Лекция 10. Протоколы распределения ключей

Косолапов Ю.В.

ЮФУ

11 ноября 2020 г.

Содержание

Определение

Протокол распределения ключей [key distribution protocol] – протокол получения пользователями ключей, необходимых для функционирования криптографической системы.

Определение

Протокол распределения ключей [key distribution protocol] – протокол получения пользователями ключей, необходимых для функционирования криптографической системы.

Выделяют следующие типы протоколов

Определение

Протокол распределения ключей [key distribution protocol] – протокол получения пользователями ключей, необходимых для функционирования криптографической системы.

Выделяют следующие типы протоколов

- протоколы **передачи** ключей (уже сгенерированных)

Определение

Протокол распределения ключей [key distribution protocol] – протокол получения пользователями ключей, необходимых для функционирования криптографической системы.

Выделяют следующие типы протоколов

- протоколы **передачи** ключей (уже сгенерированных)
- протоколы **совместной генерации** ключей (генерируются новые ключи)

Определение

Протокол распределения ключей [key distribution protocol] – протокол получения пользователями ключей, необходимых для функционирования криптографической системы.

Выделяют следующие типы протоколов

- протоколы **передачи** ключей (уже сгенерированных)
- протоколы **совместной генерации** ключей (генерируются новые ключи)
- протоколы **предварительного распределения** ключей (для генерации новых ключей в **конференциях**)

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \rightarrow B : k$$

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- Если ключ передается в **открытом** виде, то ключ можно **подсмотреть** в канале передачи.
  - Как защищаться?

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- Если ключ передается в **открытом** виде, то ключ можно **подсмотреть** в канале передачи.
  - Как защищаться? Шифровать.

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- Если ключ передается в **открытом** виде, то ключ можно **подсмотреть** в канале передачи.
  - Как защищаться? Шифровать.
- Если не выполнять предварительную аутентификацию участника A, то у B нет гарантии, что ключи ему прислал именно A.
  - Как защищаться?

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- Если ключ передается в **открытом** виде, то ключ можно **подсмотреть** в канале передачи.
  - Как защищаться? Шифровать.
- Если не выполнять предварительную аутентификацию участника A, то у B нет гарантии, что ключи ему прислал именно A.
  - Как защищаться? Проводить аутентификацию сторон.

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- Если ключ передается в **открытом** виде, то ключ можно **подсмотреть** в канале передачи.
  - Как защищаться? Шифровать.
- Если не выполнять предварительную аутентификацию участника A, то у B нет гарантии, что ключи ему прислал именно A.
  - Как защищаться? Проводить аутентификацию сторон.
- Если нет контроля «свежести» (fresh) ключа, то атакующий может попытаться навязать использование старого ключа (который он, например, смог «как-то» получить). Это и есть replay-атака.
  - Как защищаться?

Цель

Цель

Передать от участника A к участнику B ранее сгенерированный ключ k (ключ, как правило, генерирует A):

$$A \to B : k$$

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- Если ключ передается в **открытом** виде, то ключ можно **подсмотреть** в канале передачи.
  - Как защищаться? Шифровать.
- Если не выполнять предварительную аутентификацию участника A, то у B нет гарантии, что ключи ему прислал именно A.
  - Как защищаться? Проводить аутентификацию сторон.
- Если нет контроля «свежести» (fresh) ключа, то атакующий может попытаться навязать использование старого ключа (который он, например, смог «как-то» получить). Это и есть replay-атака.
  - Как защищаться? Использовать механизмы контроля свежести ключа (метки времени).

