НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ СІКОРСЬКОГО»

ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Лабораторна робота 4

Дослідження систем обміну повідомленнями та IP-телефонії

Виконали:

Галіца О.О.

Литвиненко Ю.С.

Паршин О.Ю.

ФІ-22мн

Перевірила:

Байденко П.В.

Київ 2023

Viber

Термінологія

Акаунт Viber — сукупність пристроїв, зареєстрованих у Viber на один номер телефону. Акаунт складається з одного основного пристрою і необмеженої кількості додаткових пристроїв. Повідомлення, відправлені або отримані будь-яким пристроєм, відображаються на всіх інших зареєстрованих пристроях цього ж акаунта, починаючи з моменту їх реєстрації і далі (попередня історія не відображається).

Основний пристрій — мобільний телефон під управлінням iOS, Android або Windows Phone, зареєстрований у сервісі Viber за допомогою телефонного номера, що обслуговується мобільним оператором.

Вторинний пристрій — зазвичай iPad, планшет або настільний ПК, який повинен бути пов'язаний з основним пристроєм.

ID-ключ — 256-бітна пара ключів Curve-25519, що використовується для ідентифікації облікового запису Viber. ID-ключ генерується лише основним пристроєм.

PreKeys — набір пар ключів Curve-25519, що використовуються для створення захищених сеансів один на один між пристроями. PreKeys генеруються окремо для кожного пристрою в акаунті.

Сеанс — двостороннє захищене віртуальне з'єднання між двома конкретними пристроями. Безпечні сесії автоматично встановлюються між усіма пристроями одного акаунта, а також між будь-якими двома пристроями різних акаунтів, які обмінюються повідомленнями. Безпечний сеанс з іншим акаунтом у Viber позначається сірим замком на екрані розмови.

Довіра — стан між двома обліковими записами, який вказує на те, що одноранговий обліковий запис автентифікований і що зловмисник не втручався в сеанс, встановлений між будь-якими двома пристроями в обліковому записі. Користувачі можуть встановити довіру, пройшовши процедуру автентифікації, описану нижче. Довірена сесія позначається у Viber зеленим замком.

Порушення довіри — стан між двома акаунтами, який виникає, коли ідентифікаційний ключ довіреного однорангового користувача змінився після встановлення довіри. Це може статися легітимно, якщо одноранговий користувач змінив свій основний пристрій, перевстановив Viber або якщо ключі були втрачені через збій у сховищі. Однак це також може статися, якщо зловмисник підслуховує розмову (man-in-the-middle). Порушена сесія позначається у Viber червоним замком.

Підготовка до налаштування сеансу

Під час встановлення кожен основний пристрій Viber генерує ID-ключ. Приватна частина ключа зберігається на пристрої, тоді як відкрита частина ключа завантажується на сервери Viber. Вторинні пристрої (ПК та планшети) отримують приватний ключ від основного пристрою за допомогою безпечного методу, описаного в розділі "Реєстрація вторинного пристрою"нижче.

Крім того, кожен клієнт Viber генерує серію PreKeys. Кожен PreKey має дві 256-бітові пари ключів Curve-25519, які називаються Handshake Key і Ratchet Key. Усі приватні ключі зберігаються на пристрої, тоді як відкриті ключі завантажуються на сервер.

Налаштування безпечного сеансу

Сесію потрібно створити лише один раз між кожними двома пристроями Viber, які бажають безпечно спілкуватися. Після створення сесії ви можете надсилати необмежену кількість повідомлень в обох напрямках.

Щоб надіслати безпечне повідомлення, між пристроєм-відправником і всіма пристроями одержувача, а також між пристроєм-відправником і всіма іншими пристроями відправника повинні існувати безпечні сеанси зв'язку. Наприклад, якщо користувач А, який має мобільний телефон і комп'ютер, зареєстрований у Viber під одним обліковим записом, хоче поспілкуватися з користувачем Б, який має мобільний телефон і комп'ютер, між кожною парою пристроїв повинні бути встановлені безпечні сеанси.

Сеанси між пристроями одного акаунта встановлюються після реєстрації пристроїв. Між будь-якими двома пристроями потрібна лише одна сесія, і цю сесію можна використовувати для синхронізації будь-якої кількості розмов з іншими акаунтами Viber.

Щоб встановити сеанс зв'язку з іншим акаунтом, пристрій "Аліса", який бажає встановити сеанс зв'язку з одноранговим пристроєм "Боб", надсилає запит на сервер Viber із зазначенням номера телефону одержувача. Сервер відповідає публічним ідентифікаційним ключем одноранго вого партнера та серією публічних PreKeys, по одному на кожен пристрій, зареєстрований в обліковому записі "Боба". Пристрої "Боба" не зобов'язані бути онлайн, коли це відбувається.

