Bespoke Security for Resource Constrained Cyber-Physical Systems

Miguel A. Arroyo

Ph.D. Defense Oct 30th, 2020

The Birth of Cyber-Physical Systems

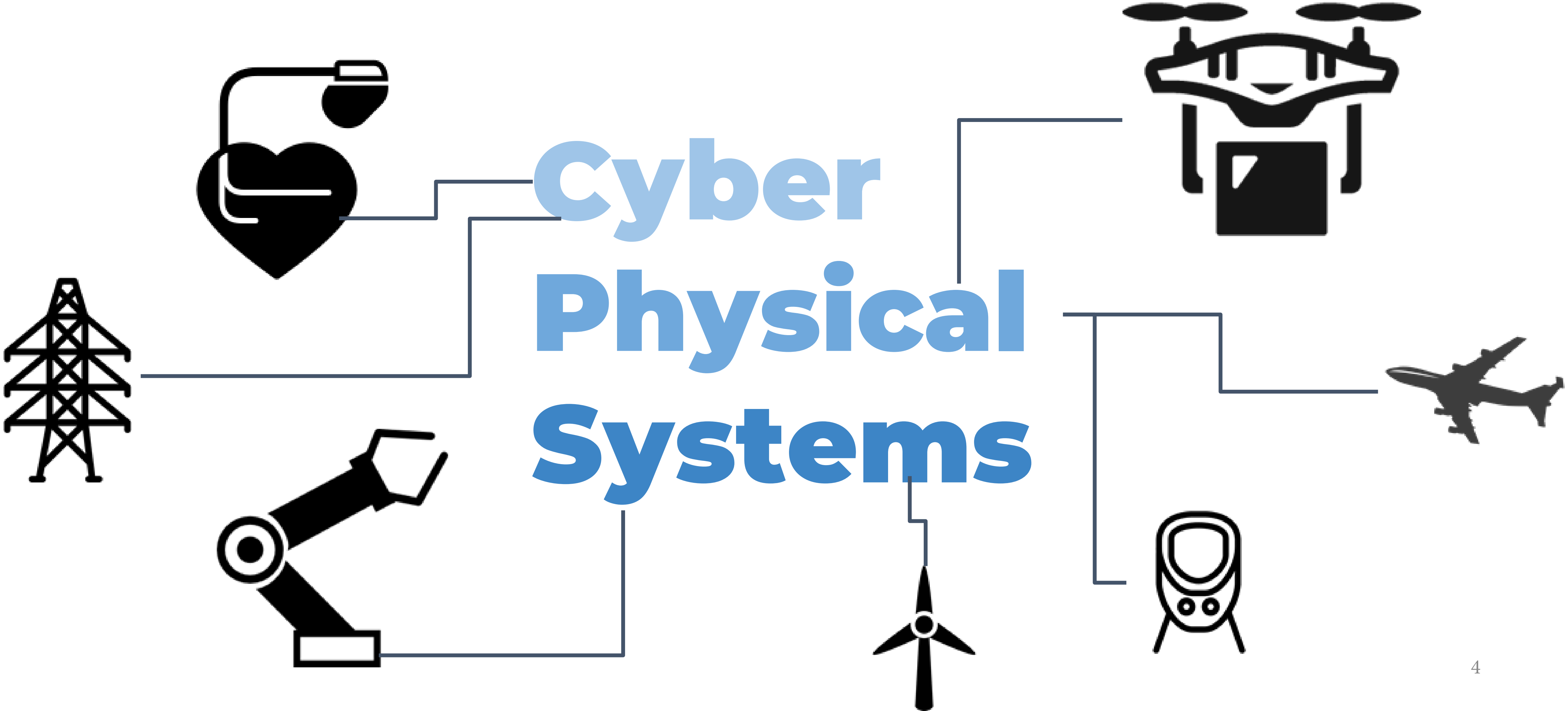
1800

The Birth of Cyber-Physical Systems

1800

2020+

CPSs are everywhere.

