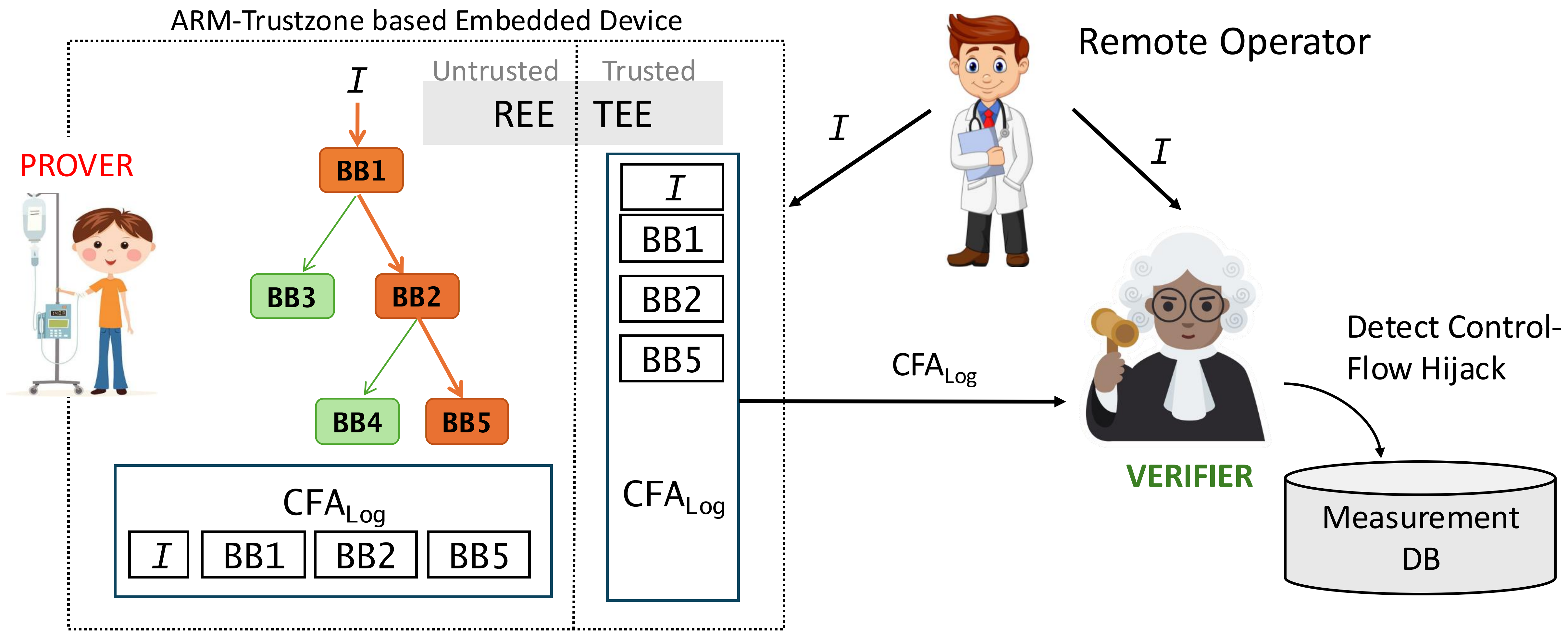
Non-Bare-Metal User-Space Control-Flow Attestation