Потім пристрій "Аліса" генерує два 256-бітних ключі Curve-25519 (пари власних хендшейк та ретчет ключів), а також виводить кореневий ключ, як показано нижче:

$$RootKey = SHA - 256(DH(IDAlice, HSBob)||DH(HSAlice, IDBob)||DH(HSAlice, HSBob))$$

Потім RootKey використовується для отримання сеансового ключа:

$$TempKey = HMAC\_SHA256(RootKey, DH(RatchetAlice, RatchetBob))$$

$$NewRootKey = HMAC\_SHA256(TempKey, "root")$$

$$SessionKey = HMAC\_SHA256(TempKey, "mesg")$$

DH вказує на використання алгоритму обміну ключами Elliptic-Curve Diffie-Hellman. HS вказує на хендшейк ключ.

Різні рядки, що передаються функціям HMAC, гарантують, що навіть якщо ключ сеансу буде скомпрометований, кореневий ключ не може бути отриманий з нього.

Потім "Аліса" надсилає "Бобу" повідомлення про початок сеансу, яке містить її власний публічний ідентифікатор, ідентифікатор попереднього ключа "Боба", який було використано для цього сеансу, а також її власні публічні хендшейк та ретчет ключі. Коли "Боб" виходить в Інтернет і отримує це повідомлення, він може відновити ті ж самі кореневий і сеансовий ключі, використовуючи ту ж саму процедуру DH.

Обмін повідомленнями

Пристрій, що надсилає повідомлення цільовому користувачеві, повинен шифрувати це повідомлення для кожного сеансу з кожним пристроєм, який має цільовий користувач. Для цього генерується ефемерний одноразовий 128-бітний симетричний ключ, який використовується для шифрування тіла повідомлення за допомогою алгоритму шифрування Salsa20. Цей ефемерний ключ повідомлення потім шифрується за допомогою сеансового ключа кожного одержувача. Пристрій-відправник надсилає на сервер уніфіковане повідомлення, що містить один шифротекст і набір зашифрованих ефемерних ключів. Сервер розрізає це повідомлення на частини і доставляє відповідні частини до кожного цільового пристрою.

Два пристрої по черзі просувають сеансові ключі в процесі, який називається "Ratchet". Щоразу, коли напрямок розмови змінюється, пристрій, чия черга, випадковим чином генерує нову пару ключів Ratchet, і знову виконує наступну послідовність дій:

$$TempKey = HMAC\_SHA256(RootKey, DH(RatchetAlice, RatchetBob))$$

$$NewRootKey = HMAC\_SHA256(TempKey, "root")$$

$$SessionKey = HMAC\_SHA256(TempKey, "mesg")$$

Де Ratchet_device є приватною частиною нової пари ключів. Разом з кожним повідомленням також надсилається публічна частина Ratchet_device. Одержувач запускає DH зі своїм останнім приватним ретчет ключем разом з публічним ретчет ключем відправника.

Подвійне шифрування спрямоване на досягання двох цілей:

1. По-перше, безперервне шифрування забезпечує секретність вперед і назад (forward and backward secrecy), тому навіть якщо ключі скомпрометовані, минулі і майбутні повідомлення не можуть бути розшифровані.

2. По-друге, алгоритм підтримує автентифікацію однорангового пристрою, оскільки ланцюжок DH кореневих ключів починається з ідентифікаційних ключів обох пристроїв. Якщо ідентифікаційному ключу однорангового пристрою довіряють у будь-якій точці, то можна довіряти і всьому ланцюжку.

Зашифровані дзвінки

Кожна сторона виклику генерує ефемерну 256-бітну пару ключів Curve-25519. Відкрита частина підписується за допомогою приватного ідентифікаційного ключа пристрою і обмінюється між двома пристроями на етапі встановлення зв'язку. Інша сторона автентифікує запит за допомогою відкритого ідентифікаційного ключа однорангового партнера. Кожен пристрій перевіряє підпис і виконує обчислення DH для отримання одноразового сеансового ключа.

Ключ сеансу дійсний лише протягом цього конкретного виклику і не зберігається. RTP-потік аудіо- або аудіо/відеодзвінка конвертується в SRTP і шифрується за алгоритмом Salsa20 з використанням сеансового ключа.

Обмін фото, відео та файлами

Клієнт-відправник генерує ефемерний симетричний ключ Salsa20 і шифрує файл. Зашифрований файл разом з HMAC-підписом завантажується на сервери Viber. Додатковий підпис MD5 ставиться на зашифровані дані і надсилається разом з файлом, щоб надати серверу простий спосіб перевірки цілісності передачі незалежно від наскрізного шифрування.

