Nailgun: Breaking the Privilege Isolation on ARM

Fengwei Zhang

GOSSIP Summer Camp

2024-Jul-22

At the Beginning

Smart phone store lots of personal data.

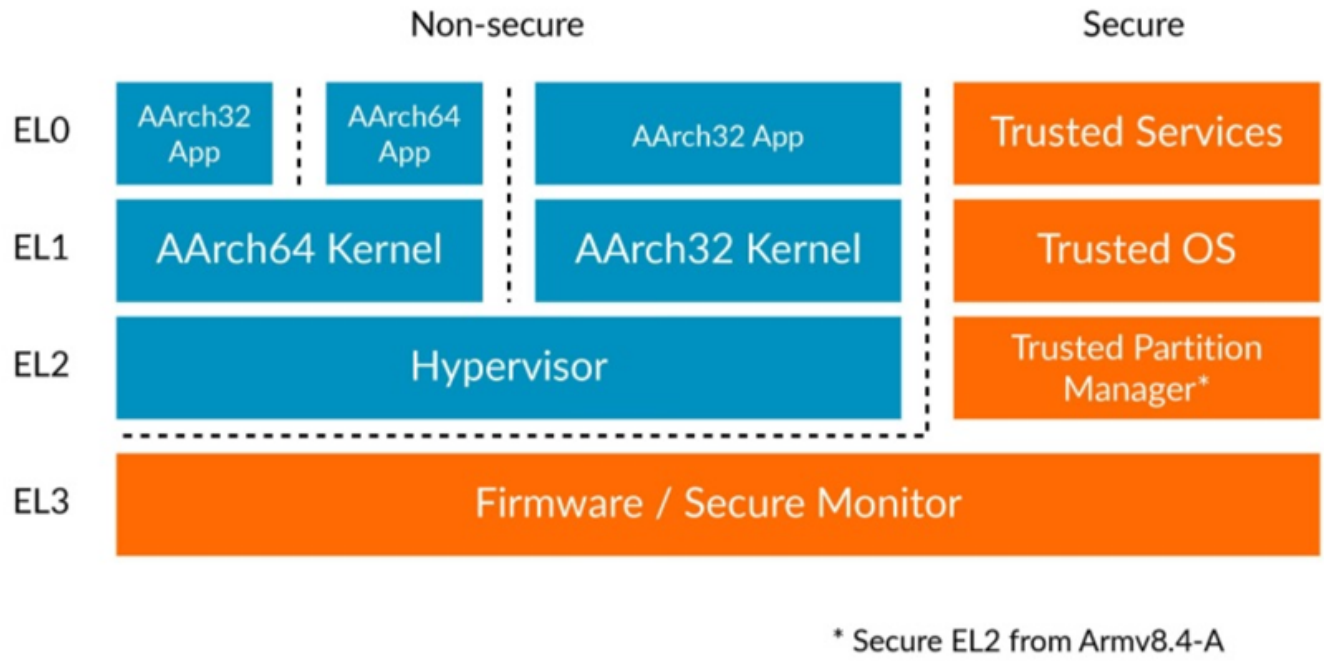
- How they protect sensitive data?
- Have you ever worry about the leakage of your fingerprint in your smartphone?

The Security Mechanism

Arm Device protect your data via privilege isolation and TrustZone.

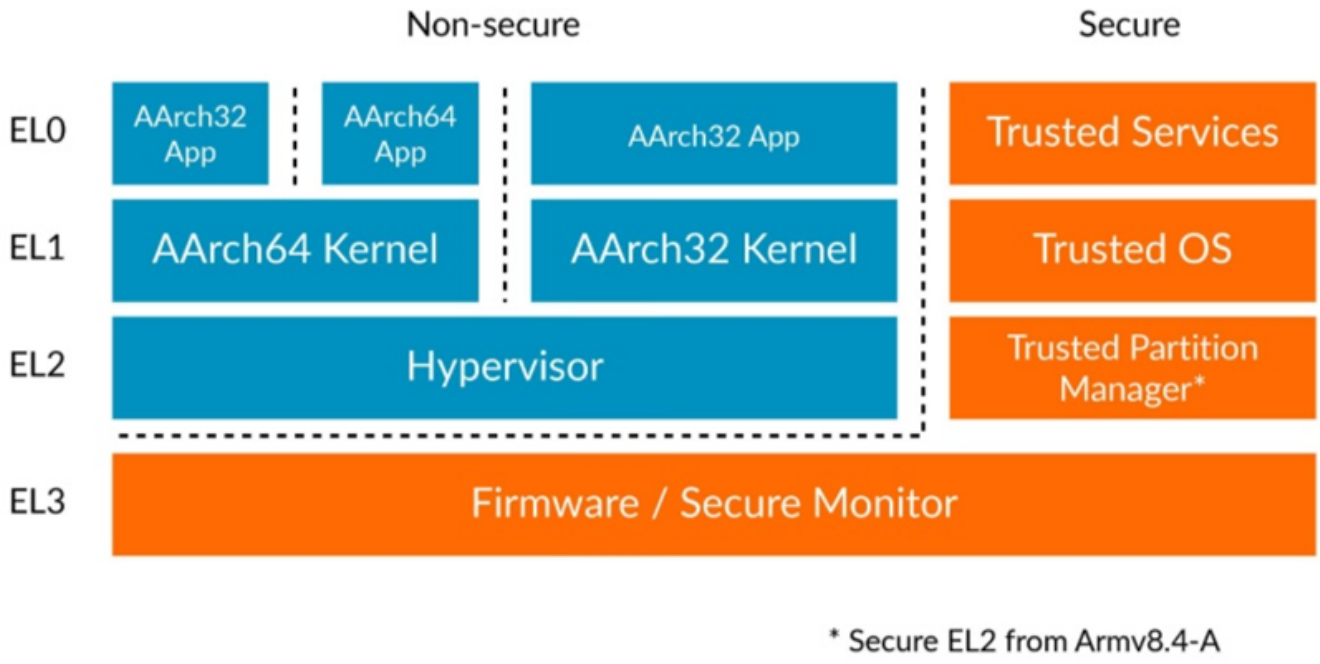
Exception Level

The exception levels is also call privilege level. The software running on each level will have different permission. For example, user space applications are executing on EL0 (Exception Level 0) and OS kernel is on EL1. EL2 is usually for Hypervisor.



Exception Level

Beside the Normal World, there is a Secure World. Each exception level in Secure World is similar to Normal World. Notice that secure EL3 acts as a gate keeper, the world switches must go through Secure EL3.



Nailgun Attack

Naigun Attack is to break the privilege isolation.

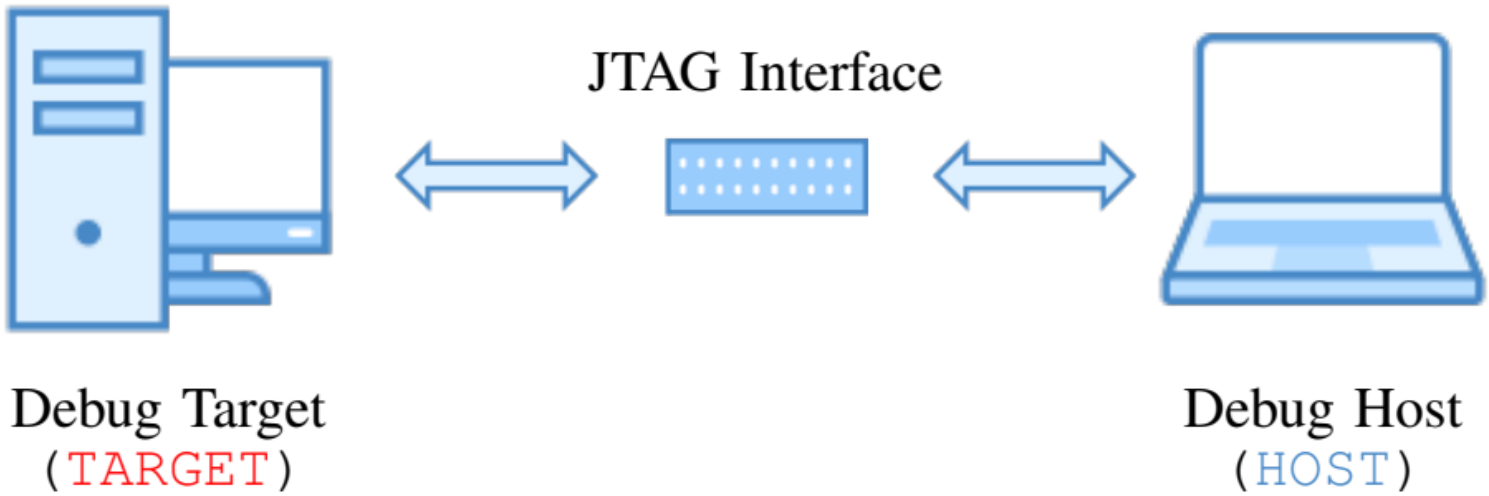
Introduction

Modern processors are equipped with hardware-based debugging features to facilitate on-chip debugging process.

