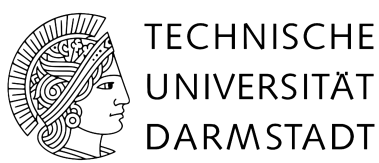
Replay Attacks on Zero Round-Trip Time: The Case of the TLS 1.3 Handshake Candidates

TECHNISCHE UNIVERSITÄT DARMSTADT

Felix Günther

Technische Universität Darmstadt, Germany

joint work with Marc Fischlin



TECHNISCHE UNIVERSITÄT DARMSTADT

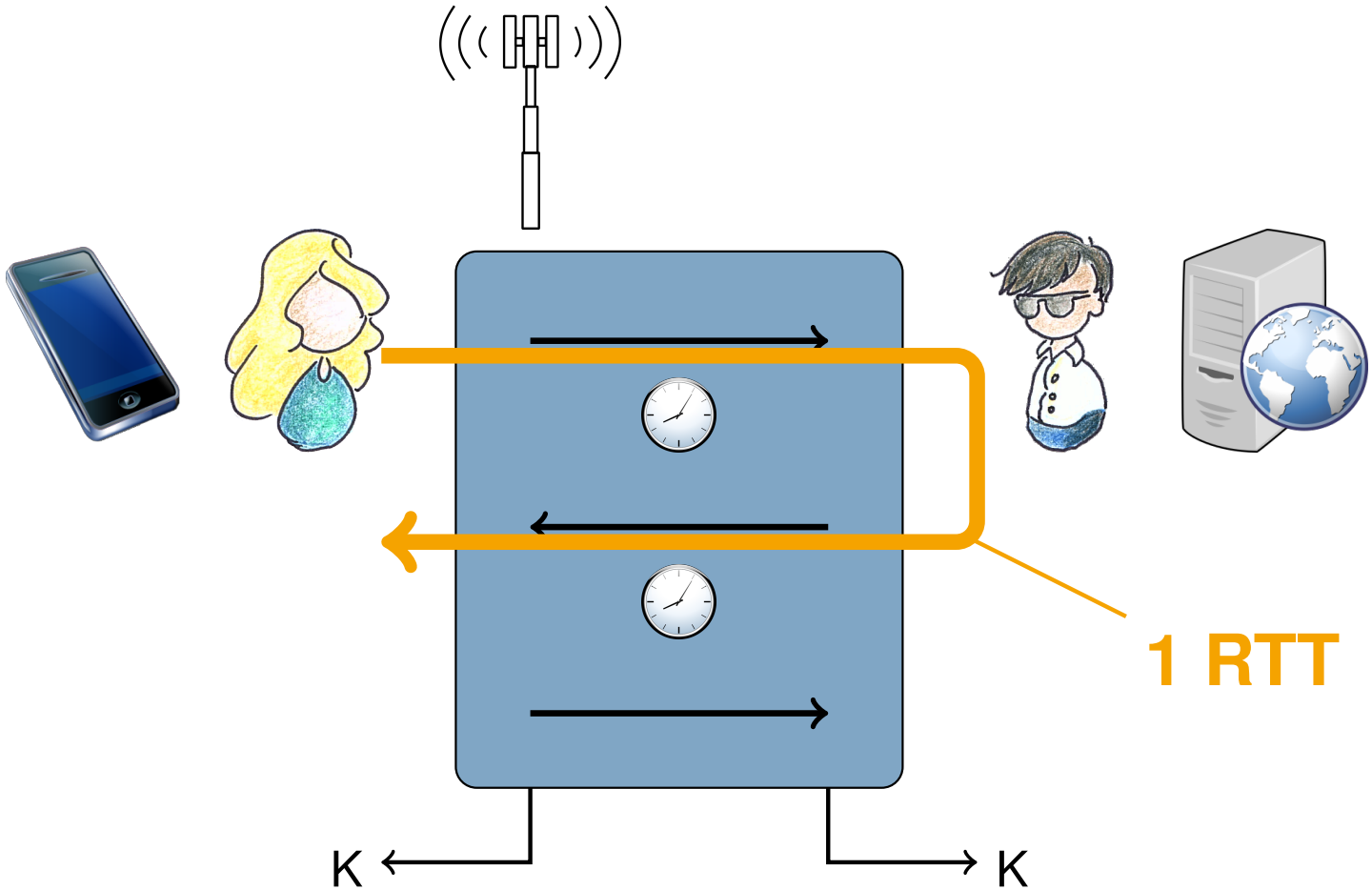
Cryptoplexity

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

CROSSING

Key Exchange
can be a bottleneck

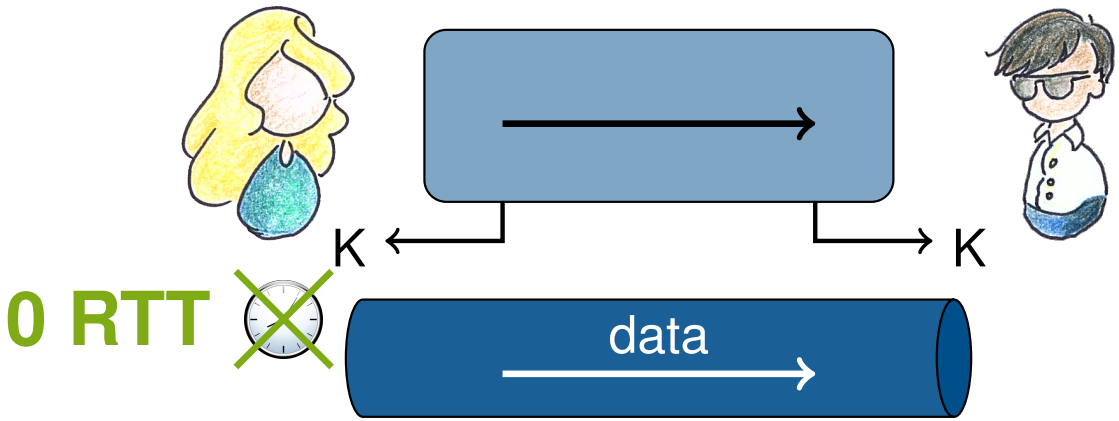
TECHNISCHE UNIVERSITÄT DARMSTADT



drawings by *Giorgia Azzurra Marson*

Solution: 0-RTT Key Exchange

