A Cryptographic Analysis of the TLS 1.3 Handshake Protocol Candidates

Felix Günther

Technische Universität Darmstadt, Germany

joint work with Benjamin Dowling, Marc Fischlin, and Douglas Stebila

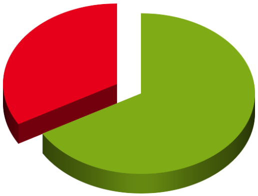
TLS History

… of widespread adoption

TECHNISCHE UNIVERSITÄT DARMSTADT

The [TLS] protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

TLS 1.2 [RFC 5246]



two-thirds of North American Internet traffic expected to be encrypted in 2016

(Sandvine: Internet Traffic Encryption Trends, 2015)

1995 **SSL 2.0**

1996 **SSL 3.0**

1999 **TLS 1.0**

2006 **TLS 1.1**

2008 **TLS 1.2**

201x **TLS 1.3**

TLS History

… of attacks and analyses

(arbitrary selection from recent years)

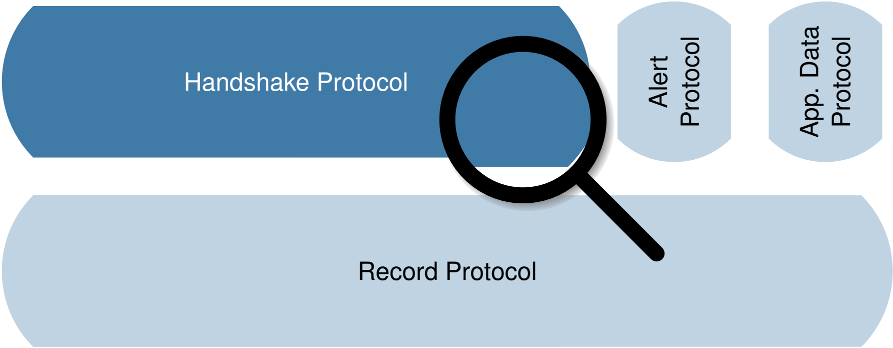
Analyses	Year	Year	Attacks
trunc. handshake [GMP+,MSW]	2008	2008	**TLS 1.2**
		2009	Insecure Renegotiation [RayDis]
record protocol (LHAE) [PRS]	2011	2011	BEAST [DuoRiz]
full TLS-DHE (ACCE) [JKSS]	2012	2012	CRIME [DuoRiz]
verified MITLS impl. [BFK+]	2013	2013	Lucky 13 [AlFPat]
TLS-DH, TLS-RSA-CCA [KSS]			RC4 biases [ABP+]
multiple ciphersuites [KPW]			
TLS 1.2 handshake [BFK+]	2014	2014	Triple Handshake [BDF+]
pre-shared key suites [LSY+]			Heartbleed [Cod]
(de-)constructing TLS [KMO+]			POODLE [MDK]
TLS 1.3 channel [BMM+]	2015	2015	SMACK + FREAK [BBD+]
			Logjam [ABD+]

TECHNISCHE UNIVERSITÄT DARMSTADT

TLS 1.3

- next TLS version, currently being specified (latest: `draft-09`, Oct 2015)
- several substantial cryptographic changes (compared to TLS 1.2)
  1. encrypting some handshake messages with intermediate session key
  2. signing the entire transcript when authenticating
  3. including handshake message hashes in key calculations
  4. generating `Finished` messages with seperate key
  5. deprecating some crypto algorithms (RC4, SHA-1, key transport, MtEE, etc.)
  6. using only AEAD schemes for the record layer encryption
  7. providing reduced-latency 0-RTT handshake
- in large part meant to address previous attacks and design weaknesses
- analysis can prove absence of unexpected cryptographic weaknesses — desirably before standardization

TLS Overview



- we analyze the **handshake protocol** (as of May 2015)

  two candidates: `draft-ietf-tls-tls13-05` and `draft-ietf-tls-tls13-dh-based`

A word of caution...

Limitations

- TLS 1.3 is work in progress
  - analysis limited to draft handshakes (of May 2015)
  - contribution to ongoing discussion
    rather than definitive analysis of TLS 1.3

STANDARD UNDER CONSTRUCTION

- focus on full and resumption handshakes
  - Diffie–Hellman-based full handshake resp. pre-shared key–based resumption
  - don't capture 0-RTT handshake (still un(der)specified at time of writing)

- we don't analyze the Record Protocol
  - but follow a compositional approach that allows independent treatment (see later)

TLS 1.3 Full Handshake (simplified)

`draft-ietf-tls-tls13-dh-based`

(based on OPTLS design by H. Krawczyk and H. Wee, integrated in `draft-09`)