- E.g., hardware breakpoints and hardware-based trace.
- It normally requires cable connection (e.g., JTAG [1]) to make use of these features.
- The debug host access the debug component on a processor, a.k.a. the debug target.

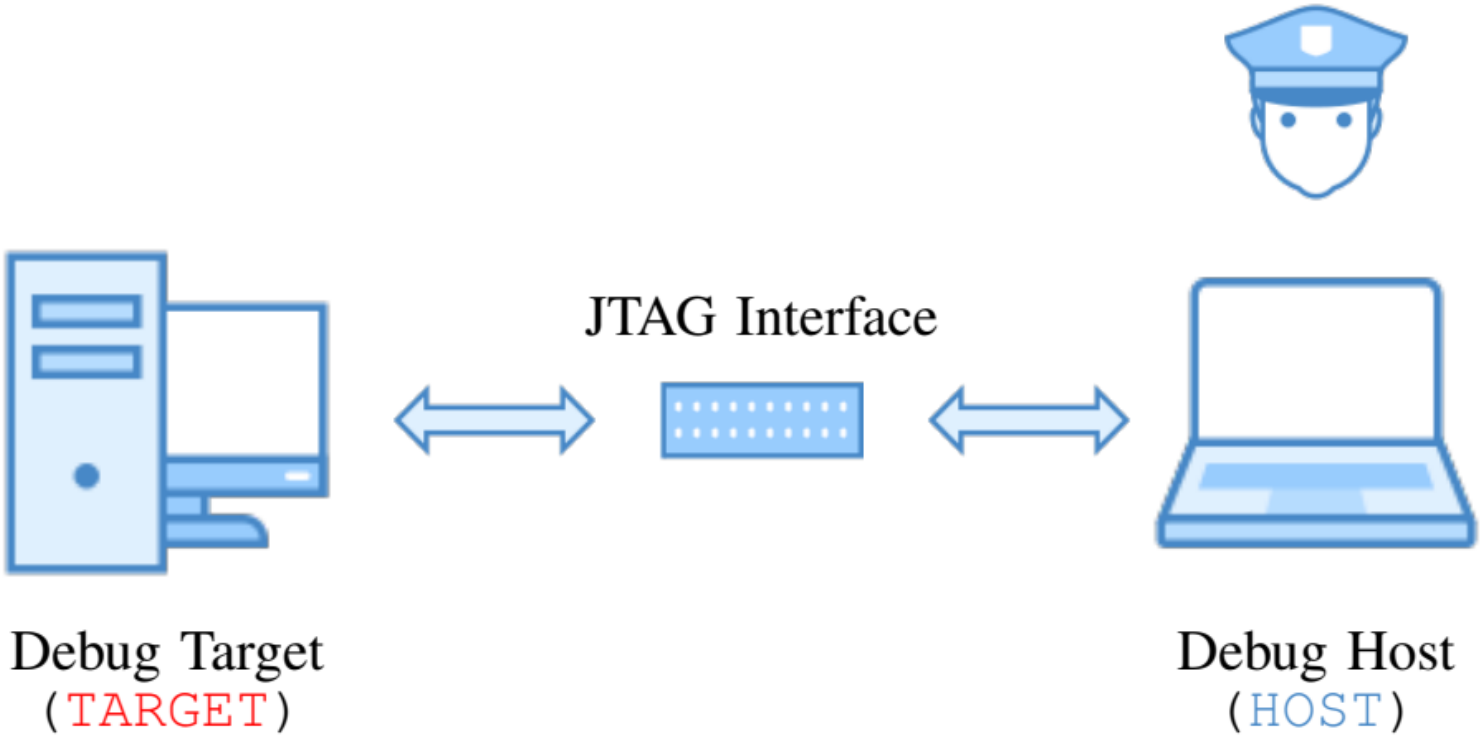
Traditional External Debugging

In traditional debugging, a cable is needed to connect the debug target and the host. There are physical connection to the debug target.



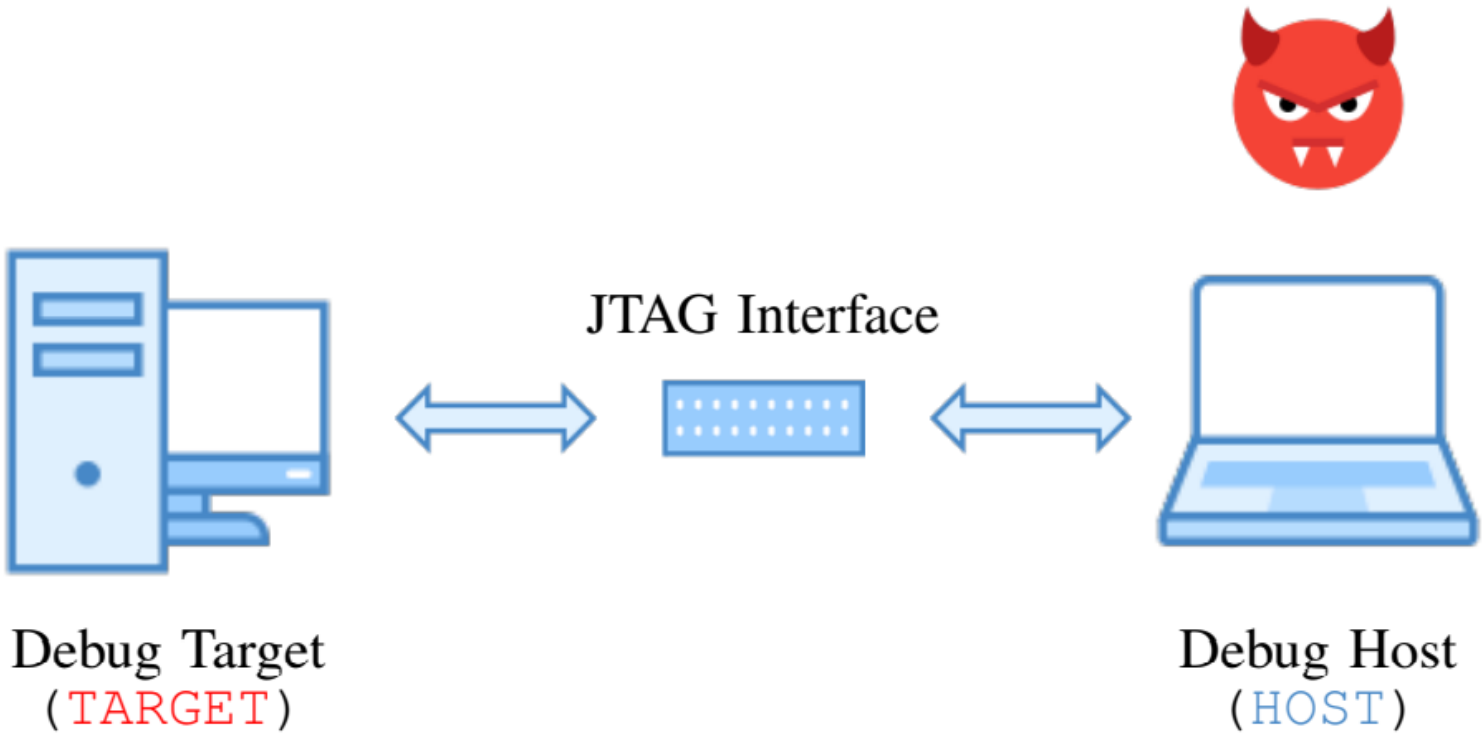
Traditional External Debugging

It is not that easy to access the target physically. In the most of the time, we can trust the debug host.



