# Защита информации

Павел Юдаев

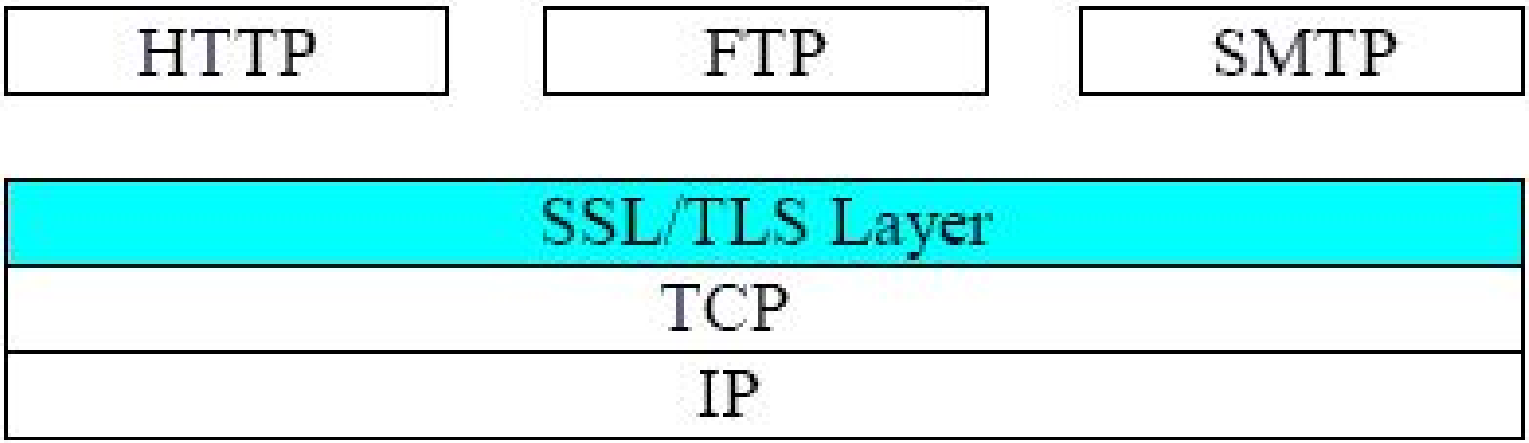
МГТУ им. Баумана, Кафедра ИУ-9

Москва, 2014

# Раздел 16 - Протокол TLS/SSL

| Модель OSI | | |
|---|---|---|
| **Тип данных** | **Уровень (layer)** | **Функции** |
| Данные | 7. Прикладной (application) | Доступ к сетевым службам |
| Поток | 6. Уровень представления (presentation) | Представление и шифрование данных |
| Сеансы | 5. Сеансовый (session) | Управление сеансом связи |
| Сегменты | 4. Транспортный (transport) | Прямая связь между конечными пунктами и надежность |
| Пакеты / Датаграммы | 3. Сетевой (network) | Определение маршрута и логическая адресация |
| Кадры | 2. Канальный (data link) | Физическая адресация |
| Биты | 1. Физический (physical) | Работа со средой передачи, сигналами и двоичными данными |

L7: HTTP, POP3, FTP, TELNET, RCP, PGP, **SSH**
L6: LPP (lw present. prot.), RDP (rem. desktop), X25 PAD (packet asm/disasm prot.), **SSL/TLS** (иниц-я - L5)
L5: L2TP, PPTP, NetBIOS, RPC
L4: TCP, UDP
L3: IP/IPv4/IPv6, **IPsec**, ICMP, RIP (routing inform. prot.)
L2: Ethernet, 802.11 wLAN, TokenRing
L1: BlueTooth, IRDA, 802.11 WiFi


HTTP
FTP
SMTP
SSL/TLS Layer
TCP
IP

**Secure Socket Layer / Transport Layer Security**

Цели:

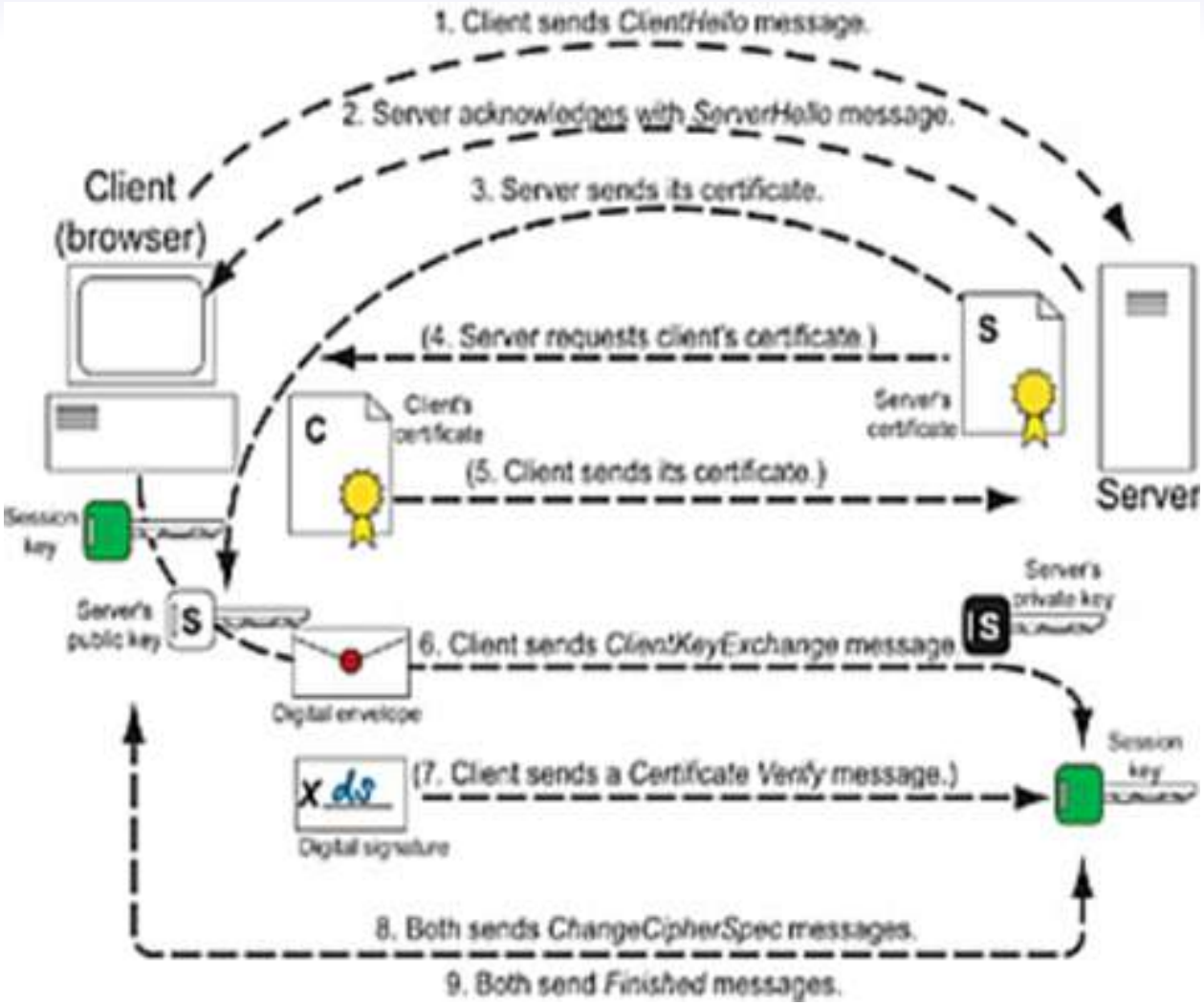
- Аутентификация сервера - ЭЦП (сертификаты)
- Аутентификация клиента - почти не исп.
- Конфиденциальность, целостность передачи информации между приложениями
- Защита от вопроизведения

## История SSL/TLS.

- SSL 1.0 - взломана во время презентации в MIT. Атака воспроизведения, нет целостности данных, поточный шифр. (1994)
- SSL 2.0 - Netscape Navigator 1.0 (1995)
- SSL 3.0 - публичное обсуждение. Предложен как стандарт IETF (1996). Работа не завершена.
- TLS 1.0 - стандарт Internet Engineering Task Force. (1999)
- TLS 1.1 - IETF, 2006. Защита от padding oracle attack, implicit IV attack (BEAST)
- TLS 1.2 - IETF, 2008. Добавлены новые шифры и пр. технические улучшения.

Подробней - см. таблицы
`http://en.wikipedia.org/wiki/Transport_Layer_Security#Algorithm`