TECHNISCHE UNIVERSITÄT DARMSTADT

Client **Server**

ClientHello
ClientKeyShare →

ServerHello
ServerKeyShare ←

ServerCertificate*
ServerParameters*
ServerFinished ←

ClientCertificate*
ClientCertificateVerify*
ClientFinished →

application data traffic key

tk_{app} ⟷ tk_{app}

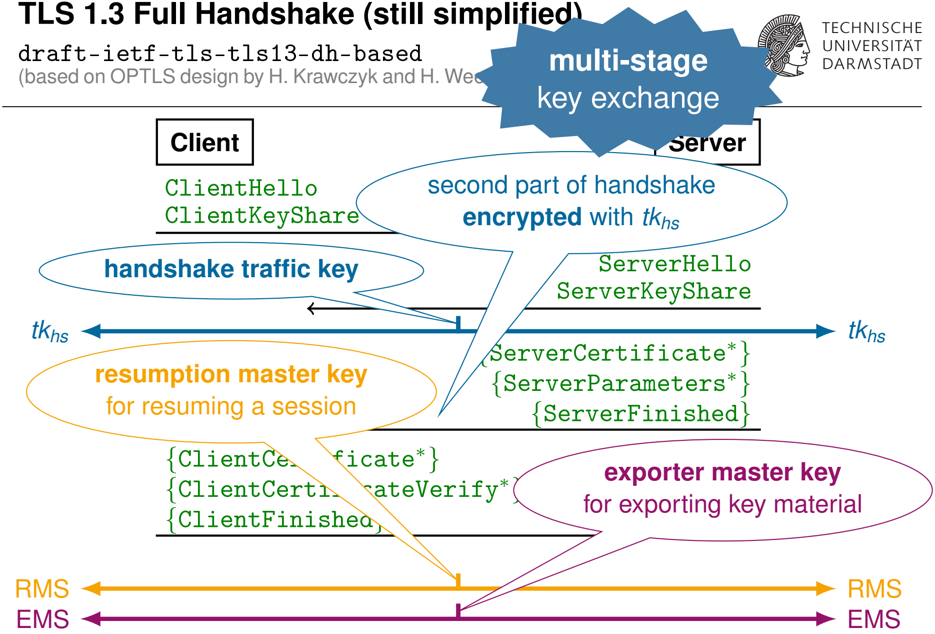
… actually, it's more complicated …

TLS 1.3 Full Handshake (still simplified)

`draft-ietf-tls-tls13-dh-based`

(based on OPTLS design by H. Krawczyk and H. Wee

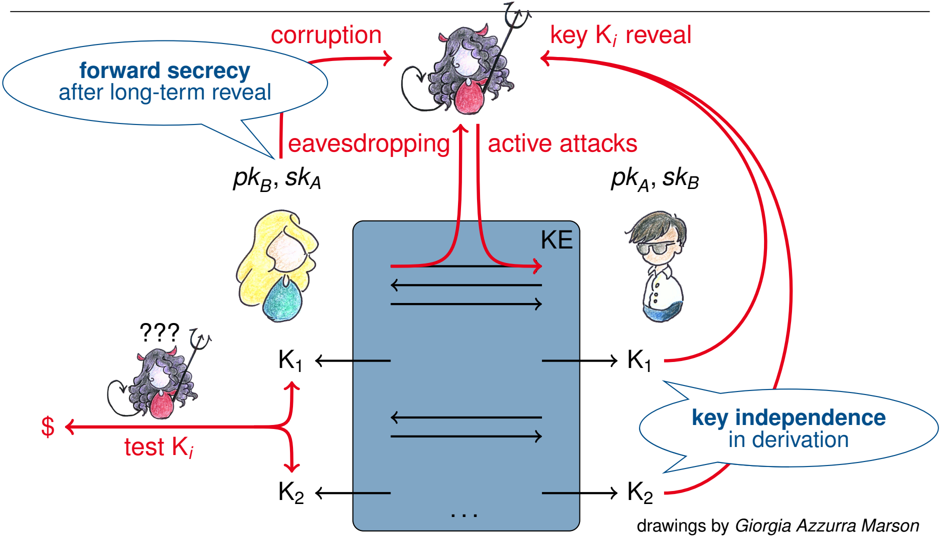
TECHNISCHE UNIVERSITÄT DARMSTADT



Modeling Multi-Stage Key Exchange

(Fischlin, Günther @ CCS 2014)

