CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Pitfalls for Security Isolation in Multi-CPU Systems

Simeon Hoffmann, Nils Ole Tippenhauer

Simeon Hoffmann | NDSS Symposium | February 25, 2025

The Internet of Things Is Everywhere

The Internet of Things Is Everywhere

The Internet of Things Is Everywhere

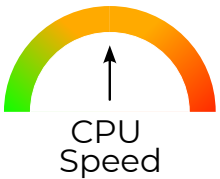
The Internet of Things Is Everywhere

The Internet of Things Is Everywhere

The Internet of Things Is Everywhere

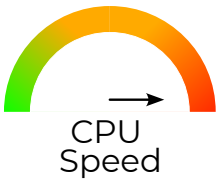
The Internet of Things Is Everywhere

- … and it demands more computational power!



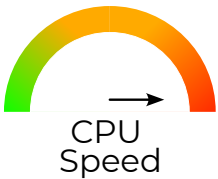
The Internet of Things Is Everywhere

- … and it demands more computational power!
- Past approach: higher CPU frequency



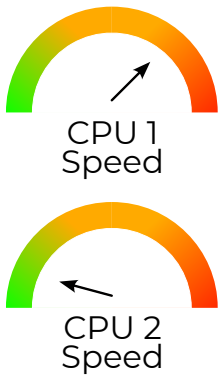
The Internet of Things Is Everywhere

- … and it demands more computational power!
- Past approach: higher CPU frequency
- Drawback: high power consumption



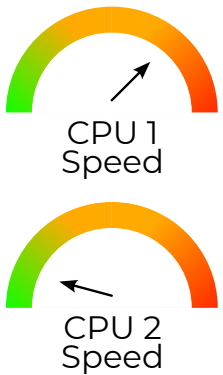
The Internet of Things Is Everywhere

- … and it demands more computational power!
- Past approach: higher CPU frequency
- Drawback: high power consumption
- Recent approach: multiple CPUs



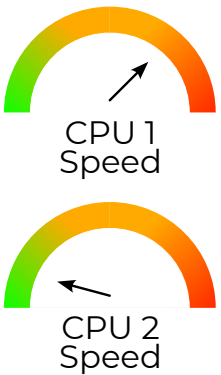
The Internet of Things Is Everywhere

- … and it demands more computational power!
- Past approach: higher CPU frequency
- Drawback: high power consumption
- Recent approach: multiple CPUs
- Different CPU configurations improve flexibility



The Internet of Things Is Everywhere

- … and it demands more computational power!
- Past approach: higher CPU frequency
- Drawback: high power consumption
- Recent approach: multiple CPUs
- Different CPU configurations improve flexibility
- What are the security implications of multiple CPUs?



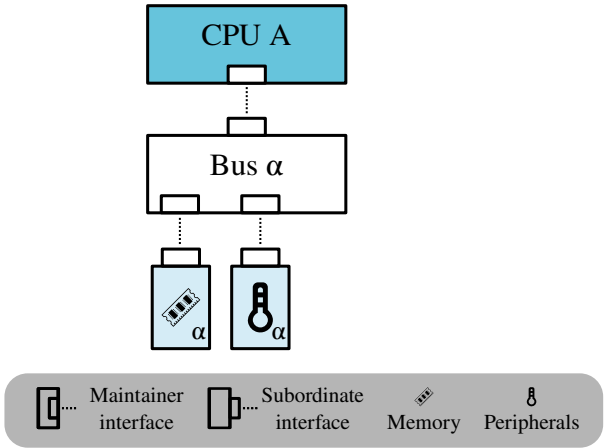
Goals

G1: Systematical assessment of security issues introduced by multi-CPU architectures in embedded devices

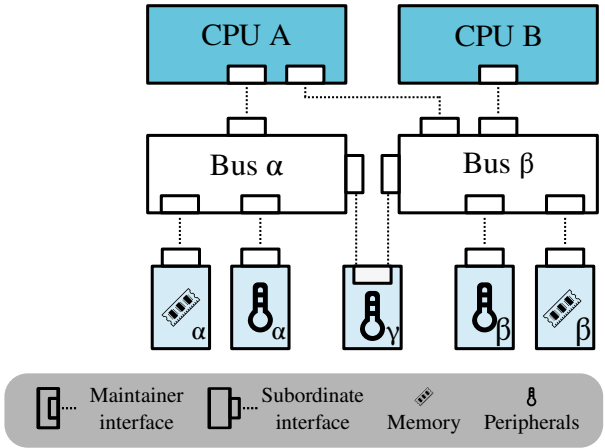
G2: Analysis of affected devices on the market

