

# Trusted Execution Environment (TEE) on Open-source RISC-V Processor System

**Authors:** Trong-Thuc Hoang, Ckristian Duran, Akira Tsukamoto,  
Kuniyasu Suzuki, and Cong-Kha Pham

# Outline

1. Introduction: RISC-V
2. Trusted Execution Environment
3. TEE-Hardware System
4. Results & Conclusion

# Outline

1. Introduction: RISC-V
2. Trusted Execution Environment
3. TEE-Hardware System
4. Results & Conclusion

# 1. Introduction: RISC-V (1/4)

## What is RISC-V?

- RISC-V is an open-source Instruction Set Architecture (ISA)
- What is ISA? *i386* and *AMD64* are ISA
- Comparing RISC-V to i386/AMD64 is like comparing Linux to Windows
- Originate from UC Berkeley, but now is maintained by RISC-V Foundation



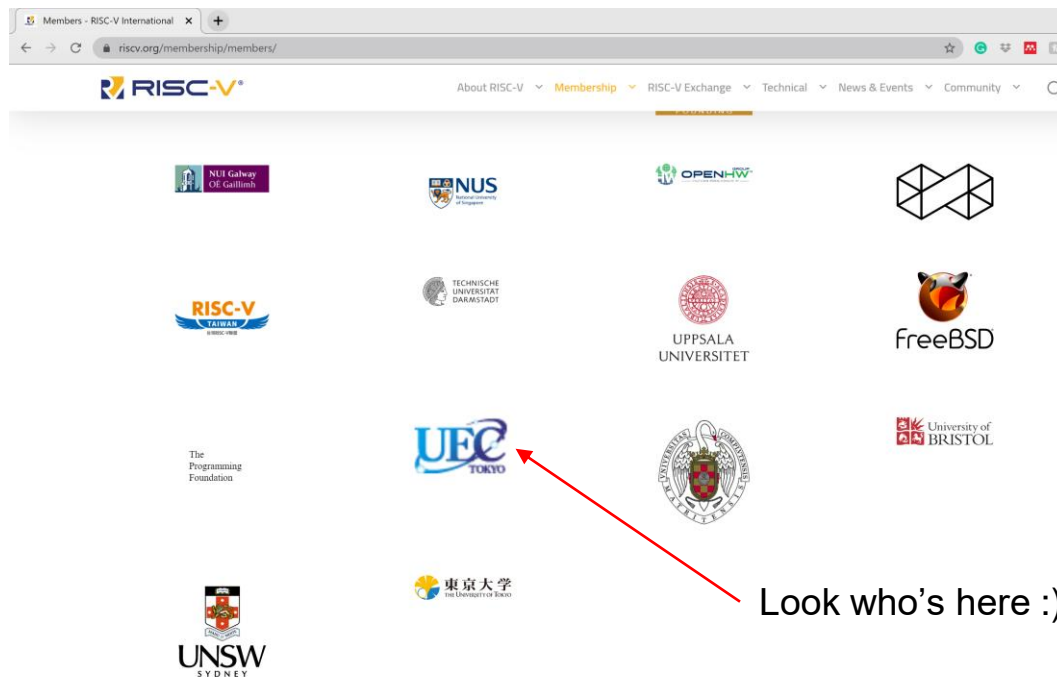
**RISC-V** Foundation: 235+ Members



# 1. Introduction: RISC-V (2/4)

## What is RISC-V?

- RISC-V is an open-source Instruction Set Architecture (ISA)
- What is ISA? *i386* and *AMD64* are ISA
- Comparing RISC-V to i386/AMD64 is like comparing Linux to Windows
- Originate from UC Berkeley, but now is maintained by RISC-V Foundation



Look who's here :)

# 1. Introduction: RISC-V (3/4)

## What is RISC-V?

- RISC-V is an open-source Instruction Set Architecture (ISA)
- What is ISA? *i386* and *AMD64* are ISA
- Comparing RISC-V to i386/AMD64 is like comparing Linux to Windows
- Originate from UC Berkeley, but now is maintained by RISC-V Foundation

## What is RISC-V capable of?

- RISC-V supports 32-bit, 64-bit, and 128-bit addressing
- Core ISA is just *I*nteger, but it has many extensions: *M*ultiplication, *A*ttomic, *F*loating-point, *D*ouble floating-point, and *C*ompressed
- When designing your system, you can combine any of those extensions. The most common are *RV64IMAFDC* and *RV32IMAFDC*
- Open ISA → anybody can custom their own CPU → highly customize processor to fit any specific requirement

**RISC-V Foundation** guarantees the open-ness of the ISA, and also maintains its toolchain (for example, assembler, linker, compiler, etc.)

