ОПИСАНИЕ SIGNAL PROTOCOL

Жемойтяк Наталья Павловна
Бельская Екатерина Артуровна
Чекалинский Никита Олегович

Содержания

Введение

В этом документе будет описываться Signal Protocol и его основные алгоритмы. Здесь будут много отсылок к документации самого сигнала, а также на теорию примитивов, которые использует данный протокол.

Данный протокол является документаций и пояснением к шифрованию сообщений в мессенджере Signal. Сейчас Facebook и Skype утверждают, что они тоже используют данный протокол в своих секретных(или приватных)чатах.

Так как этот протокол используется для сквозного шифрования, в этом документе так же будет описан алгоритм сквозного шифрования

Главные компоненты:

- Двойной храповик(double ratchet algorithm)
- Расширенный тройной Диффи-Хеллман(extended triple Diffie-Hellman)
- Ключи двух пользователей
- Эллиптические кривые
- AES
- HMAC-Sha256
- Ключи с подписью

Базовые понятие и алгоритмы.

Криптосистема RSA

Самая простая и известная криптосистема по обмену сообщений это криптосистема RSA. Она приведена в пример чтобы проще было понять отдельные компоненты данного протокола и как они работают вместе.
RSA состоит из трех частей:

- Генерация ключей
- Шифрование
- Дешифрование

Генерация ключей

Вход: Зависит от задачи. Может быть нулевой
Выход: Ключ = [public key(e,n), private key(d)]
Шаги:

1. Выбираются два простых числа p и q
2. n = p*q
3. φ(n) = (p-1)(q-1)
4. Выбрать 1 <e < φ(n) && GCD(e, φ(n))==1
5. d = e-1(modφ(n))
6. return [e,n,d]

Определение: Публичная часть – эта та, которой пользователь может делится и/или она в открытом доступе. Она используется другим пользователем для шифровки сообщения

Определение: Приватная часть – эта та, которую пользователь не должен распространять, потому что он ее использует для расшифровки сообщения

Шифрование

Вход: e,n,m

Выход: c – зашифрованное сообщение

Шаги:

1. Return[$c = m^e (\mathrm{mod}\ n)$]

Дешифрование

Вход: c,d,n

Выход: m – расшифрованное сообщение

Шаги:

1. Return[$m = c^d (\mathrm{mod}\ n)$]

Зачем нужно понимать RSA

Данный протокол работает так же, как и RSA. Есть ключи, финкции, которые шифруют и расшифруют. Они гораздо сложнее и поэтому в качестве базы мы рассматриваем RSA.

Примитивы

Данный протокол содержит много примитивов, которые проще описать отдельно.

Эллиптические кривые и Диффи-Хеллмана

Диффи-Хеллман

На этом алгоритме основывается весь обмен.

Обмен ключами:

1. Bob генерирует простое число p и число g, такое, что GCD(g,p-1) = =1. Она отправляет их Alice
2. Alice выбирает секретное число a.
3. Alice считает $A = g^a \pmod p$ и посылает Bob
4. Bob считает $B = g^b \pmod p$ и посылает Alice
5. Alice считает $P = B^a \pmod p$ и Bob считает $Q = A^b \pmod p$
6. P==Q

Доказательство:

$P = B^a \pmod p = (g^b \pmod p)^a \pmod p = g^{ba} \pmod p$

$Q = A^b \pmod p = (g^a \pmod p)^b \pmod p = g^{ab} \pmod p$

Эллиптические кривые

Это тоже является одной из основ данного протокола.

Определение: Эллиптической кривой называется множество точек , удовлетворяющих уравнению:

$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$

Они используются как альтернатива огромным простым числам.

