Teaching an Old Dog New Tricks:

Verifiable FHE Using Commodity Hardware

Jules Drean Fisher Jepsen Edward Suh Aamer Jaleel Gururaj Saileshwar

MIT NVIDIA UNIVERSITY OF TORONTO

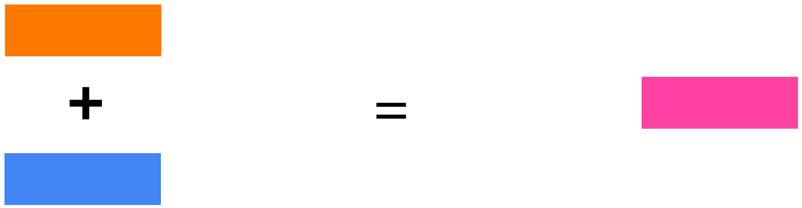
What is Fully Homomorphic Encryption (FHE)?

1000

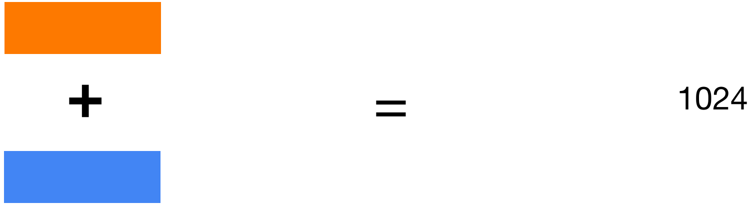
24

What is Fully Homomorphic Encryption (FHE)?

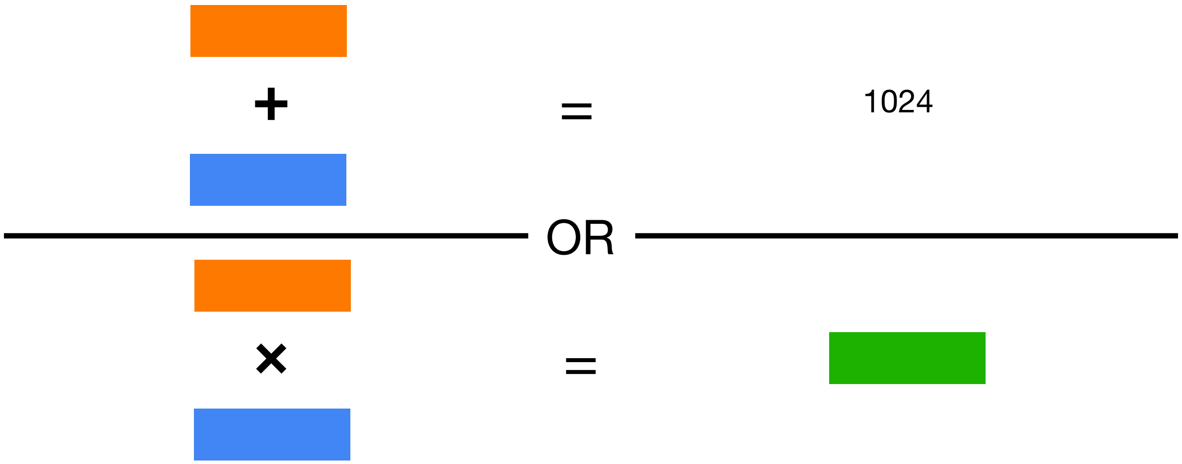
What is Fully Homomorphic Encryption (FHE)?



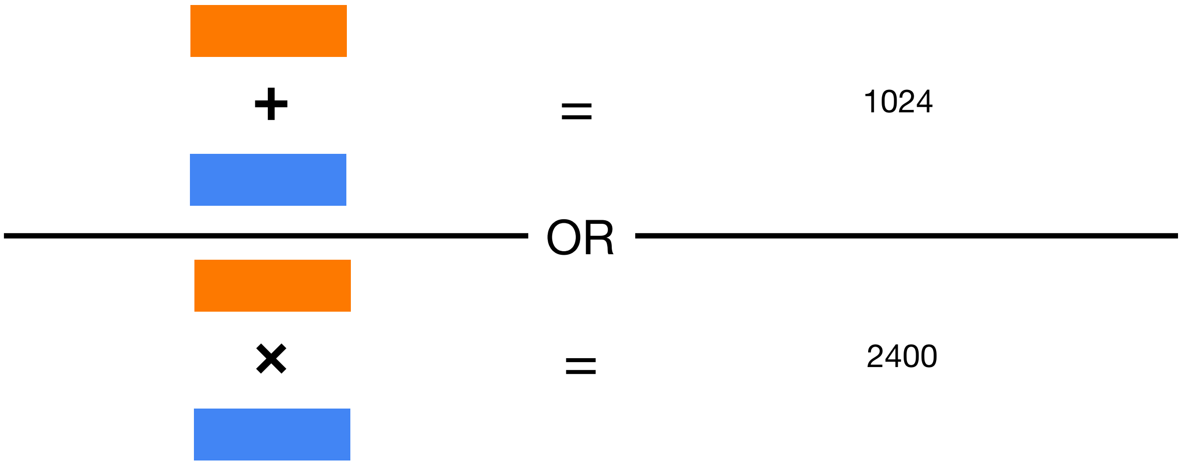
What is Fully Homomorphic Encryption (FHE)?



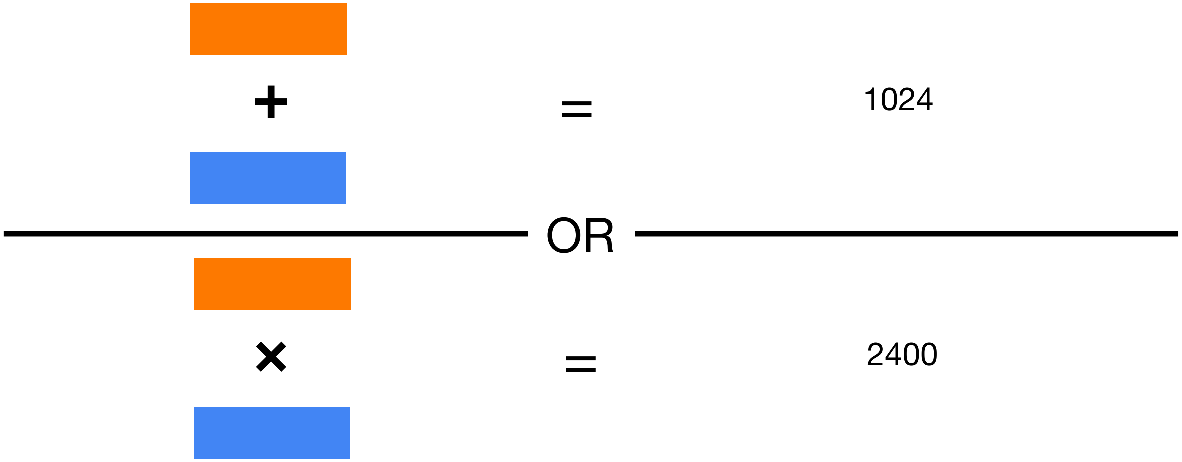
What is Fully Homomorphic Encryption (FHE)?



What is Fully Homomorphic Encryption (FHE)?



What is Fully Homomorphic Encryption (FHE)?



⚠ Significant performance overheads
(3 to 6 orders of magnitude slowdown)

FHE Applications

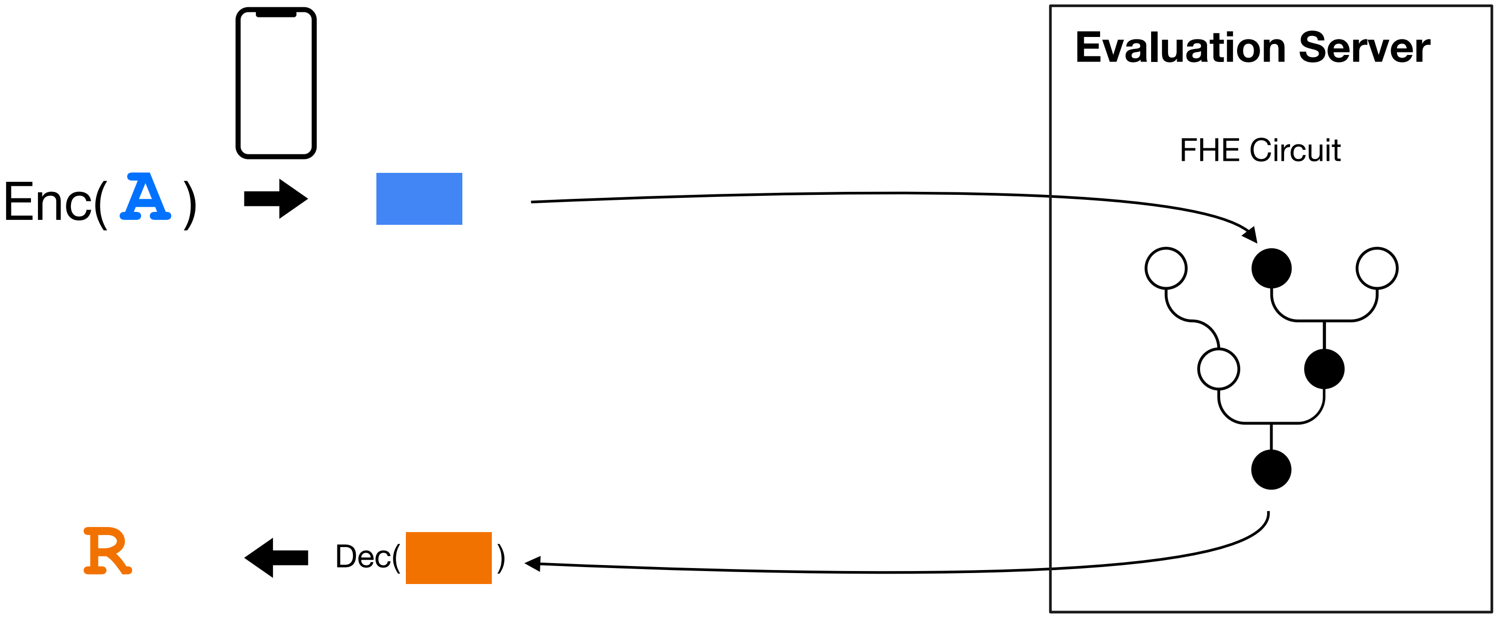
Build other interesting cryptographic primitives

- Private set intersection (PSI)
- Private information retrieval (PIR)
- Multiparty computation (MPC)

Many real-world applications:

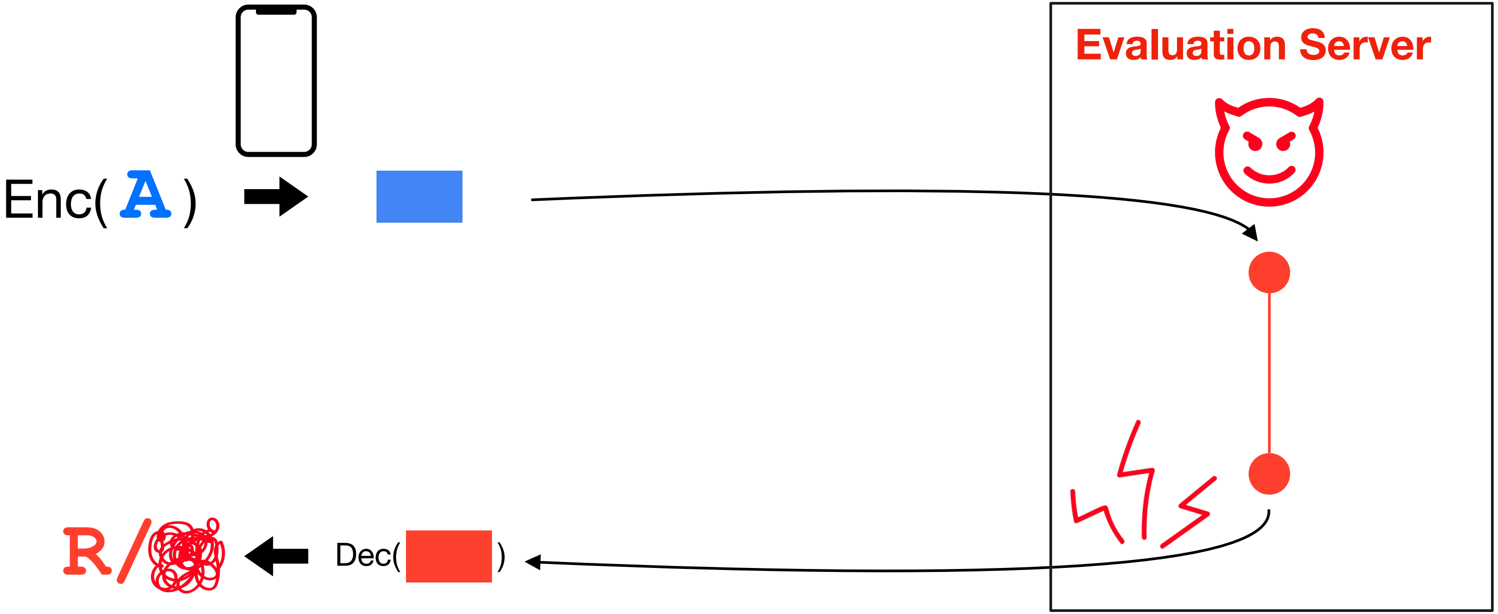
- Confidential Credit Scoring
- Biometric Identification
- Blockchain Transactions
- Medicine and Healthcare
- Private Machine Learning

FHE in the client-server setup



Problem: No Integrity!

