Return-to-Non-Secure Vulnerabilities on ARM Cortex-M TrustZone: Attack and Defense

Zheyuan Ma, Xi Tan, Lukasz Ziarek, Ning Zhang, Hongxin Hu, Ziming Zhao

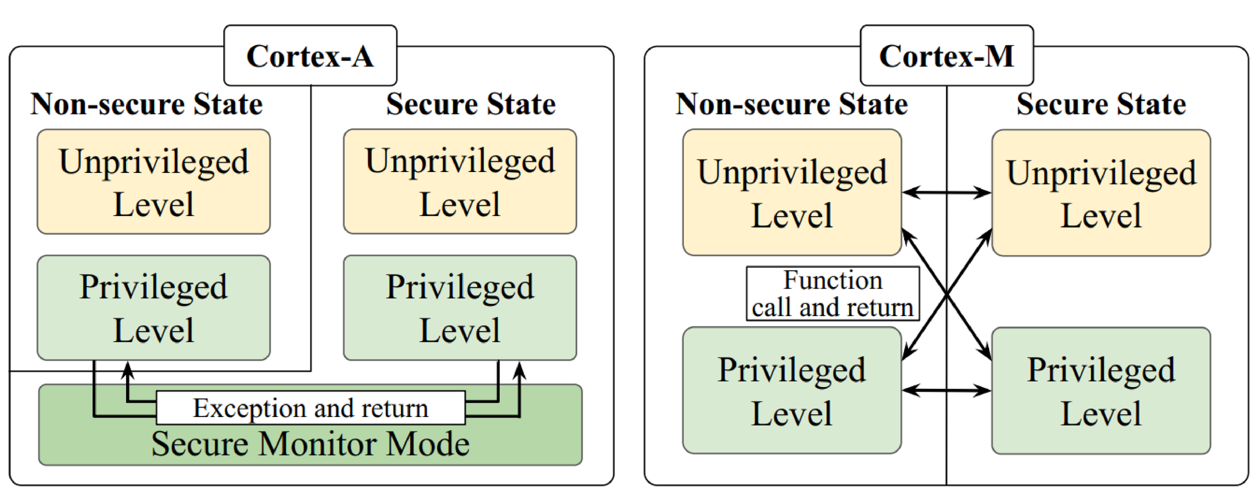
{zheyuanm, xitan, lziarek, hongxinh, zimingzh}@buffalo.edu, zhang.ning@wustl.edu

