

# Towards Modular Trusted Execution Environments

Carsten Weinhold<sup>1</sup>, Nils Asmussen<sup>1</sup>, Diana Göhringer<sup>2</sup>, Michael Roitzsch<sup>1</sup>

<sup>1</sup>Barkhausen Institut

<sup>2</sup>Technische Universität Dresden

# **Towards Modular Trusted Execution Environments**

Part 1

# **Transport Layer Security + Remote Attestation**

Part 2

# Lots of TEEs, but not so much choice ...



	TEE ISA	Implementation	TEE Type	Integration
Intel SGX	x86-64	complex core	enclave	libOS / wrapper
Intel TDX	x86-64	complex core	VM	standalone
AMD SVM	x86-64	complex core + PSP	VM	standalone
Arm CCA	Armv9	complex core	VM	standalone
Sanctum	RISC-V	core + mitigation	enclave	libOS / wrapper
Sancus	TI MSP430	simple core	enclave / module	???

**Spectre Attacks: Exploiting Speculative Execution**

Paul Kocher<sup>1</sup>, Jann Horn<sup>2</sup>, Anders Fogh<sup>3</sup>, Daniel Genkin<sup>4</sup>, Daniel Gruss<sup>5</sup>, Werner Haas<sup>6</sup>, Mike Hamburg<sup>7</sup>, Moritz Lipp<sup>8</sup>, Stefan Mangard<sup>9</sup>, Thomas Prescher<sup>10</sup>, Michael Schwarz<sup>11</sup>, Yuval Yarom<sup>12</sup>

<sup>1</sup> Independent (www.paulkocher.com), <sup>2</sup> Google Project Zero, <sup>3</sup> G DATA Advanced Analytics, <sup>4</sup> University of Pennsylvania and University of Maryland, <sup>5</sup> Graz University of Technology, <sup>6</sup> Cyberus Technology, <sup>7</sup> Rambus, Cryptography Research Division, <sup>8</sup> University of Adelaide and Data61

**Abstract**—Modern processors use branch prediction and speculative execution to maximize performance. For example, if the destination of a branch depends on a memory value that is in the process of being read, CPUs will speculatively execute the branch and leverage hardware vulnerabilities to leak sensitive information. Attacks of the latter type include microarchitectural attacks exploiting cache timing [8, 30, 48, 52, 55, 69, 74], branch

**FORESHADOW: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order Execution**

Jo Van Bulck<sup>1</sup>, Marina Minkin<sup>2</sup>, Ofir Weisse<sup>3</sup>, Daniel Genkin<sup>4</sup>, Baris Kasikci<sup>5</sup>, Frank Piessens<sup>1</sup>, Mark Silberstein<sup>6</sup>, Thomas F. Wenisch<sup>7</sup>, Yuval Yarom<sup>8</sup>, and Raoul Strackx<sup>1</sup>

<sup>1</sup>imec-DistriNet, KU Leuven, <sup>2</sup>Techion, <sup>3</sup>University of Michigan, <sup>4</sup>University of Adelaide and Data61

**Abstract** Trusted execution environments, and particularly the Software Guard Extensions (SGX) included in recent Intel x86 processors, gained significant traction in recent years. Trusting execution environments with a minimal Trusted Computing Base (TCB) that includes only the processor package and microcode. Enclave-private CPU and memory state is exclusively accessible to the code running inside it, and remains explicitly out of reach of all other enclaves and

2019 IEEE Symposium on Security and Privacy

**RIDL: Rogue In-Flight Data Load**

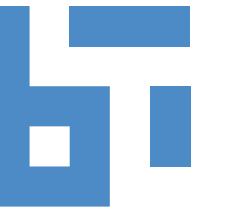
Stephan van Schaik<sup>1</sup>, Alyssa Milburn<sup>2</sup>, Sebastian Osterlund<sup>3</sup>, Pietro Frigo<sup>4</sup>, Giorgi Maisuradze<sup>1,2</sup>, Kaveh Razavi<sup>1</sup>, Herbert Bos<sup>1</sup>, and Cristiano Giuffrida<sup>2</sup>

<sup>1</sup>Department of Computer Science, Vrije Universiteit Amsterdam, The Netherlands {s.j.v.schaik, a.a.milburn, s.osterlund, p.frigo}@vu.nl, {kaveh, herbert, giuffrida}@cs.vu.nl

<sup>2</sup>CISPA Helmholtz Center for Information Security Saarland Informatics Campus, giorgi.maisuradze@cispa.saarland

**Abstract**—We present *Rogue In-Flight Data Load (RIDL)*, a new class of speculative unprivileged and practical “spot” mitigations against existing attacks [6, 7], [8], [9]. This shaped the common perception that—

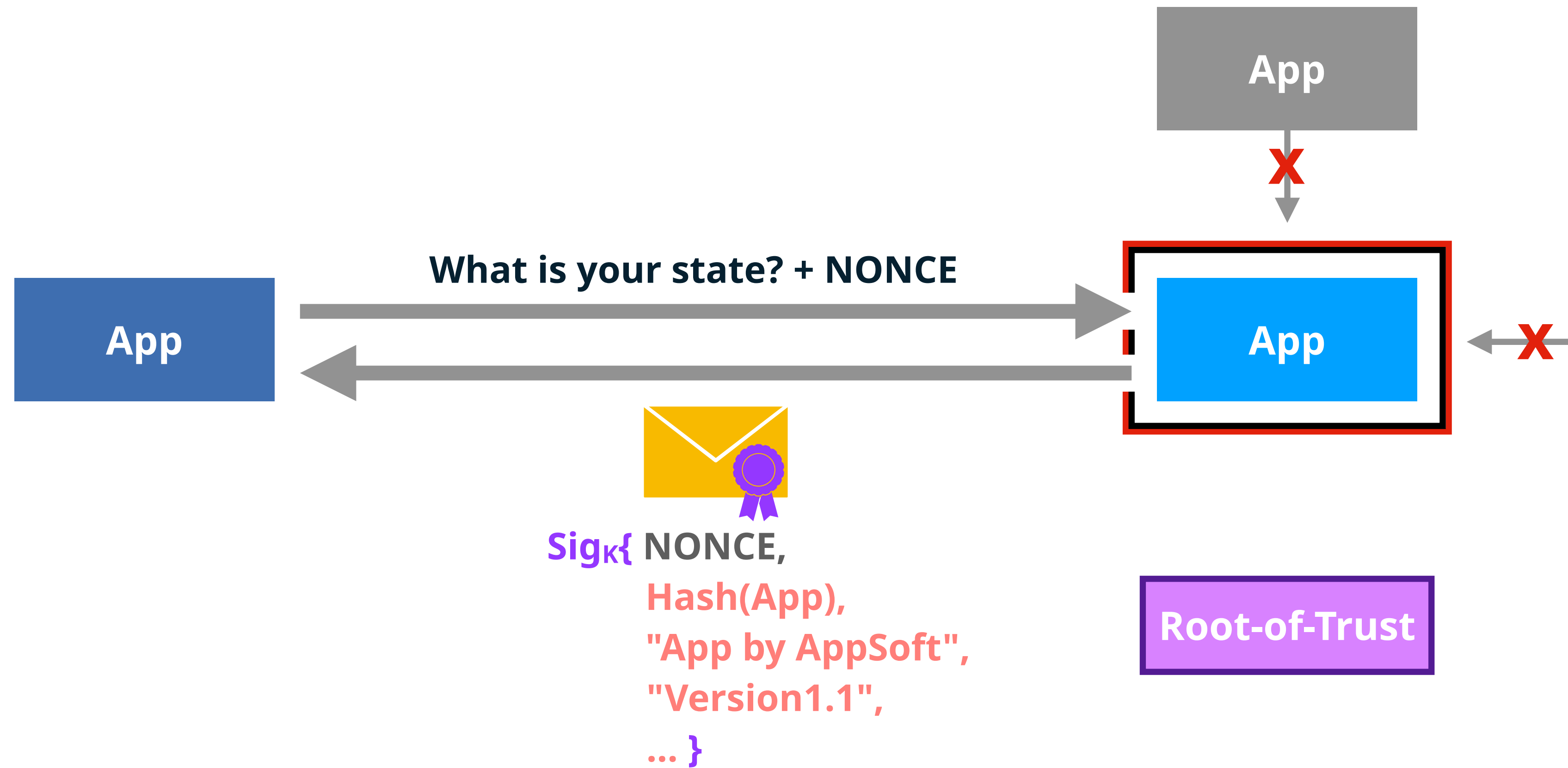
# Problems with Current TEE Implementations



- TEE and ISA cannot be chosen independently
- TEE implementation deeply integrated in core microarchitecture
- TEEs lack "good" integration with system software

