Secure Sockets Layer

Защита информационных ресурсов компьютерных систем и сетей

Определение

SSL—криптографический протокол, который обеспечивает установление безопасного соединения между клиентом и сервером.

Свойства канала

- Канал является частным. Шифрование используется для всех сообщений после простого диалога, который служит для определения секретного ключа.
- Канал аутентифицирован. Серверная сторона диалога всегда аутентифицируется, в то время как клиентская - аутентифицируется опционно.
- Канал надежен. Транспортировка сообщений включает в себя проверку целостности (с привлечением MAC).

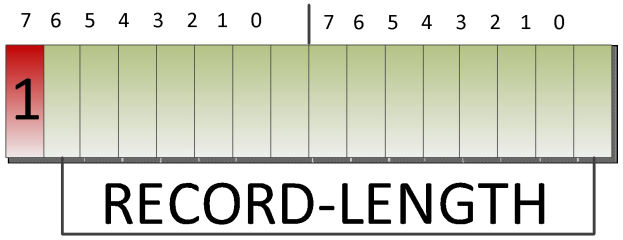
Содержание

Спецификация протокола записей SSL

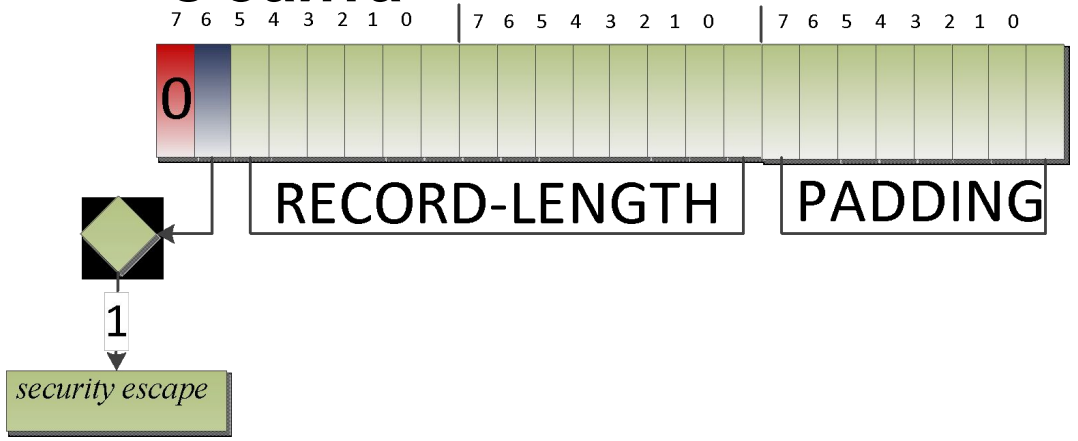
- Формат заголовка записи SSL
- Формат информационных записей SSL

Формат заголовка записи SSL

2 байта



3 байта



RECORD-LENGTH =

((byte[0] & 0x7F << 8)) | byte[1];

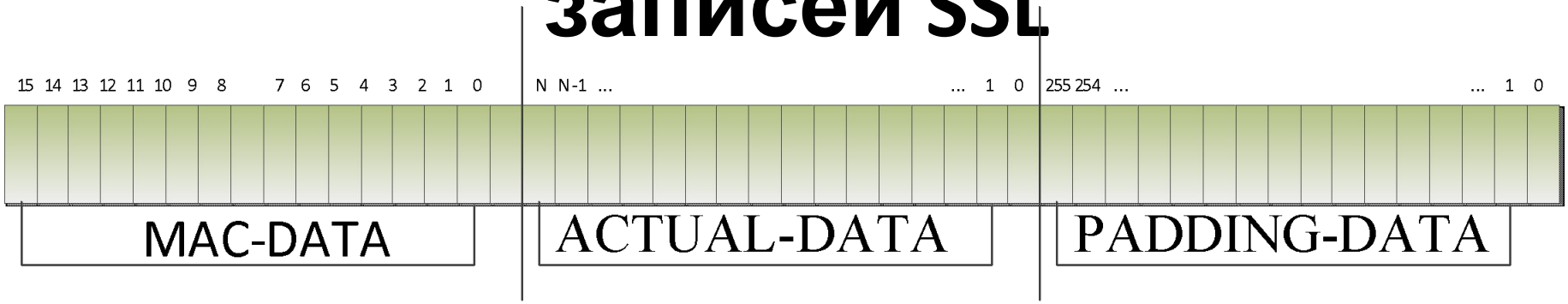
RECORD-LENGTH =

((byte[0] & 0x3F) << 8)) | byte[1];

