



Aalto University

Network Security: TLS 1.3 PSK and session resumption

Tuomas Aura, Aalto University

CS-E4300 Network security

Outline

- Recall TLS 1.3 full handshake
- Pre-shared key (PSK) mode
- Session resumption

TLS 1.3 full handshake

Client

ClientHello

+ key_share*

+ signature_algorithms*

+ supported_groups*

+ server_name*

+ certificate_authorities*

4. Client authentication
(typically omitted)

{Certificate*}
{CertificateVerify*}
{Finished}
[Application data]

Server

ServerHello

+ key_share*

{EncryptedExtensions}
{CertificateRequest*}
{Certificate*}
{CertificateVerify*}
{Finished}
[ApplicationData*]

[Application data]

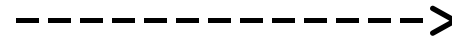
1. Parameter negotiation

2. DHE or ECDHE key exchange

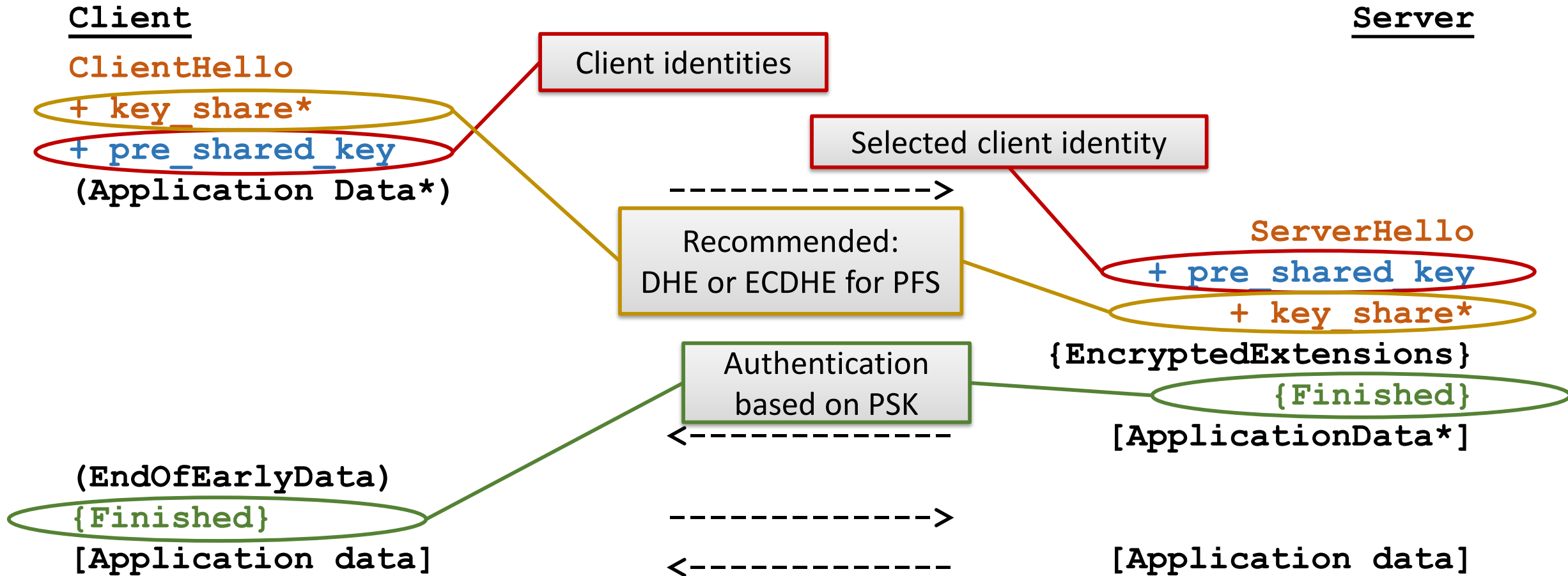
3. Server authentication

5. Key confirmation

6. Protected session data



Pre-shared key (PSK) mode



Pre-shared key (PSK) mode

1. C → S: $N_C, g^x, \text{ClientIdentity}$
2. S → C: $N_S, g^y, \text{HMAC}_{K_{fks}}(\text{TH}),$
early data
3. C → S: $\text{HMAC}_{K_{fkc}}(\text{TH})$

- **Mutual authentication** based on a **pre-established identity and session key (external PSK)**
 - **PSK** = pre-established shared key between C and S
 - HMAC keys for K_{fks} and K_{fkc} Finished message are derived from PSK, g^{xy} and TH, and so at the session keys

