A Cryptographic Analysis of the TLS 1.3 Handshake Protocol Candidates

TECHNISCHE UNIVERSITÄT DARMSTADT

Felix Günther

Technische Universität Darmstadt, Germany

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

TECHNISCHE UNIVERSITÄT DARMSTADT

Cryptoplexity
Cryptography & Complexity Theory
Technische Universität Darmstadt
www.cryptoplexity.de

Heisenberg-Programm
Deutsche Forschungsgemeinschaft

Queensland University of Technology

Australian Government
Australian Research Council

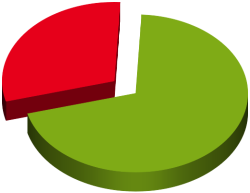
CROSSING

TLS History

. . . of widespread adoption

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]



70% of global Internet traffic expected to be encrypted in 2016

(Sandvine: Internet Traffic Encryption Trends, Feb 2016)

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)

TECHNISCHE UNIVERSITÄT DARMSTADT

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]
			2015	SMACK + FREAK [BBD+]
				Logjam [ABD+]
				PKCS#1 v1.5 vs. TLS 1.3 [JSS]
			2016	SLOTH [BhaLeu]
				DROWN [ASS+]

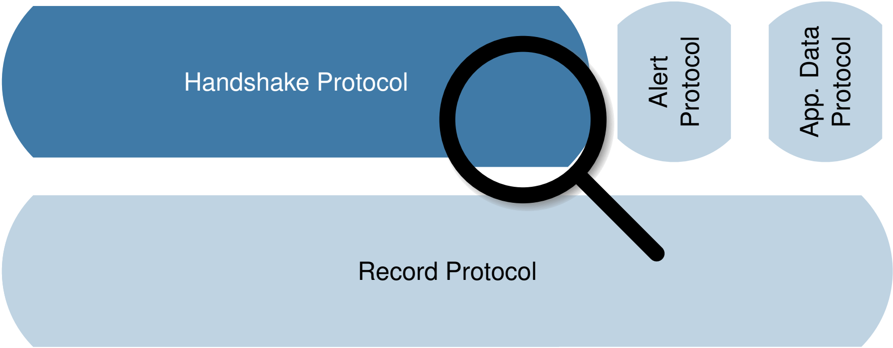
TECHNISCHE UNIVERSITÄT DARMSTADT

TLS 1.3

- next TLS version, currently being specified (latest: `draft-12`, Mar 2016)
- several substantial cryptographic changes (compared to TLS 1.2), incl.
  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. switch to HKDF for key derivation
  8. providing reduced-latency 0-RTT handshake
- in large part meant to address previous attacks and design weaknesses
- analysis can check absence of unexpected cryptographic weaknesses — desirably before standardization

TLS Overview

TECHNISCHE
UNIVERSITÄT
DARMSTADT



- we analyze the **handshake protocol candidates** for TLS 1.3

Our Scope

TECHNISCHE UNIVERSITÄT DARMSTADT

- TLS 1.3 is work in progress
  - analyze `draft-10` (Oct 2015)
  - updating our earlier analysis of `draft-05` and `draft-dh` (of May 2015, @CCS 2015)
  - contribution to ongoing discussion rather than definitive analysis of TLS 1.3

STANDARD UNDER CONSTRUCTION

- focus on full and preshared-key handshakes (separately)
  - Diffie–Hellman-based **(EC)DHE full** handshake
  - **PSK** / **PSK-(EC)DHE** preshared-key/resumption handshake
  - don't capture 0-RTT handshake
- 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-tls13-10`