IS-ESCAPE = (byte[0] & 0x40) != 0;

PADDING = byte[2];

Формат информационных записей SSL



MAC-DATA[MAC-SIZE]

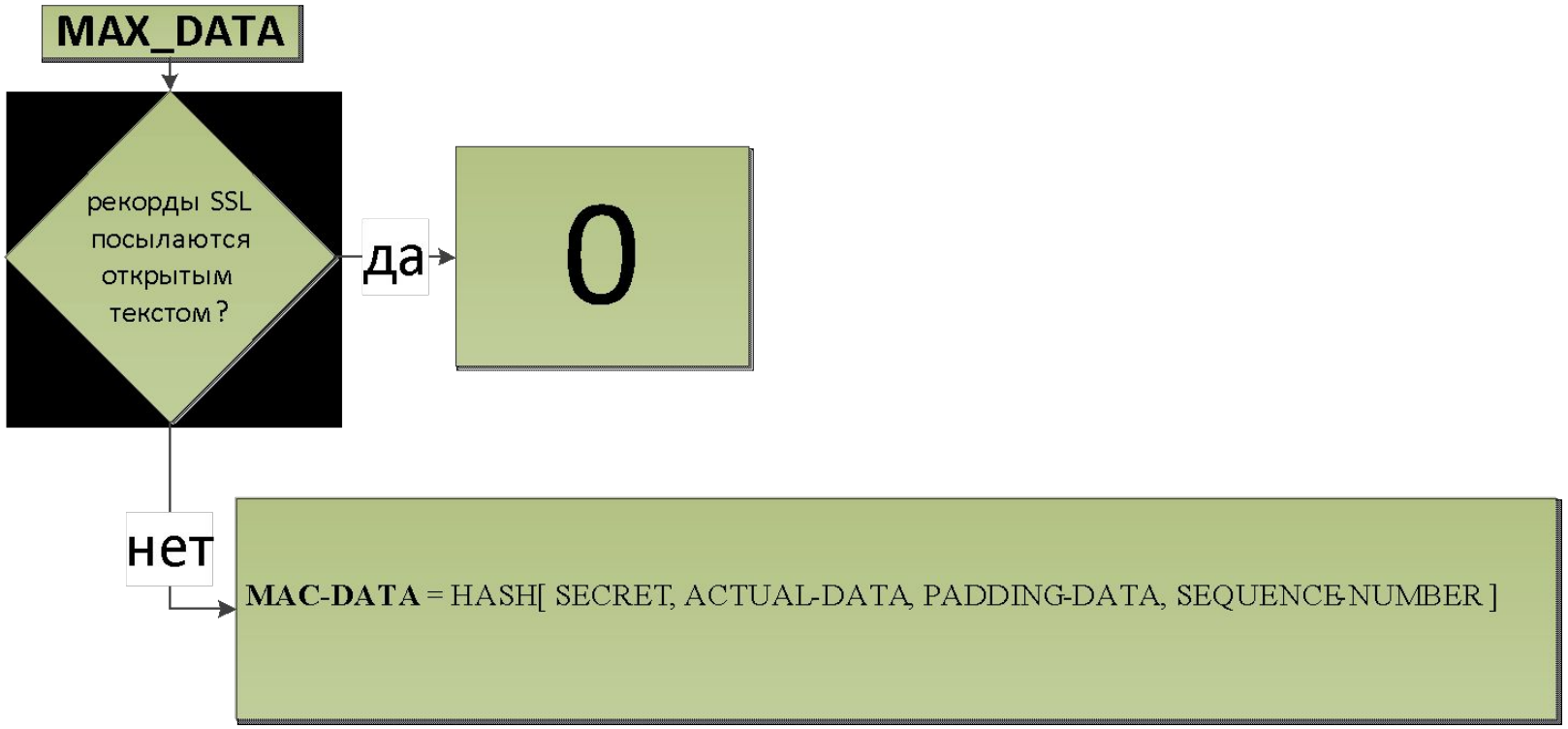
Для MD2 и MD5 MAC-SIZE равен 16 байтам

ACTUAL-DATA[N]

