TLS 1.3

A New Standard and Its Security

TECHNISCHE UNIVERSITÄT DARMSTADT

ECRYPT-NET School on Integrating Advanced Cryptography with Applications

September 16–21, 2018

Felix Günther

Technische Universität Darmstadt, Germany

based on joint work with many others (references within)

special thanks to Marc Fischlin and Kenny Paterson

and thanks to Carlos and Kenny for the invitation to come talk

TECHNISCHE UNIVERSITÄT DARMSTADT

001011100001011 Cryptoplexity

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

Agenda

TECHNISCHE UNIVERSITÄT DARMSTADT

Part I Introducing a New Standard

- The Transport Layer Security (TLS) protocol: history, design, and flaws.
- Why TLS 1.3 and what does it change?

Part II Design & Security Analyses

- TLS 1.3: the technical details
- Understanding the security of TLS 1.3
- Case study: computational security of the TLS 1.3 handshake

- **Goal:** (some) understanding of a complex real-world protocol and its crypto
- Please interrupt and ask if you have questions!

Part I

TLS 1.3
Introducing a New Standard

So What Is TLS?

TLS?

The Transport Layer Security (TLS) Protocol

TECHNISCHE UNIVERSITÄT DARMSTADT

TLS allows client/server applications to communicate over the Internet in a way that is designed to prevent eavesdropping, tampering, and message forgery.

TLS 1.3 [RFC 8446]

1994	**SSL 1.0** (unpublished)	all considered seriously broken today
1995	**SSL 2.0**	
1996	**SSL 3.0**	
1999	**TLS 1.0** – RFC 2246	$\approx$ SSL 3.0, adopted by IETF
2006	**TLS 1.1** – RFC 4346	maintained by IETF TLS working group
2008	**TLS 1.2** – RFC 5246	
2018	**TLS 1.3** – RFC 8446	

So What Is TLS?

TECHNISCHE UNIVERSITÄT DARMSTADT

An IETF standard

TLS?

The TLS Protocol

A Story of Success . . . and Failures

TECHNISCHE UNIVERSITÄT DARMSTADT

- initially introduced by Netscape to enable e-commerce on the WWW
- today: protecting billions of Internet connections every day
  - web, email, messaging, VoIP, banking, payments, e-health, . . .
  - $>80\%$ of web traffic is encrypted[1]
- an exposed target for attacks with a track record of critical flaws
  - structural/design errors
  - weaknesses in cryptographic primitives
  - implementation flaws
  - . . .
- crypto and security research important to analyze and understand security
  - finding design flaws, guiding design, discussing security trade-offs

[1] e.g., `https://www.f5.com/labs/articles/threat-intelligence/the-2017-tls-telemetry-report`

So What Is TLS?

The TLS Protocol
High-level Goals

TECHNISCHE UNIVERSITÄT DARMSTADT

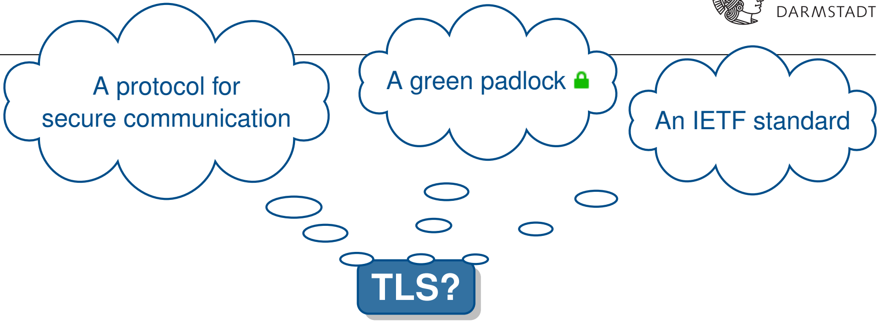
(from TLS 1.3, RFC 8446)

- “The primary goal of TLS is to provide a **secure channel between two peers**”
- only requirement from underlying transport: reliable, in-order data stream

- **Authentication**
  - **server** side of the channel is **always authenticated**
  - **client** side is **optionally authenticated**
  - via **asymmetric cryptography** (signatures) or a symmetric **pre-shared key**
- **Confidentiality**
  - **data** sent over the channel is **only visible to the endpoints**
  - TLS does **not hide the length** of the data it transmits (but allows padding)
- **Integrity**
  - **data** sent over the channel **cannot be modified by attackers** without detection

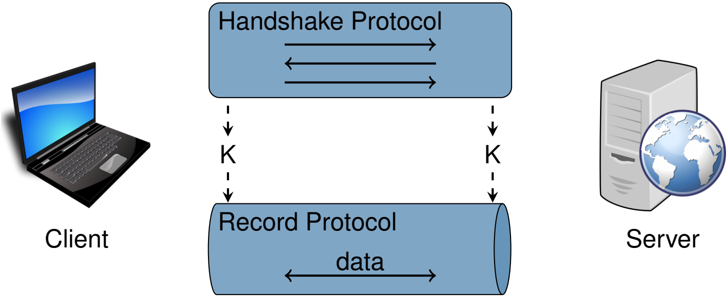
- security in the face of an **attacker who has complete control of the network**

So What Is TLS?



The TLS Protocol
Overly Simplified

Handshake Protocol:
- negotiate security parameters (“cipher suite”)
- authenticate peers
- establish key material for data protection



Record Protocol:
- protect data using key material from handshake
- ensuring confidentiality and integrity

So What Is TLS?

A protocol for secure communication

A green padlock

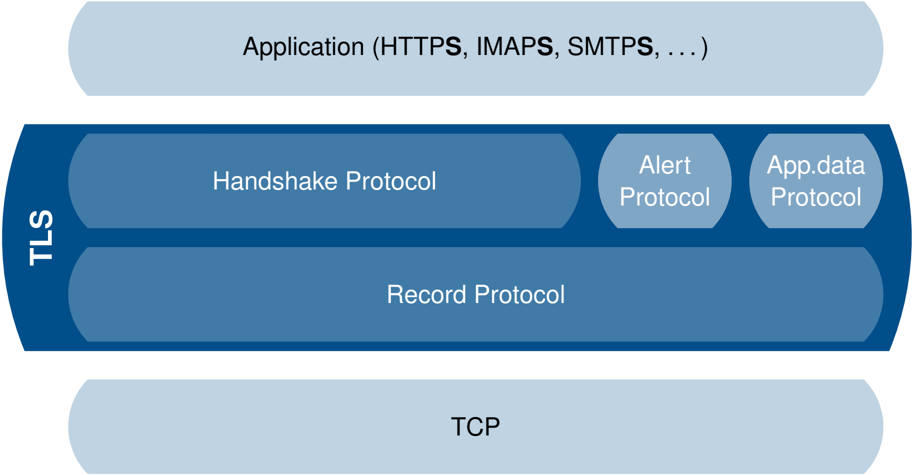
An IETF standard

Key exchange + channel
(i.e., long understood. . . (?))

TLS?

The TLS Protocol
Architecture within Network Stack

TECHNISCHE UNIVERSITÄT DARMSTADT



So What Is TLS?

TECHNISCHE UNIVERSITÄT DARMSTADT

A protocol for secure communication

A green padlock

An IETF standard

Key exchange + channel
(i.e., long understood. . . (?))

TLS?

A layer-4 protocol

The TLS Protocol

Actors

- with billions of users come **billions of devices** (for servers and clients)
- of all types, from *laptop* $\leftrightarrow$ *cloud* to *embedded device* $\leftrightarrow$ *smart home hub*

- running **various implementations** of TLS, in software and hardware
- from widely-used libraries (OpenSSL, those of Google, Facebook, ...) to small or even ad-hoc implementations

- authentication through **Certification Authorities** (100+ in standard browser)
- highly trusted and single-point-of-failure

So What Is TLS?

TECHNISCHE UNIVERSITÄT DARMSTADT

A protocol for secure communication

A green padlock

An IETF standard

Key exchange + channel
(i.e., long understood. . . (?))

TLS?

A layer-4 protocol

The Internet security backbone

The TLS Protocol

Components

- TLS is a “**self-negotiating**” protocol
- handshake first of all agrees on TLS version and cipher suite to use

- **Cipher suites:** client proposes list, server picks
- fixes crypto algorithms to be used for that session
- format (up to TLS 1.2): `TLS_KEX_AUT_WITH_CIP_MAC`

Key Exchange

`RSA` `DHE` `ECDHE` `PSK` ...

(H)MAC

`MD5` `SHA` `SHA256` ...

Cipher

`RC4_128` `3DES_EDE_CBC`
`AES_128_CBC` `AES_256_GCM` ...

