Ninja: Towards Transparent Tracing and Debugging on ARM

Zhenyu Ning & Fengwei Zhang
Wayne State University
{zhenyu.ning, fengwei}@wayne.edu

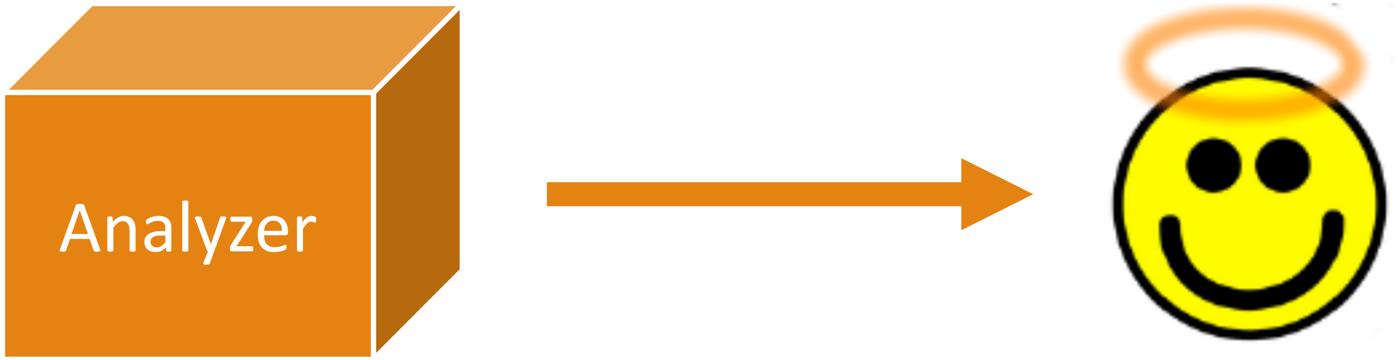
Outline

- Introduction
- Background
- System Overview
- Evaluation
- Conclusion

Outline

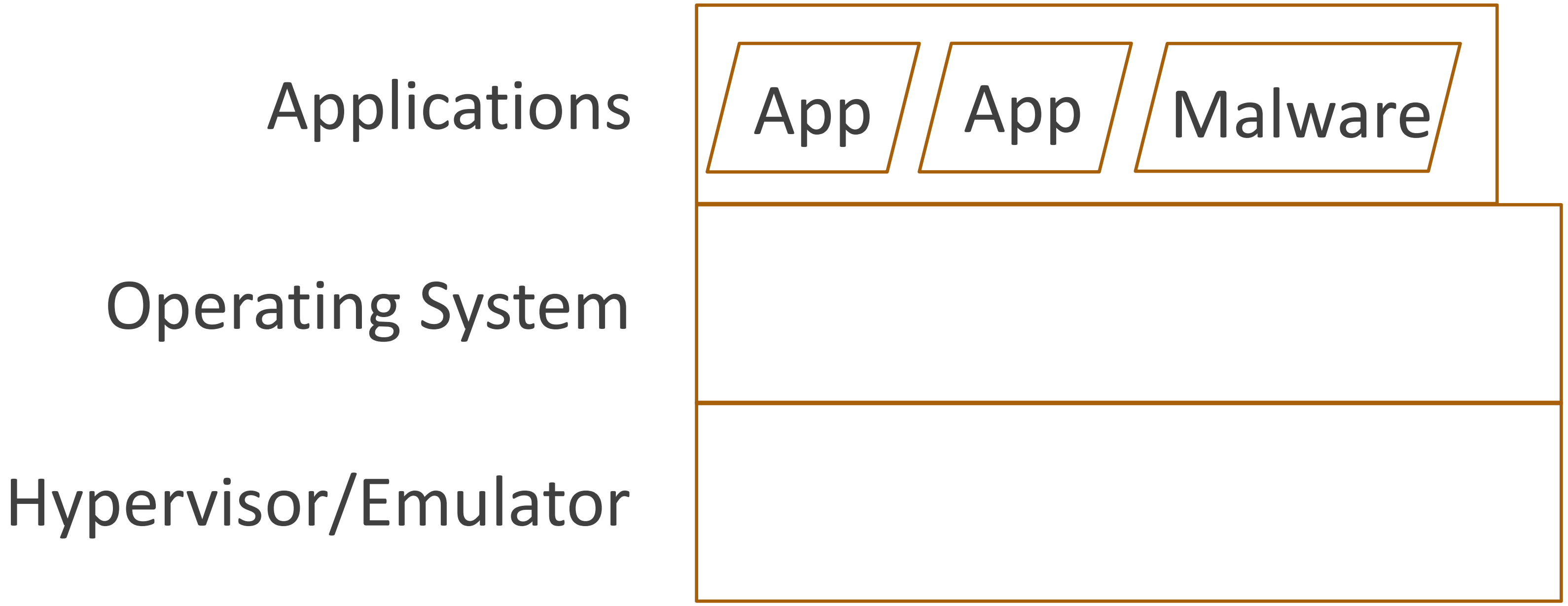
- Introduction
- Background
- System Overview
- Evaluation
- Conclusion