TECHNISCHE UNIVERSITÄT DARMSTADT

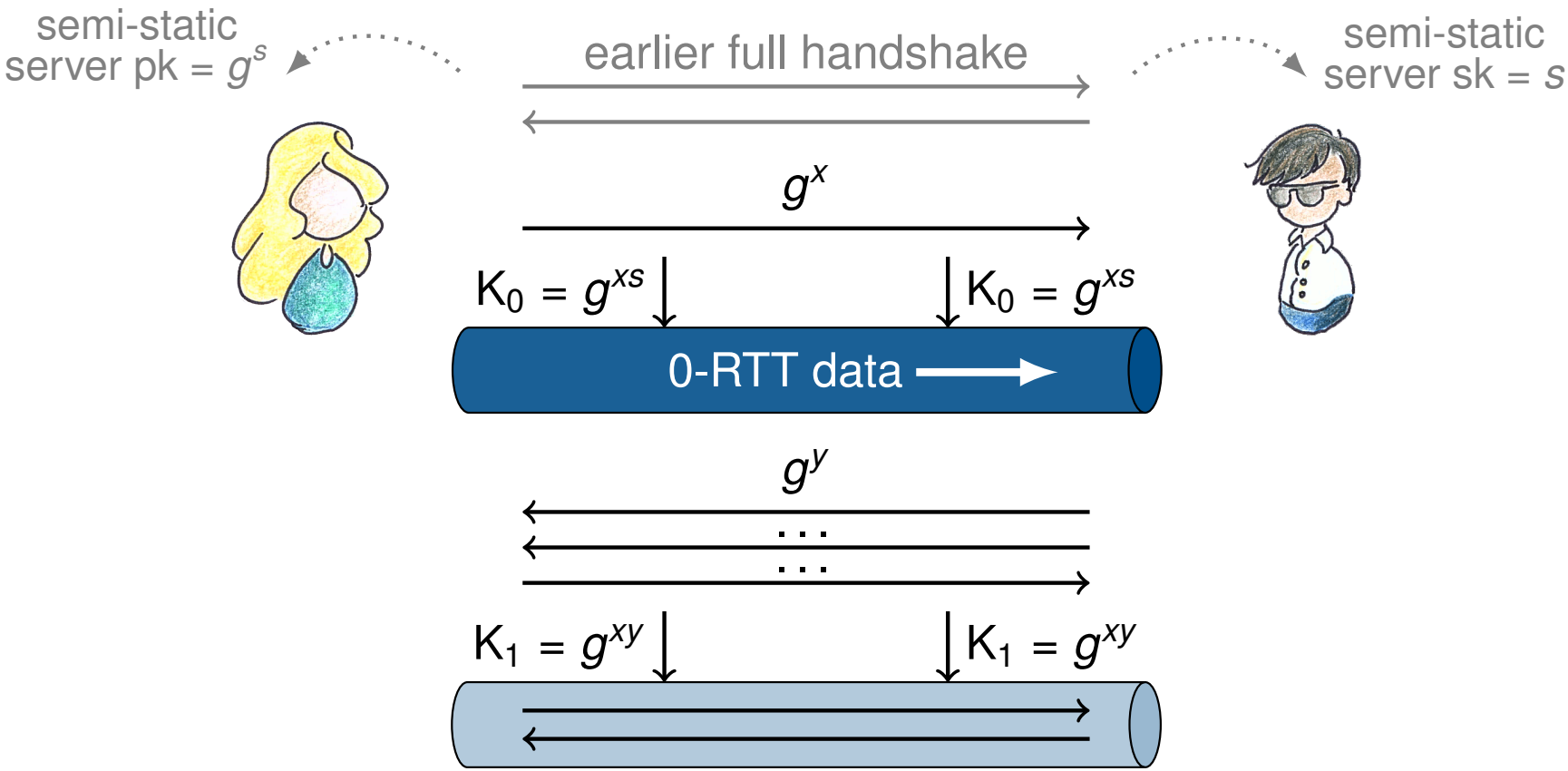


- theoretically not new
- in practice: **QUIC** (2013), **TLS 1.3** (2015+)

Diffie–Hellman-based 0-RTT

TECHNISCHE UNIVERSITÄT DARMSTADT

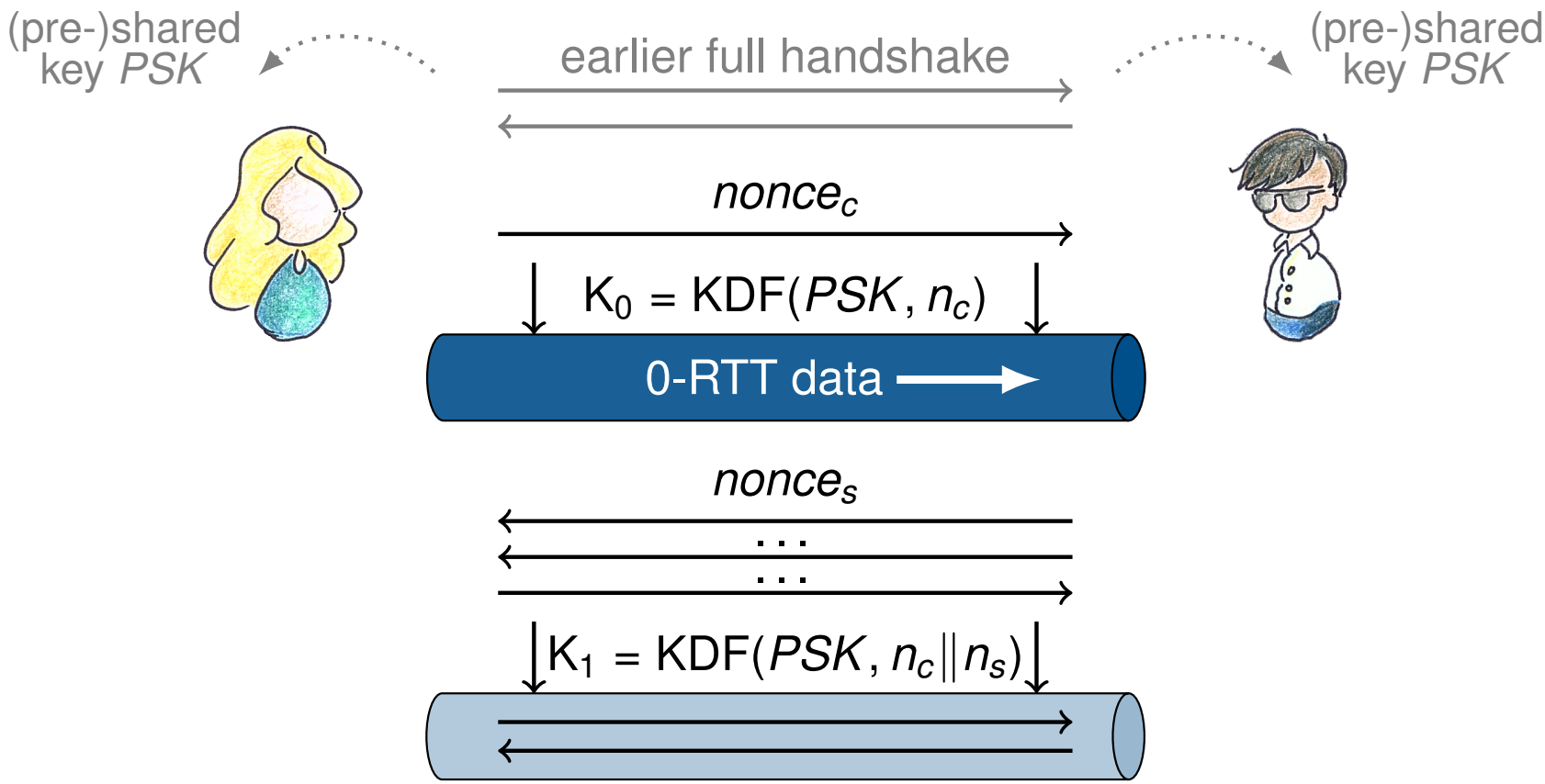
- à la QUIC, but also TLS 1.3 up to draft-12



