SoK: Integrity, Attestation, and Auditing of Program Execution

Mahmoud Ammar
Independent Researcher

Adam Caulfield*
University of Waterloo

Ivan De Oliveira Nunes*
University of Zurich

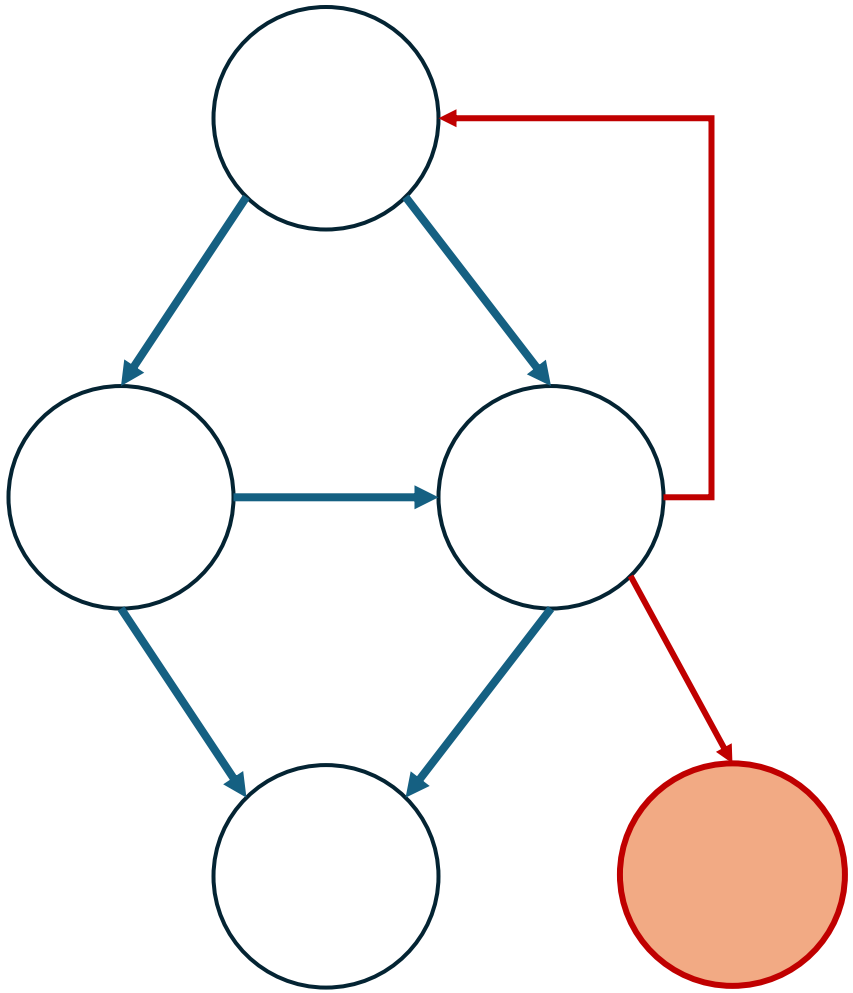
**Part of this work was performed while authors 2 and 3 were at Rochester Institute of Technology*

Motivation

Runtime Integrity vs. Runtime Attestation mechanisms

- Control Flow Integrity (CFI)
- Control Flow Attestation (CFA)
- Both are defenses for run-time attacks – do we need both?