Authentication

`RSA` `DSS` `ECDSA` `PSK` ...

So What Is TLS?

TECHNISCHE UNIVERSITÄT DARMSTADT

A protocol for secure communication

A green padlock

An IETF standard

Key exchange + channel
(i.e., long understood. . . (?))

TLS?

A layer-4 protocol

The Internet security backbone

A crypto zoo

The TLS Protocol

Handshake Protocol Structure

Client **Server**

ClientHello: TLS_..., TLS_..., ... →

authenticate server

← ServerHello: TLS_KEX_AUT_WITH_CIP_MAC
ServerCertificate*
ServerKeyExchange*
CertificateRequest*

negotiate security parameters

agree on key material

ClientCertificate*
ClientKeyExchange
ClientCertificateVerify*
[ChangeCipherSpec]
{ClientFinished} →

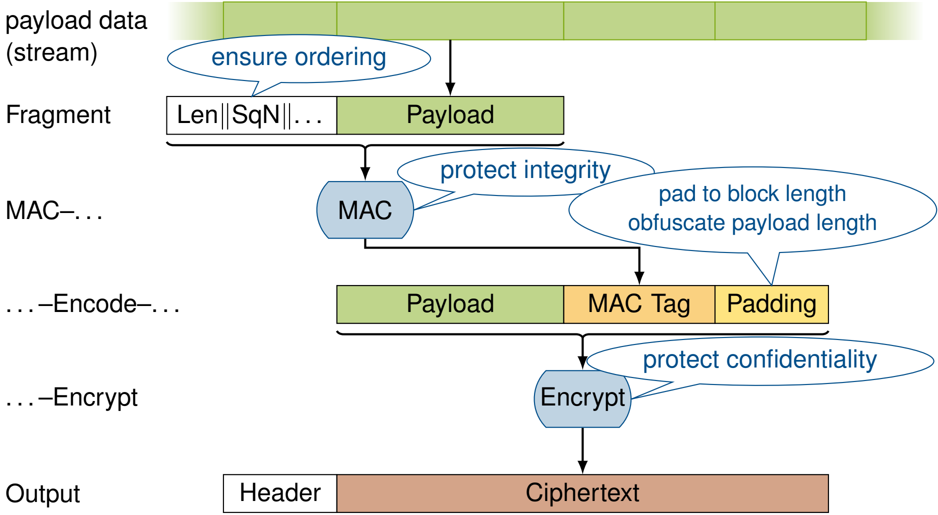
verify transcript agreement

authenticate client

← [ChangeCipherSpec]
{ServerFinished}

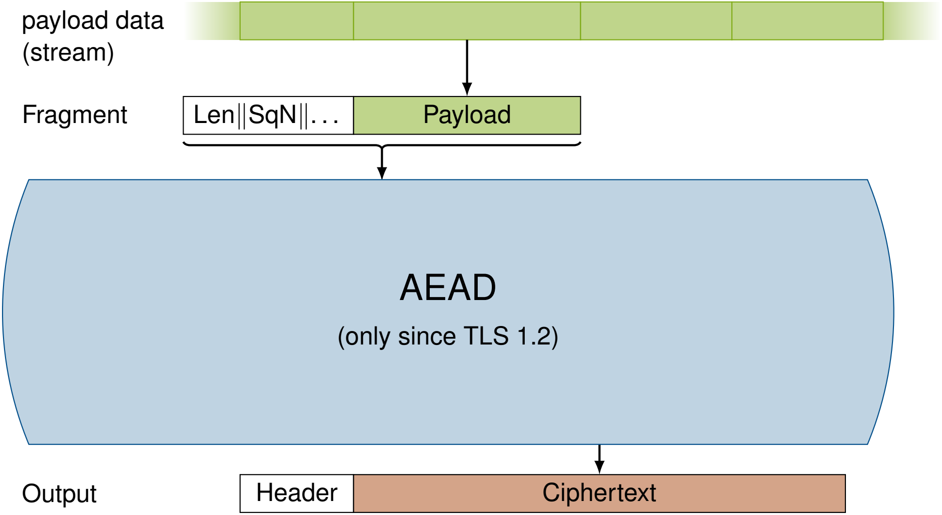
application data, protected with derived key

The TLS Protocol
Record Protocol Structure



The TLS Protocol

Record Protocol Structure



The TLS Protocol

Resumption, Renegotiation, Extensions, ...

TECHNISCHE UNIVERSITÄT DARMSTADT

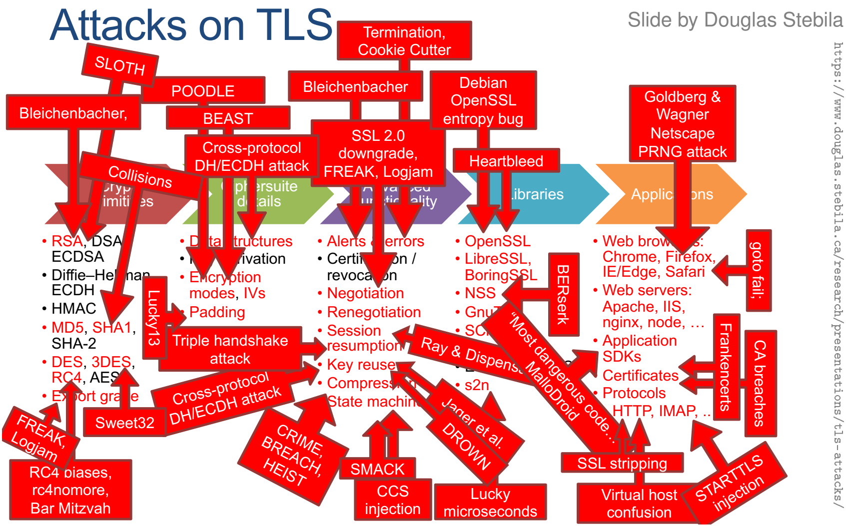
- **(Session) Resumption**
  - abbreviated handshake based on previously established shared secret
  - multiple and possibly parallel connections from same initial secret
- **Renegotiation**
  - change of cipher suite (and keys) within session, protected within Record Protocol
  - use, e.g., for late client authentication (hiding client's identity)
  - or key renewal on long-lived connections without re-establishing connection
- **Extensions & Variants**
  - extensions specify additional functionality and/or security features
  - e.g.: AEAD cipher suites, ECC, connections to other protocols, ...
  - some mandatory to implement, some security-critical patches
  - DTLS: variant for TLS over UDP
- **TLS: complex protocol with many subtly interacting sub-components**

"What could possibly go wrong?" :-) (Kenny Paterson)

TLS Security Issues

Well. . .

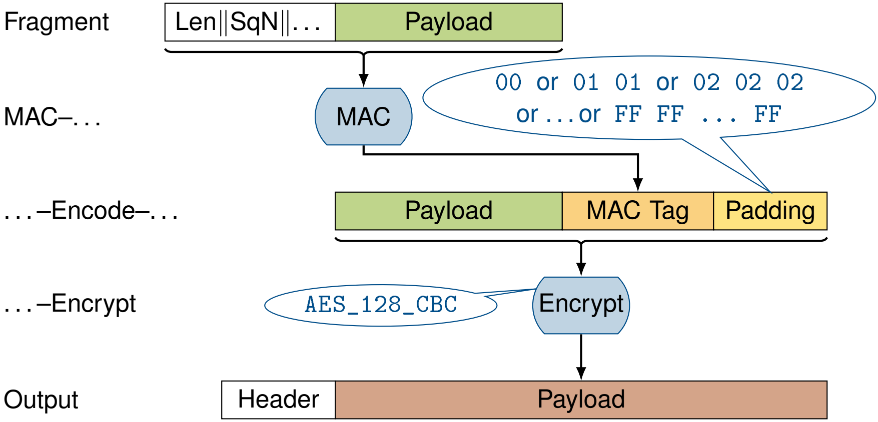
TECHNISCHE UNIVERSITÄT DARMSTADT



TLS Security Issues

@Crypto: MAC-Encode-Encrypt and Lucky13

- core issue: (good) MAC –then– (good) Encrypt ≠ **CCA-secure** AE [BN00]



TLS Security Issues

@Crypto: MAC-Encode-Encrypt and Lucky13

TECHNISCHE UNIVERSITÄT DARMSTADT

- core issue: (good) MAC –then– (good) Encrypt $\neq$ **CCA-secure** AE [BN00]
- **MAC–then–AES-CBC Decryption**
  - decrypt ciphertext to obtain Payload $\|$ MAC Tag $\|$ Padding
  - remove padding — what if padding is incorrect?
  - check MAC
- A padding oracle
  - in a modified ciphertext, either the padding check fails...
  - ... or the MAC check fails
  - if the two are distinguishable: padding oracle
  - can lift a padding oracle to a **decryption oracle** [Vau02] (conditions apply)
- instead of switch to CCA-secure Enc-then-MAC, TLS tried to hide error signal
  - "compute MAC w/ zero padding", "leaves a [non-exploitable] small timing channel"
  - **Lucky13** [AP13]: HMAC timing difference still big enough
  - really need constant time—which is extremely difficult!