# The Case for Modular TEEs

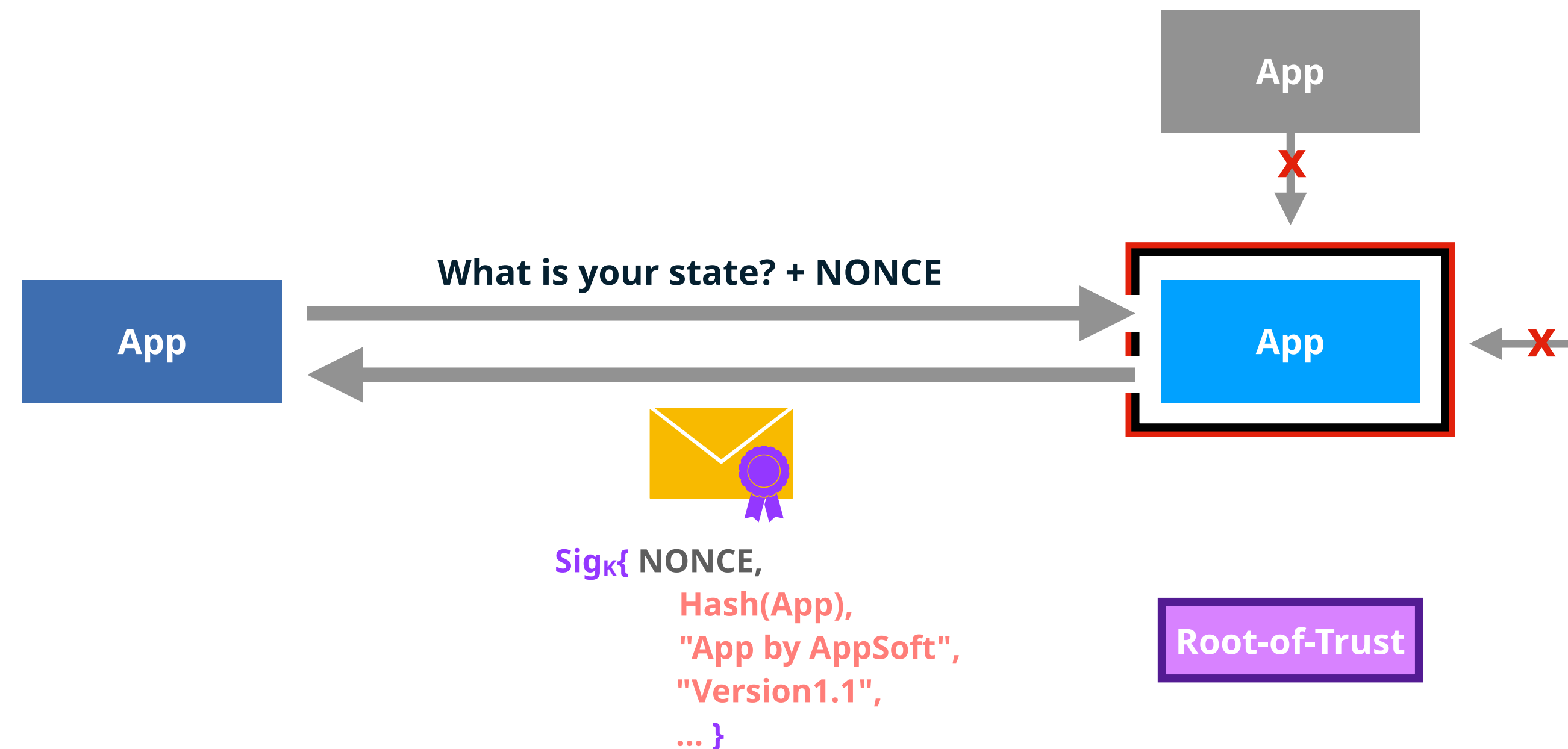
# What is a Trusted Execution Environment?



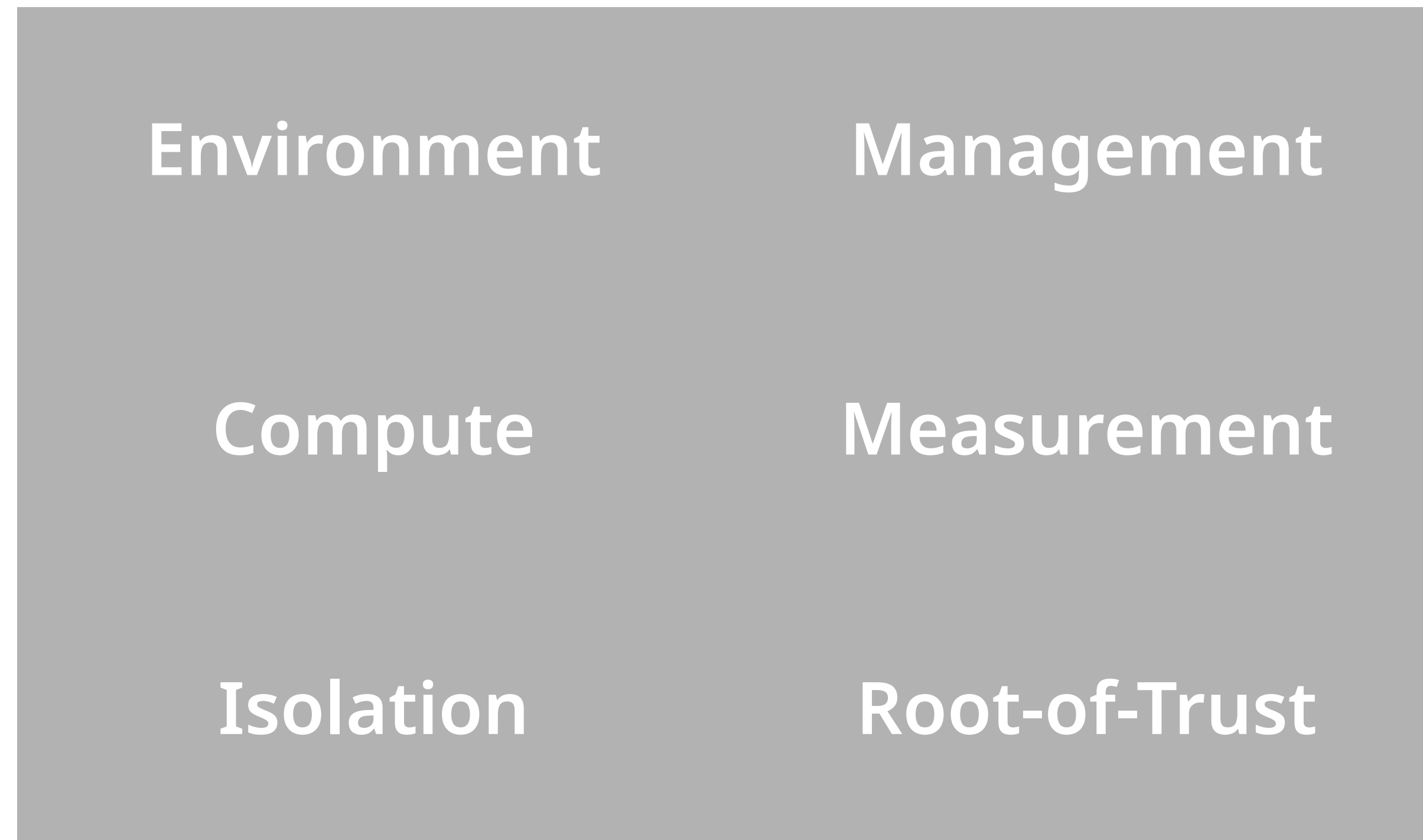
# Six Concerns for TEE Design and Implementation



- Computation
- Measurement
- Root of Trust
- Isolation
- Management
- Environment