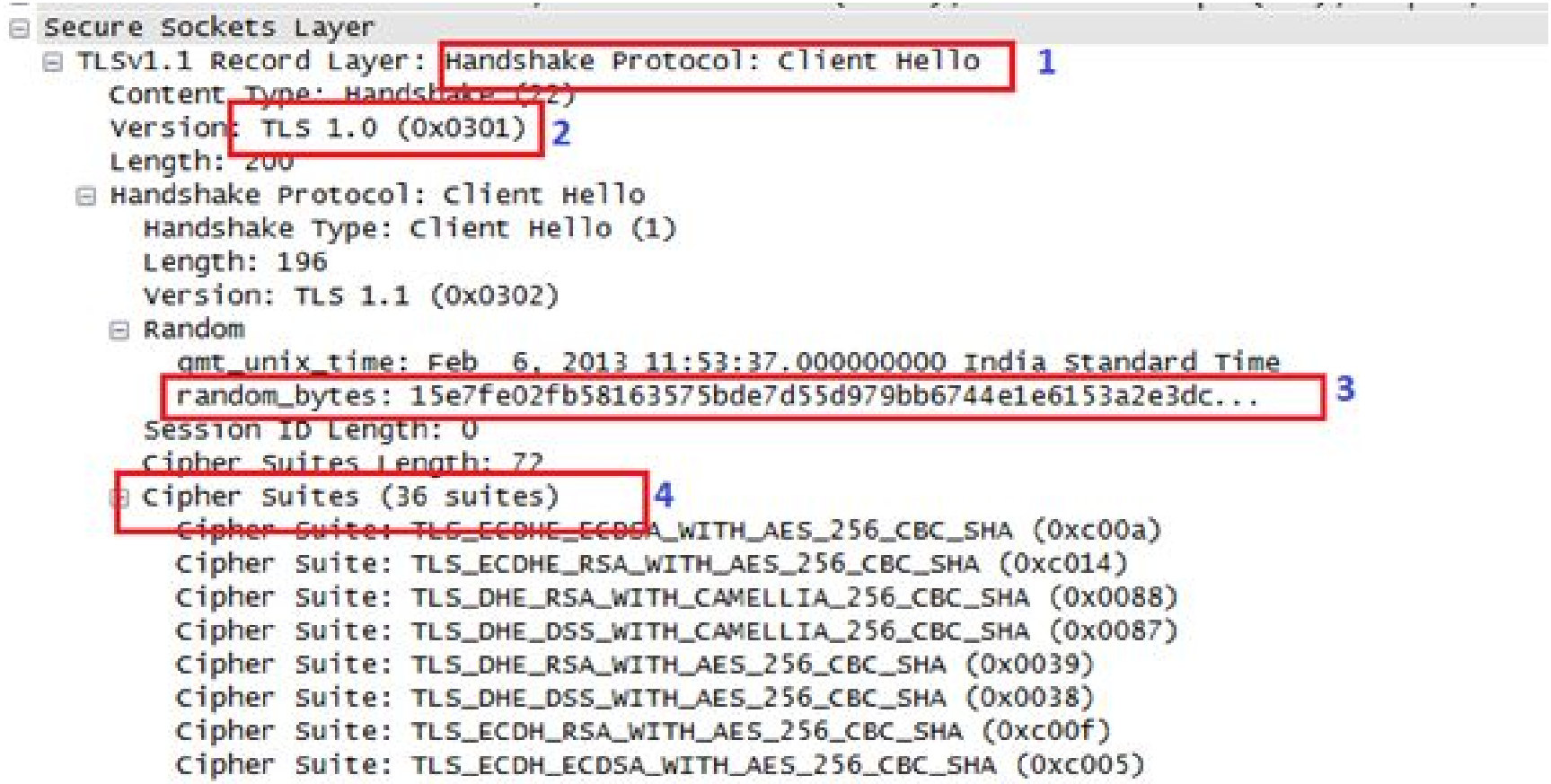
# Раздел 16 - Протокол TLS/SSL


1. Client sends ClientHello message.
2. Server acknowledges with ServerHello message.
3. Server sends its certificate.
Client
(browser)
(4. Server requests client's certificate.)
S
Server's certificate
C
Client's certificate
(5. Client sends its certificate.)
Server
Session key
Server's public key
S
Server's private key
S
6. Client sends ClientKeyExchange message.
Digital envelope
(7. Client sends a Certificate Verify message.)
Session key
Digital signature
8. Both sends ChangeCipherSpec messages.
9. Both send Finished messages.

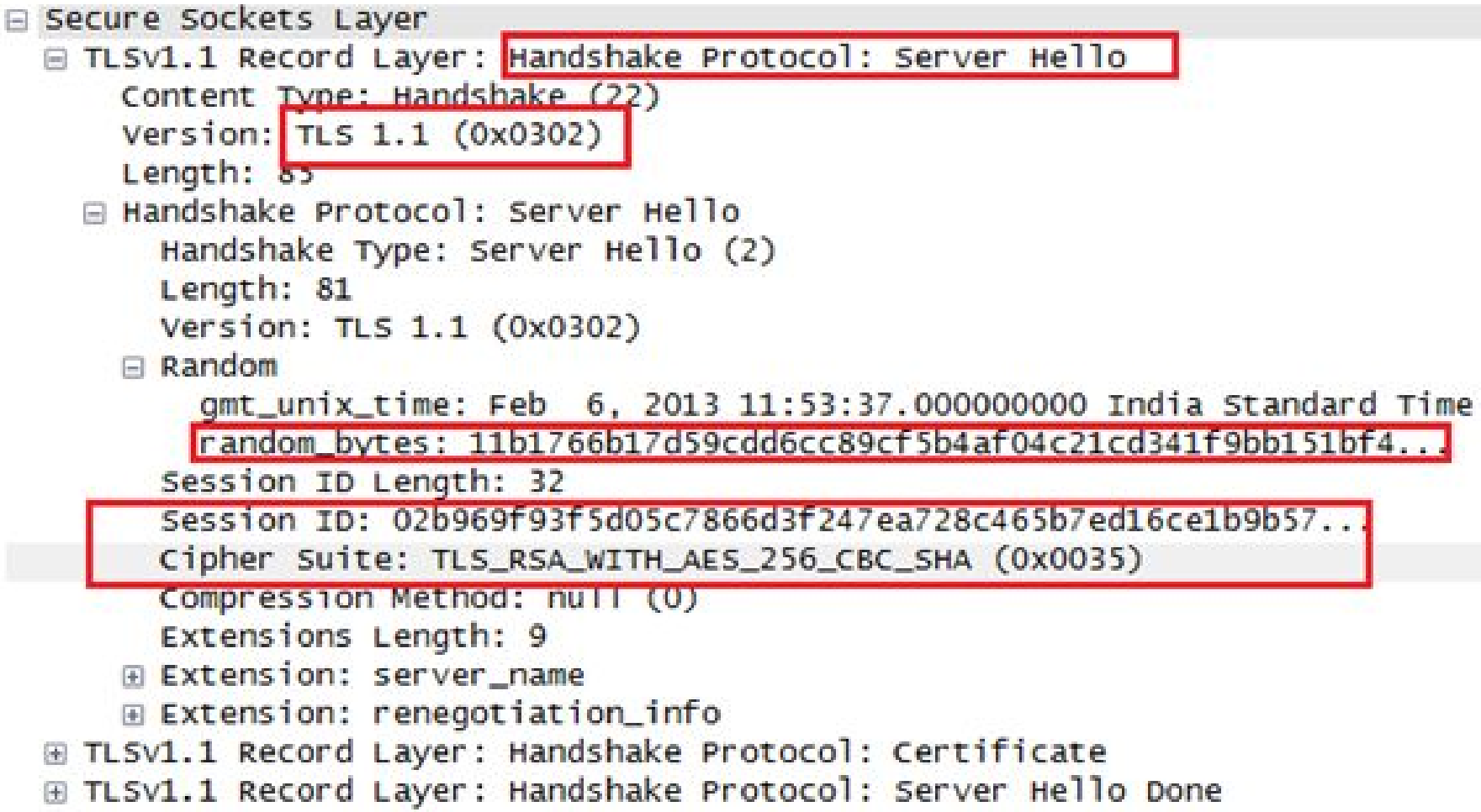
*(Источник: [2])*

1. Инициализация сессии, выбор метода обмена ключами, шифрования и х/ф.

