Криптографические протоколы

AE-, AEAD-режимы шифрования. Протокол SSH

Деркач Максим Юрьевич

December 4, 2019

Ссылки

https://www.cs.jhu.edu/~Eastubble/dss/ae.pdf
http://cseweb.ucsd.edu/~mihir/papers/oem.pdf

Введение

AEAD-режим блочного шифрования (Authenticated Encryption with Associated Data) — класс блочных режимов шифрования, при котором часть сообщения шифруется, часть остается открытой, и всё сообщение целиком аутентифицировано.

AEAD пришел на замену режиму AE(Authenticated Encryption), который используется/использовался в таких протоколах как (IPSEC, TLS, SSH...).

AE-режим

Шифрование с проверкой подлинности обеспечивает конфиденциальность и целостность данных для защищаемой информации.
Концепция аутентификации данных появилась в 1970-х годах в банковской сфере.
Сущетвует три вида реализации AE-режима:

1. Authentication and Encryption (MacAndEnc)
2. Authentication Then Encryption (MacThenEnc)
3. Encryption Then Authentication (EncThenMac)

Метод	Пример	Реализация	Результат
MacAndEnc	SSH	$h = MAC(m), C = Enc(m)$	$C\|\|h$
MacThenEnc	SSL	$h = MAC(C), C = Enc(m\|\|h)$	C
EncThenMac	IPSEC	$C = Enc(m), h = MAC(C)$	$C\|\|h$

Определения

Неразличимость шифротекста(Ciphertext indistinguishability) - это свойство многих систем шифрования. Если система обладает свойством неразличимости, то злоумышленник не сможет отличить пары шифротекстов, основываясь на открытых текстах, которые они шифруют.

IND-CPA - Неразличимость для атак на основе подобранного открытого текста
IND-CCA - Неразличимость для атак на основе подобранного шифротекста

Определения

IND-CPA

1. Испытатель генерирует ключ K и передает его злоумышленнику.
2. Злоумышленник может выполнить полиномиально ограниченное число шифрований.
3. Злоумышленник представляет два отдельных открытых текста M_0, M_1 испытателю.
4. Испытатель выбирает $b \in \{0, 1\}$ случайным образом и посылает шифротекст $C = E_K(M_b)$ обратно злоумышленнику.
5. Злоумышленник может выполнять любое количество дополнительных вычислений или шифрований, и в конце выводит b.

Криптосистема надёжна в смысле IND-CPA, если любой вероятный злоумышленник за полиномиальное время имеет лишь незначительное "преимущество" в различении шифротекстов над случайным угадыванием.

Определения

IND-CCA

1. -||-
2. Злоумышленник может выполнить полиномиально ограниченное число шифрований и вызовов дешифрования с оракулом на основе произвольных шифротекстов.
3. -||-
4. -||-
5. Злоумышленник может выполнять любое количество дополнительных вычислений или шифрований и:
   - 5.1 (IND-CCA1) злоумышленник не может выполнять дальнейшие расшифрования с оракулом.
   - 5.2 (IND-CCA2) злоумышленник может выполнять дальнейшие вызовы оракула, но не может использовать для этого шифротекст C.
6. -||-

$IND - CCA2 \Rightarrow IND - CCA1 \Rightarrow IND - CPA$

Определения

Неизменяемость шифротекста(Non-Malleability) - это свойство шифрования. Алгоритм шифрования является изменяемым («податливым»), если возможно преобразовать зашифрованный текст в другой зашифрованный текст, который расшифровывается в заданный открытый текст.

Определения

Целостность открытого текста(INT-PTXT) - это свойство означает, что невозможно создать такой шифротекст, что полученый при его расшифровке открытй текст отправитель никогда не отправлял (шифровал).

Целостность открытого текста(INT-CTXT) - это свойство означает, что невозможно создать шифротекст, ранее не созданый отпарвителем, независимо от того, являеться ли базовый секрет(откртый текст) новым.
$INT - CTXT \Rightarrow INT - PTXT$

AE-режим

Сравнение AE-режимов:

Composition Method	**Privacy**			**Integrity**	
	IND-CPA	IND-CCA	NM-CPA	INT-PTXT	INT-CTXT
Encrypt-and-MAC	insecure	insecure	insecure	secure	insecure
MAC-then-encrypt	secure	insecure	insecure	secure	insecure
Encrypt-then-MAC	secure	secure	secure	secure	secure

Введение

Протокол SSH - сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов). Данный проткол состоит из 3 подпротоколов:

- Транспортный протокол (The Transport Protocol)[SSH-TRANS] - отвечает за аутентификацию, конфендициальность, целостность и опционально сжатие данных. Этот протокол зачастую работает поверх протокола TCP/IP.
  Документация: **RFC 4253 - The Secure Shell (SSH) Transport Layer Protocol**

Введение

- Протокол пользовательской аутентификации (The User Authentication Protocol)[SSH-USERAUTH] - отвечает за аутентификацию между клиентом и сервером. Этот протокол работает поверх протокола SSH-TRANS. Документация: **RFC 4252 - The Secure Shell (SSH) Authentication Protocol**
- Протокол соединения (The Connection Protocol) [SSH-CONNECT] - протокол взаимодействия. Этот протокол работает поверх протокола SSH-USERAUTH. Документация: **RFC 4254 - The Secure Shell (SSH) Connection Protocol**

SSH-TRANS

Транспортный проткол (уровень) SSH является транспортным протоколом низкого уровня.
Он обеспечивает надежное шифрование, криптографическую аутентификацию хоста(этот протокол не выполняет аутентификацию пользователя) и защиту целостности данных.

SSH-TRANS

1. Установка TCP-соединения.
2. Обмен идентификационными данными (Protocol Version Exchange): стороны обмениваются версиями SSH-протокола и другими вспомогательными данными, необходимыми для выяснения совместимости протоколов и для выбора алгоритмов работы.
3. Выбор алгоритмов (Key Exchange): обмена ключами, шифрования и сжатия: каждая сторона, отправляет список имен поддерживаемых алгоритмов.
4. Получение сессионных ключей (Output from Key Exchange): процесс получения сессионного ключа отличается в зависимости от версии алгоритма. На этом шаге выполняеться проткол обмена ключами, в результате которого обе стороны вычисляют два значения: K - общий секрет, H - хэш.

SSH-TRANS

7.1. Algorithm Negotiation

Key exchange begins by each side sending the following packet:

```
byte         SSH_MSG_KEXINIT
byte[16]     cookie (random bytes)
name-list    kex_algorithms
name-list    server_host_key_algorithms
name-list    encryption_algorithms_client_to_server
name-list    encryption_algorithms_server_to_client
name-list    mac_algorithms_client_to_server
name-list    mac_algorithms_server_to_client
name-list    compression_algorithms_client_to_server
name-list    compression_algorithms_server_to_client
name-list    languages_client_to_server
name-list    languages_server_to_client
boolean      first_kex_packet_follows
uint32       0 (reserved for future extension)
```

SSH-TRANS

Encryption keys MUST be computed as HASH, of a known value and K, as follows:

- Initial IV client to server: HASH(K || H || "A" || session_id)
  (Here K is encoded as mpint and "A" as byte and session_id as raw data. "A" means the single character A, ASCII 65).

- Initial IV server to client: HASH(K || H || "B" || session_id)

- Encryption key client to server: HASH(K || H || "C" || session_id

- Encryption key server to client: HASH(K || H || "D" || session_id

- Integrity key client to server: HASH(K || H || "E" || session_id)

- Integrity key server to client: HASH(K || H || "F" || session_id)