Потім відправник надсилає одержувачу повідомлення Viber з ідентифікатором завантаженого файлу та ключем шифрування. Це повідомлення зашифровано наскрізним шифруванням за допомогою шифрованого сеансу між відправником і одержувачем.

Одержувач створює посилання для завантаження з ідентифікатора файлу, завантажує зашифрований файл і розшифровує його за допомогою ключа.

Безпечні групи

Всі члени захищеної групи мають спільний секретний ключ (симетричний ключ шифрування Salsa20), який не відомий ні Viber, ні третім особам.

Для нових груп цей спільний секрет генерується творцем групи і надсилається всім учасникам за допомогою захищених сеансів тет-а-тет, описаних вище. Для незахищених груп, створених у попередніх версіях Viber, цей секрет генерується першим учасником, який надсилає повідомлення в груповий чат після того, як всі учасники групи перейшли на захищену версію. У додатку Viber будь-який учасник групи може додати до неї додаткових учасників. Ці учасники отримають секрет від учасника групи, який їх додав.

Груповий секрет передається з використанняс HMAC-SHA256 на кожному відправленому повідомленні. Кожне групове повідомлення містить порядковий номер, який вказує на кількість викликів хеш-функції. Різні клієнти завжди підхоплюють те місце, де зупинився ланцюжок після останнього повідомлення, і продовжують ланцюжок хешування з цієї точки, тому ключі не використовуються повторно.

Односторонній алгоритм хешування забезпечує пряму секретність: навіть якщо ключ скомпрометований, попередні розмови не можуть бути розшифровані. Минулі ключі відкидаються користувачем і не зберігаються, за винятком короткого вікна, коли два або більше учасників пишуть в групу одночасно.

Реєстрація вторинного пристрою

Ключовою особливістю екосистеми Viber є концепція вторинних пристроїв. Вторинний пристрій — це ПК, iPad або планшет, який з'єднаний з мобільним телефоном користувача і бачить ту саму історію вхідних та вихідних повідомлень. Наскрізне шифрування Viber на вторинних пристроях працює наступним чином:

- Шифрування виконується окремо для кожного пристрою. Якщо користувач A надсилає повідомлення користувачеві B, який має два пристрої, то користувачеві A потрібне окремі наскрізні сеанси з двома пристроями і дані шифруються двічі, кожен раз використовуючи різний набір ключів.

- Автентифікація виконується лише один раз для всього акаунта. Якщо користувач A довіряє користувачеві B, довіра автоматично поширюється на всі пристрої користувача B.
- Автентифікація здійснюється шляхом спільного використання приватного ідентифікаційного ключа між усіма пристроями одного облікового запису. Ідентифікаційний ключ генерується лише первинним пристроєм і передається вторинним пристроям під час реєстрації безпечним способом, як показано нижче:
  - Вторинний пристрій генерує ефемерну 256-бітну пару ключів Curve-25519.
  - Вторинний пристрій генерує QR-код, що містить "Viber UDID" (загальнодоступний унікальний ідентифікатор пристрою, згенерований Viber) та відкриту частину пари ефемерних ключів.
  - Користувач використовує свій основний пристрій для сканування QR-коду. Основний пристрій також генерує 256-бітну пару ключів Curve-25519 і виконує обчислення гешфункції за допомогою відкритого ключа з QR-коду. Результат хешується за допомогою SHA256 для створення спільного секрету.
  - Первинний сервер шифрує власний приватний ідентифікаційний ключ за допомогою цього секрету і надсилає його разом з відкритою частиною ефемерного ключа через сервери Viber на цільовий пристрій, ідентифікований за його UDID, зчитаним з QR-коду. Зашифрований текст підписується за допомогою HMAC- SHA256.
  - Вторинний пристрій отримує повідомлення, виконує той самий DH і хешування, щоб отримати той самий секрет, і використовує його для розшифрування первинного приватного ідентифікаційного ключа.
    Ідентифікаційний ключ є частиною ланцюжка DH, який створює спільні секрети для сеансів один на один. Тому без правильного ідентифікатора вторинний пристрій не може брати участь у захищених сеансах зв'язку, в яких бере участь основний пристрій.

Автентифікація

У додатку Viber автентифікація відбувається в контексті аудіо (або аудіо/відео) дзвінка. Під час розмови кожен користувач може натиснути на екран блокування, щоб побачити числовий рядок, який обчислюється наступним чином:

- Обидва пристрої виконують розрахунок DH, використовуючи власний приватний ідентифікаційний ключ та публічний ідентифікаційний ключ однорангового пристрою, опублікований під час налаштування виклику.
- Результат DH хешується за допомогою SHA256 і обрізається до 160 біт.
- Ці 160 біт потім перетворюються в рядок з 48 десяткових символів (0-9). Обидві сторони повинні бачити один і той самий рядок цифр і можуть порівняти їх, зачитавши учаснику дзвінка. Якщо обидві сторони чують, як інша сторона зачитує вголос той самий рядок цифр,

який вони бачать на своєму екрані, це дає дуже високий ступінь впевненості в тому, що ідентифікаційні ключі не були підроблені, і що в цій розмові не існує “людини посередині”.

