Internet Engineering Task Force (IETF)
Request for Comments: 7366
Category: Standards Track
ISSN: 2070-1721

P. Gutmann
University of Auckland

September 2014

Encrypt-then-MAC for Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)

Шифрование-затем-MAC для TLS и DTLS

Аннотация

Этот документ описывает способ согласования применения механизма защиты encrypt-then-MAC[1] взамен имеющегося MAC-then-encrypt[2] для защиты транспортного уровня TLS[3] и DTLS[4]. За многие годы было обнаружено множество уязвимостей механизма MAC-then-encrypt.

Статус документа

Документ относится к категории Internet Standards Track.

Документ является результатом работы IETF[5] и представляет согласованный взгляд сообщества IETF. Документ прошёл открытое обсуждение и был одобрен для публикации IESG[6]. Не все одобренные IESG документы претендуют на статус Internet Standard (раздел 2 в RFC 5741).

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке http://www.rfc-editor.org/info/rfc7366.

Оглавление

1. Введение

TLS [2] и DTLS [4] используют конструкцию MAC-then-encrypt, которая представлялась безопасной на момент создания исходного протокола SSL[7] в середине 1990-х годов, но сейчас не считается защищённой [5] [6]. Эта конструкция при использовании в TLS, а затем в DTLS показала множество уязвимостей и подверглась различным атакам в течение многих лет. Этот документ задаёт способ перехода к более защищённой конструкции encrypt-then-MAC в процессе согласования TLS/DTLS с целью замены конструкции MAC-then-encrypt. В этом документе MAC[8] означает код аутентификации сообщения.

1.1. Уровни требований

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не нужно** (SHALL NOT), **следует** (SHOULD), **не следует** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **не рекомендуется** (NOT RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе должны интерпретироваться в соответствии с [1].

[1]Шифрование, затем MAC.
[2]MAC, затем шифрование.
[3]Transport Layer Security.
[4]Datagram Transport Layer Security - защита дейтаграмм на транспортном уровне.
[5]Internet Engineering Task Force - комиссия по решению инженерных задач Internet.
[6]Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.
[7]Secure Socket Layer - уровень защищённого сокета.
[8]Message Authentication Code.

2. Согласование Encrypt-then-MAC

Применение encrypt-then-MAC согласуется с помощью расширения TLS/DTLS, определённого в TLS [2]. При подключении клиент включает расширение encrypt_then_mac в своё сообщение client_hello, если он задаёт применять encrypt-then-MAC взамен принятого по умолчанию MAC-then-encrypt. Если сервер способен выполнить это требование, он отвечает расширением encrypt_then_mac в сообщении server_hello. В качестве значения extension_type для этого расширения **нужно** указывать 22 (0x16), а поле extension_data **нужно** оставлять пустым. Клиенту и серверу **недопустимо** использовать encrypt-then-MAC, пока стороны не обменялись расширениями encrypt_then_mac.

2.1. Обоснование

Использование расширений TLS/DTLS для согласования общего переключателя предпочтительней определения новых шифров, поскольку оно может вести к «декартову взрыву», потенциально требующему дублирования каждого имеющегося шифра новым, который будет использовать encrypt-then-MAC. Представленная здесь модель, напротив, требует лишь одного нового типа с соответствующим расширением минимального размера, передаваемым клиентом и сервером.

Другим вариантом внедрения encrypt-then-MAC было бы включение этого механизма в протокол TLS 1.3, однако это потребует полной реализации и развёртывания TLS 1.2 для того, чтобы поддержать тривиальное изменение кода, меняющее порядок операций шифрования и MAC. Реализация encrypt-then-MAC в форме расширения TLS/DTLS требует лишь заменить меньше десятка строк кода в одной реализации (не считая обработки нового типа расширения, для которой требуется около 50 строк кода).