TLS Security Issues

@Protocol Design: Weak DH Negotiation and Logjam

TECHNISCHE UNIVERSITÄT DARMSTADT

- core issue: weak algorithms make strong ones fail through **downgrades**

Client | **Server**

ClientHello: [~~G_{2048}~~, G_{512}] →

← Se
ServerCe
ServerKeyExchang

Signature?
– only covers nonces

ClientKeyExchange
{ClientFinished} →

Transcript MAC?
– with weak key

← {ServerFinished}

- **Logjam** [ABD+15]: How Diffie–Hellman Fails in Practice
  - server impersonation through support of (also) weak DH groups

drawings by *Giorgia Azzurra Marson*

TLS Security Issues

@Implementation: Buffers and Heartbleed

TECHNISCHE UNIVERSITÄT DARMSTADT

- core issue: **buffer over-read** in OpenSSL
- **Heartbeat** extension (RFC 6520)
  - client sends “ping back those 4 bytes: 00 01 02 03”
  - server responds “00 01 02 03”
- **Heartbleed** attack [Hea14]
  - client sends “ping back those **16 Kbytes**: 00 01 02 03”
  - server responds “00 01 02 03 ...<memory dump>”
  - possibly including sensitive data like server private key etc.
- high severity & public attention — and a catchy logo

So What Is TLS?

TECHNISCHE UNIVERSITÄT DARMSTADT

A protocol for secure communication

A green padlock

An IETF standard

Key exchange + channel
(i.e., long understood. . . (?))

TLS?

A career opportunity for bit flippers

A layer-4 protocol

The Internet security backbone

A crypto zoo

TLS 1.3

A New Hope?

TECHNISCHE UNIVERSITÄT DARMSTADT

- IETF TLS WG begins in **early 2014** with developing new TLS 1.3 version

So... what would you change?

TLS 1.3
Design Goals

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Clean up:** get rid of flawed and unused crypto & features
- **Improve latency:** for main handshake and repeated connections (while maintaining security)
- **Improve privacy:** encrypt as much of the handshake as possible
- **Continuity:** maintain interoperability with previous versions
  and support existing important use cases
- **Security Assurance (added later):** have supporting analyses for changes

TLS 1.3

Main changes (from TLS 1.2)

TECHNISCHE UNIVERSITÄT DARMSTADT

Clean up

- removed **legacy and broken crypto**
  - ciphers: (3)DES, RC4, . . . , MtEE (CBC & generally) — only AEAD remains
  - hash functions: MD5, SHA1
  - authentication: Kerberos, RSA PKCS#1v1.5 key transport
  - custom (EC)DHE groups
- removed **broken features**
  - compression
  - renegotiation (but added key updates + late client auth)

quite some resistance from enterprises doing passive inspection

- removed **static RSA/DH**: public-key crypto = forward secrecy
- cleaned **key derivation** based on Extract-then-Expand HKDF
- **hardened negotiation** of version/cipher suite against downgrades

TLS 1.3

Main changes (from TLS 1.2)

TECHNISCHE UNIVERSITÄT DARMSTADT

Improve latency

- TLS 1.2 is slow: 2 round trips before client can send data

Client **Server**

`ClientHello`: TLS_..., TLS_..., ...

`ServerHello`: TLS_KEX_AUT_WITH_CIP_MAC

`ServerKeyExchange`*

first round used to learn server capabilities

`ClientKeyExchange`

...

...

application data, protected with derived key

TLS 1.3

Main changes (from TLS 1.2)

TECHNISCHE UNIVERSITÄT DARMSTADT

Improve latency

- TLS 1.2 is slow: 2 round trips before client can send data
- TLS 1.3: **full handshake in 1 round trip**
  - feature reduction $\rightarrow$ we always do (EC)DHE
  - client speculatively sends several DH shares in supported groups
  - server picks one, replies with its share, and key can be already derived
- **0-RTT handshake** when resuming previous connection
  - client+server keep shared resumption secret (PSK)
  - client derives a key from that and can immediately encrypt data
  - <u>but:</u> 0-RTT *sacrifices* certain security properties (will come to that)

TLS 1.3

Main changes (from TLS 1.2)

TECHNISCHE UNIVERSITÄT DARMSTADT

Improve privacy

- TLS 1.2: complete handshake in the clear (incl. certificates, extensions)
- TLS 1.3: **encrypts almost all handshake messages**
  - derive separate key early to protect handshake messages
  - provides security against passive/active attackers (for server/client)

Continuity

- example: complex renegotiation only used for key updates and late client auth
  - just keep these features
- interoperability by having `ClientHello` the only joint message with TLS <1.3
  - Well. . . we'll see.

TLS 1.3

Timeline, Proposals, and Security Analyses

2014	April	`draft-00`	copy of TLS 1.2
	July	`draft-02`	1-RTT, – custom DH, – compression – static RSA/DH, – non-AEAD
	October	`draft-03`	ECC in base standard
2015	January	`draft-04`	remove renegotiation
	March	`draft-05`	
		`draft-dh`	variant based on OPTLS

- ↳ [KW16] OPTLS: unified design for DH/PSK/0-RTT w/ static DH
- ↳ [DF**G**S15] draft-05/dh Analysis: first KE security result

	July	`draft-07`	merging OPTLS (partially): key schedule, HKDF, 0-RTT
	August	`draft-08/9`	deprecate MD5+SHA1, add RSA-PSS signatures

- ↳ [BL16] SLOTH: transcript collision attacks
- ↳ [JSS15] TLS 1.3 vs. PKCS#1v1.5 Encryption: still bad

`https://tools.ietf.org/html/draft-ietf-tls-tls13`

TLS 1.3

Timeline, Proposals, and Security Analyses [cont'd]

TECHNISCHE UNIVERSITÄT DARMSTADT

2015	October	`draft-10`	
	December	`draft-11`	+ downgrade protection, + late client auth, + key updates

↳ [BBF+16] Downgrade Resilience: proposed hardening

↳ [Kra16] Post-Handshake Client Auth: formal treatment

2016	February	**TRON** (TLS 1.3 – Ready or Not?) @ NDSS 2016	

↳ [DF**G**S16] draft-10 Analysis: updated KE security analysis

↳ [BMM+15] Record Protocol Analysis: via constructive crypto

↳ [BBDL+16] miTLS: towards a verified implementation

↳ [CHSvdM16] Tamarin Analysis: symbolic, identified attack

⋮

	May	`draft-13`	restructure key schedule, only PSK-based 0-RTT

↳ [F**G**17] 0-RTT Analysis: PSK- & DH-based, security limitations

https://tools.ietf.org/html/draft-ietf-tls-tls13

TLS 1.3

Timeline, Proposals, and Security Analyses [cont'd]

TECHNISCHE UNIVERSITÄT DARMSTADT

2016	May	"**TRON2**" TLS 1.3 Meetup @ IEEE S&P 2016	
		↳ discussing key schedule, 0-RTT, early implementation results	
	Aug-Oct	`draft-15--17`	lots of discussion around 0-RTT
	October	`draft-18`	
		↳ [BBK17] ProVerif Analysis: tool-based formal analysis	
		↳ [DLFK+17] miTLS: verified Record Protocol implementation	
2017	April	**TLS:DIV** (Design, Implem. & Verif.) @ EuroS&P / Eurocrypt 2017	
		↳ status update & still discussing 0-RTT [Mac17] . . .	
	July	`draft-21`	+ comment on 0-RTT security & recommend mitigations
		↳ [CHH+17] Tamarin Analysis: updated	
	November	`draft-22`	"Implement changes for improved middlebox penetration"
		↳ [Ben18] TLS Ecosystem Woes: Why your Crypto isn't Real World yet	
2018	Feb/Mar	`draft-24--28`	clarifications and cleanup

`https://tools.ietf.org/html/draft-ietf-tls-tls13`

TLS 1.3

The ~~End~~ RFC!