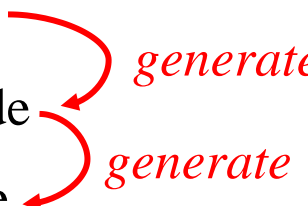
# 1. Introduction: RISC-V (4/4)

**RISC-V revolutionizes not only the open ISA,  
but also the way of hardware coding.**

“*old school*” hardware coding

- RTL (Verilog/VHDL) code

“**RISC-V style**” hardware coding

- Chisel code
  - FIRRTL code
  - Verilog code
- 
- ```
graph LR; A[Chisel code] -- generate --> D[Verilog code]; B[FIRRTL code] -- generate --> D;
```

**To clarify for someone unfamiliar with Chisel-to-Verilog scheme:**

1. Chisel coding is not a **programming** language, it is a **description** language
2. The generated Verilog code from Chisel is a TRUE RTL code
3. With proper settings, that Verilog codes after generated can be brought directly to VLSI tools (Synopsys/Cadence) to make a chip

# Outline

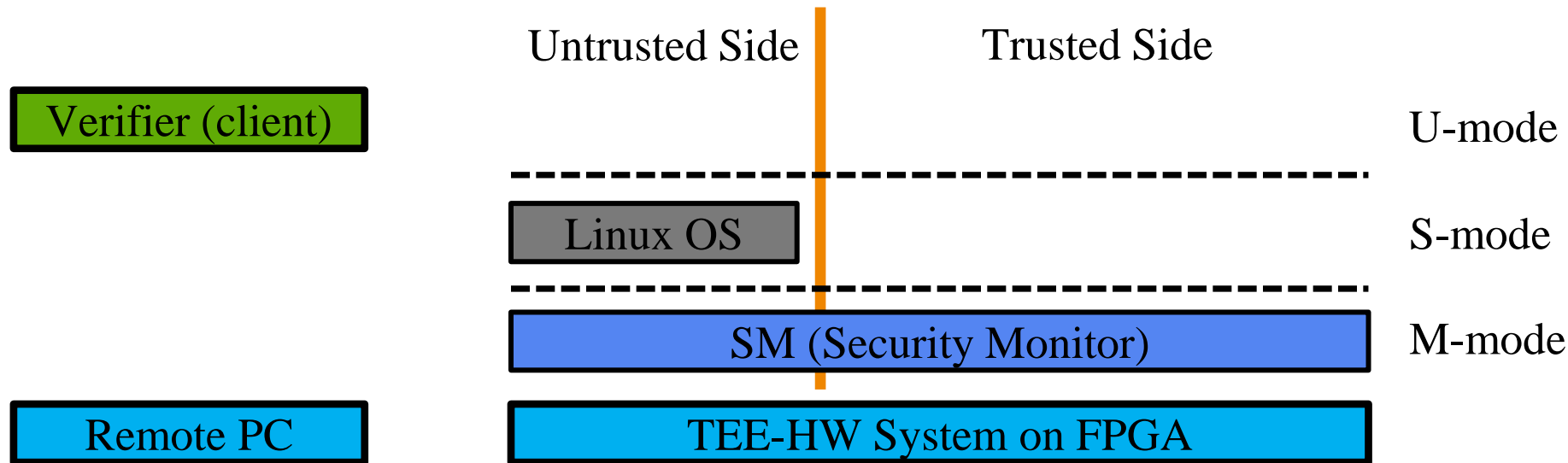
1. Introduction
- 2. Trusted Execution Environment**
3. TEE-Hardware System
4. Results & Conclusion