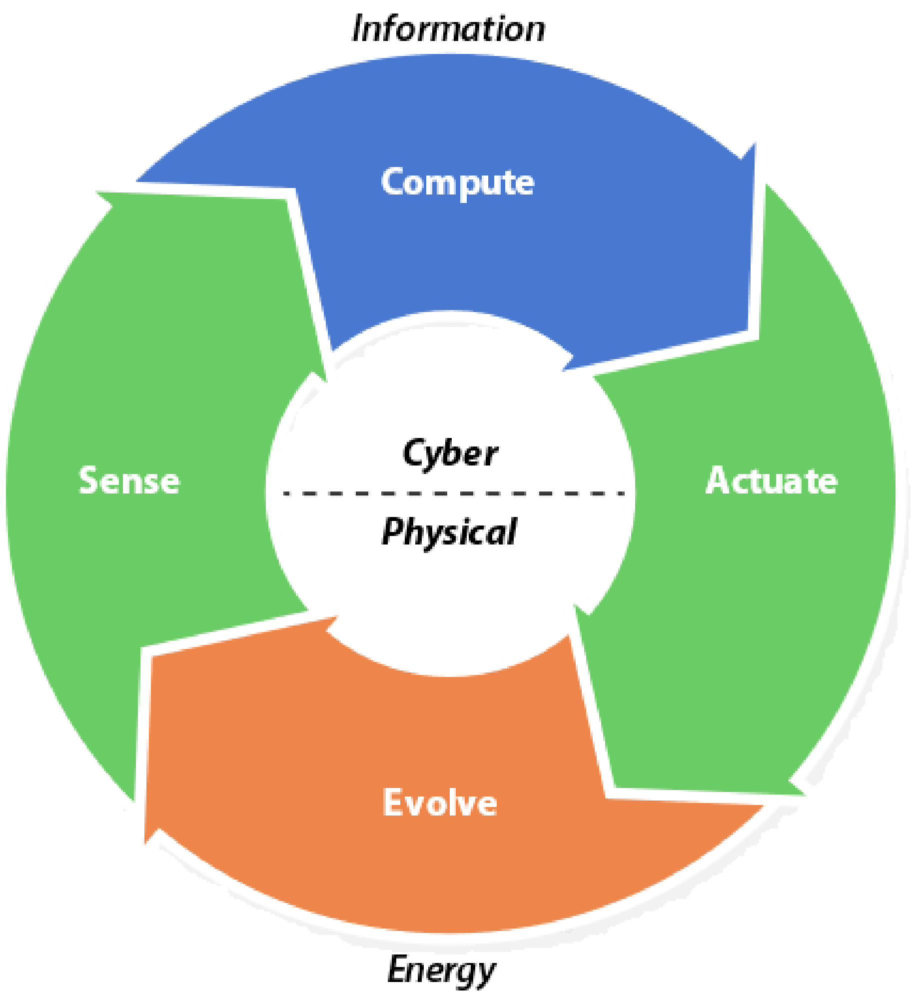


What are Cyber-Physical Systems?

A composition of physical processes, sensors, actuators, and computational units.

What are Cyber-Physical Systems?