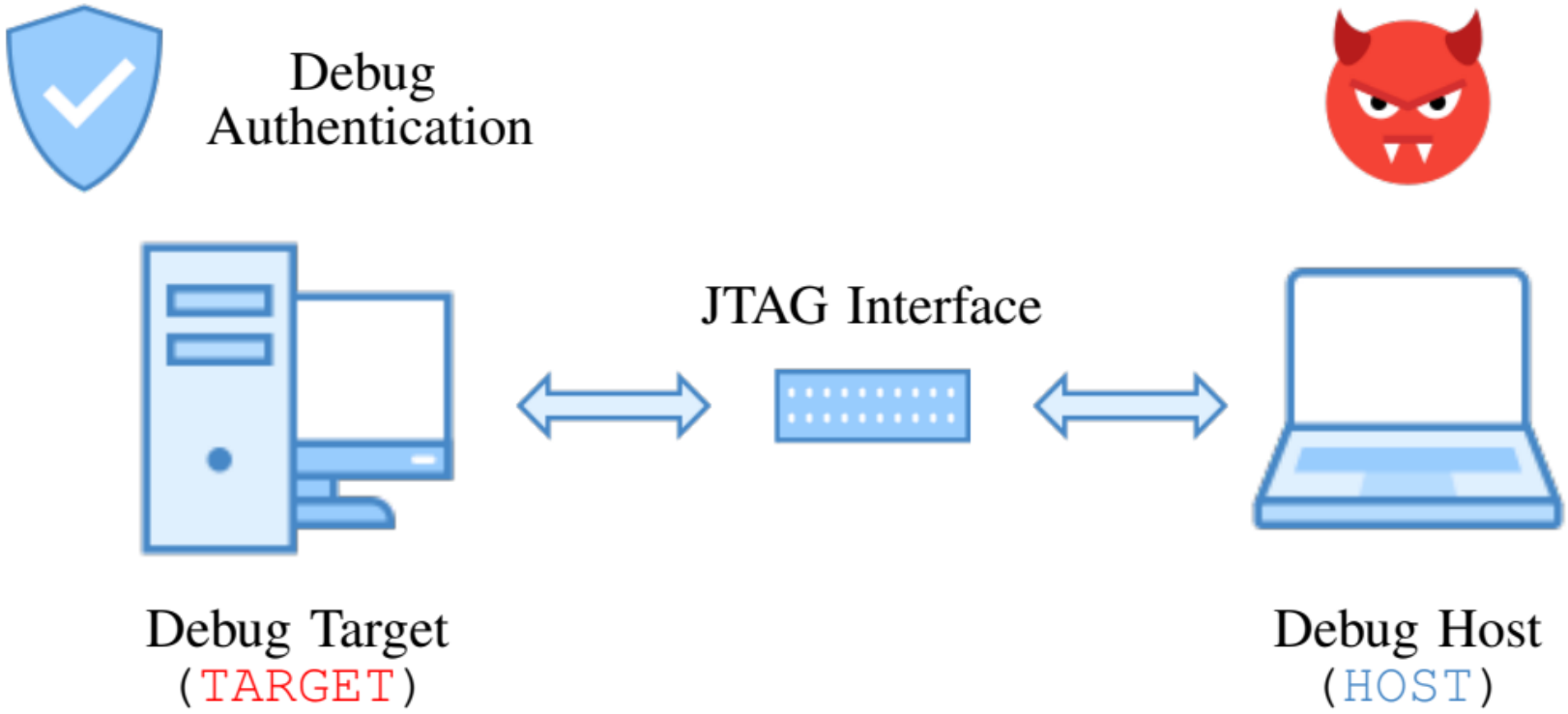
Traditional External Debugging

What if the debug host is not trusted?



Traditional External Debugging

The debug authentication can still limit the debug host.



Debug Authentication

There are four types of debug

- Secure Invasive Debug
- Secure Non-Invasive Debug
- Non-Secure Invasive Debug
- Non-Secure Non-Invasive Debug

Invasive means the host can observe and control the target. There are debug authentication signals configured by OEMs indicating whether these four are implemented and enable or not. The debug host can not change the signal.

Obstacles for misusing the traditional debugging

Obstacle for attackers:

- Obstacle 1: Physical access.
- Obstacle 2: Debug authentication mechanism.

Does both Obstacles work as expected?

Memory mapping

Memory mapping is the translation between the logical address space and the physical memory.
Can we mapped the physical address of debug component to the memory, and then access it in another process?

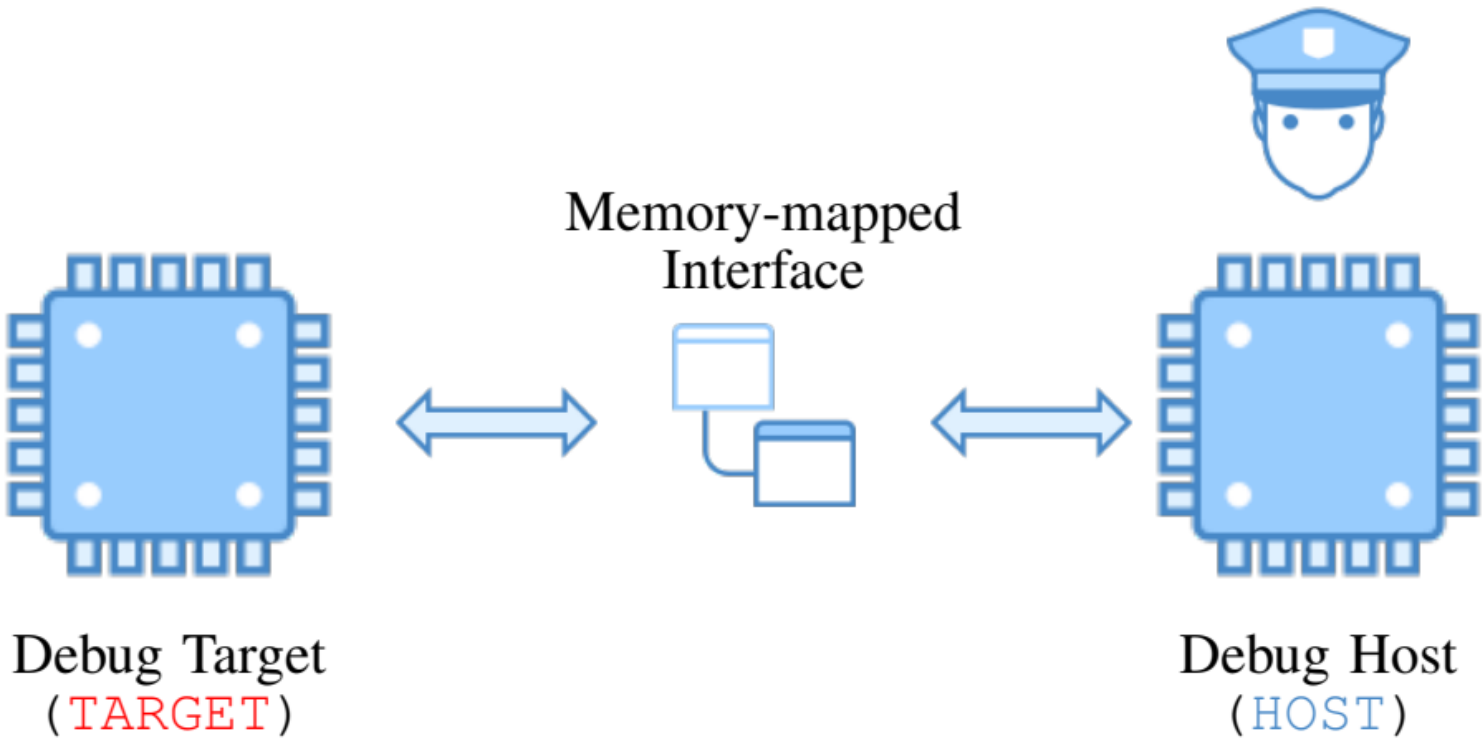
Inter-processor Debugging

We can use one processor on the chip to debug another one on the same chip, and we refer it as inter-processor debugging.

- Memory-mapped debugging registers. (Introduced since Armv7)
- No JTAG, No physical access.

Inter-processor Debugging

Some CPUs have multiple cores, and each core is a processor by itself. We can launch a debug event from one core to another core.



Debug authentication signal

Category	Platform / Device	Debug Authentication Signals			
		DBGEN	NIDEN	SPIDEN	SPNIDEN
Development Boards	ARM Juno r1 Board	✔	✔	✔	✔
	NXP i.MX53 QSB	✖	✔	✖	✖
IoT Devices	Raspberry PI 3 B+	✔	✔	✔	✔
Cloud Platforms	64-bit ARM miniNode	✔	✔	✔	✔
	Packet Type 2A Server	✔	✔	✔	✔
	Scaleway ARM C1 Server	✔	✔	✔	✔
Mobile Devices	Google Nexus 6	✖	✔	✖	✖
	Samsung Galaxy Note 2	✔	✔	✖	✖
	Huawei Mate 7	✔	✔	✔	✔
	Motorola E4 Plus	✔	✔	✔	✔
	Xiaomi Redmi 6	✔	✔	✔	✔

