Дехтярёв Арсений

Расширение TLS 1.3: extended_key_update

Проблема

Протокол Transport Layer Security (TLS) позволяет обеспечить конфиденциальность, целостность и аутентичность передаваемых данных. Несмотря на важные усовершенствования, в оригинальной спецификации TLS 1.3 стандартный механизм обновления (Key Update) не предполагает использования дополнительной энтропии (Fresh Entropy) через новый обмен ключами, что ограничивает его применимость в сценариях с длительными сессиями. В ответ на такие ограничения был разработан проект расширения под названием extended_key_update, целью которого является предоставление механизма защиты от чтения "вперед" (восстановление после компорометации, Post-Compromise Security) путем использования протокола Диффи--Хеллмана при обновлении ключей.

Стандартный механизм обновления ключей в TLS 1.3 позволяет сторонам обновлять симметричные ключи, используемые для защиты данных после завершения Handshake-фазы. Однако этот механизм основан на производной предыдущих ключей и не включает в себя дополнительной энтропии, что означает, что компрометация текущих ключей может поставить под угрозу все последующие ключи и защиту сессии. Это особенно актуально для технологий с долгоживущими сессиями, таких как интернет вещей (IoT).

Концепт решения

Расширение extended_key_update предлагает механизм обновления ключей, включающий новый обмен ключами на основе протокола Диффи-Хеллмана, что позволяет добавить энтропию и восстановить безопасность даже после временной компрометации текущих ключей трафика. Идея заключается в том, что стороны сессии периодически инициируют обмен новыми ключами в активной сессии, аналогично первоначальному Handshake-обмену, но без необходимости прерывать существующие защищённые каналы или повторно выполнять весь Handshake.

Основная цель такого механизма - ограничить временное окно действия скомпрометированных ключей и заставить потенциального злоумышленника проводить новые активные атаки для каждого цикла обновления ключей, что значительно повышает стоимость и сложность атаки в целом.

Детали решения

Оба участника сессии договариваются о поддержке расширения ещё на этапе ClientHello и EncryptedExtensions, используя механизм TLS-флагов. Если сервер

не поддерживает расширение extended_key_update, тогда он игнорирует флаг и использует стандартный механизм обновления ключей в TLS 1.3.

Расширение определяет новый тип Handshake-сообщения ExtendedKeyUpdate (EKU), который включает три подтипа:

- key_update_request — инициирует обмен новым ключевым материалом и содержит структуру KeyShareEntry, как и в обычном Handshake для обмена DH-ключами;
- key_update_response — ответ на запрос, также содержит KeyShareEntry;
- new_key_update — служебное сообщение для переключения на новые ключи после успешного обмена.

Каждое сообщение подписывается или шифруется в контексте текущих ключей, что обеспечивает целостность обмена.

Процесс работы расширения extended_key_update можно разделить на несколько шагов:

1. Сторона A: Инициатор (клиент или сервер) генерирует одноразовый открытый ключ и отправляет сообщение EKU(key_update_request) с данными ключа в структуре KeyShareEntry.
2. Сторона B: Получив сообщение EKU(key_update_request), отвечает сообщением EKU(key_update_response) со своим одноразовым открытым ключом в KeyShareEntry. После отправления сообщения сторона вычисляет общий секрет (Shared Secret) на основе двух открытых ключей, после чего вычисляет новые подчинённые ключи (Traffic Secrets).
3. Сторона A: Получает EKU(key_update_response), вычисляет общий секрет, после чего вычисляет новые подчинённые ключи и отправляет пустое сообщение EKU(new_key_update), сигнализируя о готовности переключиться на новые ключи. После отправки сообщения сторона обновляет собственные ключи записи.
4. Сторона B: После получения EKU(new_key_update) обновляет свои ключи чтения, затем отправляет инициатору сообщение EKU(new_key_update). После отправки сообщения обновляет свои ключи записи.
5. Сторона A: Окончательно обновляет свои ключи чтения и на этом завершает алгоритм обновление ключей.

Алгоритм построения ключей расширения включает в себя:

1. Генерация нового главного секрета (Main Secret, Master Secret). Новый главный секрет формируется на основе двух компонентов: производное значение от предыдущего главного секрета (используется как “соль”) и

общий секрет полученный в результате протокола Диффи-Хеллмана. Эта операция выполняется с помощью функции HKDF-Extract.

2. Привязка к контексту. На этапе вычисления подчинённых ключей происходит привязка к истории обмена. В функцию Derive-Secret в качестве контекста передаётся конкатенация сообщений обмена EKU(key_update_request) и EKU(key_update_response). Это гарантирует целостность сессии, так как итоговые ключи неразрывно связаны с конкретными сообщениями, которыми обменялись стороны. На основе этих ключей строят ключи защиты прикладных данных, как описано в стандарте протокола TLS 1.3.

Дополнительно

На момент написания реферата это расширение находится в статусе “Active Internet-Draft”, ему не присвоен номер RFC и в будущем описание расширения может измениться.

Источники

1. RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3
2. Extended Key Update for Transport Layer Security (TLS) 1.3 https://datatracker.ietf.org/doc/draft-ietf-tls-extended-key-update/