## 2. Trusted Execution Environment (1/5)

### TEE in-action (*using Keystone: A TEE Framework*)

Remote PC connects to FPGA via Serial (*UART*) terminal or a TCP connection

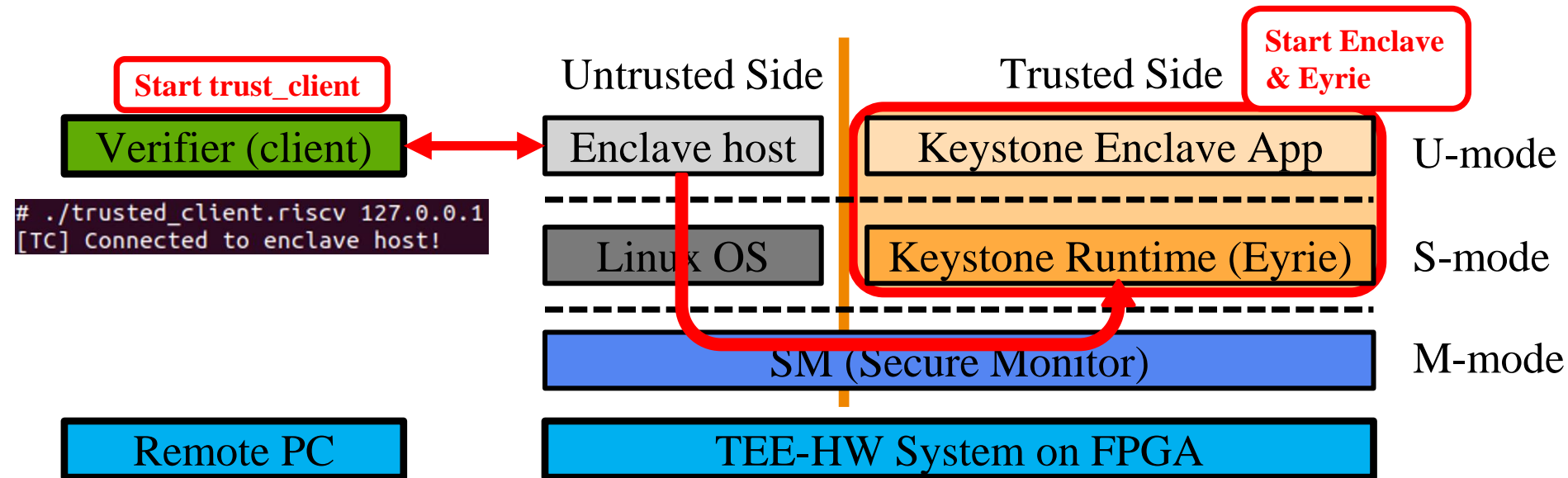


TEE (*Keystone in this case*) creates the Trusted-Side based on the chain-of-trust across multiple operating layers.

It allows client to create and operate an Enclave App in the Trusted Side.

## 2. Trusted Execution Environment (2/5)

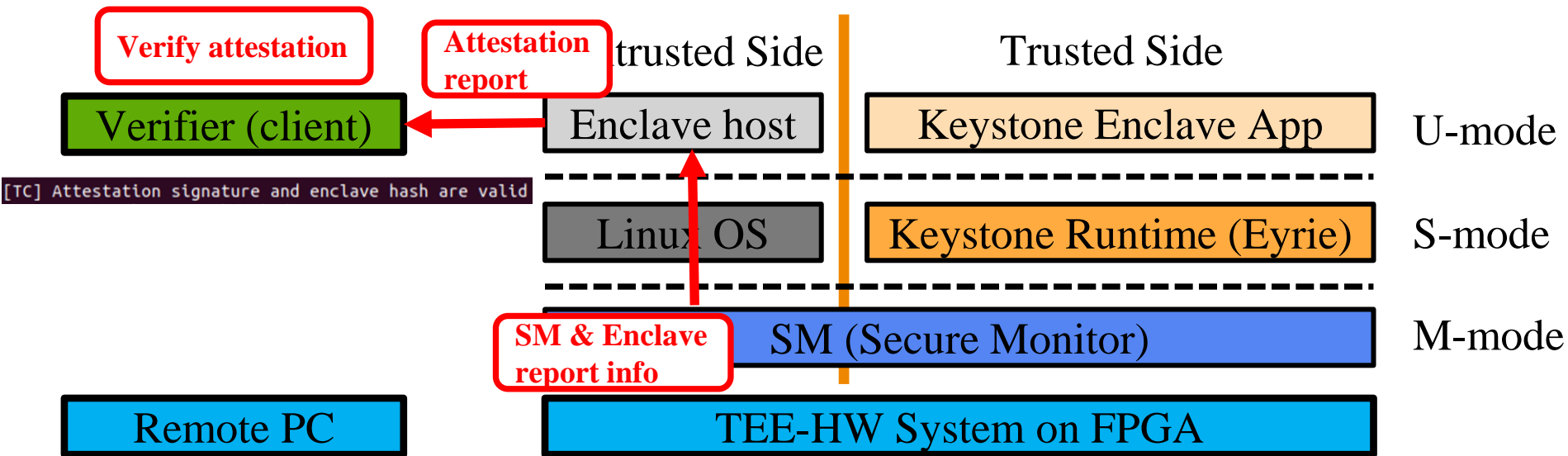
### TEE in-action (*using Keystone: A TEE Framework*)



