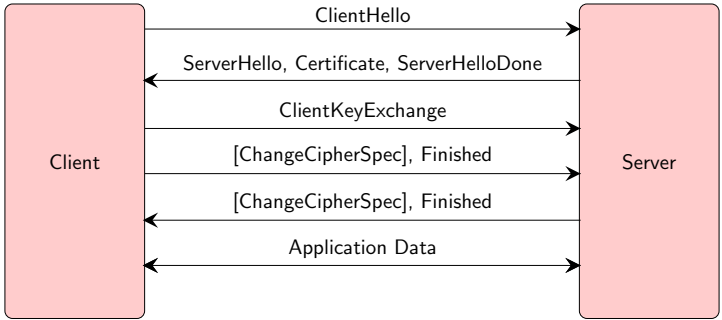
MTAT.07.017
Applied Cryptography

Transport Layer Security (TLS)
Advanced Features

University of Tartu

Spring 2021

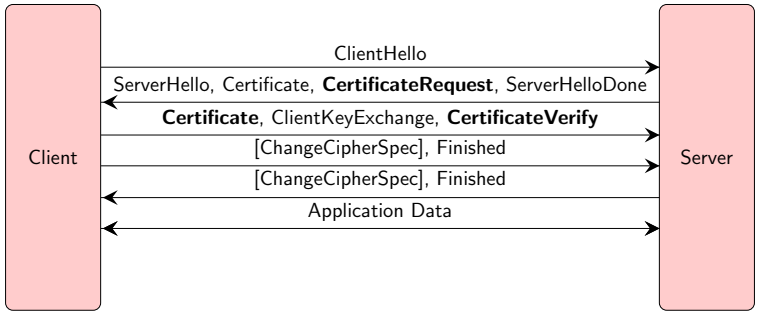
Server-authenticated TLS



Client is usually authenticated on the application level by some shared secret (e.g., password). This can go wrong:

- Server can be impersonated
- Server can be compromised
- Password might be reused in other services
- Password can be guessed
- Password can be phished

Client Certificate Authentication

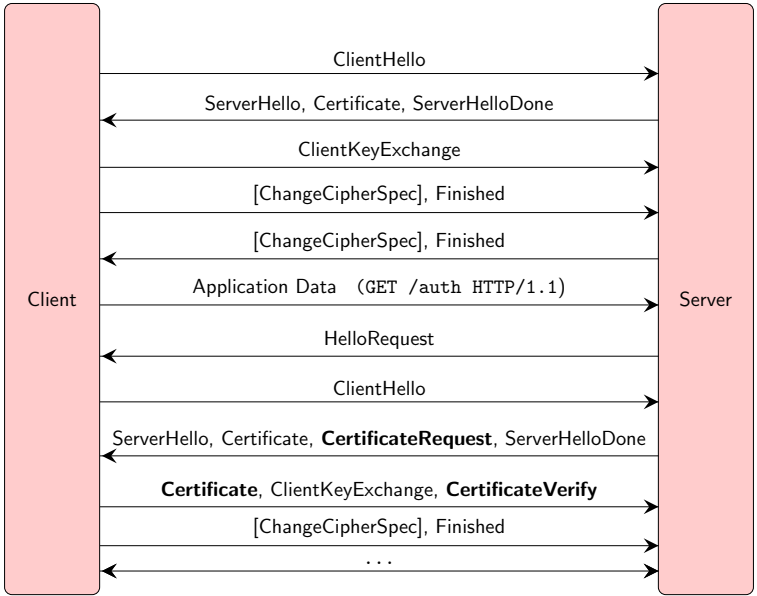


- `CertificateVerify` – client's signature over all handshake messages
- Can `CertificateVerify` be reused in another handshake?
- Why is `CertificateVerify` after `ClientKeyExchange`?
- Client's Certificate is sent before `ChangeCipherSpec`
- Client proves its identity by signing and not by decrypting
- Solves most of the problems of password authentication

Renegotiation

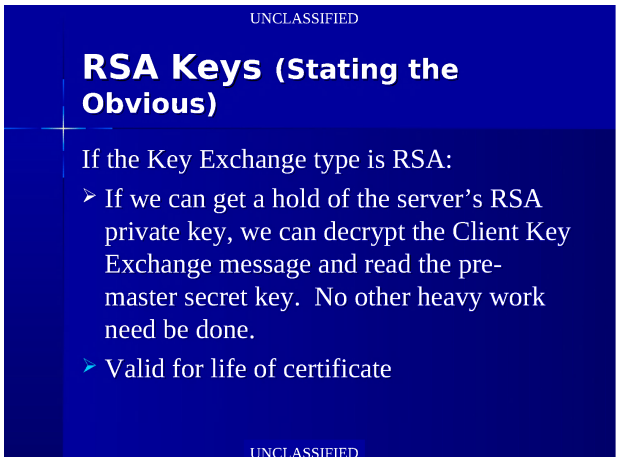
- Any party can initiate negotiation of a new TLS session:
  - Client by sending `ClientHello`
  - Server by sending `HelloRequest`
- Handshake messages of the new TLS session are protected by the cipher suite negotiated in the previous TLS session
- Used by the server to renegotiate a stronger cipher suite or to request a client certificate authentication if on the application level a client tries to access some resource that requires such a security measure
- Client-initiated renegotiation usually disabled on the server