Перевірка ідентифікаційного ключа захищає як безпечні дзвінки, так і безпечні чати. У дзвінках ідентифікаційний ключ використовується для підписання DH-повідомлення обміну ключами. У чатах ідентифікаційний ключ функціонує як корінь ланцюжка DH, що веде до генерації спільного секрету. У свою чергу, так само захищаються і групові сеанси, оскільки групові ключі обмінюються під час сеансів 1-1.

Telegram

Для того, щоб забезпечити стабільність, безпеку та надійність спілкування, Telegram використовує власний протокол MTProto. MTProto поділяється на 3 незалежні компоненти:

- Високорівневий компонент: визначає методи, за допомогою яких запити та відповіді конвертується в двійкові повідомлення.
- Криптографічний компонент: визначає метод, за допомогою якого шифруються повідомлення перед передачею по транспортному протоколу.
- Транспортний компонент: визначає метод передачі повідомлень між клієнтом та сервером (HTTP, HTTPS, WS, WSS, TCP, UDP).

З точки зору високорівневого компонента, клієнт і сервер обмінюються повідомленнями всередині деякої сесії. Сесія прив’язана до клієнтського пристрою (точніше, до програми), а не до конкретного веб-сокета/http/https/tcp-з’єднання. Крім того, кожна сесія прив’язана до ідентифікатора ключа користувача, за допомогою якого здійснюється авторизація.

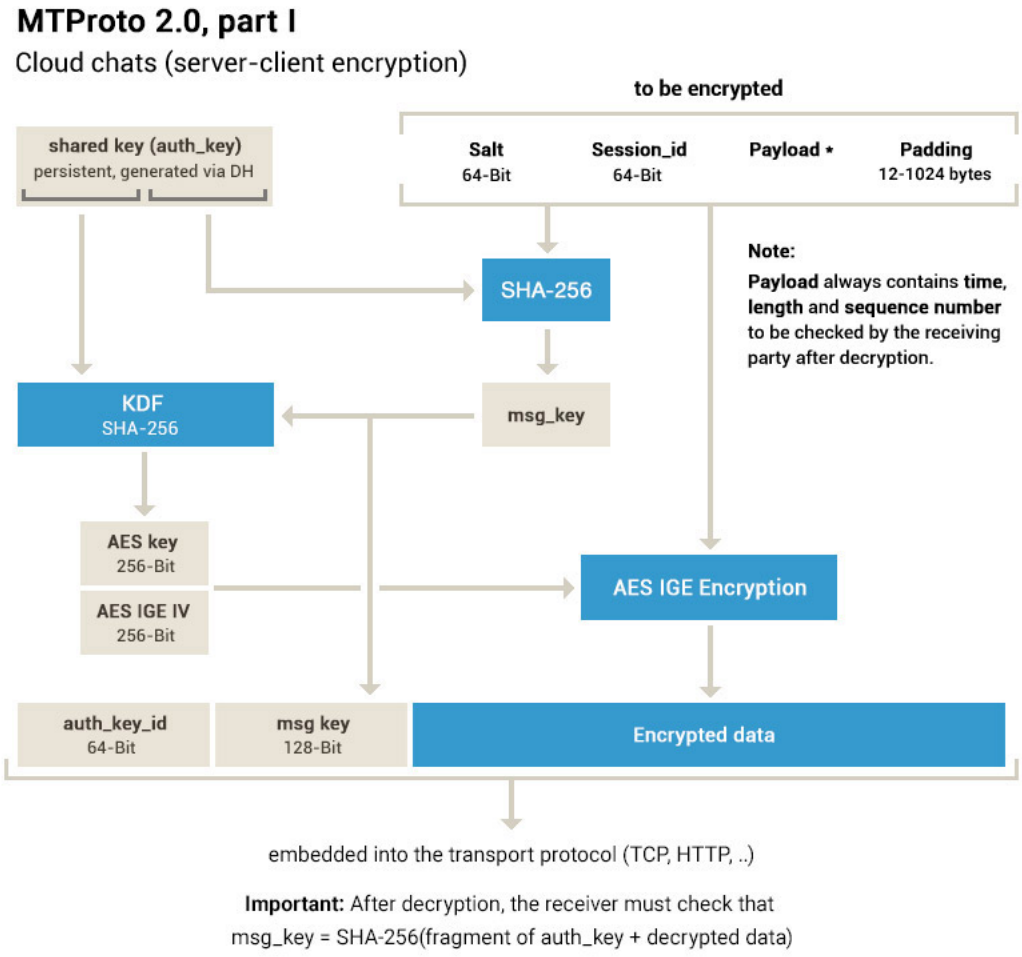
З точки зору протоколів нижчого рівня, повідомлення – це двійковий потік даних, довжиною кратною 4 або 16. Кожне повідомлення складається з:

- Ідентифікатор повідомлення: 64 біти.
- Номер повідомлення в сесії: 32 біти.
- Довжина повідомлення в байтах: 32 біти.
- Тіло: будь-який розмір (кратний 4-ом байтам).