System and Threat Model

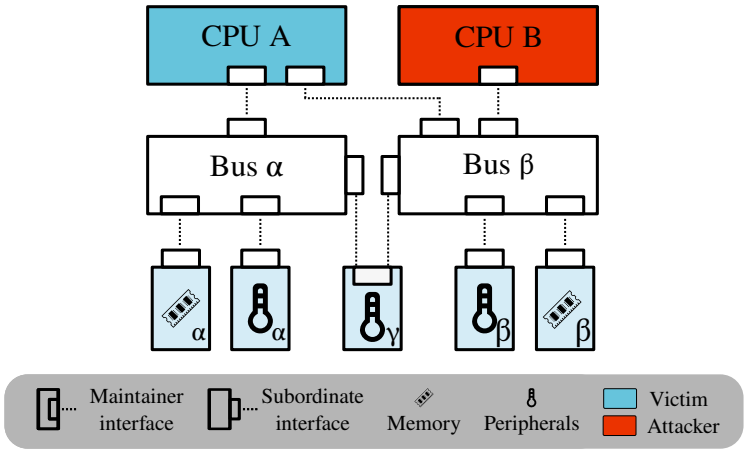
System and Threat Model



System and Threat Model



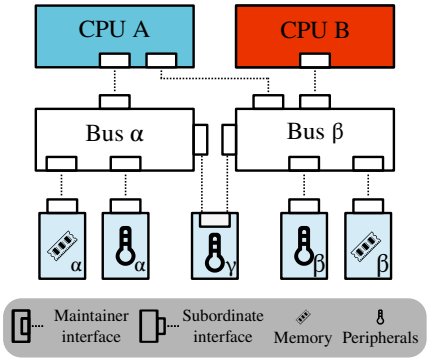
System and Threat Model



Initial Analysis

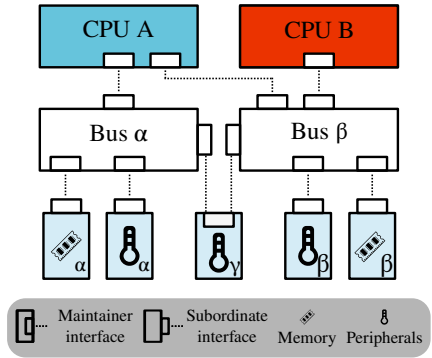
Initial Analysis

- We analyze memory, bus, peripheral domain and CPU communication



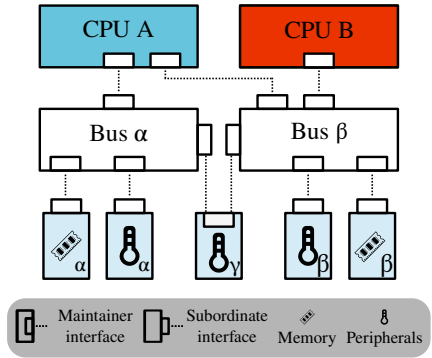
Initial Analysis

- We analyze memory, bus, peripheral domain and CPU communication
- We identify 4 attack vectors (AV):
  - AV1: MPU policy desynchronization
  - AV2: Unsynchronized communication channels
  - AV3: Non-exclusive peripheral access
  - AV4: Confused deputy peripheral



Initial Analysis

- We analyze memory, bus, peripheral domain and CPU communication
- We identify 4 attack vectors (AV):
  - AV2: Unsynchronized communication channels
  - AV4: Confused deputy peripheral



AV2: Unsynchronized Communication Channels

AV2: Unsynchronized Communication Channels

- CPUs need to coordinate resource access

AV2: Unsynchronized Communication Channels

- CPUs need to coordinate resource access
- Traditional OS solution: message passing or shared memory

AV2: Unsynchronized Communication Channels

- CPUs need to coordinate resource access
- Traditional OS solution: message passing or shared memory
- Kernel enforces unique access

AV2: Unsynchronized Communication Channels

- CPUs need to coordinate resource access
- Traditional OS solution: message passing or shared memory
- Kernel enforces unique access
- No shared software instance between different CPUs

AV2: Unsynchronized Communication Channels

- CPUs need to coordinate resource access
- Traditional OS solution: message passing or shared memory
- Kernel enforces unique access
- No shared software instance between different CPUs
- Only hardware-enforced solutions work!