1. Connection with the Enclave host

## 2. Trusted Execution Environment (3/5)

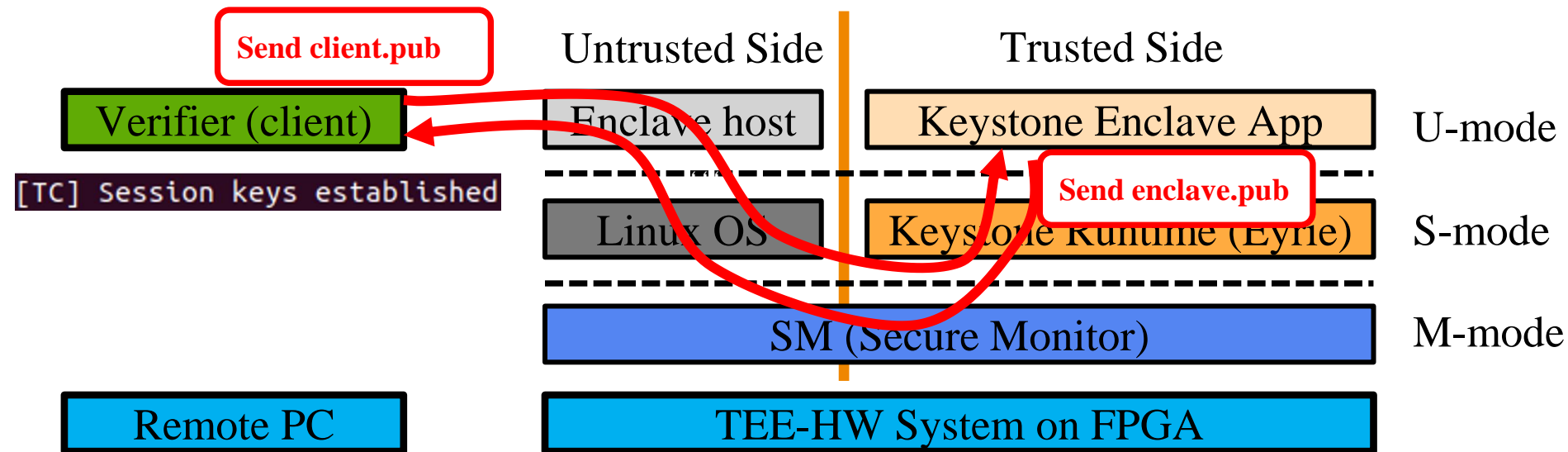
### TEE in-action (*using Keystone: A TEE Framework*)



1. Connection with the Enclave host
2. Verify attestation report

## 2. Trusted Execution Environment (4/5)

### TEE in-action (*using Keystone: A TEE Framework*)

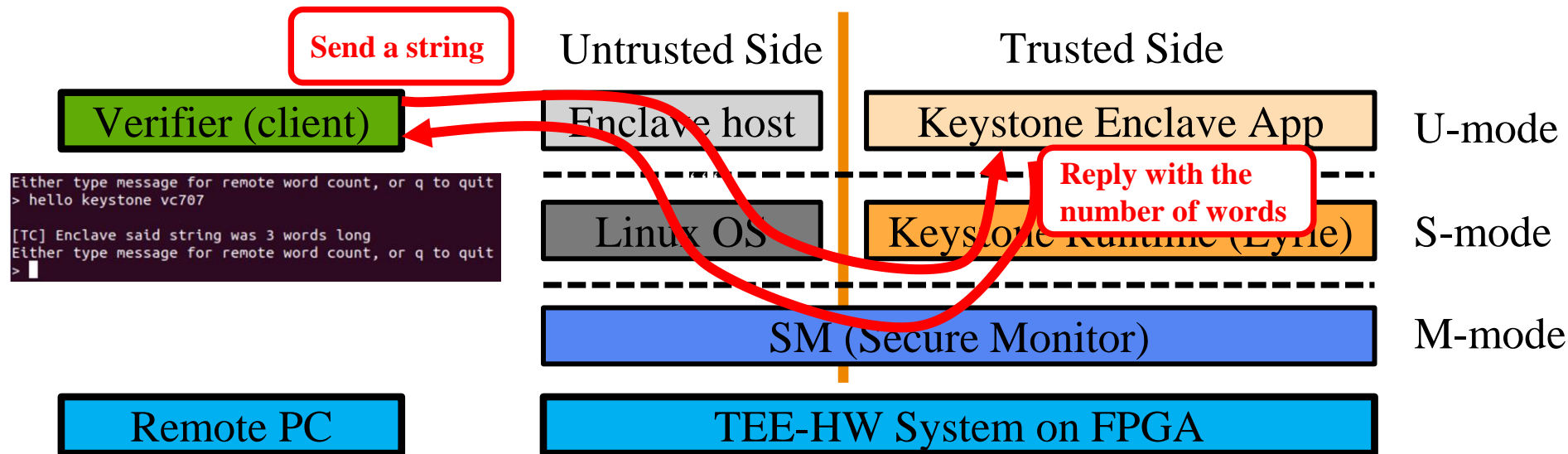


1. Connection with the Enclave host
2. Verify attestation report
3. Exchange communication keys

## 2. Trusted Execution Environment (5/5)

### TEE in-action (*using Keystone: A TEE Framework*)

Keystone demo: (1) client sends strings, then (2) request calculation from the Enclave, finally (3) the Enclave replies with the number of words