TLS 1.3 session resumption

Client

Server

ClientHello

+ key_share*

+ signature_algorithms*

+ supported_groups*

+ server_name*

+ certificate_authorities* ----->

ServerHello

+ key_share*

{EncryptedExtensions}

{CertificateRequest*}

{Certificate*}

{CertificateVerify*}

{Finished}

[ApplicationData*]

Server packages the session state into an encrypted data blob called **session ticket** and sends it to the client

<-----

{Certificate*}

{CertificateVerify*}

{Finished}

----->

<-----

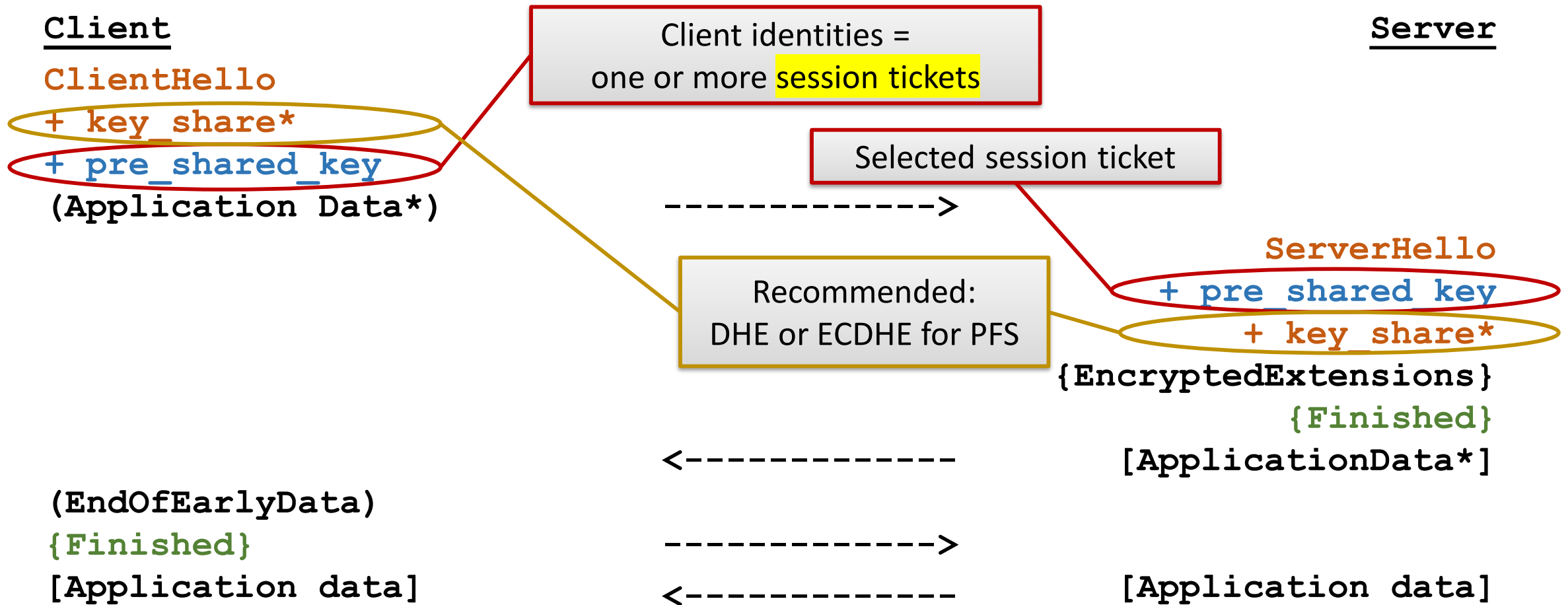
NewSessionTicket

[Application data]