PSK-based 0-RTT

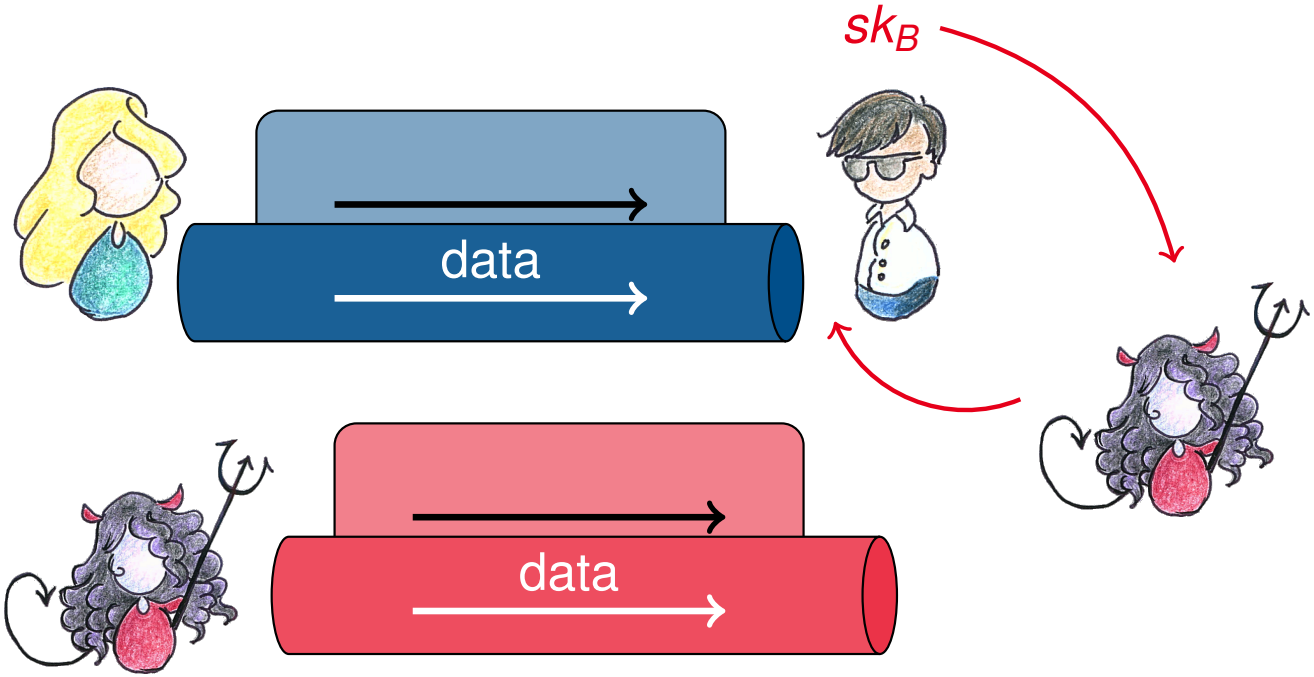
TECHNISCHE UNIVERSITÄT DARMSTADT

- TLS 1.3 since draft-13



Problems with 0-RTT Key Exchange

TECHNISCHE UNIVERSITÄT DARMSTADT



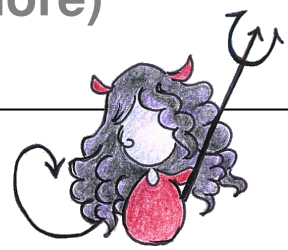
replays
(focus of this work)

no forward secrecy
[GHJL@Eurocrypt17]

