SpecCFA: Enhancing Control Flow Attestation/Auditing via Application-Aware Sub-Path Speculation

Adam Caulfield, Liam Tyler, and Ivan De Oliveira Nunes

Rochester Institute of Technology

Annual Computer Security Applications Conference (ACSAC) 2024

ACSAC 2024

December 9-13, 2024 • Honolulu, Hawaii, USA

Embedded devices

- Low-cost, energy efficient MCUs

Embedded devices

- Low-cost, energy efficient MCUs

- Remotely deployed to execute safety-critical tasks in modern systems

Embedded devices

- Low-cost, energy efficient MCUs

- Remotely deployed to execute safety-critical tasks in modern systems
  - Sensor/alarm system
  - Transportation efficiency
  - Modern medical devices

Industrial IoT Platform Market Expands with Demand for Predictive Maintenance, and Real-Time Data Analytics

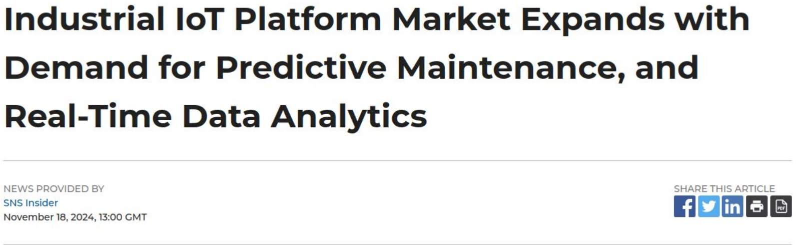
NEWS PROVIDED BY
SNS Insider
November 18, 2024, 13:00 GMT

SHARE THIS ARTICLE

Embedded devices

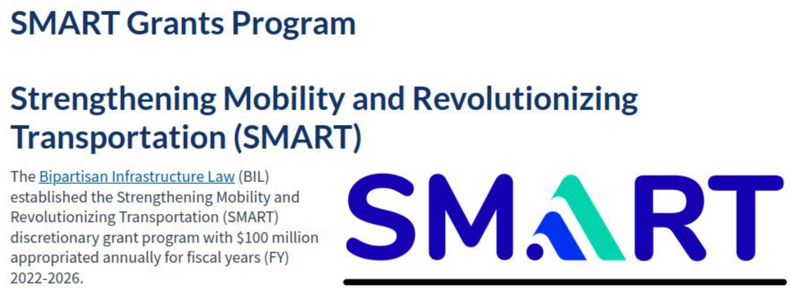
- Low-cost, energy efficient MCUs

- Remotely deployed to execute safety-critical tasks in modern systems
  - Sensor/alarm system
  - Transportation efficiency
  - Modern medical devices



Industrial IoT Platform Market Expands with Demand for Predictive Maintenance, and Real-Time Data Analytics

NEWS PROVIDED BY
SNS Insider
November 18, 2024, 13:00 GMT



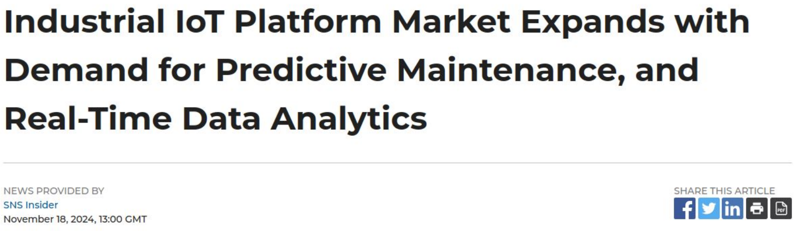
SMART Grants Program

Strengthening Mobility and Revolutionizing Transportation (SMART)

The Bipartisan Infrastructure Law (BIL) established the Strengthening Mobility and Revolutionizing Transportation (SMART) discretionary grant program with $100 million appropriated annually for fiscal years (FY) 2022-2026.

Embedded devices

- Low-cost, energy efficient MCUs
- Remotely deployed to execute safety-critical tasks in modern systems
  - Sensor/alarm system
  - Transportation efficiency
  - Modern medical devices
- Resource constrained
  - CPU & Memory
  - **Security!!!**
- Still used despite lack of inherent security



Industrial IoT Platform Market Expands with Demand for Predictive Maintenance, and Real-Time Data Analytics