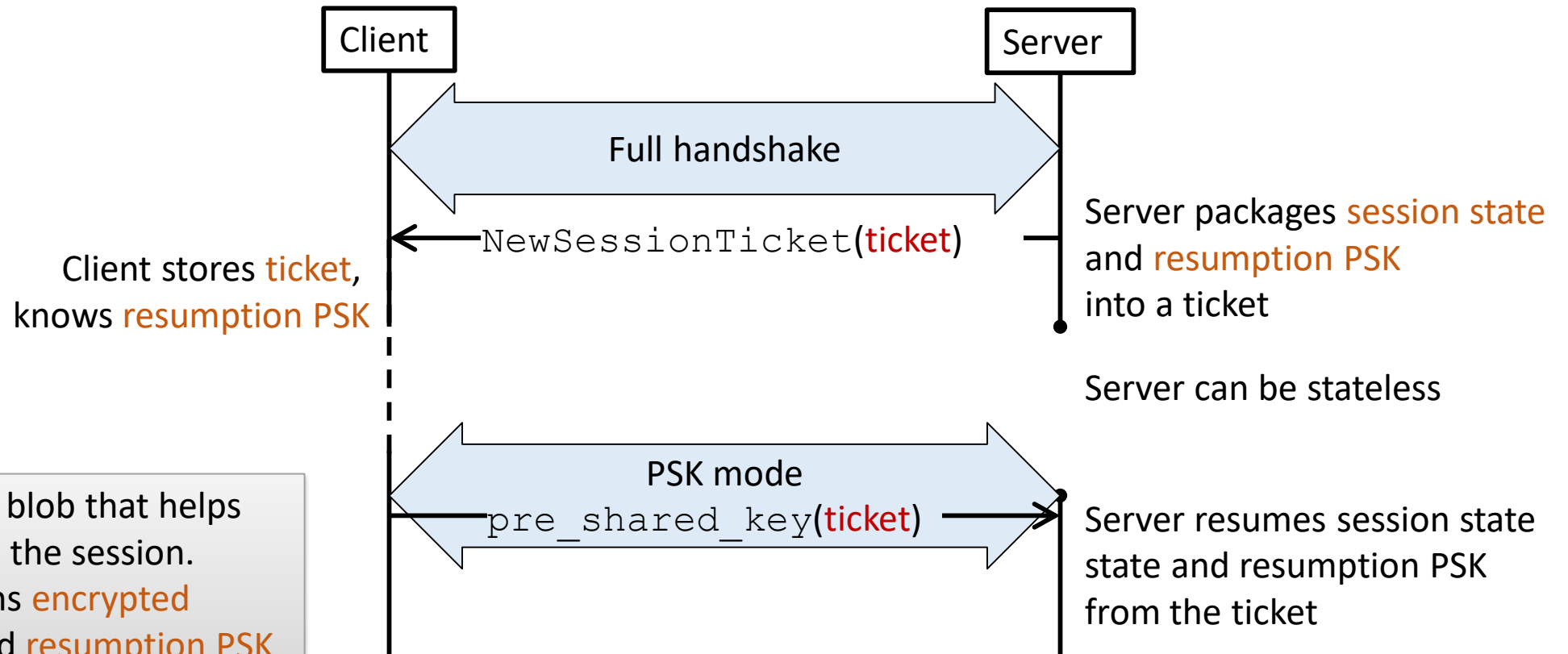
<----->

[Application data]

TLS 1.3 session resumption



TLS 1.3 session resumption



Ticket = opaque blob that helps the server recall the session. Typically contains **encrypted session state** and **resumption PSK**

TLS 1.3 session resumption

- TLS 1.3 session resumption = PSK mode handshake with ticket as client identity and resumption key as the PSK
 - Currently the main purpose of the PSK mode
- When useful?
 - Server does not want to store the TLS sessions over idle periods
 - If client authenticated with smartcard, avoids repeated user action
 - Mobile clients keep changing their IP address and need frequent reconnection
 - Resume session with a different server instance in the cloud

TLS 1.3 session resumption

Client

ClientHello

+ key_share*

+ pre_shared_key

(Application Data*)

----->

Server

ServerHello

+ pre_shared_key

+ key_share*

{EncryptedExtensions}

{Finished}

[ApplicationData*]

<-----

(EndOfEarlyData)

{Finished}

----->

<-----

NewSessionTicket

[Application data]

<-----

[Application data]

Server can refresh the ticket for PFS
and protecting client identity

Key derivation

Inputs to key derivation:

1. PSK (external PSK or resumption PSK)
 2. DHE/ECDHE secret
 3. Transcript of handshake messages, up to the point where the key is derived
- } one or both, as available

Keys:

- client_early_traffic_secret → used to derive AEAD keys for early data in 0-RTT (...)
- client/server_handshake_traffic_secret → used to derive AEAD keys for handshake messages {...} and Finished HMAC keys
- client/server_application_traffic_secret_N → used to derive AEAD encryption keys for post-handshake application data and messages [...]
- resumption_master_secret and ticket_nonce → derive resumption PSK
- exporter_master_secret → used to create keys for the application layer

Identity protection?

- Session tickets are encrypted
- Session ticket can become a pseudo-identifier
 - Server should regularly refresh ticket