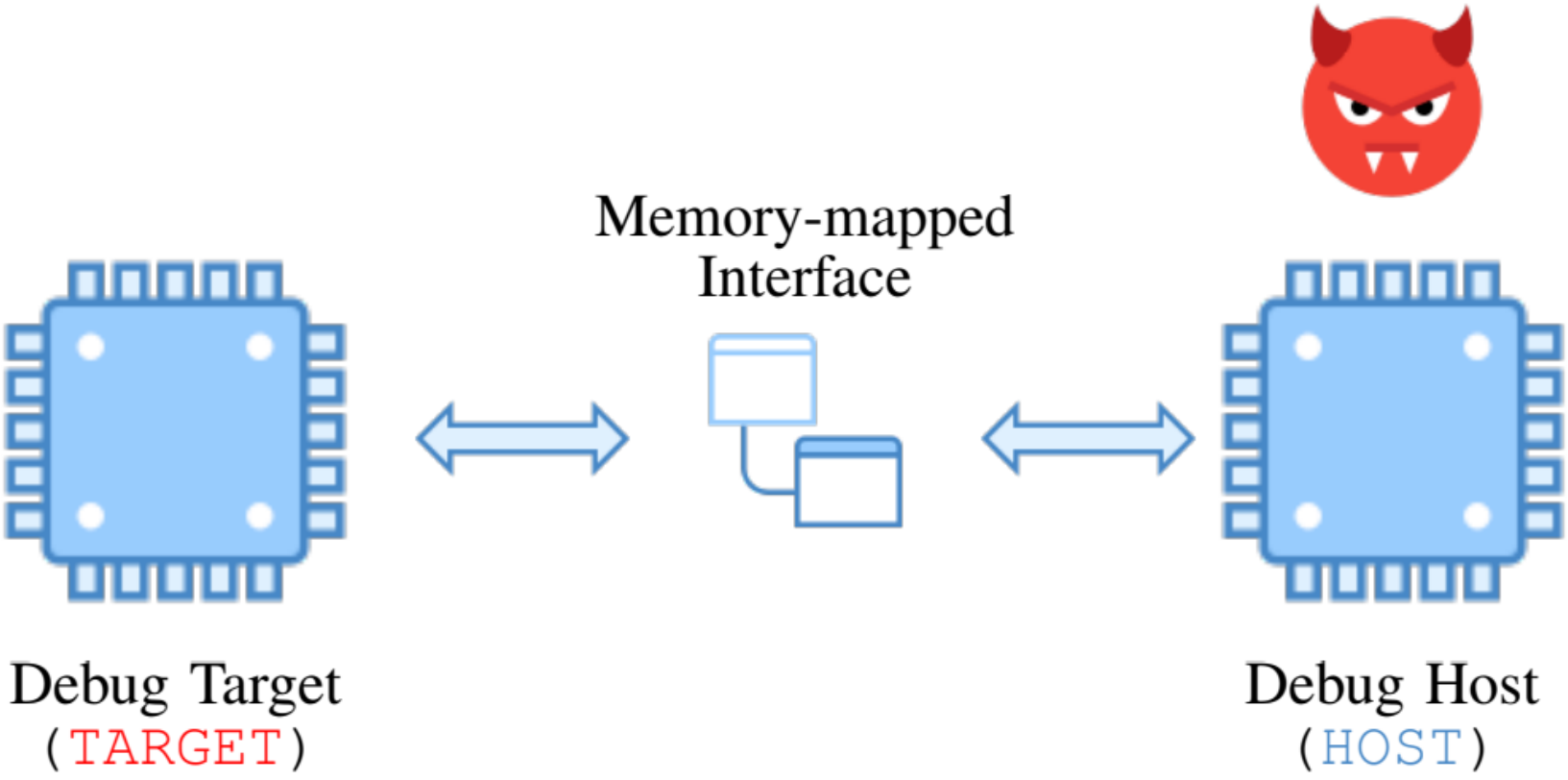
Obstacles for misusing the traditional debugging

Obstacle for attackers:

- ~~Obstacle 1: Physical access.~~
  We don't need physical access to debug a processor.
- ~~Obstacle 2: Debug authentication mechanism.~~
  The debug authentication mechanism allows us to debug the processor.

Does both Obstacles work as expected?

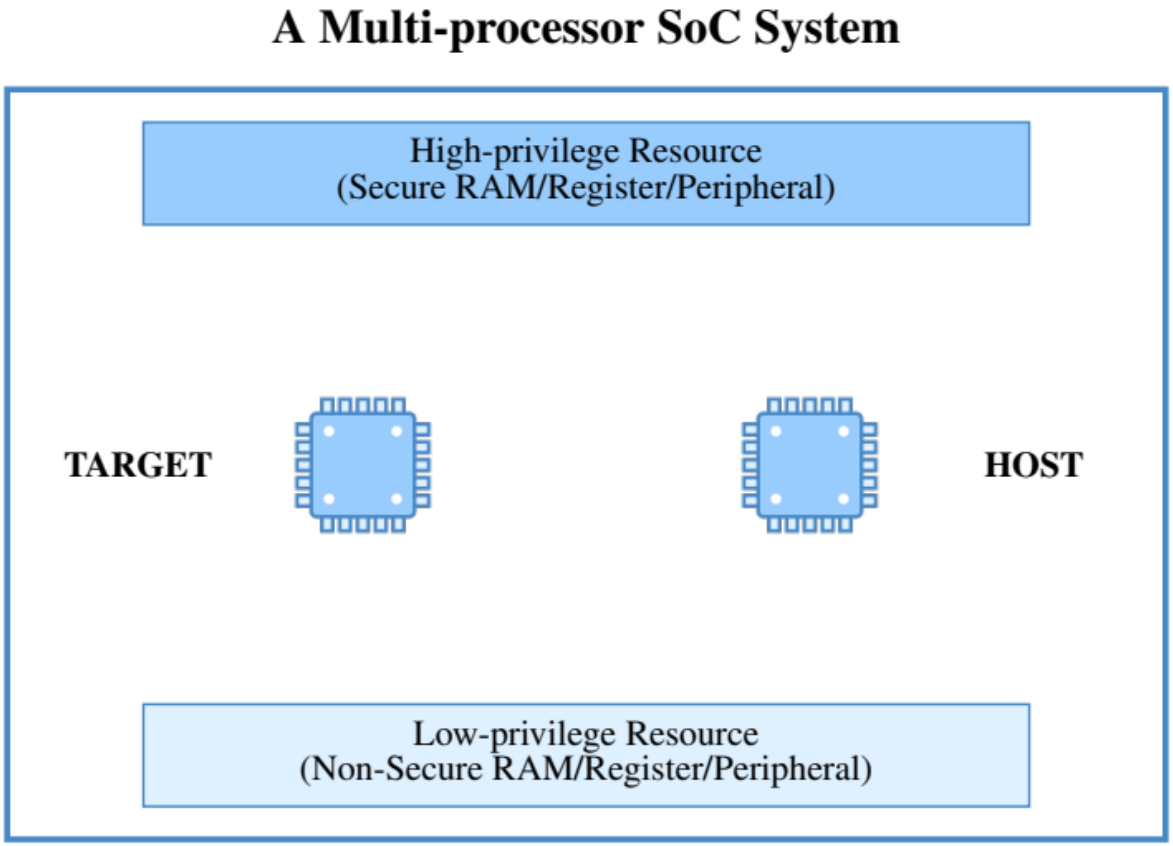
Nailgun Attack



Nailgun Attack

An example SoC system:

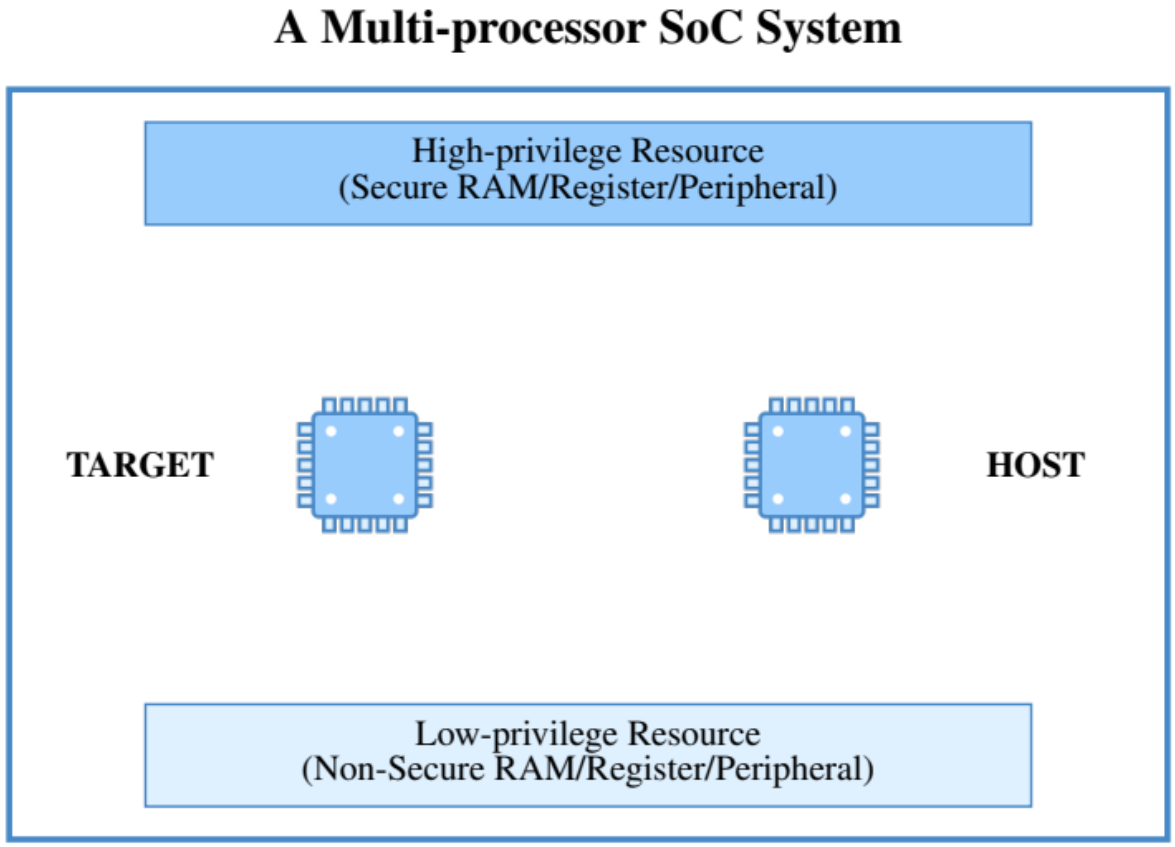
- Two processors as HOST and TARGET, respectively.
- Low-privilege and High-privilege resource.