PADDING-DATA[PADDING]

N = RECORD-LENGTH **-** MAC-SIZE - PADDING

MAC-DATA

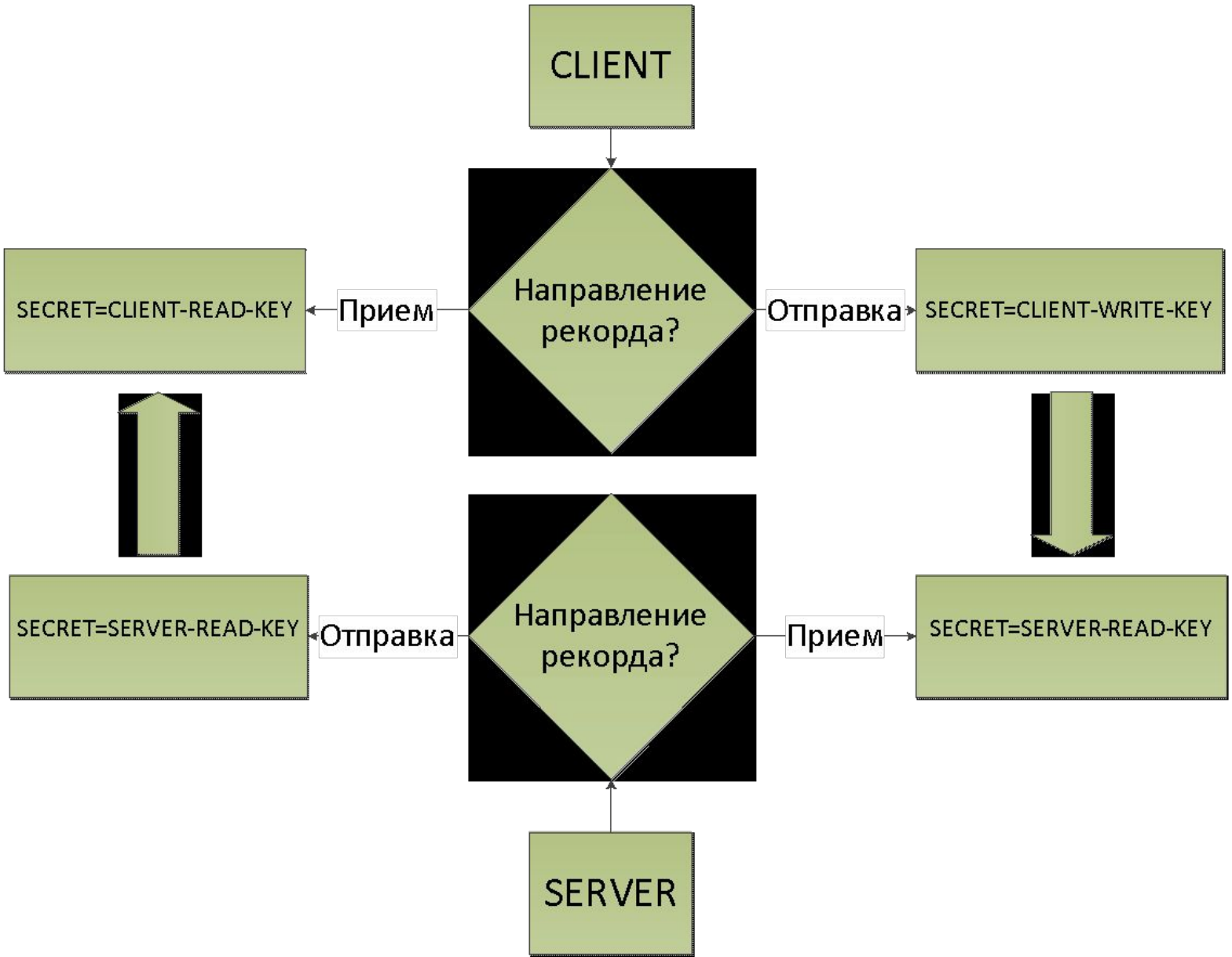

MAX_DATA
рекорды SSL посылаются открытым текстом?
да
0
нет
MAC-DATA = HASH[SECRET, ACTUAL-DATA, PADDING-DATA, SEQUENCE-NUMBER]