Certificate request on renegotiation



TLS decryption

https://www.eff.org/document/20141228-spiegel-scarletfever-program-attack-secure-sockets-layer-ssl-tls



Can we prevent it?

Perfect Forward Secrecy (PFS)

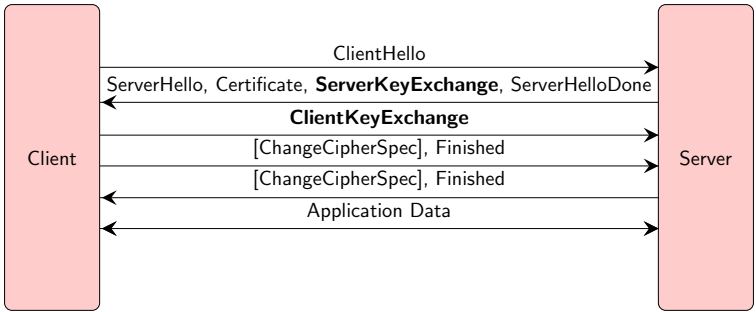
PFS is achieved by using the server's long-term private key to authenticate a short-term/ephemeral asymmetric key that is used to encrypt the actual data.

Benefits:

- Attacker who has compromised server's private key cannot decrypt network traffic
  - Attacker has to execute active MITM attacks
- Attacker has to crack x asymmetric keys to decrypt x sessions made to the server

Used in TLS cipher suites: `TLS_(EC)DHE_RSA_WITH_*`

(EC)Diffie-Hellman Key Exchange



- `ServerKeyExchange` contains DH group, server's DH public key and server's RSA signature over DH public key, client randomness and server randomness
- `ClientKeyExchange` contains client's DH public key
- How is "pre-master secret" calculated?
- Handshake requires two public key operations (DH+RSA)
- Achieves perfect forward secrecy

TLS extensions

- `ClientHello` can contain length-prefixed extensions
- `ServerHello` will contain a response to client's extensions
- Most popular extensions:
  - Server Name Indication (SNI) extension (RFC 3546)

```
▾ Extension: server_name (len=17)
    Type: server_name (0)
    Length: 17
  ▾ Server Name Indication extension
      Server Name list length: 15
      Server Name Type: host_name (0)
      Server Name length: 12
      Server Name: facebook.com
```

  - TLS Session Tickets (RFC 5077)

```
▾ Extension: session_ticket (len=0)
    Type: session_ticket (35)
    Length: 0
    Data (0 bytes)
```

```
▾ Handshake Protocol: New Session Ticket
    Handshake Type: New Session Ticket (4)
    Length: 166
  ▾ TLS Session Ticket
      Session Ticket Lifetime Hint: 7200 seconds (2 hours)
      Session Ticket Length: 160
      Session Ticket: d5a90389e1b88e2731b16af6bdff754466544442ff4a1826…
```

```
▾ Extension: session_ticket (len=160)
    Type: session_ticket (35)
    Length: 160
    Data (160 bytes)
```

TLS extensions

- Certificate Status Request (RFC 6066)

```
▾ Extension: status_request (len=5)
    Type: status_request (5)
    Length: 5
    Certificate Status Type: OCSP (1)
    Responder ID list Length: 0
    Request Extensions Length: 0
▾ TLSv1.2 Record Layer: Handshake Protocol: Certificate Status
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 286
  ▾ Handshake Protocol: Certificate Status
      Handshake Type: Certificate Status (22)
      Length: 282
      Certificate Status Type: OCSP (1)
      OCSP Response Length: 278
    ▾ OCSP Response
        responseStatus: successful (0)
      ▾ responseBytes
          ResponseType Id: 1.3.6.1.5.5.7.48.1.1 (id-pkix-ocsp-basic)
        ▸ BasicOCSPResponse
```

- Supported Elliptic Curves (RFC 4492)

```
▾ Extension: supported_groups (len=10)
    Type: supported_groups (10)
    Length: 10
    Supported Groups List Length: 8
  ▾ Supported Groups (4 groups)
      Supported Group: x25519 (0x001d)
      Supported Group: secp256r1 (0x0017)
      Supported Group: secp384r1 (0x0018)
      Supported Group: secp521r1 (0x0019)
```