Control Flow Graph



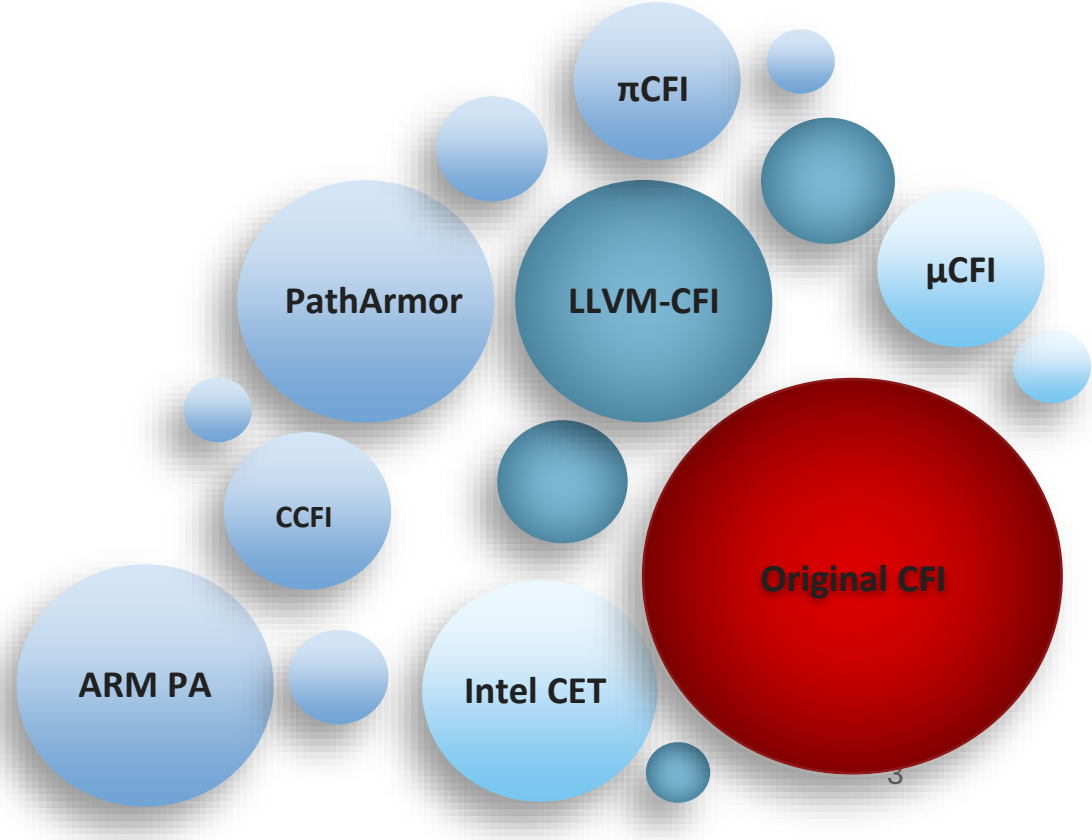
Motivation

Runtime Integrity vs. Runtime Attestation mechanisms

- Control Flow Integrity (CFI)
- Control Flow Attestation (CFA)
- Both are defenses for run-time attacks – do we need both?

CFI

- A mechanism on the executing device to detect
- Widely studied – available in commercial devices
  - e.g., Intel CET, ARM PA & BTI



Motivation

Runtime Integrity vs. Runtime Attestation mechanisms

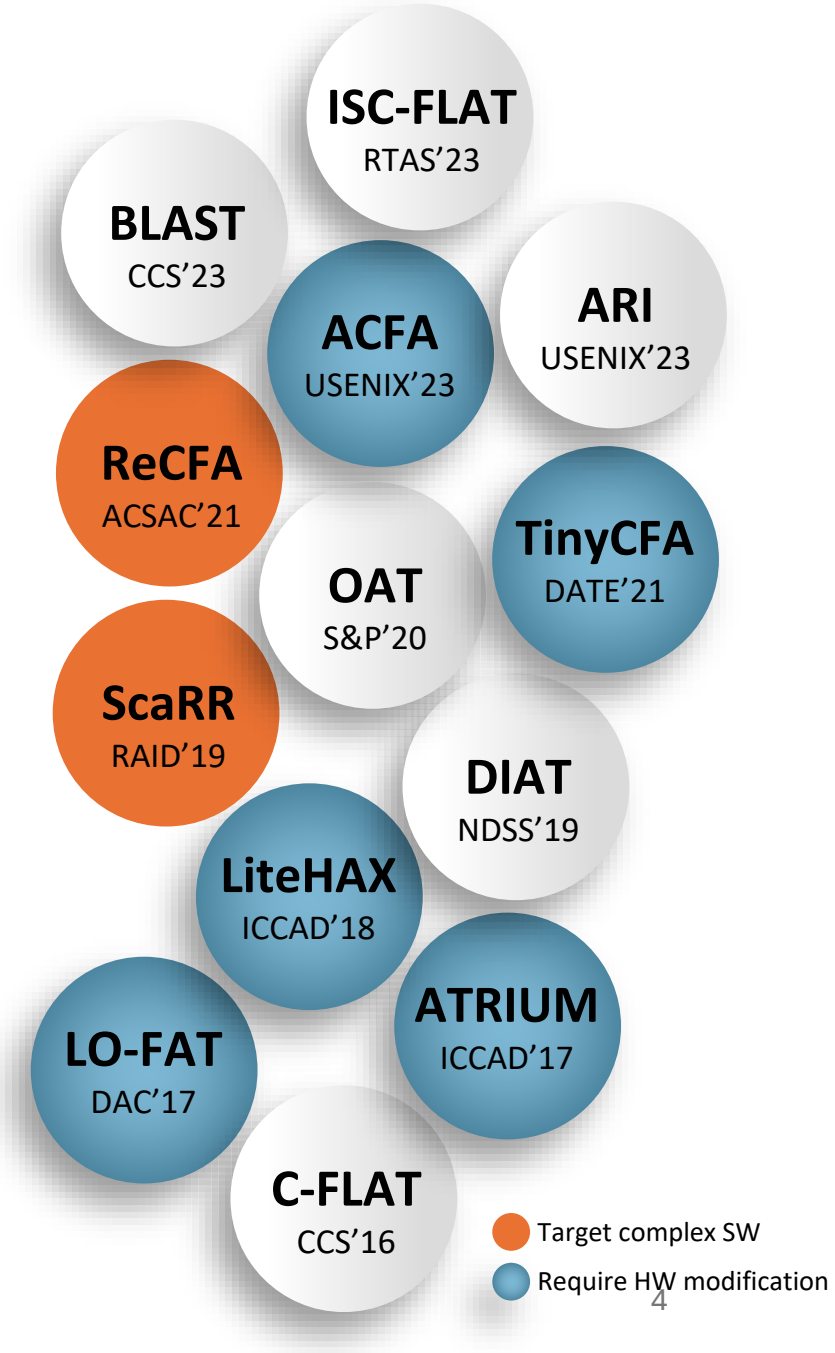
- Control Flow Integrity (CFI)
- Control Flow Attestation (CFA)
- Both are defenses for run-time attacks – do we need both?