TECHNISCHE UNIVERSITÄT DARMSTADT

2018

August 10 TLS 1.3 = RFC 8446

August 19 **Crypto Welcomes TLS 1.3** @ Crypto 2018

- **already in:** Firefox, Chrome, Cloudflare, Google, Facebook, OpenSSL, . . .
  - ~5% of traffic @ Firefox
  - 2nd-most common version @ Cloudflare
  - ~50% of traffic @ Facebook
- **strong interaction:** TLS WG $\leftrightarrow$ researchers $\leftrightarrow$ engineers
  - high-paced draft progress (29 drafts in 4 years $\approx$ one every 2nd month)
  - proactive rather than reactive standardization process (see [PvdM16])
- **vibrant research topic:** 20+ papers sharpening understanding and tools

TECHNISCHE UNIVERSITÄT DARMSTADT

Part II

TLS 1.3
Design & Security Analyses

TLS 1.3 Security Analyses

TECHNISCHE UNIVERSITÄT DARMSTADT

- recap: TLS 1.3 design process over 4 years
- **many security analyses** along the way
  - of different parts and scopes
  - with varying degree of granularity
  - using different techniques & tools
- would need a school on its own to cover all of these...

TLS 1.3

Focus today

- the **Handshake Protocol** (distinct modes, esp. PSK-(DHE) 0-RTT)
- a **computational analysis** (pen-and-paper provable security)
- will compare & discuss other analyses along the way & in summary

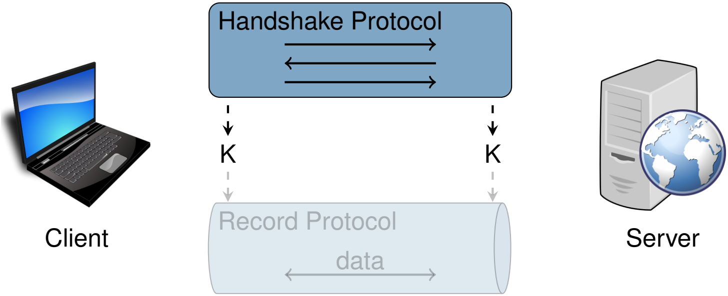
The TLS Protocol

Recap (again overly simplified)

TECHNISCHE UNIVERSITÄT DARMSTADT

Handshake Protocol:

- negotiate security parameters (“cipher suite”)
- authenticate peers
- establish key material for Record Protocol



Record Protocol:

- protect data using key material from handshake
- ensuring confidentiality and integrity

The TLS Protocol

Recap: Handshake Protocol Structure up to TLS 1.2

TECHNISCHE UNIVERSITÄT DARMSTADT

Client	Server
ClientHello	
	ServerHello
	ServerCertificate*
	ServerKeyExchange*
	CertificateRequest*
ClientCertificate*	
ClientKeyExchange	
ClientCertificateVerify*	
[ChangeCipherSpec]	
{ClientFinished}	
	[ChangeCipherSpec]
	{ServerFinished}

application data, protected with derived key

The TLS 1.3 Handshake

Full (EC)DHE Mode

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Client | **Server**

ClientHello
+ClientKeyShare

ServerHello
+ServerKeyShare

EncryptedExtensions*
CertificateRequest*
ServerCertificate
ServerCertificateVerify
ServerFinished

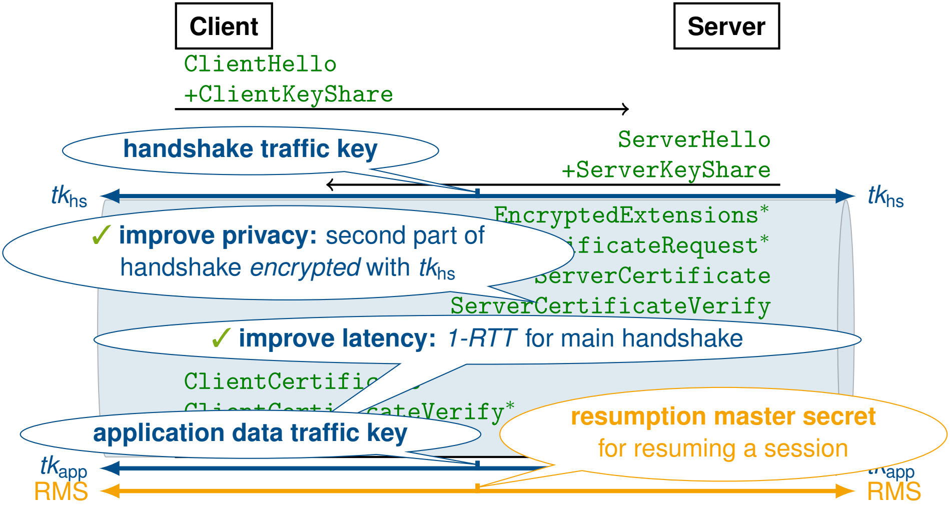
ClientCertificate*
ClientCertificateVerify*
ClientFinished

application data, protected with derived key

The TLS 1.3 Handshake

Full (EC)DHE Mode

TECHNISCHE
UNIVERSITÄT
DARMSTADT



The TLS 1.3 Handshake

PSK / PSK-(EC)DHE Resumption Mode

TECHNISCHE UNIVERSITÄT DARMSTADT

Client **Server**

`ClientHello`
`+ClientKeyShare`*
`+ClientPreSharedKey`

✓ **improve latency:** *0-RTT*
for repeated connection

tk_{0RTT} tk_{0RTT}

`ServerHello`
`+ServerKeyShare`*
`+ServerPreSharedKey`

`EncryptedExtensions`*
~~CertificateRequest*~~
~~ServerCertificate~~
~~ServerCertificateVerify~~
`ServerFinished`

~~ClientCertificate*~~
~~ClientCertificateVerify*~~
`ClientFinished`

The TLS 1.3 Handshake

0.5-RTT and Post-Handshake Messages

TECHNISCHE UNIVERSITÄT DARMSTADT

Additional features (which we won't cover here. . .):

- **0.5-RTT**
  - server can already send data after its `Finished` message
  - client not yet authenticated, but can be done retroactively [Kra16]
- **Post-Handshake Client Authentication**
  - server can ask client to authenticate even after handshake is over
  - captures renegotiation functionality from $\leq$ TLS 1.2
  - again gives retroactive authentication [Kra16]
- **Key Updates**
  - both sides can initiate an update of the traffic key (post-handshake)
  - next key is then derived from master secret in forward-secure manner [**G**M17]

TLS 1.3 Handshake Security

TECHNISCHE UNIVERSITÄT DARMSTADT

- So: What kind of security do we expect for the TLS 1.3 handshake?

- **secure key exchange**
  - derived session keys should be fresh and random
  - keys secret from the point of view of an outside adversary

- here: **provable, game-based, reductionist security**
  - allows us to capture detailed cryptographic computations
  - get precise security bounds & crypto design recommendations

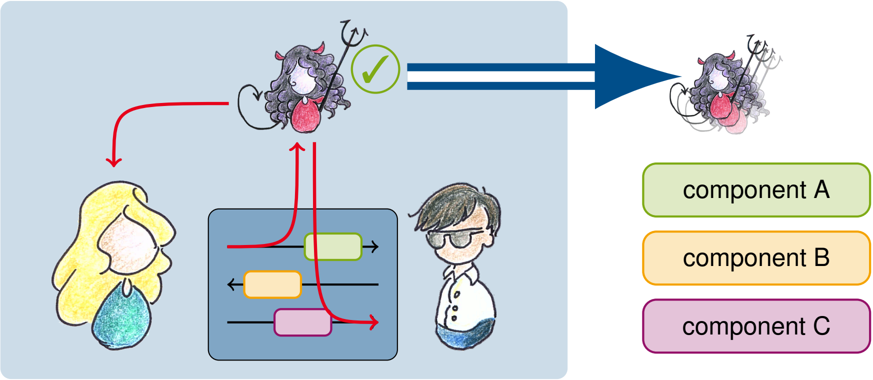
  - due to all the crypto details, security proofs can get complex
  - to handle complexity, we focus on one handshake mode at a time
  - and only look at the “cryptographic core”

  - symbolic analysis tools are better in analyzing interaction across modes
  - though somewhat coarser on the crypto details