Evasion Malware

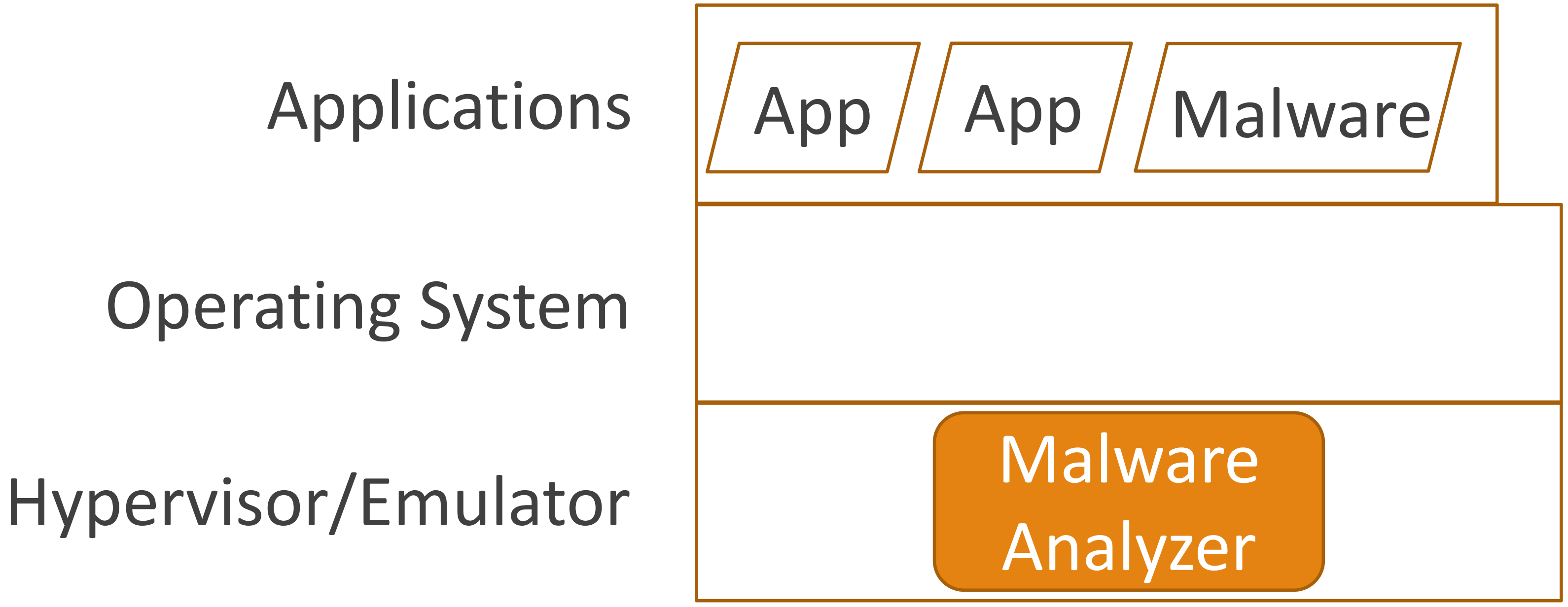


Evasion Malware

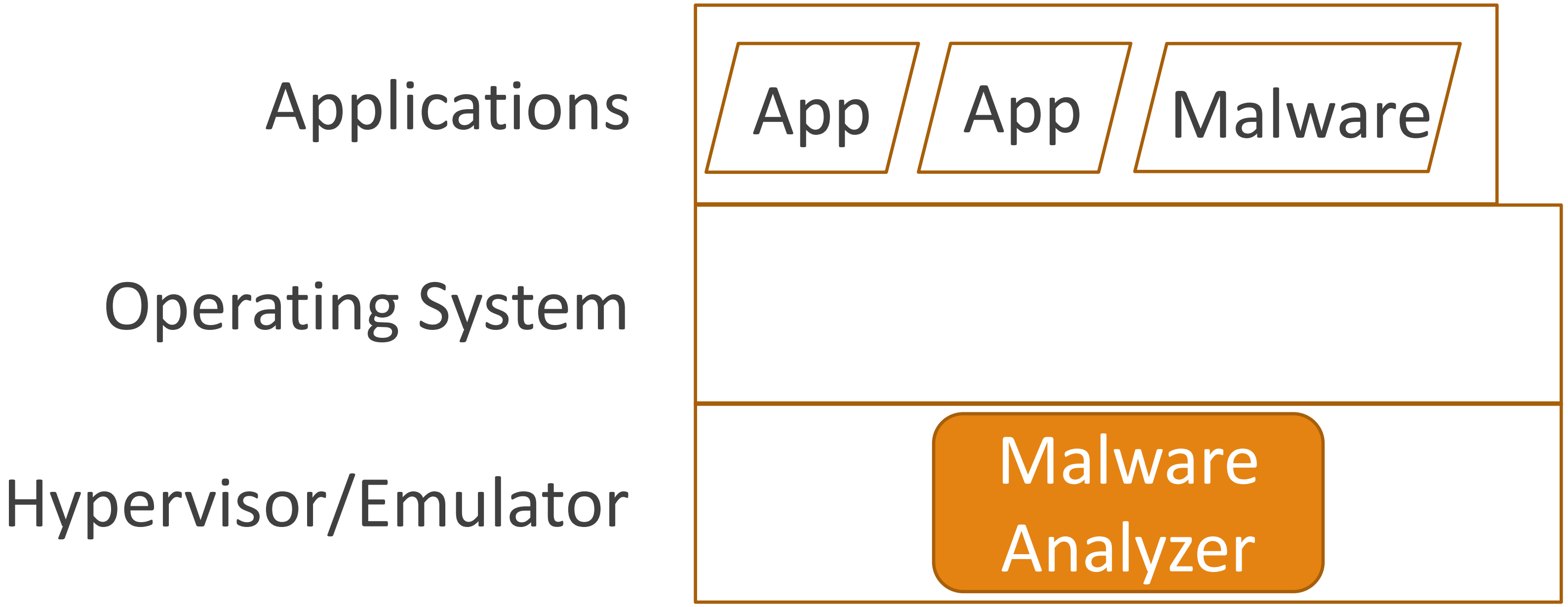
Malware Analysis



Malware Analysis



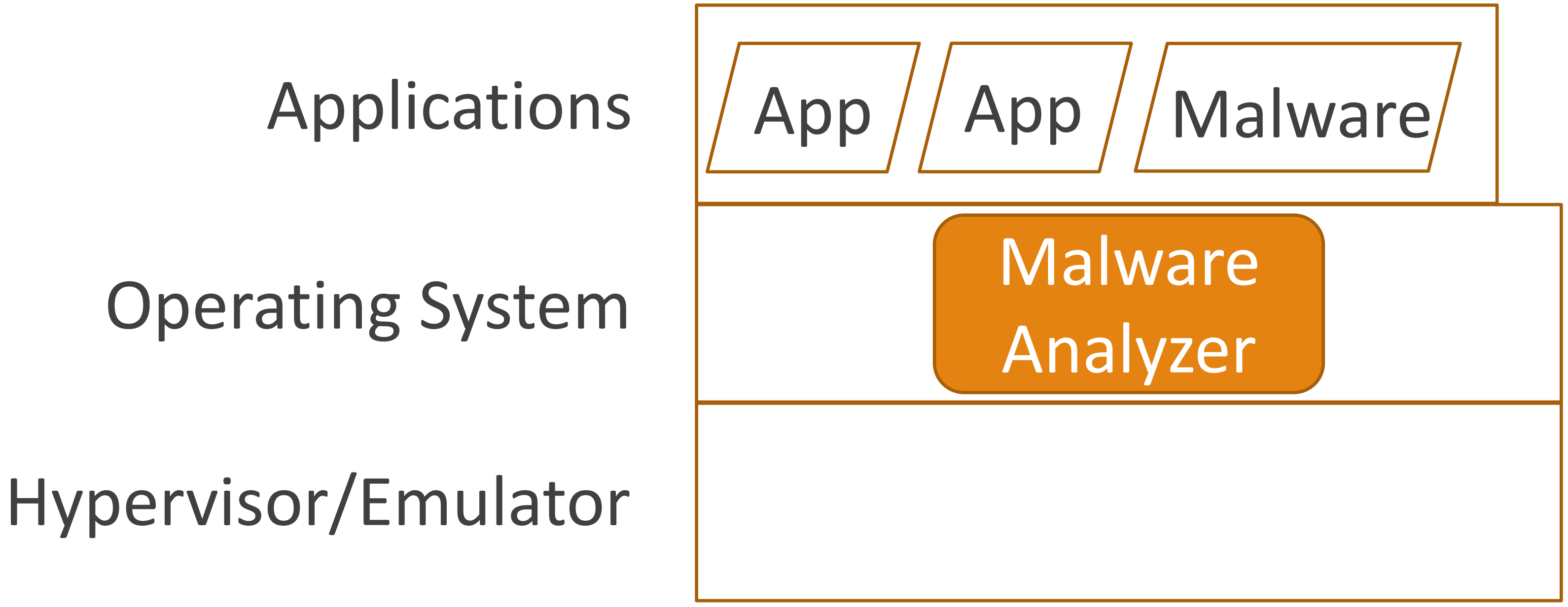
Malware Analysis



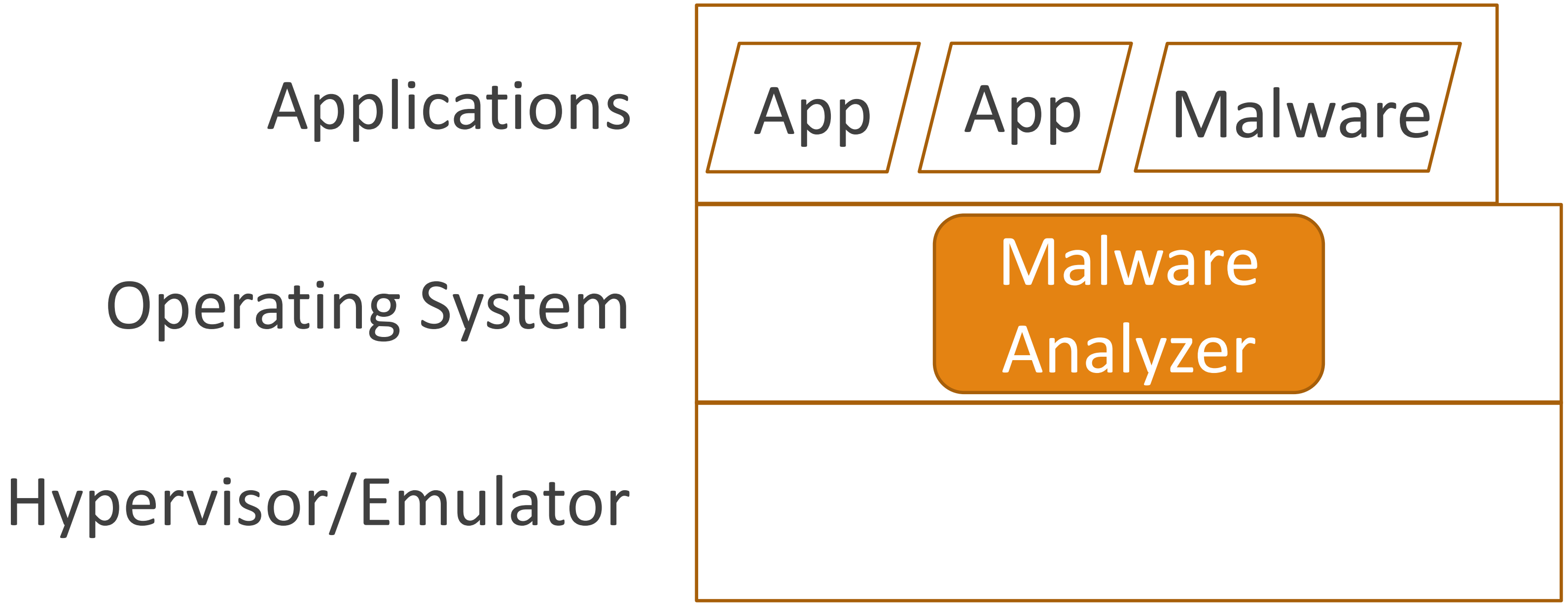
Limitation:

- Unarmed to anti-virtualization or anti-emulation techniques

Malware Analysis



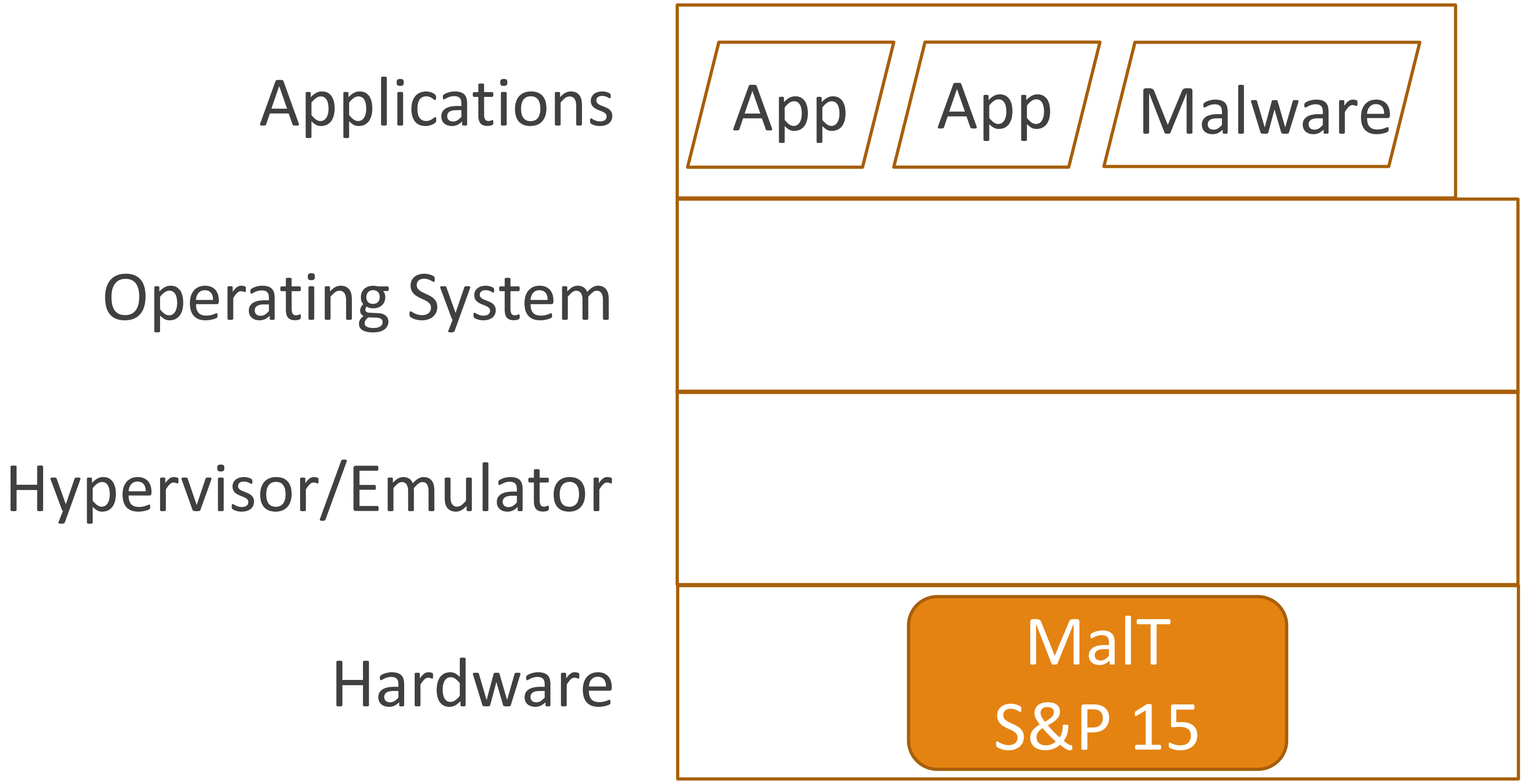
Malware Analysis



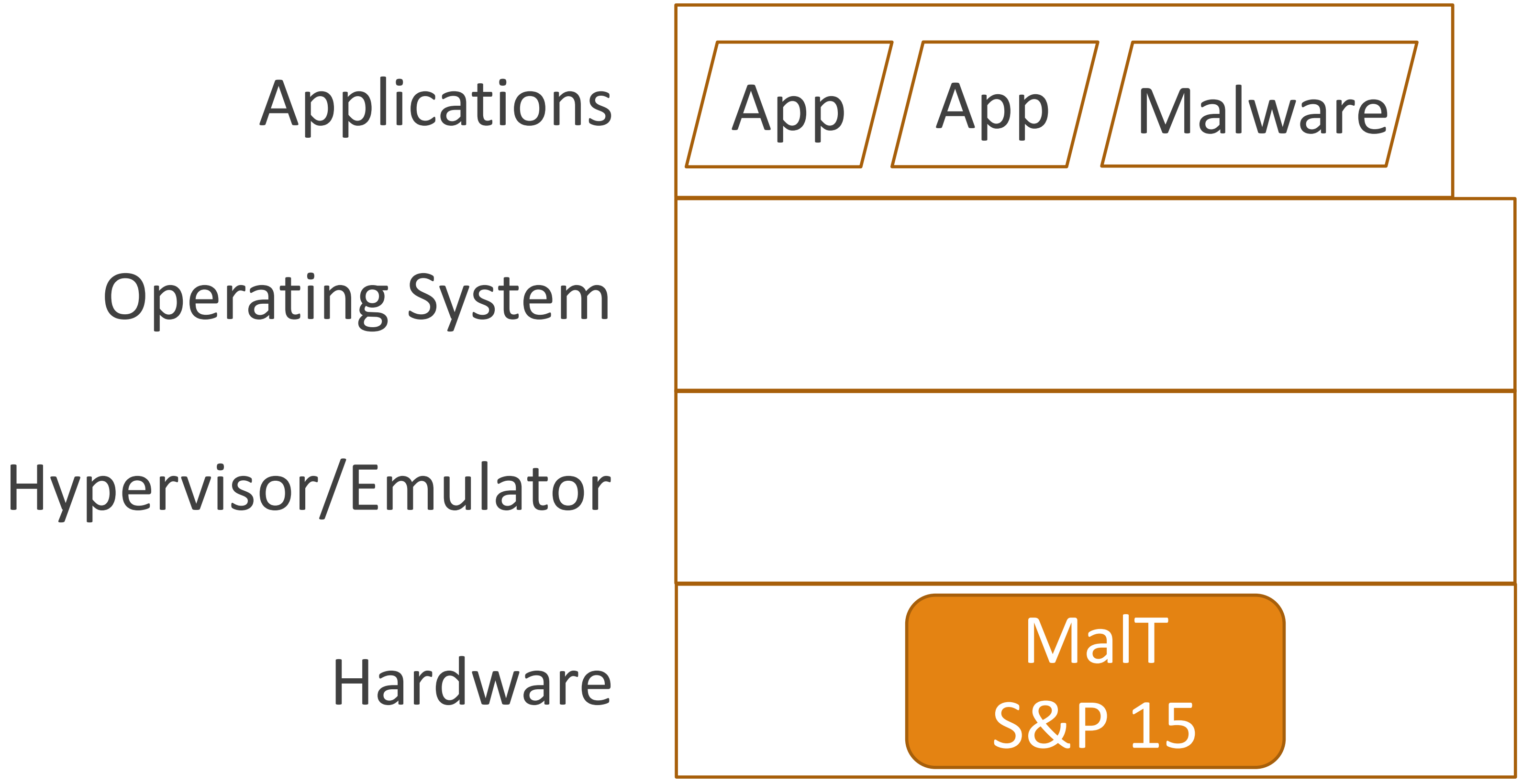
Limitation:

- Unable to handle malware with high privilege (e.g., rootkits)

Malware Analysis



Malware Analysis



Limitations:

- High performance overhead on mode switch
- Unprotected modified registers
- Vulnerable to external timing attack

Transparency Requirements

- An ***Environment*** that provides the access to the states of the target malware

- An ***Analyzer*** which is responsible for the further analysis of the states

Transparency Requirements

- An ***Environment*** that provides the access to the states of the target malware
  - It is isolated from the target malware
  - It exists on an off-the-shelf (OTS) bare-metal platform
- An ***Analyzer*** which is responsible for the further analysis of the states

Transparency Requirements

- An ***Environment*** that provides the access to the states of the target malware
  - It is isolated from the target malware
  - It exists on an off-the-shelf (OTS) bare-metal platform
- An ***Analyzer*** which is responsible for the further analysis of the states
  - It should not leave any detectable footprints to the outside of the environment

Outline

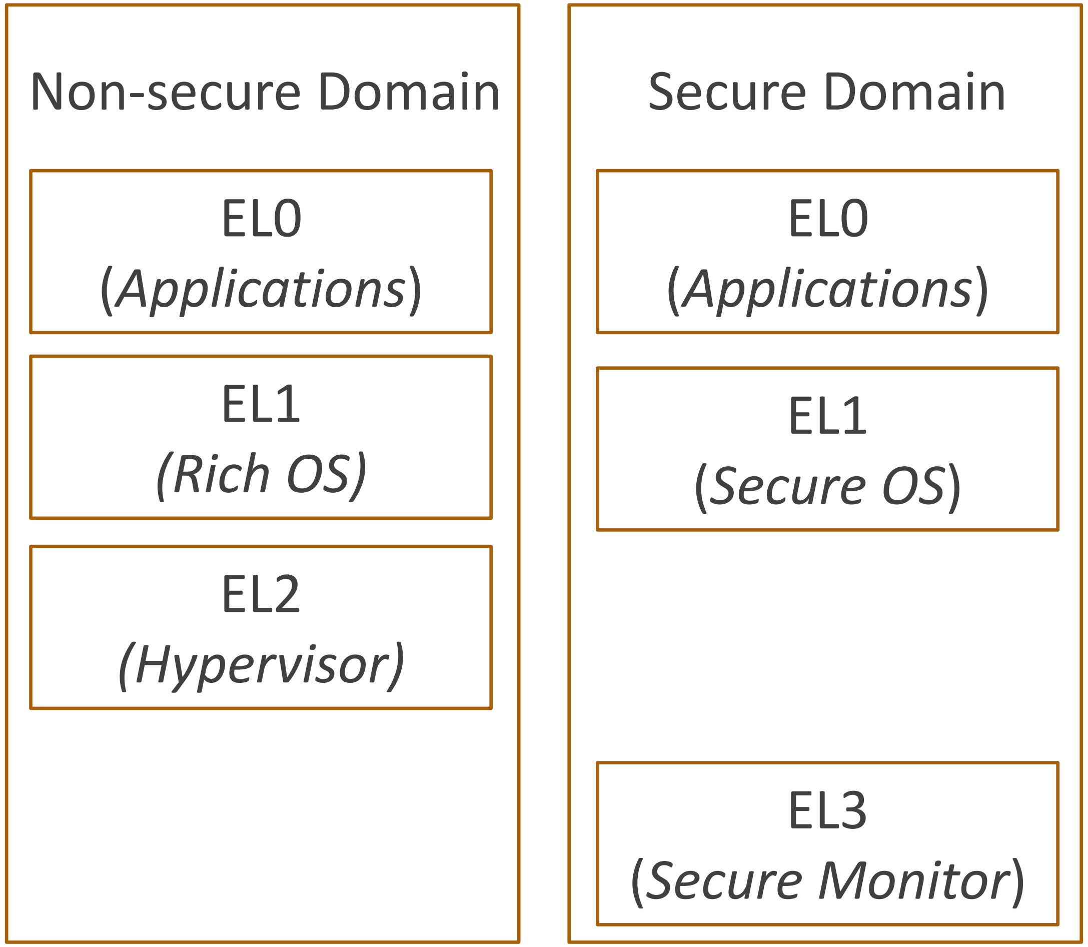
- Introduction
- Background
- System Overview
- Evaluation
- Conclusion

Background - TrustZone

ARM TrustZone technology divides the execution environment into secure domain and non-secure domain.

- The RAM is partitioned to secure and non-secure region.
- The interrupts are assigned into secure or non-secure group.
- Secure-sensitive registers can only be accessed in secure domain.
- Hardware peripherals can be configured as secure access only.

Background - TrustZone



Non-secure Domain

EL0 (*Applications*)

EL1 *(Rich OS)*

EL2 *(Hypervisor)*

Secure Domain

EL0 (*Applications*)

EL1 (*Secure OS*)

EL3 (*Secure Monitor*)

- In ARMv8 architecture, exceptions are delivered to different Exception Levels (ELs).
- The only way to enter the secure domain is to trigger a EL3 exception.
- The exception return instruction (ERET) can be used to switch back to the non-secure domain.

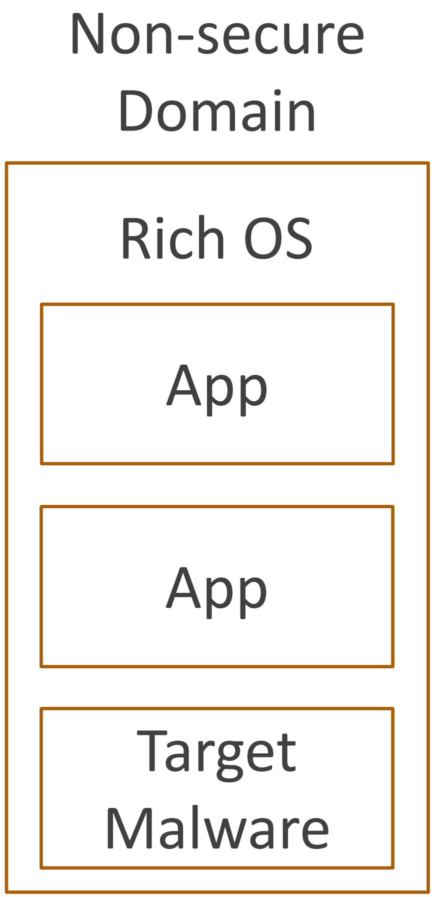
Background – PMU and ETM

- The Performance Monitor Unit (PMU) leverages a set of performance counter registers to count the occurrence of different CPU events.
- The Embedded Trace Macrocell (ETM) traces the instructions and data of the system, and output the trace stream into pre-allocated buffers on the chip.
- Both PMU and ETM exist on ARM Cortex-A5x and Cortex-A7x series CPUs, and do NOT affect the performance of the CPU.