- Client Hello: client nonce, ID сессии (resumed handshake - восстановление сессии), список поддерживаемых методов обмена ключами, шифрования и х/ф.
- Server Hello: server nonce, выбор шифра из списка.




*(Источник здесь и далее - [1])*

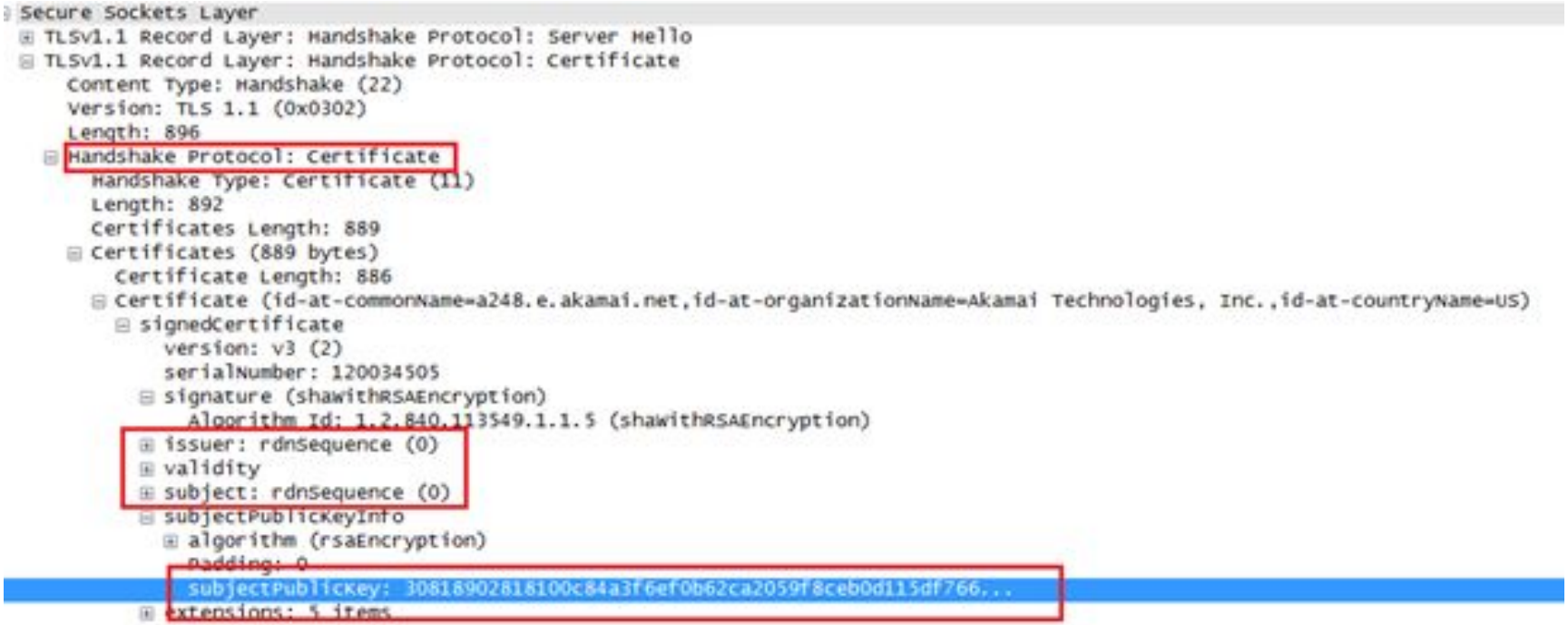

Secure Sockets Layer
TLSv1.1 Record Layer: Handshake Protocol: Server Hello
Content Type: Handshake (22)
Version: TLS 1.1 (0x0302)
Length: 85
Handshake Protocol: Server Hello
Handshake Type: Server Hello (2)
Length: 81
Version: TLS 1.1 (0x0302)
Random
gmt_unix_time: Feb 6, 2013 11:53:37.000000000 India Standard Time
random_bytes: 11b1766b17d59cdd6cc89cf5b4af04c21cd341f9bb151bf4...
Session ID Length: 32
Session ID: 02b969f93f5d05c7866d3f247ea728c465b7ed16ce1b9b57...
Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
Compression Method: null (0)
Extensions Length: 9
Extension: server_name
Extension: renegotiation_info
TLSv1.1 Record Layer: Handshake Protocol: Certificate
TLSv1.1 Record Layer: Handshake Protocol: Server Hello Done

