соответствия требованиям NITS.

Протокол распределения ключей ZRTP требует достаточно большого обмена сообщениями, время его работы зависит как от вероятности ошибок в канале связи, так и от времени задержки сообщений в этом канале. Количественная оценка характеристик времени и вероятности успешного установления защищенного соединения нуждается в проведении дополнительных исследований.

ЛИТЕРАТУРА

1. RFC 3261. **J. Rosenberg, H. Schulzrinn, G.Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler**. SIP: Session Initiation Protocol. – IETF, June 2002.
2. RFC 3550. **D.Schulzrinne, S.Casner, R.Frederick, V.Jacobson.** RTP: A Transport Protocol for Real-Time Applications. – IETF, July 2003.
3. RFC 3711. **M.Baugher D.McGrew, M.Naslund, E.Carrara, K.Norrman.** The Secure Real-time Transport Protocol (SRTP). – IETF, March 2004.
4. RFC 3830. **J.Arkko, E.Carrara, F.Lindholm, M.Naslund, K.Norrman.** MIKEY: Multimedia Internet KEYing. – IETF, August 2004.
5. RFC 4568. **F.Andreasen, M.Baugher, D.Wing.** Session Description Protocol (SDP), Security Descriptions for Media Streams. – IETF, July 2006.
6. RFC 5764. **D.McGrew, E.Rescorla.** Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP). – IETF, May 2010.
7. RFC 6189. **P.Zimmermann, A.Johnston, J.Callas.** ZRTP: Media Path Key Agreement for Unicast Secure RTP. – IETF, April 2011.
8. **Ковцур М.М., Никитин В.Н., Юркин Д.В.** Протоколы обеспечения безопасности VoIP-телефонии. – Защита информации. Инсайд, 2012, №3, с.74–81.