Nikita Yadav, Hrushikesh Salunke, Dev Tejas Gandhi, Vinod Ganapathy

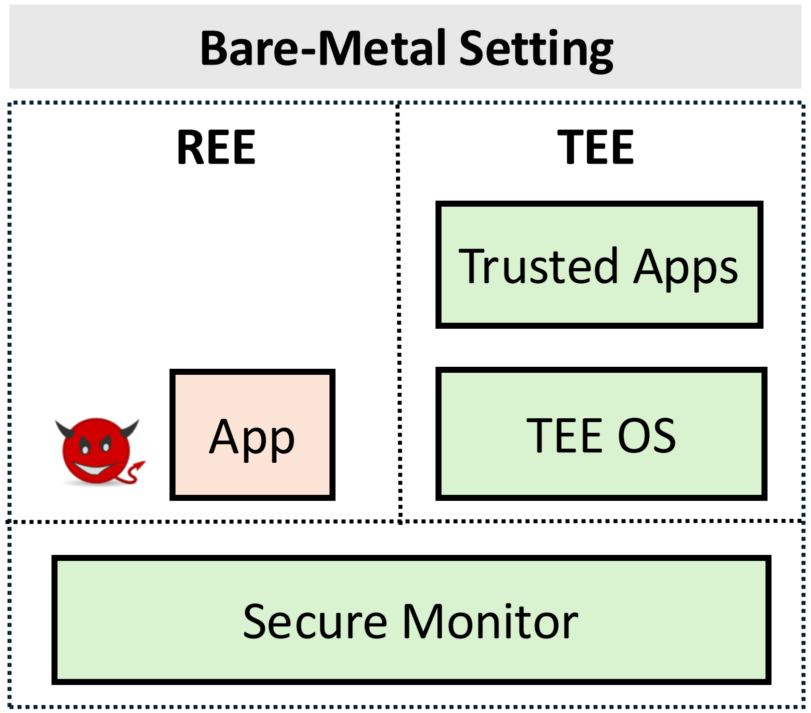
Indian Institute of Science, Bangalore

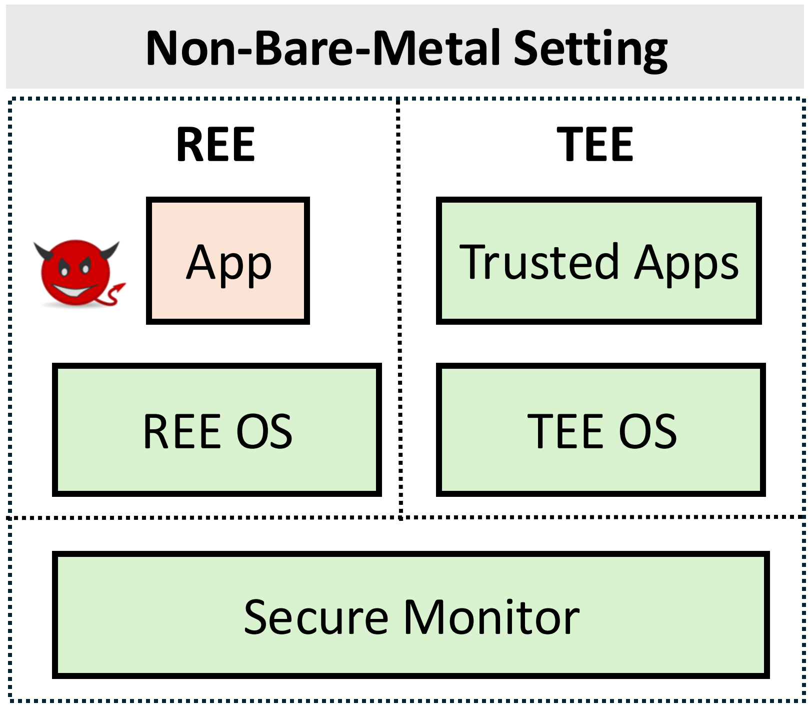
Control-Flow Attestation (CFA)



R(T)EE: Rich(Trusted) Execution Environment

CFA Threat Model and System Assumptions





Bare-metal CFA assumes full trust in TEE.

Non-bare-metal CFA trusts REE OS (this is risky).

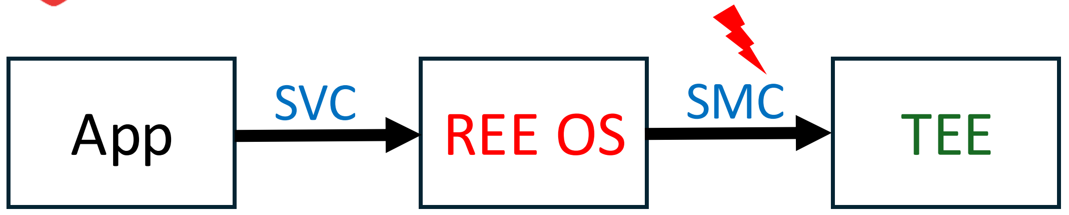
The Risk of Trusting the REE OS in CFA

If REE OS is Compromised, CFA cannot be trusted.