CFI

- A mechanism on the executing device to detect
- Widely studied – available in commercial devices
  - e.g., Intel CET, ARM PA & BTI

CFA

- A mechanism on the executing device to produce evidence
- First proposal was less than 10 years ago [C-FLAT, CCS'16]



How do CFI and CFA fit into landscape of runtime defenses and with each other?

Questions to answer …

Q1. How do CFA and CFI goals differ?

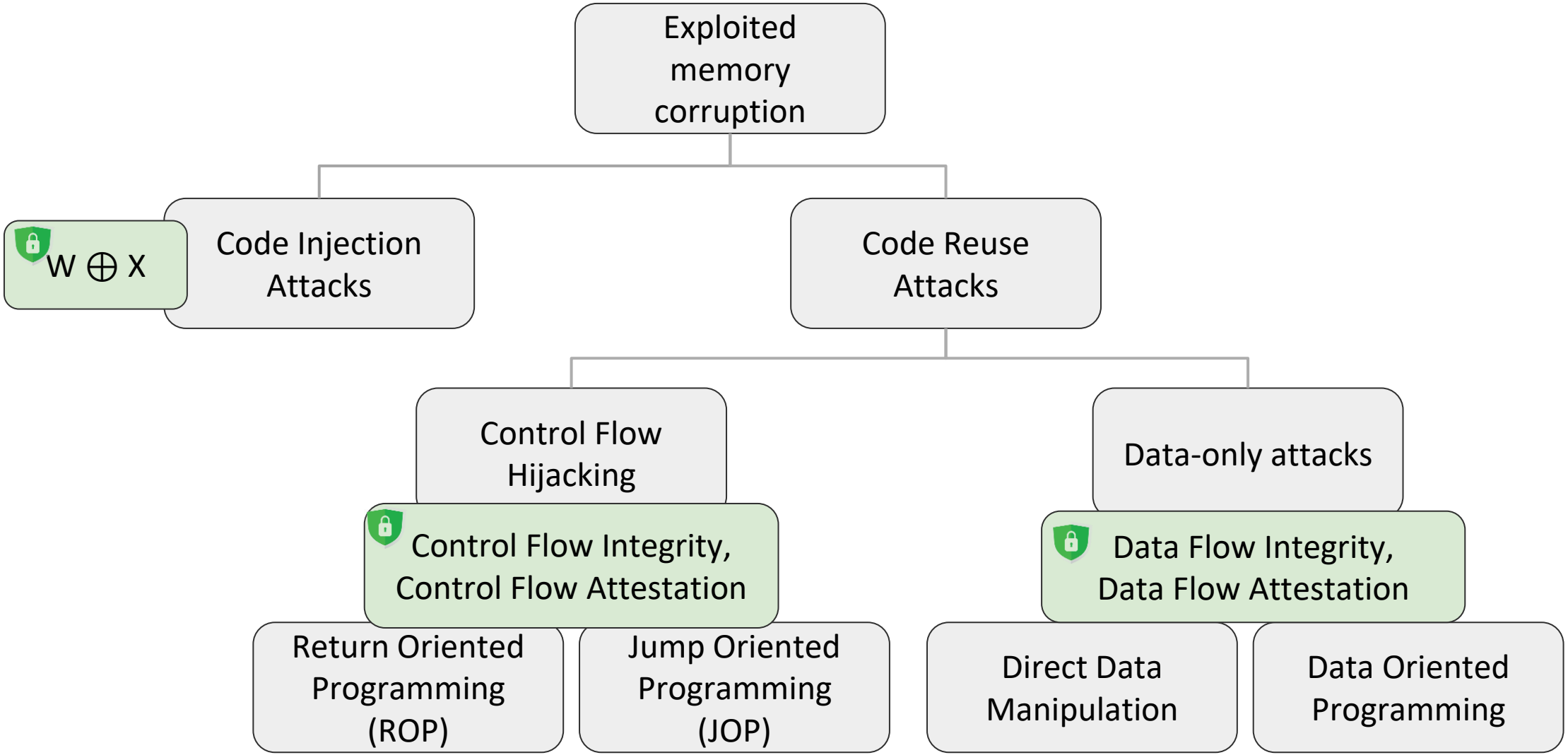
Q2. What are the assumptions, features, and design spaces of CFI vs CFA, as well as their similarities and differences?