Использование расширений исключает возможность применять SSL 3.0 и все, что работает на основе этого протокола, которому уже около двух десятков лет, остаётся уязвимым для множества других атак, поэтому нет большого смысла прилагать усилия по обеспечению совместимости с SSL 3.0.

3. Применение Encrypt-then-MAC

Когда использование encrypt-then-MAC согласовано сторонами, обработка TLS/DTLS меняется с прежней

```
encrypt( data || MAC || pad )
```

на новую

```
encrypt( data || pad ) || MAC
```

со значением MAC, охватывающим пакет целиком от начала до самого поля MAC. В нотации TLS [2] расчёт MAC для TLS 1.0 без явного вектора инициализации (IV[1]) имеет вид

```
MAC(MAC_write_key, seq_num +
    TLSCipherText.type +
    TLSCipherText.version +
    TLSCipherText.length +
    ENC(content + padding + padding_length));
```

а для TLS 1.1 и выше с явным IV

```
MAC(MAC_write_key, seq_num +
    TLSCipherText.type +
    TLSCipherText.version +
    TLSCipherText.length +
    IV +
    ENC(content + padding + padding_length));
```

Для DTLS порядковый номер заменяется комбинацией «эпохи» (epoch) и порядкового номера в соответствии с DTLS [4]. Полученное значение MAC добавляется после шифрованных полей данных и заполнения. Этот расчёт идентичен имеющемуся за исключением того, что код MAC вычисляется для шифрованных (TLSCipherText PDU), а не открытых (TLSCompressed PDU) полей.

Пакет TLS [2] представляется структурой

```
struct {
       ContentType type;
       ProtocolVersion version;
       uint16 length;
       GenericBlockCipher fragment;
       opaque MAC;
       } TLSCiphertext;
```

А пакет DTLS [4] структурой

```
struct {
       ContentType type;
       ProtocolVersion version;
       uint16 epoch;
       uint48 sequence_number;
       uint16 length;
       GenericBlockCipher fragment;
       opaque MAC;
       } TLSCiphertext;
```

Это идентично существующей схеме TLS/DTLS с различием лишь в том, что значение MAC не шифруется.

Отметим, что аннотация GenericBlockCipher применяется только к стандартным блочным шифрам, которые имеют разные операции шифрования и MAC. Она не применяется к GenericStreamCiphers и GenericAEADCiphers, которые уже включают защиту целостности. Если сервер получает запрос для расширения encrypt-then-MAC от клиента, а затем выбирает потоковый шифр или AEAD[2], ему **недопустимо** передавать клиенту отклик с расширением encrypt-then-MAC.

[1]Initialization Vector.
[2]Authenticated Encryption with Associated Data - аутентифицированное шифрование со связанными данными.

Дешифрование является обратным процессом. Значение MAC **нужно** рассчитать до дальнейшей обработки (дешифровки) и при несовпадении расчётного значения с полученным обработку **нужно** незамедлительно прервать. Для TLS в таком случае **должна** генерироваться критическая ошибка bad_record_mac [2]. Для DTLS запись **должна** отбрасываться и **может** генерироваться критическая ошибка bad_record_mac [4]. Незамедлительный отклик на ошибку MAC предотвращает возникновение любых скрытых каналов, которые могут быть доступны за счёт манипуляций с данными пакета.

Некоторые реализации могут выбрать усечённые коды MAC вместо полноразмерных. В таких случаях они **могут** согласовать использование усечённого MAC с помощью расширения TLS truncated_hmac, определённого в TLS-Ext [3].

3.1. Вопросы пересогласования

Выбор между encrypt-then-MAC и MAC-then-encrypt потенциально может меняться в результате одного или нескольких повторных согласований. Реализациям **следует** сохранять сделанный выбор при всех пересогласованиях в данной сессии (иными словами, если сессия выбрала механизм X, то при повторном согласовании следует снова выбирать X). Хотя реализациям **не следует** менять выбор при повторном согласовании, они могут при желании обеспечить большую гибкость, следуя правилам, приведённым в таблице.

