Establishing Secure Connections

A Cryptographer's Perspective and the Case of TLS 1.3

TECHNISCHE UNIVERSITÄT DARMSTADT

Felix Günther
Technische Universität Darmstadt, Germany

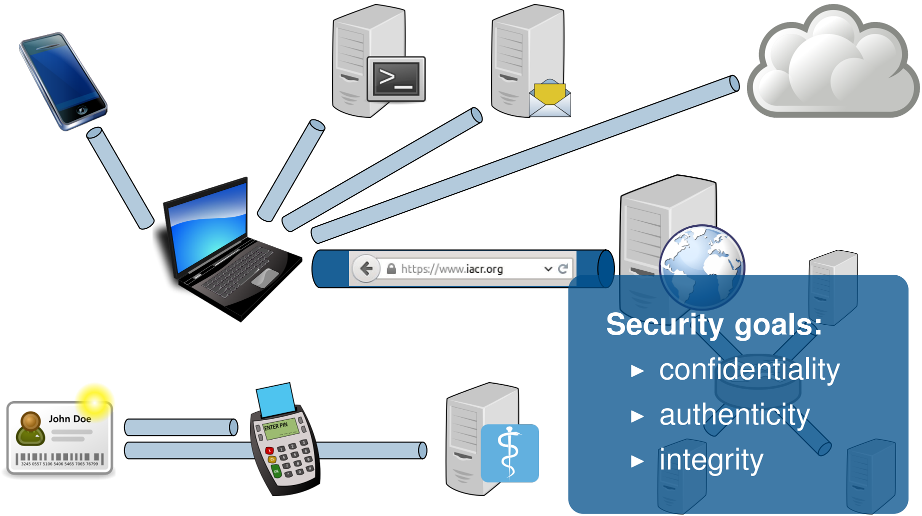
based on joint work with
Benjamin Dowling, Marc Fischlin, Britta Hale, Tibor Jager,
Sebastian Lauer, Giorgia Azzurra Marson, Kenneth G. Paterson,
Benedikt Schmidt, Douglas Stebila, Bogdan Warinschi

TECHNISCHE UNIVERSITÄT DARMSTADT

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

CROSSING

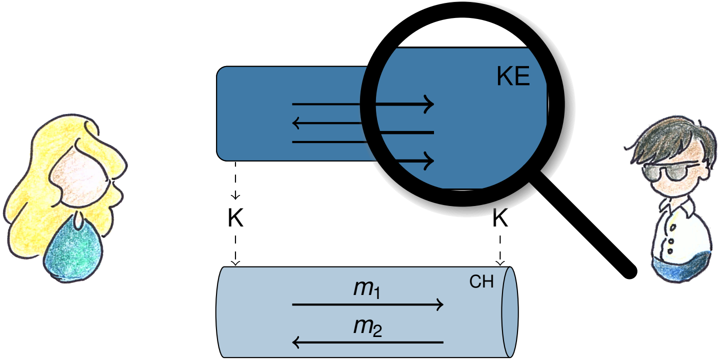
Secure Connections – Everywhere



Security goals:

- confidentiality
- authenticity
- integrity

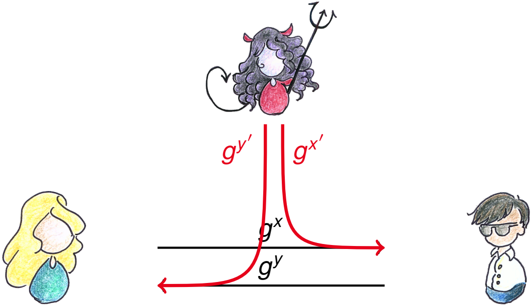
Secure Connections – Cryptographically



drawings by *Giorgia Azzurra Marson*

Key Exchange à la Diffie–Hellman (1976)