NEWS PROVIDED BY
SNS Insider
November 18, 2024, 13:00 GMT

SHARE THIS ARTICLE

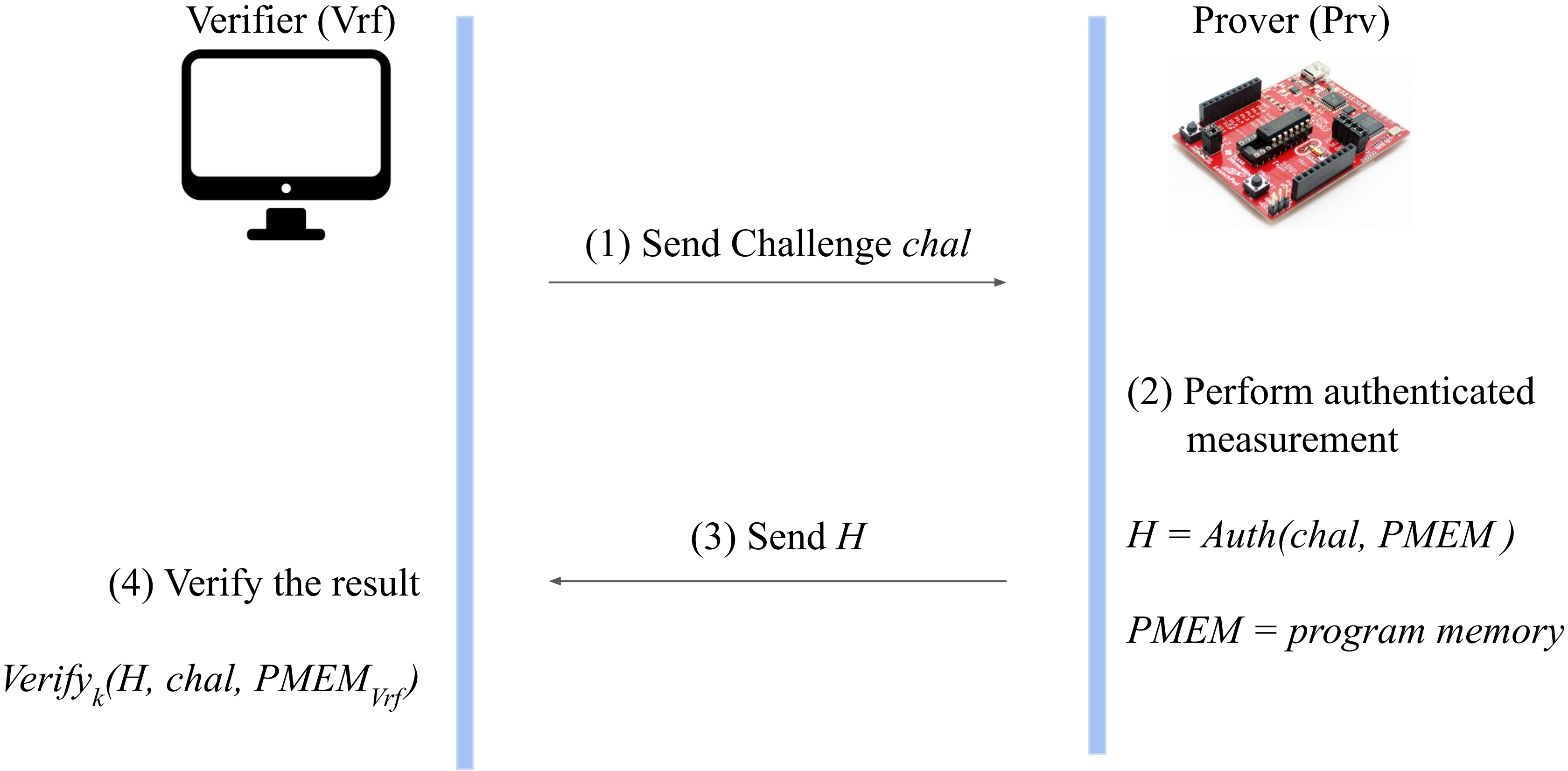
SMART Grants Program

Strengthening Mobility and Revolutionizing Transportation (SMART)

The Bipartisan Infrastructure Law (BIL) established the Strengthening Mobility and Revolutionizing Transportation (SMART) discretionary grant program with $100 million appropriated annually for fiscal years (FY) 2022-2026.

Remote Attestation (RA):

Detect software integrity compromise on remote device



Verifier (Vrf)

Prover (Prv)

(1) Send Challenge *chal*

(2) Perform authenticated measurement

$H = Auth(chal, PMEM\,)$

$PMEM = program\ memory$

(3) Send *H*

(4) Verify the result

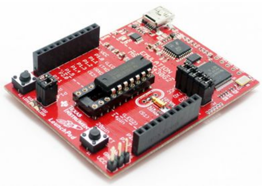
$Verify_k(H, chal, PMEM_{Vrf})$

Remote Attestation (RA):

Detect software integrity compromise on remote device

Verifier (Vrf)

Prover (Prv)



What about runtime attacks?

Control Flow Hijacking
Code Reuse Attacks

(2) Perform authenticated measurement

$H = Auth(chal, PMEM)$

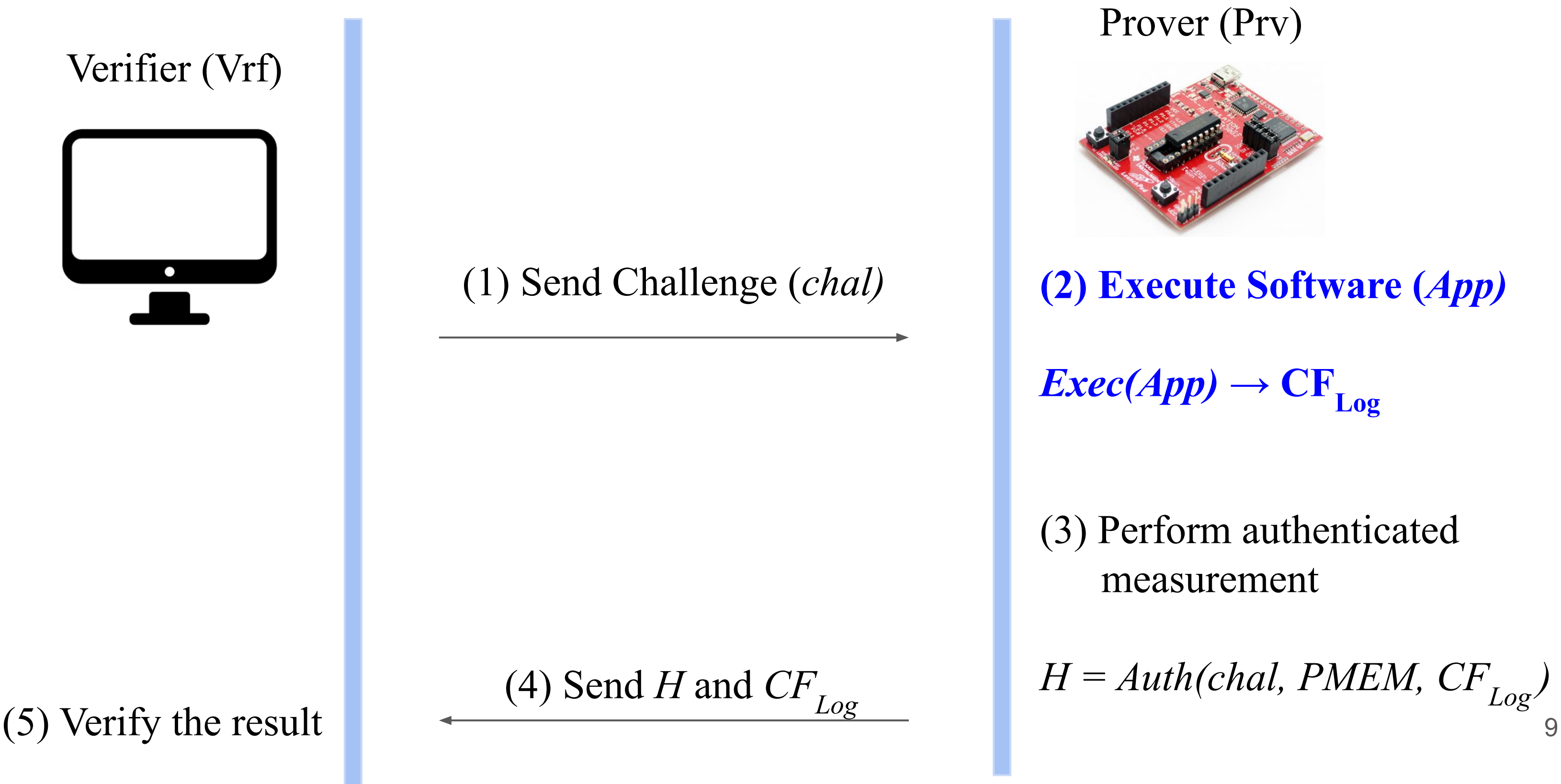
(4) Verify the result

$Verify_k(H, chal, PMEM_{Vrf})$

Control Flow Attestation/Auditing (CFA):