HASH

Тип хэш-функции определяется параметром **CIPHER-CHOICE**

Для **MD2** и **MD5 MAC-SIZE** равен 16 байтам

SECRET


CLIENT
Направление рекорда?
Прием
SECRET=CLIENT-READ-KEY
Отправка
SECRET=CLIENT-WRITE-KEY
Направление рекорда?
Отправка
SECRET=SERVER-READ-KEY
Прием
SECRET=SERVER-READ-KEY
SERVER

SEQUENCE-NUMBER

Счетчик, который инкрементируется как сервером, так и получателем. Для каждого направления передачи, используется пара счетчиков (один для отправителя, другой для получателя). При отправлении сообщения счетчик инкрементируется. Порядковыми номерами являются 32-битовые целые числа без знака, которые при переполнении обнуляются.

Спецификация протокола диалога SSL

- **Протокол диалога SSL**
- **Типовой протокол обмена сообщениями**

Протокол диалога SSL

- Фаза 1
- Фаза 2

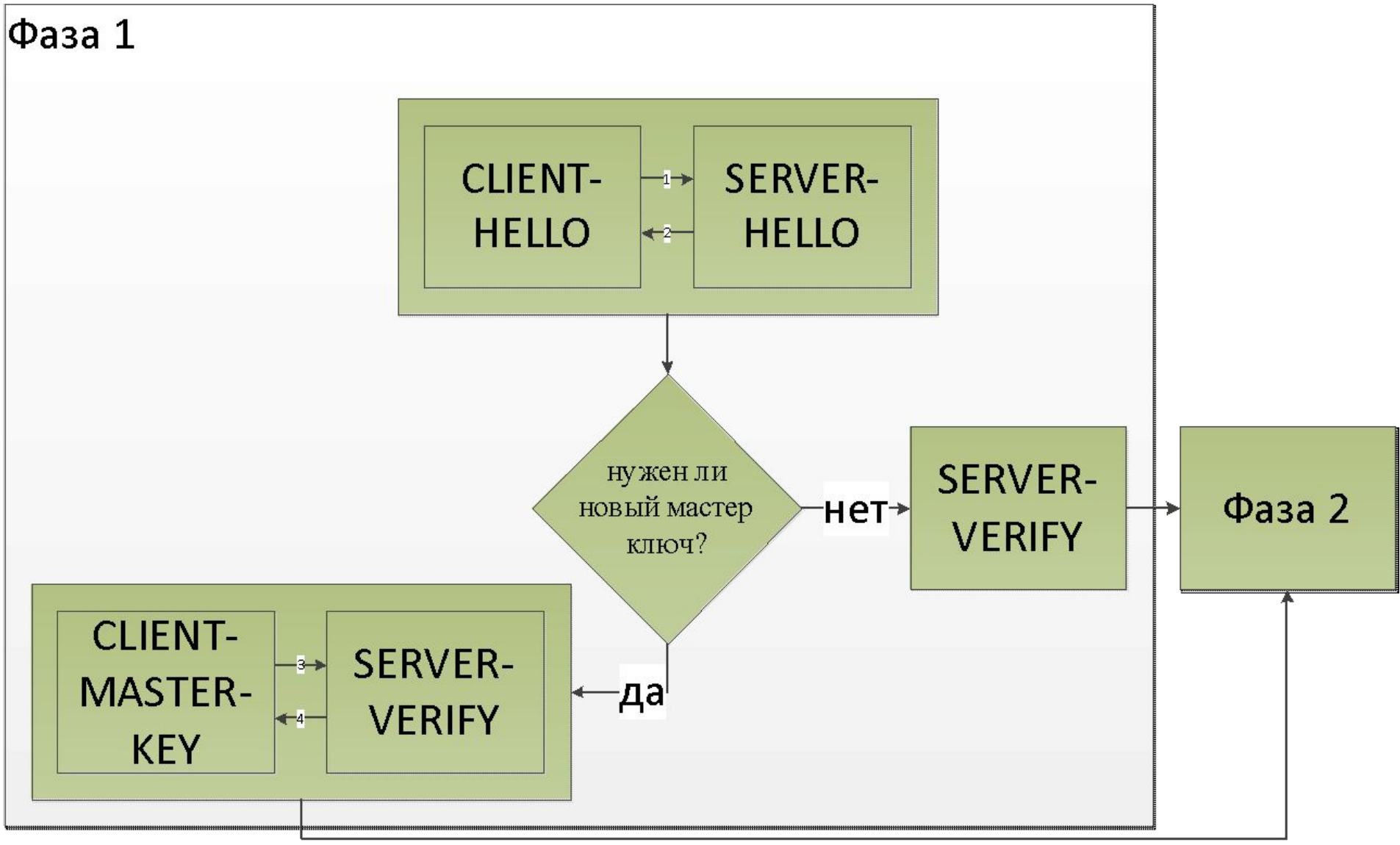
Фаза 1