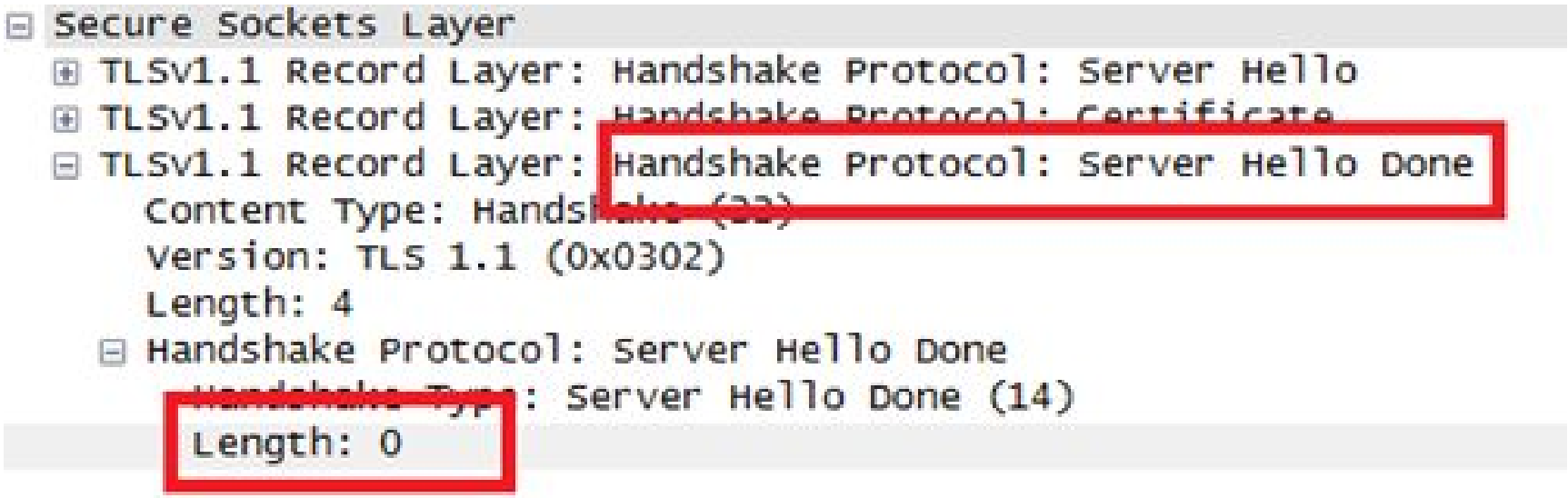
2. Аутентификация сервера

Server certificate
Сервер присылает свой открытый ключ, заверенный сертификатом.
Клиент может проверить сертификат у CA.

2a. Сервер может потребовать аутентификацию клиента.

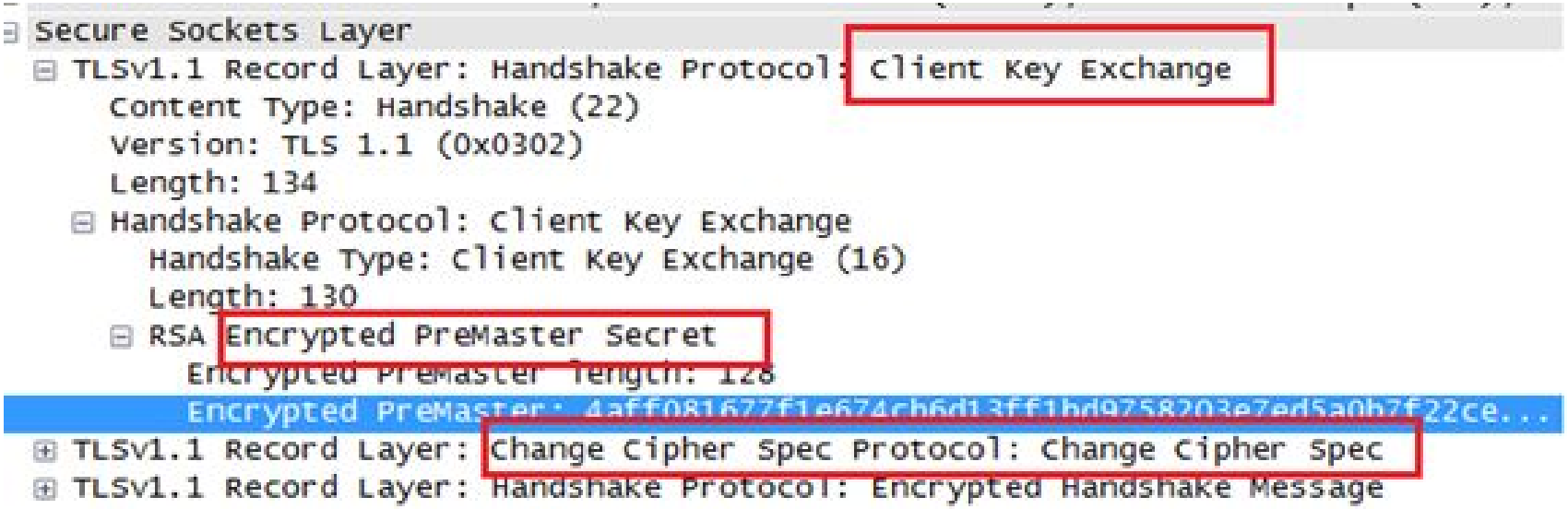
Server hello done
Не требует аутентификацию клиента


Secure Sockets Layer
TLSv1.1 Record Layer: Handshake Protocol: Server Hello
TLSv1.1 Record Layer: Handshake Protocol: Certificate
Content Type: Handshake (22)
Version: TLS 1.1 (0x0302)
Length: 896
Handshake Protocol: Certificate
Handshake Type: Certificate (11)
Length: 892
Certificates Length: 889
Certificates (889 bytes)
Certificate Length: 886
Certificate (id-at-commonName=a248.e.akamai.net,id-at-organizationName=Akamai Technologies, Inc.,id-at-countryName=US)
signedCertificate
version: v3 (2)
serialNumber: 120034505
signature (shaWithRSAEncryption)
Algorithm Id: 1.2.840.113549.1.1.5 (shaWithRSAEncryption)
issuer: rdnSequence (0)
validity
subject: rdnSequence (0)
subjectPublicKeyInfo
algorithm (rsaEncryption)
Padding: 0
subjectPublicKey: 30818902818100c84a3f6ef0b62ca2059f8ceb0d115df766...
extensions: 5 items


