Zero-Knowledge Proofs meet TLS

Paul Grubbs

based on joint work with:

Arasu Arun, Joseph Bonneau,

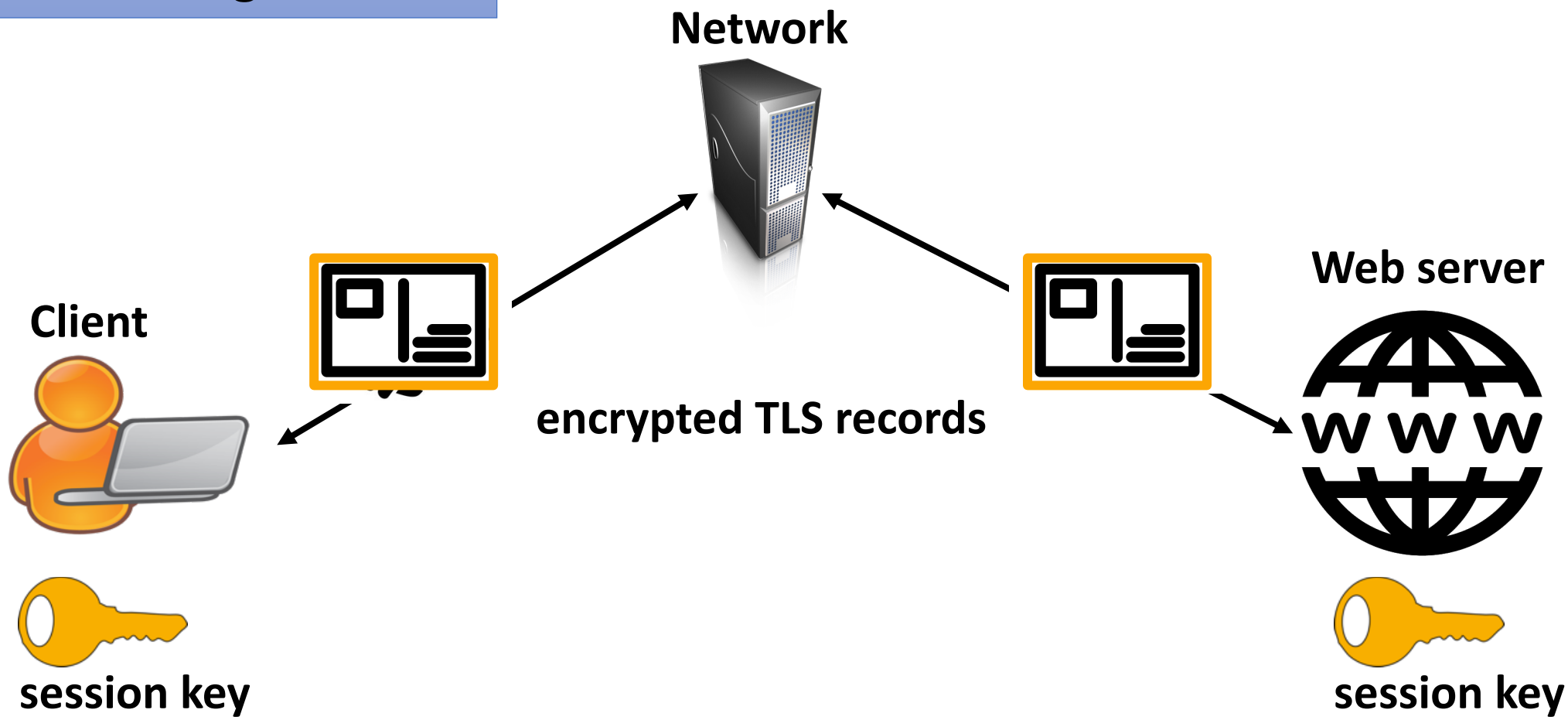
Zachary DeStefano, Michael Walfish, Ye Zhang