Extends CFA to provide additional security guarantees

- CF-Attestation: Generate authentic evidence of runtime behavior
- CF-Auditing: Guarantee Prover always responds with evidence of runtime behavior



Verifier (Vrf)

Prover (Prv)

(1) Send Challenge (*chal)*

(2) Execute Software (*App)*

$\boldsymbol{Exec(App) \rightarrow \mathrm{CF}_{\mathrm{Log}}}$

(3) Perform authenticated measurement

$H = Auth(chal, PMEM, CF_{Log})$

(4) Send H and CF_{Log}

(5) Verify the result

Storage/Communication is the bottleneck

- CFA without optimization is impractical
  - Application must periodically stop to transmit evidence

- The problem:
  - CF_{Log}s are LARGE
  - Complex software and/or continuous operations

- Some basic optimizations only take you so far:
  - Encoding/ignoring deterministic branches
  - Optimizing for "simple" loops
  - Approaches with these optimizations still incur significant overheads

Storage/Communication is the bottleneck

- What else can be leveraged?

- What about high-likelihood benign behavior?

Storage/Communication is the bottleneck

- What else can be leveraged?

- What about high-likelihood benign behavior?

- High-likelihood benign behavior:
  - Application specific
  - Parsing network commands
  - Control loops
  - Period sensing/actuation operations

Storage/Communication is the bottleneck

- What else can be leveraged?

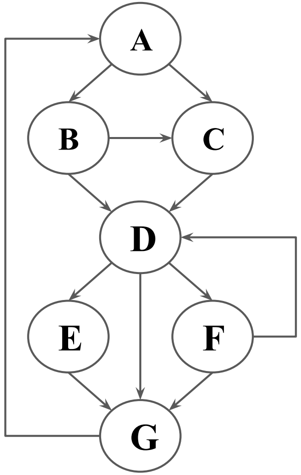
- What about high-likelihood benign behavior?

- High-likelihood benign behavior:
  - Application specific
  - Parsing network commands
  - Control loops
  - Period sensing/actuation operations

- Occur as ***program sub-paths***
  - Cannot be easily optimized with prior approaches

SpecCFA Example

Control Flow Graph

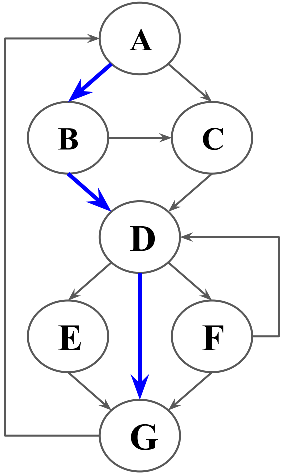


SpecCFA Example

Verifier:

{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph

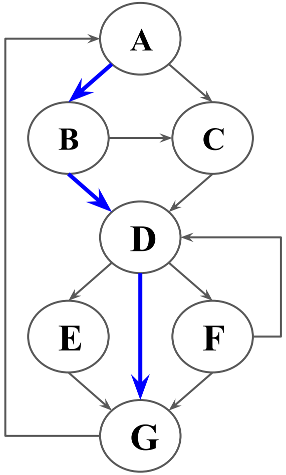


SpecCFA Example

Verifier:

{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



CF_{Log} during execution

(a)

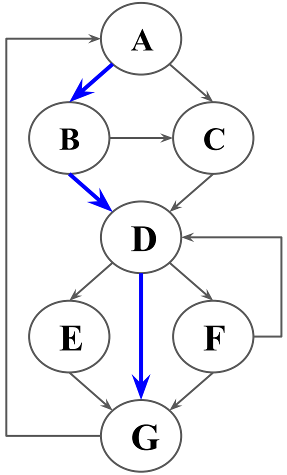
A

SpecCFA Example

Verifier:

{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



CF_{Log} during execution

(a)

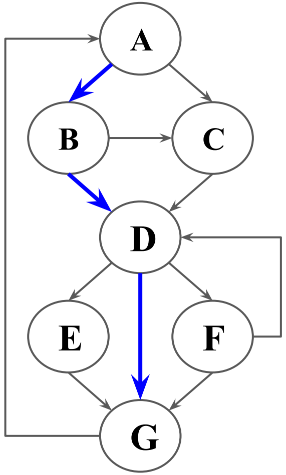
A
B

SpecCFA Example

Verifier:

{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



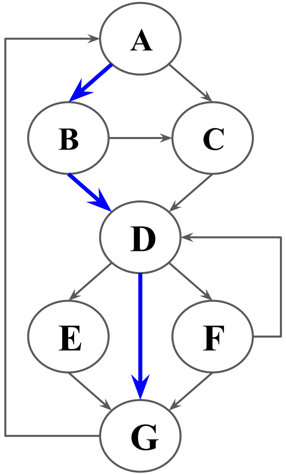
CF_{Log} during execution

(a)
A
B
D

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with "1"