FHE schemes are only **passively** secure



If client reaction can be observed

⇒ **Key recovery attacks!!**

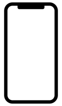
Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

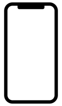
Allows a remote user to verify the results of a computation



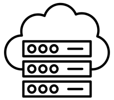
[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation



Remote Client

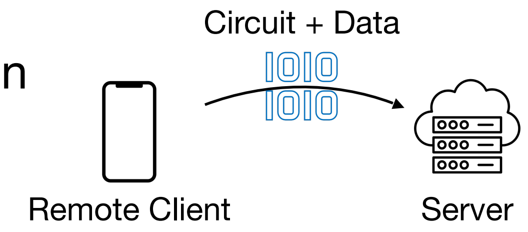


Server

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

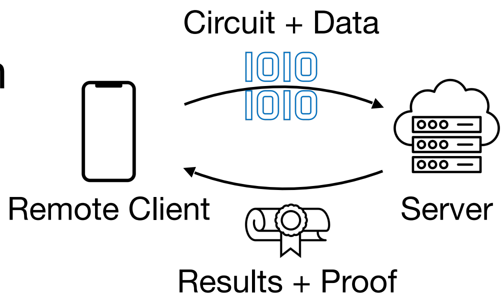
Allows a remote user to verify the results of a computation



[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation

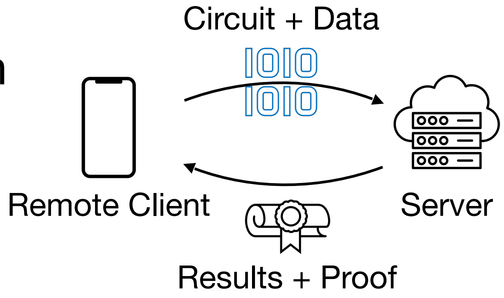


[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation

Unfortunately, existing solutions are insufficient…



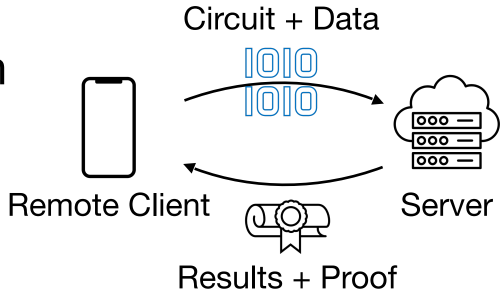
[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation

Unfortunately, existing solutions are insufficient…

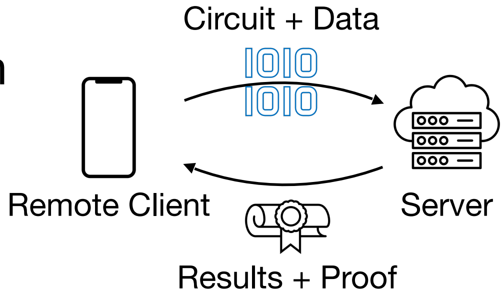
Cryptographic solutions



[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation



Unfortunately, existing solutions are insufficient…

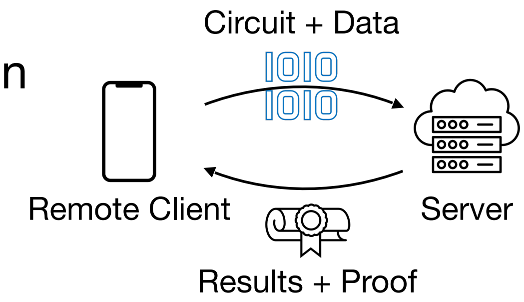
Cryptographic solutions

Zero Knowledge Proofs[0]**:** Between 5-6 order of magnitude overhead!

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation



Unfortunately, existing solutions are insufficient…

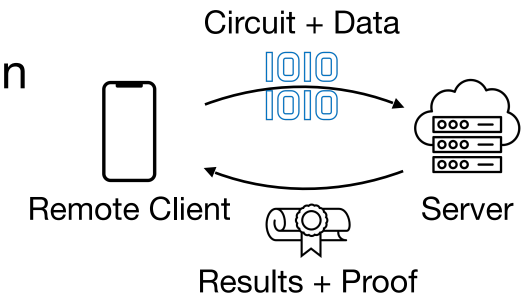
Cryptographic solutions

- **Zero Knowledge Proofs[0]:** Between **5-6 order of magnitude** overhead!
- **Replication:** Hard to achieve **distributed trust** +1-2 order of magnitude overhead

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation



Unfortunately, existing solutions are insufficient…

Cryptographic solutions

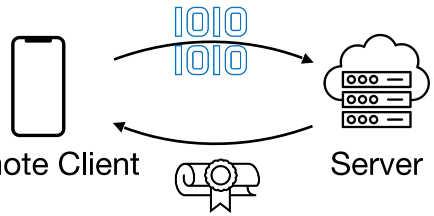
- **Zero Knowledge Proofs**[0]: Between **5-6 order of magnitude** overhead!
- **Replication:** Hard to achieve **distributed trust** +1-2 order of magnitude overhead

Trusted Hardware Solutions

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Solution: Add a Layer of Verifiable Computation

Allows a remote user to verify the results of a computation



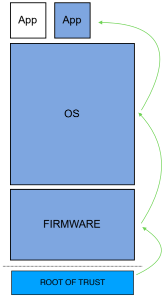
Unfortunately, existing solutions are insufficient…

Cryptographic solutions

- **Zero Knowledge Proofs**[0]: Between **5-6 order of magnitude** overhead!
- **Replication:** Hard to achieve **distributed trust** +1-2 order of magnitude overhead

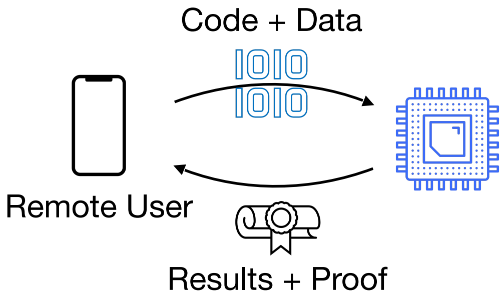
Trusted Hardware Solutions

- **Trusted execution environments or trusted enclaves:**
  Based on Remote attestation

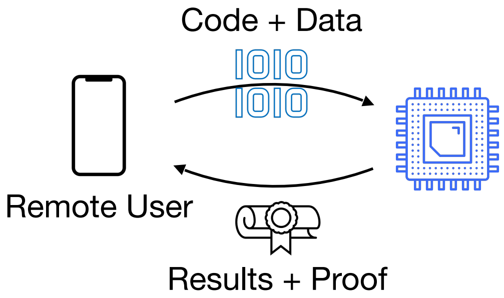


[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Remote attestation

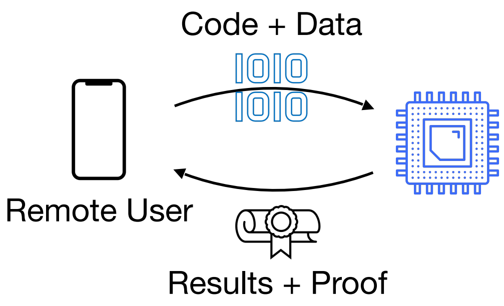


Remote attestation



Leverages Trusted Hardware

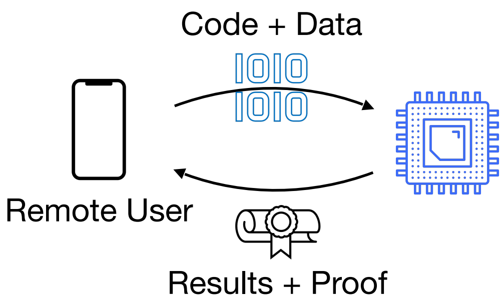
Remote attestation



Leverages Trusted Hardware

- Measure (hash) the binary and data inside the enclave

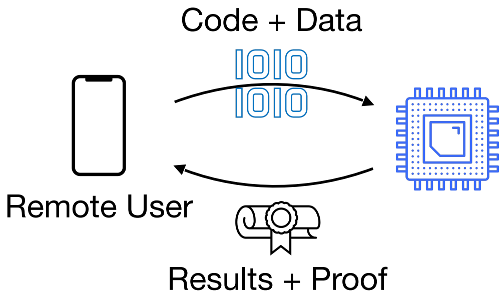
Remote attestation



Leverages Trusted Hardware

- Measure (hash) the binary and data inside the enclave
- Creates an *attestation report*

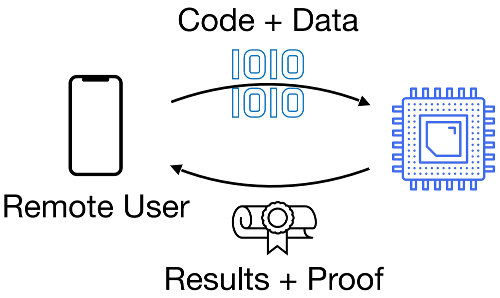
Remote attestation



Leverages Trusted Hardware

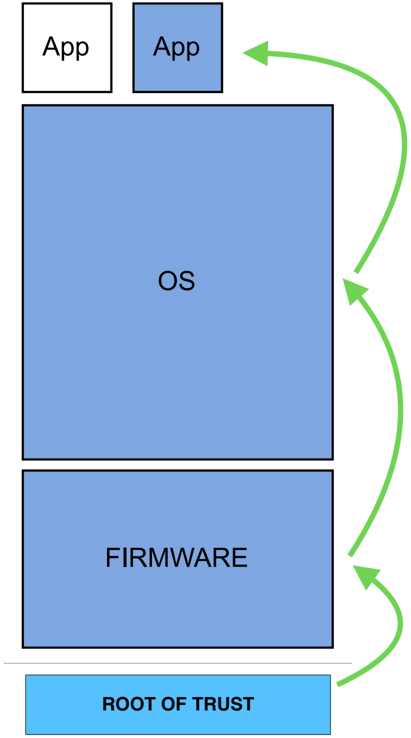
- Measure (hash) the binary and data inside the enclave
- Creates an *attestation report*
- Signs and sends it to the remote user as an attestation proof

Remote attestation

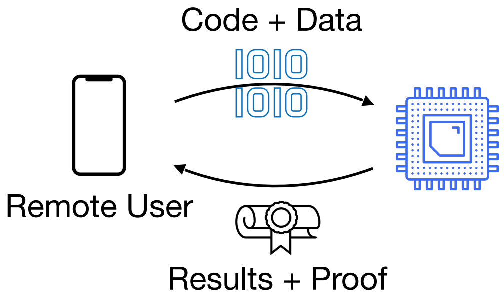


Leverages Trusted Hardware

- Measure (hash) the binary and data inside the enclave
- Creates an *attestation report*
- Signs and sends it to the remote user as an attestation proof
- Based on a root-of-trust and a chain-of-trust

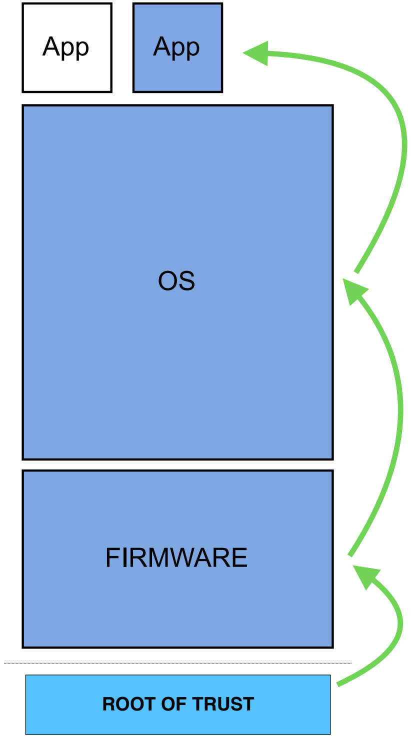


Remote attestation

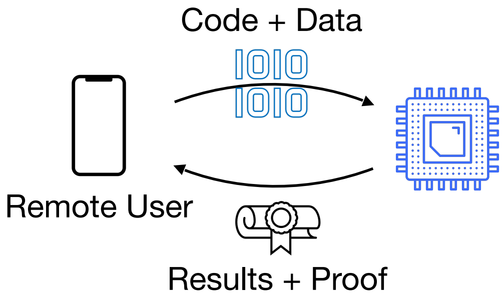


Leverages Trusted Hardware

- Measure (hash) the binary and data inside the enclave
- Creates an *attestation report*
- Signs and sends it to the remote user as an attestation proof
- Based on a root-of-trust and a chain-of-trust
- No performance overhead beyond fix setup cost



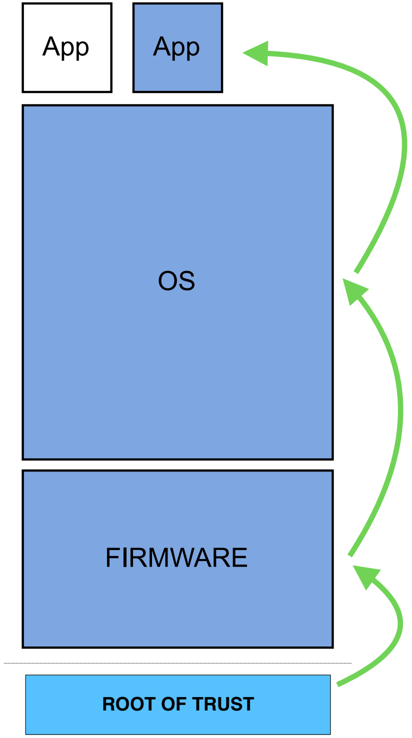
Remote attestation



Leverages Trusted Hardware

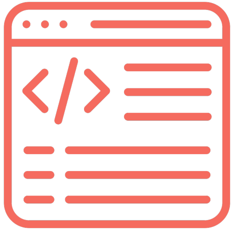
- Measure (hash) the binary and data inside the enclave
- Creates an *attestation report*
- Signs and sends it to the remote user as an attestation proof
- Based on a root-of-trust and a chain-of-trust
- No performance overhead beyond fix setup cost

Problem: vulnerable to microarchitectural side channels!