paulgrubbs12@gmail.com

Twitter: @pag_crypto

TLS is awesome

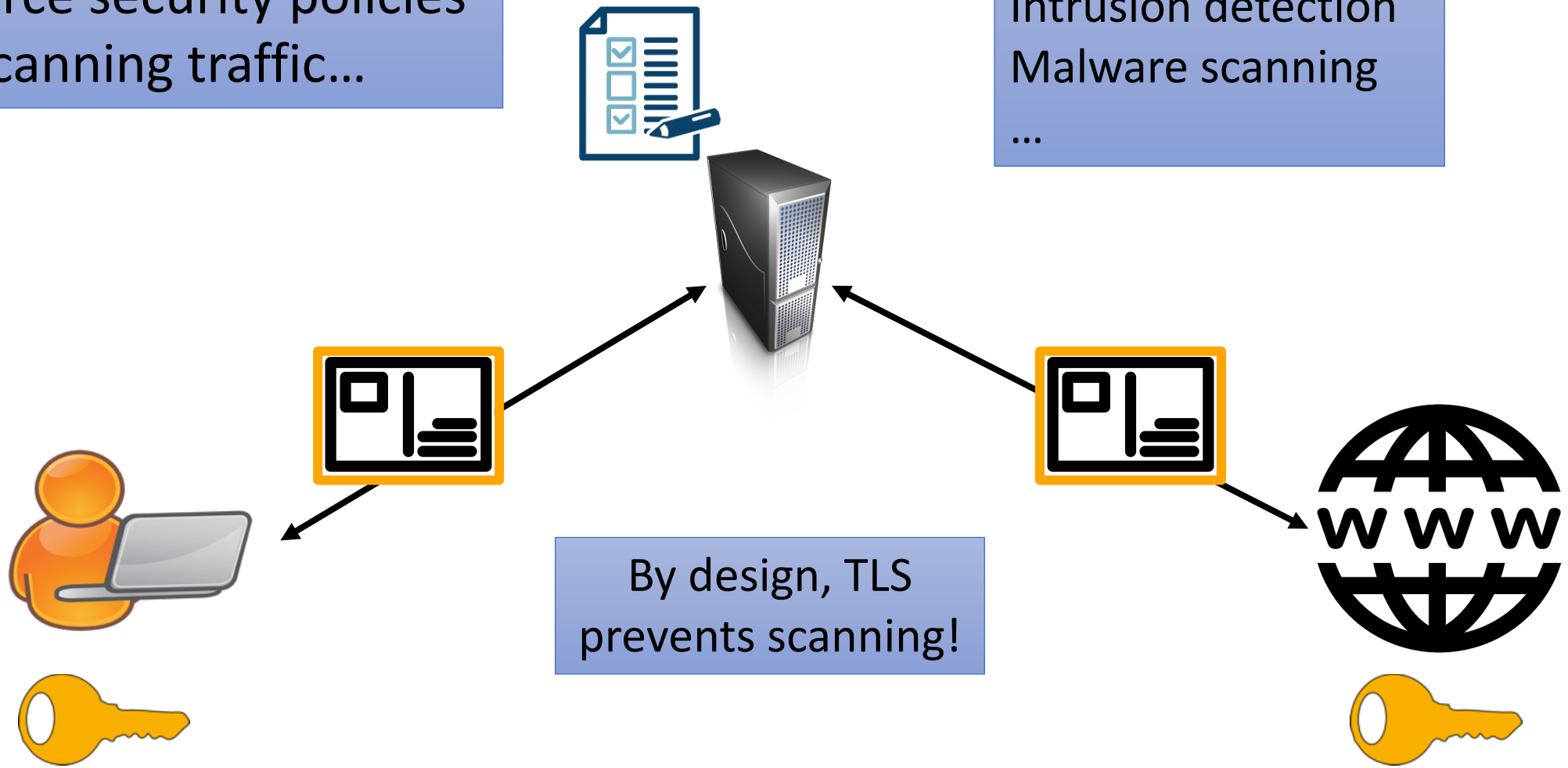
TLS 1.3 is great, and it's being deployed all over the place. Congrats!



”Visibility” Challenges

Network operators enforce security policies by scanning traffic…

DNS filtering
Data loss prevention
Intrusion detection
Malware scanning
…



By design, TLS prevents scanning!

IETF Mail Archive Sea

Date Thread

[TLS] Industry Concerns about TLS 1.3

BITS Security <BITSSecurity@fsroundtable.org> | Thu, 22 September 2016 17:24 UTC | Show header

To: IETF TLS 1.3 Working Group Members

My name is Andrew Kennedy and I work at BITS, the technology policy division of the Financial Services Roundtable (http://www.fsroundtable.org/bits). My organization represents approximately 100 of the top 150 US-based financial services companies including banks, insurance, consumer finance, and asset management firms.