TECHNISCHE UNIVERSITÄT DARMSTADT



$K = (g^y)^x = g^{xy}$

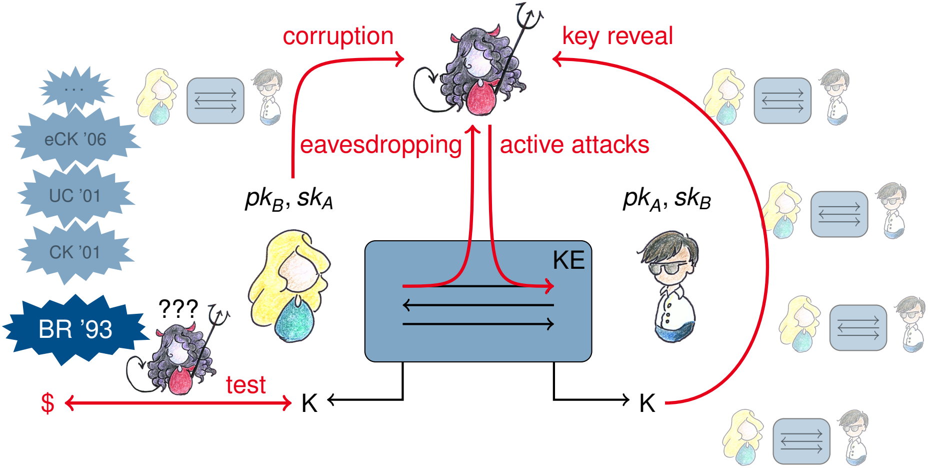
$K' = (g^{y'})^x = g^{xy'}$

$K = (g^x)^y = g^{xy}$

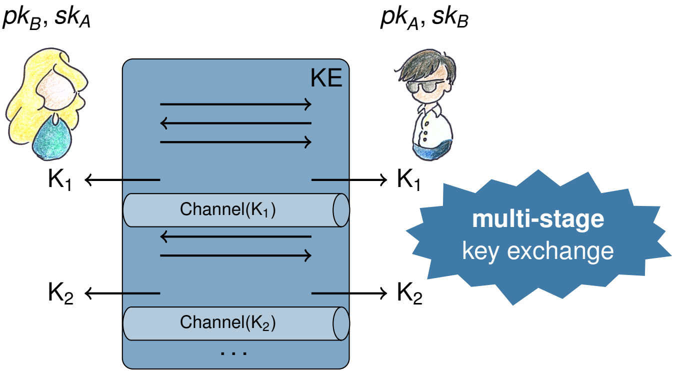
$K'' = (g^{x'})^y = g^{x'y}$

- key secrecy: given only g^x, g^y, key $K = g^{xy}$ remains secret
- no authentication: susceptible to man-in-the-middle attack

Key Exchange Security à la Bellare–Rogaway (1993)



But what if. . . ?



- key exchange establishes more than one key?
- . . . even uses the intermediary keys within the key exchange or channel?
- not covered by classical key exchange models

Should we care?

TECHNISCHE UNIVERSITÄT DARMSTADT

QUIC (“Quick UDP Internet Connections”, Google 2013)

- “low-latency transport protocol with security equivalent to TLS”
- Diffie–Hellman-based key exchange
- aims at 0-RTT, i.e., immediately encrypts under intermediate key K_1
- later rekeys to forward-secret K_2
- intermediate key K_1 used to establish K_2 (i.e., in KE part)

Marc Fischlin, Felix Günther
Multi-Stage Key Exchange and the Case of Google’s QUIC Protocol
ACM CCS 2014

Should we care?

TECHNISCHE UNIVERSITÄT DARMSTADT

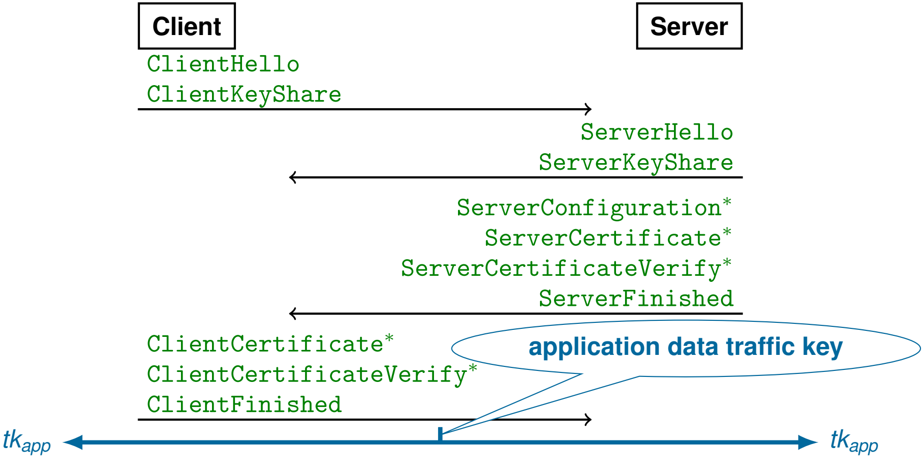
TLS 1.3

- 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. using only AEAD schemes for the record layer encryption
  3. providing reduced-latency 0-RTT handshake
  4. …