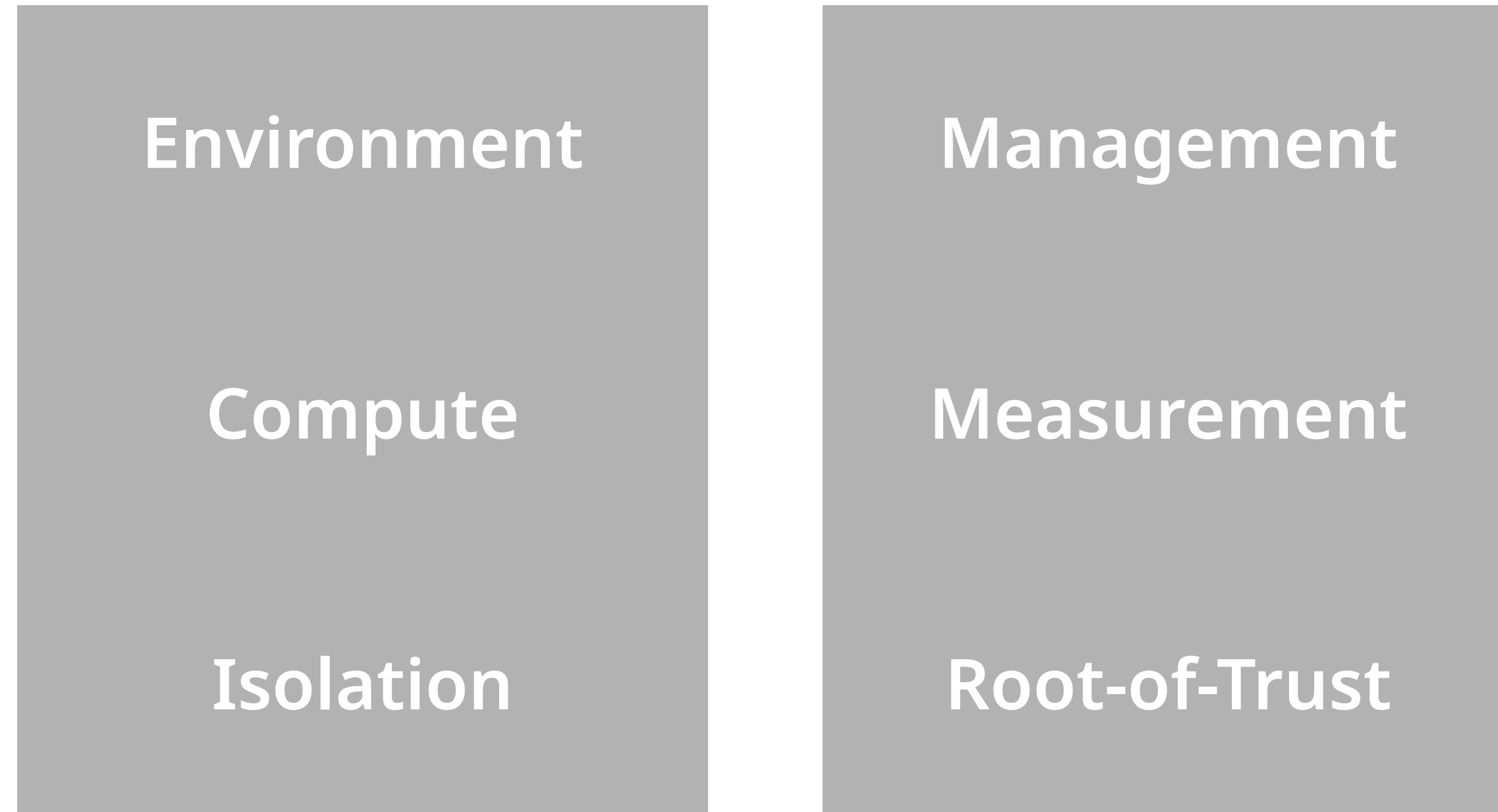


# Breaking up TEE Design and Implementation





# Breaking up TEE Design and Implementation



# Breaking up TEE Design and Implementation



Environment

Management

Compute

Measurement

Isolation

Root-of-Trust

# Modularize TEE Design and Implementation



Environment

Management

Compute

Measurement

Isolation

Root-of-Trust

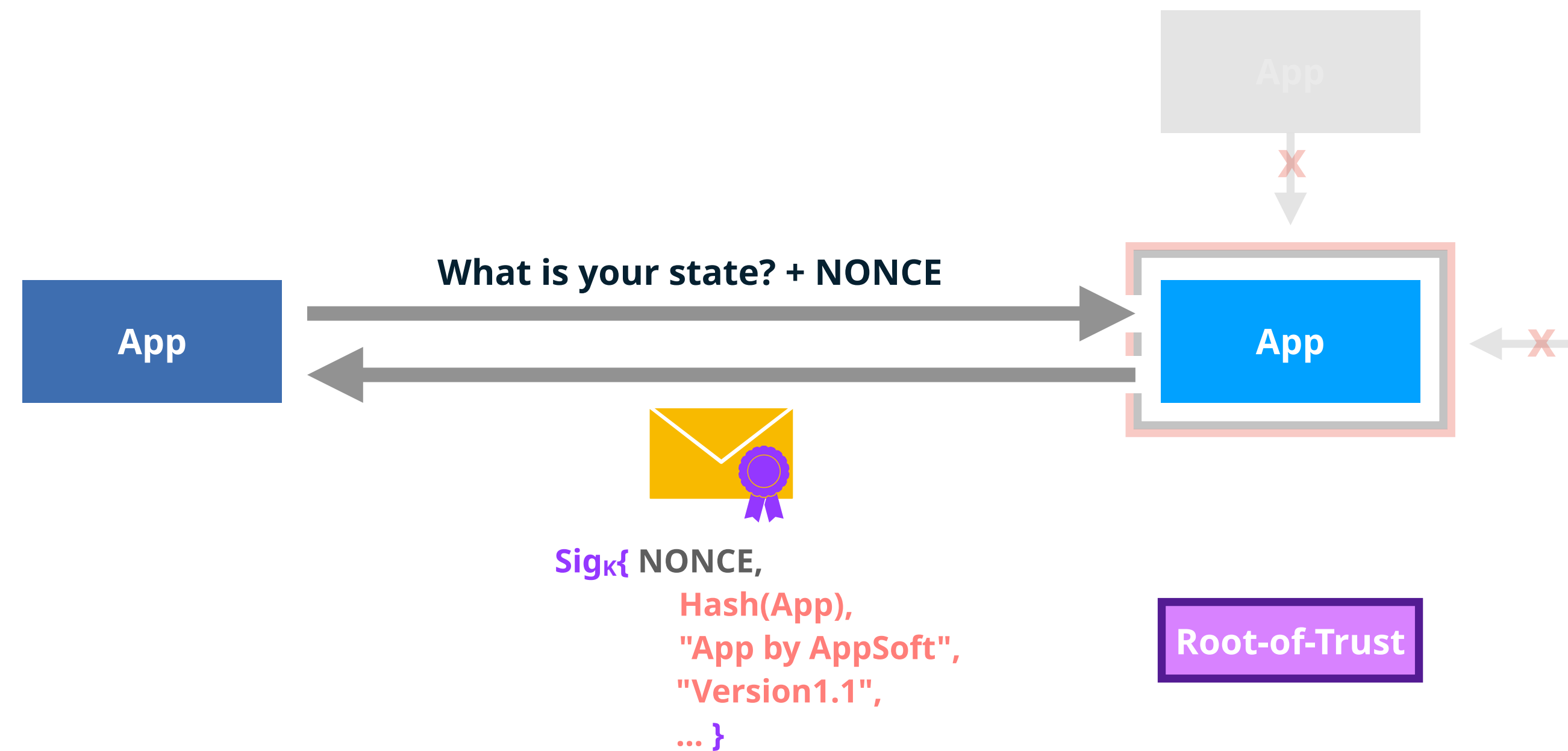
**Towards Modular Trusted Execution Environments,**  
Carsten Weinhold, Nils Asmussen, Diana Göhringer, Michael Roitzsch,  
*6th Workshop on System Software for Trusted Execution (SysTEX), 2023*

# Secure Communication between TEEs

# Six Concerns for TEE Design and Implementation



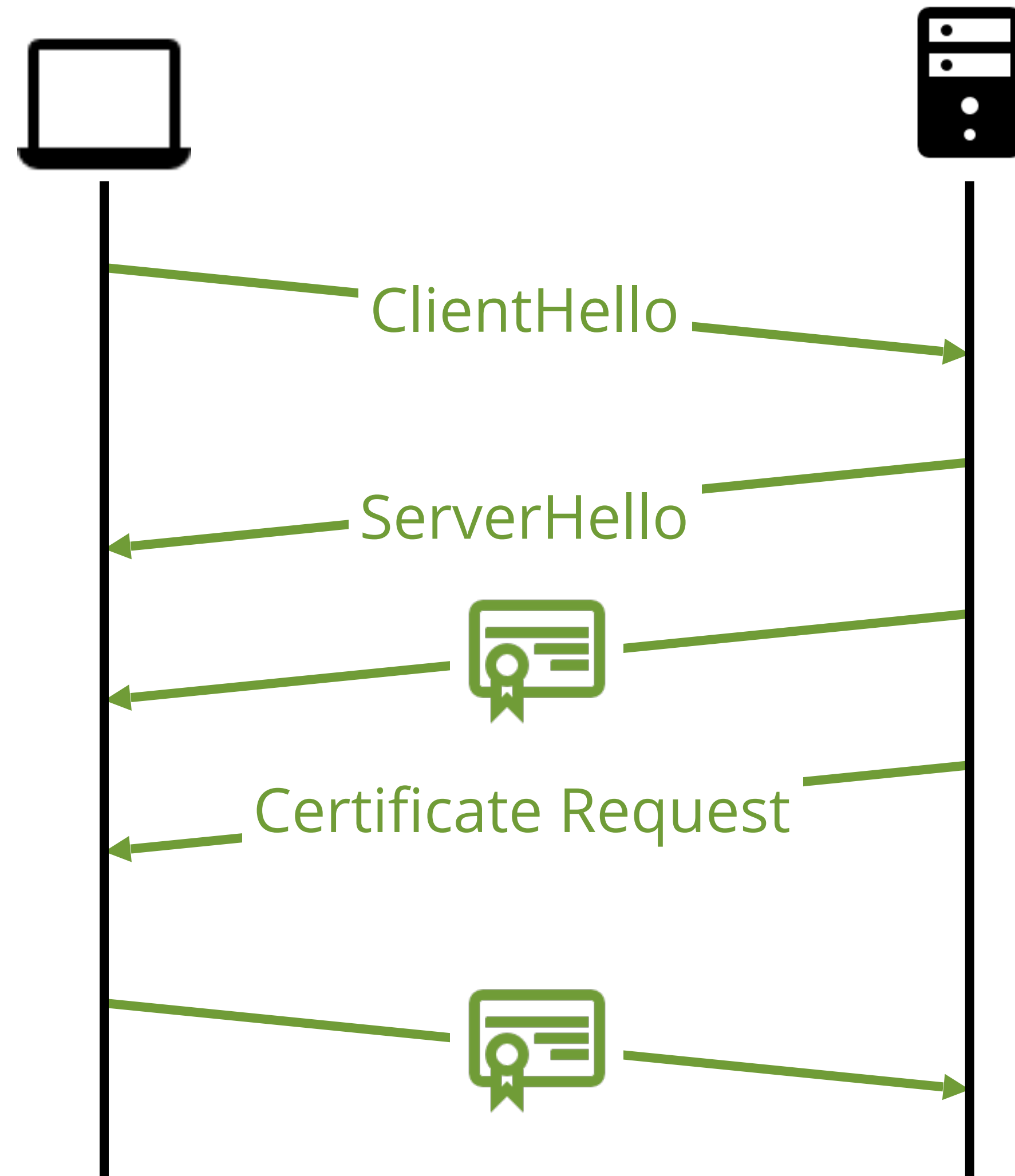
- Computation
- Measurement
- Root of Trust
- Isolation
- Management
- Environment + Communication