1) Клиент инициирует диалог посылкой сообщения **CLIENT-HELLO**. Сервер получает сообщение **CLIENT-HELLO**, обрабатывает его и откликается сообщением **SERVER-HELLO**
2) Когда нужен новый мастер ключ, сообщение **SERVER-HELLO** будет содержать достаточно данных, чтобы клиент мог сформировать такой ключ
3) Клиент генерирует мастер ключ и посылает сообщение **CLIENT-MASTER-KEY** или сообщение ERROR, если информация сервера указывает, что клиент и сервер не могут согласовать базовый шифр
4) после того как мастер ключ определен, сервер посылает клиенту сообщение **SERVER-VERIFY**

SERVER-HELLO

- сертификат сервера
- список базовых шифров
- идентификатор соединения

Блок-схема Фазы 1



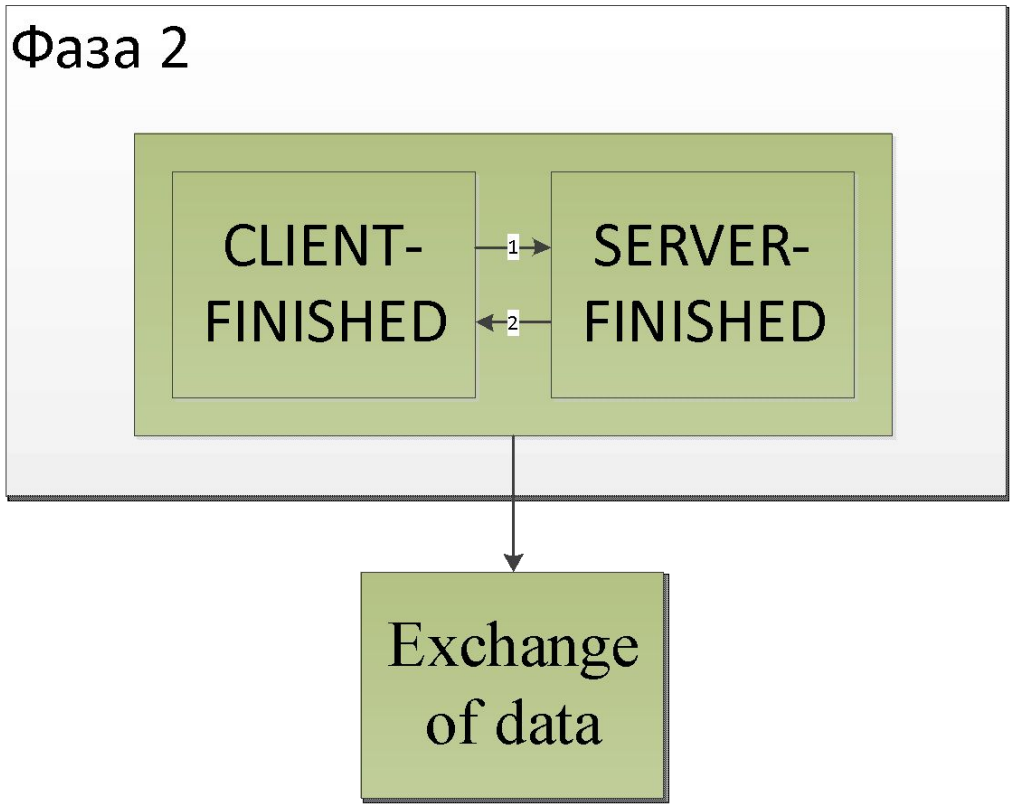
Фаза 2

Сервер уже аутентифицирован клиентом на первой фазе, по этой причине здесь осуществляется аутентификация клиента.