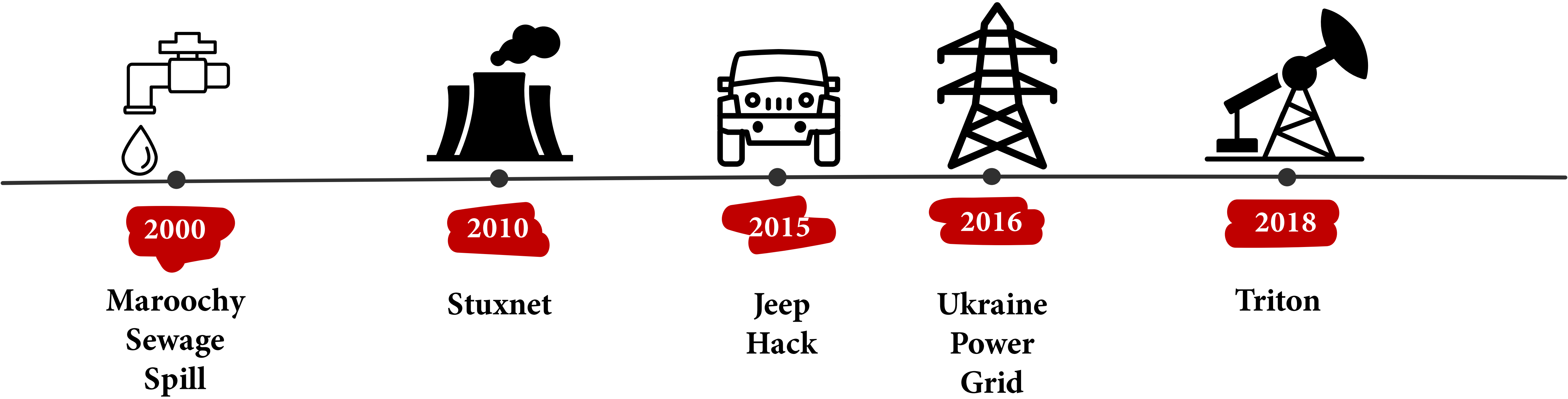
A CPS operates in a *feedback loop* between the cyber and physical domains.



CPS security is important.

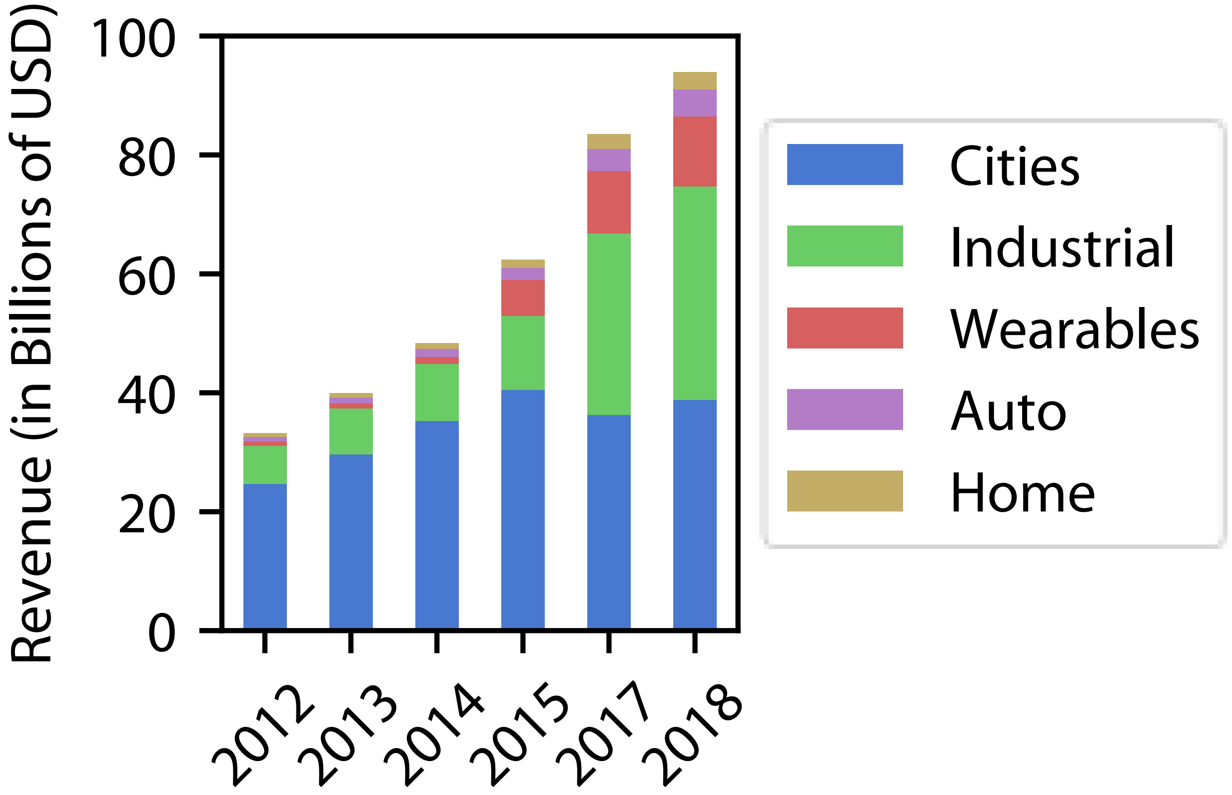
Why is CPS security important?