TECHNISCHE UNIVERSITÄT DARMSTADT

Client — **Server**

Client → Server:
`ClientHello`
`ClientKeyShare`

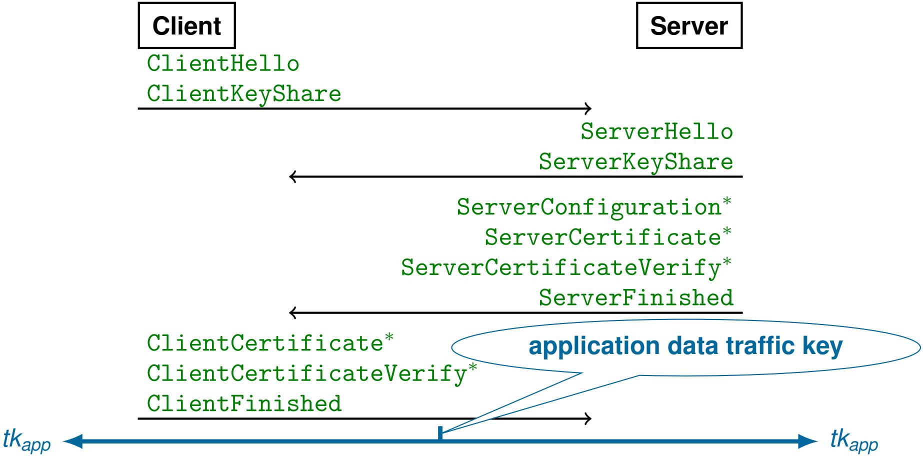
Server → Client:
`ServerHello`
`ServerKeyShare`

Server → Client:
`ServerConfiguration`*
`ServerFinished`

≈ **OPTLS** [KraWee]
cryptographic core

TLS 1.3 Full Handshake (simplified)

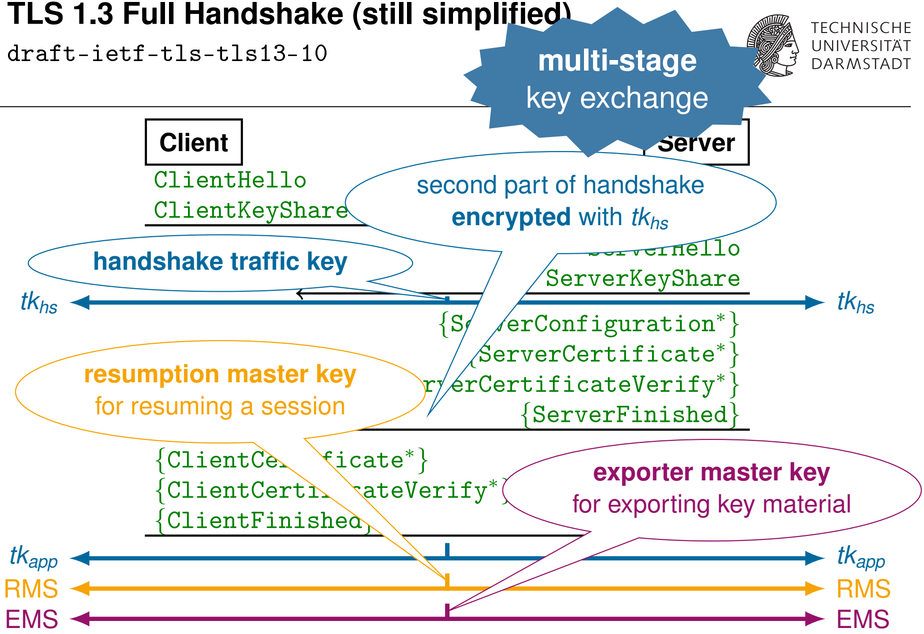
`draft-ietf-tls-tls13-10`



. . . actually, there is more . . .

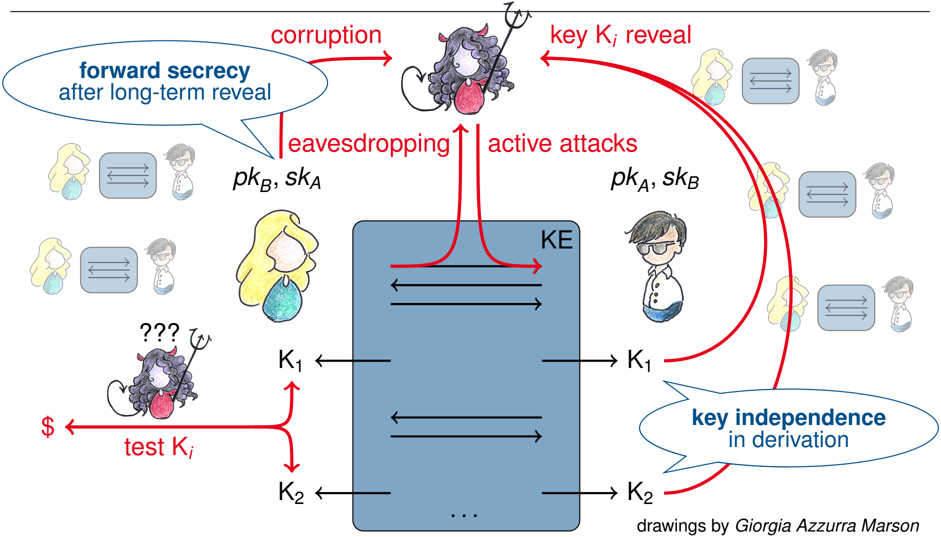
TLS 1.3 Full Handshake (still simplified)