Outline

- Introduction
- Background
- System Overview
- Evaluation
- Conclusion

Overview



Non-secure Domain

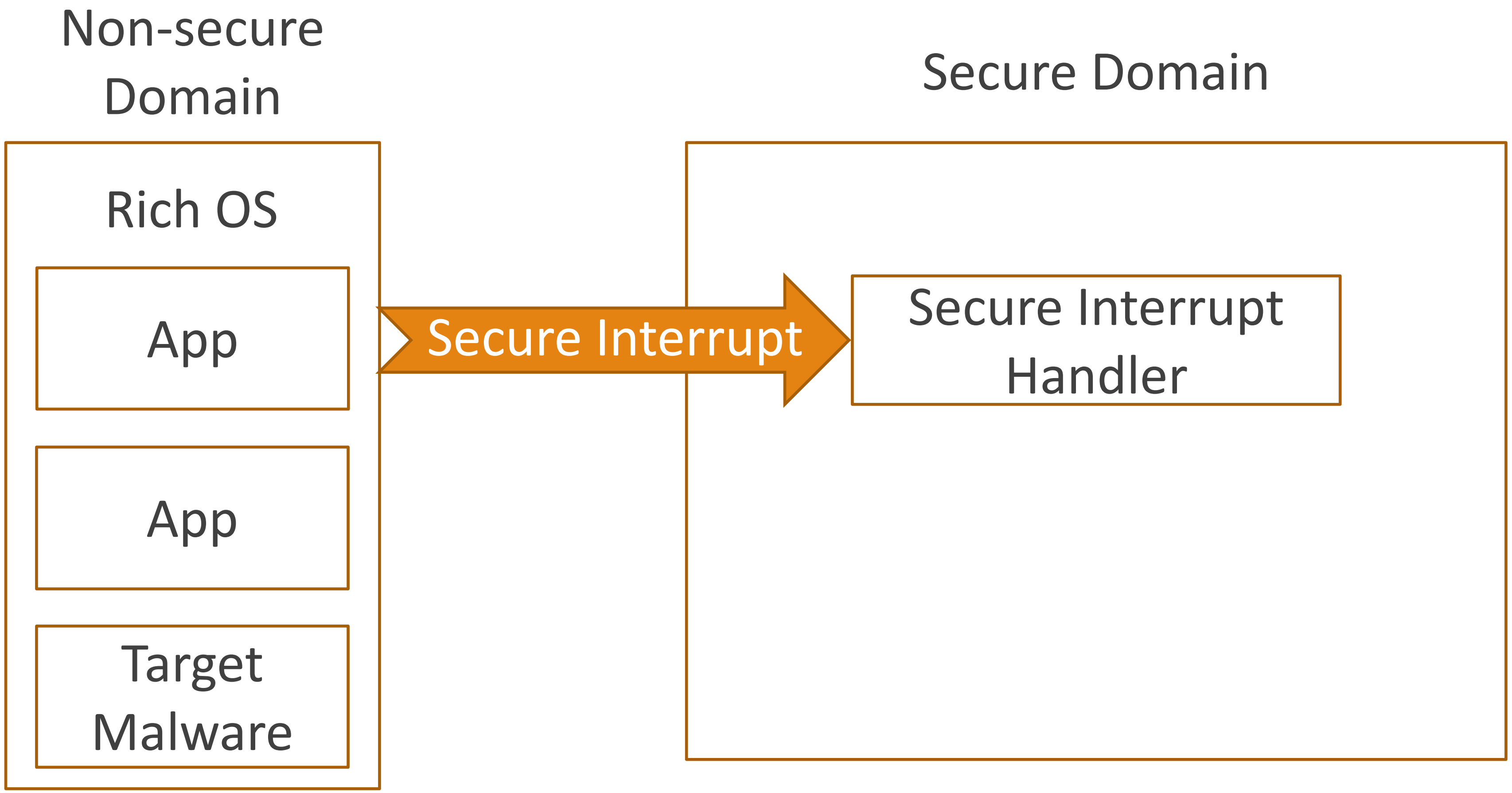
Rich OS

App

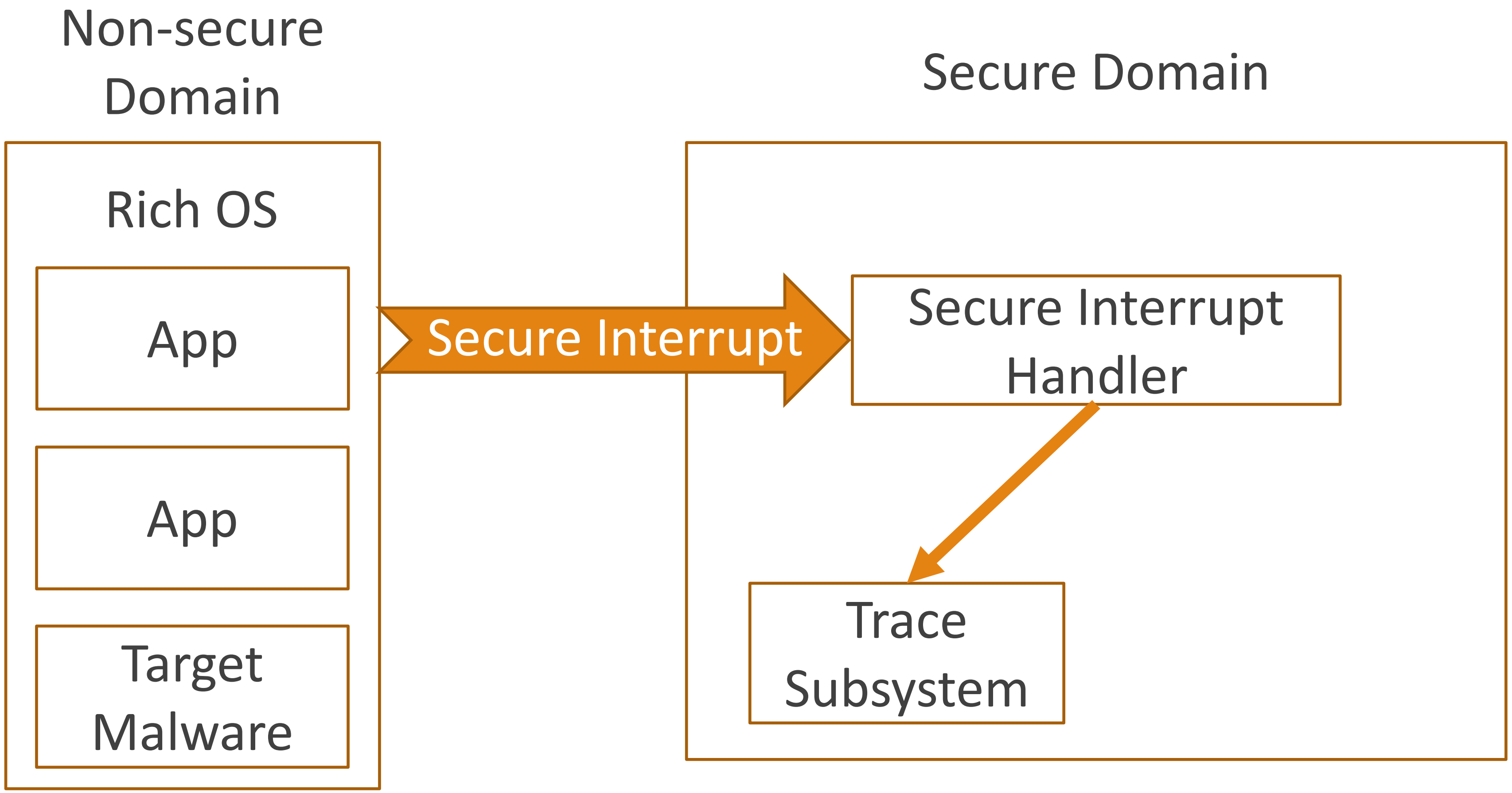
App

Target Malware

Overview


Non-secure Domain
Rich OS
App
App
Target Malware
Secure Interrupt
Secure Domain
Secure Interrupt Handler