Q3. What makes CFA different from attesting adherence to a CFI policy?

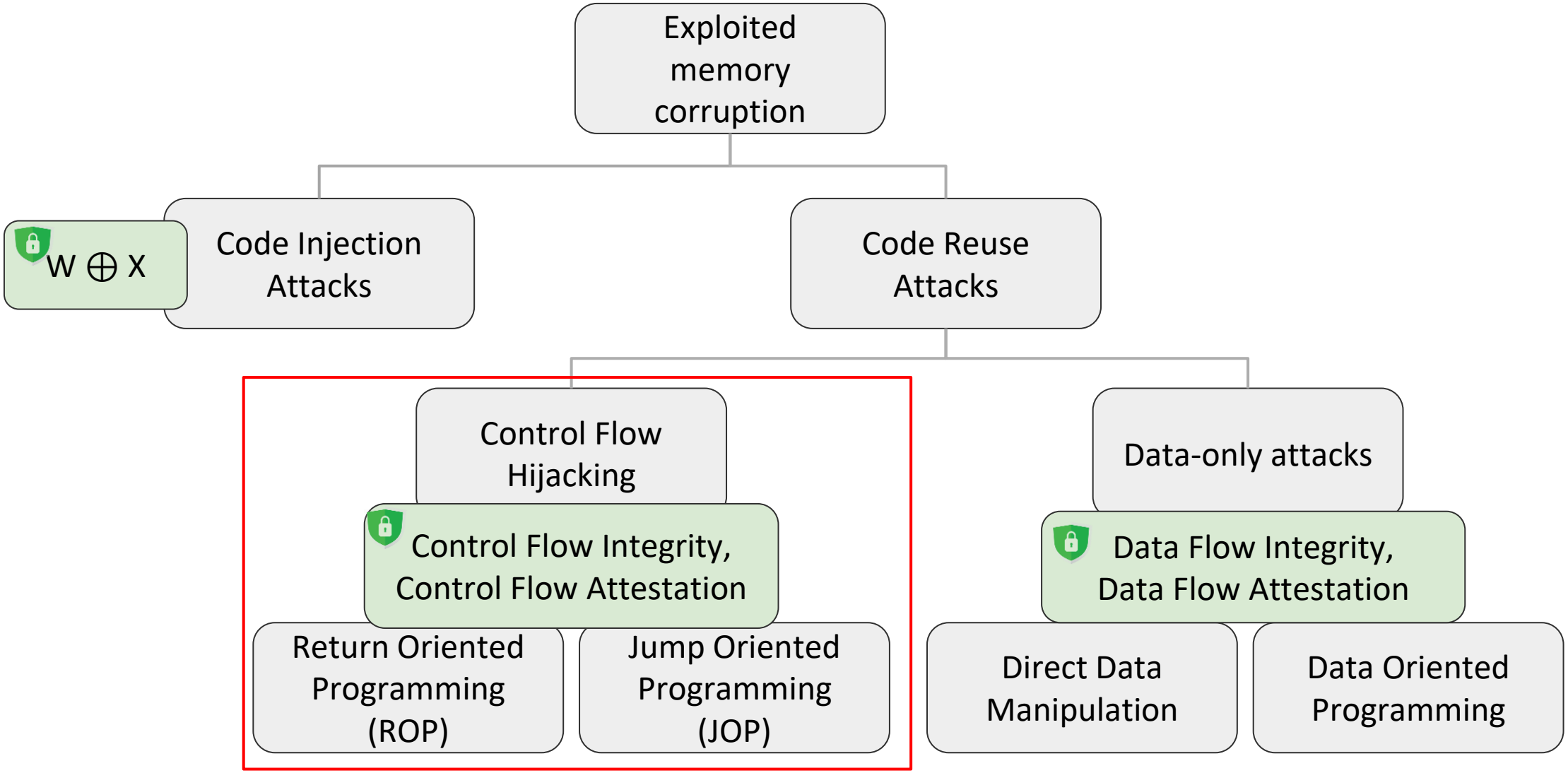
Could CFA uncover attacks that CFI would not (and vice-versa)?