`draft-ietf-tls-tls13-10`



Multi-Stage Key Exchange (Security)

(Fischlin, Günther @ CCS 2014)
game-based model, "provable security" paradigm



drawings by *Giorgia Azzurra Marson*

Modeling Multi-Stage Key Exchange

Further Aspects

TECHNISCHE UNIVERSITÄT DARMSTADT

Extensions in This Work

- **unauthenticated keys/stages** (beyond unilateral/mutual authentication)
  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
- **pre-shared secret key variant**
  TLS 1.3: PSK/PSK-DHE handshake modes from preshared secrets (RMS)

Modeling Multi-Stage Key Exchange

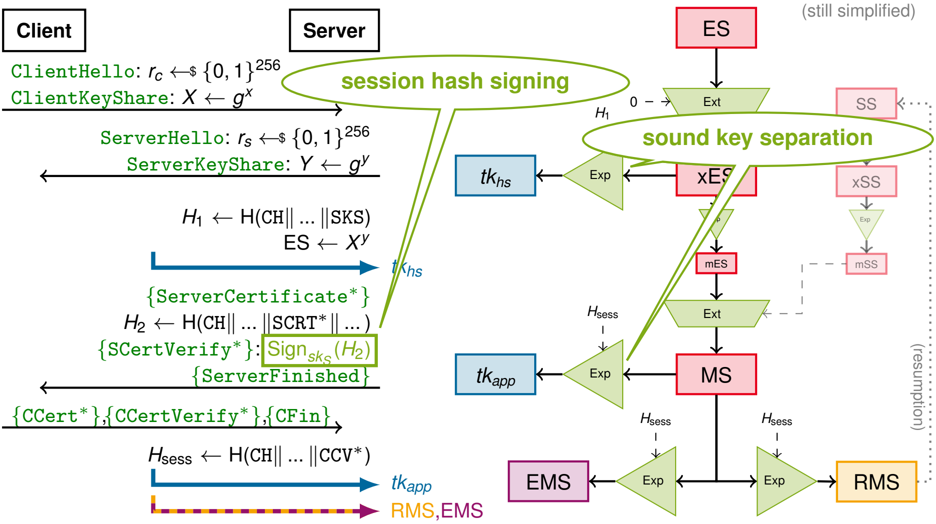
Capturing the Compromise of Secrets

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 full/PSK handshakes 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/PSK handshakes independent of this value
    but: in analysis of **0-RTT handshake** this type of leakage needs to be considered!

Security of the draft-10 Full Handshake



Security of the `draft-10` Full Handshake

TECHNISCHE UNIVERSITÄT DARMSTADT

We show that the `draft-10` full (EC)DHE 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 traffic/resumption/exporter keys in same session does not compromise each other's security)

assuming

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

standard key exchange security under standard assumptions

Security of the `draft-10` PSK Handshakes

TECHNISCHE UNIVERSITÄT DARMSTADT

PSK

- random-looking keys (tk_{hs}, tk_{app}, EMS)
- mutual authentication (down to RMS)
- key independence
- no forward secrecy

PSK-DHE

- random-looking keys (tk_{hs}, tk_{app}, EMS)
- mutual authentication (down to RMS)
- key independence
- forward secrecy for all keys

Under similar standard assumptions:

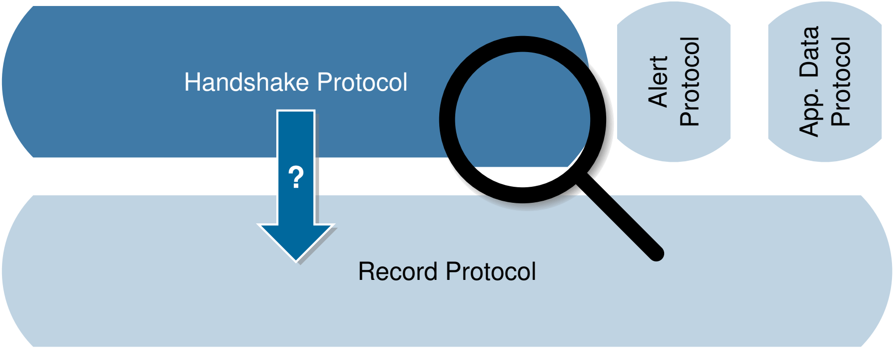
PSK:

- collision-resistant hashing
- HKDF is pseudorandom function

PSK-DHE:

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

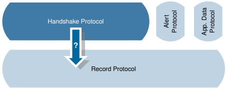
Composition



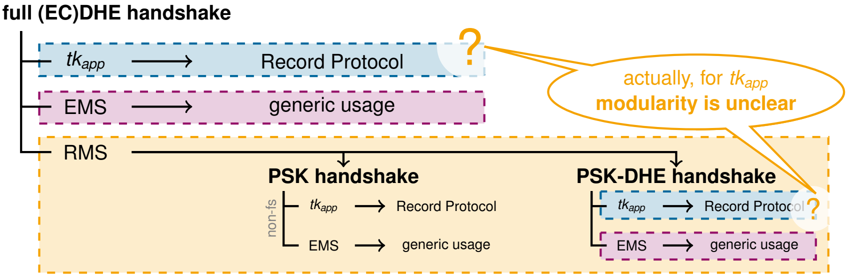
- we established security of the keys derived in the full and PSK handshakes
- what about the usage of those keys, e.g., in the Record Protocol?

Composition

- we follow a modular, compositional approach
  (extending [FG'14], originating from [BFWW'11])



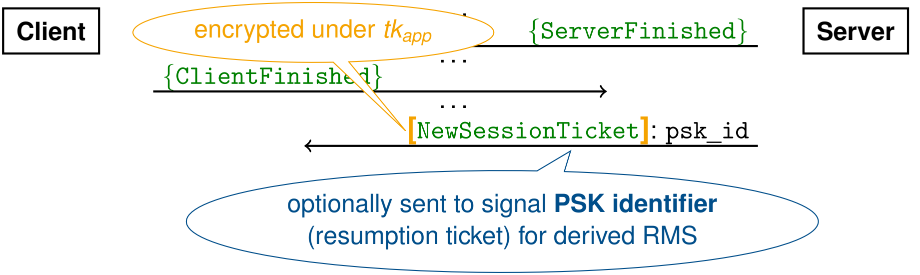
- we show: using final, forward-secret keys in any symmetric-key protocol is safe
- i.e., Record Protocol can be analyzed independently
- also captures use of exported EMS and RMS for resumption (cascading)



Post-Handshake Messages

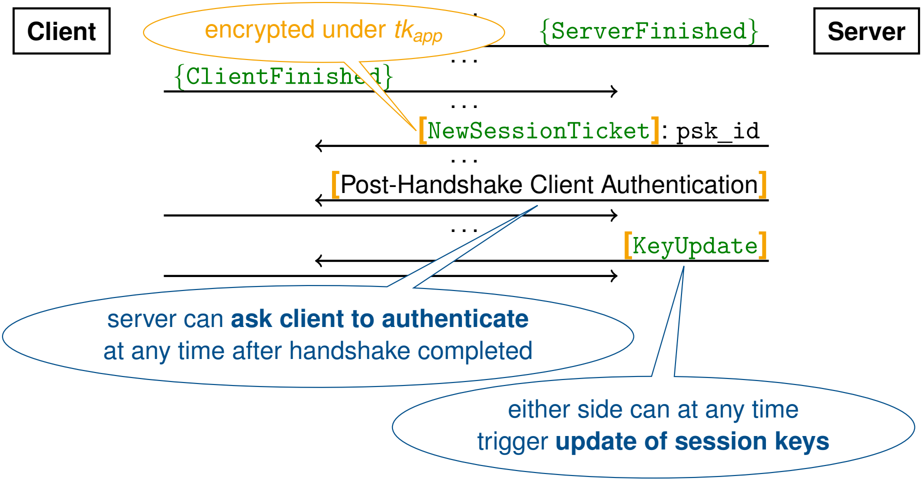
(introduced in `draft-10` and `draft-11`)

