A Cryptographic Analysis of the TLS 1.3 Handshake Protocol

Felix Günther
Technische Universität Darmstadt, Germany

joint work with Benjamin Dowling, Marc Fischlin, and Douglas Stebila
TLS1.3: Design, Implementation & Verification

(Provable) Security
Our Analyses
What we prove

Is the TLS 1.3 handshake a good key exchange protocol?

(EC)DHE, PSK(-DHE), 0-RTT (separately)

BR-like computational/game-based multi-stage key exchange model

key indistinguishability
+ auth, strong adversary

different auth modes concurrently

no 0.5-RTT
no post-handshake

don’t cover record protocol (but facilitate modular analysis)
Our Analyses

Timeline

2014
April   draft-00   copy of TLS 1.2
July    draft-02   first changes

2015
March   draft-05   variant based on OPTLS
         draft-dh
         full/(EC)DHE and resumption (≈ PSK) handshake
         [Dowling, Fischlin, Günther, Stebila @ ACM CCS 2015]
October draft-10   integrate OPTLS, add PSK-DHE, 0-RTT, ...
         full/(EC)DHE and PSK/PSK-DHE handshake
         [Dowling, Fischlin, Günther, Stebila @ TRON 2016]

2016
March   draft-12   last version with DH-based 0-RTT
July    draft-14   only PSK-based 0-RTT, restructure key schedule

2017
April 30 draft-20   key schedule changes, ...

DH- and PSK-based 0-RTT handshake
[Fischlin, Günther @ EuroS&P 2017]
TLS 1.3 Full/(EC)DHE Handshake
(simplified)

Client

ClientHello
ClientKeyShare

Server

ServerHello
ServerKeyShare

CertificateRequest*
ServerCertificate*
ServerCertificateVerify*
ServerFinished

ClientCertificate*
ClientCertificateVerify*
ClientFinished

... actually, there is more ...
TLS 1.3 Full/(EC)DHE and PSK(-DHE) Handshake (simplified)

Client

ClientHello
ClientPreSharedKey, CKS*

PSK(-DHE) variant

Server

ServerHello
SKS*, ServerPreSharedKey

CertificateRequest*
ServerCertificate*
ServerCertificateVerify*
ServerFinished

ClientCertificate*
ClientCertificateVerify*
ClientFinished

tk_{app}
TLS 1.3 Full/(EC)DHE and PSK(-DHE) Handshake
 stil simplified

Client

ClientHello
ClientKeyShare

Server

ServerHello
ServerKeyShare

encrypted with $t_k_{hs}$

second part of handshake

resumption master key

for resuming a session

$tk_{hs}$

$tk_{app}$

RMS

EMS

exporter master key

for exporting key material

$tk_{app}$
Multi-Stage Key Exchange (Security)

- **forward secrecy** after long-term reveal
- **eavesdropping**
- **active attacks**
- **key $K_i$ reveal**
- **test $K_i$**
- **$pk_B, sk_A$**
- **$pk_A, sk_B$**
- **key independence in derivation**

Multi-Stage Key Exchange (Security)
Capturing the Compromise of Secrets

Secret Compromise Paradigm

▶ We consider leakage of:
  ▶ **long-term/static secret keys** (signing/pre-shared keys of server/client)
    high potential of compromise, necessary to model forward secrecy
  ▶ **session keys** (traffic keys \( t_{hs} \) and \( t_{app} \), RMS, EMS)
    outputs of handshake used *outside* the key exchange for encryption, resumption, exporting

▶ We do not permit leakage of:
  ▶ **ephemeral secret keys** (DH exponents, signature randomness)
  ▶ **internal values / session state** (master secrets, intermediate values)
    TLS 1.3 handshake not designed to be secure against such compromise
Security of the TLS 1.3 Handshakes

Cryptographic Components

**Client**

- **ClientHello**: $r_c \leftarrow \{0, 1\}_{256}^2$
- **ClientKeyShare**: $X \leftarrow g^x$
- **ClientPreSharedKey**: $\text{psk}_\text{id}$

**Server**

- **ServerHello**: $r_s \leftarrow \{0, 1\}_{256}^2$
- **ServerKeyShare**: $Y \leftarrow g^y$
- **ServerPreSharedKey**: $\text{psk}_\text{id}$

**DHE**

$$DHE \leftarrow XY$$

**$t_{K_{hs}}$**

- Transcript hash signing

**$t_{K_{app}}$**

- Sound key separation

**$H_{trans}$**

$$H_{trans} \leftarrow H(CH||...||SCRT||...$$

**$t_{K_{hs}}$**

- **ServerCertificate**

**$t_{K_{app}}$**

- **ServerFinished**

**$t_{K_{app}}$**

- **CCert**

- **CCertVerify**

**$t_{K_{app}}$**

- **CFin**

**RMS**, **EMS**

We show that the draft-10 full (EC)DHE handshake establishes

- random-looking keys ($tk_{hs}$, $tk_{app}$, RMS, EMS) tolerating adversary that corrupts other users and reveals other session keys
- forward secrecy for all these keys
- concurrent security of anonymous, unilateral, mutual authentication
- key independence (leakage of traffic/resumption/exporter keys in same session does not compromise each other’s security)

assuming

- hash function collision resistance
- signature unforgeability
- HKDF is pseudorandom function
- PRF-ODH assumption holds

\[ \text{standard key exchange security under standard(-model) assumptions} \]