Q4. Could CFI and CFA coexist on the same platform?

Attacks altering program execution



Attacks altering program execution



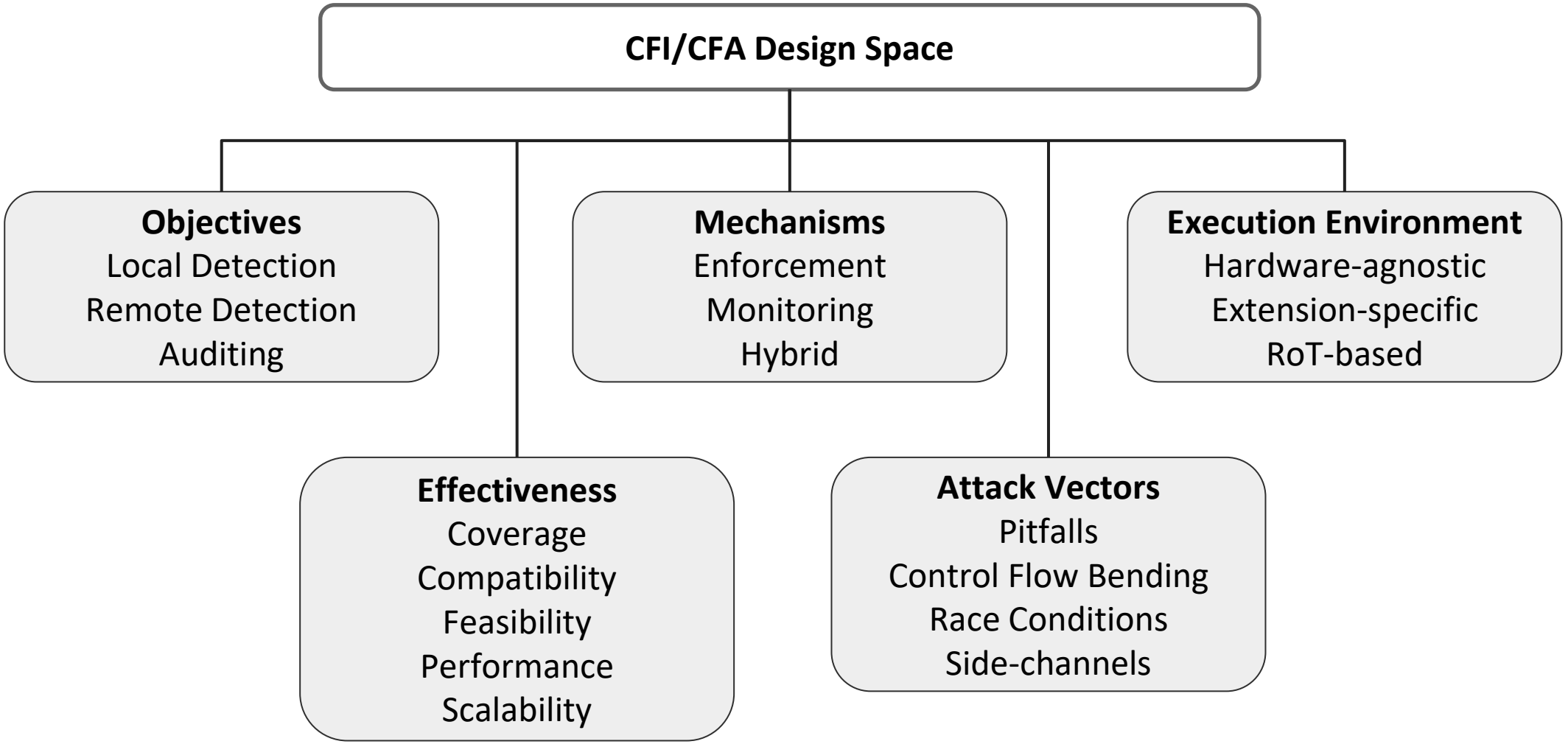
Systematized CFI and CFA techniques

Selection criteria:

- All available CFA literature (at the time)
- Categorize CFI techniques based on:
  - Papers published in big 4 security venues
  - Papers with more than 100 citations
  - Techniques adopted by mainstream compilers or hardware architectures