Эллиптические кривые Диффи-Хеллмана

Выбирается заранее:

E – эллиптическая кривая

G – точка, принадлежащая кривой E, которая используется как начальная

1. Alice генерирует рандомное a – ее секретный ключ
2. Alice считает A = aG– публичный ключ
3. Bob выбирает b – его секретный ключ
4. Bob считает B = bG – публичный ключ
5. Alice и Bob обмениваются публичными ключами
6. Alice считает K = aB
7. Bob считает K = bA
8. K == K

Доказательство:

K = aB = abG

K = bA = baG

Другие примитивы

SHA-256

Алгоритм хеширования семейства SHA-2

Вход: сообщения

Выход: хеш

Хеш-функции семейства SHA-2 построены на основе структуры Меркла — Дамгора. Исходное сообщение после дополнения разбивается на блоки, каждый блок — на 16 слов. Алгоритм пропускает каждый блок сообщения через цикл с 64 или 80 итерациями (раундами). На каждой итерации 2 слова преобразуются, функцию преобразования задают остальные слова. Результаты обработки каждого блока складываются, сумма является значением хеш-функции. Тем не менее, инициализация внутреннего состояния производится результатом обработки предыдущего блока. Поэтому независимо обрабатывать блоки и складывать результаты нельзя.

HMAC

Один из механизмов проверки целостности информации, позволяющий гарантировать то, что данные, передаваемые или хранящиеся в ненадёжной среде, не были изменены посторонними лицами.

Значения определяющиеся заранее: b - размер блока в байтах

L - размер в байтах строки, возвращаемой хеш-функцией.

L и b зависят от хеш функции

ipad — блок вида (0x36 0x36 0x36 ... 0x36), где

байт 0x36 повторяется b раз; 0x36 — константа, магическое число

Вход: ключ K, H – хеш функция,

Выход: новый ключ

Последовательность

1. Получить измененный ключ K_0 путём уменьшения или увеличения K до размера блока (до размера b байт).
2. Если длина секретного ключа равна размеру блока, то копируем секретный ключ в измененный без изменений и переходим к шагу 5.
3. Если длина секретного ключа больше размера блока, то к ключу применяем хеш-функцию H, получаем строку размером в L байт, добавляем нули к правой части этой строки для создания строки размером в b байт, копируем результат в K_0 и переходим к шагу 5.
4. Если длина ключа K меньше размера блока, то добавляем нули к правой части K для создания строки размером в b байт, копируем результат в K_0 и переходим к шагу 5.
5. Получить блок S_i размером в b байт с помощью операции xor: S_i = xor(K_0, ipad)

6. Получить блок S_o размером в b байт с помощью операции xor S_o = xor(K_0, opad).
7. Разбить сообщение (данные, набор байт) text на блоки размером b байт.
8. Склеить строку (последовательность байт) S_i с каждым блоком сообщения M.
9. К строке, полученной на прошлом шаге, применить хеш-функцию H.
10. Склеить строку S_o со строкой, полученной от хеш-функции H на прошлом шаге.
11. К строке, полученной на прошлом шаге, применить хеш-функцию H.
12. Возвращаем строку из шага 11.

AES

AES - симметричный алгоритм блочного шифрования (размер блока 128 бит, ключ 128/192/256 бит)

Предварительно входные данные разбиваются на блоки по 16 байт, если полный размер не кратен 16 байтам, то данные дополняется до размера, кратного 16 байтам. Блоки представляются в виде матрицы 4x4 — state. Далее происходит процедура расширения ключа и к каждому блоку state применяются операции 2-4.

Последовательность шагов:

1. Расширение ключа - KeyExpansion;
2. Начальный раунд - сложение state с основным ключом;
3. 9 раундов шифрования, каждый из которых состоит из преобразований:
   - замена байтов state по таблице S-box
   - циклический сдвиг строк state
   - умножения каждого столбца state на фиксированную матрицу

- раундовый ключ поэлементно добавляется к state с помощью поразрядного XOR

4. Финальный раунд, состоящий из преобразований:
    - замена байтов state по таблице S-box
    - циклический сдвиг строк state.
    - раундовый ключ поэлементно добавляется к state с помощью поразрядного XOR

Цифровая подпись для ключей

Определение: Цифровая подпись – реквизит, позволяющий подтвердить авторство.