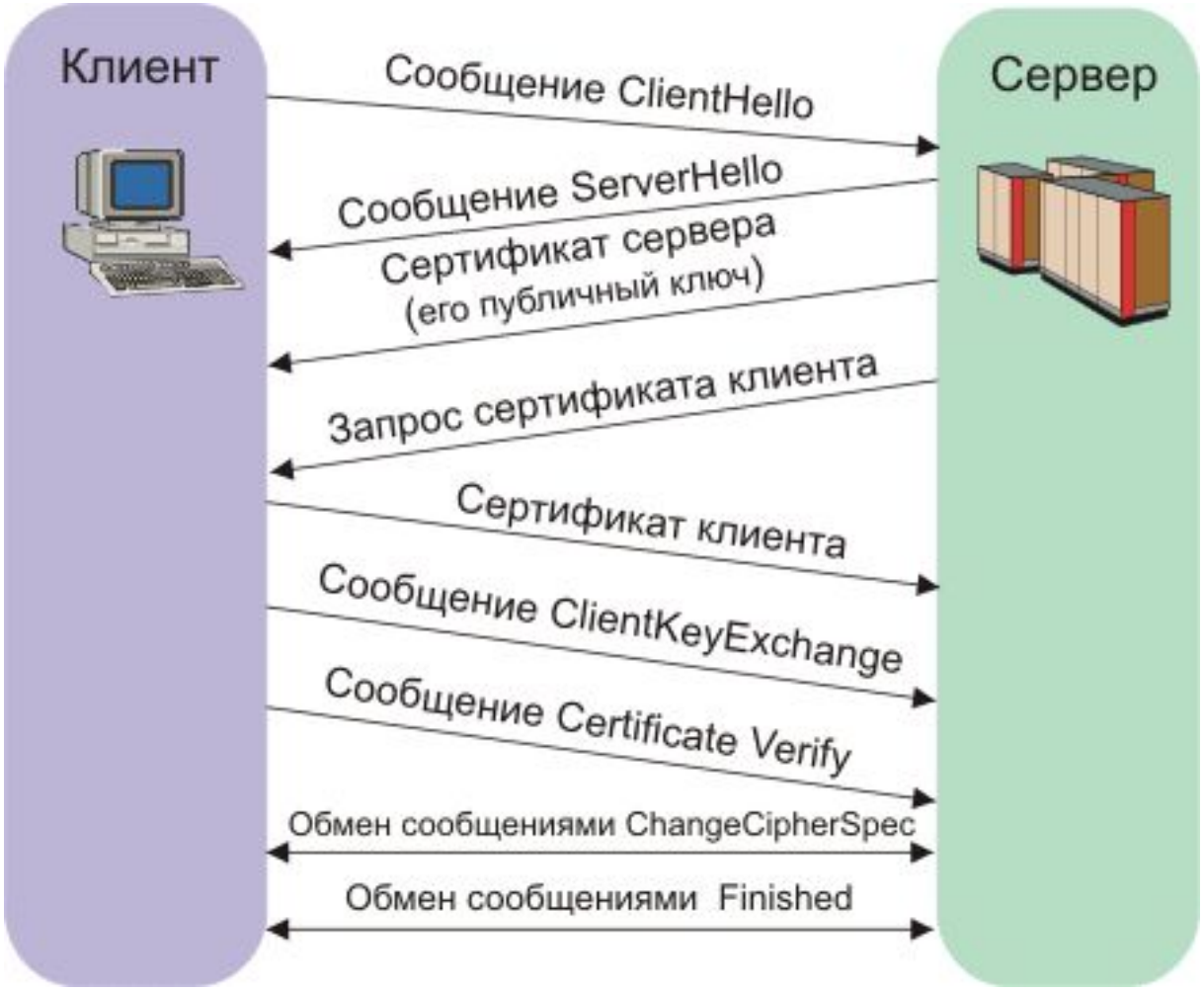
Когда один партнер выполнил аутентификацию другого партнера, он посылает сообщение **finished.**

В случае клиента сообщение **CLIENT-FINISHED** содержит зашифрованную форму идентификатора **CONNECTION-ID**, которую должен верифицировать сервер. Если верификация терпит неудачу, сервер посылает сообщение