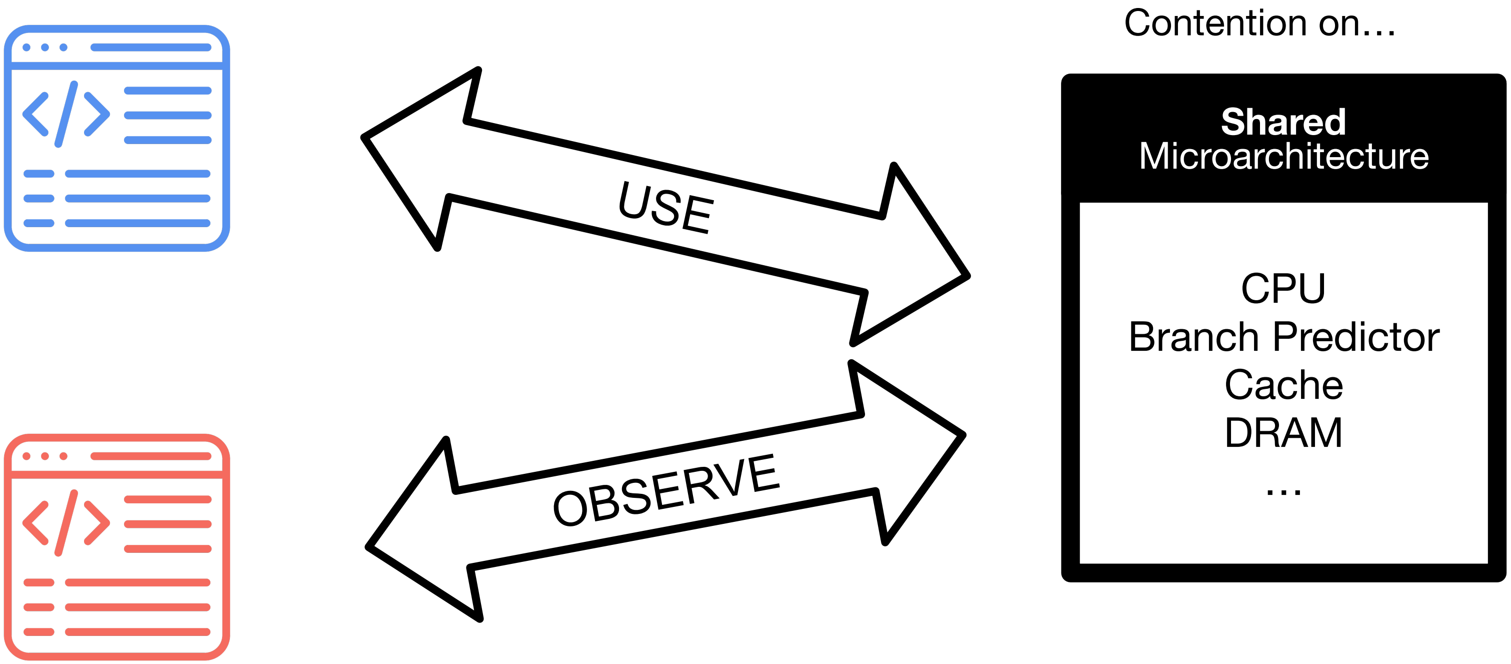


Microarchitectural Side Channels

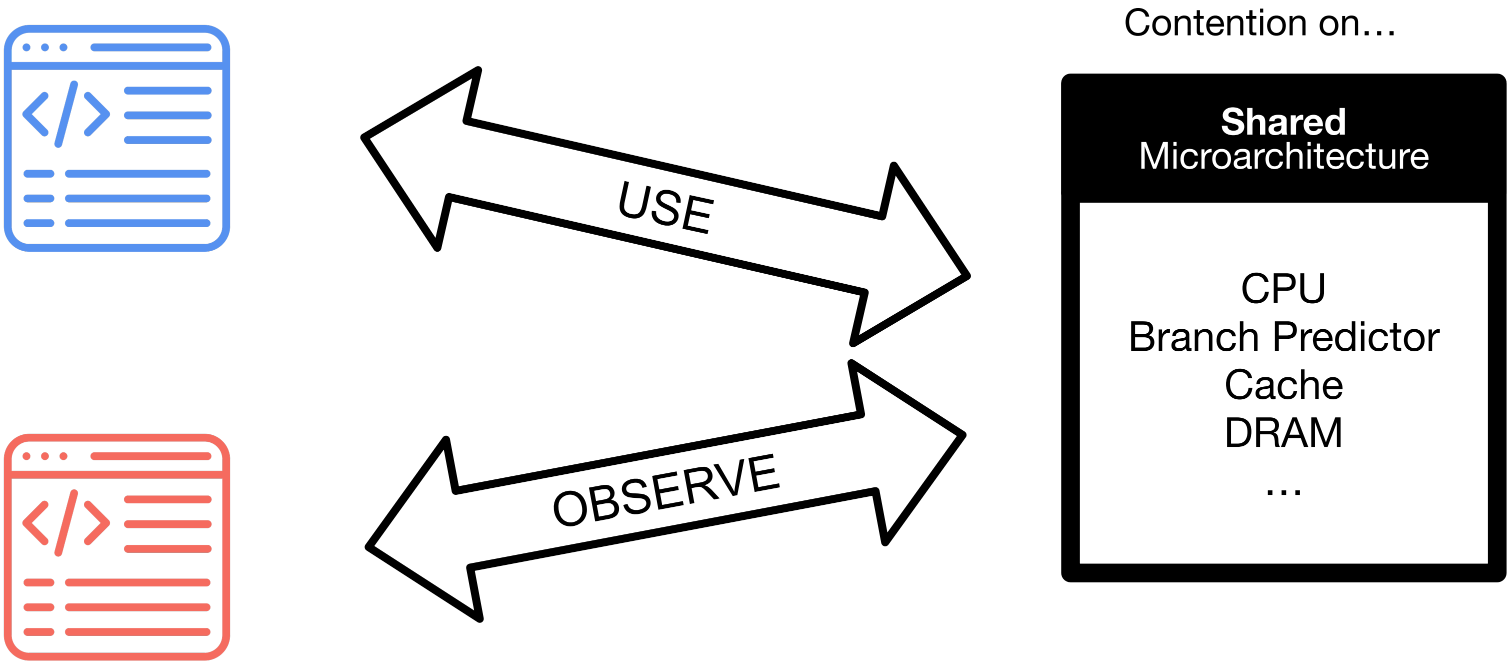
Microarchitectural Side Channels



Microarchitectural Side Channels



Microarchitectural Side Channels



Keep calm: just write good (constant-time)

But these attacks can get worse…

2018 - Spectre

…and the rest of the logo mafia

Transient execution attacks:

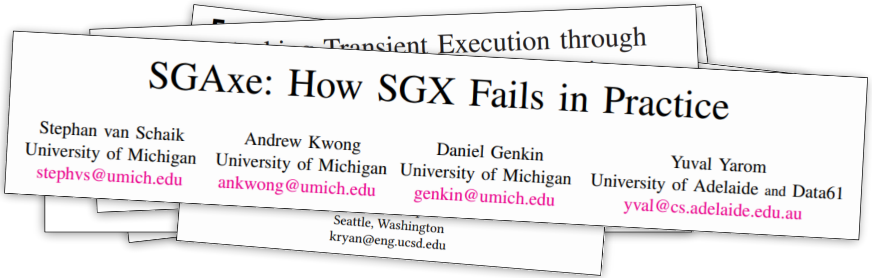
- use speculation to weaponize side channels
- break isolation of all existing TEEs

But these attacks can get worse…

2018 - Spectre

…and the rest of the logo mafia

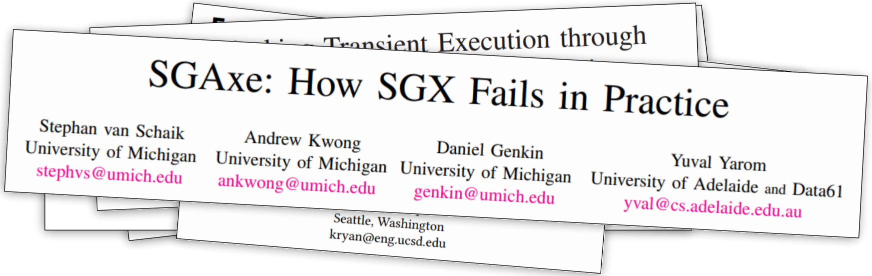
Transient execution attacks:
- use speculation to weaponize side channels
- break isolation of all existing TEEs



But these attacks can get worse…

2018 - Spectre

Transient execution attacks:
- use speculation to weaponize side channels
- break isolation of all existing TEEs


Transient Execution through

SGAxe: How SGX Fails in Practice

Stephan van Schaik, University of Michigan, stephvs@umich.edu
Andrew Kwong, University of Michigan, ankwong@umich.edu
Daniel Genkin, University of Michigan, genkin@umich.edu
Yuval Yarom, University of Adelaide and Data61, yval@cs.adelaide.edu.au

Seattle, Washington
kryan@eng.ucsd.edu

…and the rest of the logo mafia

Time to panic

Existing TEEs Are Not Enough

<table>
<tr><th></th><th>Platform</th><th>Vulnerable to Side Channels</th><th>Dedicated Hardware</th><th>Availability</th></tr>
<tr><td rowspan="4">Enclaves</td><td>ARM TrustZone</td><td rowspan="7">YES</td><td rowspan="7">YES</td><td>Available</td></tr>
<tr><td>Intel SGX V1</td><td>Depreciated</td></tr>
<tr><td>Intel SGX V2</td><td>Server-Grade Processors</td></tr>
<tr><td>Amazon Nitro Enclaves</td><td>AWS Only</td></tr>
<tr><td rowspan="3">Trusted VM</td><td>Intel TDX</td><td>Server-Grade Processors</td></tr>
<tr><td>AMD SEV</td><td>Server-Grade Processors</td></tr>
<tr><td>ARM CCA</td><td>Not Available</td></tr>
</table>

Goal:

Side-Channel-Resistant Verifiable FHE Using Commodity Hardware

Key Insight

In FHE, all sensitive data is **encrypted**

Enclaves do **not** need to guarantee any
program privacy

We can focus our efforts on providing **integrity only**

Argos

Argos

- *Integrity-only* TEE for maliciously-secure verifiable FHE

Argos

- *Integrity-only* TEE for maliciously-secure verifiable FHE
- Can be used for fully malicious and authenticated PSI and PIR

Argos

- *Integrity-only* TEE for maliciously-secure verifiable FHE
- Can be used for fully malicious and authenticated PSI and PIR
- Secure by design against microarchitectural side channels

Argos

- *Integrity-only* TEE for maliciously-secure verifiable FHE
- Can be used for fully malicious and authenticated PSI and PIR
- Secure by design against microarchitectural side channels
- No specialized hardware — Compatible with commodity processors

Argos

- *Integrity-only* TEE for maliciously-secure verifiable FHE
- Can be used for fully malicious and authenticated PSI and PIR
- Secure by design against microarchitectural side channels
- No specialized hardware — Compatible with commodity processors
- ~3% performance overhead for FHE computation, <8% for complex protocols

What's the catch?

- Argos only provides integrity
- Weaker threat model than ZK proofs
- Requires a custom hypervisor and runtime

Plan

I - INTRODUCTION

II - A QUICK HISTORY OF TRUST

II - DESIGN OVERVIEW

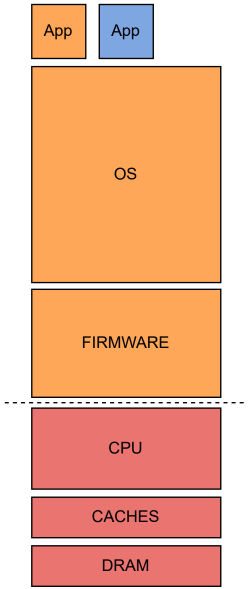
IV - PERFORMANCE OPTIMIZATION

V - PROTOTYPE AND EVALUATION

Threat Model 😈

We assume a **strong privilege software attacker**

They have compromised **most of the software stack** and can mount **any type of side channels.**



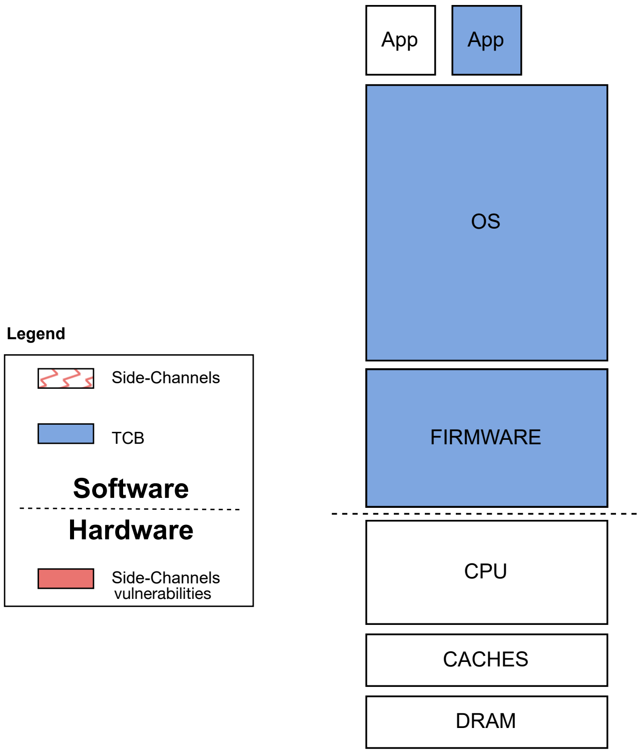
Assumptions (all standard)
- A small trusted code base (TCB)
- Constant-time cryptography
- Hardware is *functionally* correct