Перед надсиланням, повідомлення шифрується і на початок додається бітний заголовок – це 64 біти ідентифікатора ключа і 128-бітовий ключ повідомлення. Ключ повідомлення разом з 128-бітовим ключем користувача вони формують ключ довжиною 256 бітів, який і буде використаний для шифрування AES-256.

Щоб захиститися від зловмисників, які можуть перехопити зашифровані повідомлення, розшифровувати їх постфактум, отримавши ключ авторизації (наприклад, шляхом крадіжки пристрою), MTProto підтримує Perfect Forward Secrecy (шляхом використання temp auth key, який пов’язаний з оригінальним auth key, та який можна відкликати).

В цілому, MTProto підтримує два режими роботи: client-server encryption (використовується для звичайних чатів) та end-to-end encryption (використовується для так званих секретних чатів).

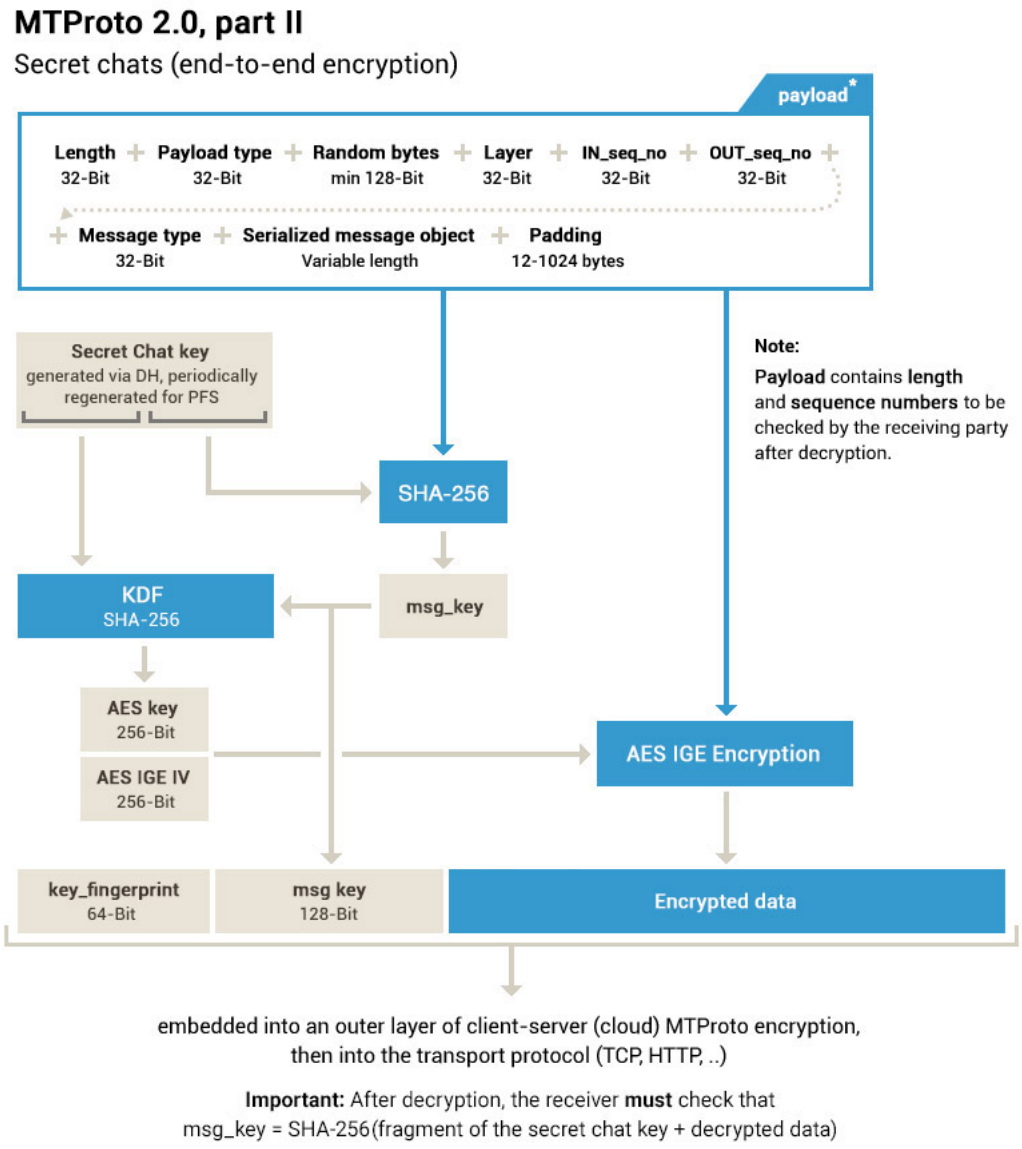


2048-бітовий ключ авторизації, спільний для клієнтського пристрою і сервера, створюється при реєстрації користувача безпосередньо на клієнтському пристрої шляхом використання алгоритму Діффі-Геллмана, і ніколи не передається по мережі. Кожен ключ авторизації є унікальним для користувача. Ніщо не заважає користувачеві мати кілька ключів (які відповідають "постійним сесіям"на різних пристроях), і деякі з них можуть бути заблоковані назавжди в разі втрати пристрою.

Ключ авторизації і 128-бітовий ключ повідомлення використовуються для обчислення 256-бітового ключа AES і 256-бітового вектора ініціалізації, які потім використовуються для шифрування частини повідомлення (майже всього, окрім зовнішнього заголовка, який додається пізніше) за допомогою AES-256 в режимі нескінченного розширення (IGE).

Молодші 1024 біти ключа авторизації не беруть участі в обчисленнях. Вони можуть (разом з рештою бітів або окремо) використовуватися на клієнтському пристрої для шифрування локальної копії даних, отриманих від сервера. 512 молодших бітів ключа авторизації не зберігаються на сервері, тому, якщо клієнтський пристрій використовує їх для шифрування локальних даних, а користувач втрачає ключ або пароль, розшифрування локальних даних неможливе (навіть якщо дані з сервера можна отримати).

Для створення auth-key-id використовуються 64 молодші біти хешу SHA1 ключа авторизації. Ключі повинні бути однозначно визначені 64 молодшими бітами SHA1, і в разі виникнення колізії ключ авторизації буде перегенеровано.



Skype