  - to be sure the actual code is secure, you need a verified implementation

Cryptographic Security Models and the Provable Security Approach

1. describe abstract protocol
2. define security
3. reduce to assumptions



TLS 1.3 as an Abstract Protocol

TECHNISCHE UNIVERSITÄT DARMSTADT

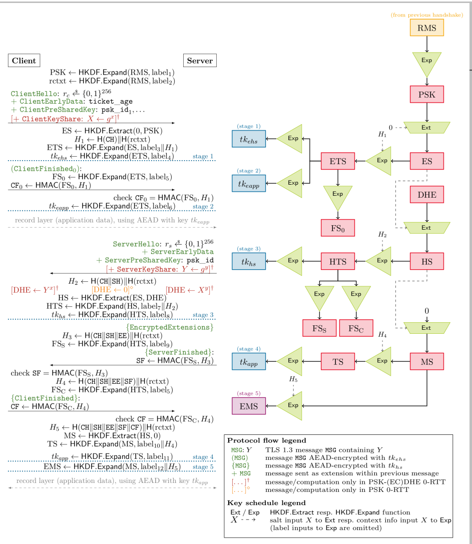
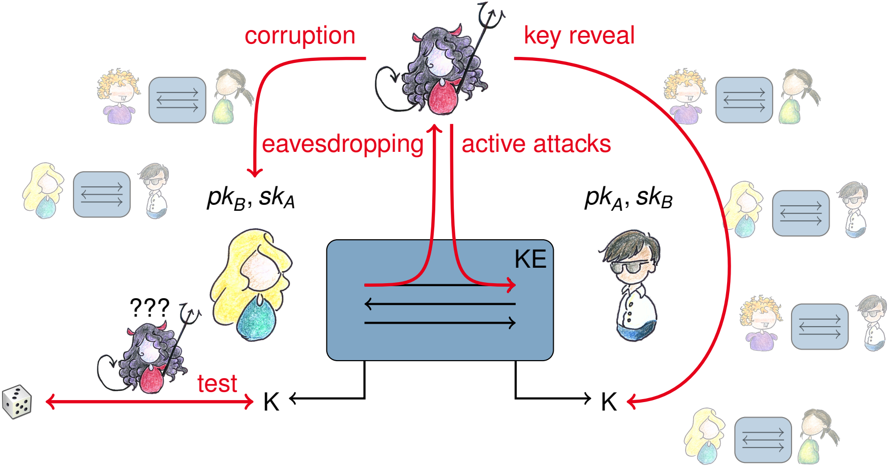


Figure 7.2: The TLS 1.3 draft-14 PSK and PSK-(EC)DHE 0-RTT handshake protocols (left) and key schedule (right).

can be done,
but let's skip that
for now. . .

Key Exchange Security
Classical Definition

- foundational security model by Bellare and Rogaway [BR94]

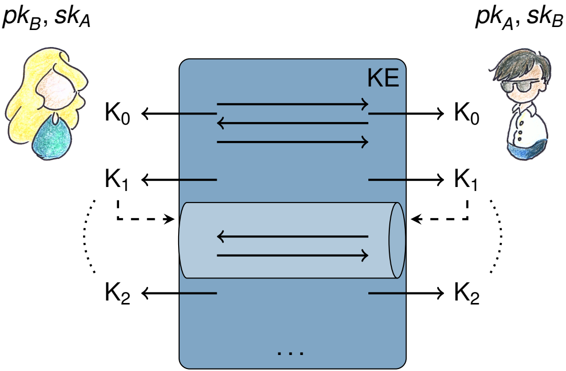


Key Exchange Security

Novel Designs

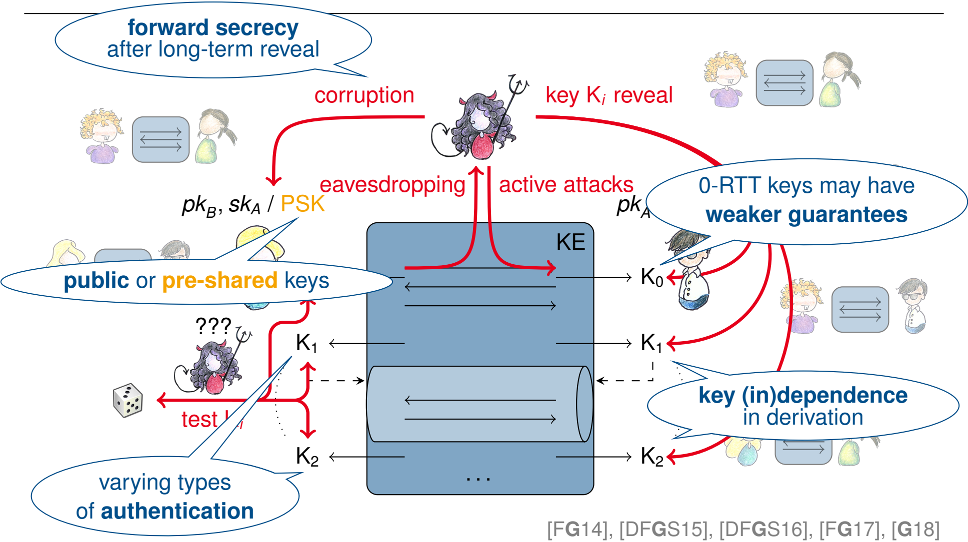
- go beyond what classical models can capture
- e.g., Google QUIC, **TLS 1.3**, Signal, ...

- multiple keys
- potential dependencies
- mixed usage within KE
- low-latency / 0-RTT



Key Exchange Security

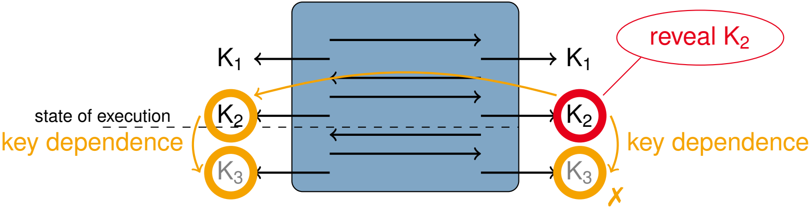
Multi-Stage Key Exchange



[FG14], [DFGS15], [DFGS16], [FG17], [G18]

(In)Dependence of Session Keys

- multi-stage $\Rightarrow$ derived keys might build upon each other
- **key-dependent**: reveal K_i before K_{i+1} accepted *may compromise* K_{i+1}



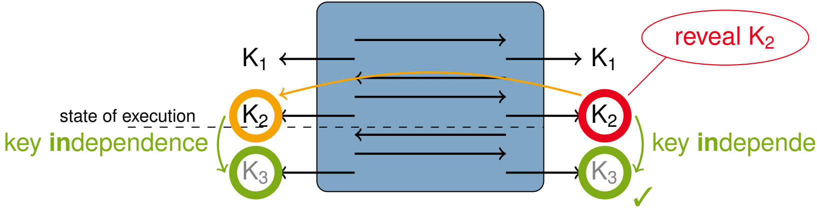
Multi-Stage Key Exchange
Extended Properties

TECHNISCHE UNIVERSITÄT DARMSTADT

(In)Dependence of Session Keys

- multi-stage $\Rightarrow$ derived keys might build upon each other

- **key-dependent**: reveal K_i before K_{i+1} accepted *may compromise* K_{i+1}
- **key-independent**: reveal of any K_i *never harms* any other K_{i+1}



Multi-Stage Key Exchange

Extended Properties

TECHNISCHE UNIVERSITÄT DARMSTADT

Forward Secrecy

- multi-stage $\Rightarrow$ forward secrecy might kick in only at some stage j
- take this into account when handling corruptions

- **non-forward-secret**: all session keys compromised by corruption
- **stage-j-forward-secret**: accepted keys at stages $i \geq j$ remain secure

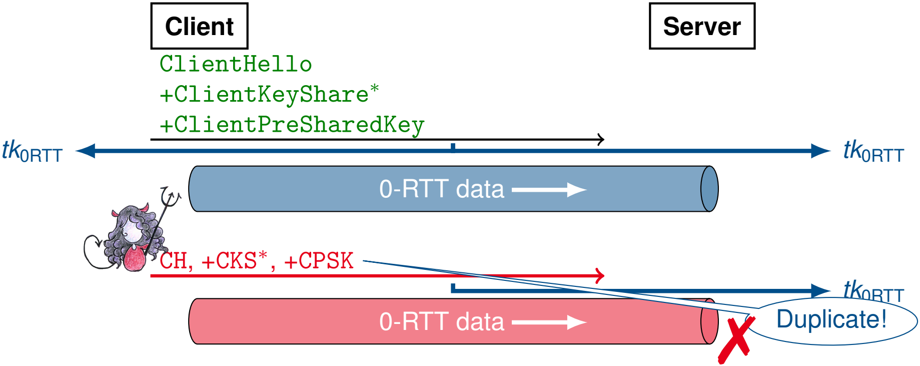
Levels of Authentication

- different stages/keys may hold different authentication properties
  - **unauthenticated** (no-one)
  - **unilateral** authentication (server-only)
  - **mutual** authentication (both)

- different types may run concurrently (TLS: adaptive client authentication)