Sub-path in Graph



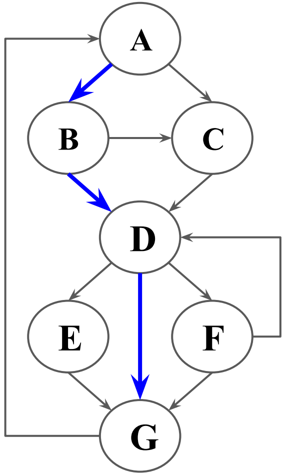
CF_{Log} during execution

(a)
A
B
D
G

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



CF_{Log} during execution

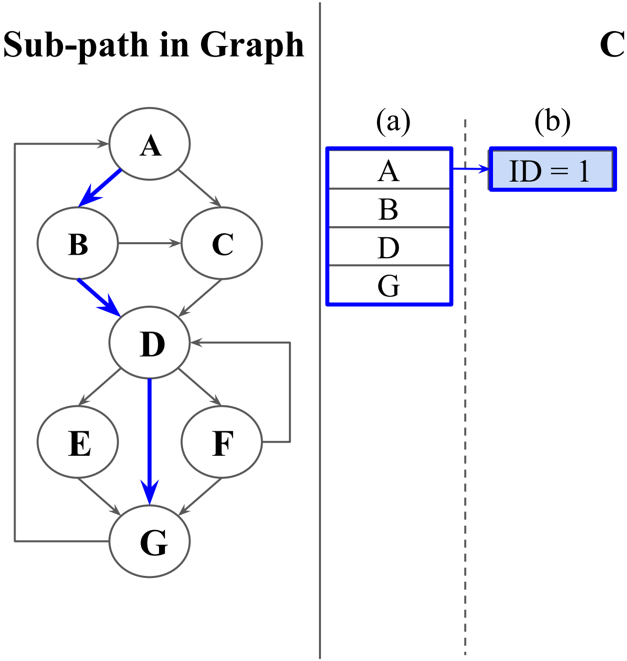
(a)

A
B
D
G

SpecCFA Example

Verifier:

{A,B,D,G} is high-likelihood! Replace with "1"



Sub-path in Graph

CF_{Log} during execution

(a)

A
B
D
G

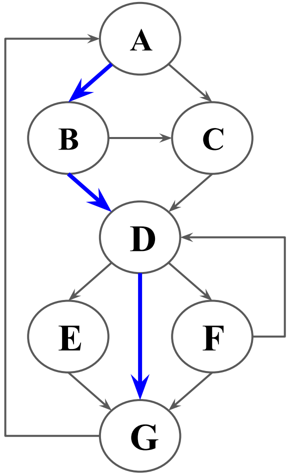
(b)

ID = 1

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



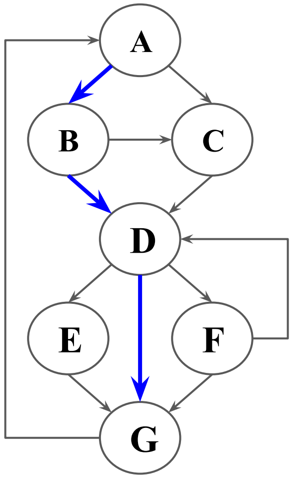
CF_{Log} during execution

(a)	(b)	(c)
A	ID = 1	ID = 1
B		A
D		C
G		D
		F
		G

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



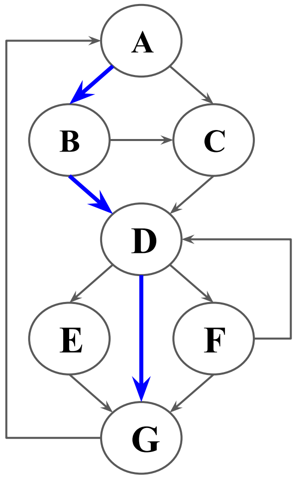
CF_{Log} during execution

(a)	(b)	(c)	(d)
A	ID = 1	ID = 1	ID = 1
B		A	A
D		C	C
G		D	D
		F	F
		G	G
			A
			B
			D
			G

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



CF_{Log} during execution

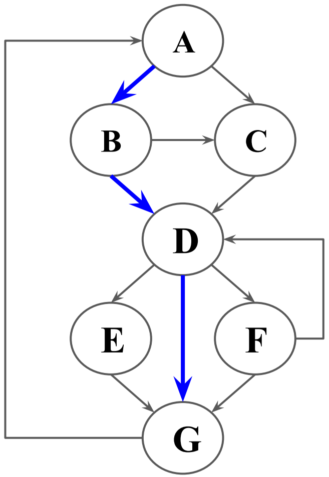
(a)	(b)	(c)	(d)	(e)
A	ID = 1	ID = 1	ID = 1	ID = 1
B		A	A	A
D		C	C	C
G		D	D	D
		F	F	F
		G	G	G
			A	ID = 1
			B	
			D	
			G	

SpecCFA Example

Verifier:

{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



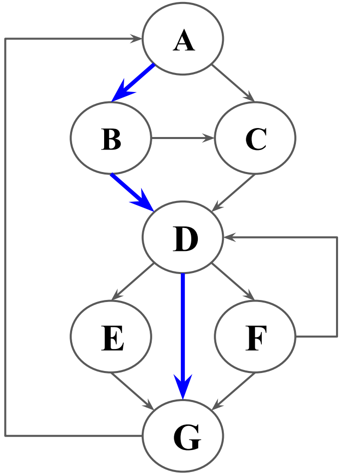
CF$_{Log}$ during execution

(a)	(b)	(c)	(d)	(e)	(f)
A	ID = 1	ID = 1	ID = 1	ID = 1	ID = 1
B		A	A	A	A
D		C	C	C	C
G		D	D	D	D
		F	F	F	F
		G	G	G	G
			A	ID = 1	ID = 1
			B		A
			D		B
			G		C
					D
					F
					G

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with “1”

Sub-path in Graph



CF_{Log} during execution

(a)	(b)	(c)	(d)	(e)	(f)
A	ID = 1	ID = 1	ID = 1	ID = 1	ID = 1
B		A	A	A	A
D		C	C	C	C
G		D	D	D	D
		F	F	F	F
		G	G	G	G
			A	ID = 1	ID = 1
			B		A
			D		B
			G		C
					D
					F
					G