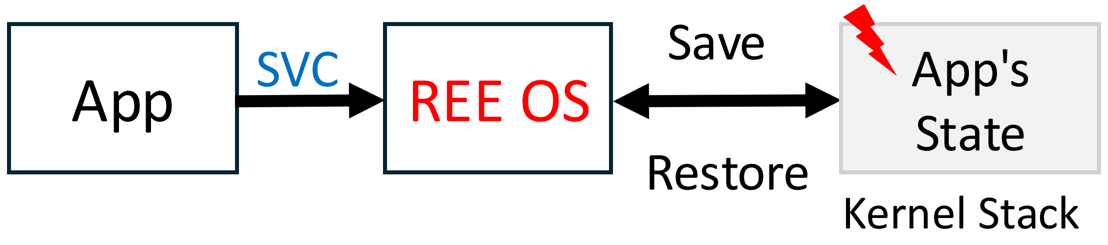
Write-Protect the App Binary

```
msr SCTRL_EL1, x0  // Reset WXN bit
```

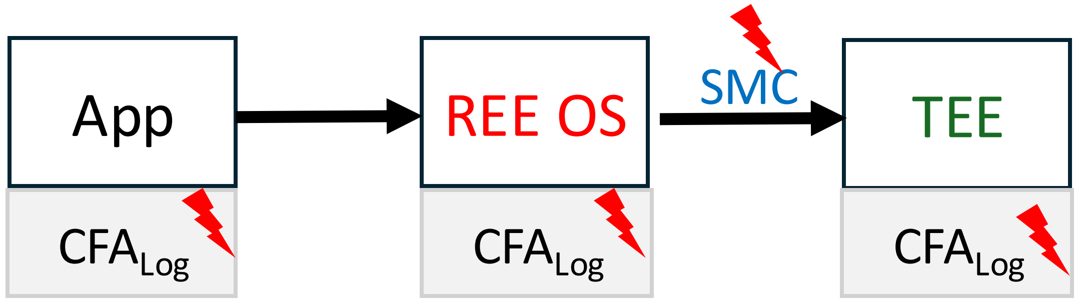
Secure Communication with TEE



Integrity of State During Interrupts



Integrity of CFA_{Log}



Key Takeaway: CFA Needs Extra Protection

CFA is not secure if the REE OS is compromised.

Extra Protections are essential for CFA in non-bare-metal settings.

Sulfur: Non-Bare-Metal User-Space CFA

Trusting the REE OS is fraught with risks, Sulfur raises the bar.

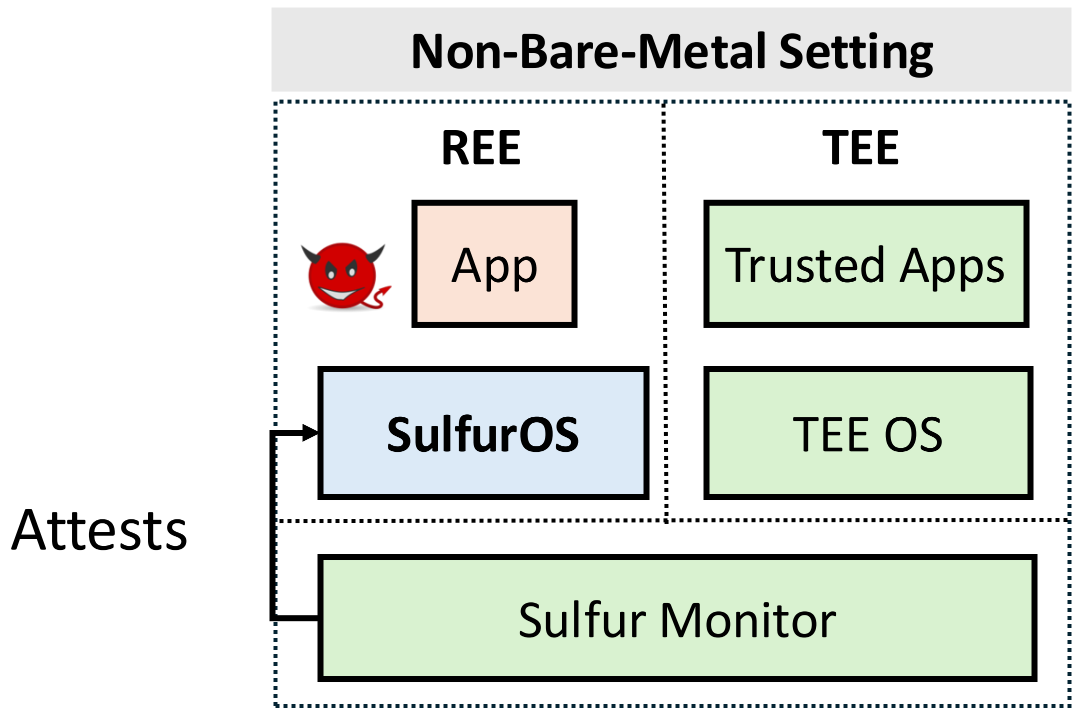
Sulfur: Core Assumptions and Scope

- Sulfur inherits all the standard assumptions from prior CFA approaches, e.g.,
  - The program is instrumented for CFA.
  - Program's image is attested before execution.