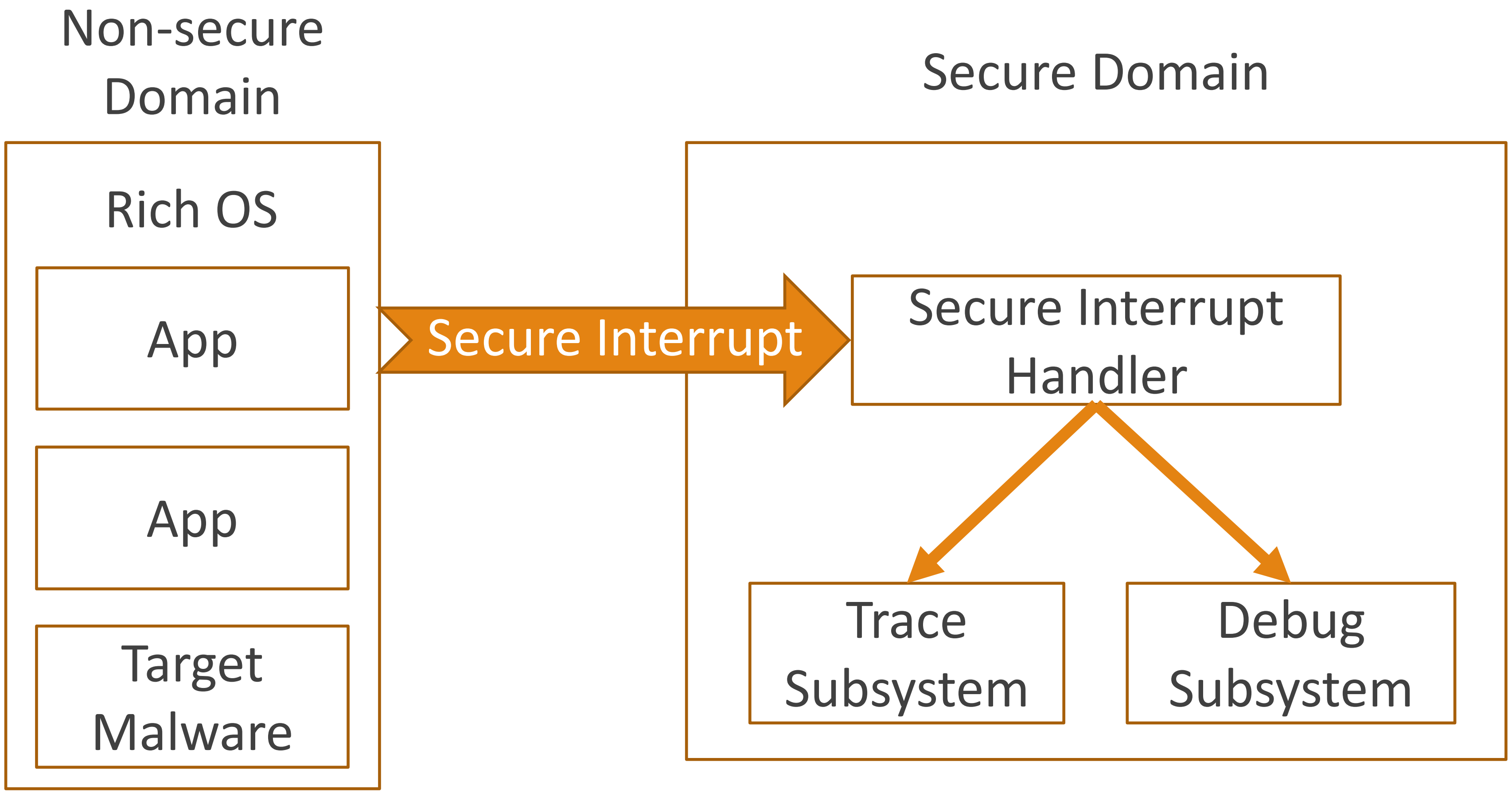
Overview



Trace Subsystem:

- Instruction Trace
- System Call Trace
- Android API Trace

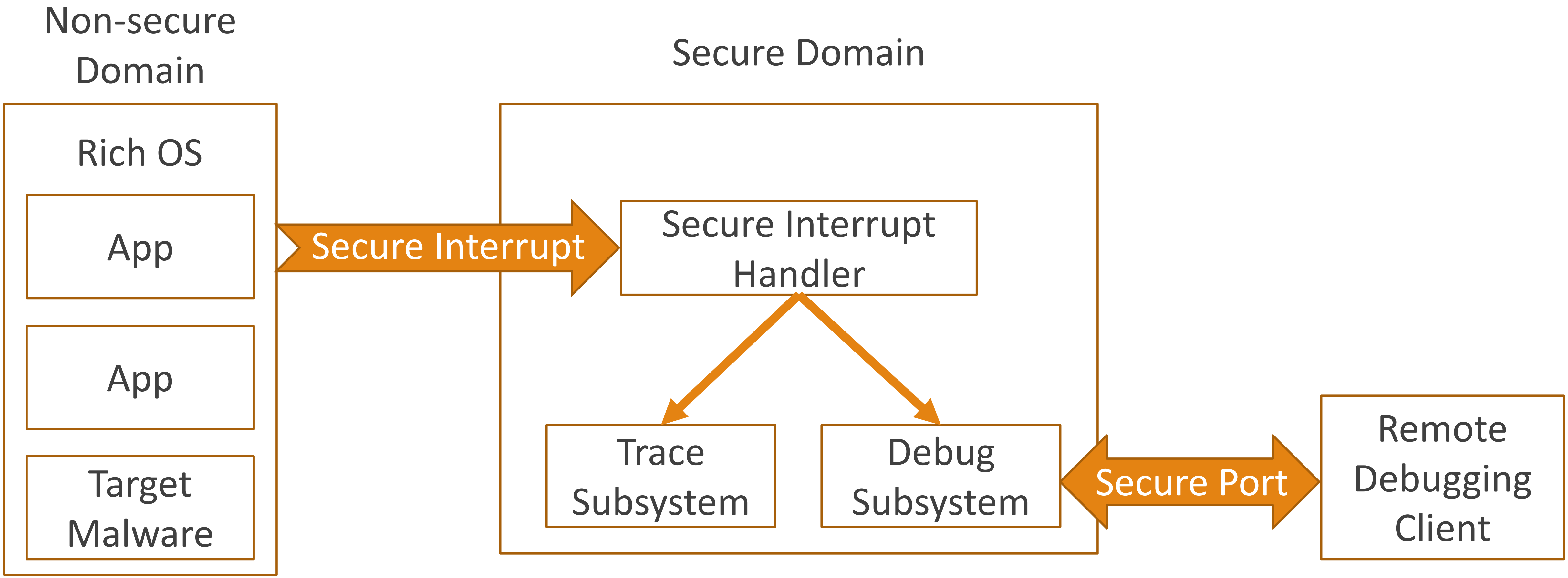
Overview



Debug Subsystem:

- Single Stepping
- Breakpoints
- Memory R/W

Overview



Non-secure Domain

Rich OS

App

App

Target Malware

Secure Interrupt

Secure Domain

Secure Interrupt Handler

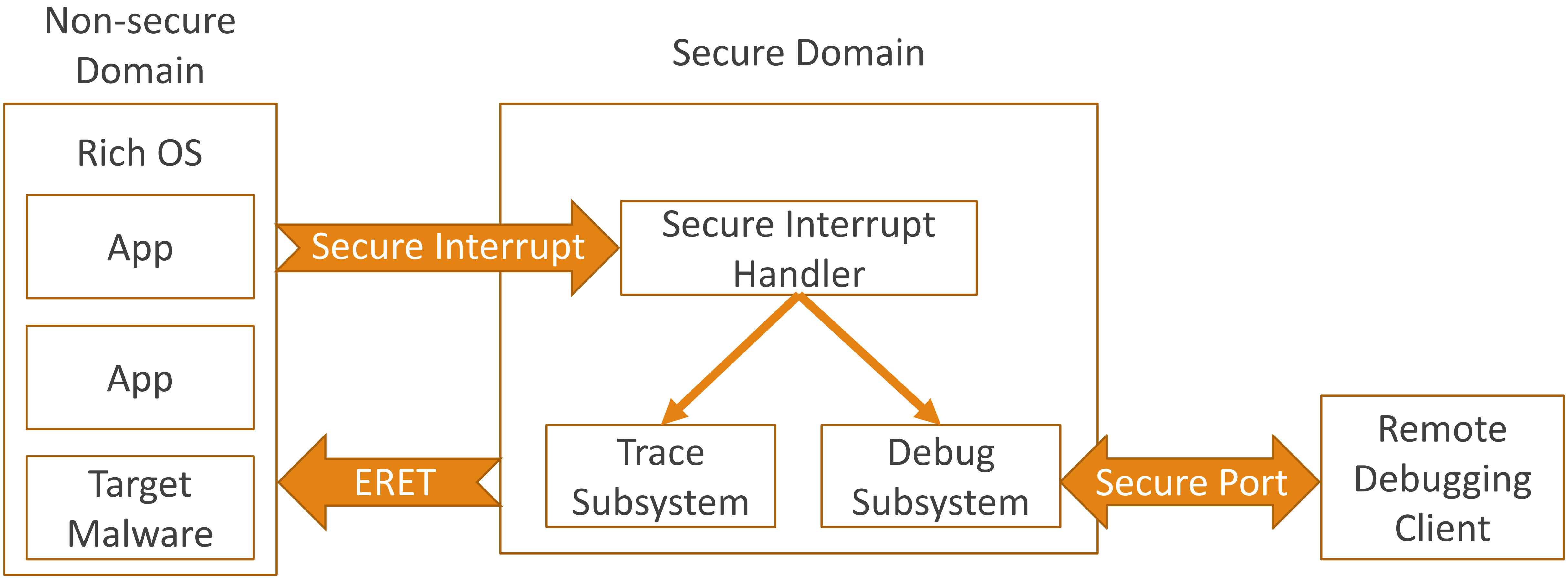
Trace Subsystem

Debug Subsystem

Secure Port

Remote Debugging Client

Overview



Non-secure Domain

Rich OS

App

App

Target Malware

Secure Interrupt

ERET

Secure Domain

Secure Interrupt Handler

Trace Subsystem

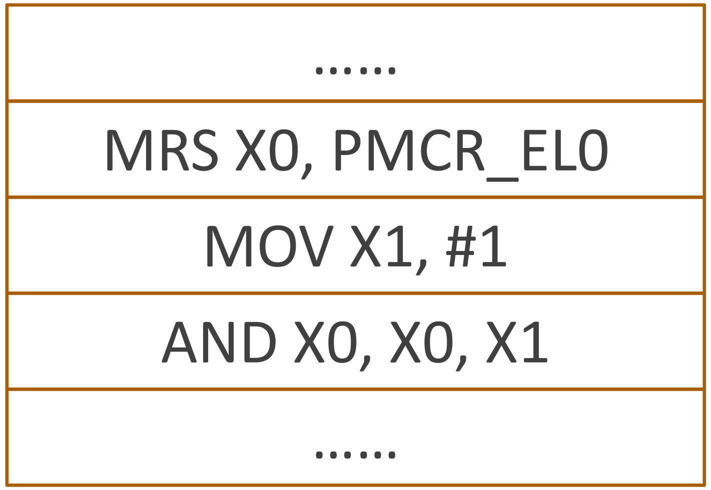
Debug Subsystem

Secure Port

Remote Debugging Client

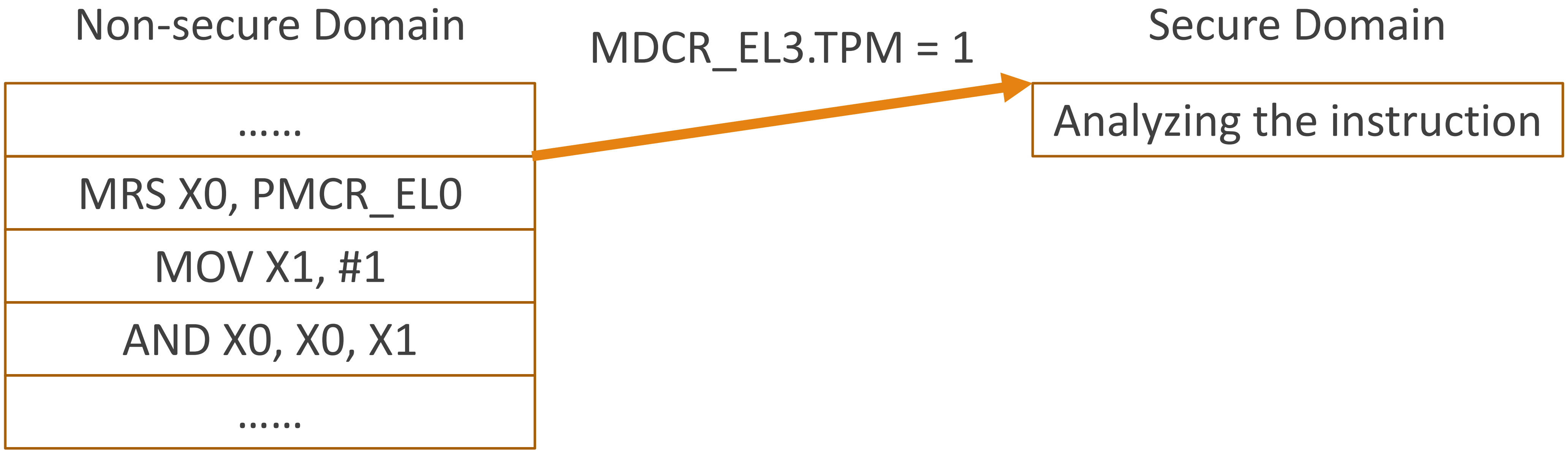
Hardware Traps

Non-secure Domain



```
......
MRS X0, PMCR_EL0
MOV X1, #1
AND X0, X0, X1
......
```