Skype зберігає реєстраційну інформацію як на комп'ютері абонента, так і на сервері Skype. Skype використовує цю інформацію для автентифікації одержувачів дзвінків і гарантує, що абоненти, яким потрібна автентифікація, отримують доступ до сервера Skype, а не до самозванця. Skype стверджує, що для цього він використовує шифрування з відкритим ключем, визначеним RSA (20480.

Сервер Skype має приватний ключ і поширює його публічний аналог з кожною копією клієнта. Під час реєстрації користувач обирає бажане ім'я користувача та пароль. Skype локально генерує відкритий і закритий ключі. Закритий ключ і хеш пароля зберігаються на комп'ютері користувача.

Потім встановлюється 256-бітний сеанс зв'язку з сервером Skype, зашифрований AES-шифруванням. Клієнт створює ключ сеансу за допомогою свого генератора випадкових чисел.

Сервер Skype перевіряє, чи вибране ім'я користувача є унікальним і відповідає правилам іменування Skype. Сервер зберігає ім'я користувача та геш пароля користувача у своїй базі даних.

Тепер сервер формує і підписує сертифікат ідентичності для імені користувача, який зв'язує ім'я користувача, ключ перевірки та ідентифікатор ключа.

Весь трафік під час сеансу шифрується за допомогою алгоритму AES у режимі ICM. Skype шифрує поточний лічильник і сіль за допомогою ключа сеансу, використовуючи 256-бітний алгоритм AES. Цей алгоритм повертає потік ключів, який потім додається до вмісту повідомлення. Сеанси Skype містять кілька потоків. Лічильник ICM залежить від потоку і місця в потоці.

WhatsApp

WhatsApp Messenger дозволяє людям обмінюватися повідомленнями (включаючи чати, групові чати, зображення, відео, голосові повідомлення та файли), ділитися статусними записами і здійснювати виклики через WhatsApp по всьому світу. Повідомлення, голосові і відеовиклики через WhatsApp між відправником і одержувачем, які використовують програмне забезпечення клієнта WhatsApp, використовують протокол Signal, описаний нижче. Дивіться "Визначення end-to-end шифрування"для інформації про те, які комунікації є end-to-end зашифрованими.

Протокол Signal, розроблений Open Whisper Systems, є основою для end-to-end шифрування в WhatsApp. Цей протокол end-to-end шифрування розроблений так, щоб запобігти третім сторонам і WhatsApp отримувати доступ до текстових повідомлень чи викликів у відкритому вигляді. Завдяки ефемерному характеру криптографічних ключів, навіть у ситуації, коли поточні ключі шифрування з пристрою користувача фізично скомпрометовані, вони не можуть бути використані для розшифрування раніше переданих повідомлень.