ExtraHop THE PLATFORM SOLUTIONS CUSTOMERS PARTNERS BLOG MORE

< BLOG

The Network Is Going Dark: TLS 1.3 and Security Operations Visibility

NIST working group

NIST

Home > Security Guidance > Addressing Visibility Challenges with TLS 1.3

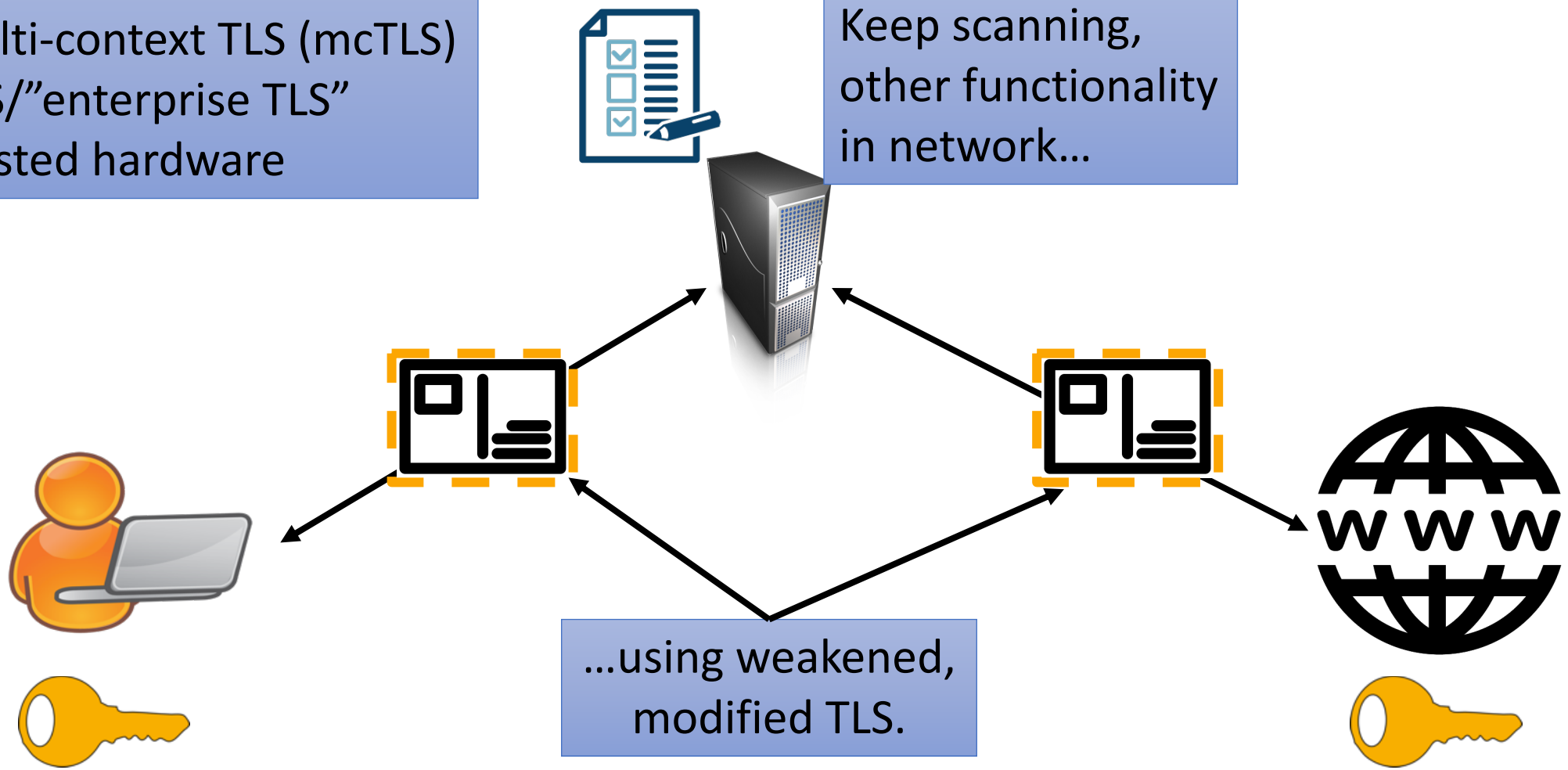
NCCoE NATIONAL CYBERSECURITY CENTER OF EXCELLENCE

SECURITY GUIDANCE

Addressing Visibility Challenges with TLS 1.3

Proposed Solutions

Prior work:
- multi-context TLS (mcTLS)
- ETS/”enterprise TLS”
- trusted hardware



Zero-Knowledge Proofs meet TLS

- Talk about recent work: “Zero-Knowledge Middleboxes”
  - Key contribution: zero-knowledge decryption of TLS records
- ZKP+TLS in general. Future directions
- Solicit feedback, criticism, suggestions for use cases

Zero-Knowledge Proofs

Public statement

Prover

Prover generates **zero-knowledge proof,** sends to verifier.