- Sulfur also requires the prover platform to be equipped with a TEE.

Note: We rule out data-only attacks.

SulfurOS: A Secure Middle Ground for CFA



Sulfur introduces SulfurOS + Sulfur Monitor to protect CFA even if REE OS is attacked.

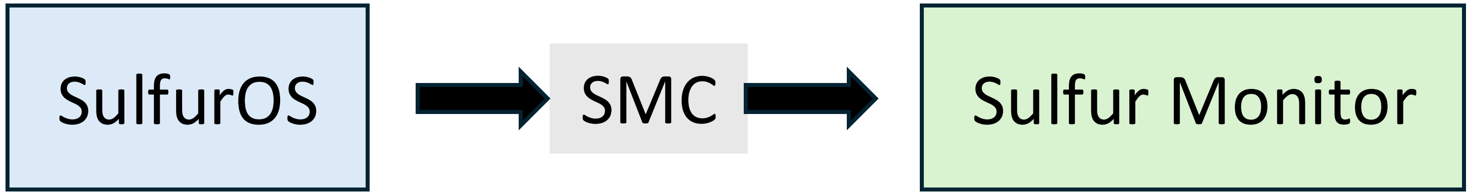
TCB: TEE (including the Sulfur Monitor).

Sulfur's Design: Two Layers of Defense

CFA-Centric Protections → Secures CFA artifacts

System Integrity Guardrails → Protects REE OS and System State

Guardrails: Protecting REE OS & System State



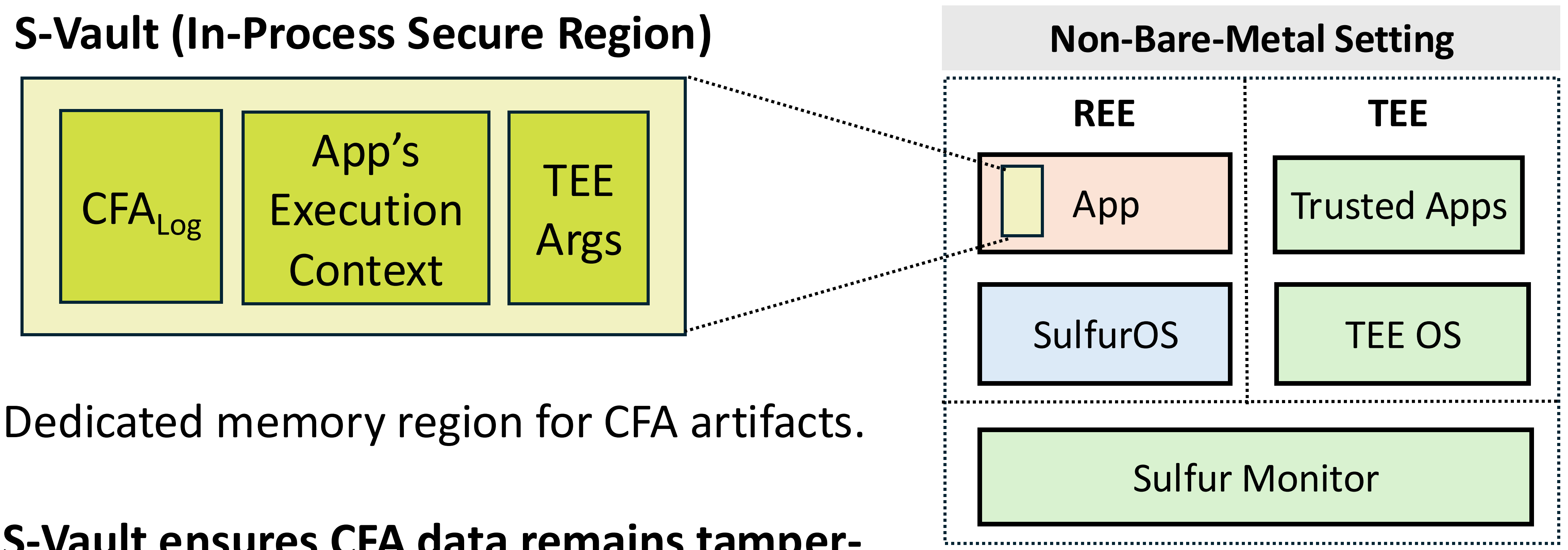
- Redirect security-sensitive operations to Sulfur Monitor for verification.
- Enforce security policies (e.g. WXN) and page table integrity.
- Harden Kernel with PACBTI-based CFI.