Out-of-Scope Attacks (see discussion later)
- Physical attacks
- Fault Injection

A Quick History of Trust

1990

Baseline

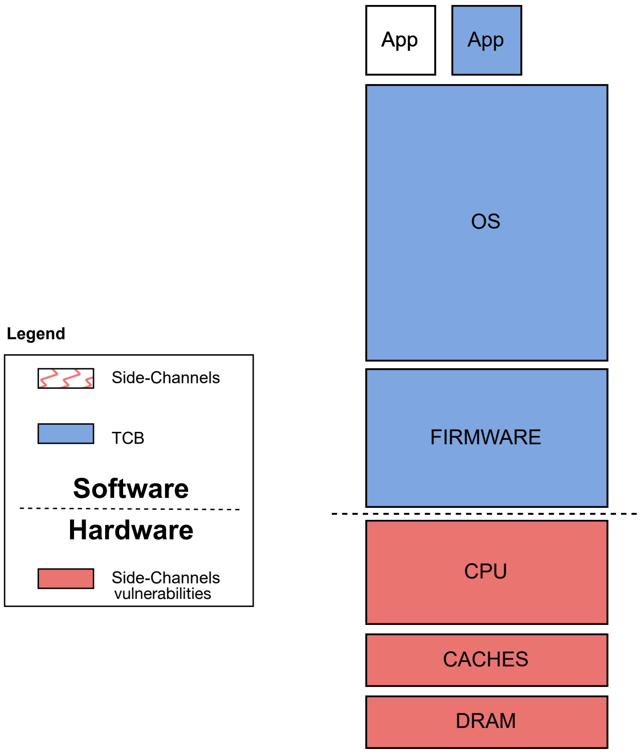


- Software-only TEE
- Trusted code base
- Hardware contains some side-channels
- Software that manipulates secrets or keys is vulnerable to side channels

A Quick History of Trust

1990

Baseline

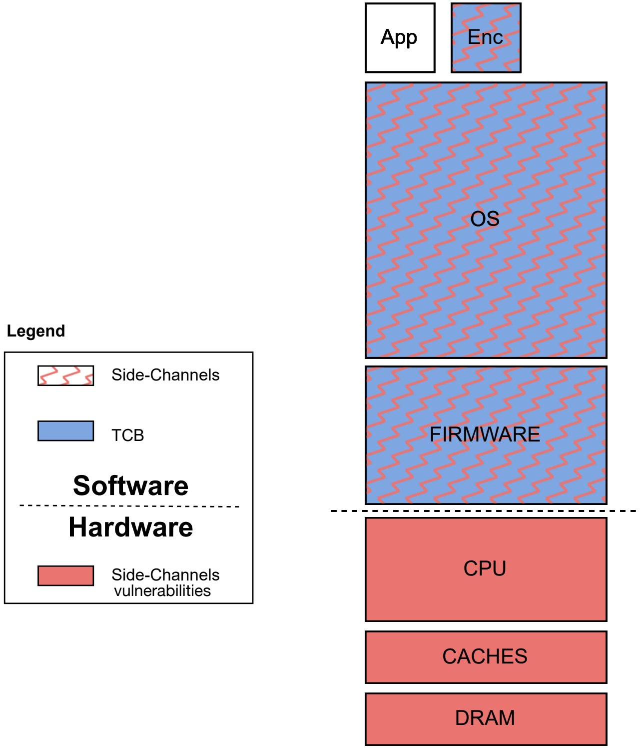


- Software-only TEE
- Trusted code base
- Hardware contains some side-channels
- Software that manipulates secrets or keys is vulnerable to side channels

A Quick History of Trust

1990

Baseline

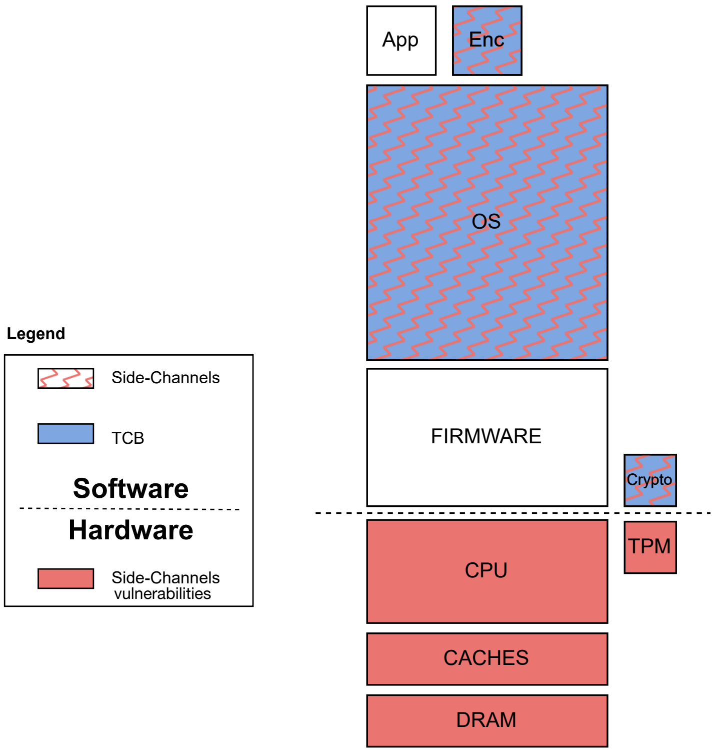


- Software-only TEE
- Trusted code base
- Hardware contains some side-channels
- Software that manipulates secrets or keys is vulnerable to side channels

2007

Dynamic Root of Trust

TPM 1.2 + Intel TXT or AMD SVM

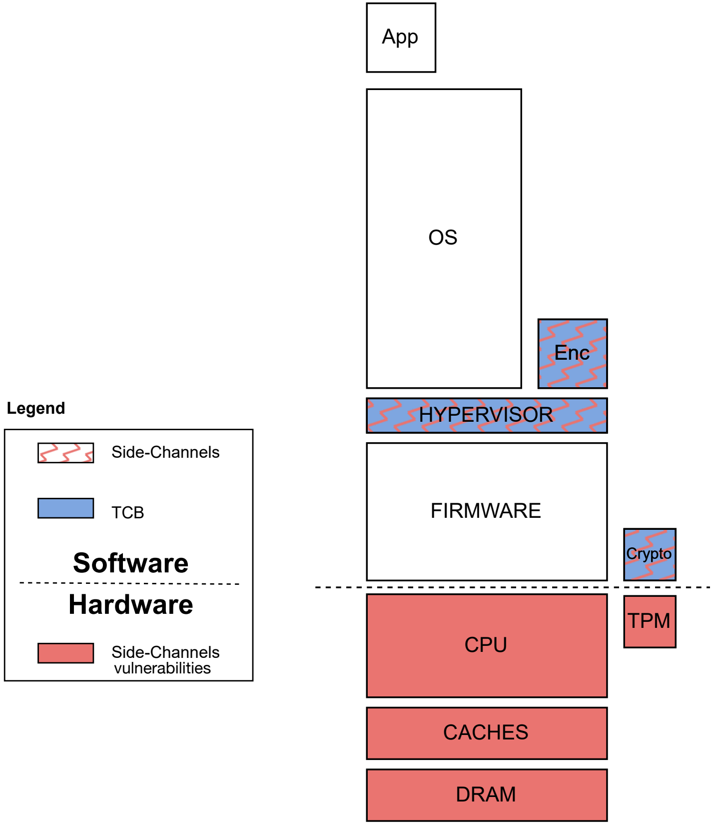


- Discrete or integrated TPM chip
- Exclude the BIOS/bootloader from the TCB
- Significantly shrinks the TCB
- Still have the entire OS

2008

Hypervisor-based Isolation

TrustVisor[0], AWS Nitro, Apple Private Cloud

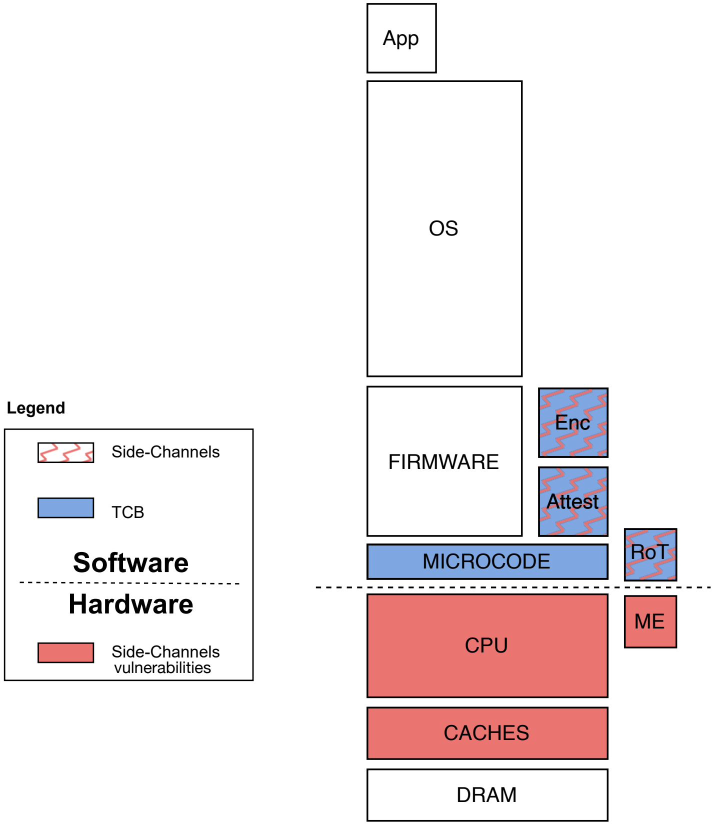


- Virtual environments can be isolated
- Use a small hypervisor as a security monitor
- TPM needs to be **virtualized** in the hypervisor
- **Exposes secrets to side channels**

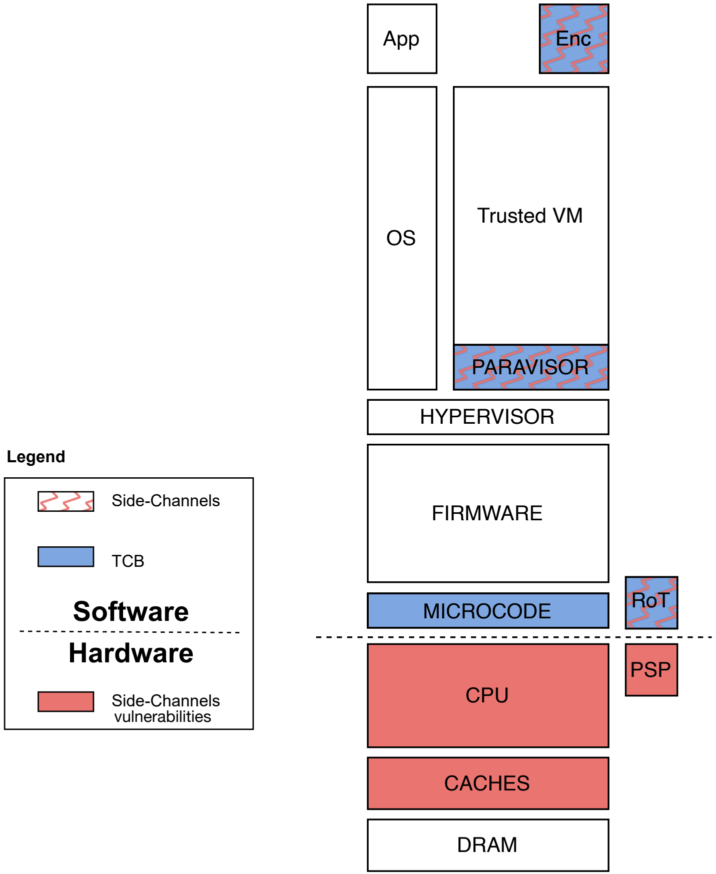
2015

Secure Enclaves

Intel SGX, ARM Trustzone



- Security monitor in implemented in microcode
- Most of the root-of-trust cryptography is performed in a co-processor (Intel ME)
- Cryptography for final remote attestation step is performed inside an enclave
- **Side channels attacks also break remote attestation**



2016

Trusted VMs

AMD SEV, Intel TDX or ARM CCA

- Exclude the hypervisor from the TCB
- Most of the root-of-trust cryptography is performed in a co-processor (AMD PSP)
- Cryptography for final remote attestation step is performed inside a paravisor
- **Exposes secrets to side channels**

Plan - Argos

I - INTRODUCTION

II - A QUICK HISTORY OF TRUST

II - DESIGN OVERVIEW

IV - PERFORMANCE OPTIMIZATION

V - PROTOTYPE AND EVALUATION

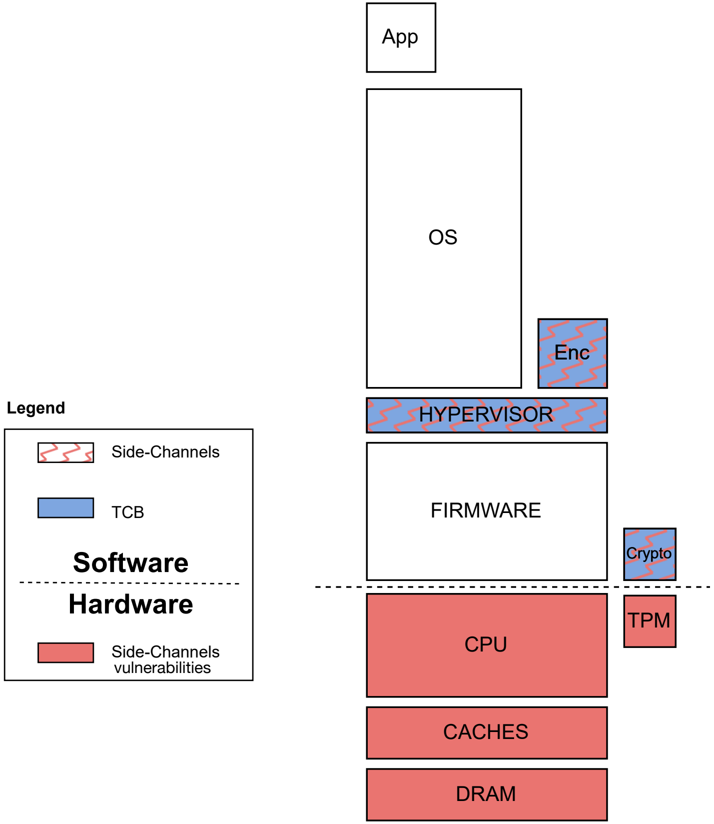
Today

What can we do from there?