Year	Scheme	Device Type/Target				Mechanism		Sensitivity	System Support	Scope				Overheads			
		Embedded (bare-metal)	Embedded (OS)	High-end (User-space)	High-end (Kernel)	Type	Strategy			Data-Only	ROP	JOP	Evidence Expressiveness	Runtime	Code Size	Custom Hardware	Network
Control Flow Integrity (CFI) Approaches																	
2013	bin-CFI [88]	✗	✗	✓	✗	Enforcement	SWI	✗	OS/MMU	✗	○	○	-	●	●	✗	-
2013	CCFIR [89]	✗	✗	✓	✗	Enforcement	SWI+R/I	✗	OS/MMU	✗	○	○	-	●	●	✗	-
2014	**LLVM CFI** [18]	✗	✗	✓	✓	Enforcement	SWI	✗	OS/MMU	✗	✗	◐	-	●	●	✗	-
2014	KCoFI [90]	✗	✗	✗	✓	Enforcement	SWI	✗	MMU	✗	◐	◐	-	●	●	✗	-
2014	MCFI [91]	✗	✗	✓	✗	Enforcement	SWI	CS	OS/MMU	✗	◐	◐	-	●	●	✗	-
2014	RockJIT [137]	✗	✗	✓	✗	Enforcement	SWI	CS	OS/MMU	✗	◐	◐	-	●	●	✗	-
2015	CCFI [92]	✗	✗	✓	✗	Enforcement	SWI	✗	OS/MMU	✗	●	●	-	●	●	✗	-
2015	HAFIX [130]	✓	✗	✗	✗	Enforcement	ISA	Path	C-HW	✗	●	✗	-	●	●	●	-
2015	CFCI [93]	✗	✗	✓	✗	Enforcement	SWI	✗	OS/MMU	✗	○	○	-	●	●	✗	-
2015	O-CFI [94]	✗	✗	✓	✗	Enforcement	SWI+R/I	✗	OS/MMU+MPX	✗	◐	◐	-	●	●	✗	-
2015	πCFI [85]	✗	✗	✓	✗	Enforcement	SWI	✗	OS/MMU	✗	◐	◐	-	●	●	✗	-
2015	PathArmor [106]	✗	✗	✓	✗	Hybrid	SWI+ISA	Path	OS/MMU+LBR	✗	●	●	-	●	●	✗	-
2015	Lockdown [95]	✗	✗	✓	✗	Enforcement	SWI+R/I	✗	OS/MMU	✗	●	●	-	●	●	✗	-
2016	TypeArmor [96]	✗	✗	✓	✗	Enforcement	SWI	✗	OS/MMU	✗	✗	◐	-	●	●	✗	-
2016	FG-CFI [97]	✗	✗	✗	✓	Enforcement	SWI	✗	MMU	✗	◐	◐	-	●	●	✗	-
2016	HCFI [127]	✗	✗	✓	✗	Enforcement	ISA	✗	OS+C-HW	✗	●	●	-	●	●	●	-
2017	PittyPAT [107]	✗	✗	✓	✗	Hybrid	ISA+HRM	Path	OS/MMU+PT	✗	●	●	-	●	●	✗	-
2017	GRIFFIN [128]	✗	✗	✓	✗	Hybrid	ISA+HRM	✗	OS/MMU+PT+TSX	✗	●	●	-	●	●	✗	-
2017	CFI-CaRE [136]	✓	✗	✗	✗	Enforcement	SWI+R/I	✗	TZ	✗	●	◐	-	●	●	✗	-
2017	**Intel CET** [21]	✗	✗	✓	✓	Enforcement	ISA	✗	OS/MMU+CET	✗	●	○	-	●	●	✗	-
2018	μCFI [84]	✗	✗	✓	✗	Hybrid	SWI+ISA	✗	OS/MMU+PT	✗	●	●	-	●	●	✗	-
2018	SCFP [108]	✓	✗	✗	✗	Enforcement	SWI+C-HW	Path	C-HW	✗	●	◐	-	●	●	●	-
2018	**ARM BTI** [98]	✓	✓	✓	✓	Enforcement	ISA	✗	BTI	✗	✗	○	-	●	●	✗	-
2018	**PAC-RET** [99]	✓	✓	✓	✓	Enforcement	ISA	✗	PA	✗	●	✗	-	●	●	✗	-
2019	OS-CFI [109]	✗	✗	✓	✗	Enforcement	SWI+R/I	Path	OS/MMU+MPX+TSX	✗	✗	●	-	●	●	✗	-
2019	CFI-LB [110]	✗	✗	✓	✗	Enforcement	SWI+R/I	CS	OS/MMU+TSX	✗	✗	●	-	●	●	✗	-
2019	PARTS [25]	✗	✗	✓	✗	Enforcement	SWI+ISA	✗	OS/MMU+PA	+	●	●	-	●	●	✗	-
2020	μRAI [111]	✓	✗	✗	✗	Enforcement	SWI+R/I	Path	MPU	✗	●	✗	-	●	●	✗	-
2020	Silhouette [122]	✓	✗	✗	✗	Enforcement	SWI+R/I	✗	MPU	✗	●	✗	-	●	●	✗	-
2021	VIP [112]	✗	✗	✓	✗	Enforcement	SWI+R/I	Path	OS/MMU+MPK	+	✗	●	-	●	●	✗	-