В данном протоколе используются EdDSA алгоритмы, которые совмещены с эллиптическими кривыми ДХ и хеш функциями. Они состоят из двух частей: подпись и верификация(проверка).

Подпись

Вход: секретный ключ, сообщение, последовательность битов

Выход: Подпись(последовательность битов)

Верификация

Вход: публичный ключ, сообщение, подпись

Выход: true/false(прошел верификацию или нет)

Последовательность отправки сообщений

В данном протоколе используют эллиптические кривые и подписи для генерации ключей, которые состоят из двух частей, как и в RSA, который описан выше.

Инициализация

На начальном этапе каждая из сторон имеет свой индивидуальный набор ключей.

Ключи Боба:

1. Идентификационный IKb (public, private)
2. Подписанный клюк SPKb (public, private)
3. Ключ-подпись

Ключи Алисы:

1. Идентификационный IKa (public, private)
2. Эпохальный ключ (public, private)

Боб отправляет публичный идентификационный и подписанный ключи Алисе. Алиса выполняет X3DH, пока Боб находится в автономном режиме, используя его загруженные ключи.

Выполняется 4 обмена Диффи-Хеллмана (X3DH):

DH1=DH(IK_a private, SPK_b public)

DH2=DH(EK_a private, IK_b public)

DH3=DH(EK_a private, SPK_b public)

DH4=DH(EK_a private, OPK_b public)

Боб подключается к сети и выполняет X3DH, используя открытые ключи Алисы.

Выполняется снова 4 обмена Диффи-Хеллмана (X3DH):

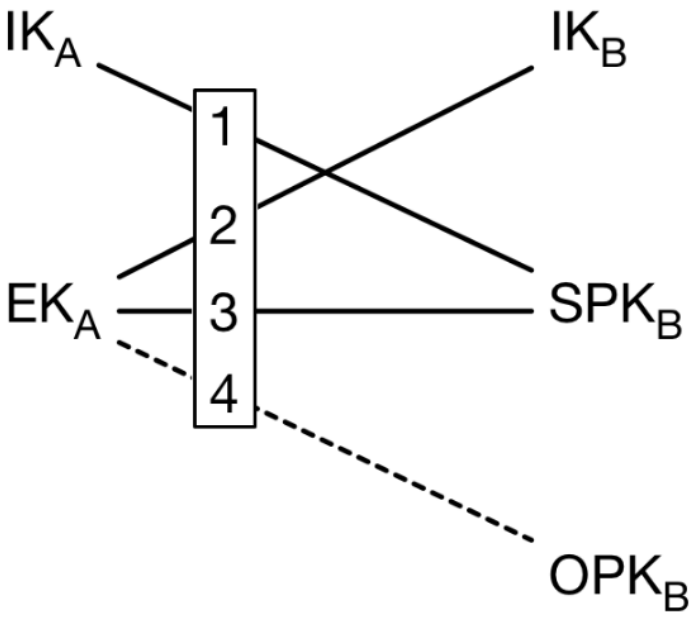
DH1=DH(IK_a public, SPK_b private)

DH2=DH(EK_a public, IK_b private)

DH3=DH(EK_a public, SPK_b private)

DH4=DH(EK_a public, OPK_b private)

Образуется общий ключ - KDF(DH1||DH2||DH3||DH4)



Цепочка KDF

Цепочка KDF является основной концепцией алгоритма двойного храповика.

Пусть будет определена криптографическая функция KDF, которой на вход поступает некоторое сообщение и секретный ключ. Выбирается рандомный KDF ключ. Эти данные преобразуются, и на выходе мы получаем выходной ключ и новый KDF ключ. В качестве примитива используется SHA-256(берется хеш от секретного ключа и сообщения и дополняется нулями до нужной длины).

Пусть определен секретный ключ. Подадим его на вход функции KDF вместе с каким-то input. Образуется новый секретный ключ и output. Полученный секретный ключ снова подаем на вход функции и т.д. Образуется KDF цепочка.