TLS 1.3 Full Handshake (simplified)

`draft-ietf-tls-tls13-10` (Oct 2015)

TECHNISCHE UNIVERSITÄT DARMSTADT



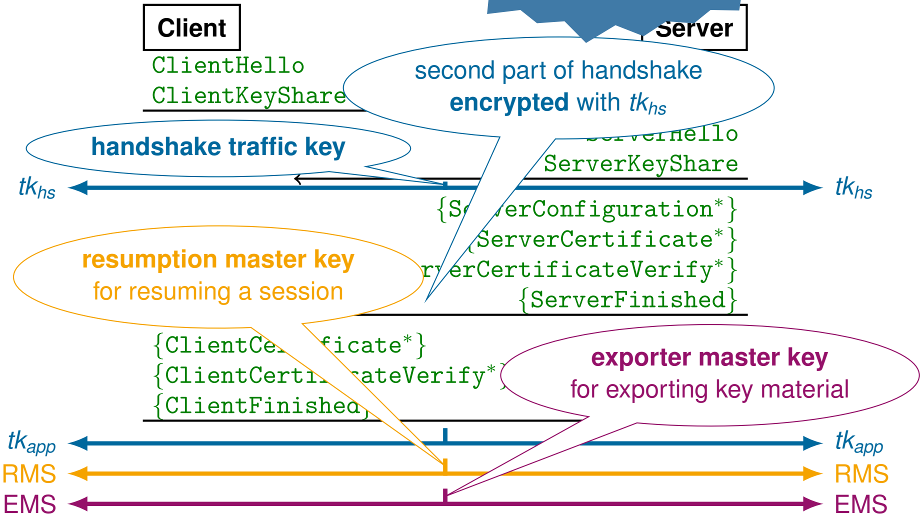
… actually, there is more …

TLS 1.3 Full Handshake (still simplified)

`draft-ietf-tls-tls13-10` (Oct 2015)

TECHNISCHE UNIVERSITÄT DARMSTADT

multi-stage
key exchange



Multi-Stage Key Exchange Analysis of TLS 1.3 Handshake Protocol Candidates

TECHNISCHE UNIVERSITÄT DARMSTADT

- full (DH) and preshared-key (resumption) handshakes (`draft-10` & earlier)
  - Benjamin Dowling, Marc Fischlin, Felix Günther, Douglas Stebila
    A Cryptographic Analysis of the TLS 1.3 Handshake Protocol Candidates
    ACM CCS 2015
  - Benjamin Dowling, Marc Fischlin, Felix Günther, Douglas Stebila
    A Cryptographic Analysis of the TLS 1.3 draft-10 Full and Pre-shared Key Handshake Protocol
    IACR ePrint 2016/081, TRON workshop @ NDSS 2016

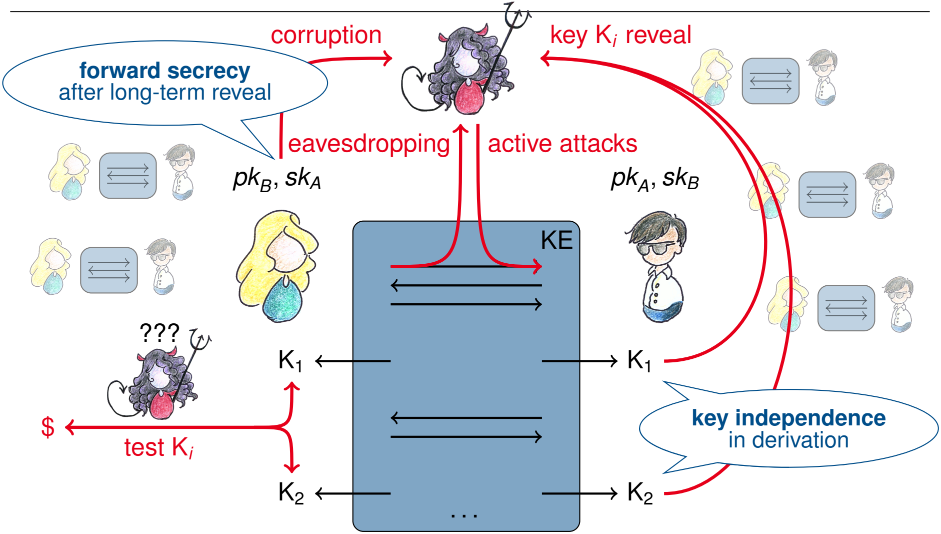
- (Diffie–Hellman-based) 0-RTT handshake (`draft-11`)
  - Marc Fischlin, Felix Günther (in submission)