Verifier

Prover wants to convince verifier the statement is true, without revealing why.

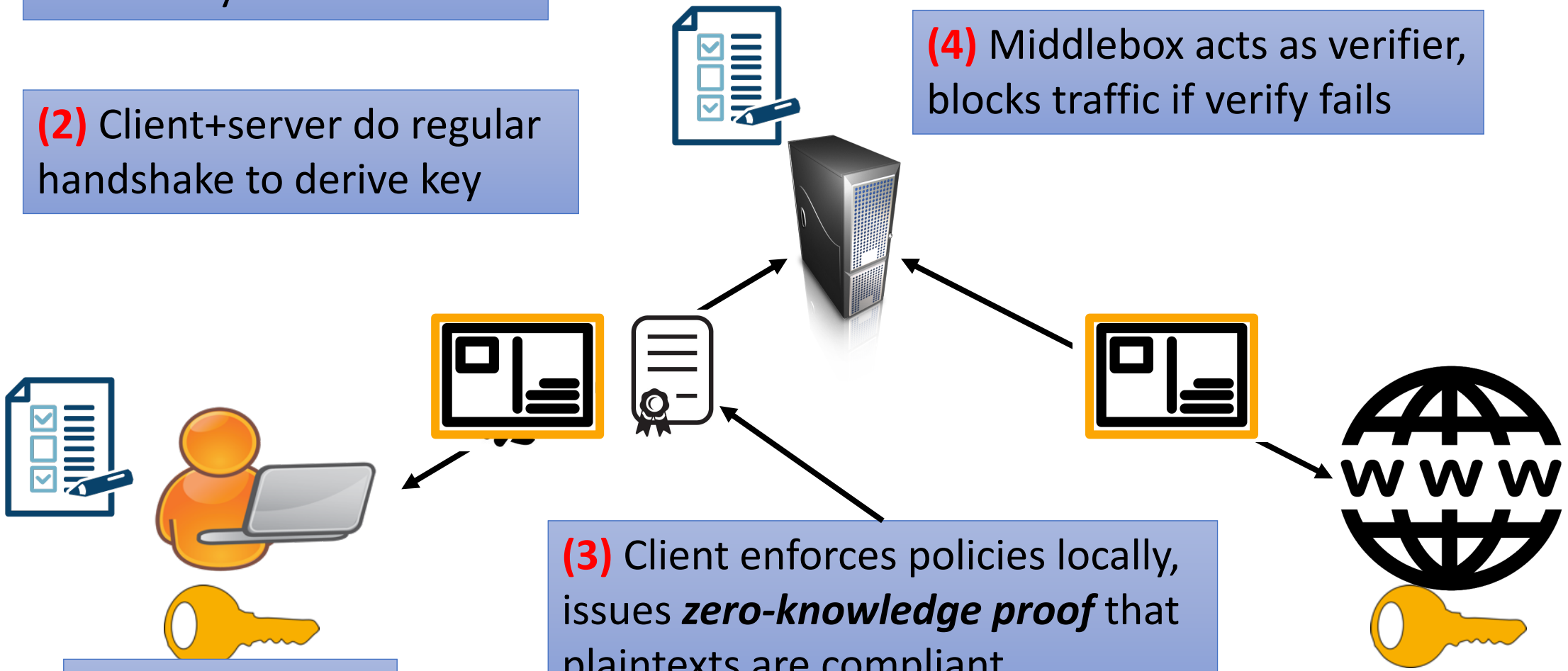
Verifier checks , learns the statement is true but not prover's "reasoning"

Zero-Knowledge Middleboxes (ZKMBs)

ZKMBs enforce policies on traffic they cannot see.