Example: Sulfur replaces direct system register writes with SMC to enforce integrity.

$$\begin{tikzcd} \textit{msr SCTRL\_EL1, x0} \arrow[r] & \textit{... smc \#0 ...} \end{tikzcd}$$

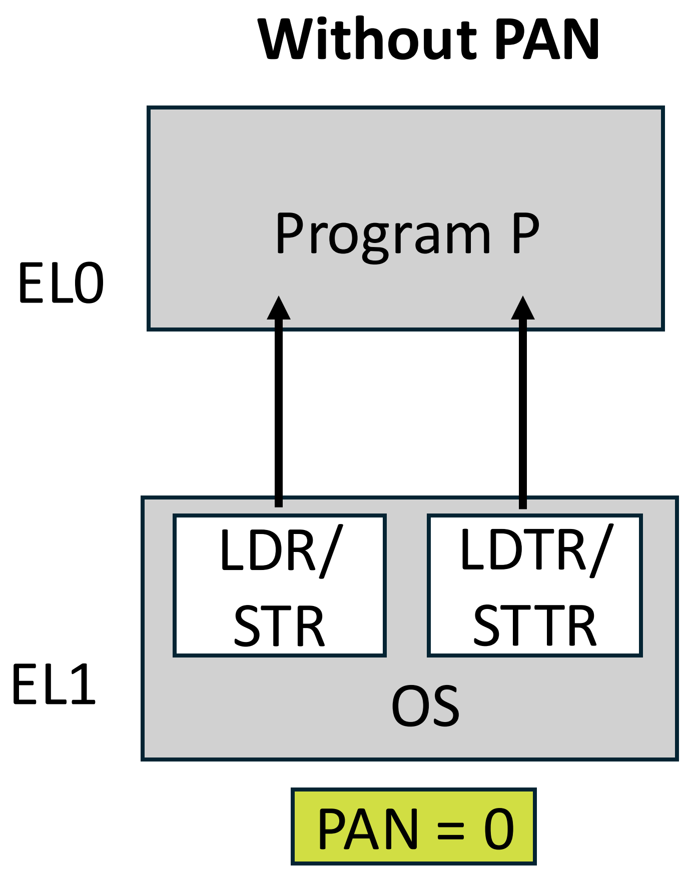
PAC: Pointer Authentication Code, BTI: Branch Target Identification

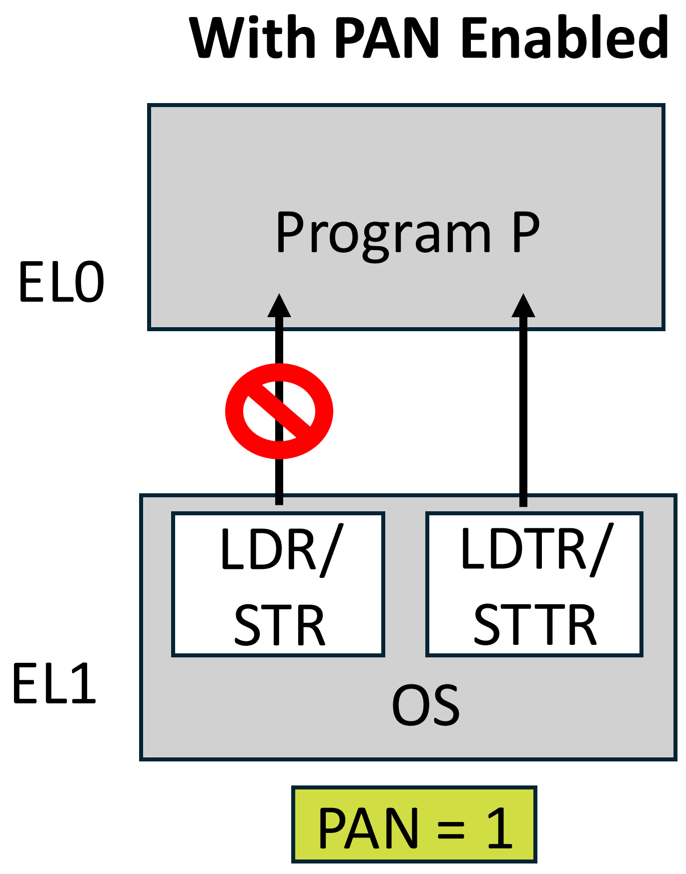
CFA-Centric Protections: Secure CFA Artifacts