Generic Replay Attack on 0-RTT

(by Daniel Kahn Gillmore)

TECHNISCHE UNIVERSITÄT DARMSTADT



0-RTT KE msg (n, g^x)

0-RTT data "/buy-something"

process "/buy-s..

accept 0-RTT, KE response (g^y)

enforce state loss (e.g., reboot)

simpler in real world:
send to two
distributed servers

replay 0-RTT KE msg (n, g^x)

QUIC: nonce
strike register

replay "/buy-s..

rej. after state loss
for security reasons

reject 0-RTT, KE response msg $(g^{y'})$

complete KE

(resend) "/buy-something" under final key

process "/buy-s..
again

Generic Replay Attack on 0-RTT

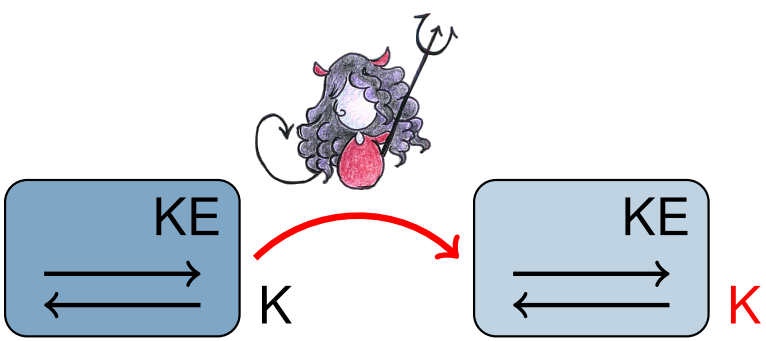
What's going on?

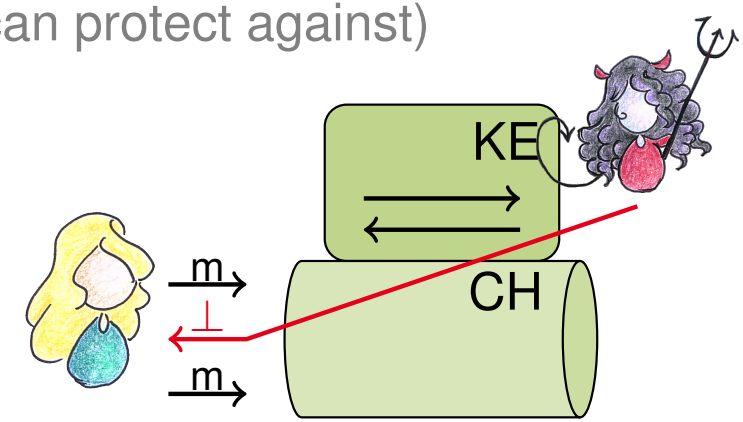
TECHNISCHE UNIVERSITÄT DARMSTADT

- is this an attack on the key exchange protocol?
- actually beyond KE: conscious replay on application level

> *"This isn't that odd, since, as AGL observes, browsers already* routinely retry *some HTTP requests that appear to fail* even for ordinary TLS *[...] but* of course that's different from having TLS give up those guarantees*."*
>
> Eric Rescorla @ TLS mailing list

- should distinguish between replay @ KE level and replay @ application level (latter fundamentally beyond what KE can protect against)





TLS 1.3's Response

TECHNISCHE UNIVERSITÄT DARMSTADT

- can't protect against replays anyway (on application level) . . .
- . . . so give up any strong replay protection for 0-RTT

  i.e.
- don't check for duplicate nonces, allow keys to be "replayed"
  optional: rough time check
- don't retransmit automatically on 0-RTT reject, but let application decide

- in theory: can be okay for idempotent requests
- in practice: hard to decide, likely to see attacks. . .