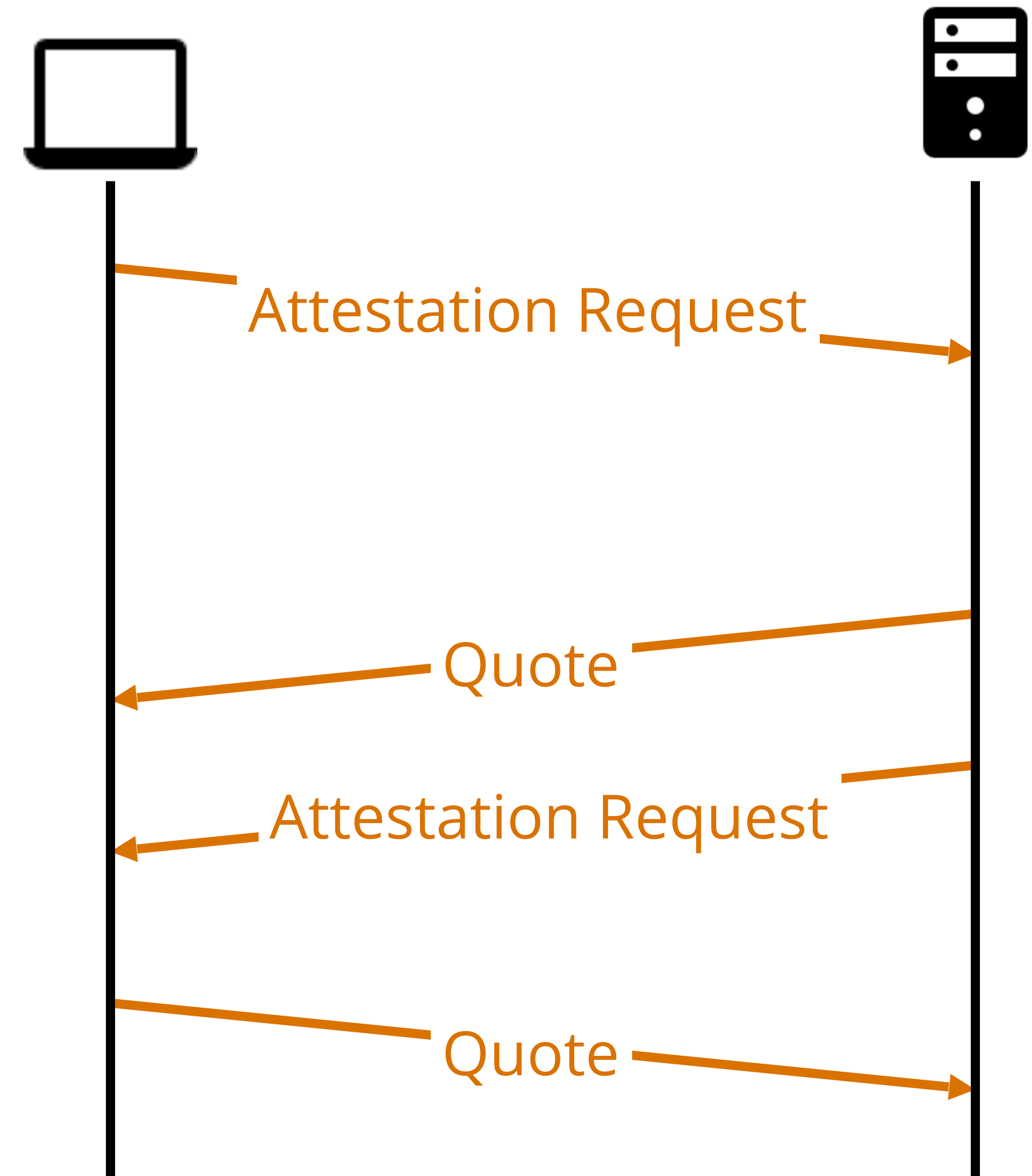
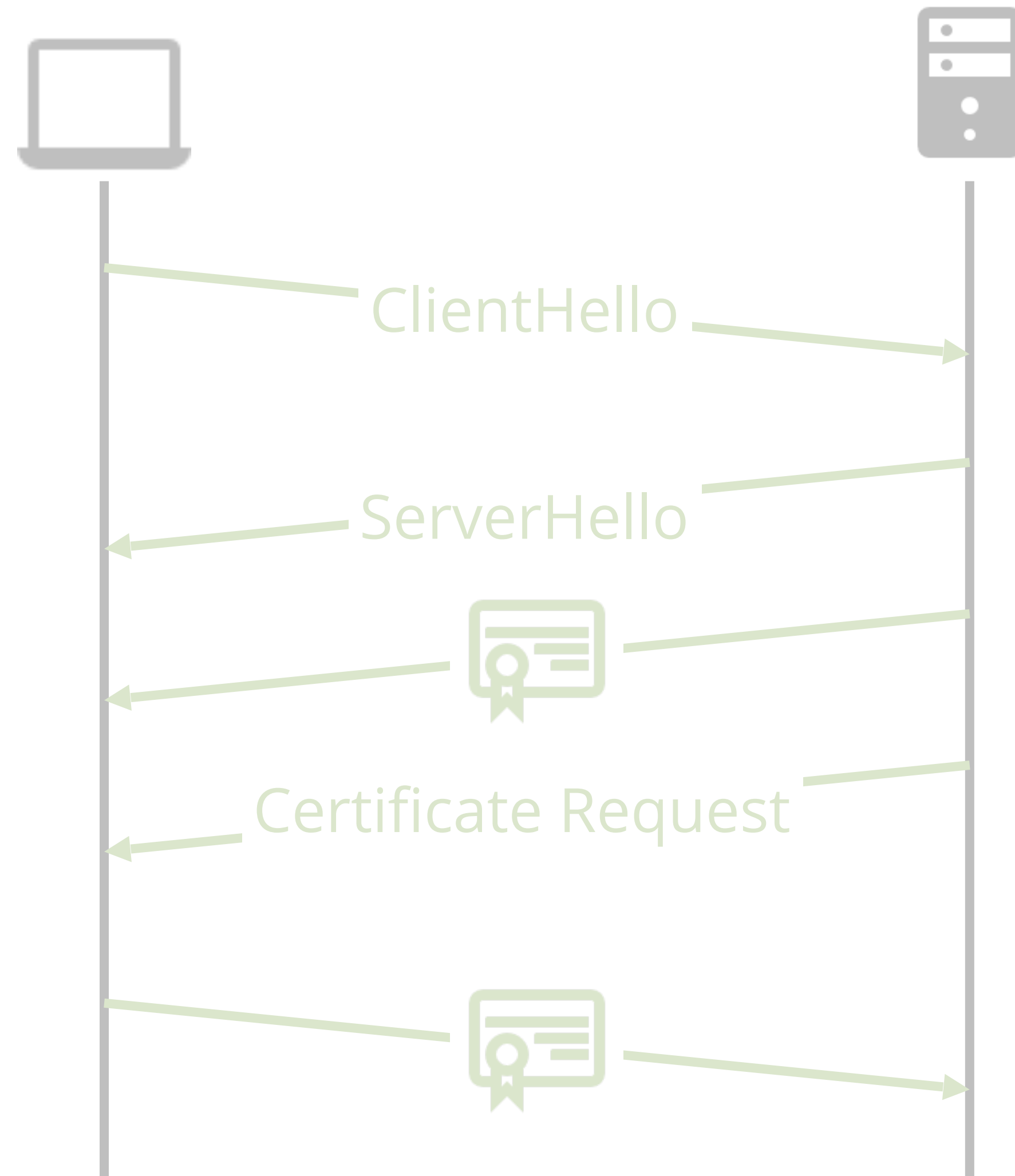
# Secure Communication between TEEs