В **сеансе Двойного храповика** между Алисой и Бобом каждая сторона хранит ключ KDF для трех цепочек: **корневой цепочки** , **цепочки отправки**

и **цепочки получения** (цепочка отправки Алисы совпадает с цепочкой получения Боба, и наоборот).

Когда Алиса и Боб обмениваются сообщениями, они также обмениваются новыми открытыми ключами Диффи-Хеллмана, а выходные секреты Диффи-Хеллмана становятся входами в корневую цепочку. Выходные ключи из корневой цепочки становятся новыми ключами KDF для цепочек отправки и получения. Это называется **храповым механизмом Диффи-Хеллмана**.

Цепочки отправки и получения обновляются по мере отправки и получения каждого сообщения. Их выходные ключи используются для шифрования и дешифрования сообщений. Это называется симметричным храповиком.

HKDF — KDF использующая HMAC как генератор псевдослучайных ключей.

Вход: предыдущий ключ, константу

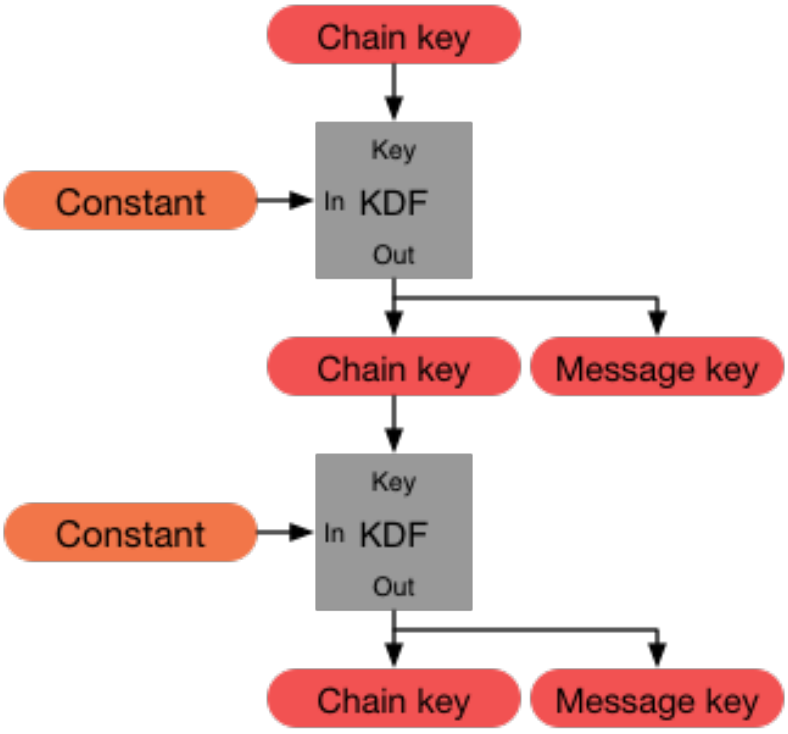
Выход: новый ключ для сообщений, следующий ключ KDF

Шаги:

1. из предоставленного ключа генерируем псевдослучайный ключ с помощью хэш-функции(hmac-256)
2. Второй шаг: с помощью хэш-функции вычисляем K(i), конкатенация которых и является результатом работы функции. Заметим, что на каждом шаге для вычисления K(i) используется предыдущее значение K(i-1)
3. Если длина не кратна заданной длине, то возвращаются первые length октетов

Симметричный храповик

Цепочки отправки и получения обеспечивают шифрование каждого сообщения уникальным ключом, который может быть удален после шифрования или дешифрования. Вычисление следующего ключа цепочки и ключа сообщения из данного ключа цепочки - это один шаг храповика в алгоритме симметричного храповика.