Nailgun Attack

An example SoC system:

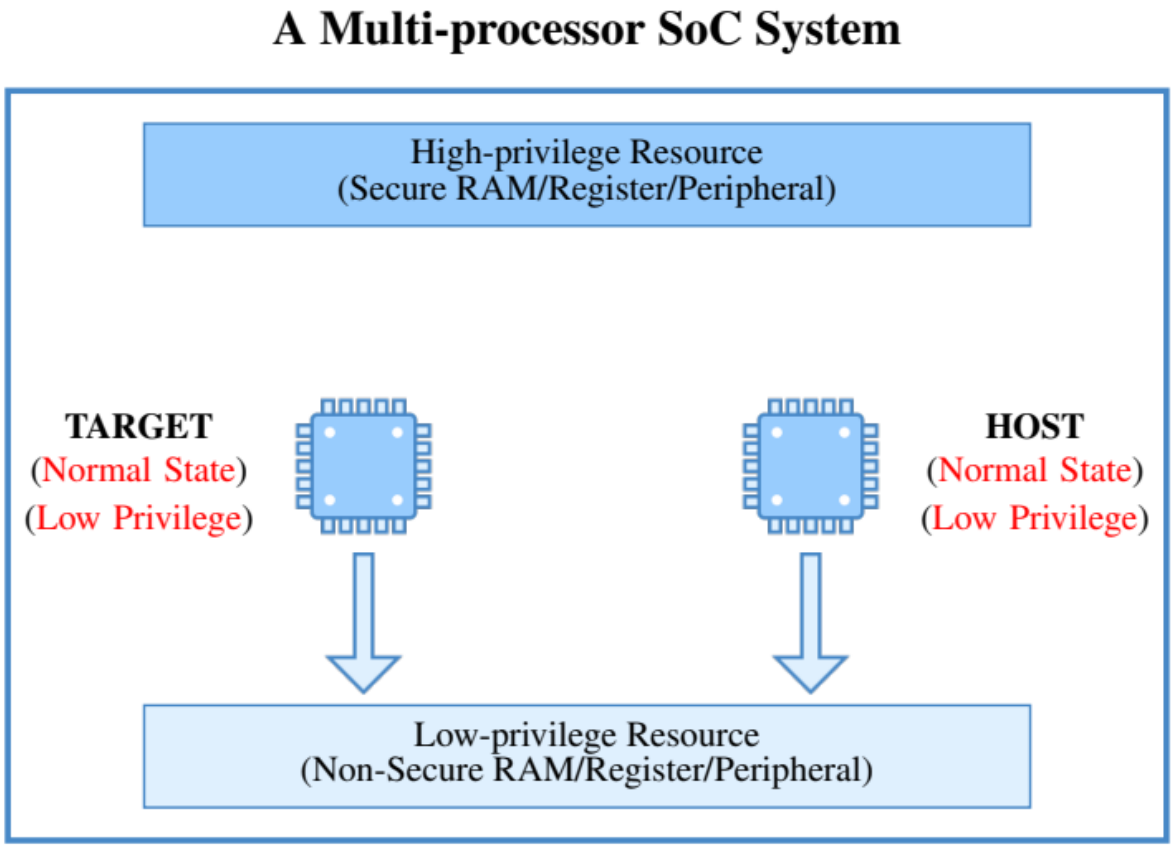
- Low-privilege refers to non-secure kernel-level privilege.
- High-privilege refers to any other higher privilege.



Nailgun Attack

Both processors are only access low-privilege resource.