Блок-схема Фазы 2



Типовой протокол обмена сообщениями



При отсутствии идентификатора сессии

Client-hello	C → S	challenge, cipher_specs
server-hello	S → C	connection-id,server_certificate,cipher_specs
client-master-key	C → S	{master_key}server_public_key
client-finish	C → S	{connection-id}client_write_key
server-verify	S → C	{challenge}server_write_key
server-finish	S → C	{new_session_id}server_write_key

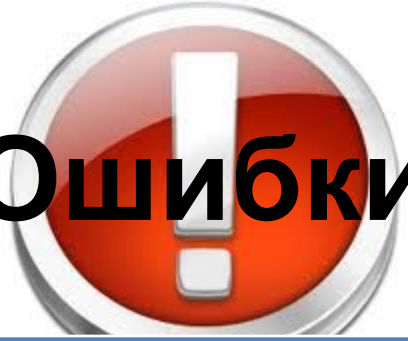
Идентификатор сессии найден клиентом и сервером

client-hello	C → S	challenge, session_id, cipher_specs
server-hello	S → C	connection-id, session_id_hit
client-finish	C → S	{connection-id}client_write_key
server-verify	S → C	{challenge}server_write_key
server-finish	S → C	{session_id}server_write_key

Использован идентификатор сессии и аутентификация клиента

client-hello	C → S:	challenge, session_id, cipher_specs
server-hello	S → C:	connection-id, session_id_hit
client-finish	C → S:	{connection-id}client_write_key
server-verify	S → C:	{challenge}server_write_key
request-certificate	S → C:	{auth_type,challenge'}server_write_key
client-certificate	C → S:	{cert_type,client_cert, response_data}client_write_key
server-finish	S → C:	{session_id}server_write_key

Ошибки



Ошибки	Описание
NO-CIPHER-ERROR	Эта ошибка присылается клиентом серверу, когда он не может найти шифр или размер ключа, который поддерживается также и сервером.
NO-CERTIFICATE-ERROR	Когда послано сообщение **REQUEST-CERTIFICATE**, эта ошибка может быть прислана, если клиент не имеет сертификата.
BAD-CERTIFICATE-ERROR	Такой отклик присылается, когда сертификат по какой-то причине считается принимающей стороной плохим. Эта ошибка устранима (только для аутентификации клиента).
UNSUPPORTED-CERTIFICATE-TYPE-ERROR	Этот отклик присылается, когда клиент/сервер получает тип сертификата, который он не поддерживает. Эта ошибка устранима (только для аутентификации клиента).

Протокольные сообщения клиента

- CLIENT-HELLO (Фаза 1; посылается открыто)
- CLIENT-MASTER-KEY (Фаза 1; посылается вначале открыто)
- CLIENT-CERTIFICATE (Фаза 2; посылается шифрованным)
- CLIENT-FINISHED (Фаза 2; посыл шифрованным)

CLIENT-HELLO (Фаза 1; посылается открыто)

- char MSG-CLIENT-HELLO
  char CLIENT-VERSION-MSB
  char CLIENT-VERSION-LSB
  char CIPHER-SPECS-LENGTH-MSB
  char CIPHER-SPECS-LENGTH-LSB
  char SESSION-ID-LENGTH-MSB
  char SESSION-ID-LENGTH-LSB
  char CHALLENGE-LENGTH-MSB
  char CHALLENGE-LENGTH-LSB
  char CIPHER-SPECS-DATA[(MSB<<8)|LSB
  char SESSION-ID-DATA[(MSB<<8)|LSB]
  char CHALLENGE-DATA[(MSB<<8)|LSB]

CLIENT-MASTER-KEY (Фаза 1; посылается вначале открыто)