Secure Sockets Layer
TLSv1.1 Record Layer: Handshake Protocol: Server Hello
TLSv1.1 Record Layer: Handshake Protocol: Certificate
TLSv1.1 Record Layer: Handshake Protocol: Server Hello Done
Content Type: Handshake (22)
Version: TLS 1.1 (0x0302)
Length: 4
Handshake Protocol: Server Hello Done
Handshake Type: Server Hello Done (14)
Length: 0

3. Передача premaster secret

Клиент создает случайный premaster secret, шифрует публичным ключом сервера и отправляет.
Это сообщение также содержит уведомление о применении в дальнейшем симметричного шифрования.


Secure Sockets Layer
TLSv1.1 Record Layer: Handshake Protocol: Client Key Exchange
Content Type: Handshake (22)
Version: TLS 1.1 (0x0302)
Length: 134
Handshake Protocol: Client Key Exchange
Handshake Type: Client Key Exchange (16)
Length: 130
RSA Encrypted PreMaster Secret
Encrypted PreMaster Length: 128
Encrypted PreMaster: 4aff0816...
TLSv1.1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
TLSv1.1 Record Layer: Handshake Protocol: Encrypted Handshake Message

4,5. Создание master secret и ключей.

Оба вычисляют:

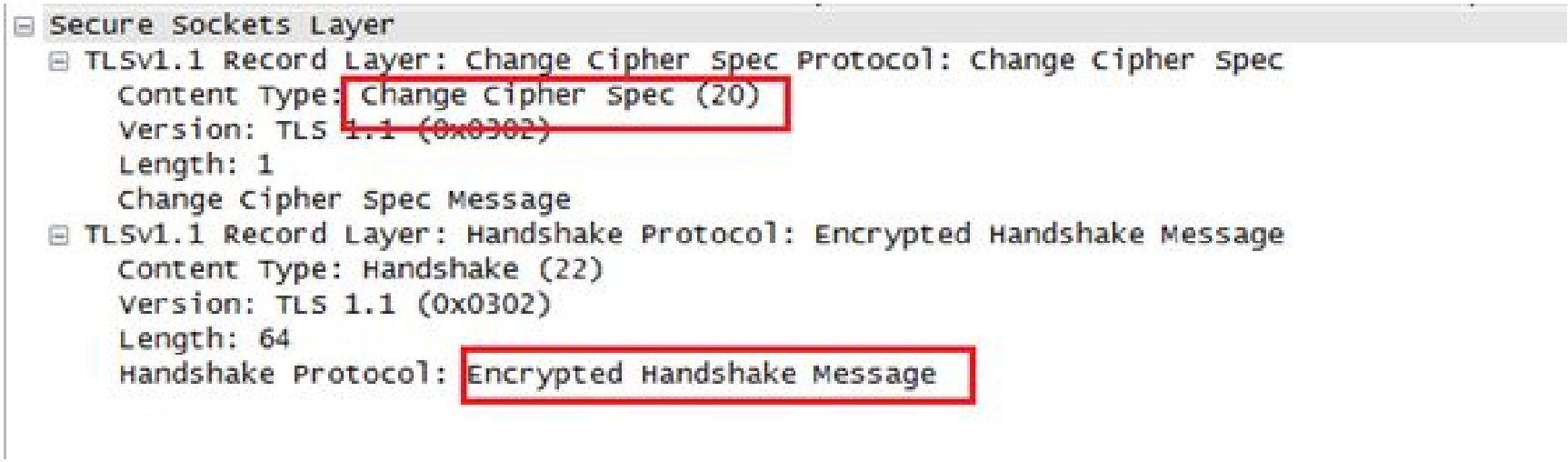
master_secret = PRF(pre_master_secret || “master secret” || ClientHello.nonce || ServerHello.nonce)

PRF = SHA256.