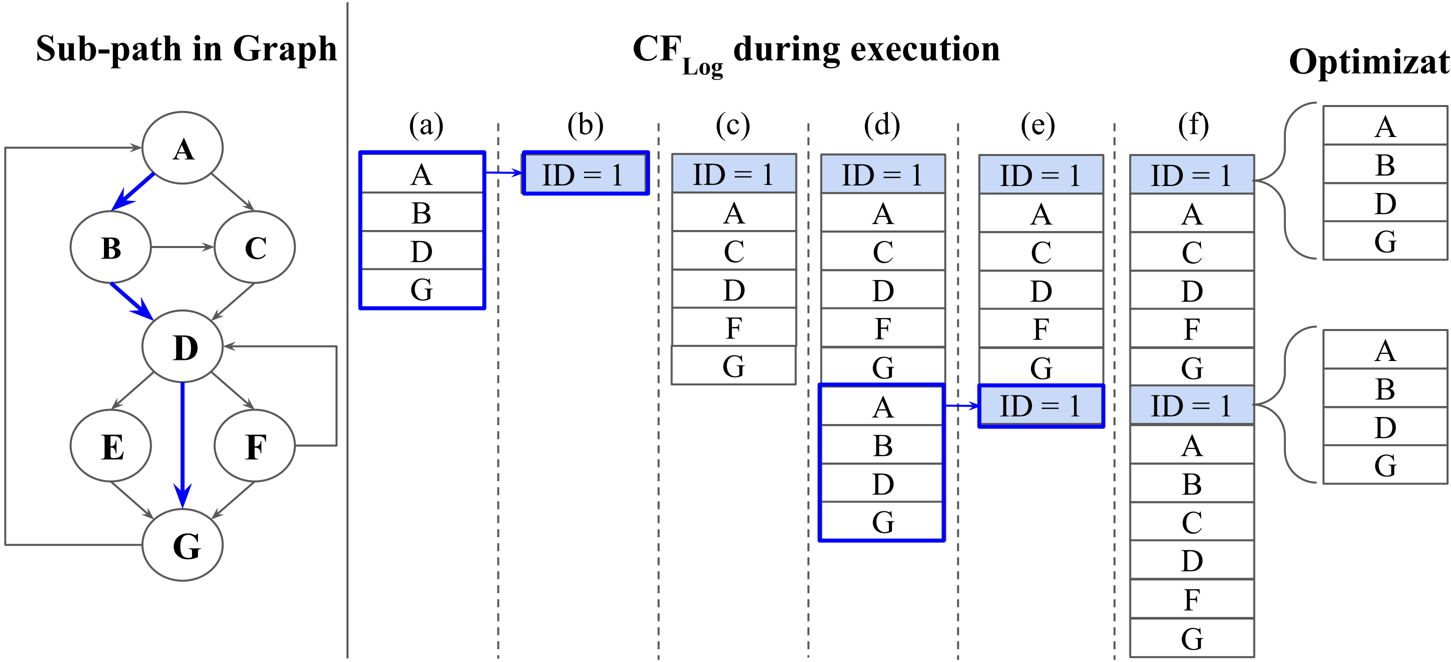
Optimization

A
B
D
G

A
B
D
G

SpecCFA Example

Verifier:
{A,B,D,G} is high-likelihood! Replace with “1”



Sub-path in Graph

CF_{Log} during execution

(a) A, B, D, G

(b) ID = 1

(c) ID = 1, A, C, D, F, G

(d) ID = 1, A, C, D, F, G, A, B, D, G

(e) ID = 1, A, C, D, F, G, ID = 1

(f) ID = 1, A, C, D, F, G, ID = 1, A, B, C, D, F, G

Optimization

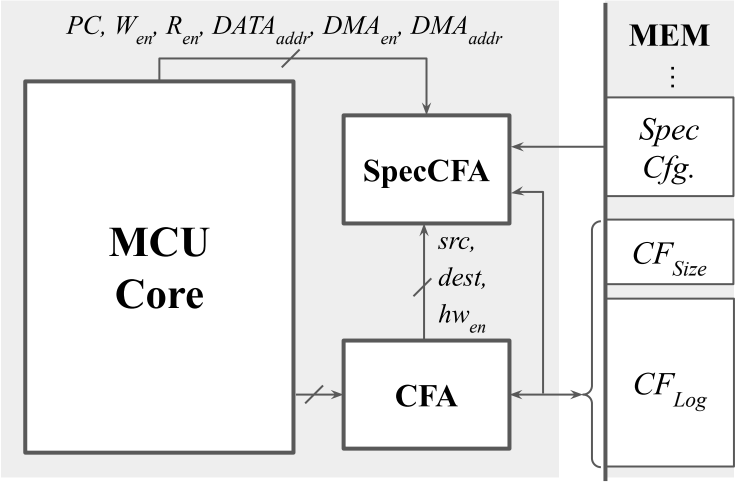
A, B, D, G

A, B, D, G

Now extend this to support multiple sub-paths of arbitrary lengths

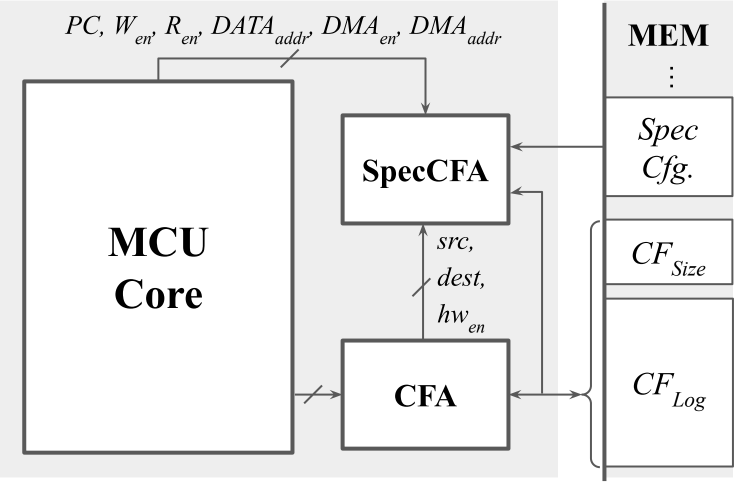
Extension for hardware-based and TEE-based arch.

Hardware-based

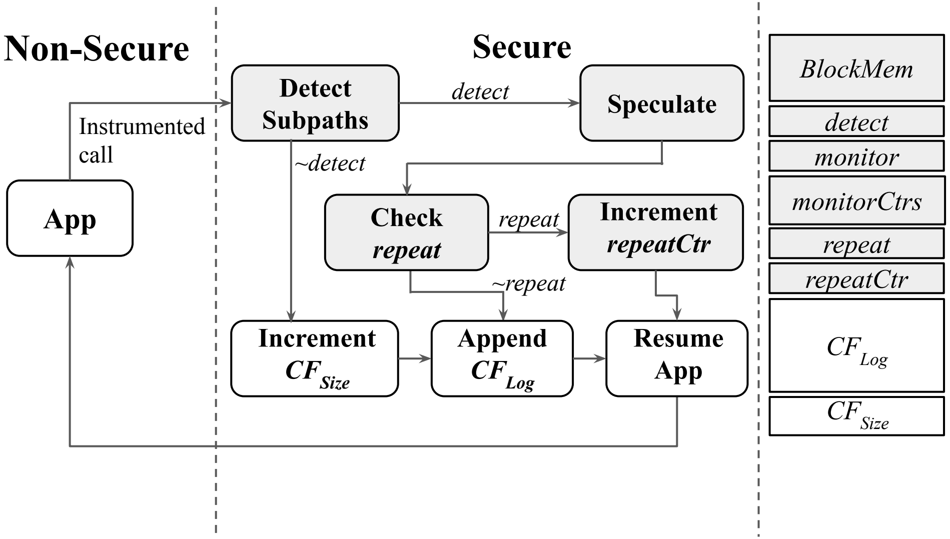


Extension for hardware-based and TEE-based arch.