TECHNISCHE UNIVERSITÄT DARMSTADT



drawings by *Giorgia Azzurra Marson*

Extensions in This Work

- **unauthenticated keys/stages**
  TLS 1.3: neither server nor client send a certificate
- **concurrent execution of different authentication types**
  TLS 1.3: anonymous, server authenticates, server+client authenticate
- **post-specified peers**
  TLS 1.3: parties learn peer's identity (= *pk*) only within handshake
- **preshared-secret key variant**
  TLS 1.3: session resumption is done from preshared secrets (RMS)

Modeling Multi-Stage Key Exchange

Capturing the Compromise of Secrets

TECHNISCHE UNIVERSITÄT DARMSTADT

Secret Compromise Paradigm

- We consider leakage of:
  - **long-term/static secret keys** (signing keys of server/client)
    high potential of compromise, necessary to model forward secrecy
  - **session keys** (traffic keys tk_{hs} and tk_{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 not designed to be secure against such compromise
  - **semi-static secret keys** (s in semi-static g^s used for 0-RTT)
    security of full handshake independent of this value
    but: analysis of **0-RTT handshake** should consider this type of leakage!

Security of the `draft-dh` Full Handshake

(still simplified)

Client | **Server**

ClientHello: $r_C \xleftarrow{\$} \{0,1\}^{128}$
ClientKeyShare: $X \leftarrow g^x$ $\longrightarrow$

ServerHello: $r_S \xleftarrow{\$} \{0,1\}^{128}$
ServerKeyShare: $Y \leftarrow g^y$ $\longleftarrow$

$H_1 \leftarrow \mathrm{H}(\mathtt{CH} \| \ldots \| \mathtt{SKS})$
$\mathrm{ES} \leftarrow X^y$

$\longrightarrow tk_{hs}$

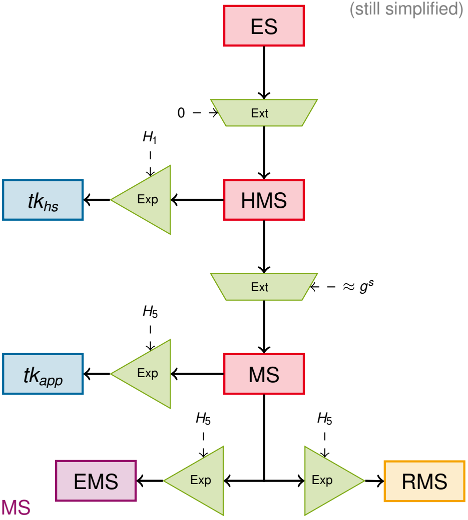
{ServerCertificate*}
$H_2 \leftarrow \mathrm{H}(\mathtt{CH} \| \ldots \| \mathtt{SCRT}^*)$
{ServerParameters*}: $g^s, \mathrm{Sign}_{sk_S}(g^s \| H_2)$
{ServerFinished} $\longleftarrow$

{CCert*},{CCertVerify*},{CFin} $\longrightarrow$

$H_5 \leftarrow \mathrm{H}(\mathtt{CH} \| \ldots \| \mathtt{CCV}^*)$

$\longrightarrow tk_{app}$

$\longrightarrow$ RMS,EMS



Security of the `draft-dh` Full Handshake

TECHNISCHE UNIVERSITÄT DARMSTADT

(still simplified)

Client | **Server**

ClientHello: $r_c \leftarrow_\$ \{0,1\}^{128}$
ClientKeyShare: $X \leftarrow g^x$ →

ServerHello: $r_s \leftarrow_\$ \{0,1\}^{128}$
ServerKeyShare: $Y \leftarrow g^y$ ←

$H_1 \leftarrow \mathrm{H}(\mathtt{CH}\|\ldots\|\mathtt{SKS})$
$\mathrm{ES} \leftarrow X^y$

→ tk_{hs}

{ServerCertificate*}
$H_2 \leftarrow \mathrm{H}(\mathtt{CH}\|\ldots\|\mathtt{SCRT}^*)$
{ServerParameters*}: g^s, $\boxed{\mathrm{Sign}_{sk_S}(g^s\|H_2)}$
{ServerFinished} ←

{CCert*},{CCertVerify*},{CFin} →

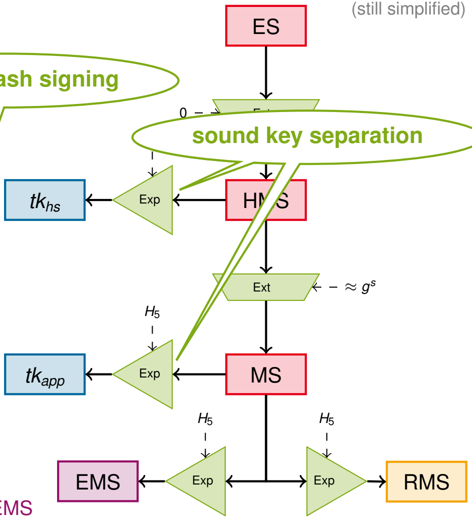
$H_5 \leftarrow \mathrm{H}(\mathtt{CH}\|\ldots\|\mathtt{CCV}^*)$