- TLS 1.3 is work in progress (i.e., analysis not definitive)
  - contribution to and involved in ongoing discussion

STANDARD UNDER CONSTRUCTION

Multi-Stage Key Exchange Security

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



Modeling Multi-Stage Key Exchange

Further Aspects

TECHNISCHE UNIVERSITÄT DARMSTADT

Extensions for TLS 1.3

- **unauthenticated keys/stages** (beyond unilateral/mutual authentication)
  neither server nor client might send a certificate
- **concurrent execution of different authentication types**
  anonymous, server authenticates, server+client authenticate
- **pre-shared secret key variant**
  PSK/PSK-DHE handshake modes from preshared secrets (RMS)
- **replayable keys**
  0-RTT handshake messages can be replayed

TLS 1.3 Handshake Security

`draft-10` Full Handshake

TECHNISCHE UNIVERSITÄT DARMSTADT

(still simplified)

Client — **Server**

`ClientHello`: $r_c \leftarrow_\$ \{0,1\}^{256}$
`ClientKeyShare`: $X \leftarrow g^x$ →

`ServerHello`: $r_s \leftarrow_\$ \{0,1\}^{256}$
`ServerKeyShare`: $Y \leftarrow g^y$ ←

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

→ tk_{hs}

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

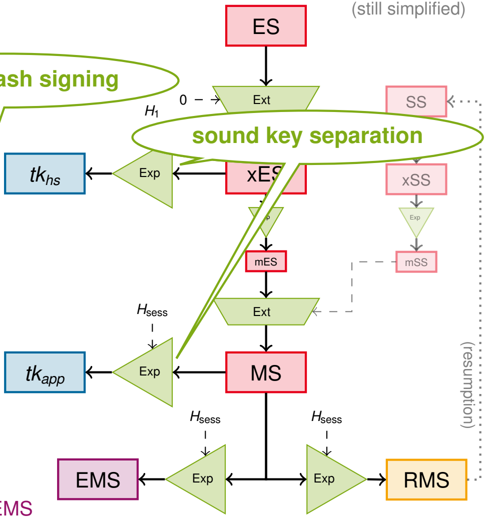
$\{\mathtt{CCert}^*\}$, $\{\mathtt{CCertVerify}^*\}$, $\{\mathtt{CFin}\}$ →

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

→ tk_{app}

→ RMS,EMS

session hash signing



TLS 1.3 Handshake Security

draft-10 Full Handshake

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

TLS 1.3 Handshake Security

Further Modes & Beyond

TECHNISCHE UNIVERSITÄT DARMSTADT

- **PSK/PSK-DHE handshake** [DFGS'15/16]
  - similar results as for full handshake
  - DHE variant enables forward secrecy
- **0-RTT handshake** [FG (sub)]
  - key & forward secrecy for all keys (with 0-RTT keys replayable + weaker fs)
  - based on pseudorandom-function oracle Diffie–Hellman (PRF-ODH) assumption
- **Key confirmation properties**
  - assurance that communication partner actually holds the shared key

  Marc Fischlin, Felix Günther, Benedikt Schmidt, Bogdan Warinschi
  Key Confirmation in Key Exchange: A Formal Treatment and Implications for TLS 1.3
  IEEE S&P 2016