Brendel, Fischlin, Günther, Janson
PRF-ODH: Relations, Instantiations, and Impossibility Results
0-RTT and its Drawbacks

- Data replays (partially unavoidable)
- No forward secrecy

[GHJL@Eurocrypt17]
TLS 1.3 draft-14 PSK(-DHE) 0-RTT

ClientHello: \( nonce_c \)
ClientPreSharedKey: \( psk_{id} \)
ClientKeyShare: \( g^x \) (optional)

\( (pre-)\)shared key PSK

(earlier full handshake)

\( (pre-)\)shared key PSK

\( tk_{ehs} \)

\( tk_{ead} \)

\( tk_{hs} \)

\( tk_{app} \)

\( 0-RTT \) data

\( tk_{ehs} \)

\( tk_{ead} \)

\( tk_{hs} \)

\( tk_{app} \)

\( (ClientFinished_0) \): MAC(\( trans \))

ServerHello: \( nonce_s \)
ServerPreSharedKey: \( psk_{id} \)
(ServerKeyShare: \( g^y \) (optional)

\{Certificate, Signature, MAC\}

\{Certificate, Signature, MAC\}

new: with replays
Multi-Stage Key Exchange (Security) with replays

- Corruption
- Eavesdropping
- Active attacks
- Semi-static reveal
- Key $K_i$ reveal

$p_{kB}, s_{kA}$

$pk_{A}, sk_{B}$

$p_{kB}, s_{kA}$

$p_{kB}, s_{kA}$

$test\ K_i$

$K_1$

$K_2$

$K_1$

$K_2$

“replayable stage/key”
Security of the TLS 1.3 Handshakes
draft-14 PSK(-DHE) 0-RTT Handshake

ClientHello: $\text{nonce}_c$
ClientPreSharedKey: $\text{psk}_\text{id}$
ClientKeyShare: $g^x$ (optional)

(ClientFinished\text{$_0$}): MAC(trans)

- random-looking keys $tk_{\text{ehs}}$, $tk_{\text{ead}}$
  (and all subsequent keys)
- 0-RTT keys & data can be replayed
- no forward secrecy for 0-RTT keys

Assuming:
- hash function collision resistance
- HKDF is pseudorandom function
- HMAC unforgeability (DHE)
- PRF-ODH assumption holds (DHE)
we established security of the keys derived in the **TLS 1.3 handshakes**
what about the **usage of those keys**, e.g., in the Record Protocol, key export?
Composition
Results for TLS 1.3

- we facilitate a modular, compositional approach
- we show: using external, forward-secret keys in any symmetric-key protocol is safe
- supports independent analysis of record protocol
- also captures use of exported EMS and RMS for resumption (cascading)

full (EC)DHE handshake

```
EMS  ➔  generic usage

RMS

PSK handshake

tk_{app}  ➔  Record Protocol

EMS  ➔  generic usage
```

PSK-DHE handshake

```
non-tls

tk_{app}  ➔  Record Protocol

EMS  ➔  generic usage
```
Main Comments on TLS 1.3 from Our Analysis

1. **Separations in key schedule**
   - separate keys for (main) handshake and application data encryption
   - allows to achieve standard key exchange security under standard assumptions
   - enables key independence: neither key affected by other’s compromise
   - thereby facilitating a compositional approach to analyzing the record protocol

2. **Full transcript authentication**
   - full transcript authenticated through signature/MAC
   - makes proof easier and allows for standard assumptions

3. **Encryption of handshake messages**
   - $tk_{hs}$ secure against passive adversaries, hence can indeed increase privacy
   - we confirm there are no negative effects on main key secrecy goal

4. **0-RTT replays and non–forward secrecy**
   - stronger anti-replay mechanisms on key exchange level debatable
   - DH-based 0-RTT had slightly better forward-secrecy properties
We

- analyze TLS 1.3 (drafts 05, dh, 10, 12, 14) full (EC)DHE, PSK(-DHE), and 0-RTT handshakes in a computational multi-stage key exchange model
- establish standard computational key secrecy notions
  - with forward secrecy (for full/PSK-DHE)
  - capturing replayable 0-RTT keys
  - running all authentication modes concurrently
  - under standard assumptions
- provide composition result for modular analysis
- are looking into latest/last TLS 1.3 draft for updated analysis

**full versions @ IACR ePrint**

- [http://ia.cr/2017/082](http://ia.cr/2017/082) (DH/PSK 0-RTT @ draft-12/14)
- [http://ia.cr/2016/081](http://ia.cr/2016/081) (full/PSK @ draft-10)
- [http://ia.cr/2015/914](http://ia.cr/2015/914) (full/PSK @ draft-05/dh)