AV2: Unsynchronized Communication Channels

- CPUs need to coordinate resource access
- Traditional OS solution: message passing or shared memory
- Kernel enforces unique access
- No shared software instance between different CPUs
- Only hardware-enforced solutions work!
- Otherwise: race conditions and TOCTOU attacks!

AV2: FreeRTOS on STM32H755

AV2: FreeRTOS on STM32H755

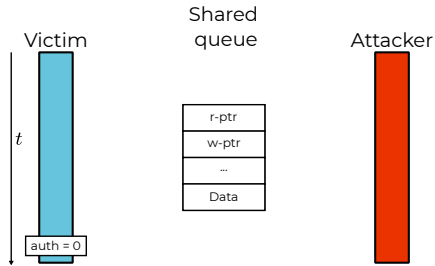
- Popular RTOS

AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)

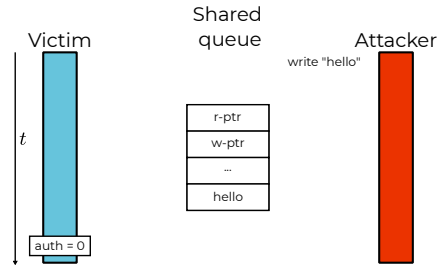
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



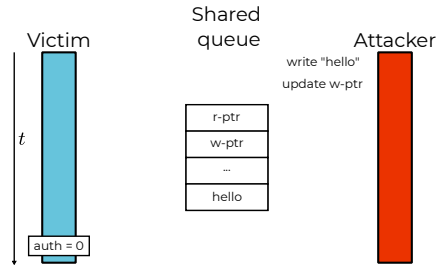
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



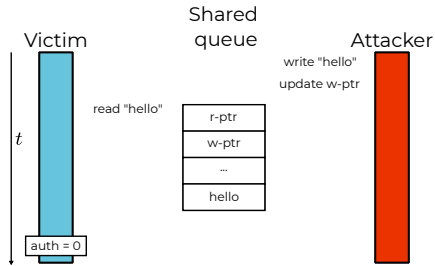
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



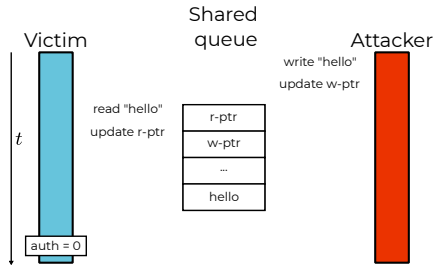
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



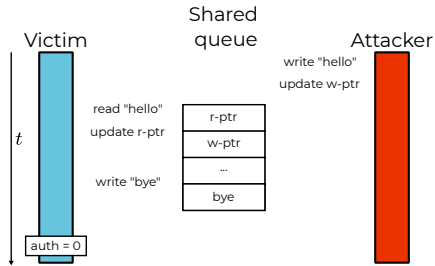
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



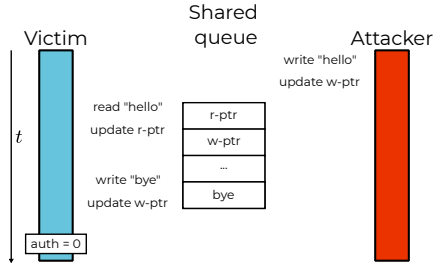
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



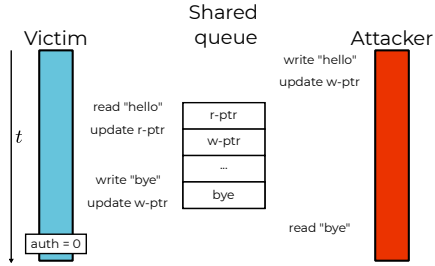
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