Today

What can we do from there?

- We don't want specific hardware: start from **hypervisor-based isolation** design



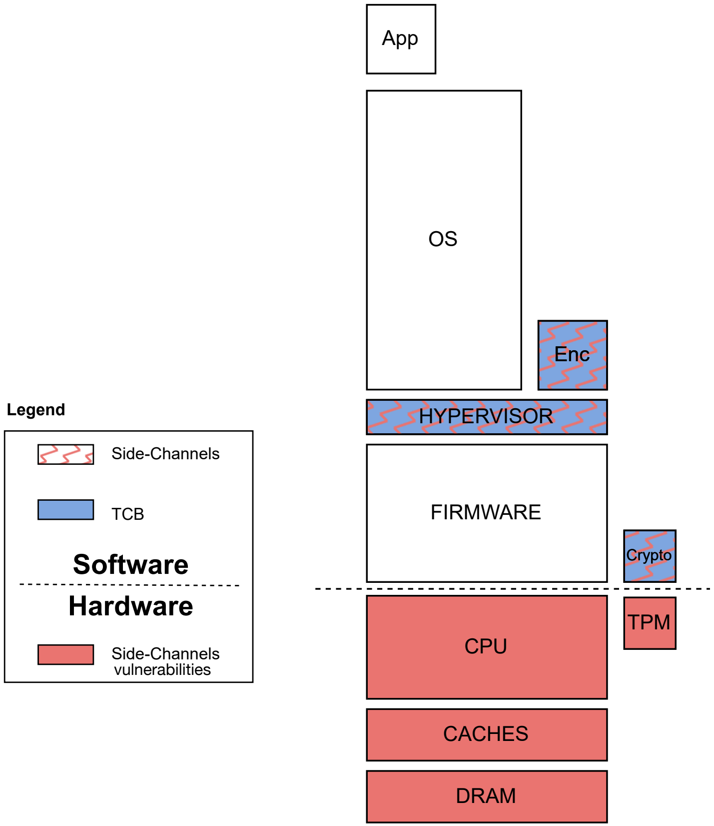
Today

What can we do from there?

- We don't want specific hardware: start from **hypervisor-based isolation** design

Today

What can we do from there?

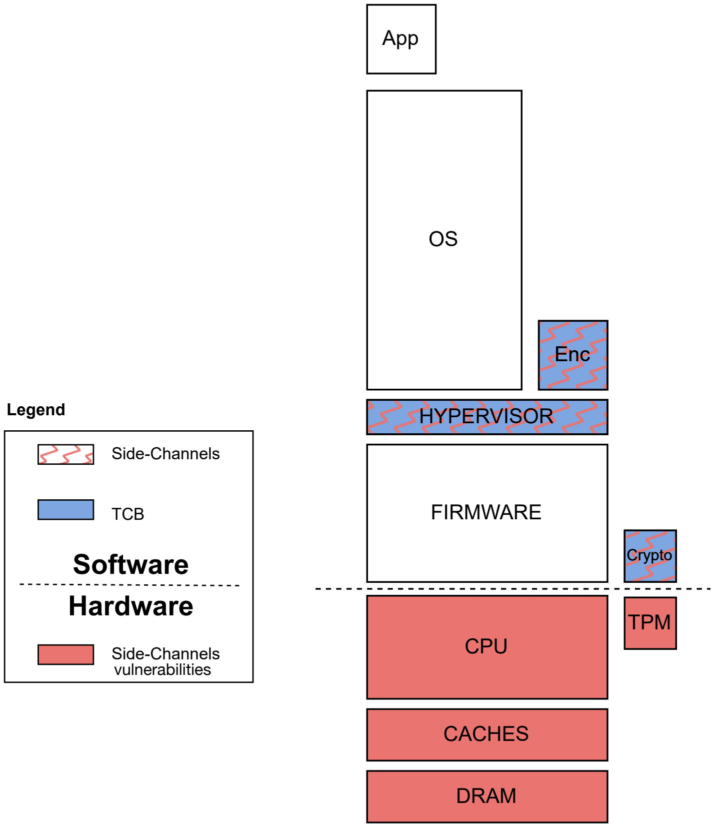


- We don't want specific hardware: start from **hypervisor-based isolation** design

- Remember or key insight:

Today

What can we do from there?



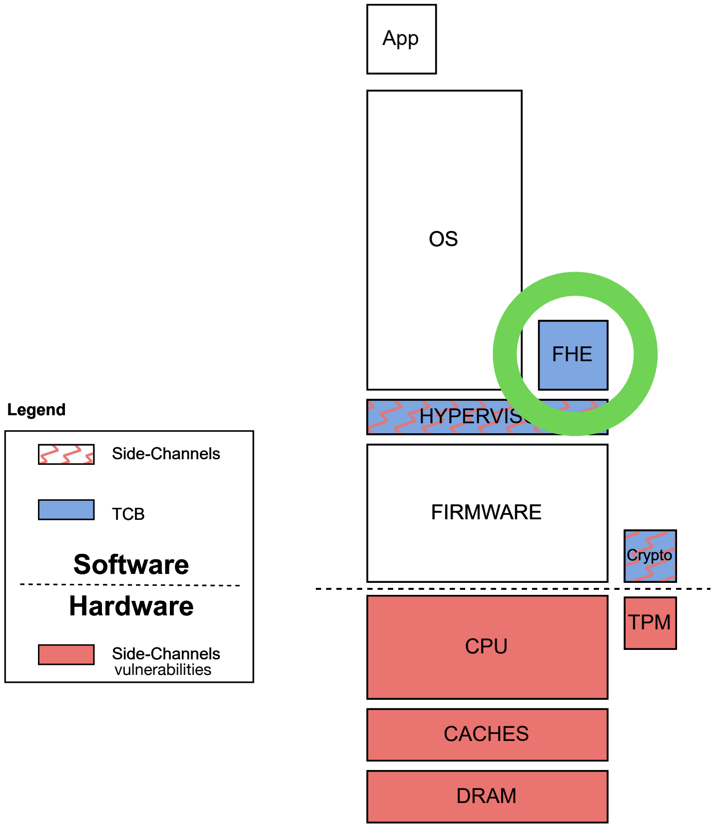
- We don't want specific hardware: start from **hypervisor-based isolation** design

- Remember or key insight:

In FHE, all sensitive data is **encrypted**

Today

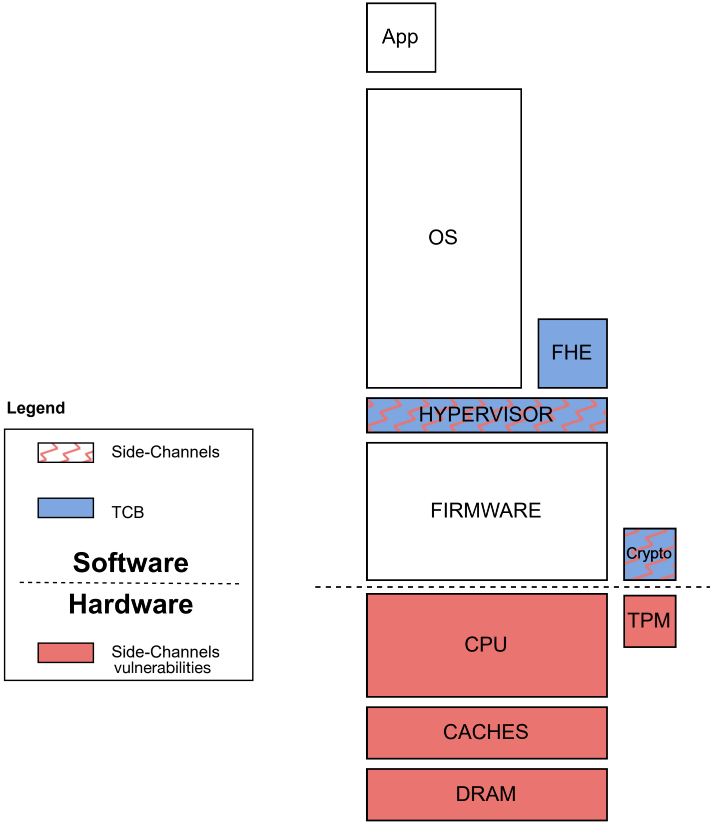
What can we do from there?



- We don't want specific hardware: start from **hypervisor-based isolation** design

- Remember or key insight:

In FHE, all sensitive data is **encrypted**



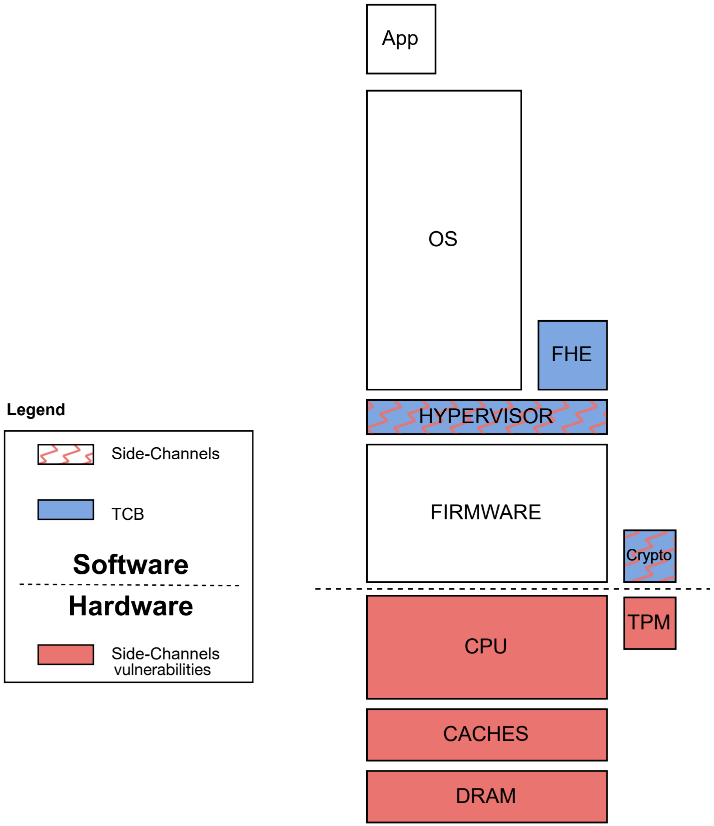
Today

What can we do from there?

- Start back from hypervisor-based isolation (no specific hardware)
- In FHE, all sensitive data is **encrypted**
- **We do not need long-term secret storage**
- Only secret left on the CPU is final attestation key

Today

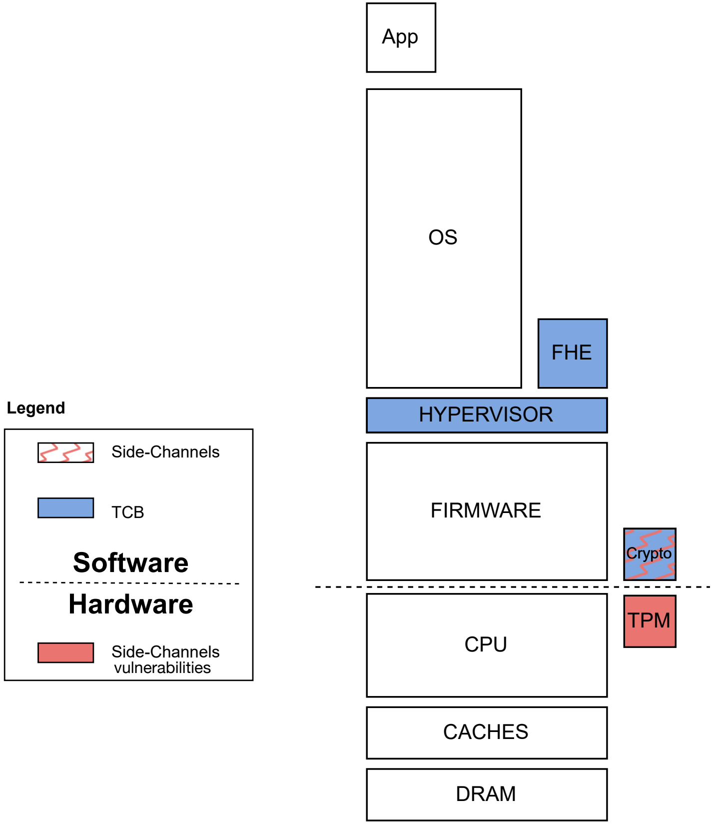
What can we do from there?