TLS 1.3 Handshake Security

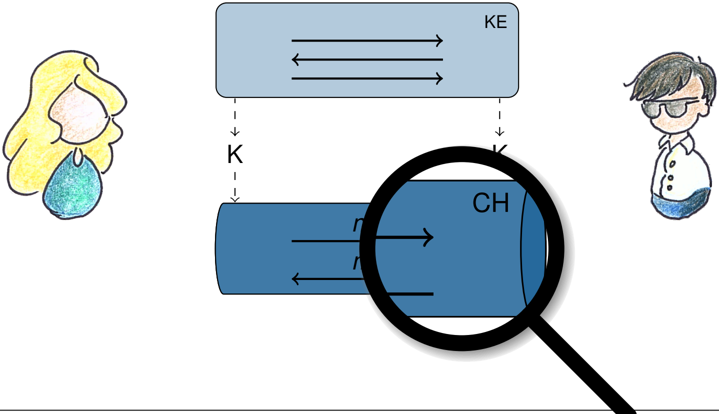
More Challenges

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Post-handshake messages**
  - allow late client authentication, key updates, and more
  - sent after initial handshake is over, but logically connected
- **Early (0.5-RTT) server data**
  - changing authentication of session key during usage
  - beyond what classical key exchange models capture
- **Forward-secret 0-RTT key exchange**
  - in current designs, forward secrecy is sacrificed in 0-RTT modes
  - new idea: leverage advanced crypto techniques to enable forward-secret 0-RTT

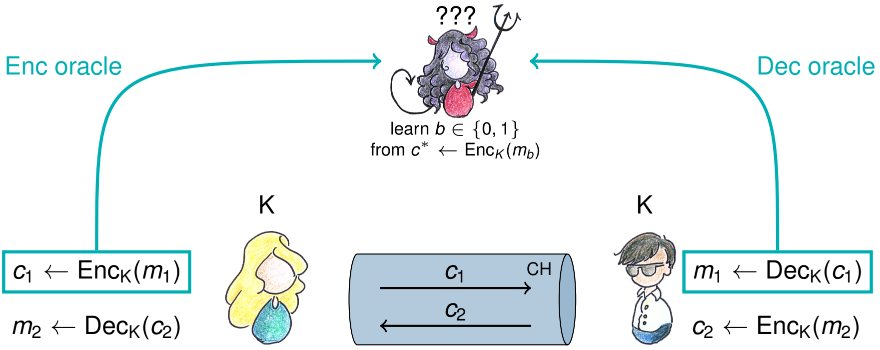
Felix Günther, Britta Hale, Tibor Jager, Sebastian Lauer (ongoing work)

Secure Connections – Cryptographically



On the Origin of Channel Models

Confidentiality

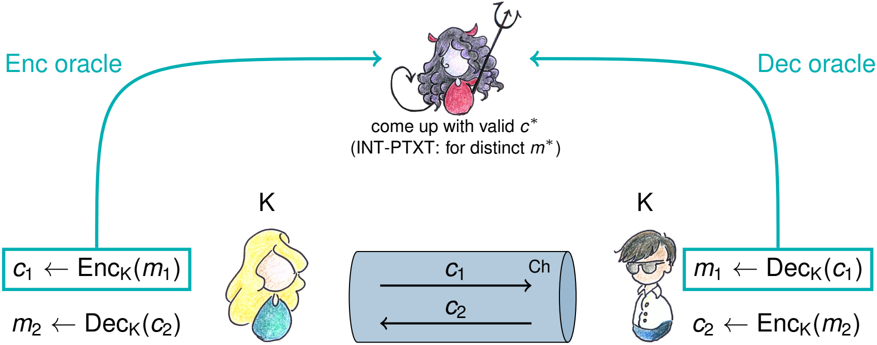


IND-CPA
(Goldwasser, Micali 1984)

IND-CCA
(Naor, Yung 1990), (Rackoff, Simon 1991)

On the Origin of Channel Models

Integrity



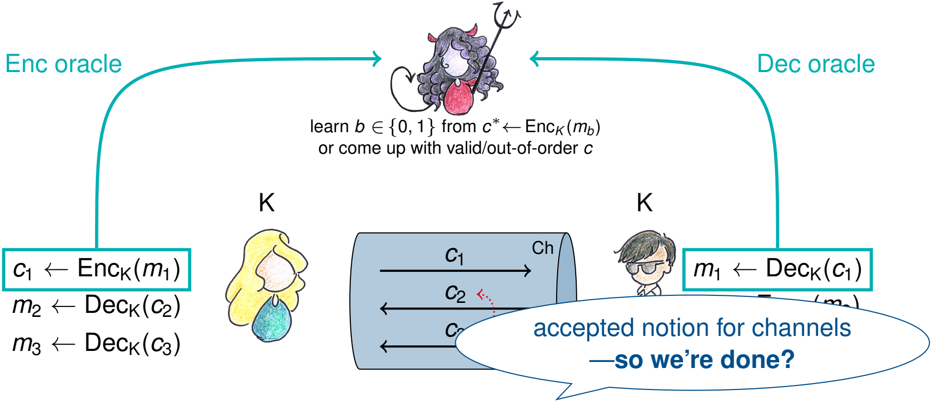
INT-PTXT
(Bellare, Namprempre 2000)