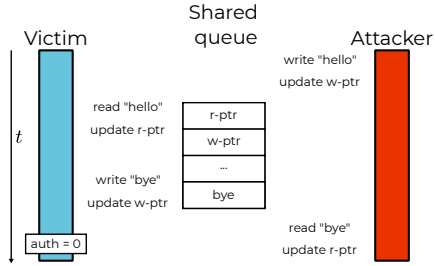
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



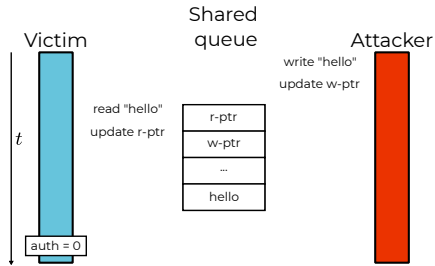
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)



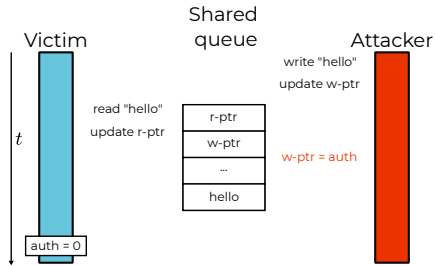
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it



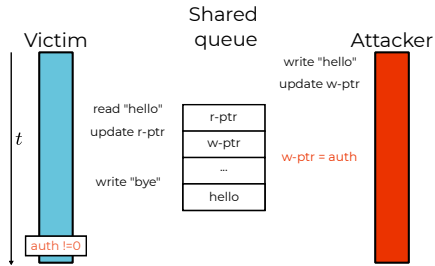
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it



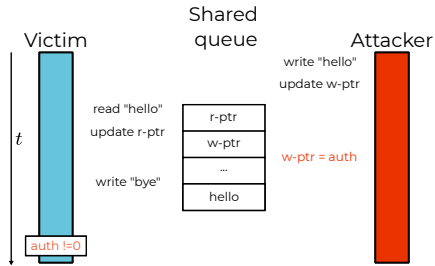
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it



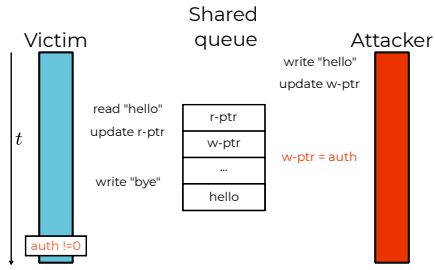
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it
- H755 implements hardware semaphores …



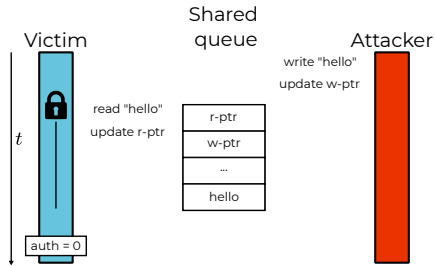
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it
- H755 implements hardware semaphores …
- … but no entity can enforce usage of the semaphores!



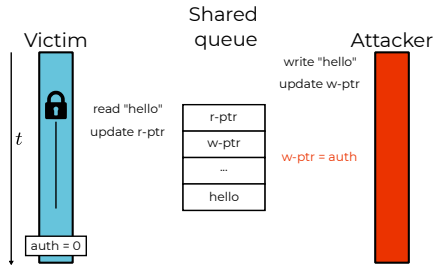
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it
- H755 implements hardware semaphores …
- … but no entity can enforce usage of the semaphores!



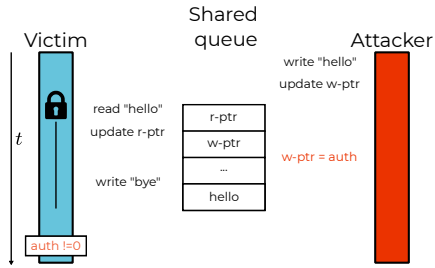
AV2: FreeRTOS on STM32H755

- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it
- H755 implements hardware semaphores …
- … but no entity can enforce usage of the semaphores!