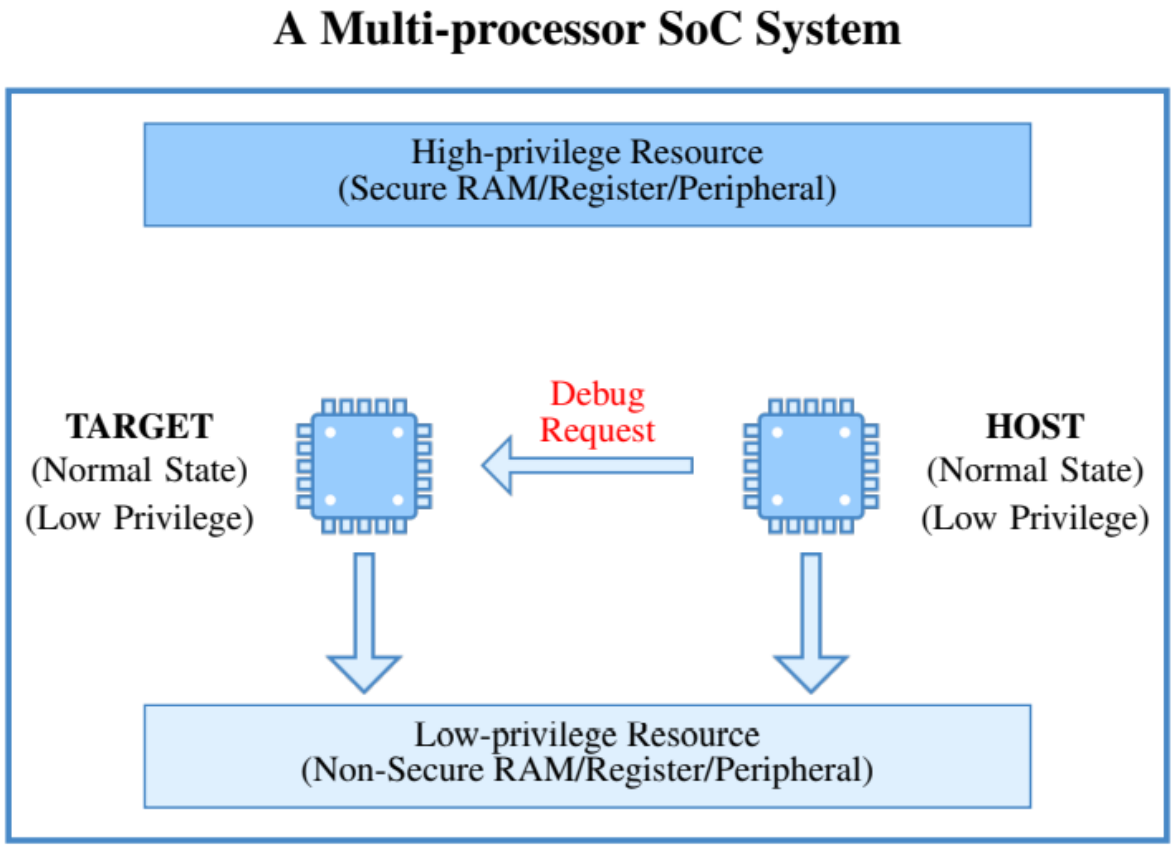
- Normal state
- Low-privilege mode



Nailgun Attack

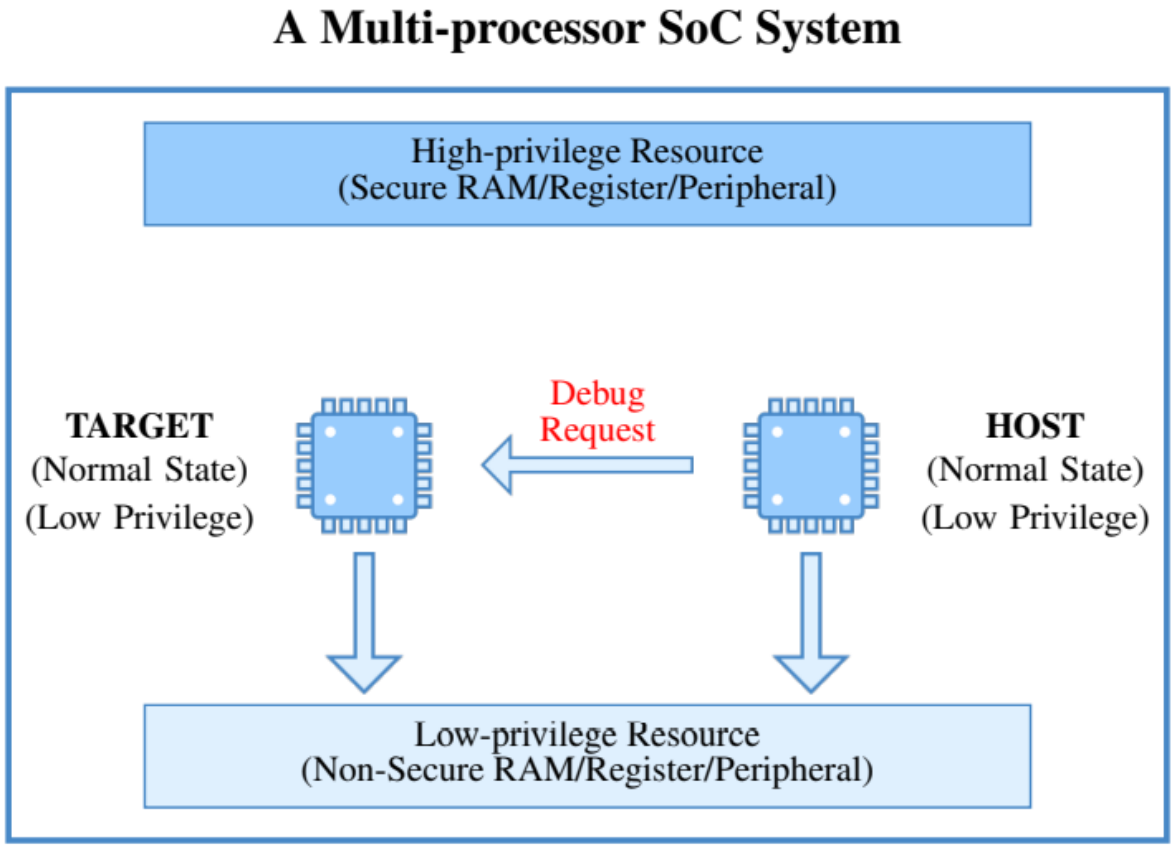
HOST sends a Debug Request to TARGET,

- TARGET checks its authentication signal.
- Privilege of HOST is ignored.



Nailgun Attack

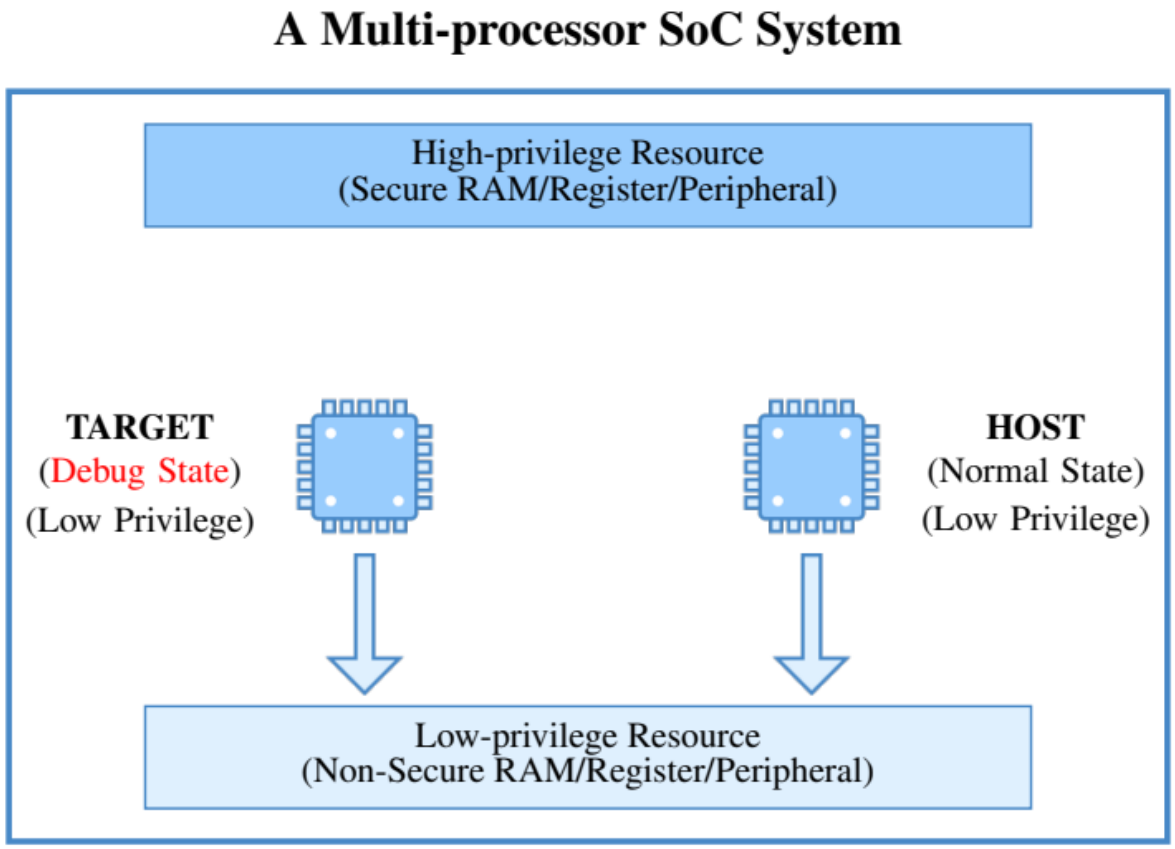
Implication: A low-privilege processor can make an arbitrary processor (even a high-privilege processor) enter the debug state.



Nailgun Attack

TARGET turns to Debug State according to the request.

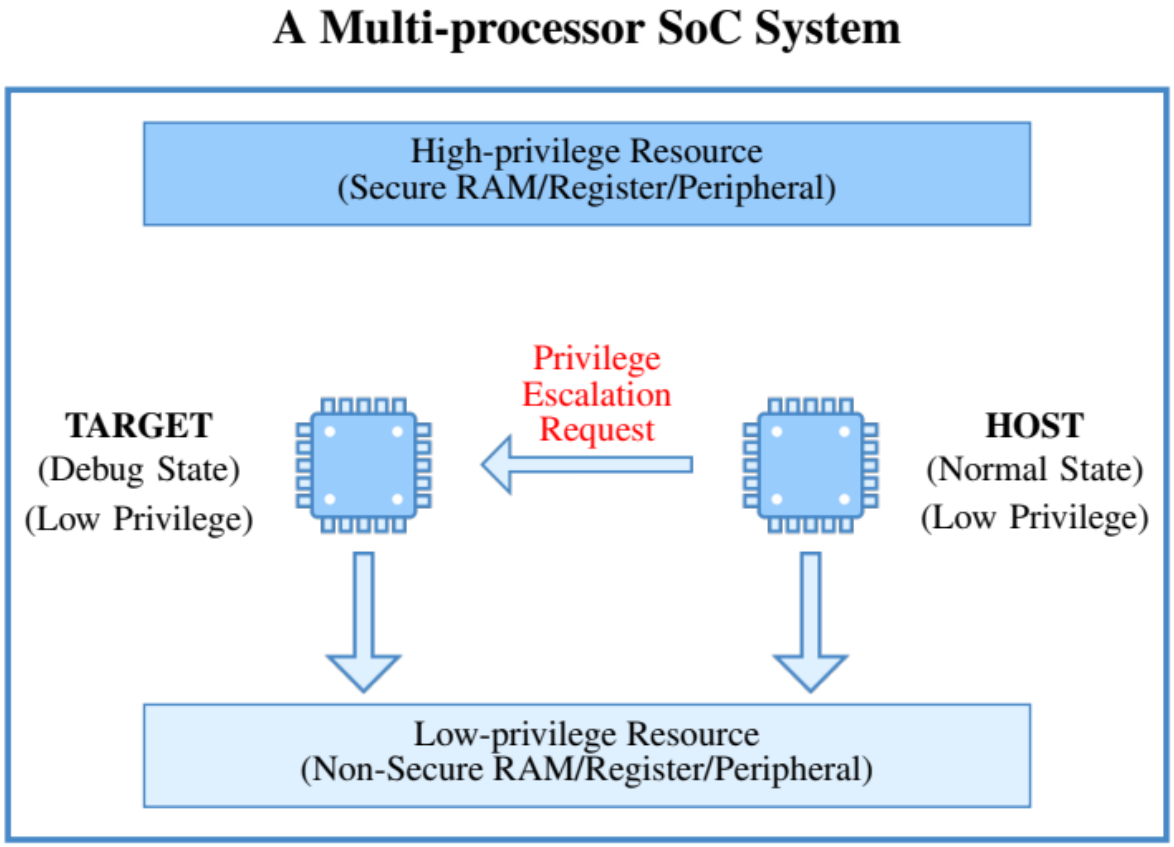
- Low-privilege mode.
- No access to high-privilege resource.