→ tk_{app}

→ RMS,EMS

session hash signing

sound key separation



Security of the `draft-dh` Full Handshake

TECHNISCHE UNIVERSITÄT DARMSTADT

We show that `draft-dh` full handshake establishes

- random-looking keys (tk_{hs}, tk_{app}, RMS, EMS) with adversary allowed to corrupt other users and reveal other session keys
- forward secrecy for all these keys
- concurrent security of anonymous, unilateral, mutual authentication
- key independence (leakage of record layer and exporter keys in same session do not compromise each other's security)

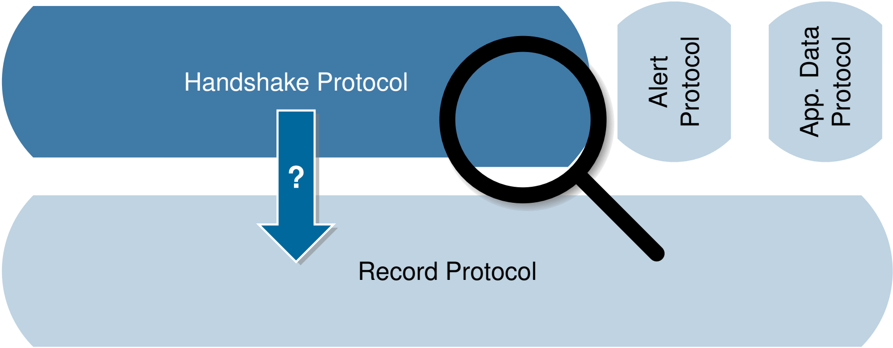
assuming

- collision-resistant hashing
- unforgeable signatures
- Decisional Diffie–Hellman is hard
- HKDF is pseudorandom function

standard assumptions in standard KE model

Composition
of Full Handshake

TECHNISCHE UNIVERSITÄT DARMSTADT

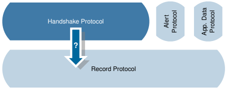


- we established security of the keys derived in the full handshake
- what about the usage of those keys in the Record Protocol?

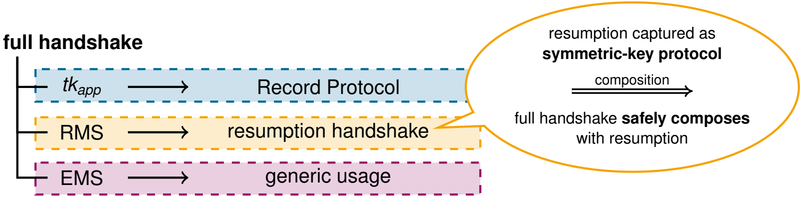
Composition
of Full Handshake

TECHNISCHE UNIVERSITÄT DARMSTADT

- we follow a compositional approach
- extending the previous result [FG'14]



- we show: established keys can safely be used in any symmetric-key protocol
- i.e., Record Protocol can be analyzed independently
- also captures use of RMS for resumption and exported EMS



- open technical question: composition for non-forward-secret key exchange (e.g., resumption handshake)

Comments on TLS 1.3

TECHNISCHE UNIVERSITÄT DARMSTADT

Main Comments on TLS 1.3 from Our Analysis

1. **Soundness of key separation**
   - separate keys for handshake and application data encryption
   - allows to achieve standard key secrecy notions using standard assumptions

2. **Key independence**
   - unique labels in key derivation
   - neither key affected by other's compromise → allows compositional approach

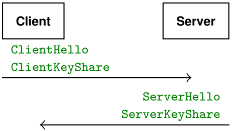
3. **Session hash in online signatures**
   - full transcript signed in `CertificateVerify` messages
   - makes proof easier and allows for standard assumptions

4. **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

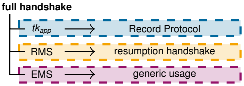
Summary

We

- analyze TLS 1.3 full and resumption handshake candidates (as of May 2015) in an extended multi-stage key exchange model



- show that the full handshakes establish random-looking keys with forward secrecy running all authentication modes concurrently
- achieve standard key secrecy notions under standard assumptions
- extend composition result to allow independent analysis of Record Protocol



No definitive analysis of TLS 1.3, but provides early cryptographic insights.

- we expect that our analysis can be adapted to latest `draft-09`

full version @ IACR ePrint (`http://ia.cr/2015/914`)

Thank You!