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

- Шифрование ключа:

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

- Шифрование ключа: Да

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя:

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Нет

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Нет
- Защита от replay-атаки:

Без ЭЦП. Вариант 1

- Участник B имеем пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
& A: k \leftarrow \textsc{Generate} \\
A \rightarrow B: & c = E_{k_p^B}(k, \textsc{timestamp}, A) \\
& B: (k, \textsc{timestamp}, A) = D_{k_s^B}(c) \\
& B: \textsc{timestamp} \stackrel{?}{\in} [\textsc{time} - \Delta, \textsc{time} + \Delta]
\end{aligned}
$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Нет
- Защита от replay-атаки: Да

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

- Шифрование ключа:

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

- Шифрование ключа: Да

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя:

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Да

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Да
- Защита от replay-атаки:

Без ЭЦП. Вариант 2

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

$$
\begin{aligned}
A &: k_1 \leftarrow \textsc{Generate} \\
A \rightarrow B &: E_{k_p^B}(k_1, A) \\
B &: k_2 \leftarrow \textsc{Generate} \\
B \rightarrow A &: E_{k_p^A}(k_1, k_2) \\
A \rightarrow B &: E_{k_p^B}(k_2)
\end{aligned}
$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Да
- Защита от replay-атаки: Да

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \textsc{Generate}$$

$$A \rightarrow B : E_{k_p^B}(k, \textsc{timestamp}, \mathrm{SIGN}_{k_s^A}(B, \textsc{timestamp}, k))$$

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \textsc{Generate}$$
$$A \rightarrow B : E_{k_p^B}(k, \textsc{timestamp}, \mathrm{SIGN}_{k_s^A}(B, \textsc{timestamp}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \textsc{Generate}$$
$$A \rightarrow B : E_{k_p^B}(k, \textsc{timestamp}), \mathrm{SIGN}_{k_s^A}(B, \textsc{timestamp}, k)$$

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

Анализ

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \textsc{Generate}$$
$$A \rightarrow B : E_{k_p^B}(k, \textsc{timestamp}, \textsc{sign}_{k_s^A}(B, \textsc{timestamp}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \textsc{Generate}$$
$$A \rightarrow B : E_{k_p^B}(k, \textsc{timestamp}), \textsc{sign}_{k_s^A}(B, \textsc{timestamp}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \textsc{Generate}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \textsc{sign}_{k_s^A}(B, \textsc{timestamp}, E_{k_p^B}(A, k))$$

Анализ

- Шифрование ключа:

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

Анализ

- Шифрование ключа: Да

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя:

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Да

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Да
- Защита от replay-атаки:

С применением ЭЦП

- A имеет пару ключей: публичный ключ k_p^A и секретный k_s^A.
- B имеет пару ключей: публичный ключ k_p^B и секретный k_s^B.

Вариант 1. Authenticate then Encrypt (AtE), используется в SSL/TLS

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}, \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k))$$

Вариант 2. Authenticate & Encrypt (A&E), используется в SSH

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(k, \text{TIMESTAMP}), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, k)$$

Вариант 3. Encrypt then authenticate (EtA), используется в IPSec

$$A : k \leftarrow \text{GENERATE}$$
$$A \rightarrow B : E_{k_p^B}(A, k), \text{SIGN}_{k_s^A}(B, \text{TIMESTAMP}, E_{k_p^B}(A, k))$$

Анализ

- Шифрование ключа: Да
- Аутентификация отправителя: Да
- Защита от replay-атаки: Да

Передача ключа с использованием симметричной криптографии

- A имеет секретный ключ k_{AS} взаимодействия с доверенной стороной S.
- B имеет секретный ключ k_{BS} взаимодействия с доверенной стороной S.
- Доверенная сторона S имеет ключи k_{AS} и k_{BS} для взаимодействия с A и B соответственно.

Протокол Нидхема-Шрёдера[1]

$$
\begin{aligned}
A \to CA &: A, B, \text{NONCE}_A \\
CA &: k \leftarrow \text{GENERATE} \\
CA \to A &: E_{k_{AS}}(\text{NONCE}_A, B, k, Y), \text{где } Y = E_{k_{BS}}(k, A) \\
A \to B &: Y
\end{aligned}
$$

[1] В лекции 6 мы рассматривали урезанную версию протокола на основе

Передача ключа с использованием симметричной криптографии

- A имеет секретный ключ k_{AS} взаимодействия с доверенной стороной S.
- B имеет секретный ключ k_{BS} взаимодействия с доверенной стороной S.
- Доверенная сторона S имеет ключи k_{AS} и k_{BS} для взаимодействия с A и B соответственно.