0-RTT and Replays

TECHNISCHE UNIVERSITÄT DARMSTADT

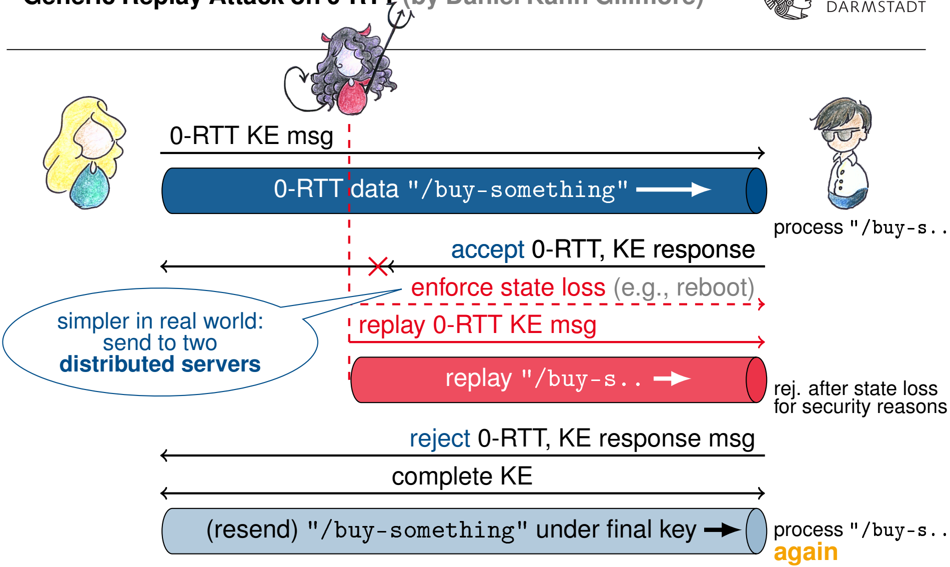


- allows client to send data without waiting for server reply
- but without server input, how does server know the request is fresh?
- adversary can replay `ClientHello` together with 0-RTT data
- idea: remember `ClientHello` identifier and reject duplicates

0-RTT and Replays

Generic Replay Attack on 0-RTT (by Daniel Kahn Gillmore)

TECHNISCHE UNIVERSITÄT DARMSTADT



0-RTT and Replays

TLS 1.3's Take on Replays

TLS does not provide inherent replay protection for 0-RTT data.

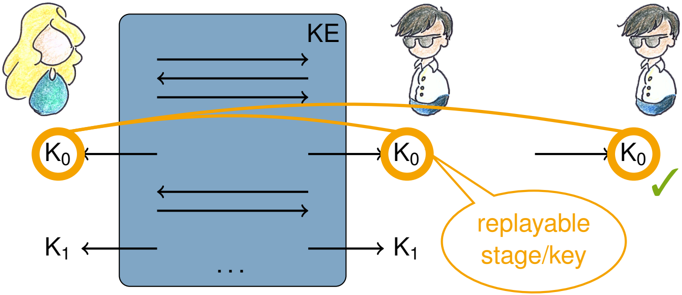
[Simple duplicates] can be prevented by sharing state to guarantee that the 0-RTT data is accepted at most once.

Servers SHOULD provide that level of replay safety by implementing one of the methods described in this section [. . .] [RFC 8446, Section 8]

- **suggested mechanisms**
  - single-use tickets: allow each RMS to be used only once (simplest)
  - `ClientHello` recording: reject by unique identifier
  - freshness checks: reject based on `ClientHello` time
- "SHOULD" $\rightarrow$ treat 0-RTT keys generally as **replayable in analysis**
  - so, what security remains?

Replays

- some stages' keys may be **replayable**
- may be **accepted multiple times**, this shouldn't count as an attack
- but should **still remain secret** from adversary even if replayed

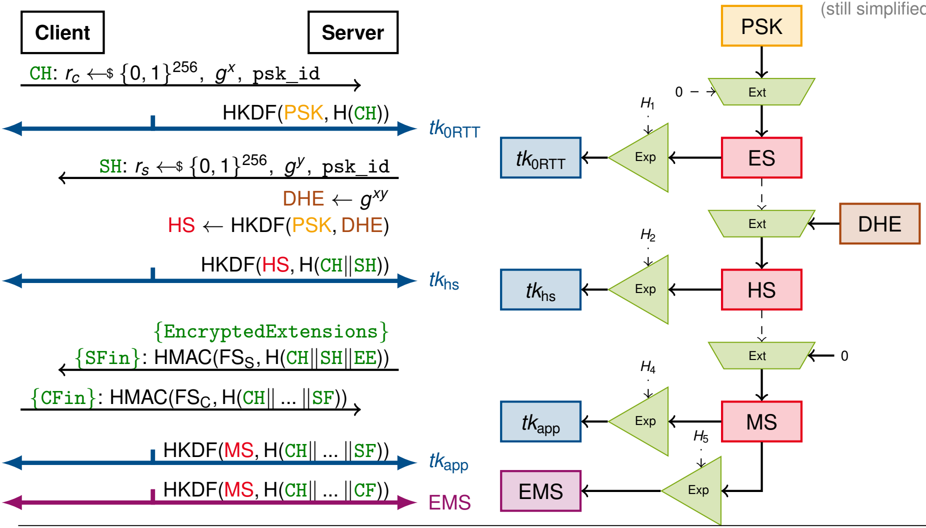


The TLS 1.3 Handshake

draft-14 PSK-(EC)DHE 0-RTT

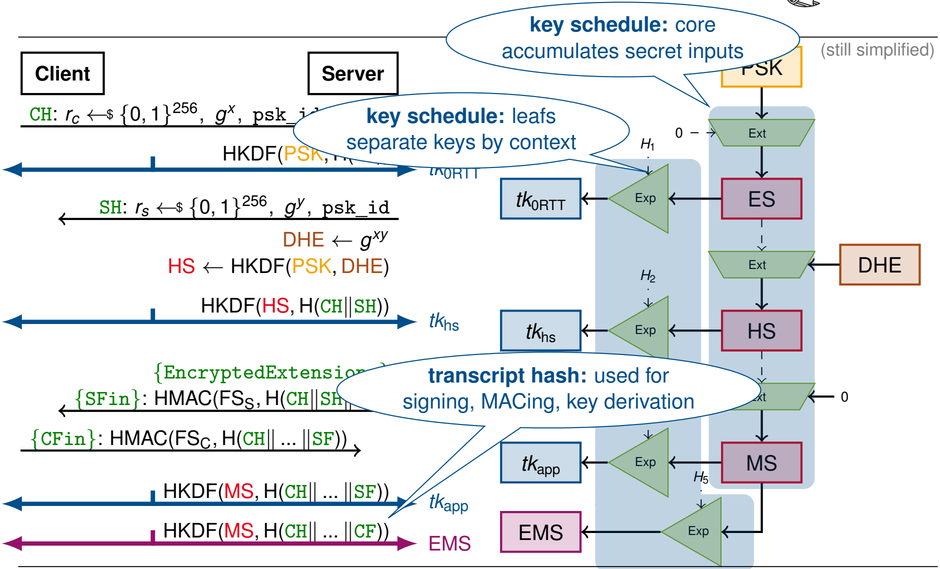
TECHNISCHE
UNIVERSITÄT
DARMSTADT

(still simplified)



The TLS 1.3 Handshake

draft-14 PSK-(EC)DHE 0-RTT



The TLS 1.3 Handshake

draft-14 PSK-(EC)DHE 0-RTT

The full details...

- more intermediate keys (e.g., deriving MAC keys)
- a fifth key tk_{0hs} for 0-RTT handshake encryption (got dropped again later)
- and more...

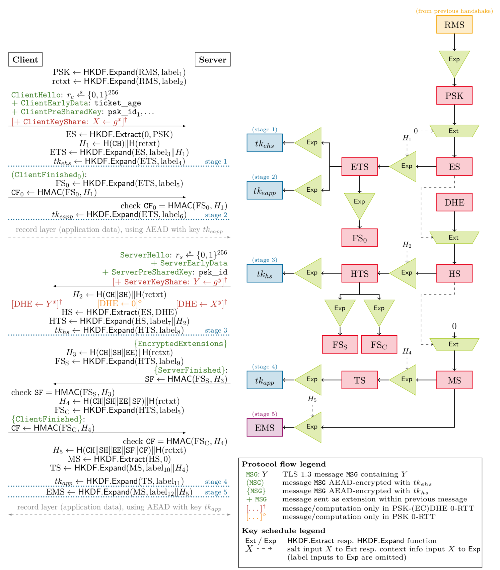


Figure 7.2: The TLS 1.3 draft-14 PSK and PSK-(EC)DHE 0-RTT handshake protocols (left) and key schedule (right).