- Start back from hypervisor-based isolation (no specific hardware)
- In FHE, all sensitive data is **encrypted**
- **We do not need long-term secret storage**
- Only secret left on the CPU is final attestation key

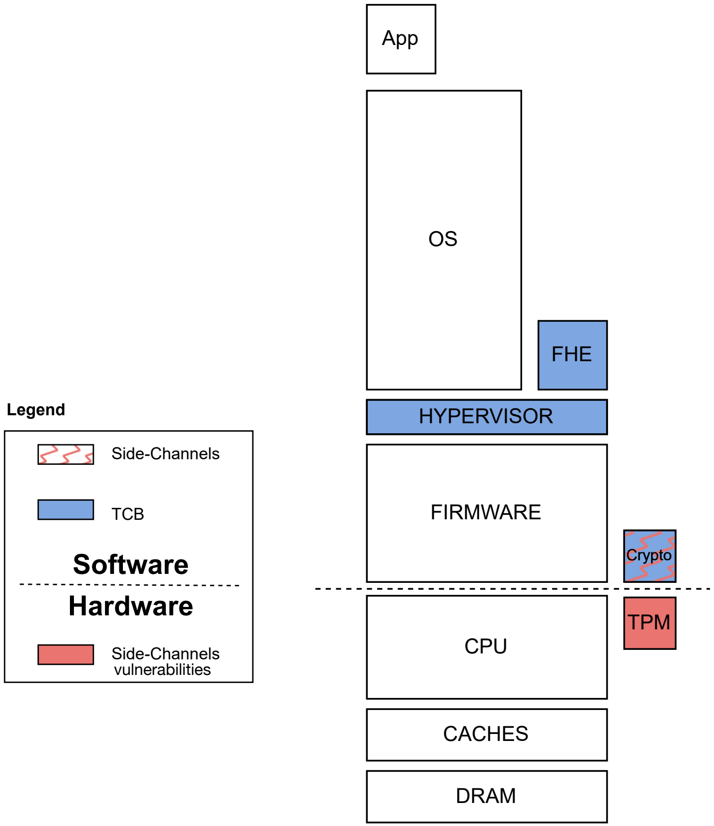
Keep the final attestation key in the TPM!

Argos



- Start back from hypervisor-based isolation (no specific hardware)
- In FHE, all sensitive data is **encrypted**
- **We do not need long-term secret storage**
- **Final attestation key is kept in the TPM**

Argos



- Start back from hypervisor-based isolation (no specific hardware)
- In FHE, all sensitive data is **encrypted**
- **We do not need long-term secret storage**
- **Final attestation key is kept in the TPM**

➡ **No side channels beyond the TPM**

No side channels beyond the TPM

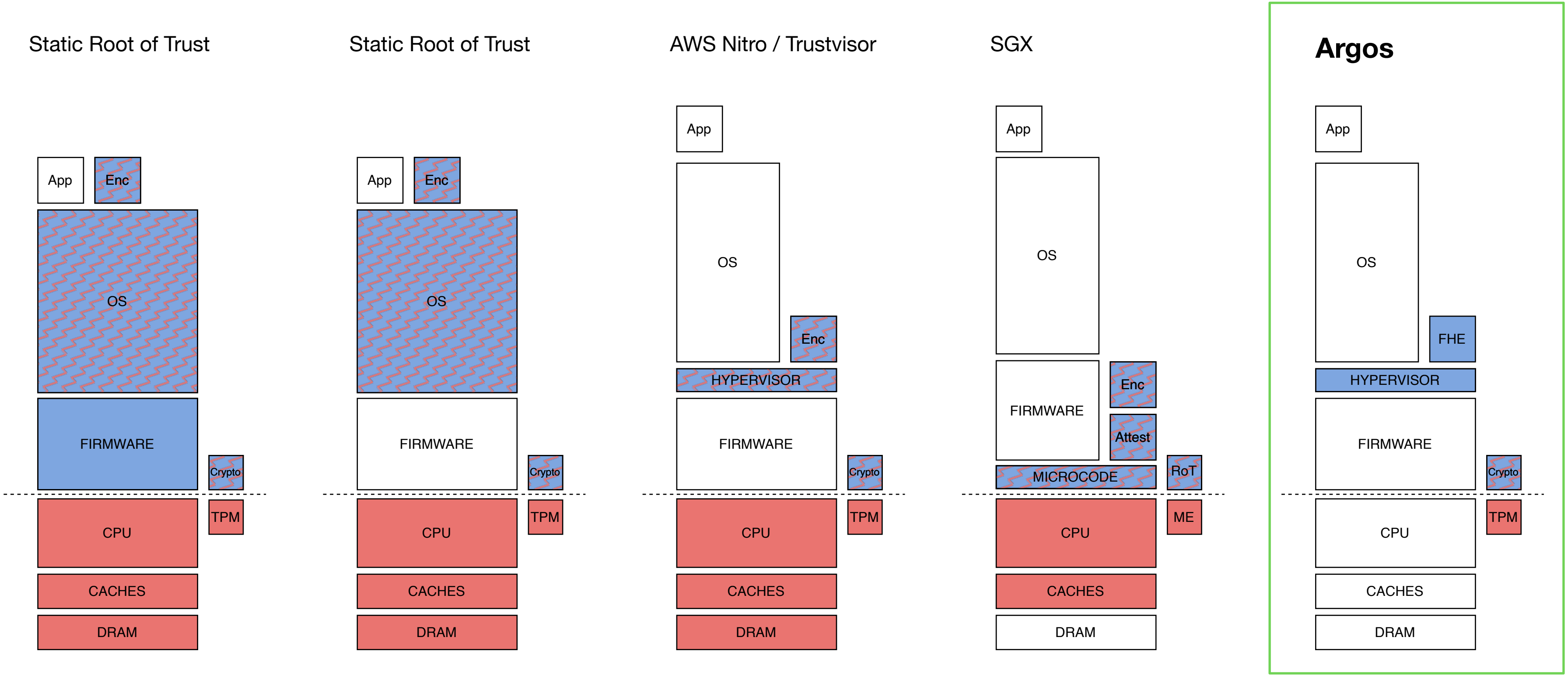
TPM does not share microarchitecture with the CPU.
TPM cannot run (attacker) arbitrary code.
TPM microarchitecture is intentionally extremely simple.

There is no shared microarchitectural state between the TPM and an attacker
The attacker cannot leverage microarchitectural side channels

Only side channels left
- TPM-based timing or completion time[0] (addressed by constant-time cryptography)
- TPM physical side channels such as power, electromagnetic (out-of-scope for now)

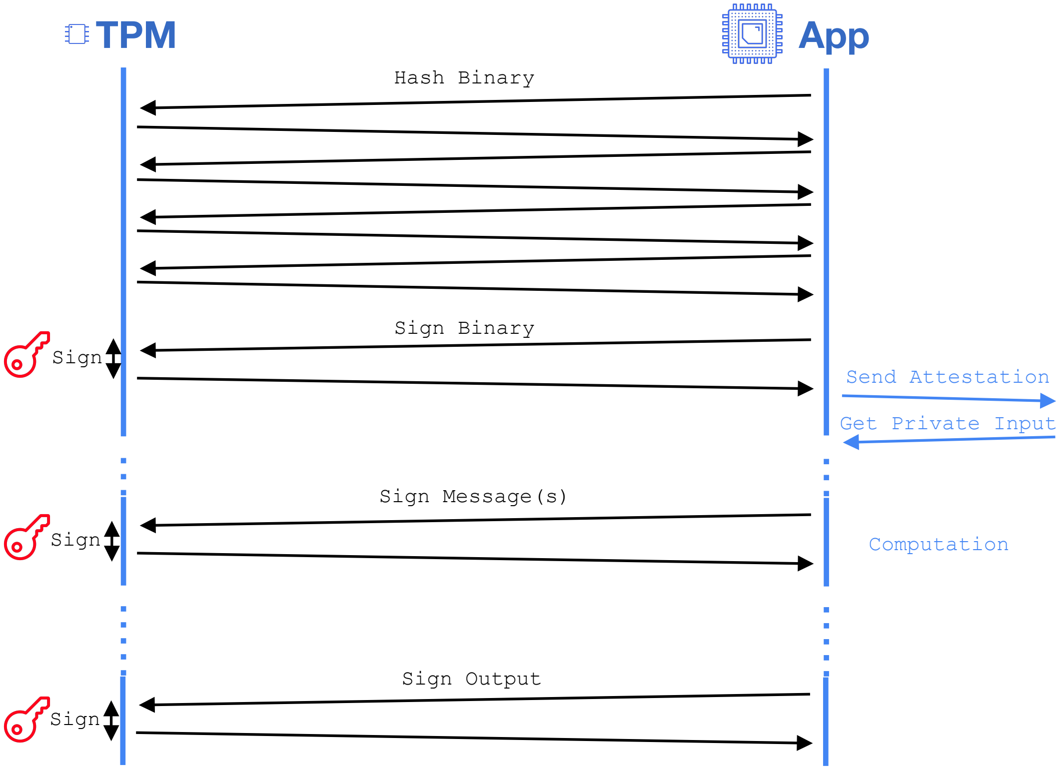
[0] Moghimi, Daniel, et al. "TPM-FAIL:TPM meets timing and lattice attacks." *29th USENIX Security Symposium (USENIX Security 20)*. 2020.

A Quick History of Trust



Plan

Performance Impact



TPM communications and operations are **expensive**

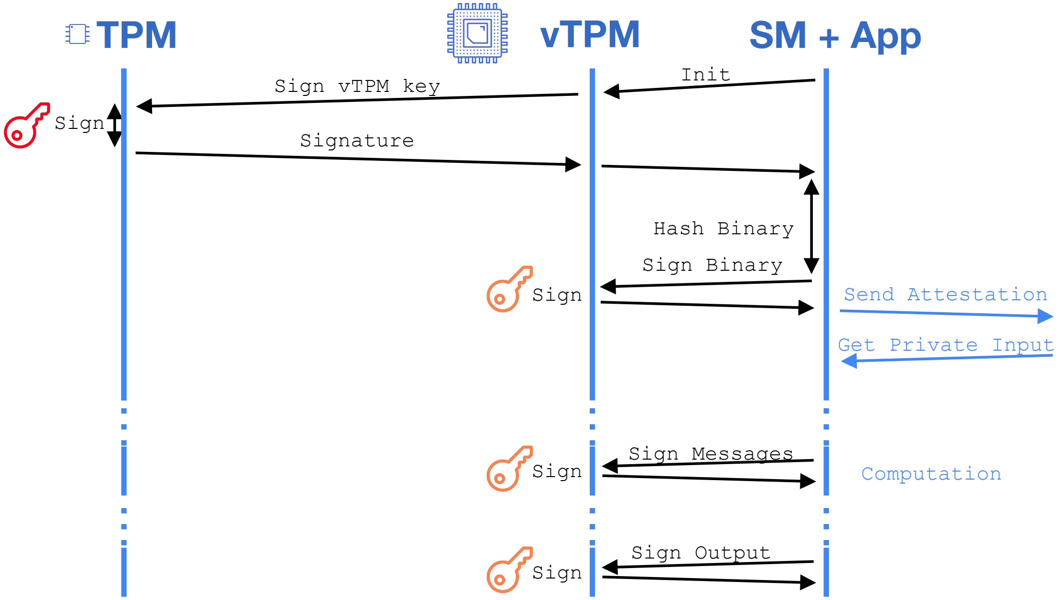
TPM signature **~200ms**

For complex interactive protocols

1 signature per message

Current solution: virtualized TPM

Performance Impact



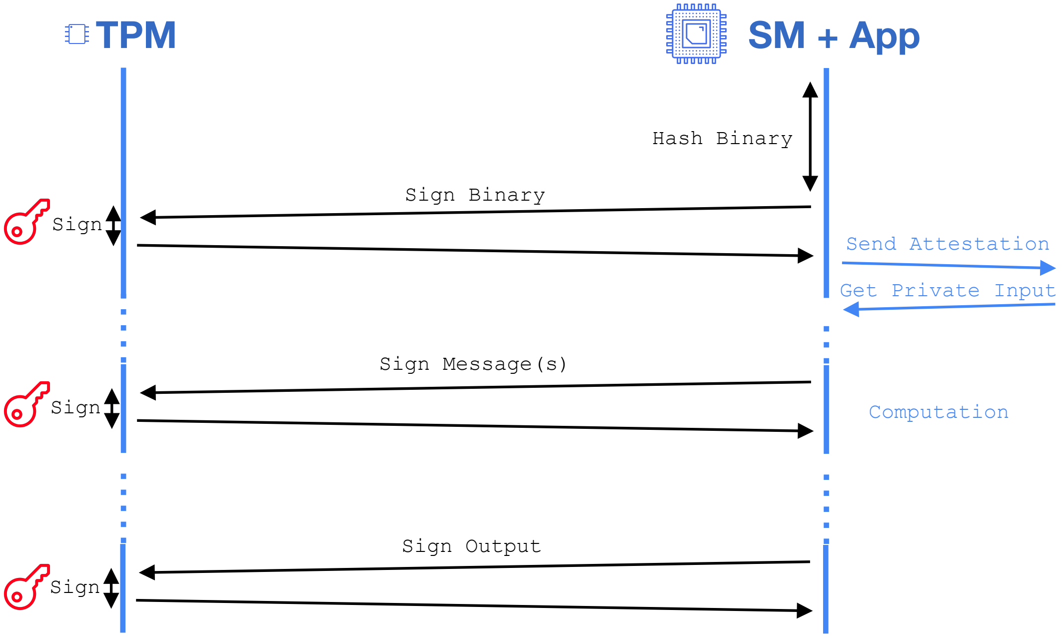
vTPM needs to get attested by a real TPM