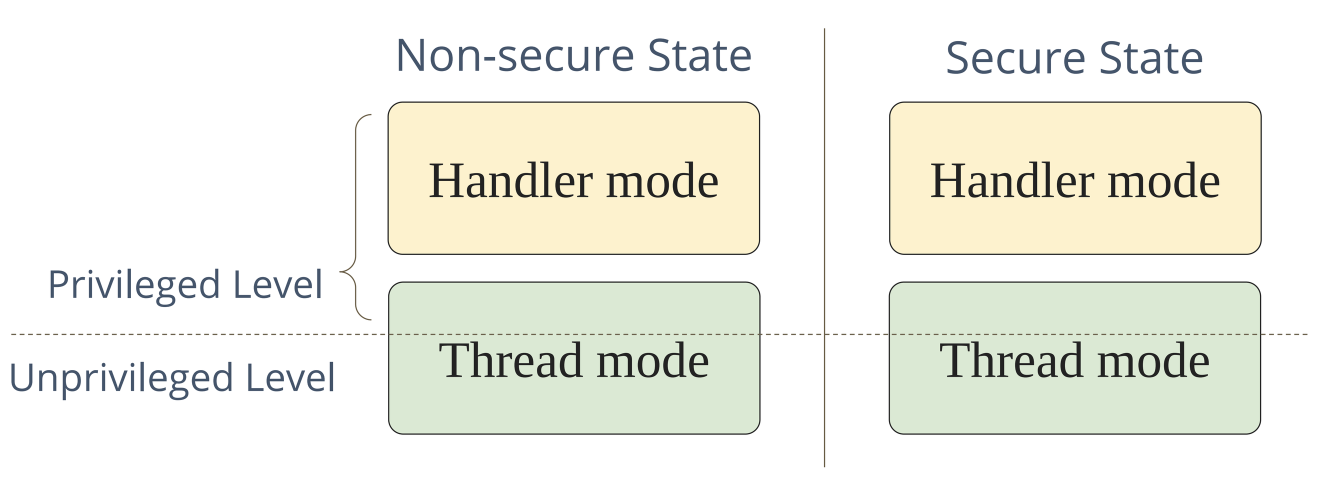
① Introduction

Exploring the security landscape of ARM Cortex-M TrustZone, we uncover a new class of vulnerabilities, termed as 'return-to-non-secure' (ret2ns) attacks. These attacks exploit the fast state switch mechanism of the TrustZone, leading to arbitrary code execution with escalated privilege in the non-secure state. We not only confirm the feasibility of ret2ns attacks but also propose effective countermeasures, introducing two address sanitizing mechanisms with a minimal performance impact.

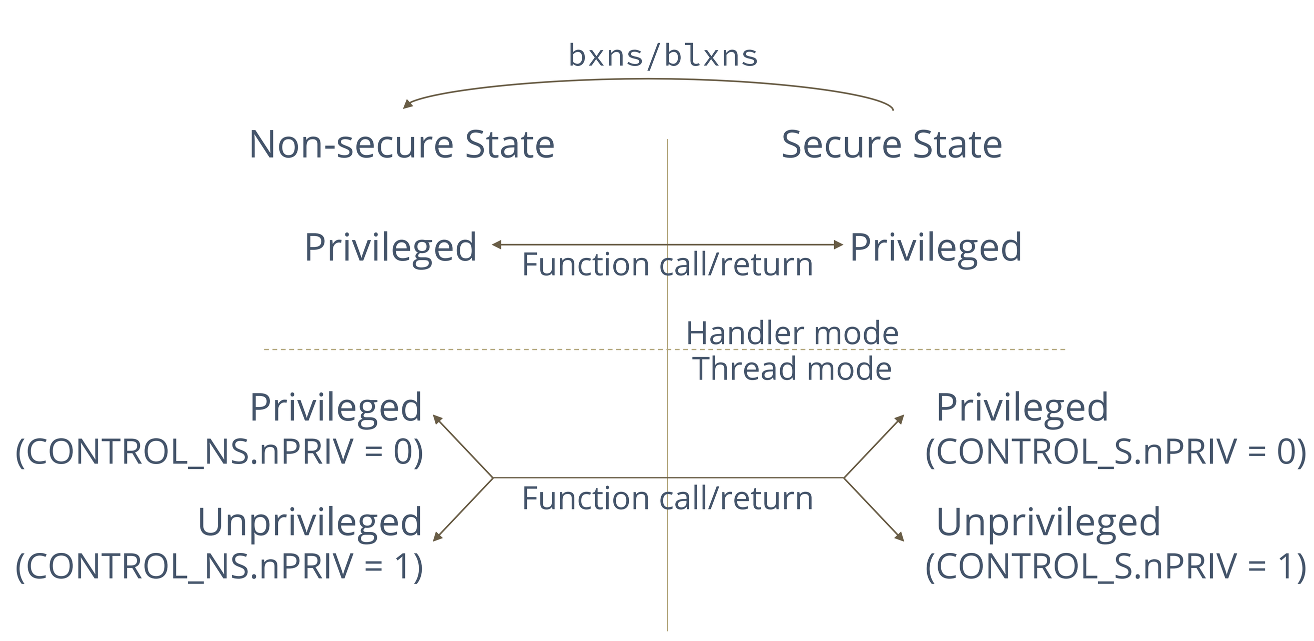
② Background

- Cortex-M's rapid state switch has security implications
- The semantic gap results in potential confused-deputy attacks





- `IPSR != 0`, Handler mode
- `IPSR = 0`, Thread mode
- `IPSR` register is shared between states; `CONTROL` register is banked for each state



③ Threat Model

Goal: a user-space attacker in NS conducts privilege escalation

Assumptions:

- memory corruption vulnerability in S
- attacker utilizes NS system calls (SVC) for S interaction
- arbitrary code execution in S is not possible for an attacker

Target: to corrupt code pointer used by `bxns/blxns` in S

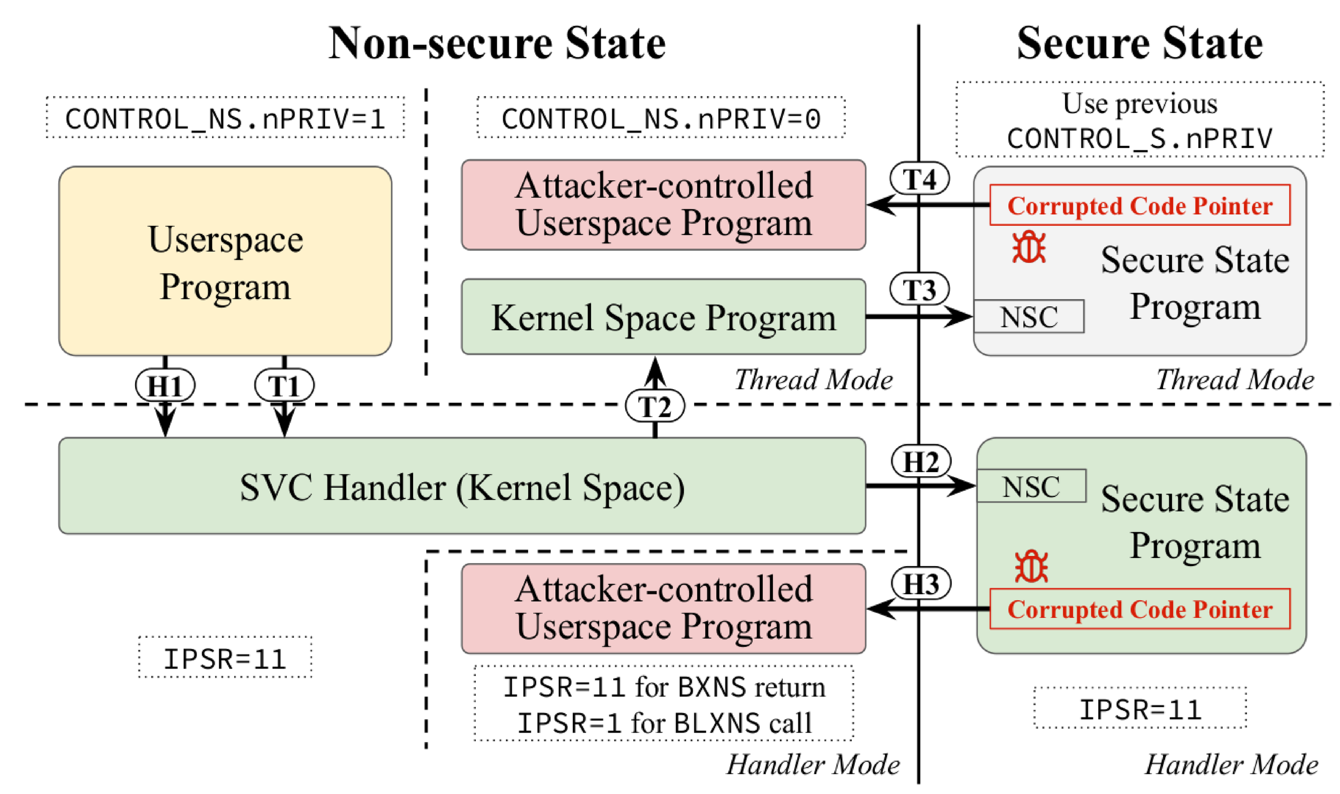
④ Overview

Handler-mode-originated attacks

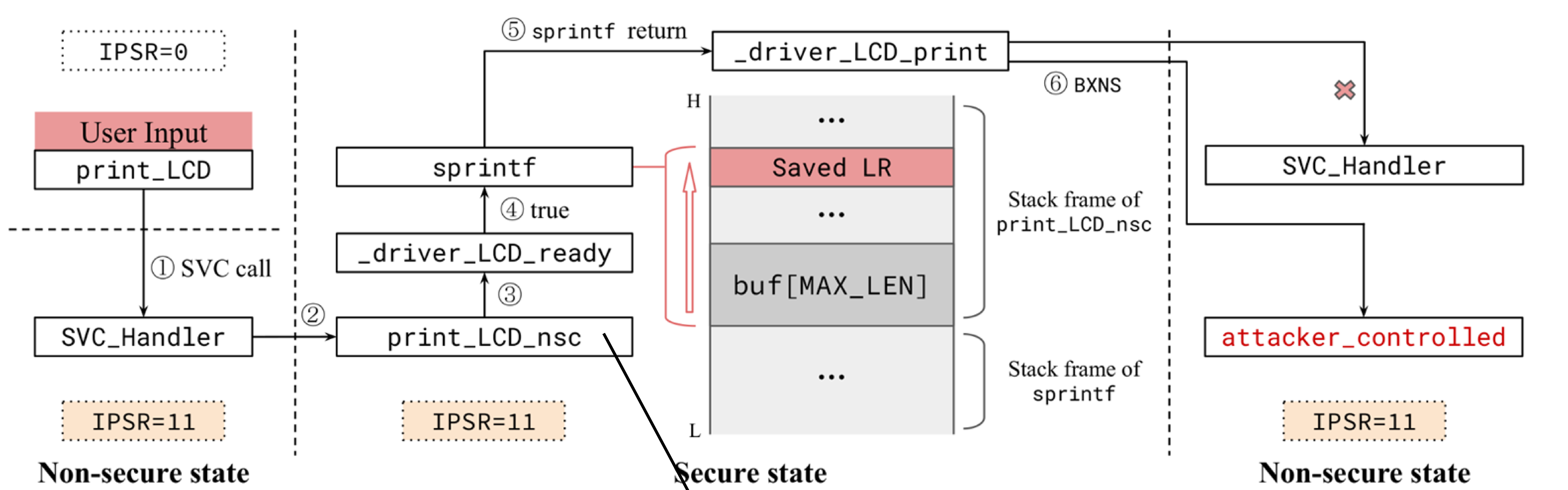
- `IPSR` is shared between states

Thread-mode-originated attacks

- `CONTROL.nPRIV` is banked between states



⑤ A Walking Example




```
#define MAX_LEN 128
int32_t _driver_LCD_ready();
int32_t _driver_LCD_print(char *);

/* Non-secure callable function */
int32_t print_LCD_nsc(char *msg)
__attribute__((cmse_nonsecure_entry));

int32_t print_LCD_nsc(char *msg)
{
    char buf[MAX_LEN] = {0};

    if (_driver_LCD_ready())
    {
        sprintf(buf, "%s %s: %s", _TIME_STAMP,
_SYSTEM_STATUS, msg); /* Buffer overflow */

        return _driver_LCD_print(buf); /* bxns return */
    }
    else
        return -1; /* bxns return */
}
```

⑥ Defense 1 - MPU-assisted Address Sanitizer

- Validate memory access permissions for NS target
- Verify NS destination address against NS MPU configuration before `bxns/blxns` executes

Defense 2 - Address Masking

- Assume user/kernel space programs in distinct, known memory regions
- Apply bit-wise masking to NS target address

⑦ Defense Evaluation

T, N	Blinky	MPU-assisted Addr Sanitizer	Address Masking
$10^7, 10^7$	1,200,503,441	1,200,508,359 (0.0004%)	1,200,506,190 (0.0002%)
$10^5, 10^5$	12,503,869	12,508,385 (0.0361%)	12,506,793 (0.0234%)
$10^7, 10^5$	12,473,897	12,474,465 (0.0046%)	12,474,243 (0.0028%)
$10^5, 10^7$	1,203,433,289	1,203,892,073 (0.0381%)	1,203,674,733 (0.0201%)

⑧ Summary

- The semantic gap in Cortex-M TrustZone results in potential confused-deputy attacks
- Four types of ret2ns attacks
- Two address sanitizing mechanisms
- Negligible defense runtime overhead