Протокол Нидхема-Шрёдера[1]

$$
\begin{aligned}
A \to CA &: A, B, \text{NONCE}_A \\
CA &: k \leftarrow \text{GENERATE} \\
CA \to A &: E_{k_{AS}}(\text{NONCE}_A, B, k, Y), \text{где } Y = E_{k_{BS}}(k, A) \\
A \to B &: Y
\end{aligned}
$$

Проблема

Проверяя NONCE_A, участник A можут убедиться, что сообщение получено от S (т.е. replay-атака не проводится), однако у B нет гарантии, что сообщение Y не является старым (например, из предыдущей сессии). Для защиты используется так называемое «рукопожатие» (handshake).

[1] В лекции 6 мы рассматривали урезанную версию протокола на основе

Протокол Нидхема-Шрёдера для передачи ключа

- A имеет секретный ключ k_{AS} взаимодействия с доверенной стороной S.
- B имеет секретный ключ k_{BS} взаимодействия с доверенной стороной S.
- Доверенная сторона S имеет ключи k_{AS} и k_{BS} для взаимодействия с A и B соответственно.

Протокол

$$
\begin{aligned}
A \to CA &: A, B, \text{NONCE}_A \\
CA &: k \leftarrow \text{GENERATE} \\
CA \to A &: E_{k_{AS}}(\text{NONCE}_A, B, k, Y), \text{где } Y = E_{k_{BS}}(k, A) \\
A \to B &: Y \\
B \to A &: E_k(\text{NONCE}_B) \\
A \to B &: E_k(\text{NONCE}_B - 1)
\end{aligned}
$$

Протокол Нидхема-Шрёдера для передачи ключа

- A имеет секретный ключ k_{AS} взаимодействия с доверенной стороной S.
- B имеет секретный ключ k_{BS} взаимодействия с доверенной стороной S.
- Доверенная сторона S имеет ключи k_{AS} и k_{BS} для взаимодействия с A и B соответственно.

Протокол

$$
\begin{aligned}
A \to CA &: A, B, \text{NONCE}_A \\
CA &: k \leftarrow \text{GENERATE} \\
CA \to A &: E_{k_{AS}}(\text{NONCE}_A, B, k, Y), \text{где } Y = E_{k_{BS}}(k, A) \\
A \to B &: Y \\
B \to A &: E_k(\text{NONCE}_B) \\
A \to B &: E_k(\text{NONCE}_B - 1)
\end{aligned}
$$

Проблема

Если атакующий C сможет перехватить Y и как-то извлечь из него ключ k, то после этого C может навязывать участнику B использование скомпрометированного ключа k. Как от этого защититься?

Передача ключа с **односторонней** аутентификацией на основе сертификатов публичных ключей

Замечание

В предыдущих протоколах на основе асимметричной криптографии предполагается, что участники заранее обменялись открытыми ключами (без подмены).

Передача ключа с **односторонней** аутентификацией на основе сертификатов публичных ключей

Замечание

В предыдущих протоколах на основе асимметричной криптографии предполагается, что участники заранее обменялись открытыми ключами (без подмены).

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача ключа с **односторонней** аутентификацией на основе сертификатов публичных ключей

Замечание

В предыдущих протоколах на основе асимметричной криптографии предполагается, что участники заранее обменялись открытыми ключами (без подмены).