vTPM signature **~1ms**

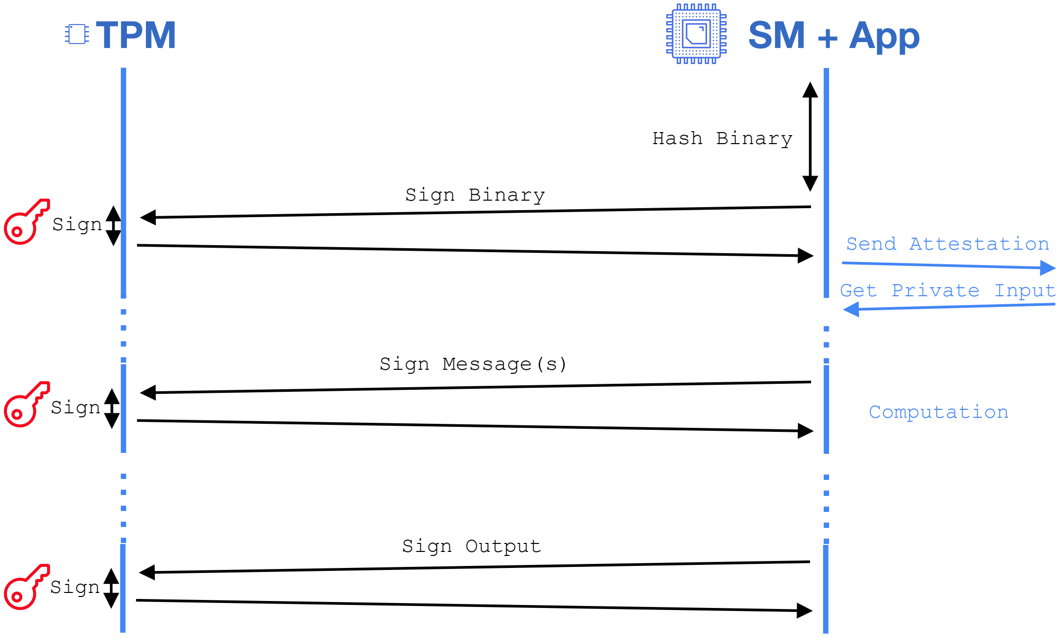
1 real TPM signature

How to keep secrets in the TPM without performance overheads?

Performance Impact

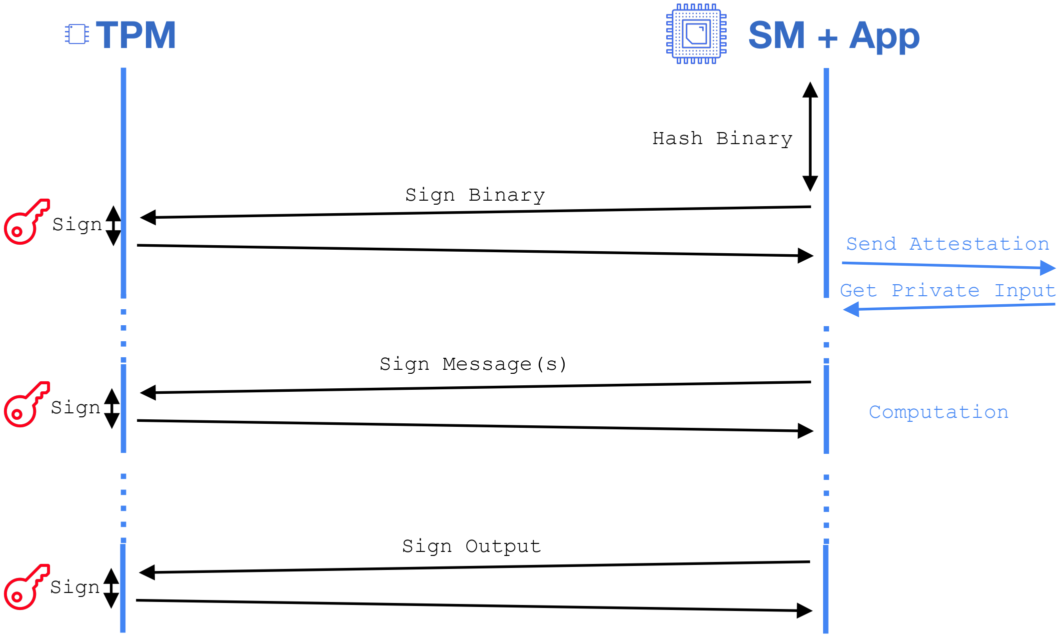


Performance Impact



In FHE, all sensitive data is **encrypted**

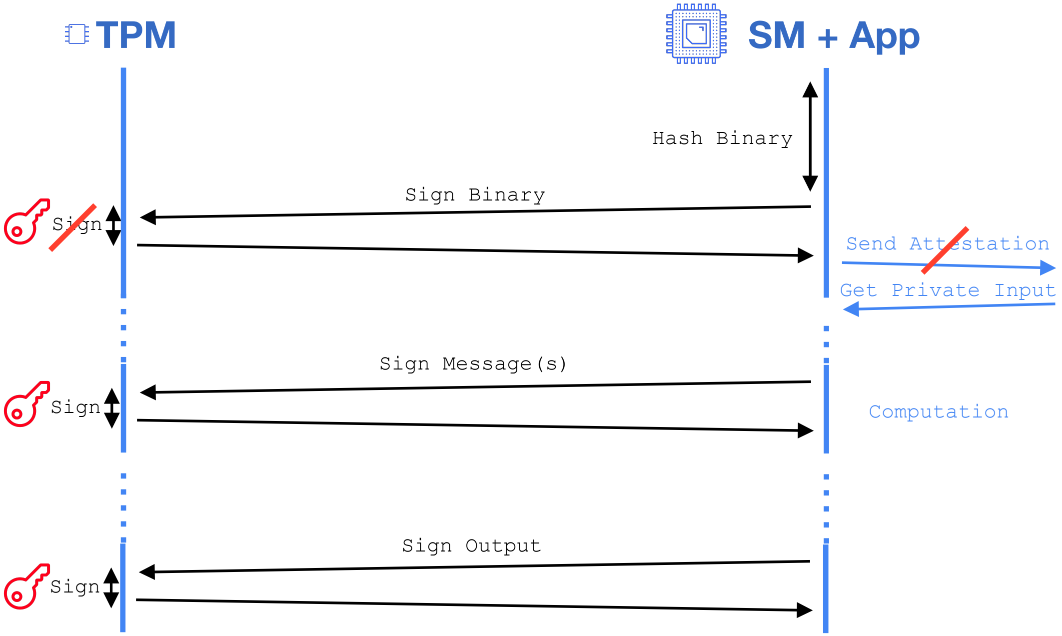
Performance Impact



In FHE, all sensitive data is **encrypted**

- Hashing does not require any secrets

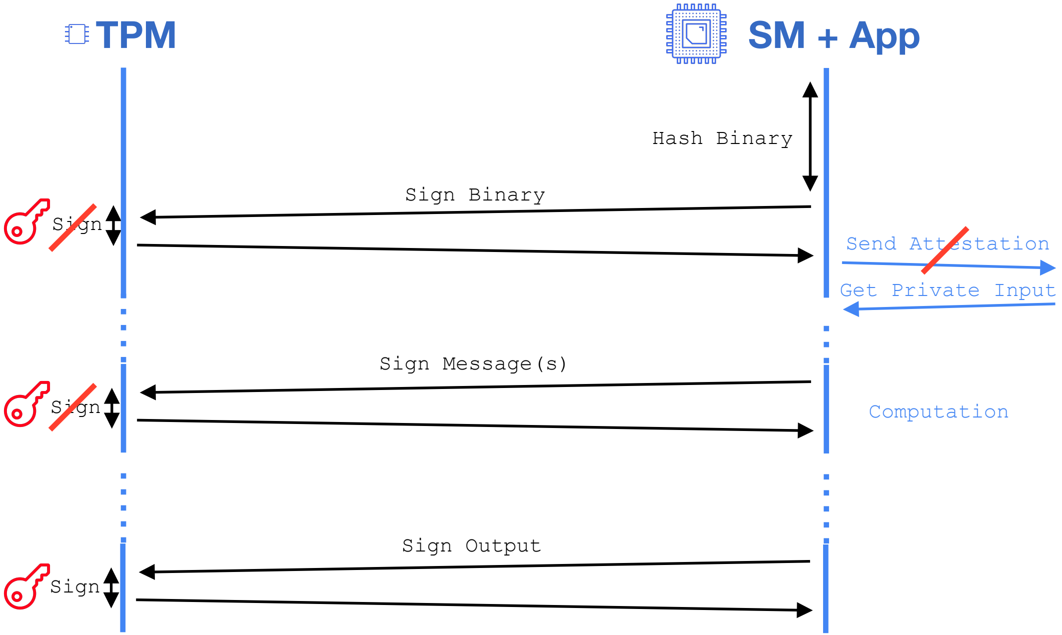
Performance Impact



In FHE, all sensitive data is **encrypted**

- Hashing does not require any secrets
- No need to attest the enclave before sending input

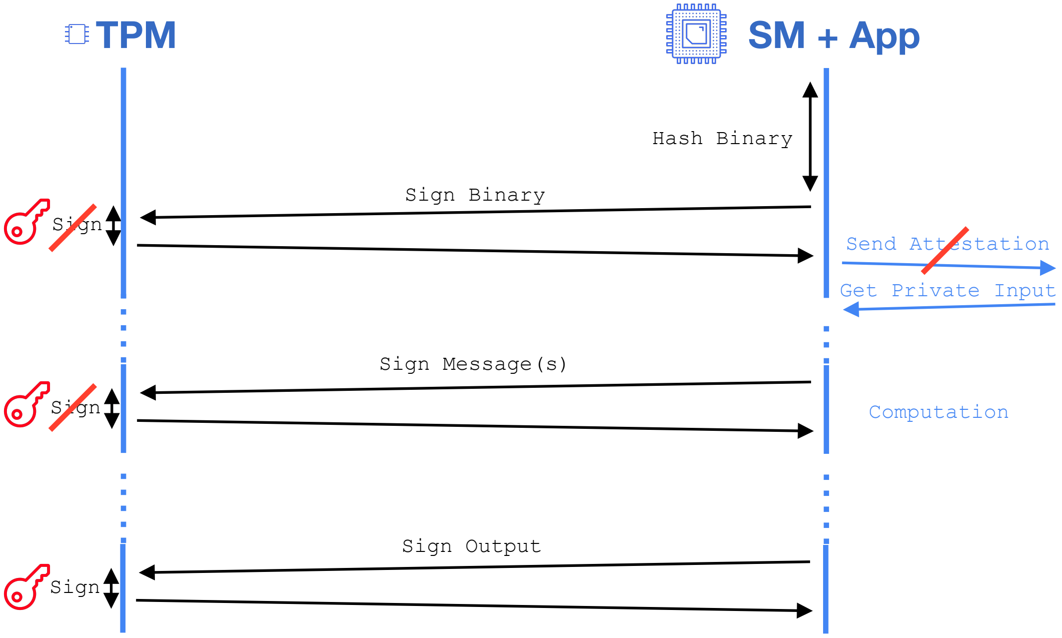
Performance Impact



In FHE, all sensitive data is **encrypted**

- Hashing does not require any secrets
- No need to attest the enclave before sending input
- Depending on the protocol, no need to sign individual messages

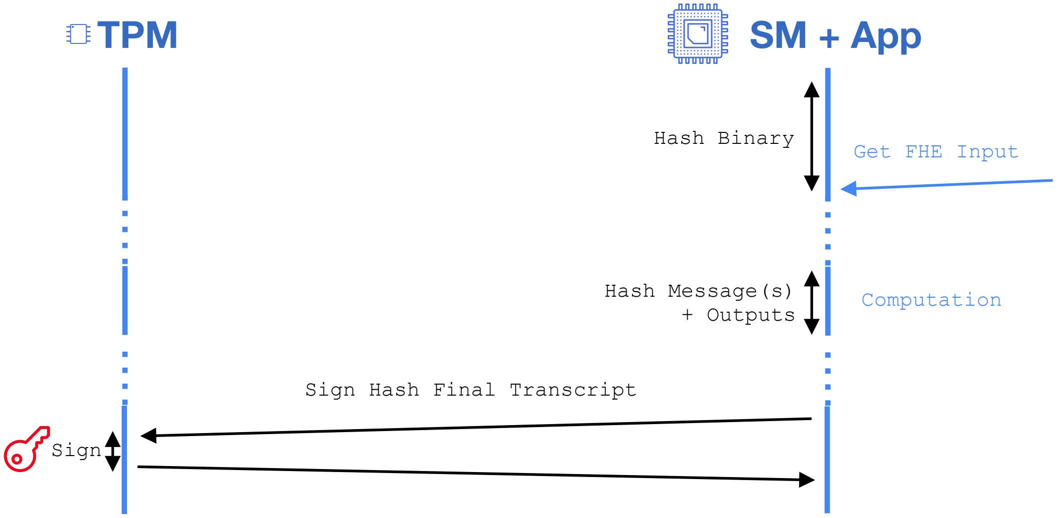
Performance Impact



In FHE, all sensitive data is **encrypted**

- Hashing does not require any secrets
- No need to attest the enclave before sending input
- Depending on the protocol, no need to sign individual messages
- Only sign final transcript

Performance Impact



1 TPM signature

In FHE, all sensitive data is **encrypted**

- Hashing does not require any secrets
- No need to attest the enclave before sending input
- Depending on the protocol, no need to sign individual messages
- Only sign final transcript