- Dedicated memory region for CFA artifacts.
- **S-Vault ensures CFA data remains tamper-proof even if OS is compromised.**

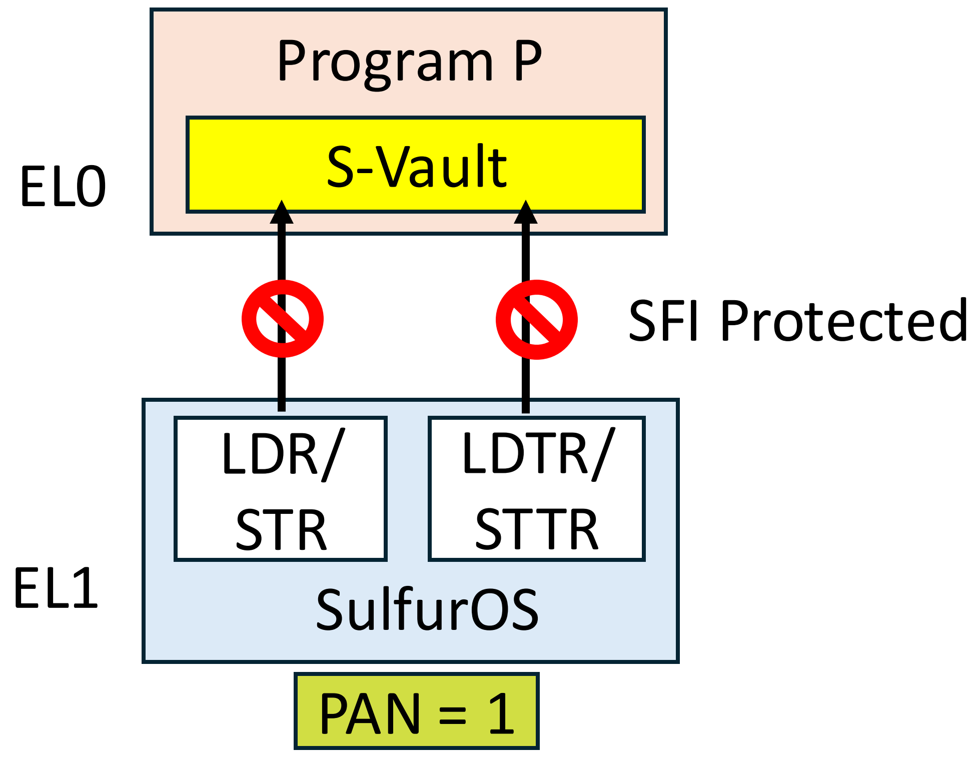
Key Insight: ARM PAN for Memory Isolation





PAN: Privileged Access Never

S-Vault: Protected User-Space Region



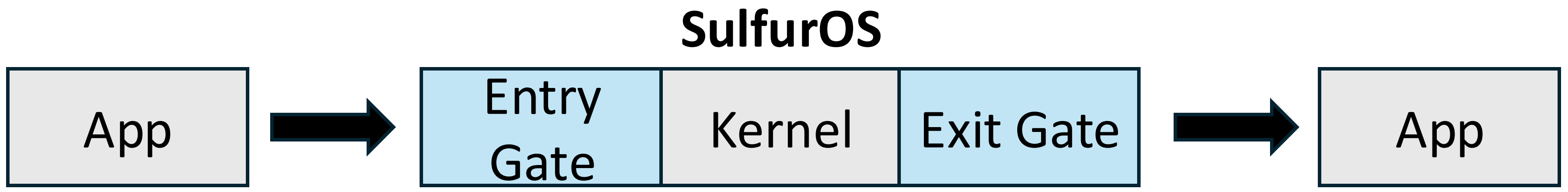
- SulfurOS uses LDTR/STTR for user memory access.
- SFI checks prevent writes to S-Vault.

SFI check:

```
and x3, addr, #0x7ffffffffff000
cmp x3, =S-Vault_addr
b.e abort
sttr reg, [addr]
```

Key Takeaway: S-Vault ensures CFA artifacts remain inaccessible to the SulfurOS.