Authenticated Encryption
IND-CPA + INT-CTXT
($\Rightarrow$ IND-CCA)

INT-CTXT
(Bellare, Rogaway 2000)

On the Origin of Channel Models

Stateful Authenticated Encryption



Stateful Authenticated Encryption

used to analyze SSH

IND-sfCCA (Bellare, Kohno, Namprempre 2002) **INT-sfCTXT**

Attack on SSH

TECHNISCHE UNIVERSITÄT DARMSTADT

Albrecht, Paterson, Watson 2009: **plaintext recovery attack against SSH**
(SSH Binary Packet Protocol with CBC-mode Encode-then-Encrypt&MAC)

- adversary feeds ciphertext in *block-wise* (via TCP fragmentation)
- observable MAC failure can be used to leak plaintext → confidentiality break

Wait. . .

- SSH was proven IND-sfCCA and INT-sfCTXT secure! (BKN 2002)
- . . . but these only allow *atomic* ciphertexts in Dec oracle

Symmetric Encryption Supporting Fragmentation

(Boldyreva, Degabriele, Paterson, Stam 2012)

- general security model for ciphertext fragmentation
- standard Enc algorithm (and left-or-right oracle)
- Dec algorithm obtains ciphertext fragments, reassembles original messages

Are we there yet?

Attack on TLS

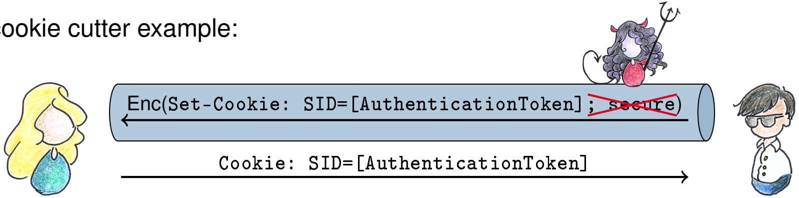
Cutting Cookies

TECHNISCHE UNIVERSITÄT DARMSTADT

Bhargavan, Delignat-Lavaud, Fournet, Pironti, Strub 2014: **cookie cutter attack**

- attacker truncates TLS connection by closing underlying TCP connection
- forces part of the HTTP header (e.g., cookie) to be cut off
- partial message/header arrives and might be misinterpreted

- cookie cutter example:



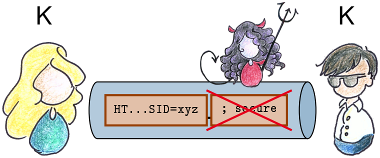
Wait. . . deleting message parts within ciphertext—how can this be possible?

Cookie Cutter Attack

A Closer Look

TECHNISCHE UNIVERSITÄT DARMSTADT

$c \leftarrow$ Enc(`HTTP/1.1 200 OK`
`...`
`Set-Cookie: SID=xyz; secure`)




```
#include <openssl/ssl.h>
SSL_write("HTTP/1.1 200 OK
          ...
          Set-Cookie: SID=xyz; secure")
```

```
HTTP/1.1 200 OK
...
Set-Cookie: SID=xyz
```

2 TLS records: [`HTTP/1.1 200 OK ... Set-Cookie: SID=xyz`] [`; secure`]

- fragmentation in TLS is implementation-specific
- adversary can potentially enforce a split at any point
  $\rightarrow$ receiver sees arbitrarily fragmented messages / no message boundaries

Data Is a Stream!