# Handshake: Transport Layer Security (TLS)

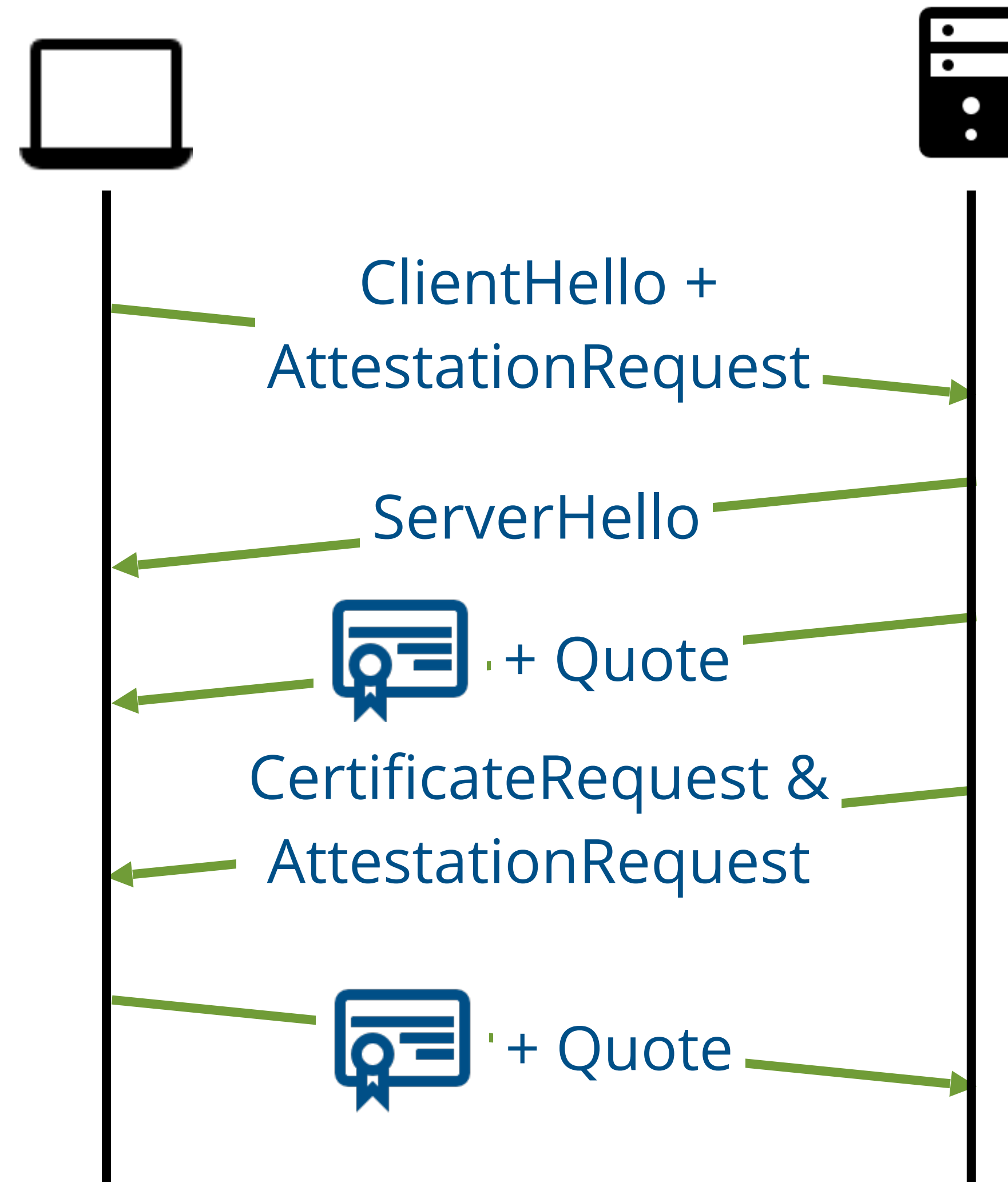


# Handshake: Remote Attestation (RA)

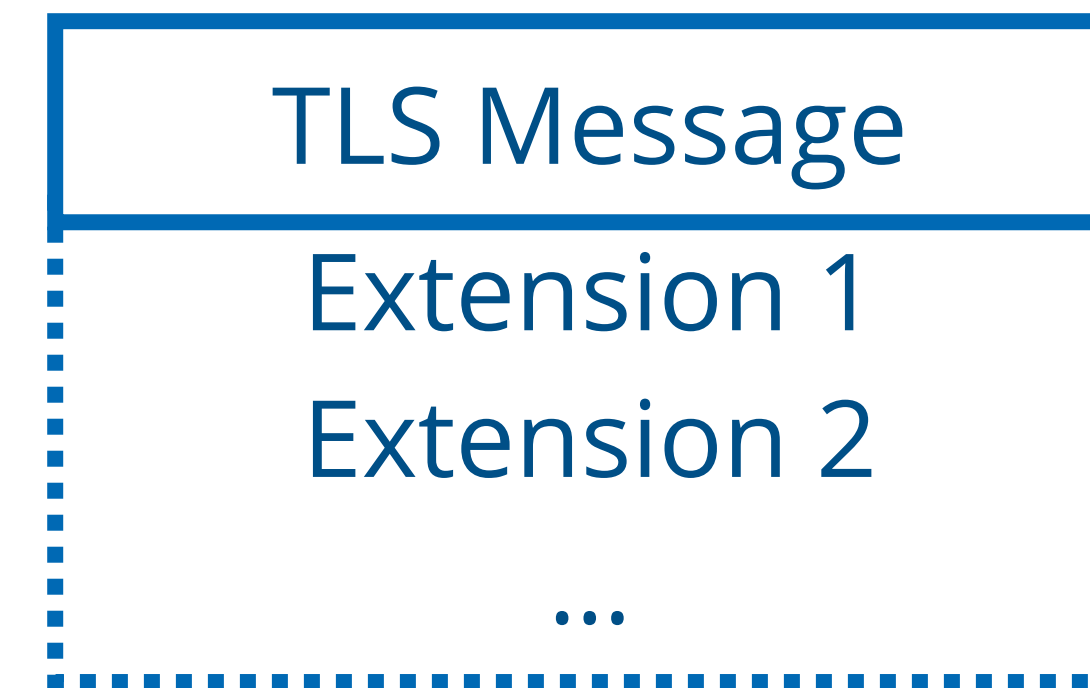
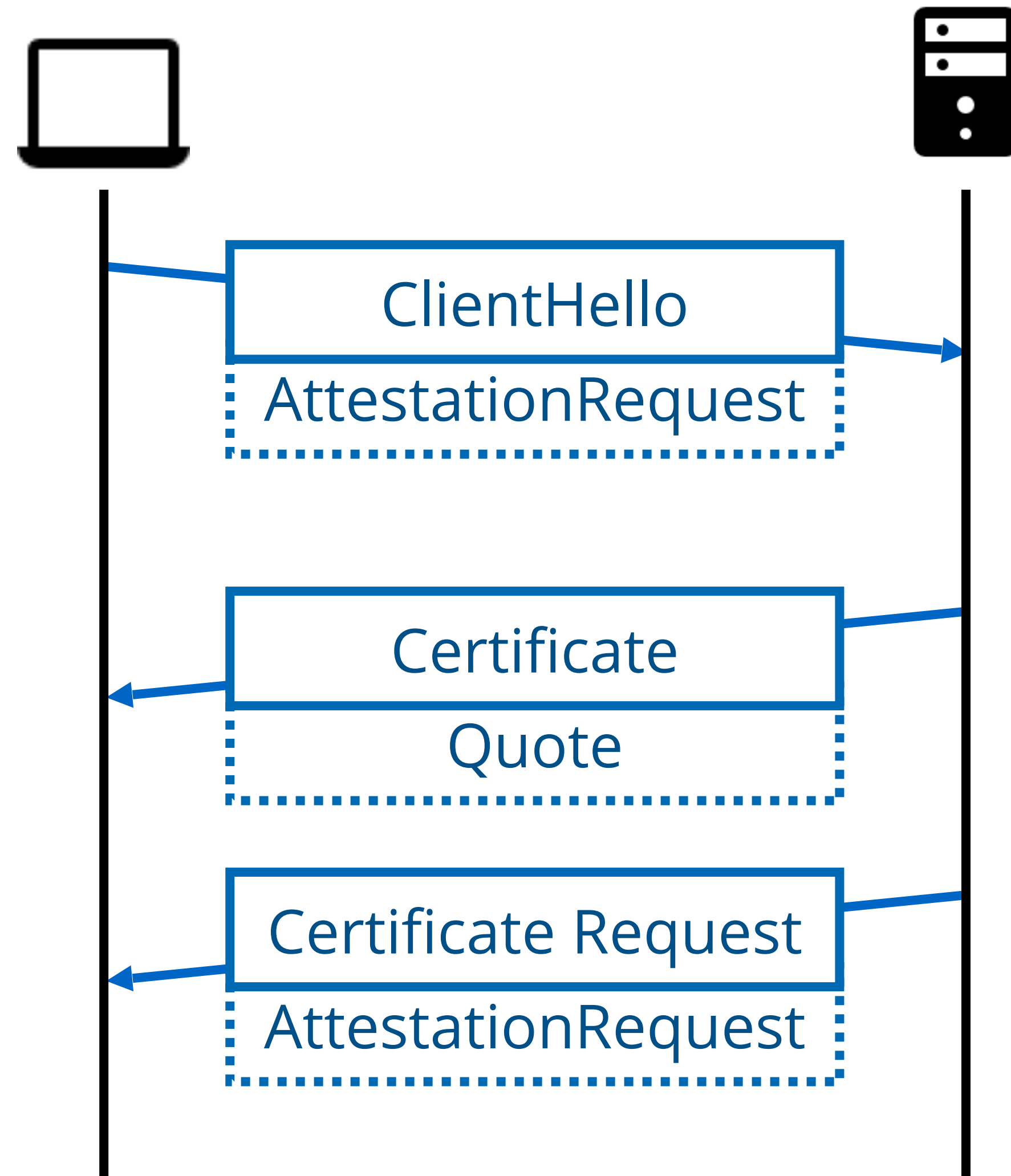
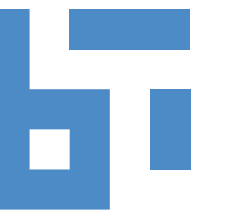