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача сгенерированного ключа:

$$
\begin{aligned}
A &: k \leftarrow \textsc{Generate} \\
A \rightarrow CA &: A, B \\
CA \rightarrow A &: Cert_A, Cert_B \\
A \rightarrow B &: Cert_A, Cert_B, E_{k_p^B}(\textsc{sign}_{k_s^A}(k, \textsc{timestamp}))
\end{aligned}
$$

Передача ключа с **односторонней** аутентификацией на основе сертификатов публичных ключей

Замечание

В предыдущих протоколах на основе асимметричной криптографии предполагается, что участники заранее обменялись открытыми ключами (без подмены).

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача сгенерированного ключа:

$$\begin{aligned} A &: k \leftarrow \textsc{Generate} \\ A \rightarrow CA &: A, B \\ CA \rightarrow A &: Cert_A, Cert_B \\ A \rightarrow B &: Cert_A, Cert_B, E_{k_p^B}(\textsc{sign}_{k_s^A}(k, \textsc{timestamp})) \end{aligned}$$

Анализ:

Передача ключа с **односторонней** аутентификацией на основе сертификатов публичных ключей

Замечание

В предыдущих протоколах на основе асимметричной криптографии предполагается, что участники заранее обменялись открытыми ключами (без подмены).

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача сгенерированного ключа:

$$
\begin{aligned}
A &: k \leftarrow \textsc{Generate} \\
A \rightarrow CA &: A, B \\
CA \rightarrow A &: Cert_A, Cert_B \\
A \rightarrow B &: Cert_A, Cert_B, E_{k_p^B}(\textsc{sign}_{k_s^A}(k, \textsc{timestamp}))
\end{aligned}
$$

Анализ: Самостоятельно

Передача ключа с **взаимной** аутентификацией на основе сертификатов публичных ключей

Передача ключа с **взаимной** аутентификацией на основе сертификатов публичных ключей

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача ключа с **взаимной** аутентификацией на основе сертификатов публичных ключей

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача ключа с взаимной аутентификацией (CCITT X.509):

$$A \to CA : A, B$$

$$CA \to A : Cert_A, Cert_B$$

$$A : k_1 \leftarrow \textsc{Generate}$$

$$A \to B : Cert_A, Cert_B, D_A, \mathrm{SIGN}_{k_s^A}(D_A), \quad D_A = (\textsc{timestamp}_A, R_A, B, E_{k_p^B}(k_1))$$

$$B : k_2 \leftarrow \textsc{Generate}$$

$$B \to A : D_B, \mathrm{SIGN}_{k_s^B}(D_B), \quad D_B = (\textsc{timestamp}_B, R_B, A, R_A, E_{k_p^A}(k_2))$$

$$A \to B : R_B, B, \mathrm{SIGN}_{k_s^A}(R_B, B)$$

Общий ключ: $k = k_1 \oplus k_2$

Передача ключа с **взаимной** аутентификацией на основе сертификатов публичных ключей

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача ключа с взаимной аутентификацией (CCITT X.509):

$$A \to CA : A, B$$

$$CA \to A : Cert_A, Cert_B$$

$$A : k_1 \leftarrow \textsc{Generate}$$

$$A \to B : Cert_A, Cert_B, D_A, \mathrm{SIGN}_{k_s^A}(D_A), \quad D_A = (\textsc{timestamp}_A, R_A, B, E_{k_p^B}(k_1))$$

$$B : k_2 \leftarrow \textsc{Generate}$$

$$B \to A : D_B, \mathrm{SIGN}_{k_s^B}(D_B), \quad D_B = (\textsc{timestamp}_B, R_B, A, R_A, E_{k_p^A}(k_2))$$

$$A \to B : R_B, B, \mathrm{SIGN}_{k_s^A}(R_B, B)$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ:

Передача ключа с **взаимной** аутентификацией на основе сертификатов публичных ключей

- A имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^A.
- B имеет сертификат $Cert_{CA}$ ключа доверенной стороны CA и свой секретный ключ k_s^B.
- Доверенная сторона CA имеет сертификаты $Cert_A$ и $Cert_B$ сторон A и B соответственно.