AV2: FreeRTOS on STM32H755

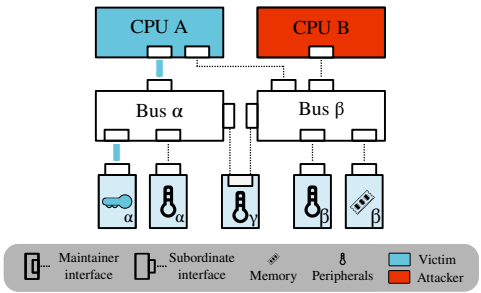
- Popular RTOS
- IPC via shared memory (queue struct)
- An attacker can manipulate the queue struct while the victim is using it
- H755 implements hardware semaphores …
- … but no entity can enforce usage of the semaphores!



AV4: Confused Deputy Peripheral

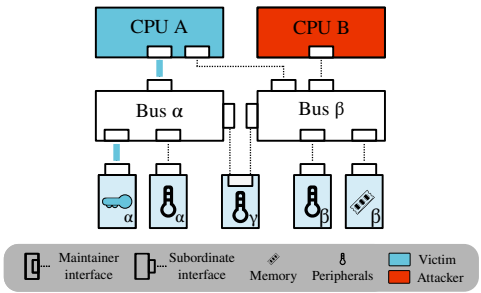
AV4: Confused Deputy Peripheral

- CPU A stores a secret in memory α



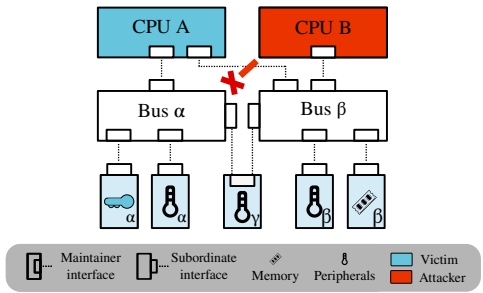
AV4: Confused Deputy Peripheral

- CPU A stores a secret in memory α
- CPU B wants to read the secret



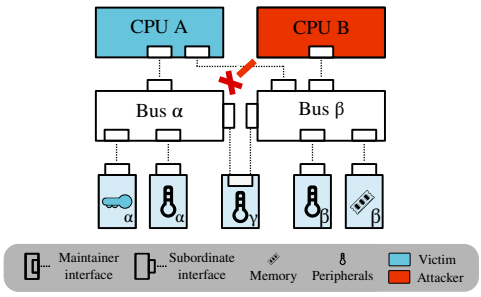
AV4: Confused Deputy Peripheral

- CPU A stores a secret in memory α
- CPU B wants to read the secret
- CPU B is physically not connected to memory α



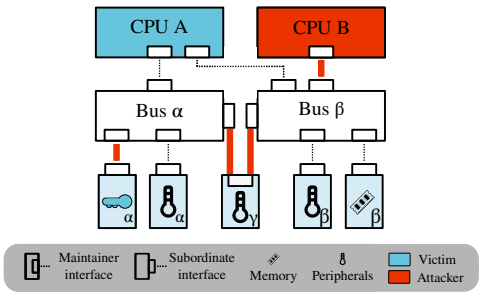
AV4: Confused Deputy Peripheral

- CPU A stores a secret in memory α
- CPU B wants to read the secret
- CPU B is physically not connected to memory α
- Peripheral γ is connected to both CPU B and to memory α



AV4: Confused Deputy Peripheral

- CPU A stores a secret in memory α
- CPU B wants to read the secret
- CPU B is physically not connected to memory α
- Peripheral γ is connected to both CPU B and to memory α
- CPU B can configure peripheral γ to access memory α



Vulnerability Assessment

Vulnerability Assessment

- Top 10 most important MCU manufacturers [1]

Vulnerability Assessment

- Top 10 most important MCU manufacturers [1]
- Collect all devices with multiple Cortex-M CPUs