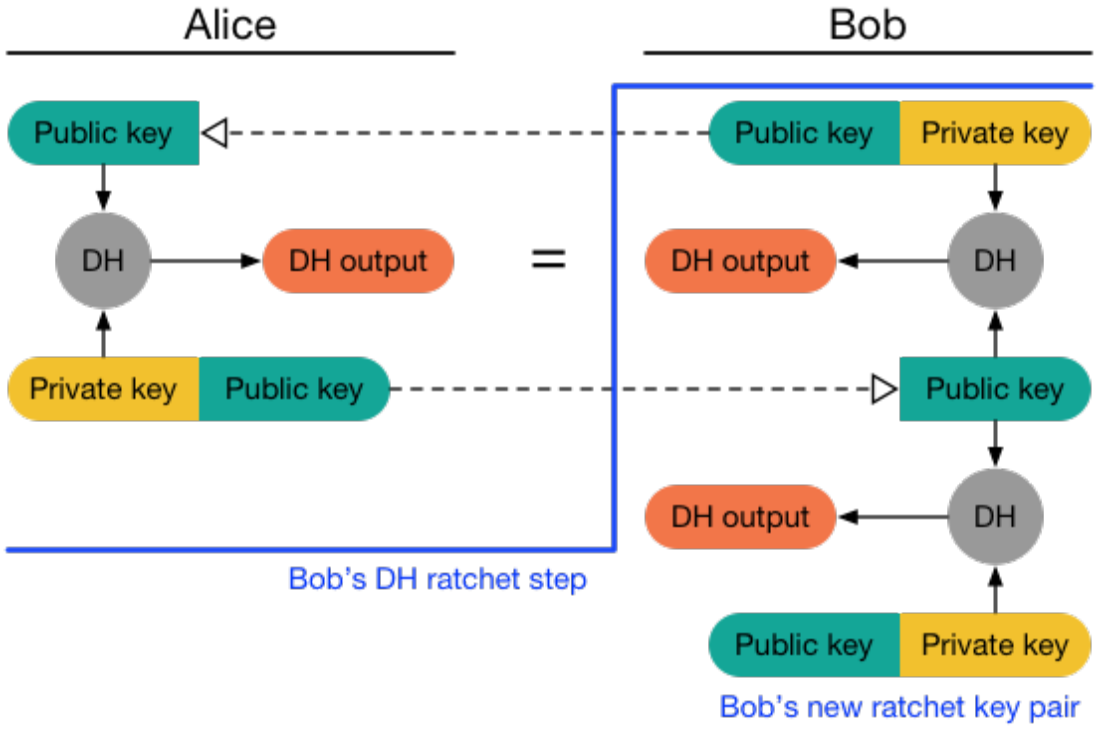


Храповик Диффи-Хеллмана

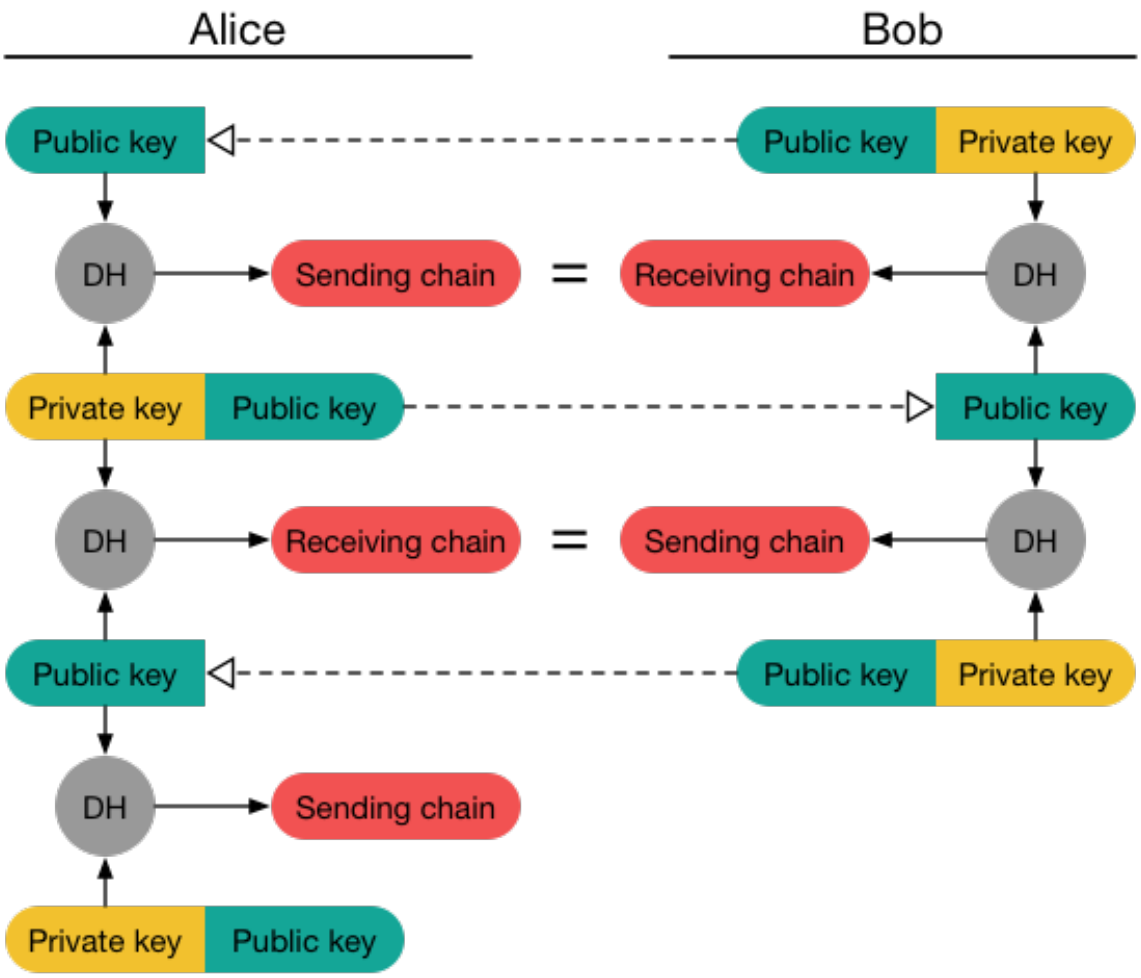
Если злоумышленник крадет ключи цепочки отправки и получения одной из сторон, он может вычислить все будущие ключи сообщений и расшифровать все будущие сообщения. Чтобы предотвратить это, двойной храповик объединяет симметричный храповик с храповиком Диффи-

Хеллмана, который обновляет ключи цепочки на основе результатов Диффи-Хеллмана.