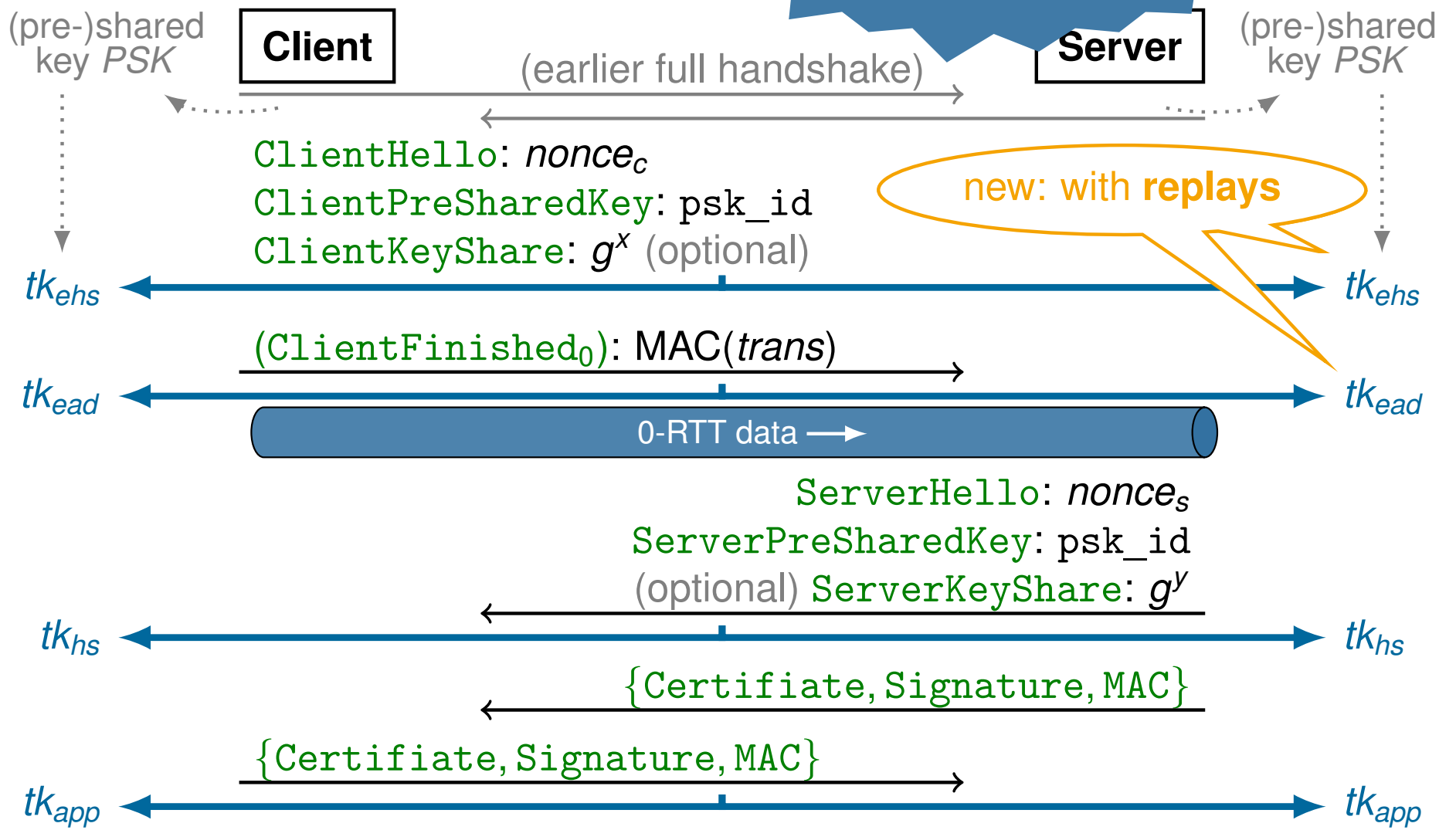
for more discussion
TLS:DIV
(Apr 30)

TLS 1.3 draft-14 PSK(-DHE) 0-RTT

(July 2016)