2021	PACStack [27]	✗	✗	✓	✗	Enforcement	SWI+ISA	✗	OS/MMU+PA	✗	●	✗	-	●	●	✗	-
2022	TyPro [129]	✗	✗	✓	✗	Enforcement	SWI	✗	OS/MMU	✗	✗	●	-	●	●	✗	-
2022	PAL [26]	✗	✗	✗	✓	Enforcement	SWI+ISA	✗	PA+MMU	✗	●	●	-	●	●	✗	-
2022	PACTight [100]	✗	✗	✓	✗	Enforcement	SWI+ISA	✗	OS/MMU+PA	✗	●	●	-	●	●	✗	-
2023	FineIBT [22]	✗	✗	✓	✓	Enforcement	SWI+ISA	✗	CET+MMU	✗	✗	◐	-	●	●	✗	-
2023	SHERLOC [114]	✓	✗	✗	✗	Hybrid	ISA+HRM	Path	TZ+MTB+DWT	✗	●	◐	-	●	●	✗	-
2023	TypeSqueezer [138]	✗	✗	✓	✗	Enforcement	SWI	Path	OS/MMU	✗	✗	●	-	●	●	✗	-
2024	HEK-CFI [139]	✗	✗	✗	✓	Enforcement	ISA	✗	CET+MMU	✗	●	◐	-	●	●	✗	-
Control Flow Attestation (CFA) Approaches																	
2016	C-FLAT [33]	✓	✗	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	TZ	□	●	●	△	●	●	✗	☆
2017	LO-FAT [35]	✓	✗	✗	✗	Monitoring	C-HW	$\mathcal{V}rf$-based	C-HW	□	●	●	△	✗	✗	●	☆
2017	ATRIUM [36]	✓	✗	✗	✗	Monitoring	C-HW	$\mathcal{V}rf$-based	C-HW	□	●	●	△	✗	✗	●	☆
2018	LiteHAX [37]	✓	✗	✗	✗	Monitoring	C-HW	$\mathcal{V}rf$-based	C-HW	⊞	●	●	▲	✗	✗	●	★
2019	DIAT [140]	✓	✓	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	TZ	□	●	●	△	●	●	✗	☆
2019	ScaRR [119]	✗	✗	✓	✗	Monitoring	SWI	$\mathcal{V}rf$-based	OS/MMU	□	●	●	▲	●	●	✗	★
2019	RIM [124]	✓	✗	✗	✗	Monitoring	C-HW	Path	C-HW	⊞	●	●	△	✗	✗	?	☆
2020	OAT [38]	✓	✓	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	TZ	⊞	●	●	◭	●	●	✗	☆
2020	LAHEL [141]	✓	✓	✗	✗	Monitoring	C-HW	$\mathcal{V}rf$-based	C-HW	□	◐	◐	◭	●	●	debug HW	☆
2020	LAPE [142]	✓	✗	✗	✗	Monitoring	SWI+R/I	$\mathcal{V}rf$-based	MPU	□	◐	◐	△	●	●	✗	☆
2021	Tiny-CFA [34]	✓	✗	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	C-HW	□	●	●	▲	●	●	✗	☆
2021	DIALED [76]	✓	✗	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	C-HW	⊞	●	●	▲	●	●	✗	☆
2021	ReCFA [125]	✗	✗	✓	✗	Monitoring	SWI+R/I	$\mathcal{V}rf$-based	OS+MPK	□	●	●	▲	●	●	✗	☆
2022	GuaranTEE [126]	✗	✗	✓	✗	Monitoring	SWI	$\mathcal{V}rf$-based	Intel SGX	□	●	●	△	●	●	✗	☆
2023	ACFA [41]	✓	✗	✗	✗	Monitoring	C-HW	$\mathcal{V}rf$-based	C-HW	□	●	●	▲	✗	✗	●	★
2023	ARI [40]	✓	✓	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	TZ	□	◐	◐	◭	●	●	✗	☆
2023	BLAST [39]	✓	✓	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	TZ	□	●	●	▲	●	●	✗	☆
2023	ISC-FLAT [43]	✓	✗	✗	✗	Hybrid	SWI	$\mathcal{V}rf$-based	TZ	□	●	●	▲	●	●	✗	☆
2024	TRACES [42]	✓	✗	✗	✗	Monitoring	SWI	$\mathcal{V}rf$-based	TZ	□	●	●	▲	●	●	✗	★
2024	CFA+ [120]	✓	✓	✓	✗	Hybrid	SWI+ISA	$\mathcal{V}rf$-based	OS/MMU+TPM	□	●	●	▲	●	●	✗	☆