(2) Client+server do regular handshake to derive key

(4) Middlebox acts as verifier, blocks traffic if verify fails

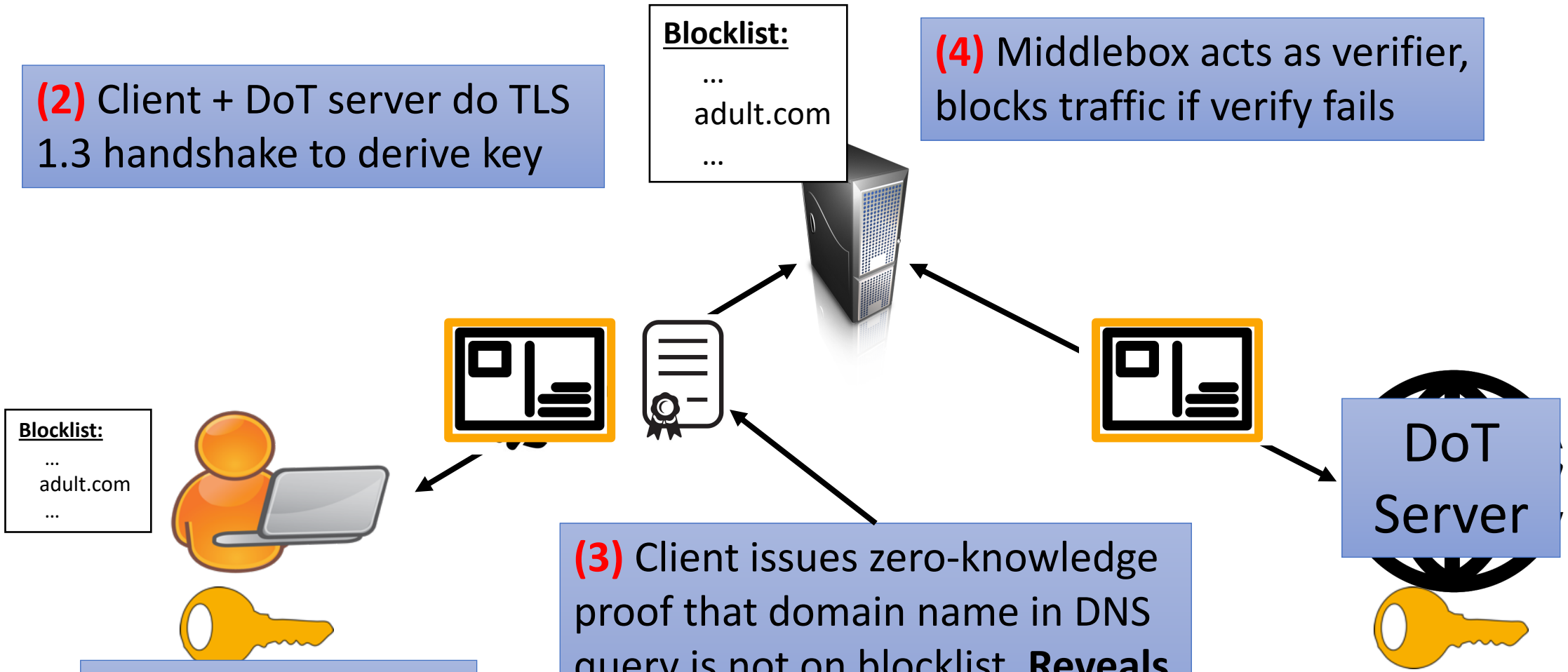


(3) Client enforces policies locally, issues ***zero-knowledge proof*** that plaintexts are compliant