Передача ключа с взаимной аутентификацией (CCITT X.509):

$$A \to CA : A, B$$

$$CA \to A : Cert_A, Cert_B$$

$$A : k_1 \leftarrow \textsc{Generate}$$

$$A \to B : Cert_A, Cert_B, D_A, \mathrm{SIGN}_{k_s^A}(D_A), \quad D_A = (\textsc{timestamp}_A, R_A, B, E_{k_p^B}(k_1))$$

$$B : k_2 \leftarrow \textsc{Generate}$$

$$B \to A : D_B, \mathrm{SIGN}_{k_s^B}(D_B), \quad D_B = (\textsc{timestamp}_B, R_B, A, R_A, E_{k_p^A}(k_2))$$

$$A \to B : R_B, B, \mathrm{SIGN}_{k_s^A}(R_B, B)$$

Общий ключ: $k = k_1 \oplus k_2$

Анализ: Самостоятельно

Цель

Цель

Сгенерировать совместно участниками A и B ключ k.

Цель

Цель

Сгенерировать совместно участниками A и B ключ k.

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

Цель

Цель

Сгенерировать совместно участниками A и B ключ k.

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- По передаваемым в канале сообщениям наблюдатель может попытаться сгенерировать такой же ключ.
  - Как защищаться?

Цель

Цель

Сгенерировать совместно участниками A и B ключ k.

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- По передаваемым в канале сообщениям наблюдатель может попытаться сгенерировать такой же ключ.
  - Как защищаться? Использовать односторонние функции.

Цель

Цель

Сгенерировать совместно участниками A и B ключ k.

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- По передаваемым в канале сообщениям наблюдатель может попытаться сгенерировать такой же ключ.
  - Как защищаться? Использовать односторонние функции.
- Если не выполнять предварительную взаимную аутентификацию, то можно сгенерировать ключ с противником.
  - Как защищаться?

Цель

Цель

Сгенерировать совместно участниками A и B ключ k.

Какие могут быть атаки на такие протоколы в рамках модели Долева-Яо?

- По передаваемым в канале сообщениям наблюдатель может попытаться сгенерировать такой же ключ.
  - Как защищаться? Использовать односторонние функции.
- Если не выполнять предварительную взаимную аутентификацию, то можно сгенерировать ключ с противником.
  - Как защищаться? Проводить аутентификацию сторон.

Протокол Диффи-Хэллмана

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол:

$$\begin{aligned} A &: a \in_R \mathbb{Z}_p \\ A \to B &: X = g^a \\ B &: b \in_R \mathbb{Z}_p \\ B \to A &: Y = g^b \end{aligned}$$

Общий ключ: $k = g^{ab} = X^b = Y^a$

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол:

$$\begin{aligned} A &: a \in_R \mathbb{Z}_p \\ A \to B &: X = g^a \\ B &: b \in_R \mathbb{Z}_p \\ B \to A &: Y = g^b \end{aligned}$$

Общий ключ: $k = g^{ab} = X^b = Y^a$

Анализ:

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол:

$$\begin{aligned} A &: a \in_R \mathbb{Z}_p \\ A \to B &: X = g^a \\ B &: b \in_R \mathbb{Z}_p \\ B \to A &: Y = g^b \end{aligned}$$

Общий ключ: $k = g^{ab} = X^b = Y^a$

Анализ:

- Секретность:

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол:

$$\begin{aligned} A &: a \in_R \mathbb{Z}_p \\ A \to B &: X = g^a \\ B &: b \in_R \mathbb{Z}_p \\ B \to A &: Y = g^b \end{aligned}$$

Общий ключ: $k = g^{ab} = X^b = Y^a$

Анализ:

- Секретность: Да, так как по $X = (g, y = g^b, g)$ сложно найти g^{ab}.