multi-stage key exchange

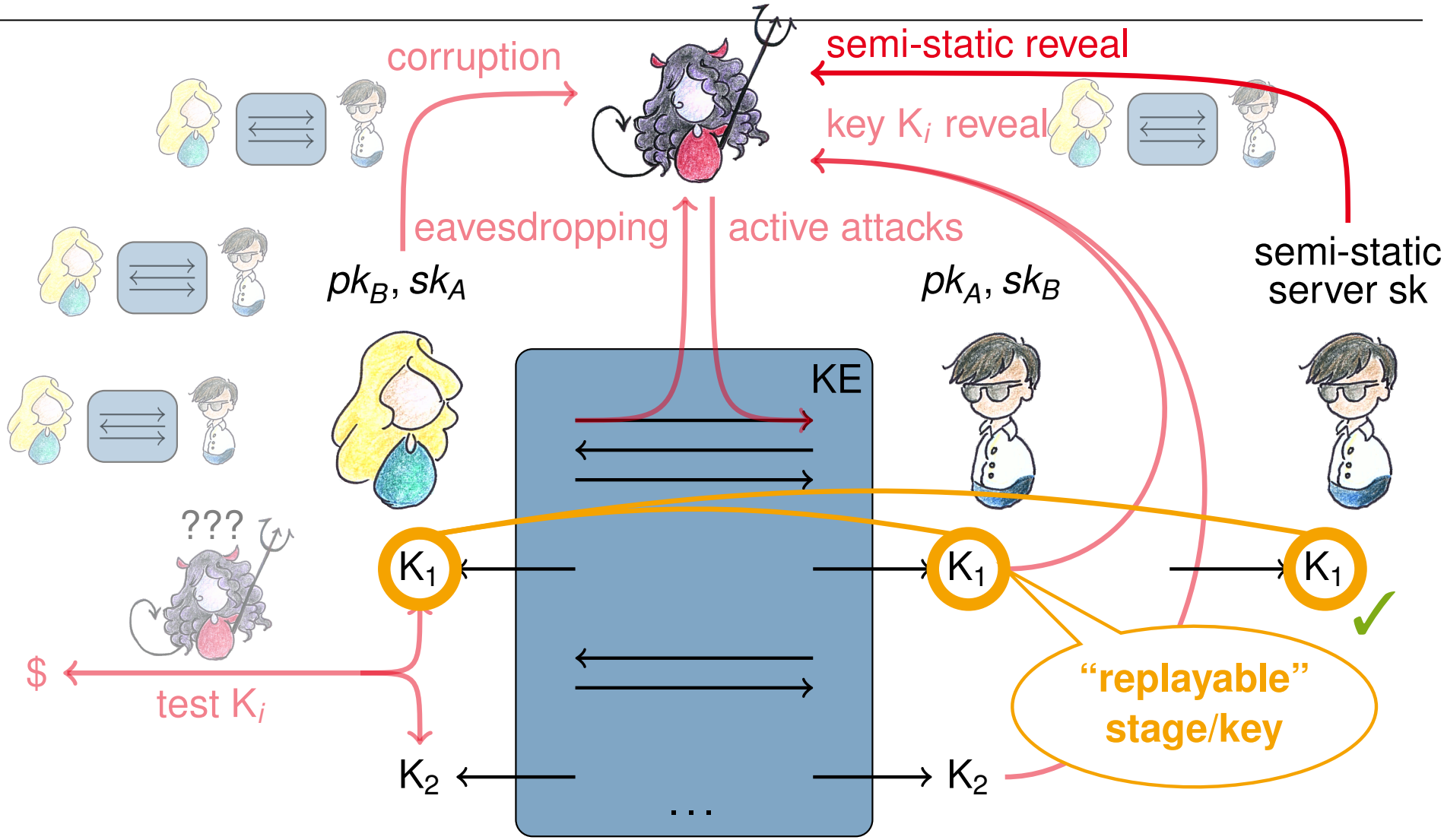
TECHNISCHE UNIVERSITÄT DARMSTADT



Multi-Stage Key Exchange (Security)