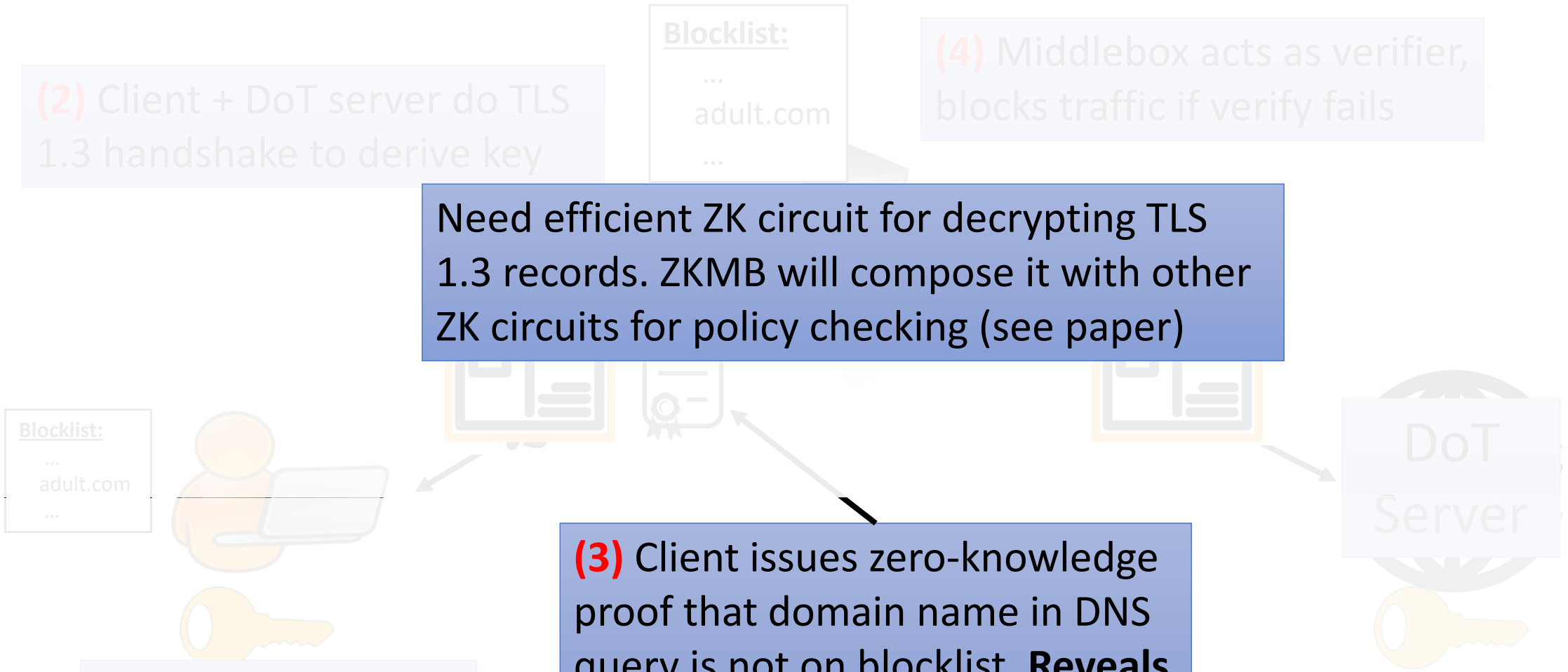
(1) Clients get policy on network join

ZKMB example:

DNS-over-TLS/HTTPS (DoT/DoH) filtering

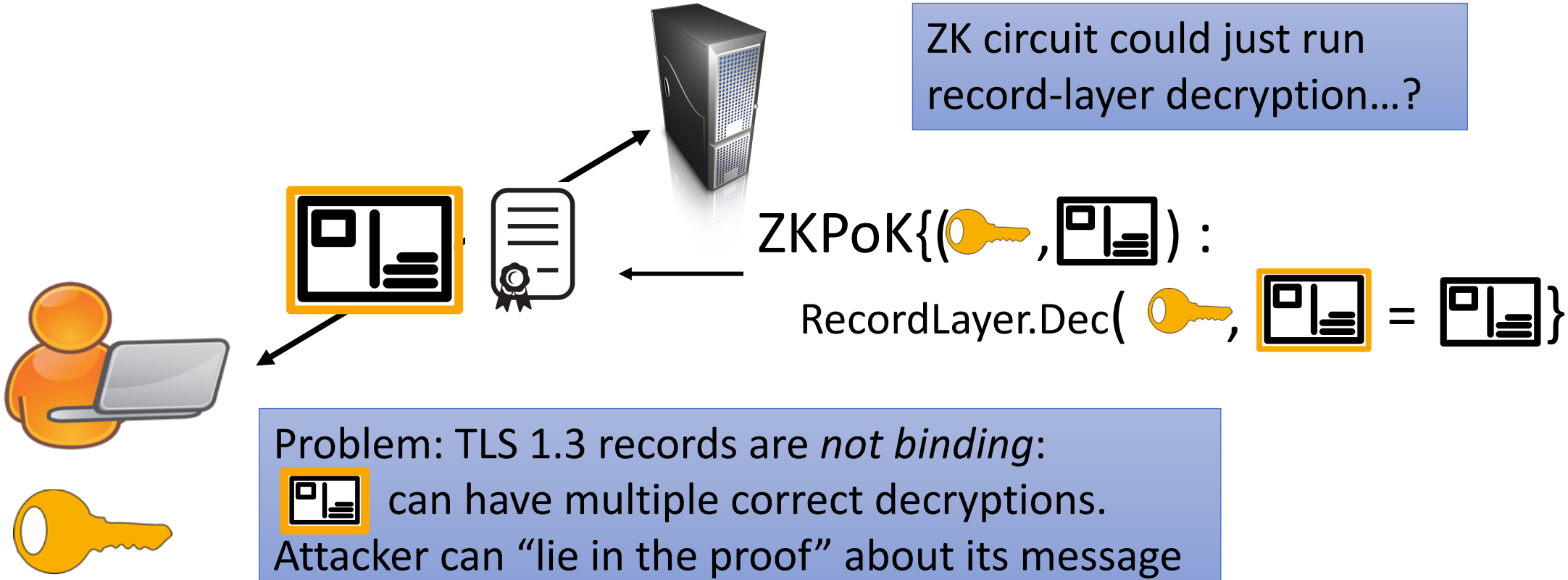


Key challenge



Decrypting TLS 1.3 Records in ZK

How to build ZK circuit that decrypts a TLS 1.3 record?