# Handshake: TLS+RA



# RA piggy-backed in TLS message extensions



**RATLS: Integrating Transport Layer Security with Remote Attestation**,  
Robert Walther, Carsten Weinhold, Michael Roitzsch,  
4th Workshop on Cloud Security and Privacy (Cloud S&P), 2022

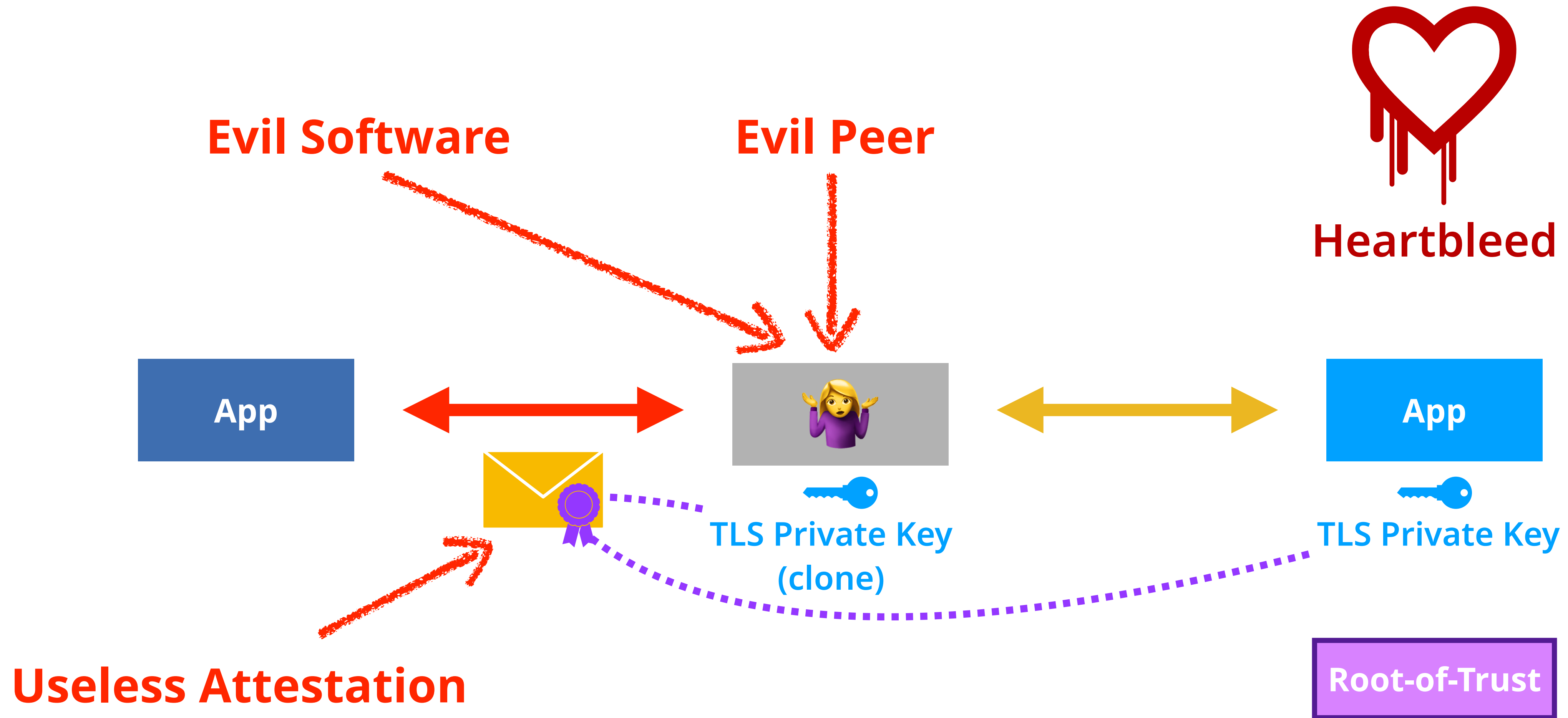
**Using Attestation in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)**,  
Hannes Tschofenig, Yaron Sheffer, Paul Howard, Ionuț Mihalcea, Yogesh Deshpande,  
Arto Niemi, Thomas Fossati,  
IETF Draft, last updated 2024-03-19,  
<https://datatracker.ietf.org/doc/draft-fossati-tls-attestation/>

# Combining TLS and RA



	No added round trips	No added encryption
HTTTPA		
Platform Certificate	✓	✓
RA-TLS	✓	✓
RA-TLS with CA		✓
Extending TLS		✓
RATLS	✓	✓
Trusted Channels	✓	✓

# Relay Attacks



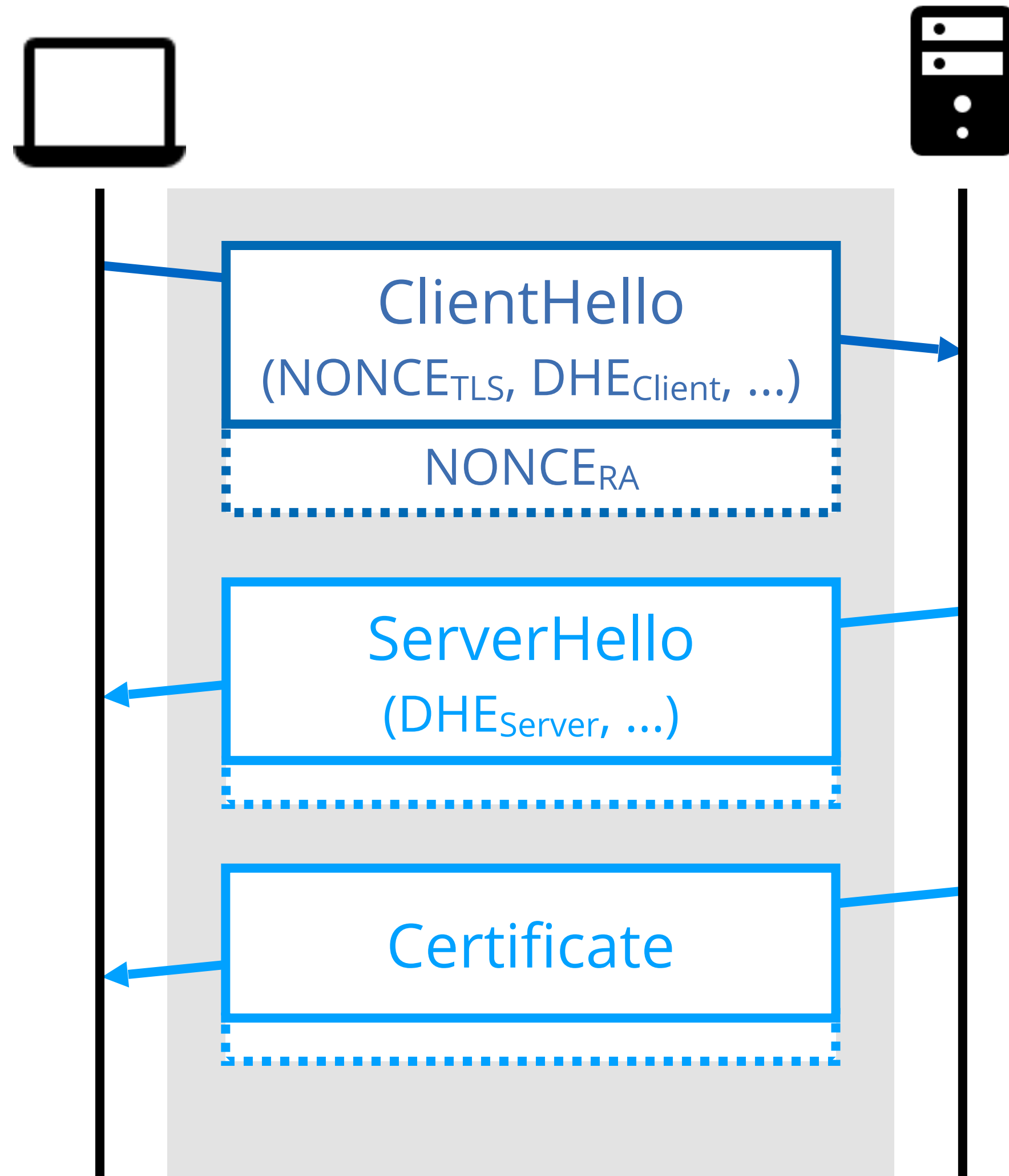
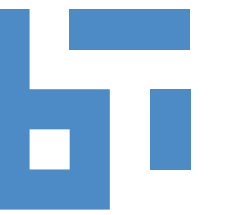
**Compromising TLS "breaks" Attestation, too!**