Hardware Traps



Non-secure Domain

……

MRS X0, PMCR_EL0

MOV X1, #1

AND X0, X0, X1

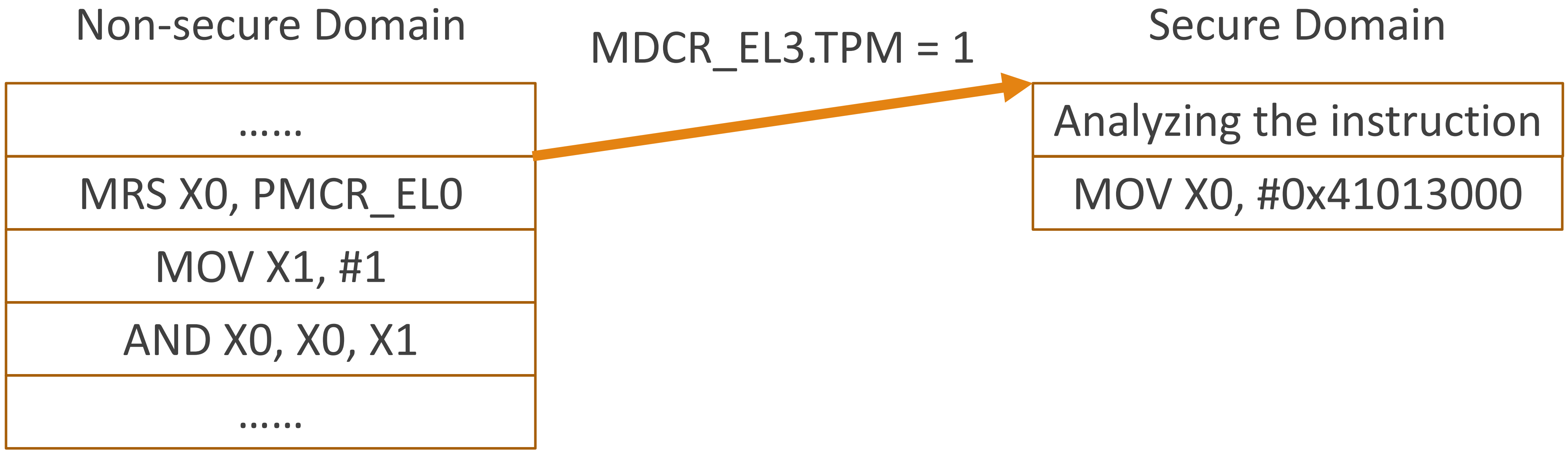
……

MDCR_EL3.TPM = 1

Secure Domain

Analyzing the instruction

Hardware Traps



Non-secure Domain

```
……
MRS X0, PMCR_EL0
MOV X1, #1
AND X0, X0, X1
……
```

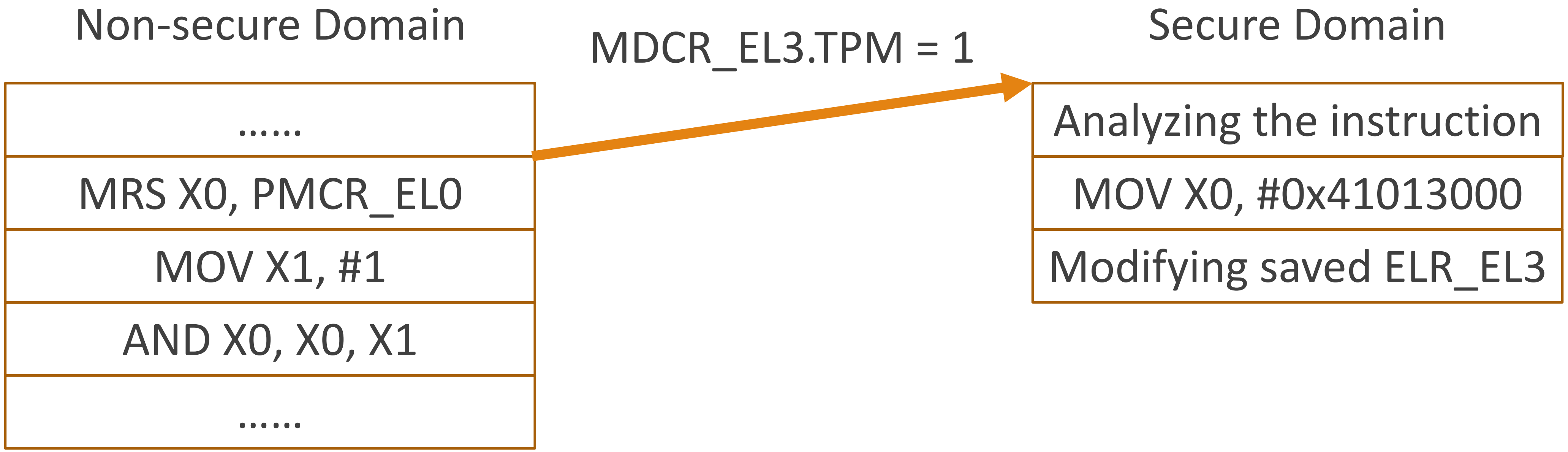
MDCR_EL3.TPM = 1

Secure Domain

Analyzing the instruction

```
MOV X0, #0x41013000
```

Hardware Traps



Non-secure Domain

```
……
MRS X0, PMCR_EL0
MOV X1, #1
AND X0, X0, X1
……
```

MDCR_EL3.TPM = 1

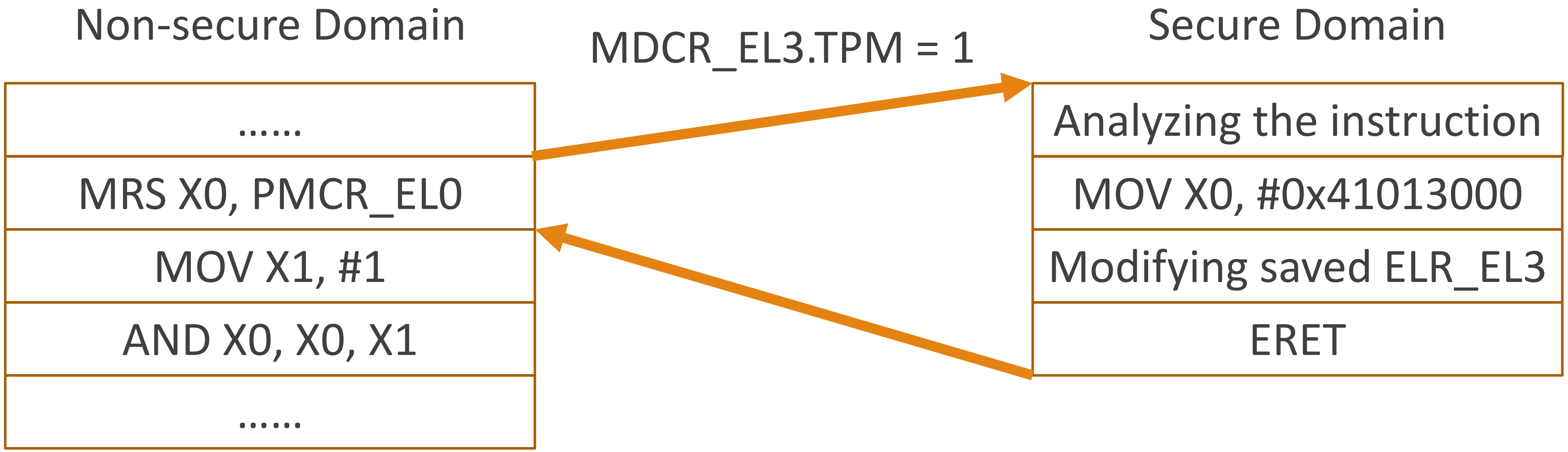
Secure Domain

Analyzing the instruction

```
MOV X0, #0x41013000
```

Modifying saved ELR_EL3

Hardware Traps



Non-secure Domain

……

MRS X0, PMCR_EL0

MOV X1, #1

AND X0, X0, X1

……

MDCR_EL3.TPM = 1

Secure Domain

Analyzing the instruction

MOV X0, #0x41013000

Modifying saved ELR_EL3

ERET

Outline

- Introduction
- Background
- System Overview
- Evaluation
- Conclusion

Evaluation - Transparency

- Environment:

- Analyzer:

Evaluation - Transparency

- Environment:
  - ✓ Isolated
- Analyzer:

Evaluation - Transparency

- Environment:
  - ✓ Isolated
  - ✓ Exists on OTS platforms
- Analyzer:

Evaluation - Transparency

- Environment:
  - ✓ Isolated
  - ✓ Exists on OTS platforms
- Analyzer:
  - ✓ No detectable footprints?

Evaluation - Transparency

- Environment:
  - ✓ Isolated
  - ✓ Exists on OTS platforms
- Analyzer:
  - ✓ No detectable footprints?

We believe that the hardware-based approach provides better transparency.

To build a fully transparent system, we may need additional hardware support.

Evaluation – Performance of the TS

- Testbed Specification
  - ARM Juno v1 development board
  - A dual-core 800 MHZ Cortex-A57 cluster and a quad-core 700 MHZ Cortex-A53 cluster
  - ARM Trusted Firmware (ATF) v1.1 and Android 5.1.1

Evaluation – Performance of the TS

- Calculating one million digits of π
  - GNU Multiple Precision Arithmetic Library

	Mean	STD	#Slowdown
Base: Tracing Disabled	2.133 s	0.69 ms	
Instruction Tracing	2.135 s	2.79 ms	1x
System call Tracing	2.134 s	5.13 ms	1x
Android API Tracing	149.372 s	1287.88 ms	70x

Evaluation – Performance of the TS

- Performance scores evaluated by CF-Bench

	Native Scores		Java Scores		Overall Scores	
	Mean	#Slowdown	Mean	#Slowdown	Mean	#Slowdown
Basic: Tracing Disabled	25380		18758		21407	
Instruction Tracing	25364	1x	18673	1x	21349	1x
System call Tracing	25360	1x	18664	1x	21342	1x
Android API Tracing	6452	4x	122	154x	2654	8x

Evaluation – Domain Switching Time

- Time consumption of domain switching (in µs)
  - 34x-1674x faster than MalT (11.72 µs)

ATF Enabled	Ninja Enabled	Mean	STD	95% CI
✖	✖	0.007	0.000	[0.007, 0.007]
✔	✖	0.202	0.013	[0.197, 0.207]
✔	✔	0.342	0.021	[0.334, 0.349]

Outline

- Introduction
- Background
- System Overview
- Evaluation
- Conclusion

Conclusion

- Ninja: A malware analysis framework on ARM.
  - A debug subsystem and a trace subsystem
  - Using TrustZone, PMU, and ETM to improve transparency
  - The hardware-assisted trace subsystem is immune to timing attack.

Thank you!

Email: zhenyu.ning@wayne.edu

Questions?