Explored the design space of existing CFI and CFA techniques



Questions to answer …

Q1. How do CFA and CFI goals differ?

Q3. What makes CFA different from remotely attesting adherence to a CFI policy? Could CFA uncover attacks that CFI would not (and vice-versa)?

CFI focuses on **local** detection of control-flow violations.

CFA provides **remote evidence** of execution behavior regardless of underlying policy enforcement.

Questions to answer …

CFI and CFA schemes **share many commonalities** in their strategies.

But, they also have **distinct system requirements**- including threat model, execution environment, and underlying mechanisms.

Q2. What are the assumptions, features, and design spaces of CFI vs CFA, as well as their similarities and differences?

Q4. Could CFI and CFA coexist on the same platform?

Questions to answer …

Q1. How do CFA and CFI goals differ?

Q3. What makes CFA different from remotely attesting adherence to a CFI policy? Could CFA uncover attacks that CFI would not (and vice-versa)?

CFI is clearly the best choice for **local** detection of runtime attacks.

CFA enables **remote (and offline)** execution path analysis, giving remote visibility to complex path deviations that would be oblivious to most CFI.

CFA makes **logic** control path bugs observable.

CFA facilitates **auditing** and root cause analysis if the evidence is reliably delivered to the verifier.

Questions to answer …

Given the trade-offs between CFI and CFA, a **hybrid** approach could offer both **local responses** to simpler runtime attacks and **remote visibility** to complex attacks.

On the other hand, **overheads** of both approaches on the same platform could challenge practical adoption.

Q2. What are the assumptions, features, and design spaces of CFI vs CFA, as well as their similarities and differences?

Q4. Could CFI and CFA coexist on the same platform?

Takeaways

CFI focuses on **local** detection of control-flow violations.

CFA provides **remote evidence** of execution behavior regardless of underlying policy enforcement.

CFI and CFA schemes **share many commonalities** in their strategies.

But, they have **distinct system requirements**

CFI is clearly the best choice for **local** detection of runtime attacks.

CFA enables **remote** execution path analysis: potentially revealing logical bugs, complex path deviations, exploit root causes.

A **hybrid CFI-CFA** approach could offer **local responses** to simple attacks and **remote visibility** to complex ones.

On the other hand, **overheads** of both approaches on the same platform could challenge practical adoption.

For more information, see our poster!

← To read the full paper.

Thank you!