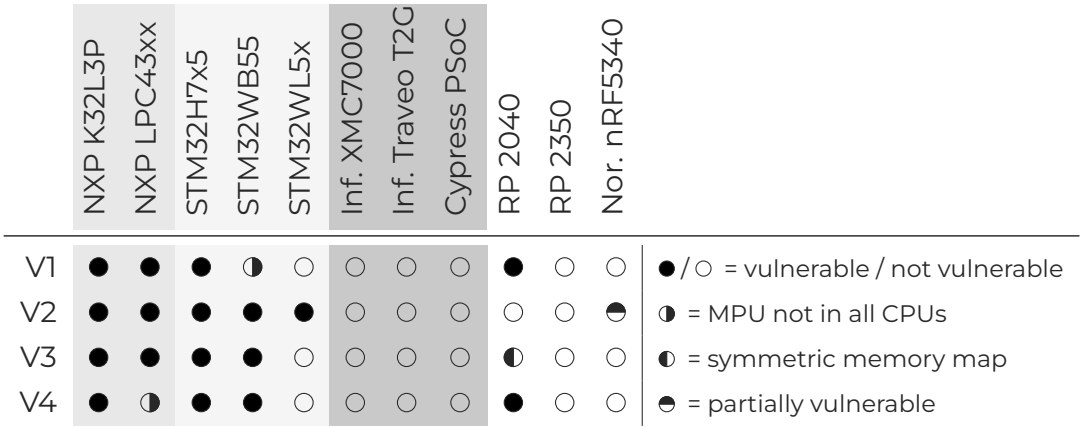
Vulnerability Assessment

- Top 10 most important MCU manufacturers [1]
- Collect all devices with multiple Cortex-M CPUs
- 11 device families

Vulnerability Assessment

- Top 10 most important MCU manufacturers [1]
- Collect all devices with multiple Cortex-M CPUs
- 11 device families
- Some device families > 60 unique, active devices

Vulnerability Assessment



MCUs with multiple CPUs and their potential vulnerability to the attack vectors.

Galaxy Ring Case Study

Galaxy Ring Case Study

- Fitness tracker in ring form

Galaxy Ring Case Study

- Fitness tracker in ring form
- Uses Nordic nrf5340

Galaxy Ring Case Study

- Fitness tracker in ring form
- Uses Nordic nrf5340
- Runs Zephyr RTOS on net and app core

Galaxy Ring Case Study

- Fitness tracker in ring form
- Uses Nordic nrf5340
- Runs Zephyr RTOS on net and app core
- Zephyr threat model: separate kernel and user mode

Galaxy Ring Case Study

- Fitness tracker in ring form
- Uses Nordic nrf5340
- Runs Zephyr RTOS on net and app core
- Zephyr threat model: separate kernel and user mode
- App core: sensing/processing logic (user mode)

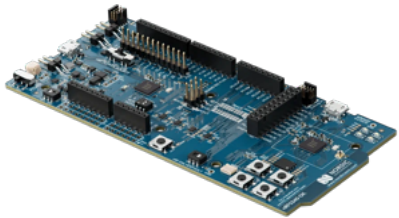
Galaxy Ring Case Study

- Fitness tracker in ring form
- Uses Nordic nrf5340
- Runs Zephyr RTOS on net and app core
- Zephyr threat model: separate kernel and user mode
- App core: sensing/processing logic (user mode)
- Net core: BLE setup very close to Zephyr reference (kernel mode)

Galaxy Ring Case Study

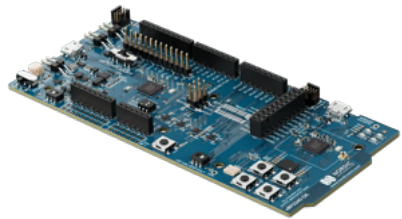
Galaxy Ring Case Study

- Analogous setup on nRF5340 DK



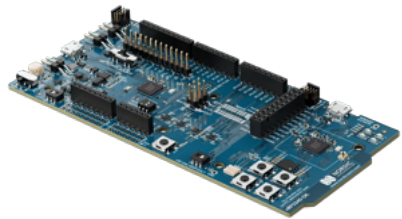
Galaxy Ring Case Study

- Analogous setup on nRF5340 DK
  - Net core firmware: Zephyr reference



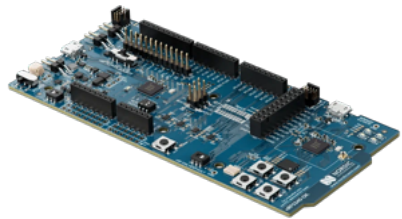
Galaxy Ring Case Study

- Analogous setup on nRF5340 DK
  - Net core firmware: Zephyr reference
  - App core firmware: dummy firmware