TLS 1.3 Handshake Security

draft-14 PSK-(EC)DHE 0-RTT as Multi-Stage KE

[FG17]

The **TLS 1.3 PSK-(EC)DHE 0-RTT** handshake provides

- random-looking secret keys (tk_{0hs}, tk_{0RTT}, tk_{hs}, tk_{app}, EMS)
- forward secrecy for non–0-RTT keys
- mutual authentication wrt. PSK
- key independence
- replayable 0-RTT keys

assuming ...

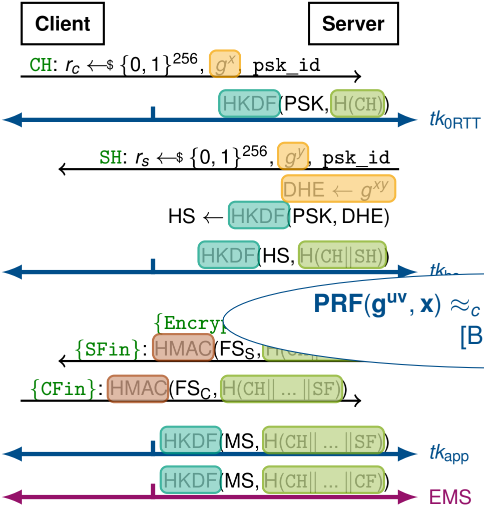
Theorem 7.4. *The TLS 1.3* `draft-14` *PSK-(EC)DHE 0-RTT handshake is* **Multi-Stage***-secure in a* ***key-independent*** *and* ***stage-3-forward-secret*** *manner with properties* (M, **AUTH**, USE, **REPLAY**).

$$\begin{aligned}
\mathsf{Adv}^{\text{Multi-Stage},\mathcal{D}}_{\texttt{draft-14-PSK-(EC)DHE-0RTT},\mathcal{A}} \leq 5n_s \cdot \Big(& \mathsf{Adv}^{\text{COLL}}_{\text{H},\mathcal{B}_1} \\
& + n_p \cdot \Big(\mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_2} + \mathsf{Adv}^{\text{HMAC(0,\$)-\$}}_{\text{HMAC},\mathcal{B}_3} \\
& \qquad + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HMAC},\mathcal{B}_4} + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_5} \Big) \\
& + n_s \cdot n_p \cdot \Big(\mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_6} + \mathsf{Adv}^{\text{HMAC(0,\$)-\$}}_{\text{HMAC},\mathcal{B}_7} \\
& \qquad + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HMAC},\mathcal{B}_8} + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HMAC},\mathcal{B}_9} \\
& \qquad + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{10}} + \mathsf{Adv}^{\text{EUF-CMA}}_{\text{HMAC},\mathcal{B}_{11}} \Big) \\
& + n_s \cdot n_p \cdot \Big(\mathsf{Adv}^{\text{snPRF-ODH}}_{\text{HKDF.Extract},\mathbb{G},\mathcal{B}_{12}} + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HMAC},\mathcal{B}_{13}} \\
& \qquad + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{14}} + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{15}} \\
& \qquad + \mathsf{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{16}} \Big) \Big).
\end{aligned}$$

TLS 1.3 Handshake Security

draft-14 PSK-(EC)DHE 0-RTT as Multi-Stage KE

[FG17]



Theorem 7.4. *The TLS 1.3* draft-14 *PSK-(EC)DHE 0-RTT handshake is* **Multi-Stage**-*secure in a* ***key-independent*** *and* ***stage-3-forward-secret*** *manner with properties* (M, **AUTH**, USE, **REPLAY**).

$$\mathrm{Adv}^{\text{Multi-Stage},\mathcal{D}}_{\texttt{draft-14-PSK-(EC)DHE-0RTT},\mathcal{A}} \leq 5n_s \cdot \Big(\mathrm{Adv}^{\mathrm{COLL}}_{\mathrm{H},\mathcal{B}_1}$$

$$+\, n_p \cdot \Big(\mathrm{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_2} + \mathrm{Adv}^{\text{HMAC(0,\$)-\$}}_{\text{HMAC},\mathcal{B}_3} + \mathrm{Adv}^{\text{PRF-sec}}_{\text{HMAC},\mathcal{B}_4} + \mathrm{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_5} \Big)$$

$$+ \mathrm{Adv}^{\text{HMAC(0,\$)-\$}}_{\text{HMAC},\mathcal{B}_7}$$

$$\text{F-sec}_{\text{HMAC},\mathcal{B}_9}$$

$$+ \mathrm{Adv}\text{-sec}_{\text{HKDF.Expand},\mathcal{B}_{10}} + \mathrm{Adv}^{\text{EUF-CMA}}_{\text{HMAC},\mathcal{B}_{11}}$$

$$+\, n_s \cdot n_p \cdot \Big(\mathrm{Adv}^{\text{snPRF-ODH}}_{\text{HKDF.Extract},\mathbb{G},\mathcal{B}_{12}} + \mathrm{Adv}^{\text{PRF-sec}}_{\text{HMAC},\mathcal{B}_{13}} + \mathrm{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{14}} + \mathrm{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{15}} + \mathrm{Adv}^{\text{PRF-sec}}_{\text{HKDF.Expand},\mathcal{B}_{16}} \Big) \Big).$$

PRF($\mathbf{g^{uv}}$, **x**) $\approx_c$ \$, given oracle **PRF**($\cdot^{\mathbf{u}}$, $\cdot$)

[BFGJ17]

TLS 1.3 Handshake Security

In perspective

TECHNISCHE UNIVERSITÄT DARMSTADT

- **cryptographic design** of TLS 1.3 handshake is **sound**
- strong security results for main keys (both full and PSK handshakes)
- replays and lacking forward secrecy for 0-RTT are a (recognized) downside

- recall: we focused on handshake modes in isolation, for draft-14 (and earlier)

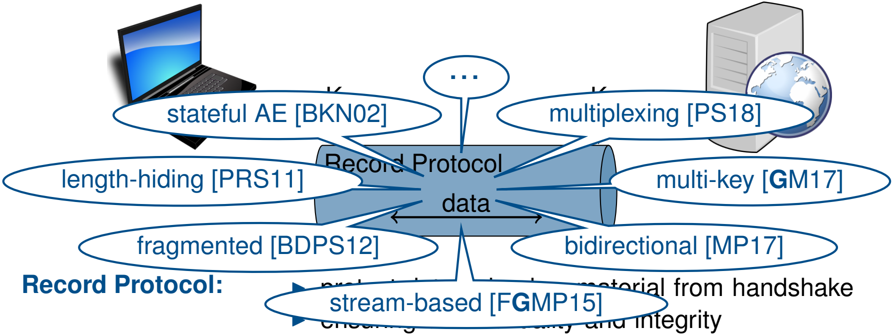
- further analyses (cf. Part I):
  - other computational analyses of sub-parts (e.g., post-handshake client auth)
  - tool-based/symbolic analyses up to full protocol and on multiple drafts
  - work-in-progress verified implementation

- jointly, these analyses give rise to confidence in TLS 1.3 handshake design
- still, doesn't mean there won't be any attacks (bets are on 0-RTT. . .)

TLS 1.3 Security

So… what about the Record Protocol?

- AEAD-based design looks sound…
- but the crypto community hasn't really conclusively ventilated the question: **What is a secure channel protocol?**



Record Protocol:

- … material from handshake
- … and integrity

Conclusions

- **TLS 1.3** = RFC 8446
  - clean up / improve latency / improve privacy / continuity / security assurance
- **proactive standardization:** successful involvement of research community
  - significantly higher confidence from the start than for previous versions
- **0-RTT:** new functionality & new risks

Conclusions

TECHNISCHE UNIVERSITÄT DARMSTADT

- **crypto protocol design** is highly complex
  - even when from “boring crypto” components (that’s a plus!)
  - even when looking only at the “cryptographic core”
- **key exchange and channels**
  - basics considered to be understood
  - but “real-world” challenges demand for more understanding, i.e., research
- **interaction cryptographers $\leftrightarrow$ engineers**
  - necessary to make real-world protocols run securely
  - can be very fruitful for both sides (technical and scientific outcome)
  - cryptographers: go read RFCs, engineers: go read security proofs
    — both can be equally daunting
- **get involved early on**
  - next upcoming: **Messaging Layer Security** Working Group @ IETF [MLS]

Thank You!

Felix Günther
Technische Universität Darmstadt, Germany

mail@**felixguenther.info**

References I

TECHNISCHE
UNIVERSITÄT
DARMSTADT

[ABD+15] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J. Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, Benjamin VanderSloot, Eric Wustrow, Santiago Zanella Béguelin, and Paul Zimmermann.
Imperfect forward secrecy: How Diffie-Hellman fails in practice.
In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15*, pages 5–17. ACM Press, October 2015.