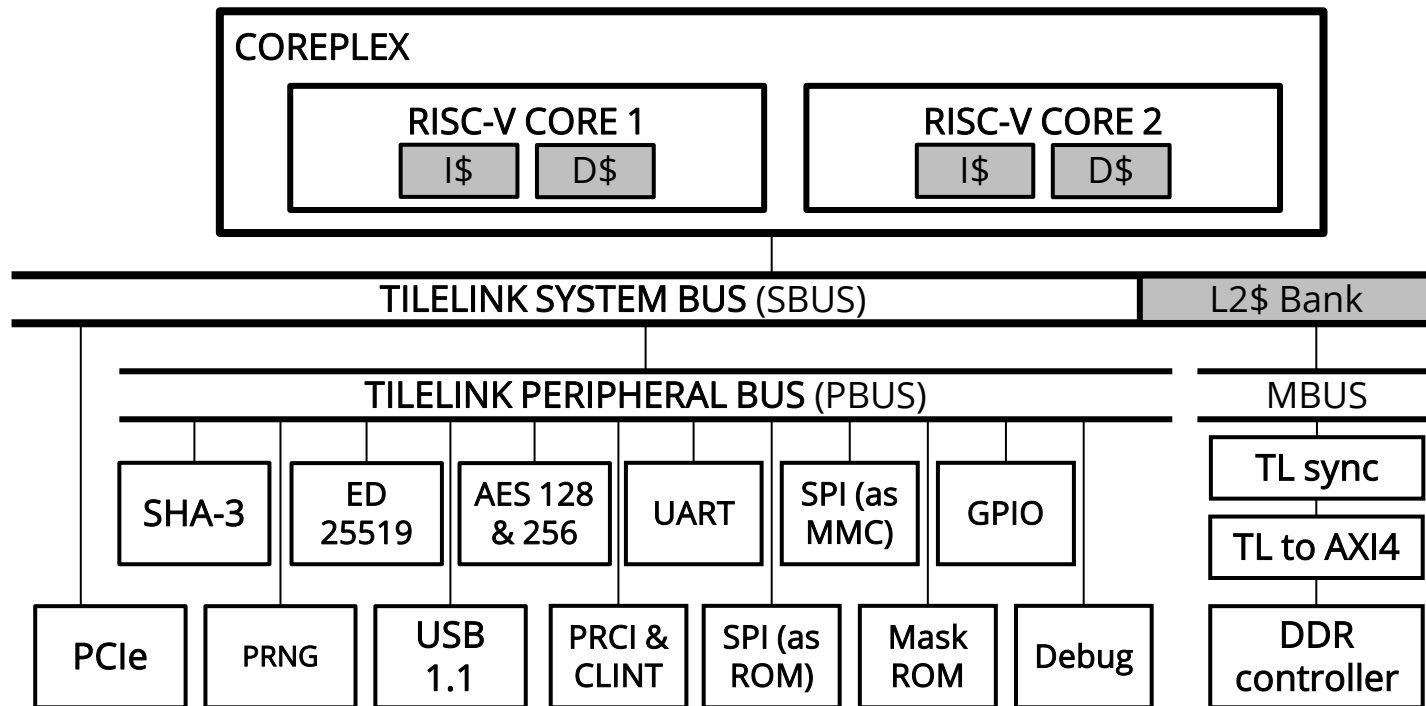
1. Connection with the Enclave host
2. Verify attestation report
3. Exchange communication keys
4. Client's app runs on the established TEE

# Outline

1. Introduction
2. Trusted Execution Environment
- 3. TEE-Hardware System**
4. Results & Conclusion

### 3. TEE-Hardware System (1/6)

#### System Architecture:



- Not fixed at dual-core, can increase/decrease the number of cores as you wanted
- Available cores: ***Rocket-chip*** (in-of-order core) and ***BOOM*** (out-of-order core)
- Some hardware modules can be easily included/excluded to/from the system