Protecting App's Context with Gates & S-Vault



- Gates save/restore register state in S-Vault.
- Implemented at syscall entry/exit, interrupts, context switches.
- PACBTI ensures gates execute securely.

```
Entry_gate: // Save register state to S-Vault
ldr x19, =S-Vault_addr
sttr x0, [x19, #8 * 0]
sttr x1, [x19, #8 * 1]
...
sttr x30, [x19, #8 * 30]
```

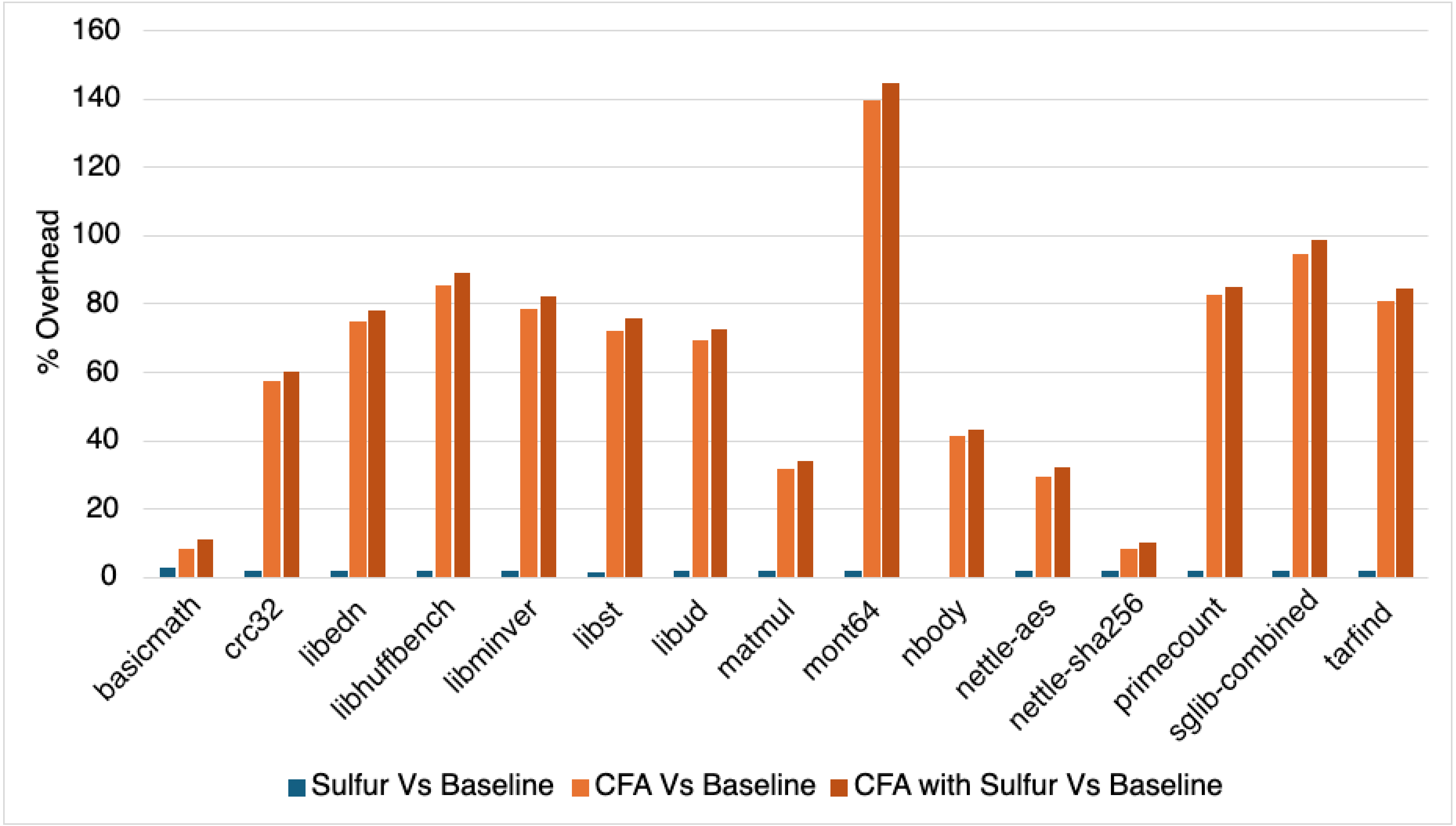
```
Exit_gate: // Restore register state from S-Vault
ldr x19, =S-Vault_addr
ldtr x0, [x19, #8 * 0]
ldtr x1, [x19, #8 * 1]
...
ldtr x30, [x19, #8 * 30]
```

Experimental Setup

- AArch64 Trustzone-based Prover Platform with PAN, PACBTI
- SulfurOS: Modified Linux REE OS
- OPTEE + Sulfur Monitor for TEE services.
- Prototype built on *Blast* CFA instrumentation.

Blast: Whole Program Control-Flow Attestation, CCS 2023.