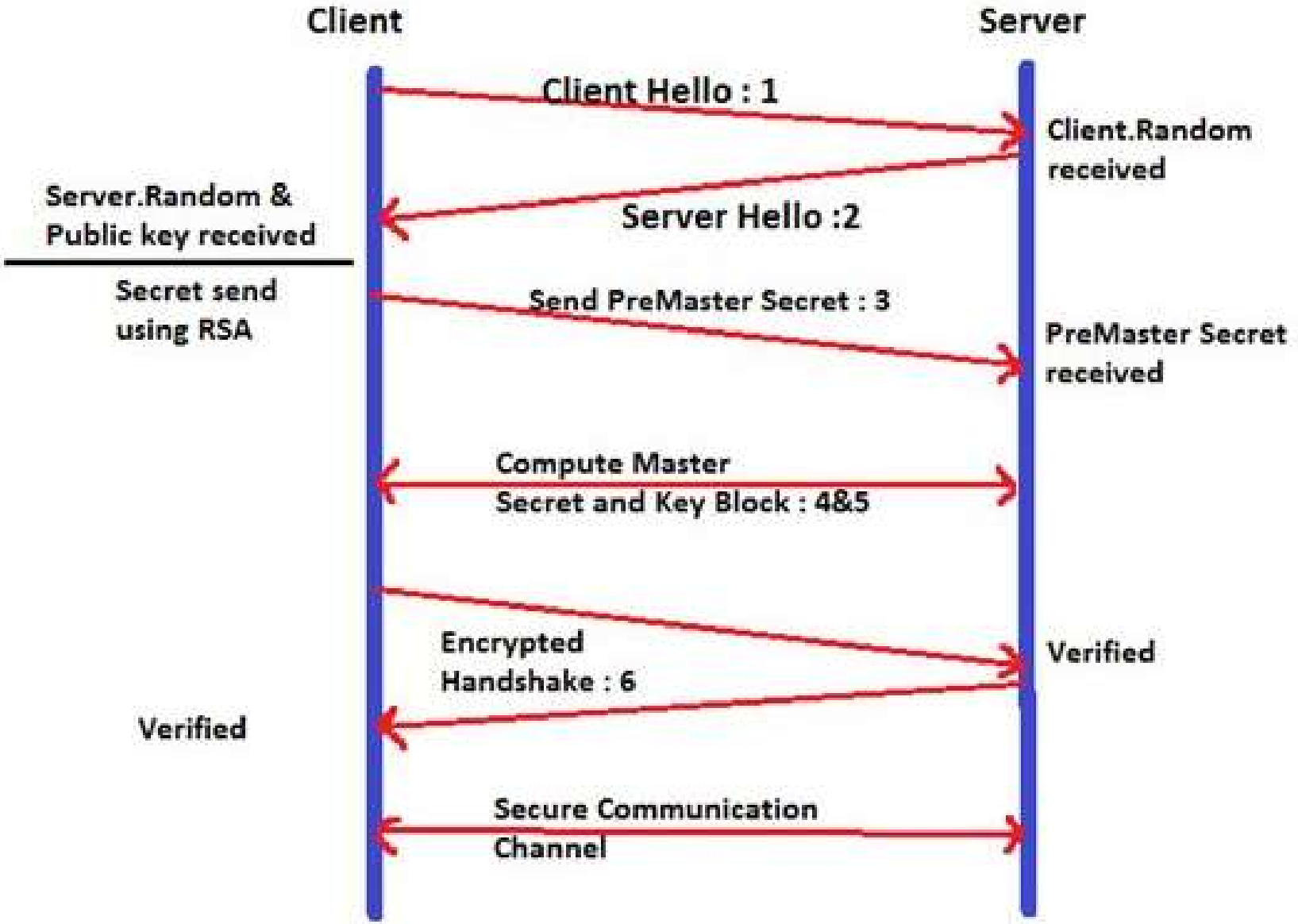
key_schedule = PRF(master_secret || “key expansion” || ClientHello.nonce || ServerHello.nonce) - сессионные ключи.

6. Клиент посылает зашифрованное сессионным ключом (заверенное шифрование) “client finished”

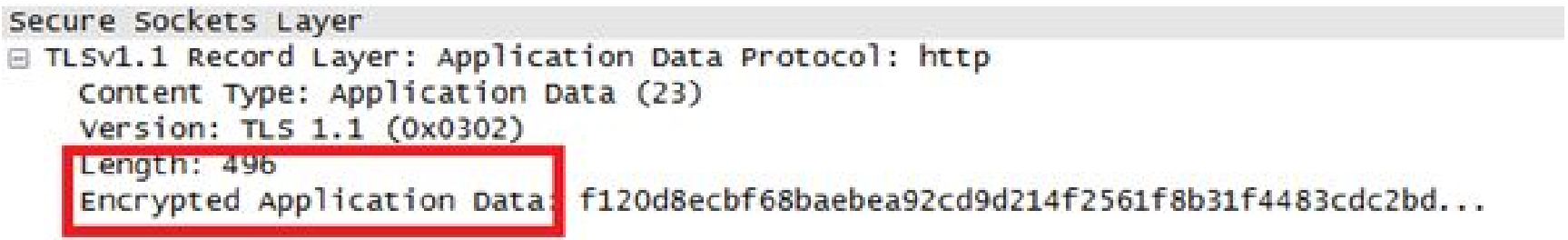
7. Сервер расшифровывает сообщение.
Неверно - отказ в сессии.
Верно - подтверждает прием: отвечает зашифрованным “server finished”.


Secure Sockets Layer
TLSv1.1 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
Content Type: Change Cipher Spec (20)
Version: TLS 1.1 (0x0302)
Length: 1
Change Cipher Spec Message
TLSv1.1 Record Layer: Handshake Protocol: Encrypted Handshake Message
Content Type: Handshake (22)
Version: TLS 1.1 (0x0302)
Length: 64
Handshake Protocol: Encrypted Handshake Message