[AP13] Nadhem J. AlFardan and Kenneth G. Paterson.
Lucky thirteen: Breaking the TLS and DTLS record protocols.
In *2013 IEEE Symposium on Security and Privacy*, pages 526–540. IEEE Computer Society Press, May 2013.

[BBDL+16] Benjamin Beurdouche, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cedric Fournet, Samin Ishtiaq, Markulf Kohlweiss, Jonathan Protzenko, Nikhil Swamy, Santiago Zanella-Béguelin, and Jean Karim Zinzindohoué.
Towards a provably secure implementation of TLS 1.3.
Presented at the TRON Workshop at NDSS 2016, 2016.

[BBF+16] Karthikeyan Bhargavan, Christina Brzuska, Cédric Fournet, Matthew Green, Markulf Kohlweiss, and Santiago Zanella Béguelin.
Downgrade resilience in key-exchange protocols.
In *2016 IEEE Symposium on Security and Privacy*, pages 506–525. IEEE Computer Society Press, May 2016.

References II

[BBK17] Karthikeyan Bhargavan, Bruno Blanchet, and Nadim Kobeissi.
Verified models and reference implementations for the TLS 1.3 standard candidate.
In *2017 IEEE Symposium on Security and Privacy*, pages 483–502. IEEE Computer Society Press, May 2017.

[BDPS12] Alexandra Boldyreva, Jean Paul Degabriele, Kenneth G. Paterson, and Martijn Stam.
Security of symmetric encryption in the presence of ciphertext fragmentation.
In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 682–699. Springer, Heidelberg, April 2012.

[Ben18] David Benjamin.
TLS ecosystem woes: Why your crypto isn't real world yet.
Presented at the Real World Crypto Symposium 2018, `https://docs.google.com/presentation/d/1jqyTwZlTPD_xp4rTD4FmbsdKYWRHcUkN5lfMeGQZQ_o/`, 2018.

[BFGJ17] Jacqueline Brendel, Marc Fischlin, Felix Günther, and Christian Janson.
PRF-ODH: Relations, instantiations, and impossibility results.
In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 651–681. Springer, Heidelberg, August 2017.

References III

[BKN02] Mihir Bellare, Tadayoshi Kohno, and Chanathip Namprempre.
Authenticated encryption in SSH: Provably fixing the SSH binary packet protocol.
In Vijayalakshmi Atluri, editor, *ACM CCS 02*, pages 1–11. ACM Press, November 2002.

[BL16] Karthikeyan Bhargavan and Gaëtan Leurent.
Transcript collision attacks: Breaking authentication in TLS, IKE and SSH.
In *NDSS 2016*. The Internet Society, February 2016.

[BMM+15] Christian Badertscher, Christian Matt, Ueli Maurer, Phillip Rogaway, and Björn Tackmann.
Augmented secure channels and the goal of the TLS 1.3 record layer.
In Man Ho Au and Atsuko Miyaji, editors, *ProvSec 2015*, volume 9451 of *LNCS*, pages 85–104. Springer, Heidelberg, November 2015.

[BN00] Mihir Bellare and Chanathip Namprempre.
Authenticated encryption: Relations among notions and analysis of the generic composition paradigm.
In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer, Heidelberg, December 2000.

[BR94] Mihir Bellare and Phillip Rogaway.
Entity authentication and key distribution.
In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 232–249. Springer, Heidelberg, August 1994.

References IV

TECHNISCHE UNIVERSITÄT DARMSTADT

[CHH+17] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe.
A comprehensive symbolic analysis of TLS 1.3.
In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 17*, pages 1773–1788. ACM Press, October / November 2017.

[CHSvdM16] Cas Cremers, Marko Horvat, Sam Scott, and Thyla van der Merwe.
Automated analysis and verification of TLS 1.3: 0-RTT, resumption and delayed authentication.
In *2016 IEEE Symposium on Security and Privacy*, pages 470–485. IEEE Computer Society Press, May 2016.

[DFGS15] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila.
A cryptographic analysis of the TLS 1.3 handshake protocol candidates.
In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15*, pages 1197–1210. ACM Press, October 2015.

[DFGS16] Benjamin Dowling, Marc Fischlin, Felix Günther, and Douglas Stebila.
A cryptographic analysis of the TLS 1.3 draft-10 full and pre-shared key handshake protocol.
Cryptology ePrint Archive, Report 2016/081, 2016.
`http://eprint.iacr.org/2016/081`.

References V

TECHNISCHE
UNIVERSITÄT
DARMSTADT

[DLFK+17] Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Jonathan Protzenko, Aseem Rastogi, Nikhil Swamy, Santiago Zanella Béguelin, Karthikeyan Bhargavan, Jianyang Pan, and Jean Karim Zinzindohoue.
Implementing and proving the TLS 1.3 record layer.
In *2017 IEEE Symposium on Security and Privacy*, pages 463–482. IEEE Computer Society Press, May 2017.

[FG14] Marc Fischlin and Felix Günther.
Multi-stage key exchange and the case of Google's QUIC protocol.
In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 14*, pages 1193–1204. ACM Press, November 2014.

[FG17] Marc Fischlin and Felix Günther.
Replay attacks on zero round-trip time: The case of the TLS 1.3 handshake candidates.
In *2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017*, pages 60–75, Paris, France, April 26–28, 2017. IEEE.

[FGMP15] Marc Fischlin, Felix Günther, Giorgia Azzurra Marson, and Kenneth G. Paterson.
Data is a stream: Security of stream-based channels.
In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 545–564. Springer, Heidelberg, August 2015.

References VI

TECHNISCHE UNIVERSITÄT DARMSTADT

[GM17] Felix Günther and Sogol Mazaheri.
A formal treatment of multi-key channels.
In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 587–618. Springer, Heidelberg, August 2017.

[Gün18] Felix Günther.
Modeling Advanced Security Aspects of Key Exchange and Secure Channel Protocols.
PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, 2018.
`http://tuprints.ulb.tu-darmstadt.de/7162/`.

[Hea14] Heartbleed bug.
`http://heartbleed.com/`, 2014.

[JSS15] Tibor Jager, Jörg Schwenk, and Juraj Somorovsky.
On the security of TLS 1.3 and QUIC against weaknesses in PKCS#1 v1.5 encryption.
In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15*, pages 1185–1196. ACM Press, October 2015.

[Kra16] Hugo Krawczyk.
A unilateral-to-mutual authentication compiler for key exchange (with applications to client authentication in TLS 1.3).
In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 16*, pages 1438–1450. ACM Press, October 2016.

References VII

TECHNISCHE UNIVERSITÄT DARMSTADT

[KW16] Hugo Krawczyk and Hoeteck Wee.
The OPTLS protocol and TLS 1.3.
In *2016 IEEE European Symposium on Security and Privacy, EuroS&P 2016*, pages 81–96, Saarbrücken, Germany, March 21–24, 2016. IEEE.

[Mac17] Colm MacCárthaigh.
Security review of TLS 1.3 0-RTT.
`https://github.com/tlswg/tls13-spec/issues/1001`, 2017.

[MLS] Messaging layer security ietf wg.
`https://datatracker.ietf.org/wg/mls/about/`.

[MP17] Giorgia Azzurra Marson and Bertram Poettering.
Security notions for bidirectional channels.
IACR Trans. Symm. Cryptol., 2017(1):405–426, 2017.

[PRS11] Kenneth G. Paterson, Thomas Ristenpart, and Thomas Shrimpton.
Tag size does matter: Attacks and proofs for the TLS record protocol.
In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 372–389. Springer, Heidelberg, December 2011.

References VIII

TECHNISCHE UNIVERSITÄT DARMSTADT

[PS18] Christopher Patton and Thomas Shrimpton.
Partially specified channels: The TLS 1.3 record layer without elision.
Cryptology ePrint Archive, Report 2018/634, 2018.
`https://eprint.iacr.org/2018/634`.

[PvdM16] Kenneth G. Paterson and Thyla van der Merwe.
Reactive and proactive standardisation of TLS.
In Lidong Chen, David A. McGrew, and Chris J. Mitchell, editors, *Security Standardisation Research: Third International Conference (SSR 2016)*, volume 10074 of *Lecture Notes in Computer Science*, pages 160–186, Gaithersburg, MD, USA, December 5–6, 2016. Springer.

[Vau02] Serge Vaudenay.
Security flaws induced by CBC padding - applications to SSL, IPSEC, WTLS...
In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 534–546. Springer, Heidelberg, April / May 2002.