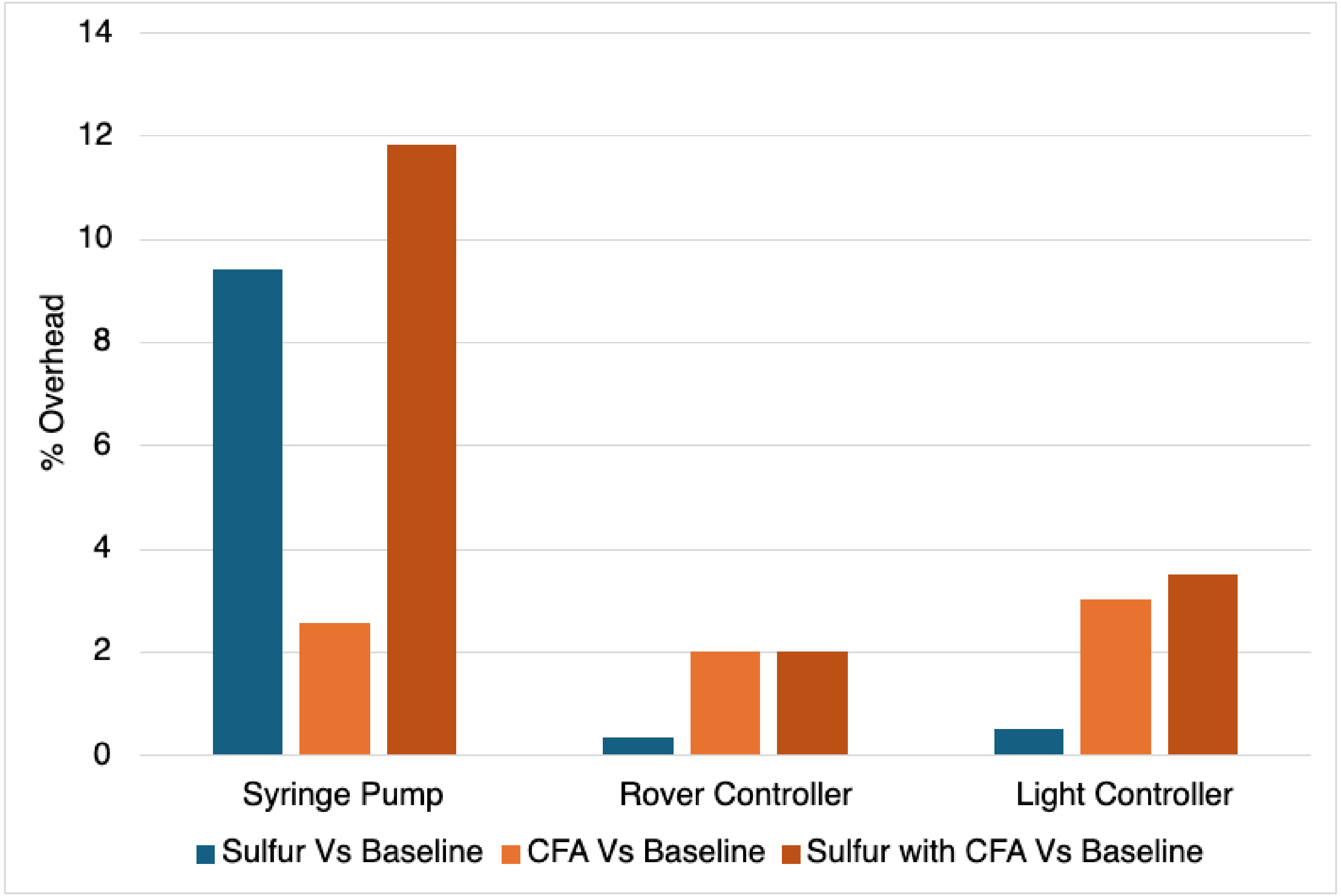
Evaluation: Embedded Benchmarks



Geo mean +1.67% for Sulfur alone

Geo mean +1.94% on top of CFA with Sulfur

Evaluation: Embedded Applications



Near-zero to moderate added cost; worst case stays <12% on top of CFA.

Summary

- Robust CFA in non-bare-metal settings.
- Protects CFA artifacts and execution state of the system/OS.
- Lightweight via hardware features (PAN/PACBTI).

Conclusion: Sulfur delivers robust CFA in non-bare-metal settings with low overhead.

Thank you

Full Paper: https://www.csa.iisc.ac.in/~vg/papers/acsac2025/

Artifact: https://github.com/sulfurcfa/Sulfur.git