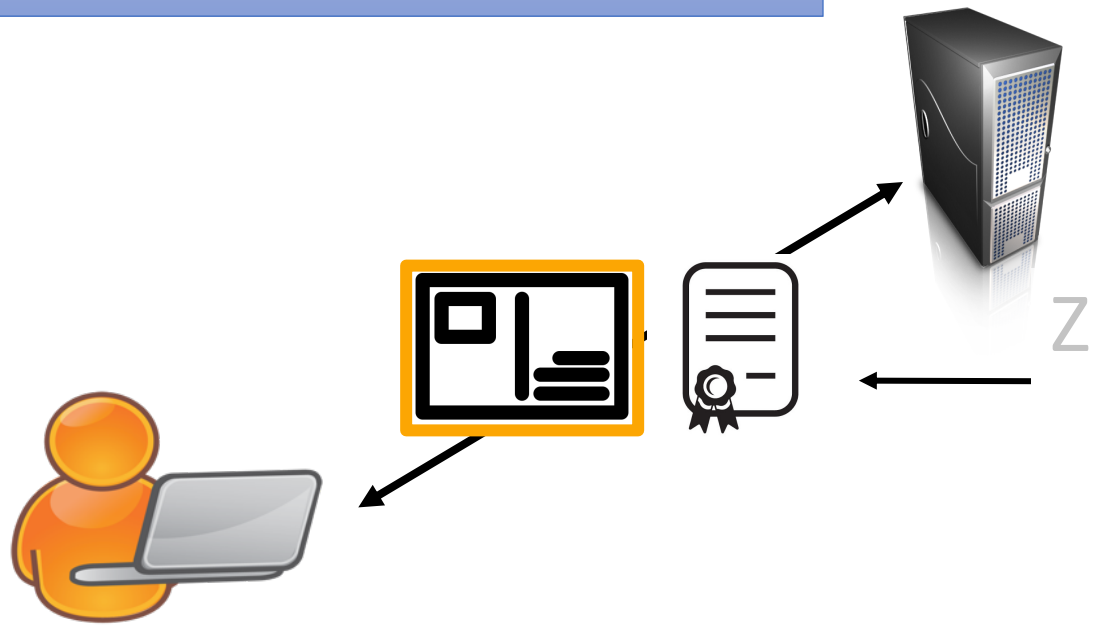
ZK circuit could just run record-layer decryption…?

Problem: TLS 1.3 records are *not binding*:
can have multiple correct decryptions.
Attacker can “lie in the proof” about its message

Decrypting TLS 1.3 Records in ZK

How to build ZK circuit that decrypts a TLS 1.3 record?