Типи пристроїв

- Основний пристрій - пристрій, який використовується для реєстрації облікового запису WhatsApp за номером телефону. Кожен обліковий запис WhatsApp пов'язаний з одним основним пристроєм. Цей основний пристрій може бути використаний для пов'язання додаткових супутніх пристроїв із обліковим записом. Платформи основних пристроїв, які підтримуються, включають Android і iPhone.

- Супутній пристрій - пристрій, який пов'язаний із існуючим обліковим записом WhatsApp за допомогою основного пристрою облікового запису. Платформи основних пристроїв, таких як iPhone та Android, не підтримують можливість пов'язання їх як супутникових пристроїв.

Типи відкритих ключів

- Ключ пари ідентифікації - Довгострокова пара ключів Curve25519, створена під час встановлення.

- Підписаний ключ передплати - Середньострокова пара ключів Curve25519, створена під час встановлення, підписана ключем ідентифікації та регулярно обертана за графіком.
- Одноразові ключі передплати - Черга пар ключів Curve25519 для одноразового використання, створена під час встановлення і поповнювана за потребою.

Типи ключів сеансу

- Кореневий ключ - Значення у 32 байти, яке використовується для створення Ланцюгових ключів.
- Ланцюговий ключ - Значення у 32 байти, яке використовується для створення Ключів повідомлень.
- Ключ повідомлення - Значення у 80 байтів, яке використовується для шифрування вмісту повідомлення. 32 байти використовуються для ключа AES-256, 32 байти для ключа HMAC-SHA256, і 16 байтів для IV.

Інші типи ключів

- Ключ секретного пов'язання - Значення у 32 байти, яке генерується на супутньому пристрої та повинно передаватися захищеним каналом на основний пристрій, використовується для перевірки HMAC навантаження зв'язування, отриманого від основного пристрою. Передача цього ключа від супутніх пристроїв на основний пристрій відбувається за допомогою сканування QR-коду.

Зв'язування

- Метадані зв'язування - Кодована маса метаданих, призначена для супутнього пристрою під час зв'язування, використовується разом із ключем ідентифікації супутнього пристрою для ідентифікації зв'язаного супутника на клієнтах WhatsApp.
- Підписані дані списку пристроїв - Кодований список, що ідентифікує в даний момент зв'язані супутні пристрої на момент підпису. Підписаний ключем ідентифікації основного пристрою за допомогою префіксу 0x0602.
- Підпис облікового запису - Підпис Curve25519, розрахований за префіксом 0x0600, метаданими зв'язування та публічним ключем ідентифікації супутнього пристрою за допомогою ключа ідентифікації основного пристрою.
- Підпис пристрою - Підпис Curve25519, розрахований за префіксом 0x0601, метаданими зв'язування та публічним ключем ідентифікації супутнього пристрою, а також публічним ключем ідентифікації основного пристрою за допомогою ключа ідентифікації супутнього пристрою.

Обмін повідомленнями

Після встановлення сесії клієнти обмінюються повідомленнями, які захищені Ключем повідомлення за допомогою AES256 у режимі CBC для шифрування та HMAC-SHA256 для аутентифікації. Клієнт використовує розгалуження для всіх обмінюваних повідомлень, що означає, що кожне повідомлення шифрується для кожного пристрою за відповідною парною сесією.

Ключ повідомлення змінюється для кожного переданого повідомлення і є ефемерним, так що Ключ повідомлення, який використовується для шифрування повідомлення, не може бути відновлений зі стану сесії після передачі чи отримання повідомлення. Ключ повідомлення походить від Ланцюжкового ключа відправника, який "повертається вперед"з кожним відправленим повідомленням. Крім того, з кожним обміном повідомленнями виконується нова угода ECDH для створення нового Ланцюжкового ключа. Це забезпечує передбачуваність майбутніх ключів за рахунок як "геш-ретчетування"і "DH-ретчетування".

Розрахунок Ключа повідомлення з Ланцюжкового ключа

Кожного разу, коли відправнику потрібен новий Ключ повідомлення, він обчислюється так:

Ключ повідомлення = HMAC-SHA256(Ланцюжковий ключ, 0x01).

Потім Ланцюжковий ключ оновлюється як Ланцюжковий ключ = HMAC-SHA256(Ланцюжковий ключ, 0x02). Це призводить до "ретчетування"Ланцюжкового ключа вперед, а також означає, що збережений Ключ повідомлення не може бути використаний для отримання поточних чи минулих значень Ланцюжкового ключа.