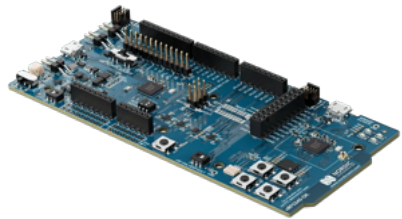
Galaxy Ring Case Study

- Analogous setup on nRF5340 DK
  - Net core firmware: Zephyr reference
  - App core firmware: dummy firmware
- Add vulnerability in net core firmware



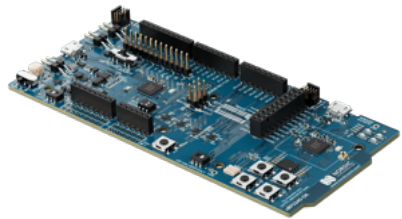
Galaxy Ring Case Study

- Analogous setup on nRF5340 DK
  - Net core firmware: Zephyr reference
  - App core firmware: dummy firmware
- Add vulnerability in net core firmware
- Code execution on net core
  → kernel-level access on app core!



Galaxy Ring Case Study

- Analogous setup on nRF5340 DK
  - Net core firmware: Zephyr reference
  - App core firmware: dummy firmware
- Add vulnerability in net core firmware
- Code execution on net core
  → kernel-level access on app core!
- On Galaxy Ring: net core vulnerability might lead to access of health data!



Disclosure

Disclosure

- ST: security advisory

Disclosure

- ST: security advisory
- Samsung, Nordic and Zephyr

Disclosure

- ST: security advisory
- Samsung, Nordic and Zephyr

  → responsibility diffusion

Countermeasures

Countermeasures

- Software countermeasures

Countermeasures

- Software countermeasures
  - AV2: reduce metadata in shared memory
  - AV2: copy data to inaccessible memory
  - AV4: none

Countermeasures

- Software countermeasures
  - AV2: reduce metadata in shared memory
  - AV2: copy data to inaccessible memory
  - AV4: none
- Hardware countermeasures

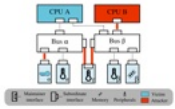
Countermeasures

- Software countermeasures
  - AV2: reduce metadata in shared memory
  - AV2: copy data to inaccessible memory
  - AV4: none
- Hardware countermeasures
  - AV2: Hardware queues
  - AV4: Bus-level access management

Conclusion

AV4: Confused Deputy Peripheral

- CPU A stores a secret in memory α
- CPU B wants to read the secret
- CPU B is physically not connected to memory α
- Peripheral γ is connected to both CPU B and to memory α
- CPU B can configure peripheral γ to access memory α



Vulnerability Assessment

	NXP K32L3P	NXP LPC43xx	STM32H7x5	STM32WB55	STM32WL5x	Inf. XMC7000	Inf. Traveo T2G	Cypress PSoC	RP 2040	RP 2350	Nor. nRF5340
V1	●	●	●	◑	○	○	○	○	●	○	○
V2	●	●	●	●	●	○	○	○	○	○	◓
V3	●	●	●	●	○	○	○	○	◐	○	○
V4	●	◑	●	●	○	○	○	○	●	○	○

● / ○ = vulnerable / not vulnerable
◑ = MPU not in all CPUs
◐ = symmetric memory map
◓ = partially vulnerable

MCUs with multiple CPUs and their potential vulnerability to the attack vectors.

Galaxy Ring case study

- Fitness tracker in ring form
- Uses Nordic nrf5340
- Runs Zephyr RTOS on net and app core
- Zephyr threat model: separate kernel and user mode
- App core: sensing/processing logic (user mode)
- Net core: minimal BLE setup (kernel mode)
- Code execution in net core gives kernel-level access to app core with health data!

Countermeasures

- Software countermeasures
  - AV2: reduce metadata in shared memory
  - AV2: copy data to inaccessible memory
  - AV4: none
- Hardware countermeasures
  - AV2: Hardware queues
  - AV4: Bus-level access management

References

[1] Infineon. **Infineon Second Quarter FY 2022 Quarterly Update.** https://www.infineon.com/dgdl/2022-05-09+Q2+FY22+Investor+Presentation.pdf?fileId=8ac78c8b808544e20180a4d32bb70009. Accessed: 2024-08-14.