### 3. TEE-Hardware System (2/6)

| Variable | Available option                                                                                                       | Description                                                                                                                                                                                                                    |
|----------|------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BOARD    | <ul style="list-style-type: none"> <li>- VC707</li> <li>- DE4</li> <li>- TR4</li> </ul>                                | Select the FPGA board                                                                                                                                                                                                          |
| ISACONF  | <ul style="list-style-type: none"> <li>- RV64GC</li> <li>- RV64IMAC</li> <li>- RV32GC</li> <li>- RV32IMAC</li> </ul>   | Select the ISA                                                                                                                                                                                                                 |
| MBUS     | <ul style="list-style-type: none"> <li>- MBus64</li> <li>- MBus32</li> </ul>                                           | Select the bit-width for the memory bus                                                                                                                                                                                        |
| BOOTSRC  | <ul style="list-style-type: none"> <li>- BOOTROM</li> <li>- QSPI</li> </ul>                                            | Select the boot source                                                                                                                                                                                                         |
| PCIE     | <ul style="list-style-type: none"> <li>- WPCIE</li> <li>- WoPCIE</li> </ul>                                            | <ul style="list-style-type: none"> <li>- Include PCIe module in the system</li> <li>- Remove PCIe module from the system</li> </ul>                                                                                            |
| DDRCLK   | <ul style="list-style-type: none"> <li>- WSepaDDRClk</li> <li>- WoSepaDDRClk</li> </ul>                                | <ul style="list-style-type: none"> <li>- Separate DDR-clock with System-clock</li> <li>- Not separate DDR-clock with System-clock</li> </ul>                                                                                   |
| HYBRID   | <ul style="list-style-type: none"> <li>- Rocket</li> <li>- Boom</li> <li>- RocketBoom</li> <li>- BoomRocket</li> </ul> | <ul style="list-style-type: none"> <li>- Two Rocket cores</li> <li>- Two Boom cores</li> <li>- Rocket core 1<sup>st</sup>, Boom core 2<sup>nd</sup></li> <li>- Boom core 1<sup>st</sup>, Rocket core 2<sup>nd</sup></li> </ul> |

In the Makefile system, these variables are available.

Example usage:

```
BOARD=VC707
ISACONF=RV64GC
MBUS=MBus64
BOOTSRC=BOOTROM
PCIE=WoPCIE
DDRCLK=WoSepaDDRClk
HYBRID=Rocket
```

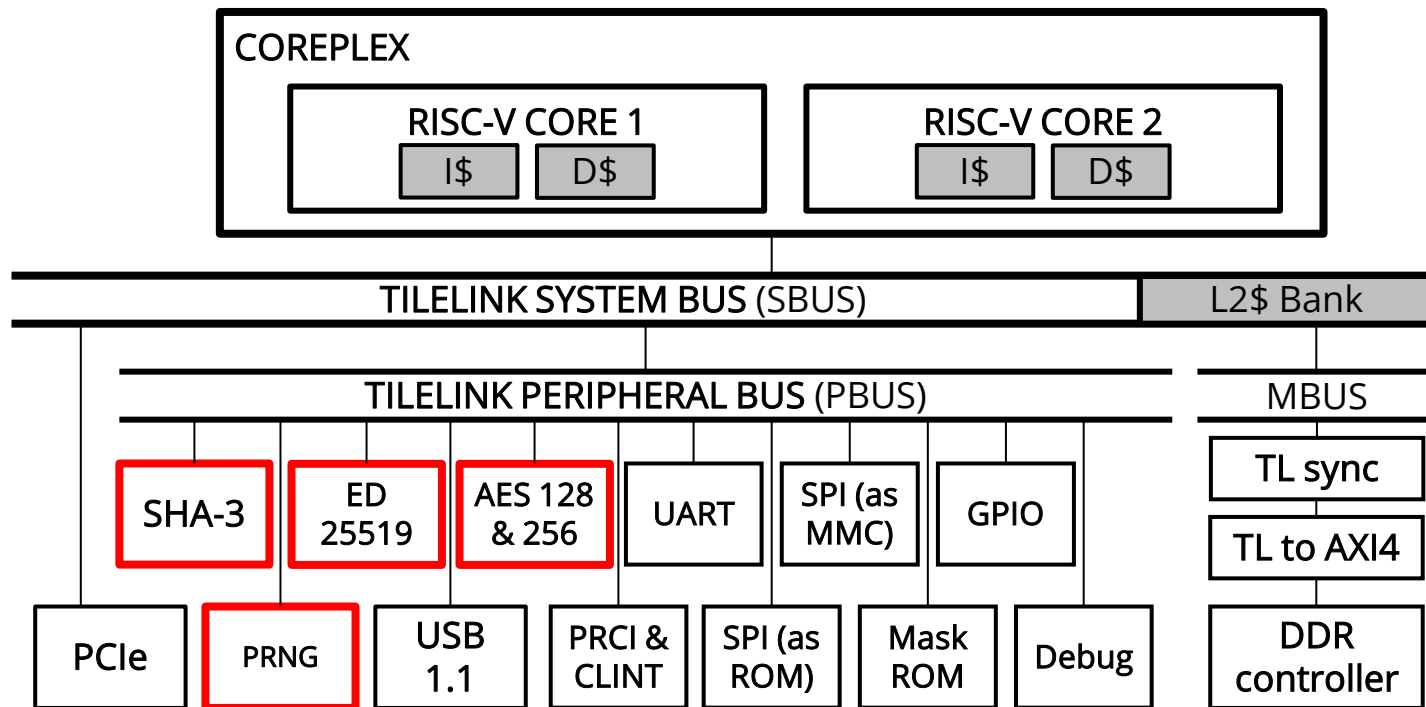


### 3. TEE-Hardware System (3/6)

Summary table of FPGA logic utilization (*on VC707*) with various core configurations:

| ISACONF  | HYBRID |        | FPGA logic utilization<br>(LUT) ( <i>on VC707</i> ) |        |
|----------|--------|--------|-----------------------------------------------------|--------|
|          | Core0  | Core1  |                                                     |        |
| RV64GC   | Boom   | Boom   | 160,873                                             | 52.99% |
|          | Rocket | Rocket | 96,571                                              | 31.81% |
|          | Boom   | Rocket | 128,708                                             | 42.39% |
|          | Rocket | Boom   | 128,719                                             | 42.40% |
| RV64GC   | Rocket | Rocket | 96,571                                              | 31.81% |
| RV64IMAC |        |        | 72,007                                              | 23.72% |
| RV32GC   |        |        | 89,356                                              | 29.43% |
| RV32IMAC |        |        | 65,899                                              | 21.71% |

### 3. TEE-Hardware System (4/6)



- Crypto-cores:**
- SHA-3 512
  - Ed25519 (*genkey and certification*)
  - AES-128/256
  - PRNG (*Pseudo-random generator*)

### 3. TEE-Hardware System (5/6)

## Crypto-cores on Stratix-IV FPGA

|            | SHA-3 | AES-128/256 | Ed25519 |       |
|------------|-------|-------------|---------|-------|
|            |       |             | Mult    | Sign  |
| ALUT       | 8,108 | 3,195       | 2,737   | 3,969 |
| Registers  | 2,790 | 2,854       | 4,778   | 4,617 |
| Fmax (MHz) | 100   | 100         | 100     | 100   |
| Memory     | 0     | 0           | 8KB     | 0     |
| DSP block  | 0     | 0           | 48      | 0     |

### 3. TEE-Hardware System (6/6)

#### Crypto-cores in ASIC (*ROHM-180nm*)

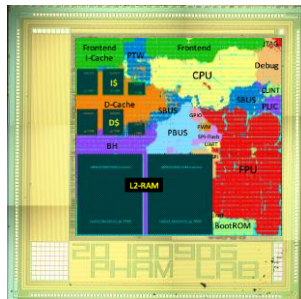
|                      | SHA-3                                   | AES-128/256                              | Ed25519                                       |                                               |
|----------------------|-----------------------------------------|------------------------------------------|-----------------------------------------------|-----------------------------------------------|
|                      |                                         |                                          | Mult                                          | Sign                                          |
| Size                 | 1,150 $\mu$ m $\times$<br>1,150 $\mu$ m | 808.96 $\mu$ m $\times$<br>806.4 $\mu$ m | 1,694.72 $\mu$ m $\times$<br>1,693.44 $\mu$ m | 1,346.56 $\mu$ m $\times$<br>1,345.68 $\mu$ m |
| Gate-count<br>(NAND) | 102,500                                 | 50,560                                   | 222,432                                       | 140,442                                       |
| Fmax (MHz)           | 104                                     | 90                                       | 106                                           | 91                                            |
| Power (mW)           | 42.745                                  | 37.566                                   | 53.061                                        | 80.894                                        |

# Outline

1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Results & Conclusion

## 6. Results & Conclusion (1/3)

ROHM180 5.0×5.0mm



RV64GC  
Rocket-  
chip(×1)

64-bit MCU

2019

Sep.

Aug.

TEE-HW



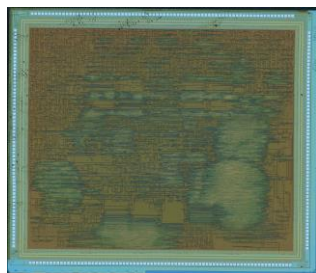
RV64GC  
Rocket-  
chip(×4)  
*Add:* Linux-  
compatible  
ROHM180  
5.0×7.0mm

2020

Oct.

Jan.

