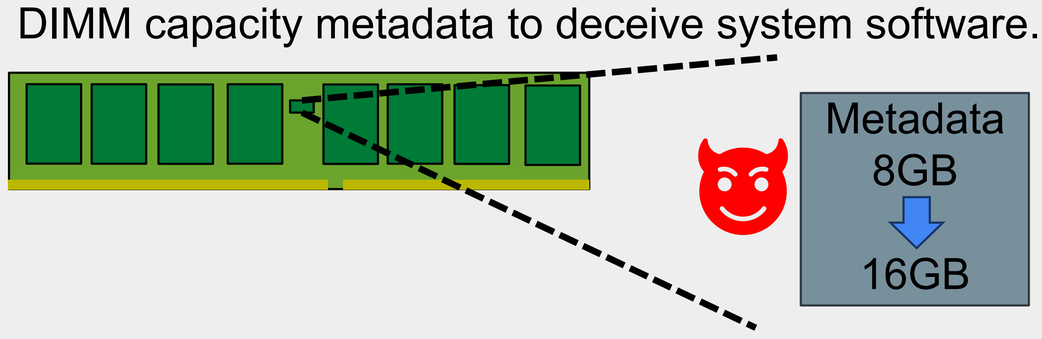
BadRAM Attack Detection from User-Level Privileges

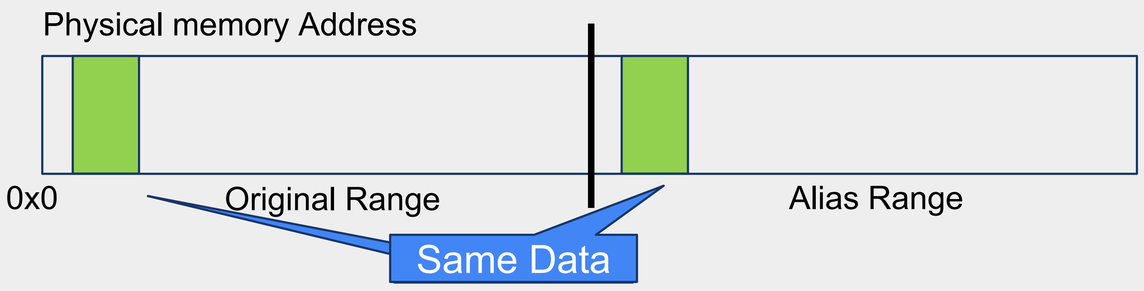
Shuta Wakamiya, Soramichi Akiyama
Advanced System Group, Ritsumeikan University

-Background-

- **BadRAM Attack** [1] :
  Creates aliases in physical address space by spoofing DIMM capacity metadata to deceive system software.