Hardware-based



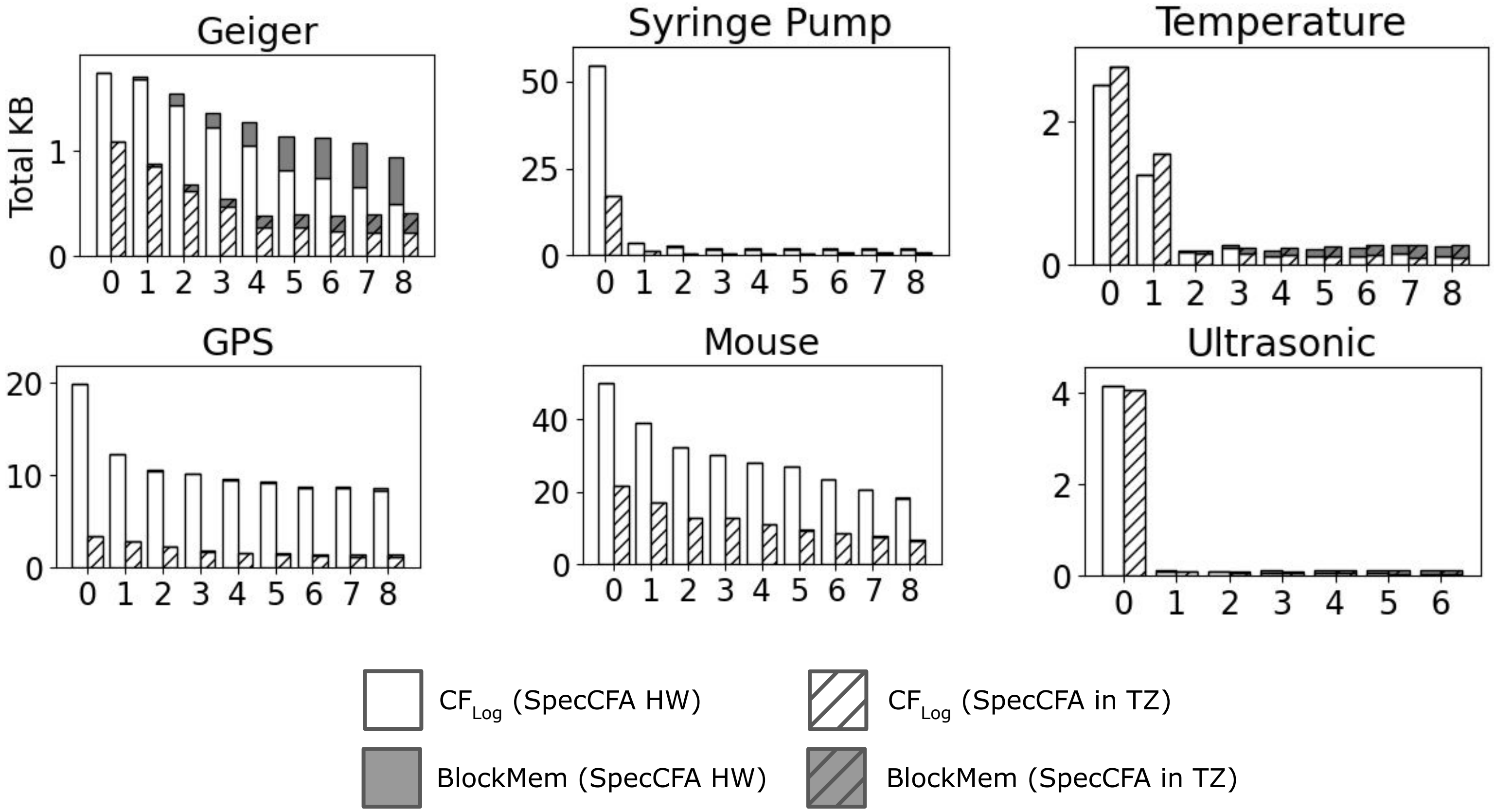
TEE-based



Evaluation

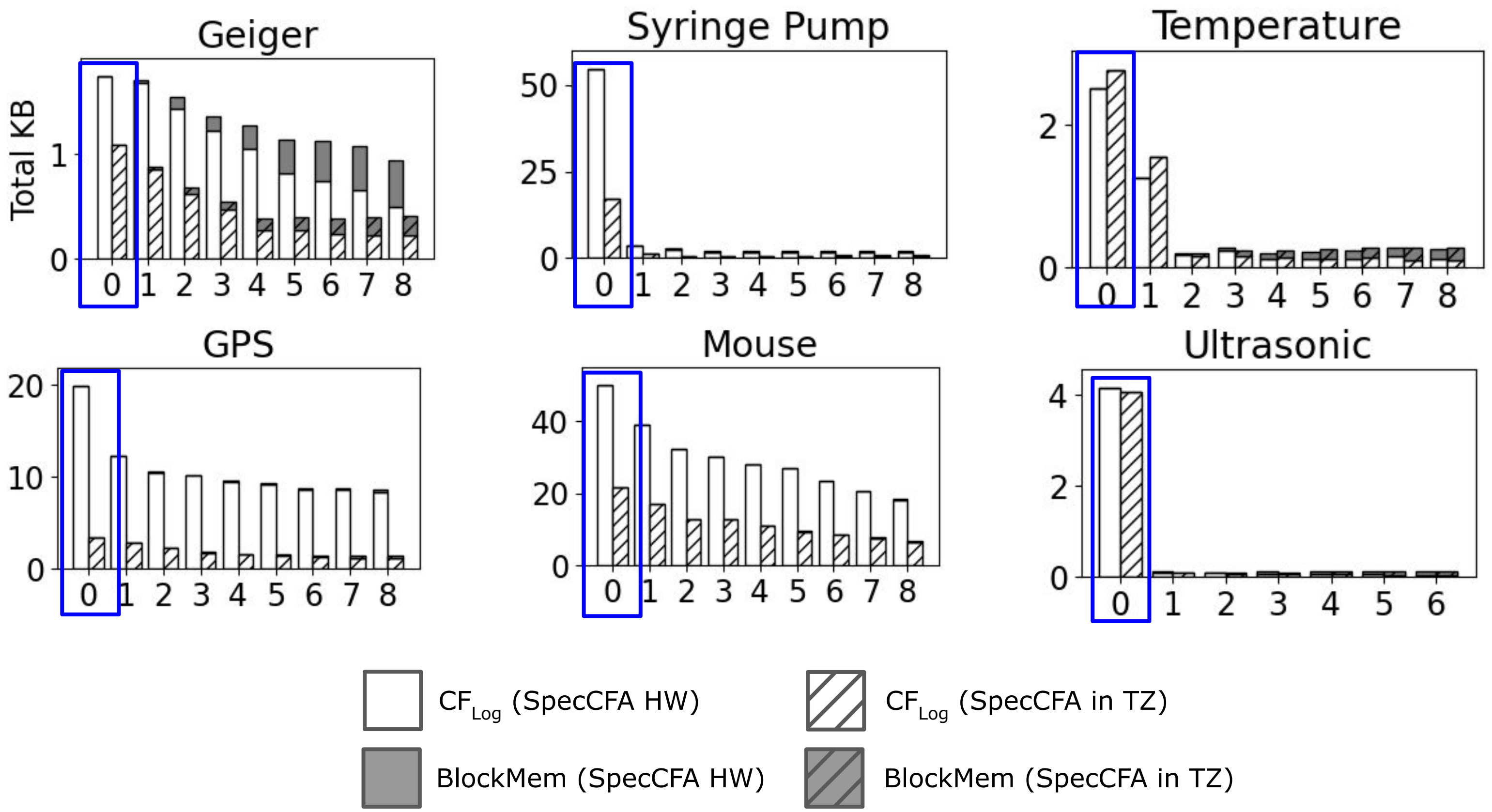
- Measure the storage & communication costs of CF_{Log}s from open-source MCU applications
  - Using existing HW-based and TEE-based CFA as a baselines
- Measure storage and communication costs when SpecCFA speculates on 1-8 application sub-paths
- Compare to evaluate the effect of SpecCFA