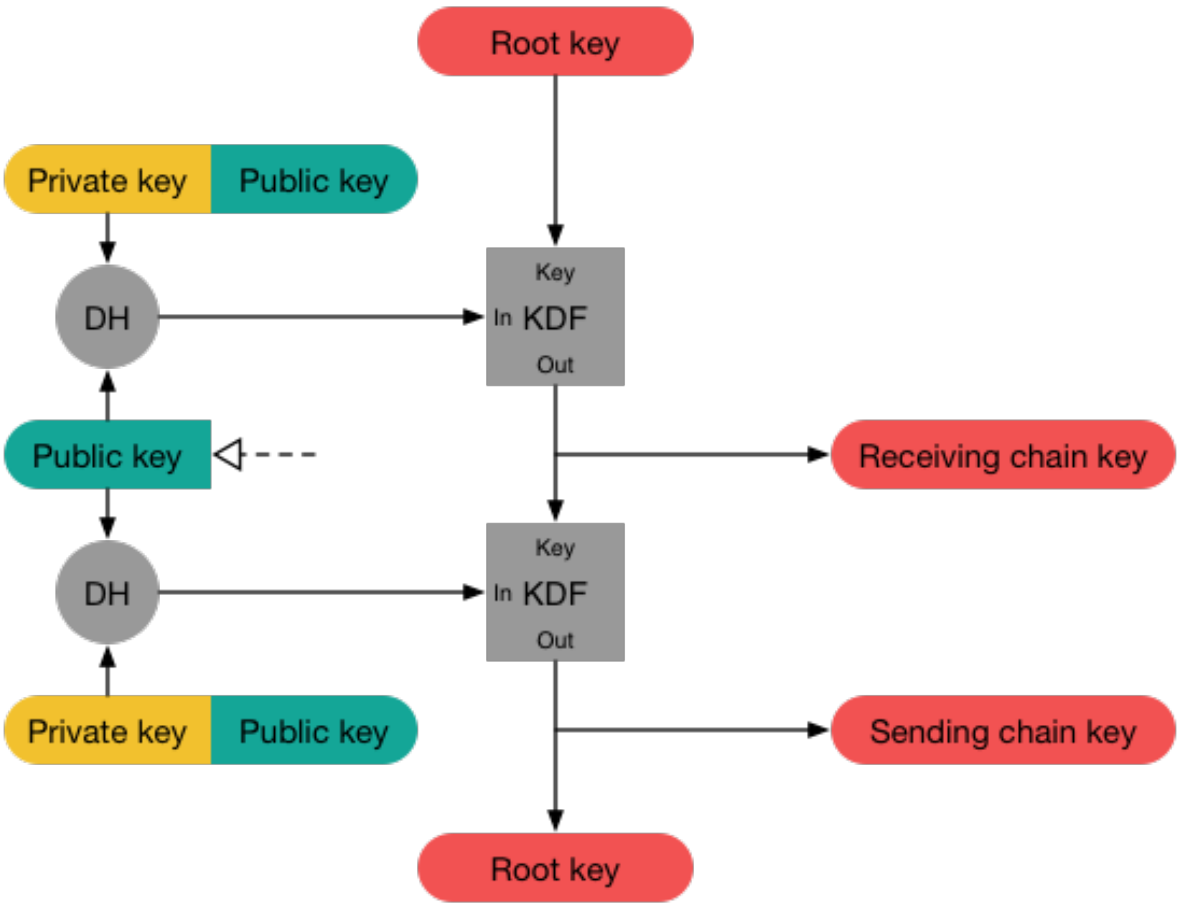
Для реализации храповика DH каждая сторона генерирует пару ключей DH (открытый ключ и закрытый ключ Диффи-Хеллмана), которая становится их текущей парой ключей храповика. Когда новый открытый ключ храповика получен от удаленной стороны, выполняется шаг храповика DH, который заменяет текущую пару ключей храповика локальной стороны новой парой ключей.



Выходы DH, генерируемые на каждом шаге храповика DH, используются для получения новых ключей цепочки отправки и получения.


Alice
Bob
Public key
Public key
Private key
DH
Sending chain
=
Receiving chain
DH
Private key
Public key
Public key
DH
Receiving chain
=
Sending chain
DH
Public key
Public key
Private key
DH
Sending chain
Private key
Public key