Розрахунок Ланцюжкового ключа від Кореневого ключа

Кожного разу, коли передається повідомлення, із ним йде ефемерний публічний ключ Curve25519. Як тільки надійшла відповідь, розраховується новий Ланцюжковий ключ і Кореневий ключ так:

ephemeralSecret = ECDH(EphemeralSender, EphemeralRecipient).

Ланцюжковий ключ, Кореневий ключ = HKDF(Кореневий ключ, ephemeralSecret).

Ланцюжок використовується тільки для відправки повідомлень від одного користувача, тому ключі повідомлень не повторюються. Через спосіб розрахунку Ключів повідомлень і Ланцюжкових ключів повідомлення можуть надходити з затримкою, у порушеному порядку або можуть бути втрачені повністю, не призводячи до жодних проблем.

Групові повідомлення

End-to-end шифрування повідомлень, відправлених у групи WhatsApp, використовує встановлені парні зашифровані сесії, як описано раніше в розділі "Ініціація встановлення сесії для розподілу компоненту "Ключа відправника"протоколу обміну повідомленнями Signal.

При відправці повідомлення в групу вперше генерується і розповсюджується "Ключ відправника"для кожного пристрою-учасника групи, використовуючи парні зашифровані сесії. Зміст по-

відомлення шифрується за допомогою "Ключа відправника забезпечуючи ефективне та безпечне розгалуження для повідомлень, що відправляються у групи.

Коли учасник групи WhatsApp вперше надсилає повідомлення в групу:

- Відправник генерує випадковий Ланцюжковий ключ у 32 байти.
- Відправник генерує випадкову пару ключів для підпису Curve25519.
- Відправник об'єднує 32-байтовий Ланцюжковий ключ та відкритий ключ від Ключа підпису в повідомлення "Ключ відправника".
- Відправник окремо шифрує "Ключ відправника"кожному учаснику групи, використовуючи раніше пояснений протокол обміну повідомленнями.

Для всіх наступних повідомлень у групі:

- Відправник виводить Ключ повідомлення з Ланцюжкового ключа та оновлює Ланцюжковий ключ.
- Відправник шифрує повідомлення, використовуючи AES256 у режимі CBC.
- Відправник підписує криптотекст використовуючи Ключ підпису.
- Відправник передає одиночне криптотекстове повідомлення на сервер, який розгалужує його на стороні сервера для всіх учасників групи.

"Хеш-ретчет"Ланцюжкового ключа відправника повідомлення забезпечує передбачуваність вперед, і коли учасник групи залишає групу, всі учасники групи очищають свій "Ключ відправника"і починають заново.

У Chat Device інформація про консистентність включається при розповсюдженні "Ключа відправника а потім виключається з наступних повідомлень, зашифрованих за допомогою "Ключа відправника".

Перевірка ключів

Користувачі WhatsApp додатково можуть перевірити ключі своїх пристроїв та пристроїв користувачів, з якими вони спілкуються у чатах з end-to-end шифруванням, щоб підтвердити, що несанкціонована третя сторона (або WhatsApp) не ініціювала атаку типу "чоловік посередині". Перевірку можна здійснити, скануючи QR-код або порівнюючи 60-цифровий номер між двома основними пристроями. Користувачі WhatsApp також можуть вручну перевірити окремі супутні пристрої, використовуючи основний пристрій для перевірки того самого QR-коду або 60-цифрового номера.

QR-код містить:

- Версію.

- Ідентифікатор користувача для обох сторін.
- Повний 32-байтовий відкритий ключ ідентичності для всіх пристроїв обох сторін.
- Відправник передає одиночне криптотекстове повідомлення на сервер, який розгалужує його на стороні сервера для всіх учасників групи.

Коли будь-який з пристроїв сканує QR-код іншого, ключі порівнюються для того, щоб переконатися, що те, що міститься в QR-коді, відповідає Ключу ідентичності, отриманому з сервера.

60-цифровий номер обчислюється шляхом конкатенації двох 30-цифрових числових відбитків для Ключів ідентичності пристроїв кожного користувача. Для обчислення 30-цифрового числового відбитка:

- Лексикографічно сортуються відкриті ключі ідентичності для всіх пристроїв користувача і конкатенуються.
- Ітеративно хешуються відсортовані Ключі ідентичності та ідентифікатор користувача за допомогою SHA-512 5200 разів.
- Беруться перші 30 байтів остаточного виводу хеша.
- Розділіть 30-байтовий результат на шість 5-байтових частин.
- Кожен 5-байтовий шматок конвертується в 5 цифр, інтерпретуючи кожен 5-байтовий шматок як беззнакове ціле великого порядку та зменшуючи його за модулем 100000.
- Конкатенуються шість груп по п'ять цифр в тридцять цифр.