. . . and TLS is not alone

TECHNISCHE UNIVERSITÄT DARMSTADT

- That behavior is actually okay—and specified:

  6.2.1. Fragmentation
  The record layer fragments information blocks into TLSPlaintext records [...]. Client ***message boundaries are not preserved*** *in the record layer (i.e., multiple client messages of the same ContentType MAY be coalesced into a single TLSPlaintext record, or a single message MAY be fragmented across several records).*

  RFC 5246 TLS v1.2

- TLS never promised to treat messages atomically!
- indeed, many important channel protocols treat data as a stream
  - TLS
  - SSH tunnel-mode
  - QUIC
- so, there's a gap between what

  channel models capture  and channels expose to the application

Stream-Based Channels

Intuition and Security Notions

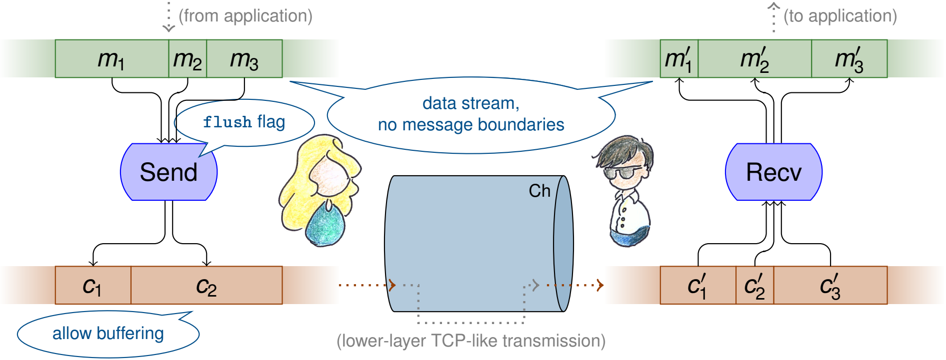
TECHNISCHE UNIVERSITÄT DARMSTADT

P2

Marc Fischlin, Felix Günther, Giorgia Azzurra Marson, Kenneth G. Paterson

Data Is a Stream: Security of Stream-Based Channels

CRYPTO 2015

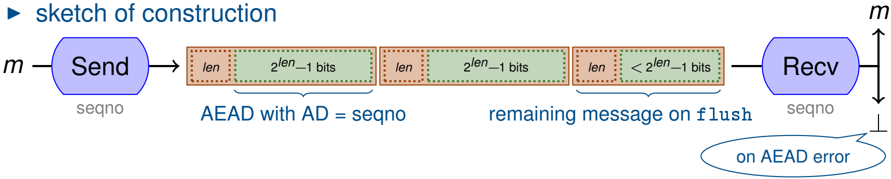


- adapted confidentiality and integrity notions for the stream-based setting

Stream-Based Channels
Generic Construction

TECHNISCHE UNIVERSITÄT DARMSTADT

- secure stream-based channels can be built
  - based on authenticated encryption with associated data (AEAD)
  - achieving strong IND-CCFA confidentiality
  - achieving strong INT-CST integrity
- sketch of construction



- close to TLS record layer design using AEAD (providing some validation)
  - ✓ sequence number authenticated, but not sent
  - ✓ sent length field, unauthenticated (in TLS 1.3)
  - ✗ TLS additionally includes, e.g., content type (sent authenticated)

The Journey Continues...

TECHNISCHE UNIVERSITÄT DARMSTADT

Further Properties

- Length-hiding for streams?
- Multiplexing of data (explicitly in QUIC, implicitly in TLS)

- How to safely encode atomic messages in a stream?
  - Marc Fischlin, Felix Günther, Giorgia Azzurra Marson, Kenneth G. Paterson
    Data Is a Stream: Security of Stream-Based Channels
    (upcoming full version)
    - integration into OpenCCE, preventing cookie cutter attack (demo) E1

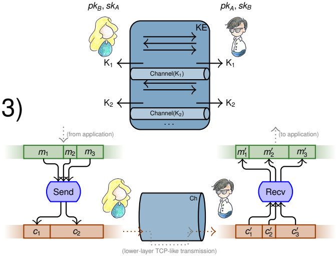
TLS 1.3 Record Protocol

- employs several traffic keys in the same protocol (for handshake + data)
- key switching requires care to prevent truncation attacks
  - **Multi-Key Channels** (ongoing work) P2

Conclusions

TECHNISCHE UNIVERSITÄT DARMSTADT

- basic properties of key exchange and secure channels are well-understood ?
- but advanced properties pose new challenges for security models



- in this talk:
  - **multi-stage key exchange** (QUIC, TLS 1.3)
  - **stream-based channels** (generic, TLS)
- **positive:** interaction of crypto, formal methods, and engineering community in development of TLS 1.3
- see **www.felixguenther.info** for the papers

Thank You!