Reductions to CF_{Log} size



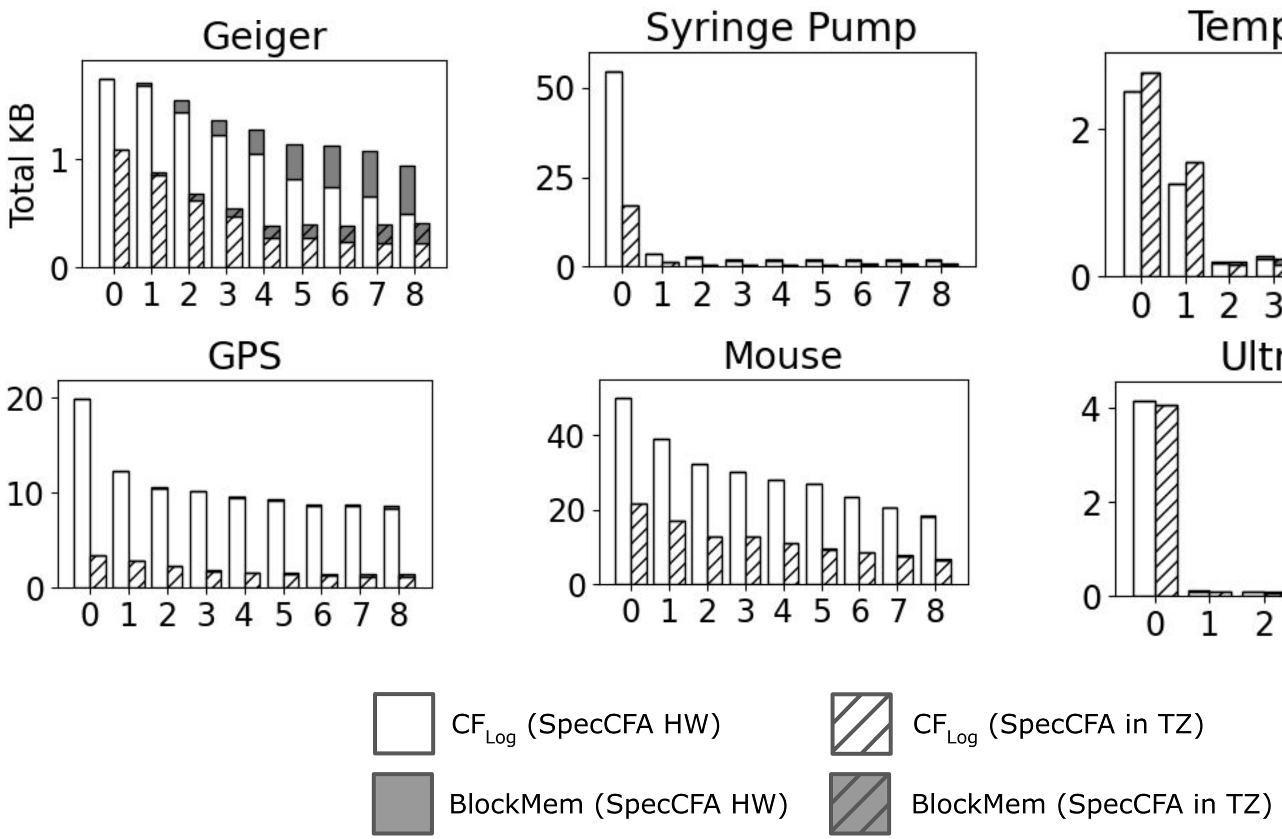
Reductions to CF_{Log} size

Zero sub-paths = Baseline

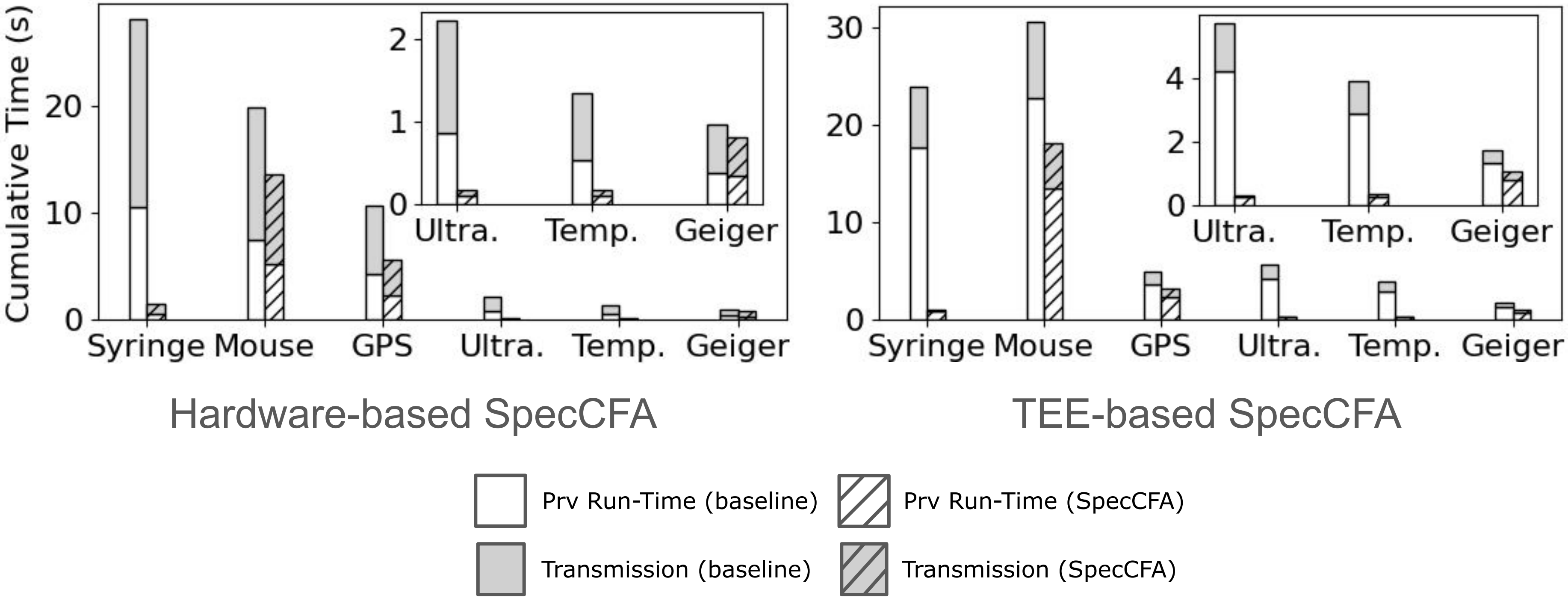


Reductions to CF$_{Log}$ size

Up to 97.9% reduction!!



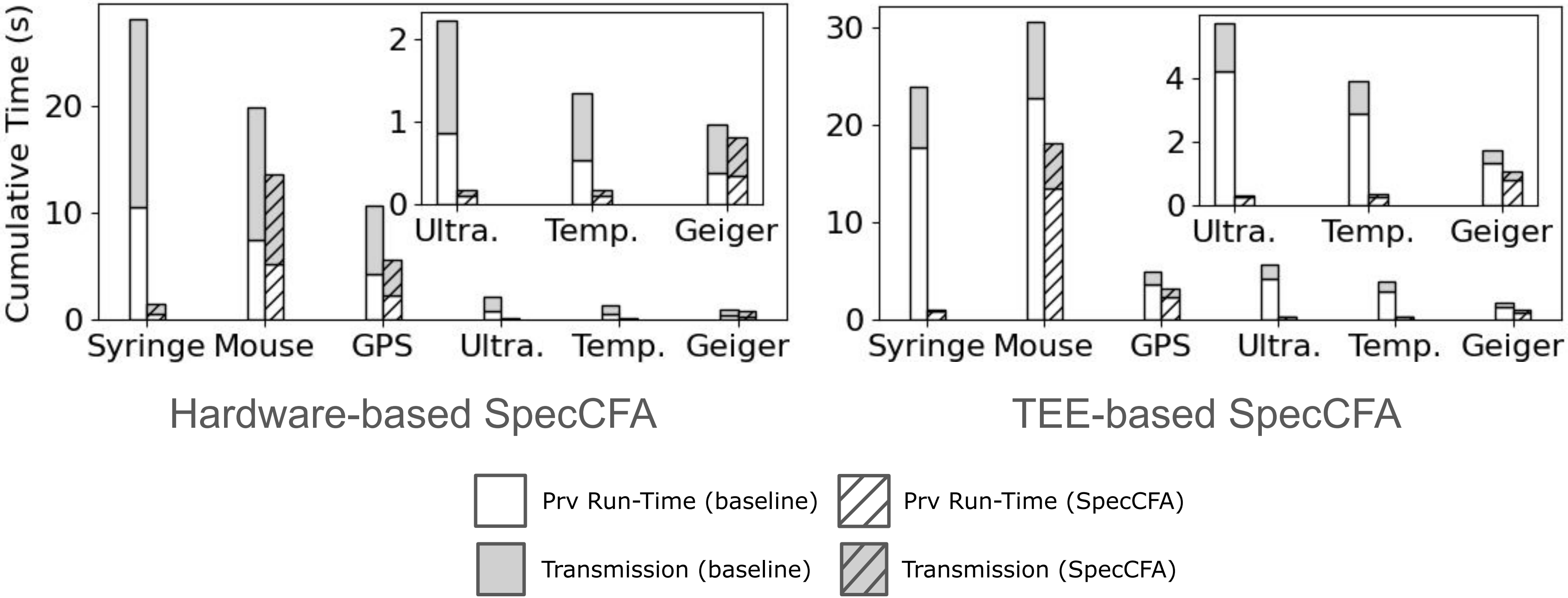
Reductions to communication overhead



Hardware-based SpecCFA

TEE-based SpecCFA

Reductions to communication overhead



Hardware-based SpecCFA

TEE-based SpecCFA

Up to 97.0% reduction!!

Thank you

Working Prototype:

Available on our CHAOS-Sec repository

https://github.com/RIT-CHAOS-SEC/SpecCFA

Paper:

Preprint is available on arxiv:

https://arxiv.org/abs/2409.18403