Plan

I - INTRODUCTION

II - A QUICK HISTORY OF TRUST

II - DESIGN OVERVIEW

IV - PERFORMANCE OPTIMIZATION

V - PROTOTYPE AND EVALUATION

Building Our Prototype

Hardware — Minimal Requirements

Any machine > 2008

We use an Intel i7-7700 3.60GHz from 2017

Detailed requirements:
- Privilege execution level > OS — Hardware virtualization support (e.g., Intel VT-x) **> 2006**
- TPM + Hardware RoT (e.g., Intel Boot Guard) + Dynamic RoT (e.g., Intel TXT) **> 2007**
- Memory isolation mechanism — Extended page table (e.g., Intel EPT) **> 2008**

Building Our Prototype

Hardware — Minimal Requirements

Any machine > 2008

We use an Intel i7-7700 3.60GHz from 2017

Detailed requirements:
- Privilege execution level > OS — Hardware virtualization support (e.g., Intel VT-x) > **2006**
- TPM + Hardware RoT (e.g., Intel Boot Guard) + Dynamic RoT (e.g., Intel TXT) > **2007**
- Memory isolation mechanism — Extended page table (e.g., Intel EPT) > **2008**

Software - Building Blocks

Mini-hypervisor from EPFL : Tyche

OS for resource management: Linux + KVM

Runtime custom or Gramine

FHE library SEAL 4.1

Applications PSI or PIR benchmark

TCB is between **40KLOC and 80KLOC**

Evaluation

FHE Evaluation

Evaluation of 3 different circuit sizes

Argos is 2-7 **orders of magnitude** faster than ZK proofs

83x improvement over previous work on SGXv1[0]

Similar performance that commercial TEEs

+ better security

Complex protocols

Authenticated Private Information Retrieval

1M 128B elements

Pre-processing: 3.4s (+21%)

Query Processing: 1.6s (+1%)

Authenticated Private Set Intersection

1M ∩ 3K unlabeled elements

Pre-processing: 37s (-8%)

Query Processing: 1.6s (+8%)

+Negligible communication overhead (<1%)

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Discussion — What about physical attacks? 😈

Argos protects against *some* physical attacks:

- Cold boot attacks (common)
- BIOS tampering (common)
- Physical side channels on the **CPU** such as electro-magnetic, sound, power etc.

Argos does not protects against row-hammer or fault-injection attacks

No published RH attacks demonstrate gaining hypervisor privilege on a hardware VM

Software can be harden against fault injections with control flow checks

Argos does not protects against physical side channels on the TPM

TPM are now integrated - No published physical attacks

For the most part, is addressed by constant-time cryptography

Argos

- *Integrity-only* TEE for maliciously-secure verifiable FHE
- Can be used for fully malicious and authenticated PSI and PIR
- Secure by design against microarchitectural side channels
- No specialized hardware — Compatible with commodity processors
- ~3% performance overhead for FHE computation, <8% for complex protocols

?

https://arxiv.org/pdf/2412.03550

https://github.com/mit-enclaves/argos

“As they were speaking, a dog that had been lying asleep raised his head and pricked up his ears. This was Argos, whom ~~Odysseus~~ had bred before setting out for ~~Troy~~"

Intel

the side-channel war

Backup Slides

Performance Evaluation

Simple FHE evaluation of various circuit sizes

	Platform	Tiny	Small	Medium
ZK Proofs	Baseline	2ms	11ms	14ms
	Bulletproofs	7569s	3957s	8697s
	Aurora	1554s	3750s	5028s
	Groth16	196s	473s	634s
	Rinocchio	320ms	305s	443s
	SGXv1	154ms	1100ms	1260ms
Trusted Hardware	Baseline Azure	283us	1727us	3170us
	SGXv2 Azure	290us (+3%)	1840us (+7%)	3638us (+15%)
	Baseline AWS	324us	1889us	3456us
	Nitro Enclave	317us (-2%)	1827us (-3%)	3450us (-0%)
	Baseline Local	351us	2341us	4376us
	Argos+G	392us (+12%)	2702us (+16%)	5202us (+19%)
	Argos	352us (+0%)	2480us (+6%)	4447us (+2%)

Argos is 2-7 **orders of magnitude** faster than ZK proofs

~83x improvement over previous work on SGX[0]

Similar performance that commercial TEEs **+ better security**

[0] Viand, Alexander, Christian Knabenhans, and Anwar Hithnawi. "Verifiable fully homomorphic encryption." arXiv preprint arXiv:2301.07041 (2023).

Evaluation - TCB

Component	LOC
BIOS	1.5M
Linux	28M
Security Monitor	18K
Runtime (Custom / Gramine)	870 / 20K
SEAL Library [179]	20K
Application	1K—20K

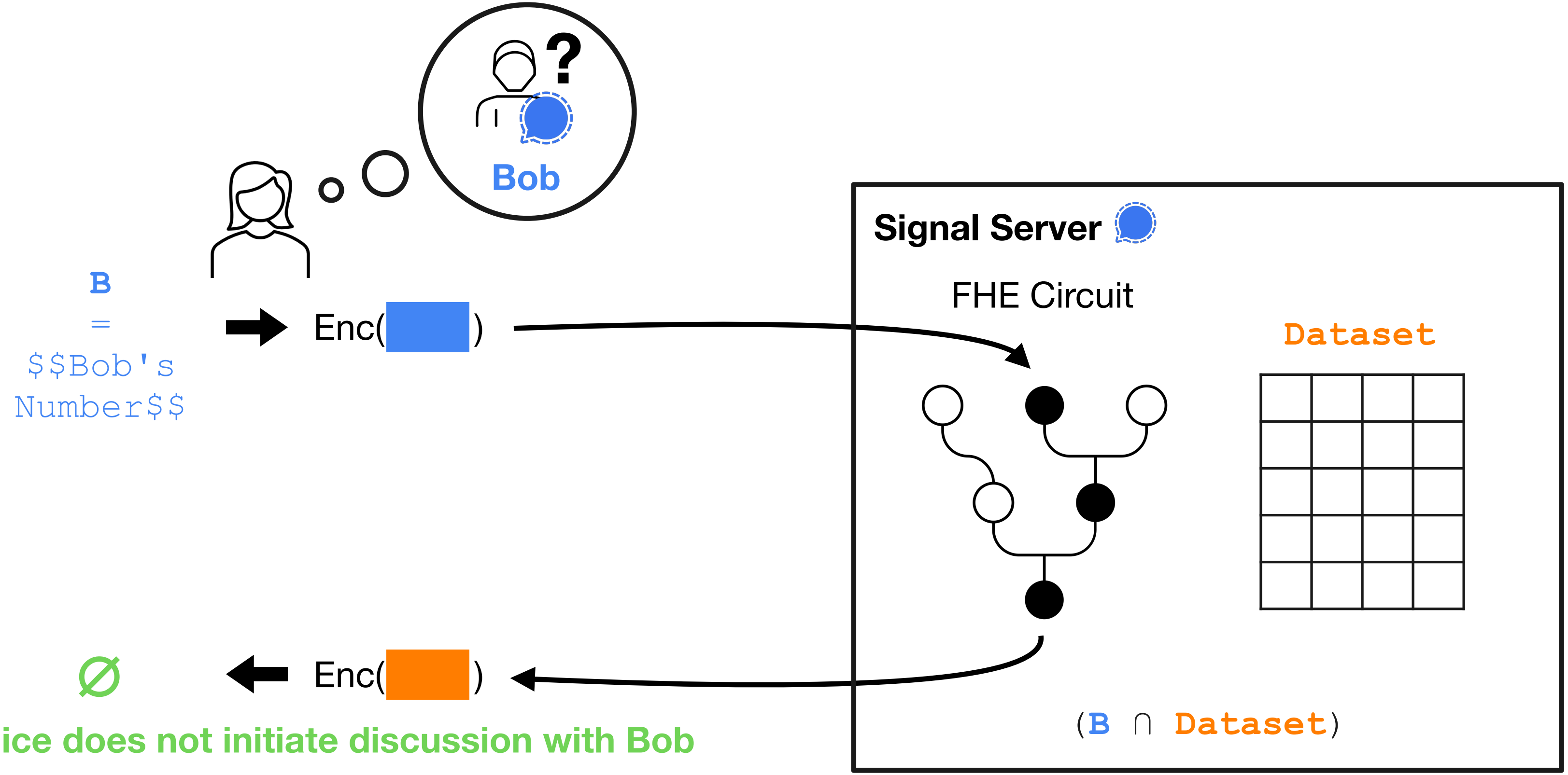
Minimal compared to Linux or other existing hypervisors (e.g. XEN ~200KLOC)

Comparison With Existing Platforms

TEE Platform	Security						Usability			Performance		
	TCB	μ-arch SC	Cold Boot	Phys. SC CPU	Fault Inject.	Phys. SC TPM	Availability	Dedic. HW	Implementation	Setup	Attest.	Comp.
ZK Proofs	Null	P	P	P	P	P	SW	SW	Open Source	− − − −	− − − −	− − − −
Nitro Enclaves	Large	V	V	V	V	V	AWS	Yes	Closed Source	− − −	+ +	+ +
Arm TrustZone	Small	V	V	V	V	V	High	Yes	HW Closed Source	+	+	−
Intel SGX V1	Small	V	P	V	V	V	Deprecated	Yes	HW Closed-Source	− −	+	− −
Intel SGX V2	Small	V	P	V	V	V	Cloud	Yes	HW Closed-Source	− −	+	+
AMD SEV	Large	V	P	V	V	V	Cloud	Yes	HW Closed-Source	− − −	+	+ +
Intel TDX	Large	V	P	V	V	V	Coming	Yes	HW Closed-Source	− − −	+	+ +
ARM CCA	Large	V	P	V	V	V	Coming	Yes	HW Closed-Source	− − −	+	+ +
TrustVisor[92]	Small	V	V	V	V	V	Deprecated	No	Open Source	−	+ +	+ +
Flicker [93]	Tiny	P	P	P	V	V	Deprecated	No	Open Source	− − −	− − −	− − −
Argos	Small	P	P	P	V	V	High	No	Open Source	+	−	+ +

SC: Side Channel, P: Protected, V: Vulnerable. +'s and -'s represent relative (and subjective) measure of performance.

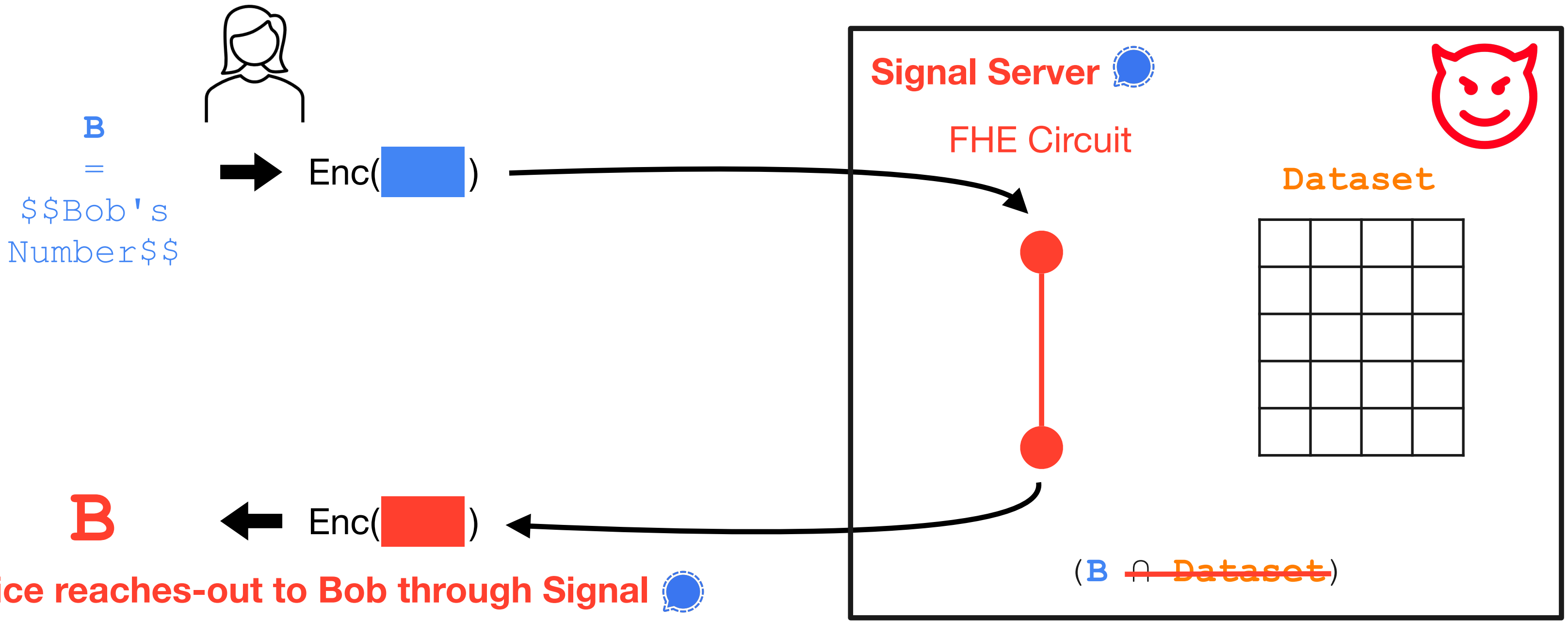
Example: PSI for Contact Discovery[0]



Alice does not initiate discussion with Bob through Signal — Bob's privacy is preserved

Problem: No Integrity!

FHE protocols are only **passively** secure



Alice reaches-out to Bob through Signal
leaking Bob's phone number