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол:

$$\begin{aligned} A &: a \in_R \mathbb{Z}_p \\ A \to B &: X = g^a \\ B &: b \in_R \mathbb{Z}_p \\ B \to A &: Y = g^b \end{aligned}$$

Общий ключ: $k = g^{ab} = X^b = Y^a$

Анализ:

- Секретность: Да, так как по $X = (g, y = g^b, g)$ сложно найти g^{ab}.
- Аутентификация:

Протокол Диффи-Хэллмана

- $\langle g \rangle$ — циклическая группа желательно простого порядка p, g – порождающий элемент.

Протокол:

$$\begin{aligned} A &: a \in_R \mathbb{Z}_p \\ A \to B &: X = g^a \\ B &: b \in_R \mathbb{Z}_p \\ B \to A &: Y = g^b \end{aligned}$$

Общий ключ: $k = g^{ab} = X^b = Y^a$

Анализ:

- Секретность: Да, так как по $X = (g, y = g^b, g)$ сложно найти g^{ab}.
- Аутентификация: Нет (MITM, можно защититься, например, с помощью ЭЦП).

Свойства

Свойства

Секретность будущих сообщений (**forward-secrecy**)

При утечке долговременного секретного ключа в асимметрическом алгоритме (private key), все **будущие** сообщения можно будет расшифровать «на лету» (то есть все будущие сессионные или кратковременные ключи могут быть скомпрометированы).

Свойства

Секретность будущих сообщений (**forward-secrecy**)

При утечке долговременного секретного ключа в асимметрическом алгоритме (private key), все **будущие** сообщения можно будет расшифровать «на лету» (то есть все будущие сессионные или кратковременные ключи могут быть скомпрометированы).

Секретность прошлых сообщений (**backward-secrecy**)

При утечке долговременного секретного ключа в асимметрическом алгоритме (private key), все **прошлые** сообщения записанные противником (network attacker threat model) можно будет расшифровать.

Пример 1

Исходные данные:

- A – клиент (например, Интернет-браузер)
- B – сервер (например, web-сервер)
- Протокол выработки ключа – протокол Диффи-Хэллмана

Пример 1

Исходные данные:

- A – клиент (например, Интернет-браузер)
- B – сервер (например, web-сервер)
- Протокол выработки ключа – протокол Диффи-Хэллмана

Условия на протокол:

- A выбирает $a(\in \mathbb{Z})$ **случайно**
- B использует **постоянное** значение $b(\in \mathbb{Z})$

Пример 1

Исходные данные:

- A – клиент (например, Интернет-браузер)
- B – сервер (например, web-сервер)
- Протокол выработки ключа – протокол Диффи-Хэллмана

Условия на протокол:

- A выбирает $a(\in \mathbb{Z})$ **случайно**
- B использует **постоянное** значение $b(\in \mathbb{Z})$

Это Fixed-DH

Этот протокол не обладает ни FS, ни BS!

Пример 2

Исходные данные:

- A – клиент (например, Интернет-браузер)
- B – сервер (например, web-сервер)
- Протокол выработки ключа – протокол Диффи-Хэллмана

Пример 2

Исходные данные:

- A – клиент (например, Интернет-браузер)
- B – сервер (например, web-сервер)
- Протокол выработки ключа – протокол Диффи-Хэллмана

Условия на протокол:

- A выбирает $a(\in \mathbb{Z})$ **случайно**
- B выбирает $b(\in \mathbb{Z})$ **случайно**

Пример 2

Исходные данные:

- A – клиент (например, Интернет-браузер)
- B – сервер (например, web-сервер)
- Протокол выработки ключа – протокол Диффи-Хэллмана

Условия на протокол:

- A выбирает $a(\in \mathbb{Z})$ **случайно**
- B выбирает $b(\in \mathbb{Z})$ **случайно**

Это Ephemenral-DH (DHE)

Этот протокол обладает и FS, и BS!

Заключение

Спасибо за внимание!