Nailgun Attack

HOST sends a Privilege Escalation Request to TARGET,

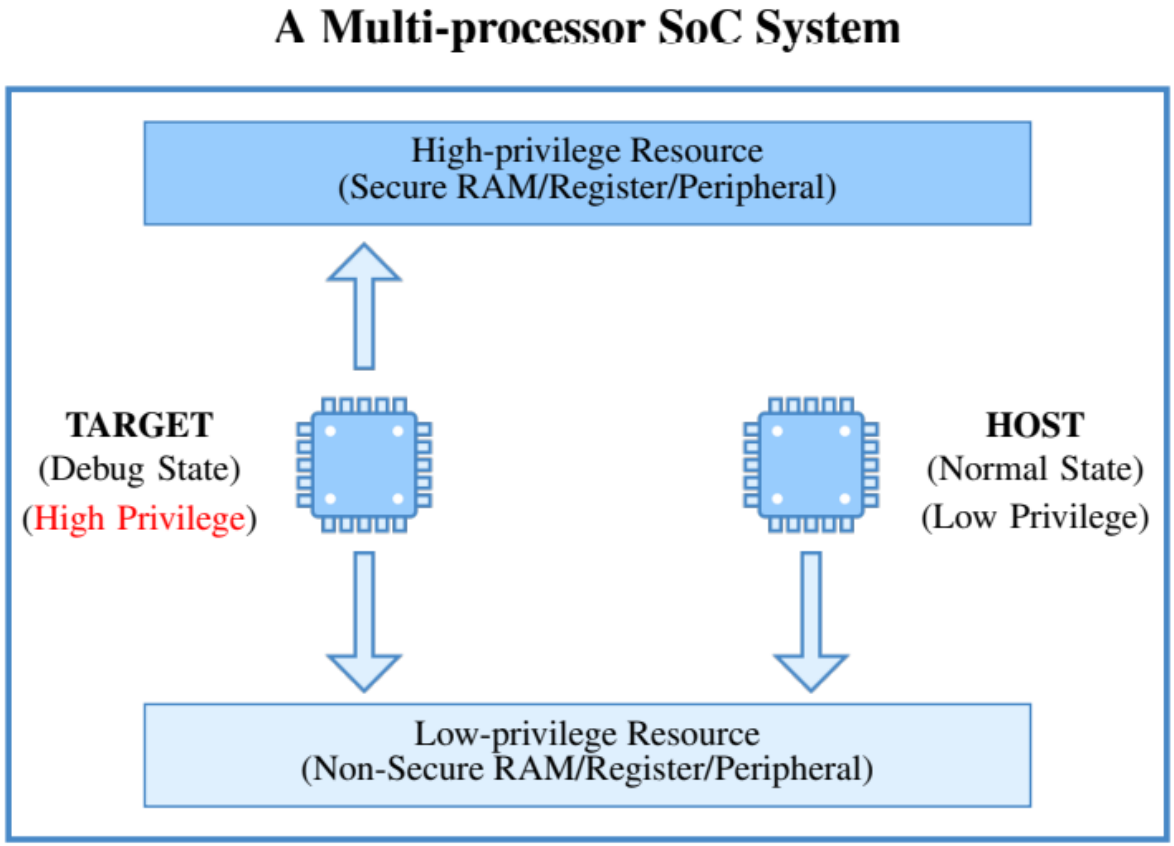
- E.g., executing DCPS series instructions.
- The instructions can be executed at any privilege level.



Nailgun Attack

TARGET turns to High-privilege Mode according to the request.

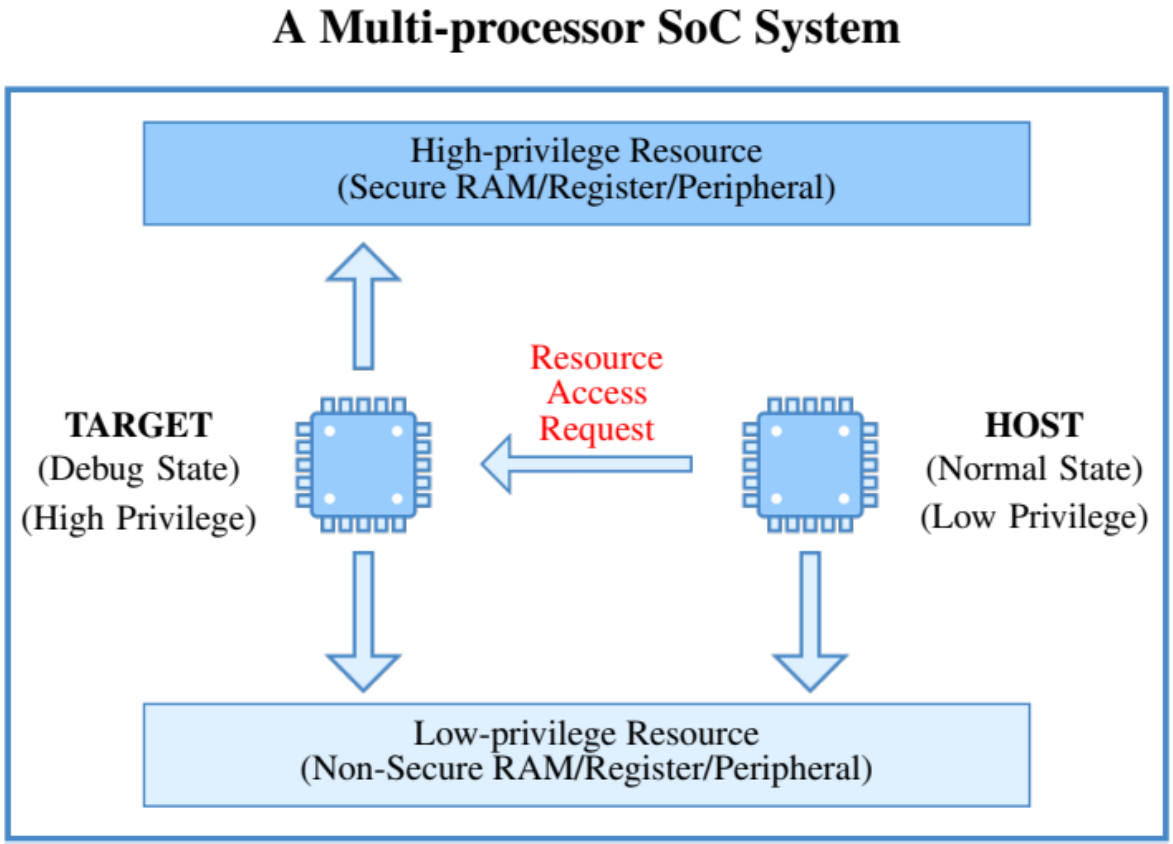
- Debug state, high-privilege mode
- Gained access to high-privilege resource



Nailgun Attack

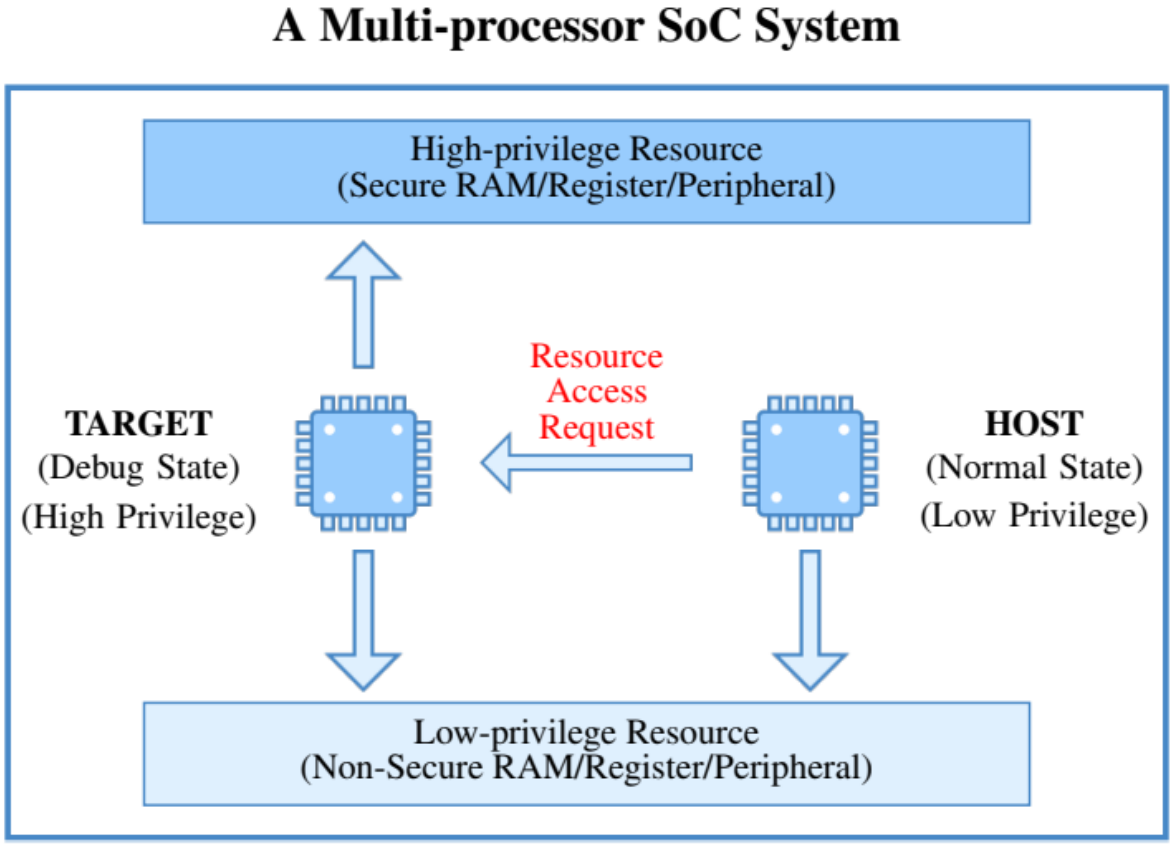
HOST sends a Resource Access Request to TARGET,