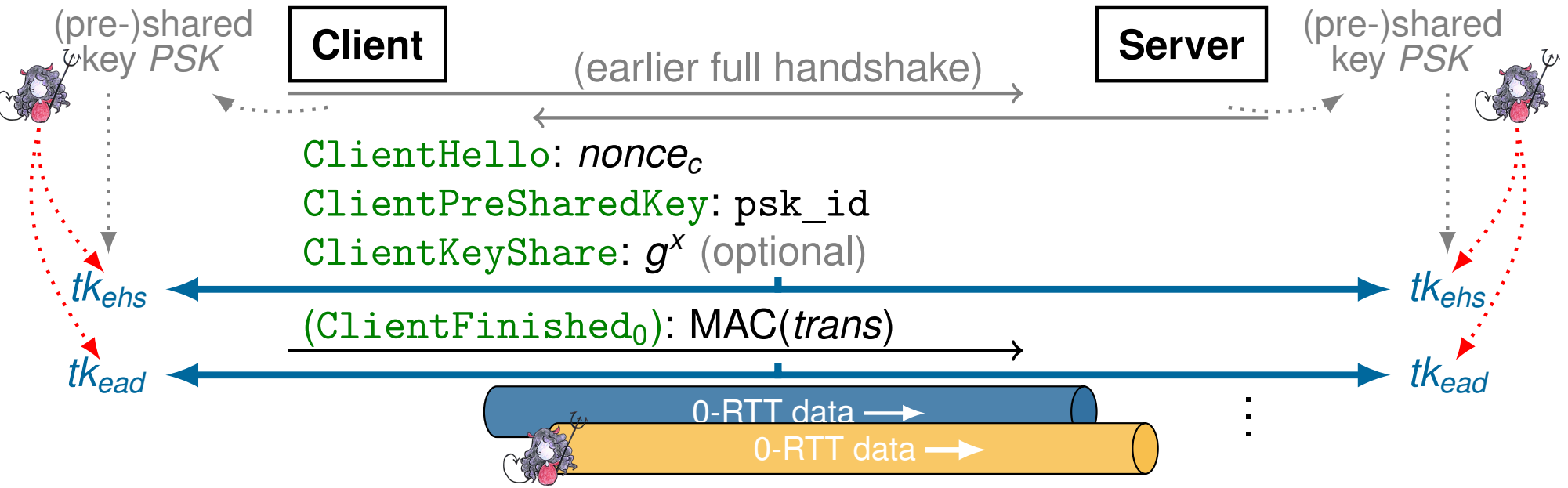
with replays

TECHNISCHE UNIVERSITÄT DARMSTADT



TLS 1.3 draft-14 PSK(-DHE) 0-RTT Security

TECHNISCHE UNIVERSITÄT DARMSTADT



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

Assuming:

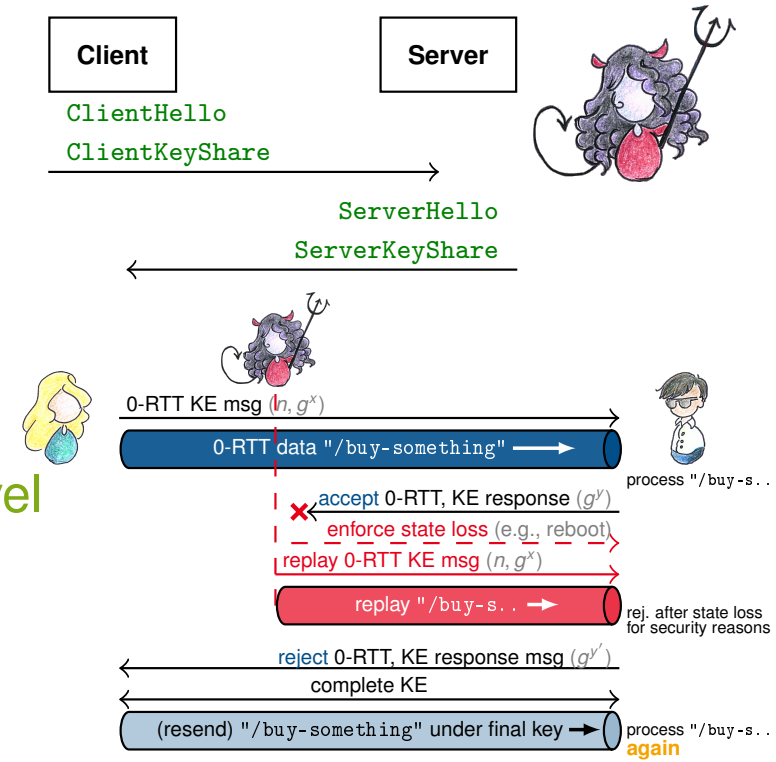
- hash function collision resistance
- HKDF is pseudorandom function
- HMAC unforgeability (DHE)
- PRF-ODH assumption holds (DHE)

Summary

TECHNISCHE UNIVERSITÄT DARMSTADT

We

- analyze the TLS 1.3 0-RTT modes (draft-14 PSK-based and draft-12 DH-based) as multi-stage key exchange with replays
- for 0-RTT, distinguish replays @ KE level from (unpreventable) replays @ application level
- careful with 0-RTT (replay) in practice
  - enables new attack vectors
  - easy to be misused by applications



full version @ IACR ePrint: `https://ia.cr/2017/082`

Thank You!

mail@**felixguenther.info**