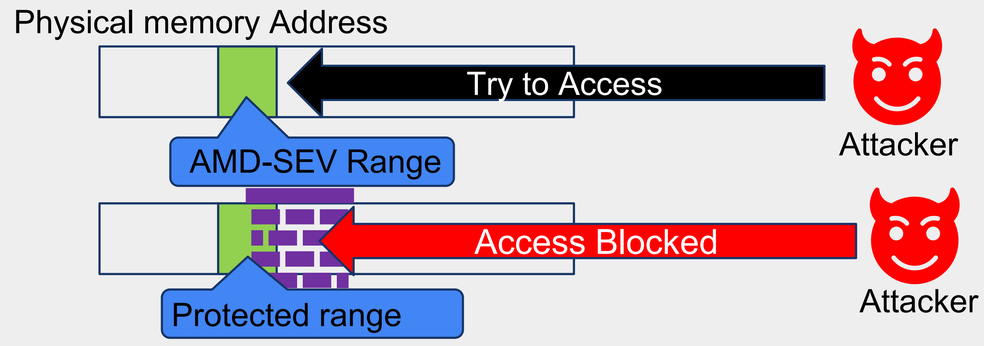
- **Attack Effects**



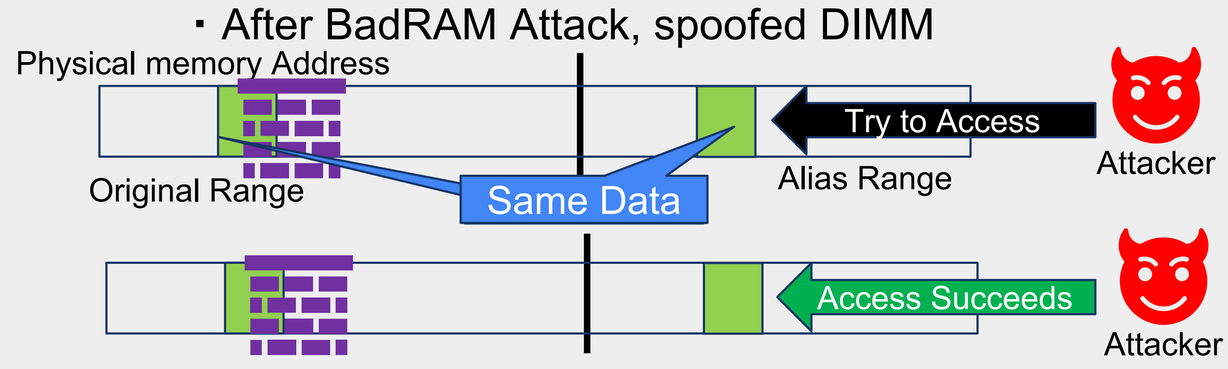
- BadRAM creates aliased physical addresses
  - point to the same memory location
  - Software sees the same data in two addresses

- **Breaking the trust of AMD SEV-SNP**
  - AMD SEV-SNP: Trusted Execution Environment (TEE)
  - Hardware-based VM protection
  - Protects against malicious hypervisor

- Normal Operation, non-spoofed DIMM



- After BadRAM Attack, spoofed DIMM



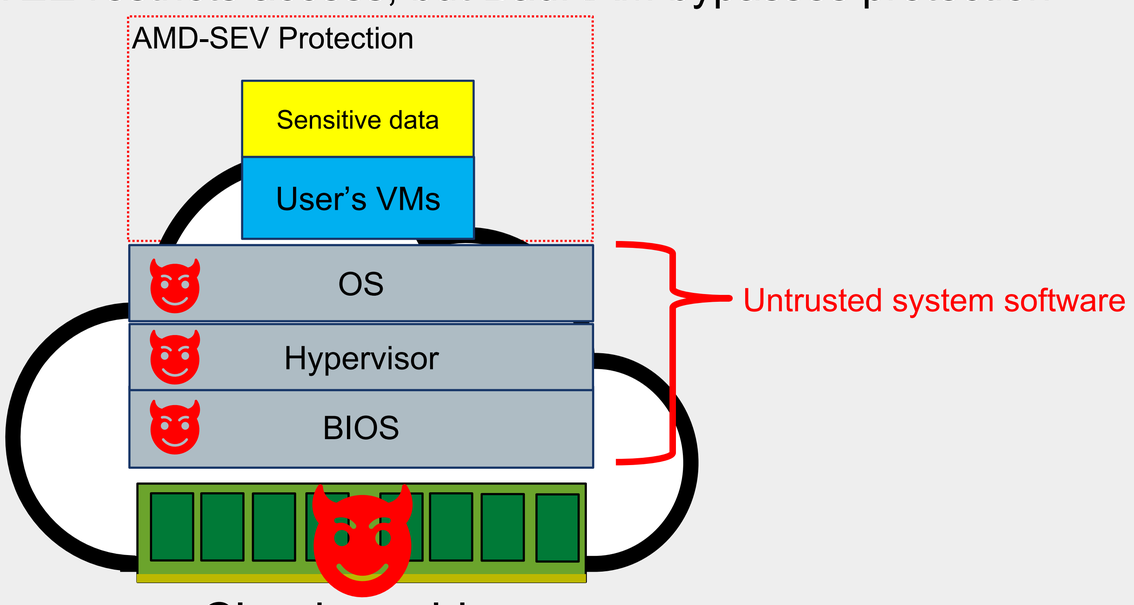
- No protection applied to aliased address
- System sees this as normal memory
- Same memory data as protected range