TECHNISCHE UNIVERSITÄT DARMSTADT



Post-Handshake Messages

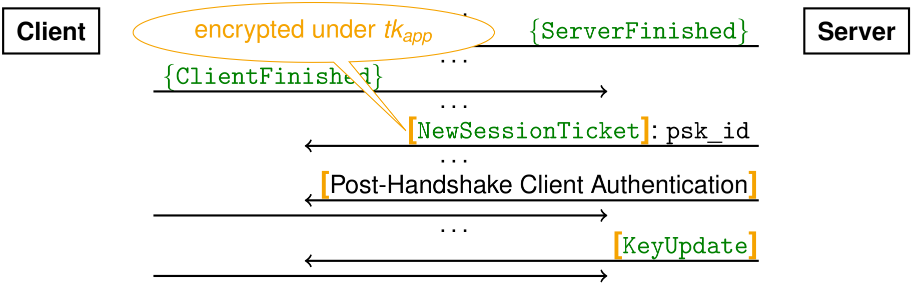
(introduced in `draft-10` and `draft-11`)



Post-Handshake Messages

(introduced in `draft-10` and `draft-11`)

TECHNISCHE UNIVERSITÄT DARMSTADT

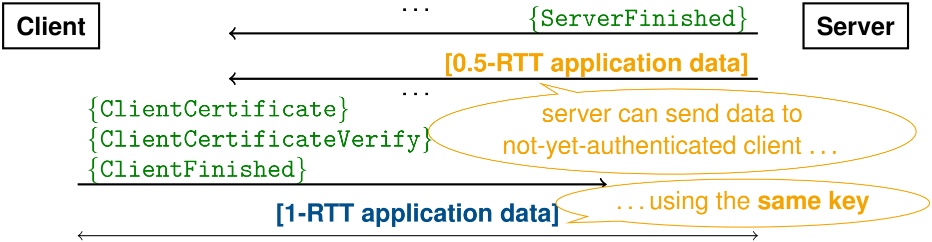


- final/main **session key** tk_{app} **used for (post-)handshake messages**
- reminds of TLS 1.2 `Finished` message (requiring monolithic/special analysis)
- or: should we just understand the initial messages as the TLS 1.3 handshake?

- note: there is no immediate attack arising from this …
- … but means (post-)handshake does not achieve generic KE security
- violates classical modularity between handshake and record layer

0.5-RTT Data with Late Authentication

(introduced in `draft-11`)



- changing authentication of keys **during usage** (first server-only, then mutual)
- beyond what classical key exchange models capture

- take it conservatively?
  - can't provide security guarantees
  - concern: server might send auth-requiring data to unauthenticated client
- take it progressively?
  - computational models need adaption to cover late authentication
  - intuition: just retroactively authenticating the client we're already talking to

Main Comments on TLS 1.3 from Our Analysis

TECHNISCHE UNIVERSITÄT DARMSTADT

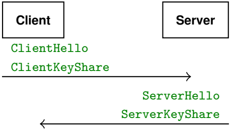
1. **Soundness of key separation**
   - separate keys for handshake and application data encryption*
   - allows to achieve standard key secrecy notions using standard assumptions
2. **Key independence**
   - unique labels in key derivation
   - neither key affected by other's compromise $\rightarrow$ 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
5. **Challenges due to Post-Handshake Messages and 0.5-RTT Data***
   - using application data key for post-handshake violates modularity
   - unclear authentication guarantees when changing key auth during usage

Summary

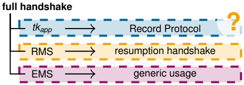
TECHNISCHE UNIVERSITÄT DARMSTADT

We

- analyze TLS 1.3 (`draft-10`)
  full (EC)DHE, PSK, and PSK-DHE handshake
  in an extended multi-stage key exchange model



- establish standard key secrecy notions
  - with forward secrecy (for full/PSK-DHE)
  - running all authentication modes concurrently
  - under standard assumptions



- extend composition result for modular analysis

- point out challenges due to post-handshake messages and 0.5-RTT data

full versions @ IACR ePrint

- `http://ia.cr/2016/081` (draft-10)
- `http://ia.cr/2015/914` (draft-05 + draft-dh)

Thank You!