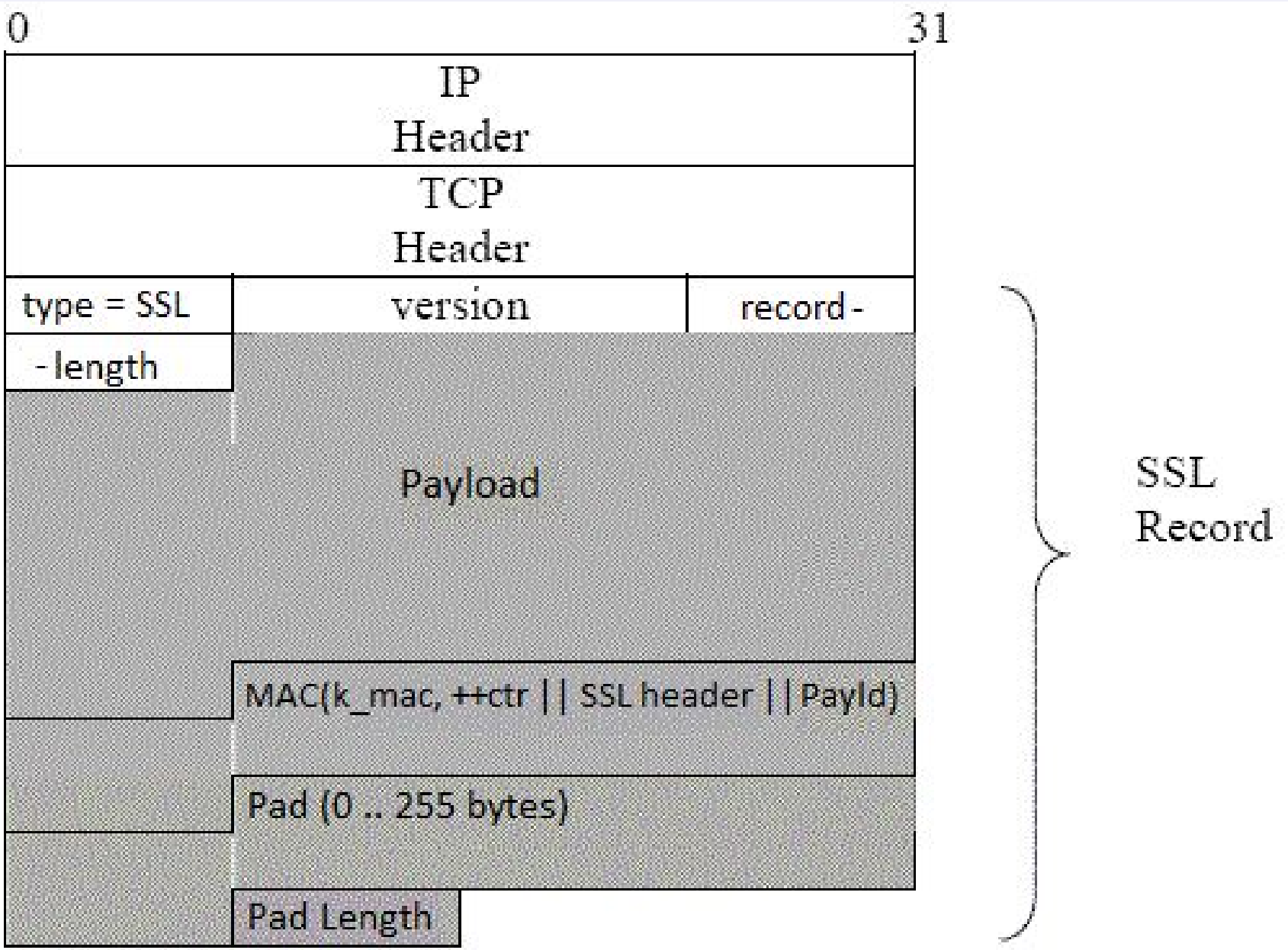
Общий вид handshake:



Сессия установлена. Данные передаются в зашифрованном виде.




(Подробней о структуре пакетов сессии см. секцию “Атаки на TLS 1.0” ранее.)


0
31
IP Header
TCP Header
type = SSL
version
record -
- length
Payload
MAC(k_mac, ++ctr || SSL header || Payld)
Pad (0 .. 255 bytes)
Pad Length
SSL Record

Шифрование (E): MtE.
Напр., HMAC-SHA1, CBC-AES-128.

$k_{cs} = (k_I, k_E)$

header: тип, версия протокола; длина сообщения (без заголовка). ctr не посылается.

1) $payload := compress(data)$
2) $tag = S(k_I, (++ ctr_{cs} || header || payload))$
3) $pad =$ дополнение до длины блока $(payload || tag)$
4) $E(IV, k_E, payload || tag || pad)$ с новым случайным IV.
5) приписать в начало сообщения открытый header

Сообщение $= header || E(payload || tag || pad)$

(см. спецификацию в RFC-5246,
http://tools.ietf.org/html/rfc5246#section-6)

# Раздел 16 - Протокол TLS/SSL

Достоинства TLS:

- Конфиденциальность данных, защита от изменений и воспроизведения.
- HTTPS сессии “знают” о том, что они используют TLS. Клиенты уверены в том, что соединение безопасное.
- TLS реализован на уровне приложений, значит, не в ядре ОС. Не нужна поддержка ОС. Встроен в браузеры, не требует настройки на стороне клиента.
- Можно раздать разные права доступа разным приложениям
- Аутентификация пользователя, а не оборудования
- TLS-VPN: удаленный доступ в защищенную сеть из браузера внешней машины по HTTPS. Не нужен спец. софт и настройки.
- Сессия закрывается явным образом при отключении клиента: TLS close, TCP FIN.
- Используется в электронной коммерции
- Netscape Commerce server (1995) - попытка монетизац. SSL

Недостатки TLS:

- PKI требует много ресурсов, т.к. исп. асимметричную криптографию. “Dependence on a PKI that cannot work before the ocean is boiled”
- Медленное и ресурсоемкое восстановление сессии.
- Проблемы масштабирования и потребления памяти при наличии большого числа TCP соединений. Каждый клиент устанавливает свое соединение с хостом. IPsec лучше масштабируется. (VPN между офисами компании.)

## Литература к лекции

1. Parul Garg, *TLS handshake* http://resources.infosecinstitute.com/cryptography-101-with-ssl/

2*. MITOpenCourseWare, *Network and Computer Security course.* http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-857-network-and-computer-security-fall-2003/ - взяты некоторые рисунки

3. Sourav Mukhopadhyay, *The Secure Sockets Layer*. http://www.facweb.iitkgp.ernet.in/~sourav/SSL.pdf