-Problem-

- **Existing Solutions**
  - DIMM Standard Changes
    Strictly protect metadata
  - BIOS/Firmware Updates
    Add memory alias detection at boot time
  - Hardware-based Detection
    Built-in alias detection in CPU/chipset

- **Limitations**
  - DIMM Standard Changes
    Impossible for already deployed memory modules
  - BIOS/Firmware Updates
    Dependent on cloud providers' decisions
  - Hardware-based Detection
    Not applicable update existing systems

- **Threat Model**
  - Malicious cloud provider (insider threat)
  - Physical access to insert tampered DIMM modules
  - Cloud Provider adopts AMD-SEV
  - All system software under attacker's control →
    TEE restricts access, but BadRAM bypasses protection

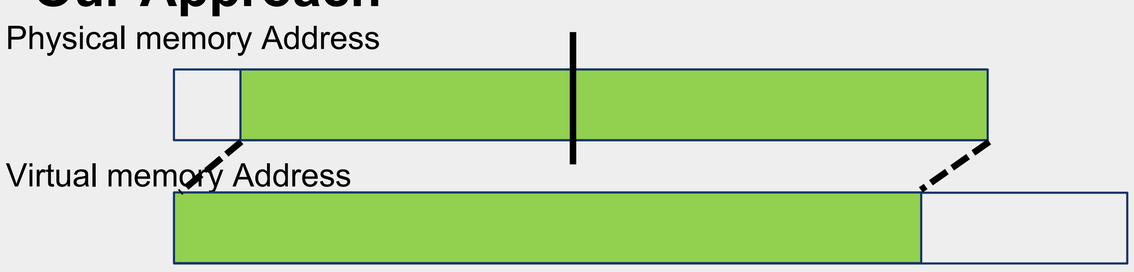


-Our Research Goal-

Detect BadRAM attacks without privileged access

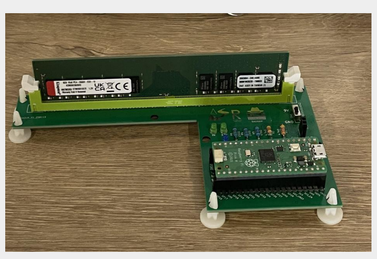
Detect memory aliasing from user-level processes using virtual memory allocation

- **Our Approach**



Large virtual address space allocation → detect memory aliasing

- **Current Progress**
  - BadRAM Attack Reproduction
    - Created custom PCB for metadata modification based on original paper's circuit design
    - Identified original-to-alias address correspondence verified memory aliasing mechanism



```
現在データ (0x0000000000100000): 49 5f 41 4d 5f 53 68 75 74 61 5f 57 41 4b 41 4d 49 59 41 | I_AM_Shuta_WAKAMIYA
現在データ (0x0000000000100000): 49 5f 41 4d 5f 53 68 75 74 61 5f 57 41 4b 41 4d 49 59 41 | I_AM_Shuta_WAKAMIYA
```

Original Address

```
アドレス 0x0000000200108000 に "IAMBADRAM" (10 バイト) を1秒おきに書き込みます。
Ctrl+C で終了。
書込完了: 0x0000000200108000
書込完了: 0x0000000200108000
```

Write at alias Address

```
現在データ (0x0000000000100000): 49 41 4d 42 41 44 52 41 4d 00 5f 57 41 4b 41 4d 49 59 41 | IAMBADRAM._WAKAMIYA
現在データ (0x0000000000100000): 49 41 4d 42 41 44 52 41 4d 00 5f 57 41 4b 41 4d 49 59 41 | IAMBADRAM._WAKAMIYA
```

Original Address infringed

[1] De Meulemeester, Jesse, et al. "BadRAM: Practical memory aliasing attacks on trusted execution environments." 2025 IEEE Symposium on Security and Privacy (SP). IEEE, 2025.