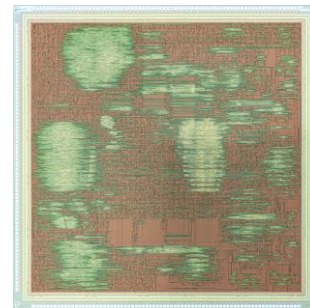
TEE-HW



RV64GC  
Rocket-  
chip(×2)  
*Add:*  
Crypto  
cores

ROHM180 5.0×5.0mm

ROHM180 5.0×5.0mm

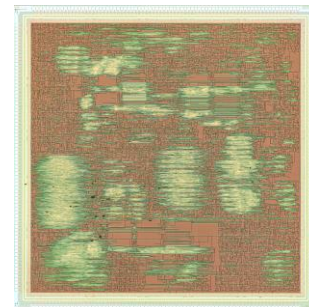


RV64GC  
BOOM(×1)  
*Add:*  
BOOM  
core

TEE-HW

Jun.

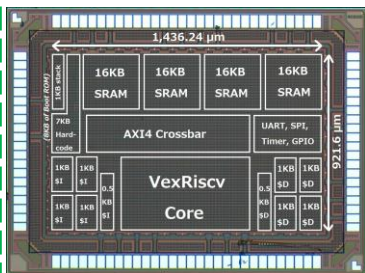
TEE-HW



RV32IMAC  
Rocket-  
chip(×1)+  
BOOM(×1)  
*Add:* 32-bit  
system

ROHM180 5.0×5.0mm

32-bit MCU



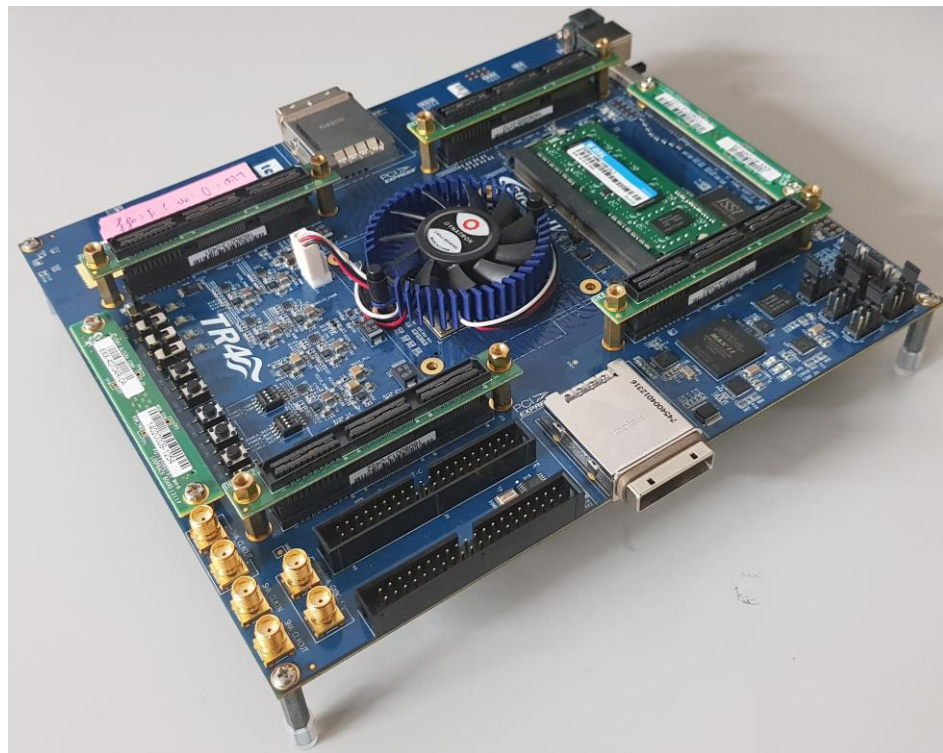
RV32IM  
VexRiscv(×1)

SOTB65 2.0×1.5mm

## 6. Results & Conclusion (2/3)

Solving the DDR problem (*for Linux-boot*) for the chip by:

1. Using the DIMM RAM in the TR4
2. Having the PCB (*with socket-chip*) mounted on the TR4



## 6. Results & Conclusion (3/3)

- We presented a system platform for Trusted Execution Environment (TEE) featuring crypto-cores accelerators.
- Completed TEE-Hardware system was developed with various configurations to fit specific needs;  
*such as core options, hybrid options, ISA options, etc.*
- The system was implemented and tested on various FPGAs (*VC707, DE4, TR4*) and ASIC (*ROHM-180nm*).
- The execution time of the TEE with hardware accelerators dropped significantly compared to software.





国立大学法人  
電気通信大学



# THANK YOU FOR YOUR LISTENING

## Acknowledgement

This work is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

The chip tape-out is supported by VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Synopsys, Inc., Cadence Design Systems, Inc., Renesas Electronics Corp., and Nippon Systemware Co., Ltd.