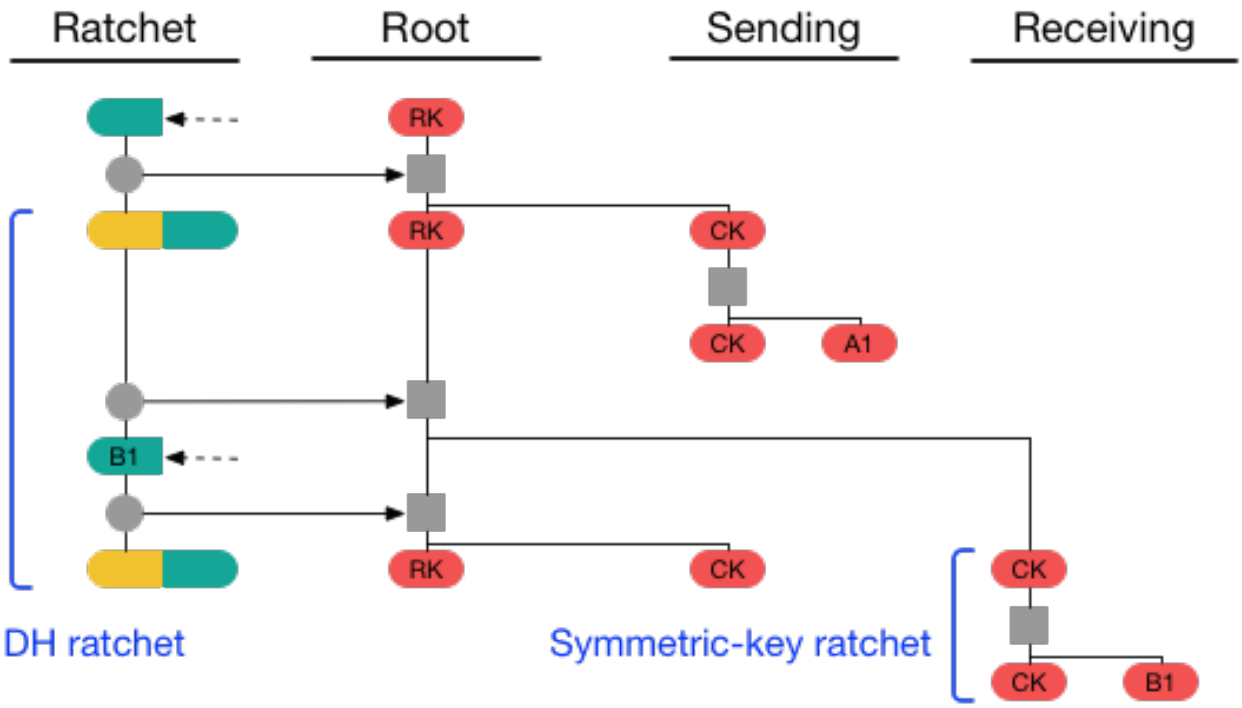
Таким образом, полный шаг храповика выглядит следующим образом:



Двойной храповик

Объединяя симметричный храповик и храповики Диффи Хеллмана получается двойной храповик:

- Когда сообщение получено или отправлено, шаг симметричного храповика применяется к цепочки отправки(получения) для образования ключа сообщения
- Когда новый публичный ключ храповика получен, выполняется шаг храповика Диффи Хеллмана для замены ключей цепочек.


Ratchet
Root
Sending
Receiving
RK
RK
CK
CK
A1
B1
RK
CK
CK
CK
B1
DH ratchet
Symmetric-key ratchet

Список литературы

Чергинец Д.Н. Курс лекций по математическим основам защиты информации - БГУ,2019

https://www.elprocus.com/communication-protocols/

https://en.wikipedia.org/wiki/Signal_Protocol

https://signal.org/docs/

https://ru.wikipedia.org/wiki/Криптографический_протокол

https://qvault.io/cryptography/very-basic-intro-to-elliptic-curve-cryptography/

https://en.wikipedia.org/wiki/RSA_(cryptosystem)#Operation

https://ru.wikipedia.org/wiki/Эллиптическая_криптография#Шифрование/расшифрование_с_использованием_эллиптических_кривых

https://en.wikipedia.org/wiki/Elliptic-curve_Diffie–Hellman

https://medium.com/swlh/understanding-ec-diffie-hellman-9c07be338d4a

https://nfil.dev/coding/encryption/python/double-ratchet-example/

https://en.wikipedia.org/wiki/HKDF

https://habr.com/ru/post/531324/

https://medium.com/swlh/lets-write-a-chat-app-in-python-f6783a9ac170

https://tproger.ru/translations/sha-2-step-by-step/

https://ru.wikipedia.org/wiki/SHA-2

https://ru.wikipedia.org/wiki/HMAC

https://habr.com/ru/post/534620/

https://ru.wikipedia.org/wiki/EdDSA