# Combining TLS and RA



	No added round trips	No added encryption	Prevents Relay Attacks	Independent failure
HTTPA			✓	✓
Platform Certificate	✓	✓	✓	
RA-TLS	✓	✓	✓	
RA-TLS with CA		✓	✓	
Extending TLS		✓	✓	✓
RATLS	✓	✓	✓	
Trusted Channels	✓	✓	✓	

# TLS+RA: DHE Shared Secret and Handshake Transcript

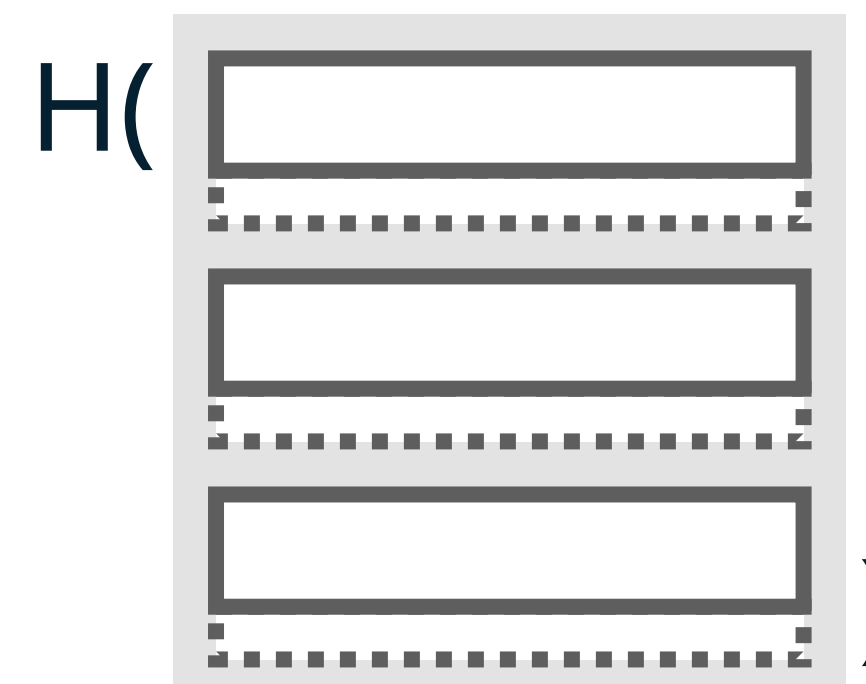


## Ephemeral Diffie-Hellman Key Exchange:

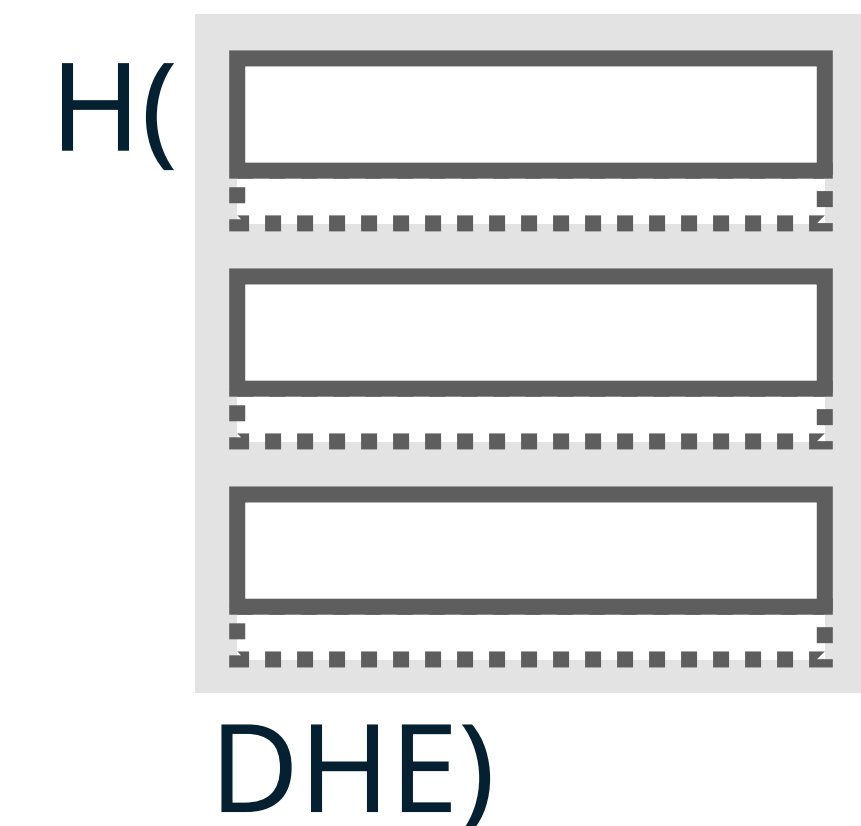
**Client:**  $DHE := DHE_{\text{Client\_private}} \cdot DHE_{\text{Server}}$

**Server:**  $DHE := DHE_{\text{Server\_private}} \cdot DHE_{\text{Client}}$

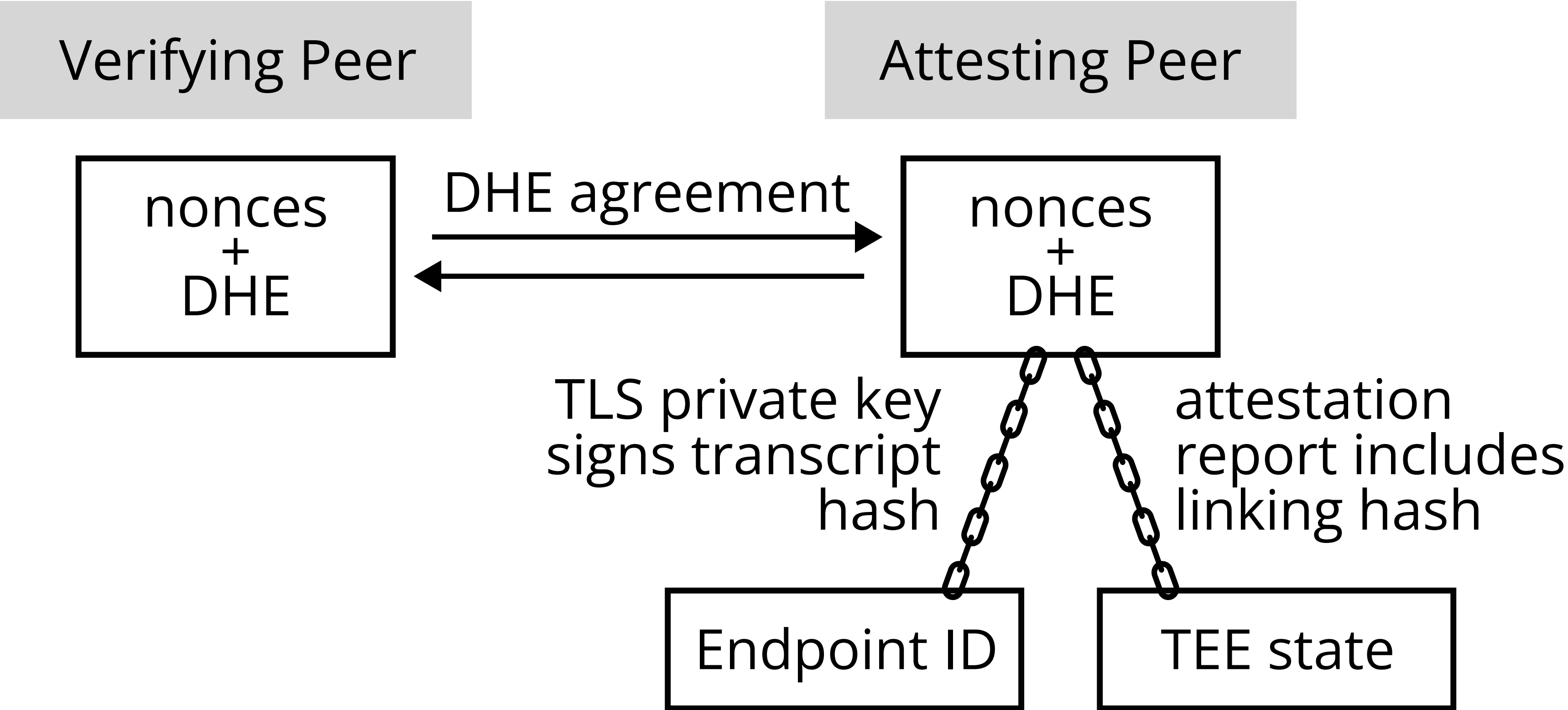
## Transcript hash:



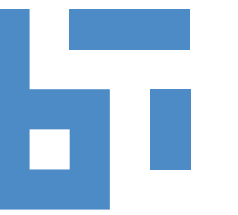
## Linking hash:



# TLS+RA: Additive Security



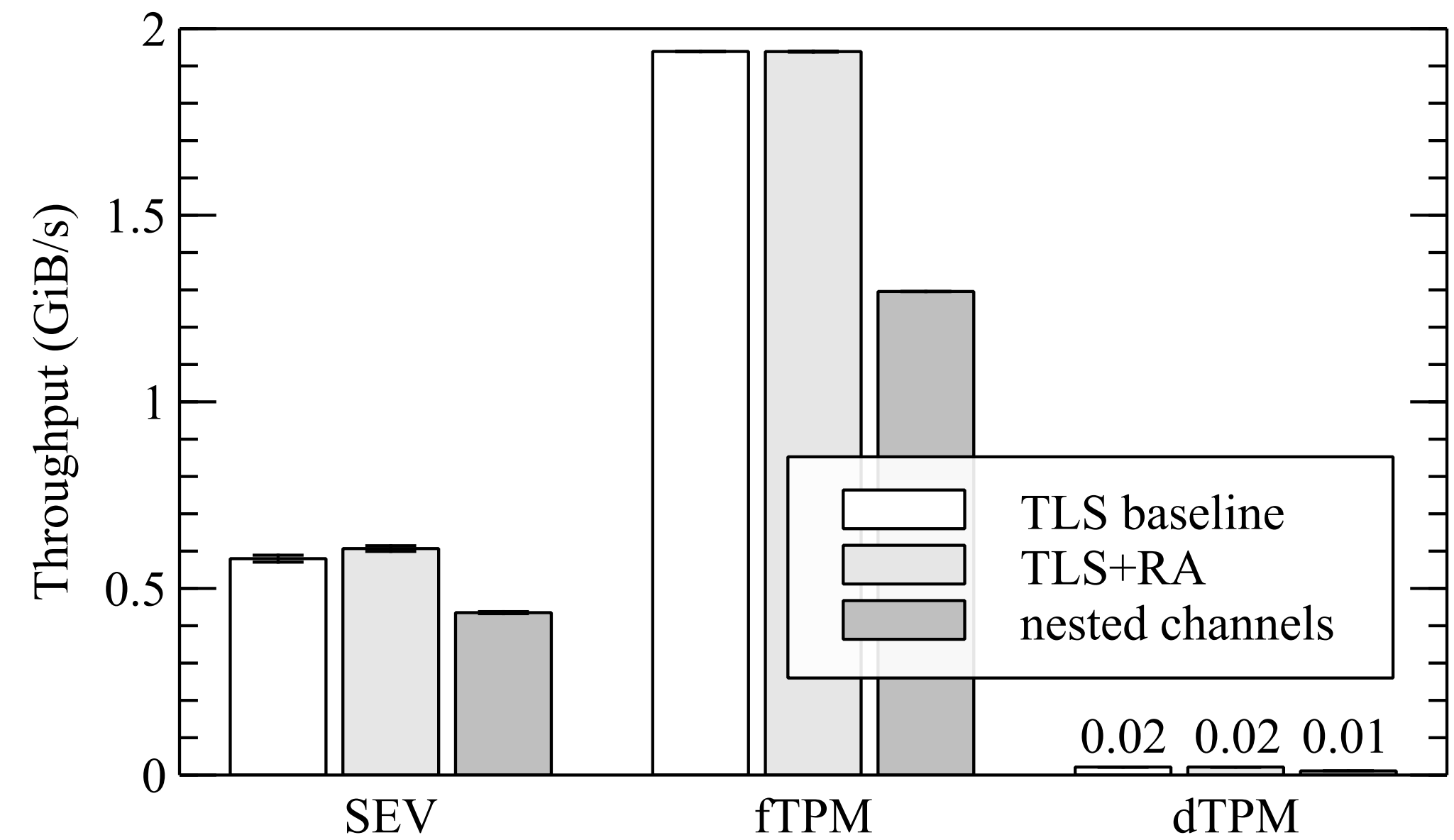
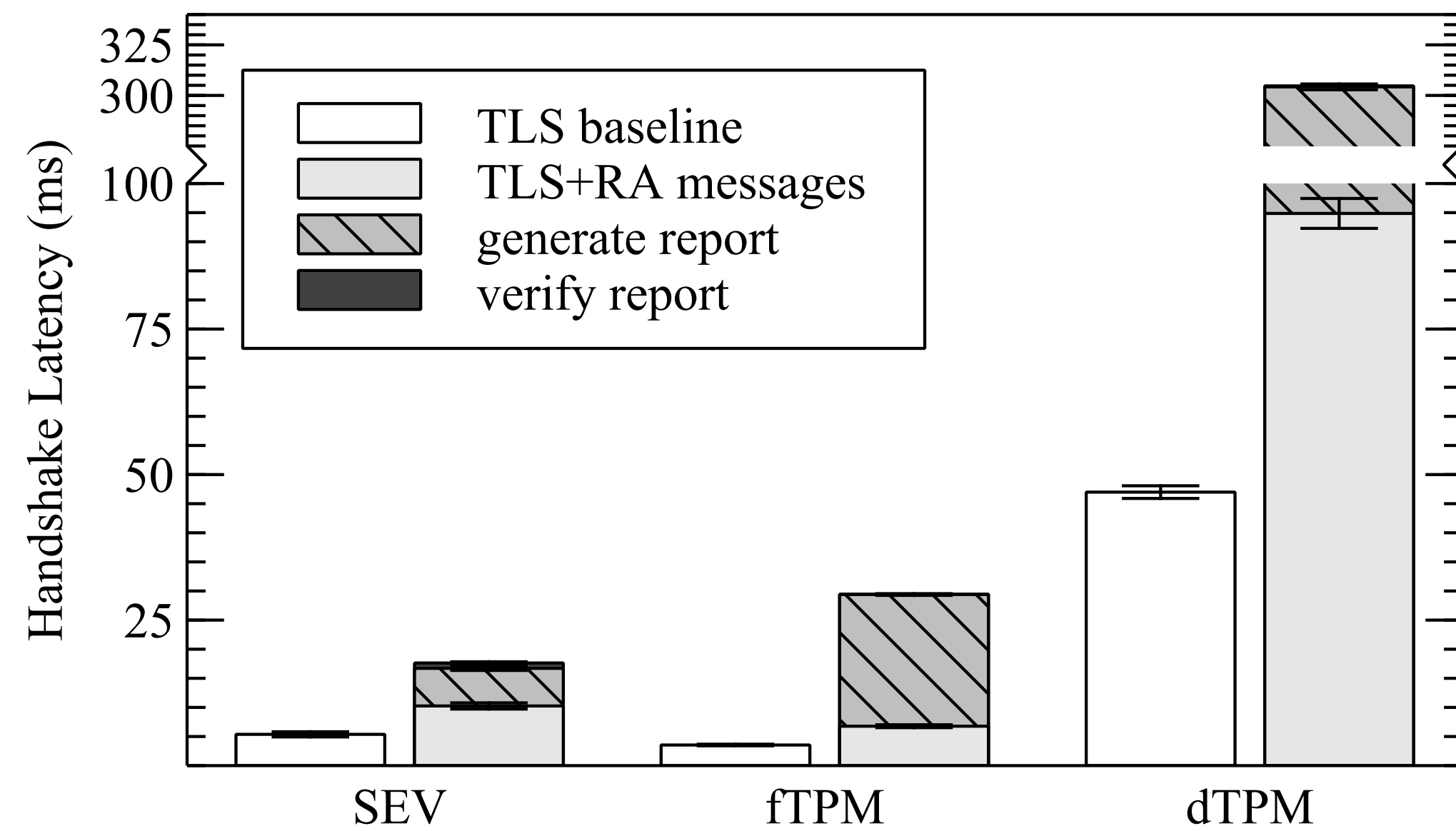
# Combining TLS and RA



	No added round trips	No added encryption	Prevents Relay Attacks	Independent failure	Independent deployment
HTTPA			✓	✓	✓
Platform Certificate	✓	✓	✓		
RA-TLS	✓	✓	✓		
RA-TLS with CA		✓	✓		
Extending TLS		✓	✓	✓	
RATLS	✓	✓	✓		✓
Trusted Channels	✓	✓	✓		✓
<b>TLS+RA</b>	✓	✓	✓	✓	✓



# TLS+RA: Performance



# Summary

- **TEEs** should be modular
- **TLS+RA** is modular, too:
  - Message extensions
  - Independent of root-of-trust
  - Independent deployment
  - Independent failure
- **TLS+RA** provides **additive security**



CORENEXT