- E.g., accessing secure RAM/register/peripheral.
- Privilege of HOST is ignored.



Nailgun Attack

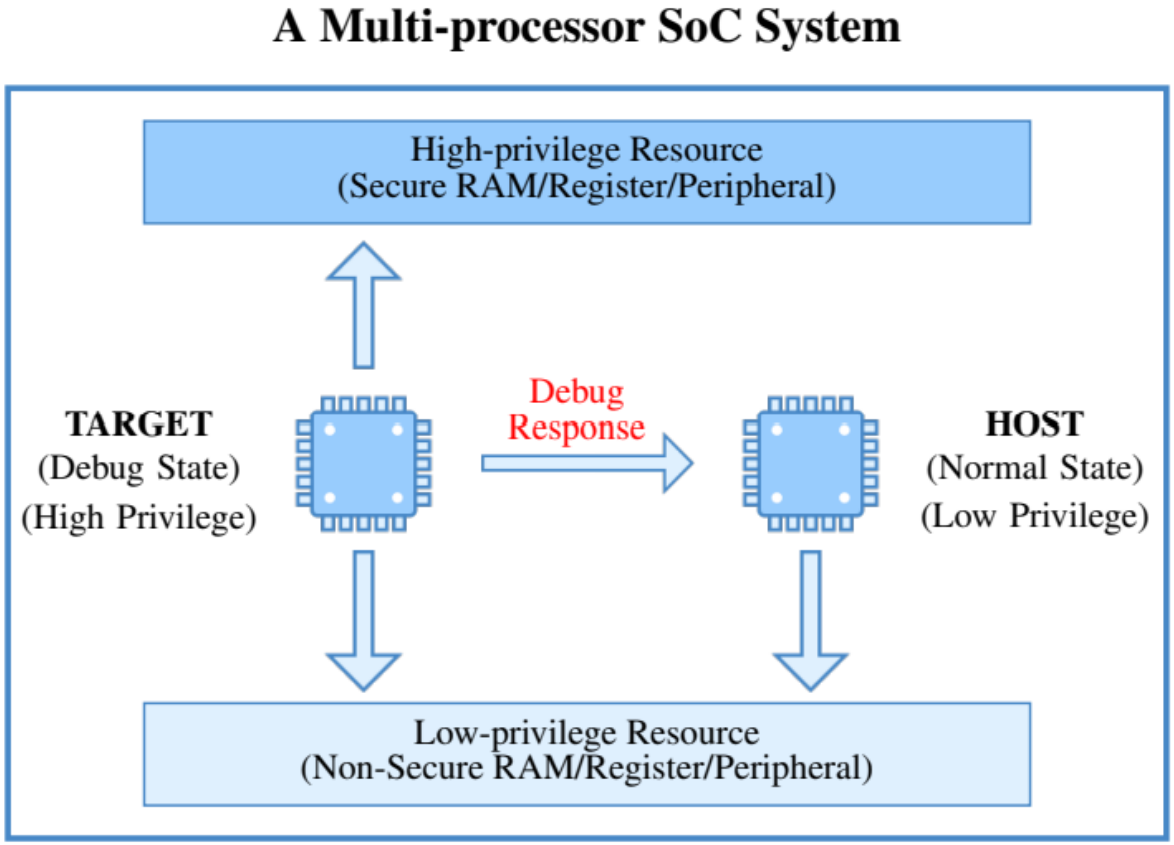
Implication: The instruction execution and resource access in TARGET does not take the privilege of HOST into account.



Nailgun Attack

TARGET return the result to HOST,

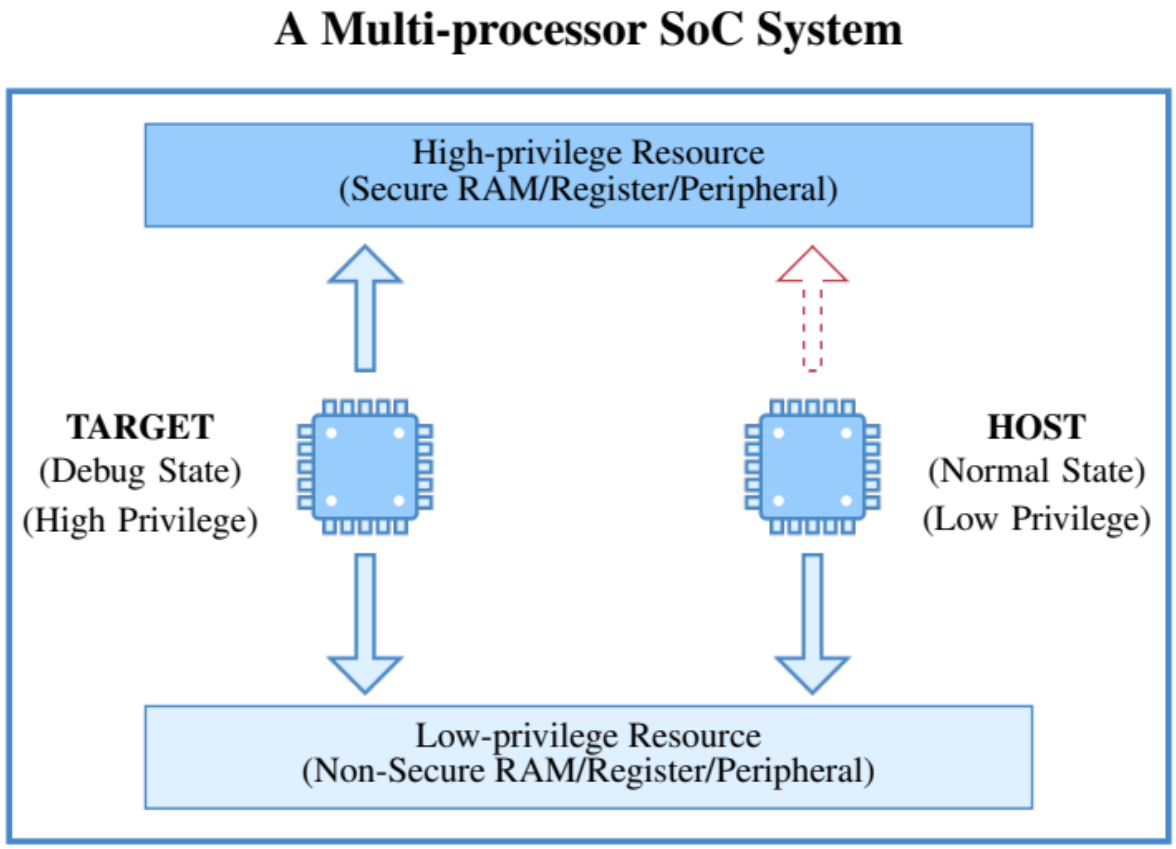
- i.e., content of the high-privilege resource.
- I Privilege of HOST is ignored.



Nailgun Attack

HOST gains access to the high-privilege resource while running in,

- Normal state
- Low-privilege mode



1. ARM.ARMv8-A reference manual.
http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0487a.k/index.html.
2. Ning Zhenyu.Nailgun: Breaking the Privilege Isolation on ARM.
https://fengweiz.github.io/19fa-cs315/slides/nailgun-zhenyu-ning.pdf.
3. Zhenyu Ning and Fengwei Zhang.Understanding the security of arm debugging features.

Thanks!