Таблица 1. Encrypt-then-MAC с пересогласованием.

Текущая сессия	***Пересогласованная сессия***	***Действия***
MAC-then-encrypt	MAC-then-encrypt	Без изменений
MAC-then-encrypt	Encrypt-then-MAC	Переход на Encrypt-then-MAC
Encrypt-then-MAC	MAC-then-encrypt	Ошибка
Encrypt-then-MAC	Encrypt-then-MAC	Без изменений

Как указано в таблице, реализациям **недопустимо** согласовывать переход с механизма encrypt-then-MAC на MAC-then-encrypt. Отметим, что клиенты и серверы, не желающие реализовать «возможность смены механизма при повторном согласовании» (mechanism-change-during-rehandshake), могут (в качестве клиента) не запрашивать смену механизма или (в качестве сервера) отвергать её.

Отметим, что приведённые правила применимы ко множеству повторных согласований. Например, если сессия использовала encrypt-then-MAC, при повторном согласовании был выбран шифр GenericAEADCiphers, а при другом повторном согласовании выбран шрифт MAC-then-encrypt это будет ошибкой, поскольку процесс пересогласования привёл к переходу с механизма encrypt-then-MAC на MAC-then-encrypt (через шифр AEAD).

Как уже отмечено выше, реализациям **следует** избегать «упражнений» со сменой выбранного механизма при повторных согласованиях.

Если согласован переход с механизма MAC-then-encrypt на encrypt-then-MAC как во второй строке таблицы, изменение произойдёт в первом сообщении, следующем за сообщением CCS[1]. Иными словами, все сообщения вплоть до CCS (включительно) будут использовать MAC-then-encrypt, а последующие сообщения - encrypt-then-MAC.

4. Вопросы безопасности

Этот документ определяет механизм усиленной защиты encrypt-then-MAC для замены имеющегося механизма MAC-then-encrypt. Шифрование перед вычислением MAC считается более защищённым по сравнению с текущим механизмом [5] [6] и должно ослабить или предотвратить множество атак на используемый сейчас механизм при условии выполнения инструкций по обработке MAC, приведённых в разделе 3.

В активной атаке злоумышленник, способный эмулировать клиента или сервер, не принимающего расширения, может вынудить некоторые реализации к использованию старых версий протоколов, которые не поддерживают расширения, и в конечном итоге к применению механизма без шифрования - non-encrypt-then-MAC. Простым решением этой проблемы будет отказ от возврата к старым, слабо защищённым версиям протоколов. Если этого не удаётся избежать, следует пользоваться разрабатываемыми группой TLS механизмами [7], воздействующими на все возможности, которые согласуются через расширения TLS. Всем, кто озабочен атаками этого типа, рекомендуется ознакомиться с документами рабочей группы TLS, где даны рекомендации по выбору механизмов защиты.

5. Взаимодействие с IANA

Агентство IANA добавило код расширения 22 (0x16) для encrypt_then_mac в реестр TLS «ExtensionType Values», как указано в TLS [2].

6. Благодарности

Автор благодарен Martin Rex, Dan Shumow и участникам почтовой конференции TLS за их отклики.

7. Литература

7.1. Нормативные документы

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[3] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

[4] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

7.2. Дополнительная литература

[5] Bellare, M. and C. Namprempre, "Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm", Proceedings of AsiaCrypt '00, Springer-Verlag LNCS No. 1976, p. 531, December 2000.

[1]Change Cipher Spec - смена шифра.

[6] Krawczyk, H., "The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?)", Proceedings of Crypto '01, Springer-Verlag LNCS No. 2139, p. 310, August 2001.

[7] Moeller, B. and A. Langley, "TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks", Work in Progress[1], July 2014.

Адрес автора

Peter Gutmann

University of Auckland

Department of Computer Science

New Zealand

EMail: pgut001@cs.auckland.ac.nz

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru

[1]Работа опубликована в RFC 7507. *Прим. перев.*