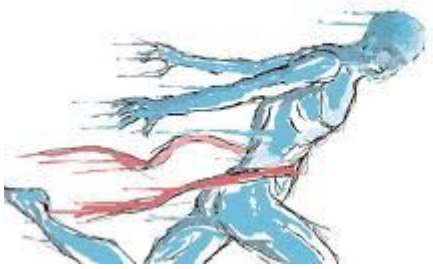
char MSG-CLIENT-MASTER-KEY
char CIPHER-KIND[3]
char CLEAR-KEY-LENGTH-MSB
char CLEAR-KEY-LENGTH-LSB
char ENCRYPTED-KEY-LENGTH-MSB
char ENCRYPTED-KEY-LENGTH-LSB
char KEY-ARG-LENGTH-MSB
char KEY-ARG-LENGTH-LSB
char CLEAR-KEY-DATA[MSB<<8|LSB]
char ENCRYPTED-KEY-DATA[MSB<<8|LSB]
char KEY-ARG-DATA[MSB<<8|LSB]

CLIENT-CERTIFICATE (Фаза 2; посылается шифрованным)

char MSG-CLIENT-CERTIFICATE
char CERTIFICATE-TYPE
char CERTIFICATE-LENGTH-MSB
char CERTIFICATE-LENGTH-LSB
char RESPONSE-LENGTH-MSB
char RESPONSE-LENGTH-LSB
char CERTIFICATE-DATA[MSB<<8|LSB]
char RESPONSE-DATA[MSB<<8|LSB]

CLIENT-FINISHED (Фаза 2; посылается шифрованным)

char MSG-CLIENT-FINISHED
char CONNECTION-ID[N-1]



Протокольные сообщения сервера

- SERVER-HELLO (Фаза 1; посылается открыто)
- SERVER-VERIFY (Фаза 1; посылается шифрованным)
- SERVER-FINISHED (Фаза 2; посылается зашифрованным)
- REQUEST-CERTIFICATE (Фаза 2; посылается шифрованным)

SERVER-HELLO (Фаза 1; посылается открыто

char MSG-SERVER-HELLO
char SESSION-ID-HIT
char CERTIFICATE-TYPE
char SERVER-VERSION-MSB
char SERVER-VERSION-LSB
char CERTIFICATE-LENGTH-MSB
char CERTIFICATE-LENGTH-LSB
char CIPHER-SPECS-LENGTH-MSB
char CIPHER-SPECS-LENGTH-LSB
char CONNECTION-ID-LENGTH-MSB
char CONNECTION-ID-LENGTH-LSB
char CERTIFICATE-DATA[MSB<<8|LSB]
char CIPHER-SPECS-DATA[MSB<<8|LSB]
char CONNECTION-ID-DATA[MSB<<8|LSB]

CIPHER-KIND

Набор	Уровень безопасности	Описание
DES-CBC3-MD5	Очень высокий	Тройной DES в режиме CBC, хэш MD5, 168-битный ключ сессии
DES-CBC3-SHA	Очень высокий	Тройной DES в режиме CBC, хэш SHA, 168-битный ключ сессии
RC4-MD5	Высокий	RC4, хэш MD5, 128-битный ключ
RC4-SHA	Высокий	RC4, хэш SHA, 128-битный ключ
RC2-CBC-MD5	Высокий	RC2 в режиме CBC, хэш MD5, 128-битный ключ
DES-CBC-MD5	Средний	DES в режиме CBC, хэш MD5, 56-битный ключ
DES-CBC-SHA	Средний	DES в режиме CBC, хэш SHA, 56-битный ключ
EXP-DES-CBC-SHA	Низкий	DES в режиме CBC, хэш SHA, 40-битный ключ
EXP-RC4-MD5	Низкий	Экспортное качество RC4, хэш MD5, 40-битный ключ
EXP-RC2-CBC-MD5	Низкий	Экспортное качество RC2, хэш MD5, 40-битный ключ
NULL-MD5	-	Без шифрования, хэш MD5, только аутентификация
NULL-SHA		Без шифрования, хэш SHA, только аутентификация

SERVER-VERIFY (Фаза 1; посылается шифрованным)

char MSG-SERVER-VERIFY
char CHALLENGE-DATA[N-1]

SERVER-FINISHED (Фаза 2; посылается зашифрованным)

char MSG-SERVER-FINISHED
char SESSION-ID-DATA[N-1]

REQUEST-CERTIFICATE (Фаза 2; посылается шифрованным)

char MSG-REQUEST-CERTIFICATE
char AUTHENTICATION-TYPE
char CERTIFICATE-CHALLENGE-DATA[N-2]

Атаки

- Раскрытие шифров
- Атака открытого текста
- Атака отклика
- Человек посередине