Idea to fix: prove key was output from handshake



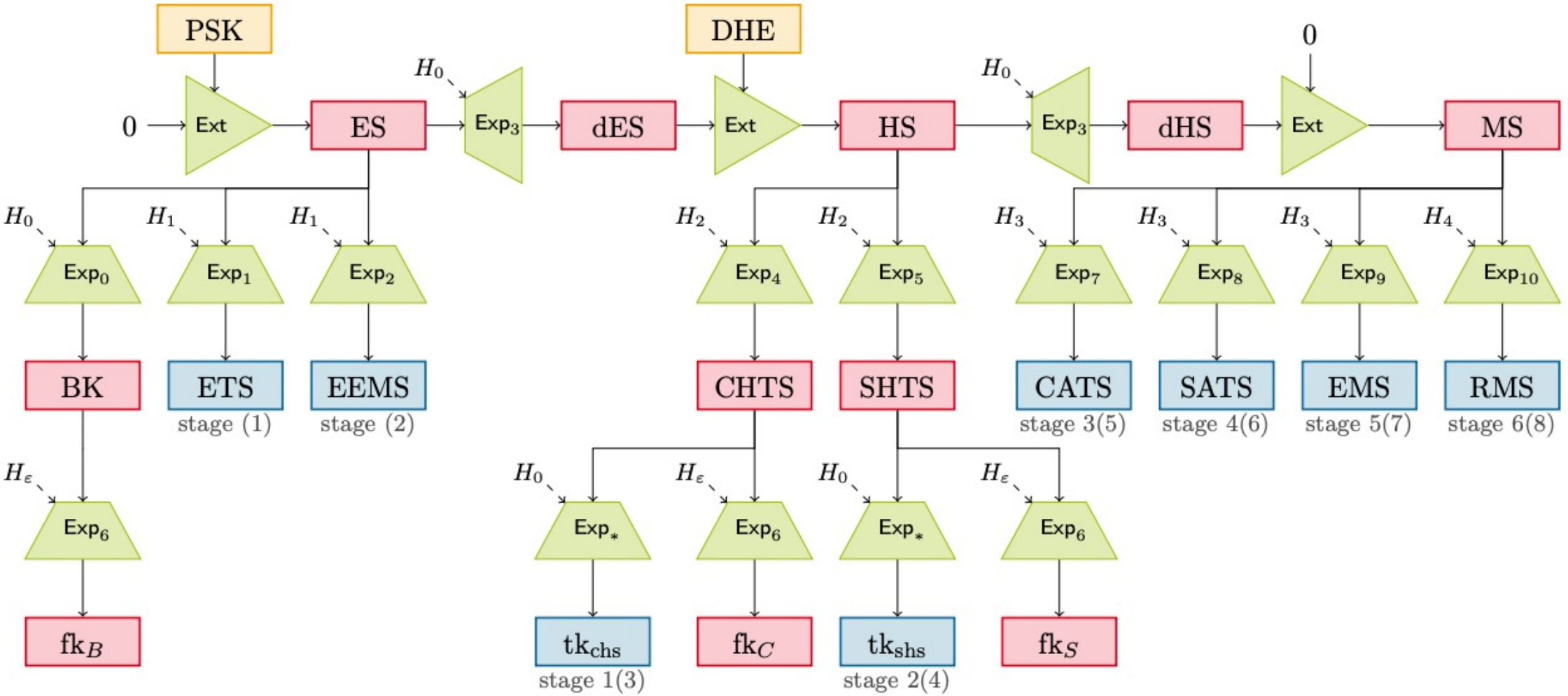
ZKPoK{(,) :

RecordLayer.Dec(,) =

and consistent with }

How do we build this key consistency check for TLS 1.3?

Key Consistency Check for TLS 1.3



Key Consistency Check for TLS 1.3

(the short version)

- Simple, inefficient: re-run most of client's key derivation in circuit.
  - Diffie-Hellman values are binding to shared secret
- Observation: handshake "commits to" intermediate steps of key derivation. Check these to shortcut key derivation

Experimental Results

- Key consistency check (one-time cost per TLS session): ~15 seconds
- Cost for DoT filtering use case: ~3 seconds per DNS query
- Proof verification cost: 5 milliseconds
- tl;dr: not practical yet, but close.
  - Many possible optimizations – ongoing work reduces client costs by 70x, with caveats.
  - See paper online for more numbers

Zero-Knowledge Proofs meet TLS

- Talk about recent work: “Zero-Knowledge Middleboxes”
  - Key contribution: zero-knowledge decryption of TLS record
- ZKP+TLS in general. Future directions
- Solicit feedback, criticism, suggestions for use cases

ZKP+TLS future directions

- ZKPs are an intriguing new tool. Many interesting applications to TLS.
  - ZKPs of Encrypted Client Hello (ECH) plaintexts
- Open problems, future directions as well.
  - Key updates, client authentication, HRR, PHA…
  - Explicit support for ZKPs in TLS via extensions?
- Outlandish suggestion: could design future versions of TLS to be ”ZKP-friendly”

Conclusion

- TLS deployments cause ”visibility” challenges in networks. ZKPs can help.
- Described new research on *zero-knowledge middleboxes*, which use ZKPs to enable privacy-preserving enforcement of network policies like DNS filtering
  - https://eprint.iacr.org/2021/1022
- Many remaining research opportunities in ZKP+TLS. Interesting design consideration for future TLS versions.

Thanks for listening! Any questions?

paulgrubbs12@gmail.com
Twitter: @pag_crypto

Experimental Results

Results:

(* = ChaCha20 instead of AES)

Channel Opening Microbenchmark

CO Method	#Gates x 10^4	Time (s)	SRS (MB)
Baseline	748 (7x)	94.0 (6x)	1200
Shortcut	111	16.5	149
Early data	61	8.6	81

DNS Case Studies (after setup)

Case Study	Ctxt Size	#Gates x 10^4	Time (s)	SRS (MB)
DoT*	255	19.5	3.1	32
DoH	500	49.5	6.8	75
ODoH	500	48.0	6.4	61

Prototype can generate proof for nontrivial ZKMB in 3 seconds.