Physical interactions can damage the environment and harm people.



Why is CPS security important?

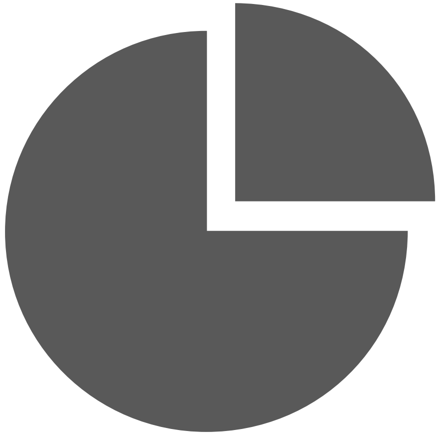
CPSs are becoming increasingly pervasive across multiple sectors.



Source: *IC Insights, The McClean Report 2018*

Why is CPS security important?

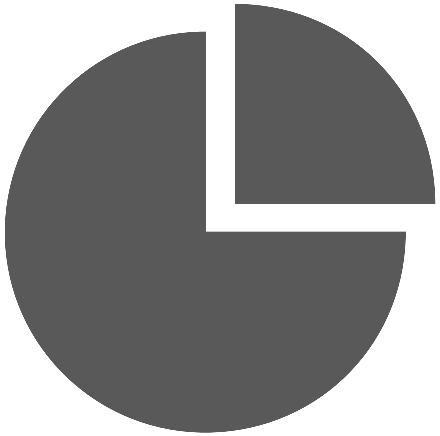
Physical systems have become increasingly dependent on software.



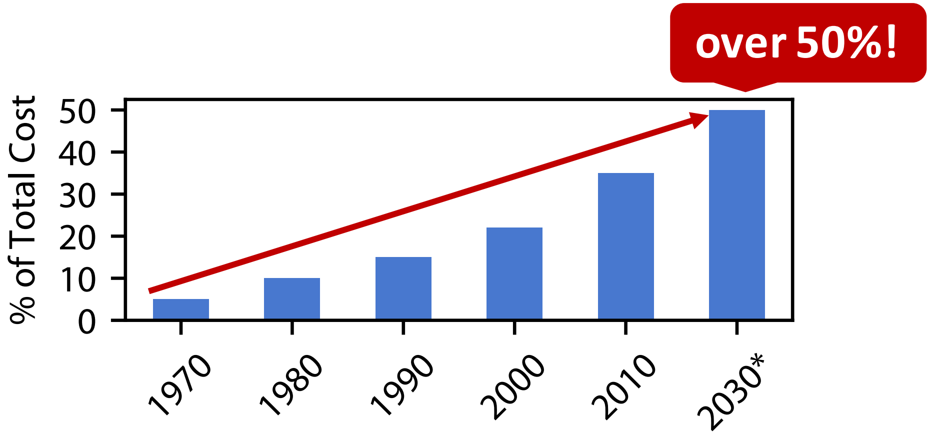
Software makes up a large portion of a CPS.

Why is CPS security important?

Physical systems have become increasingly dependent on software.



Software makes up a large portion of a CPS.

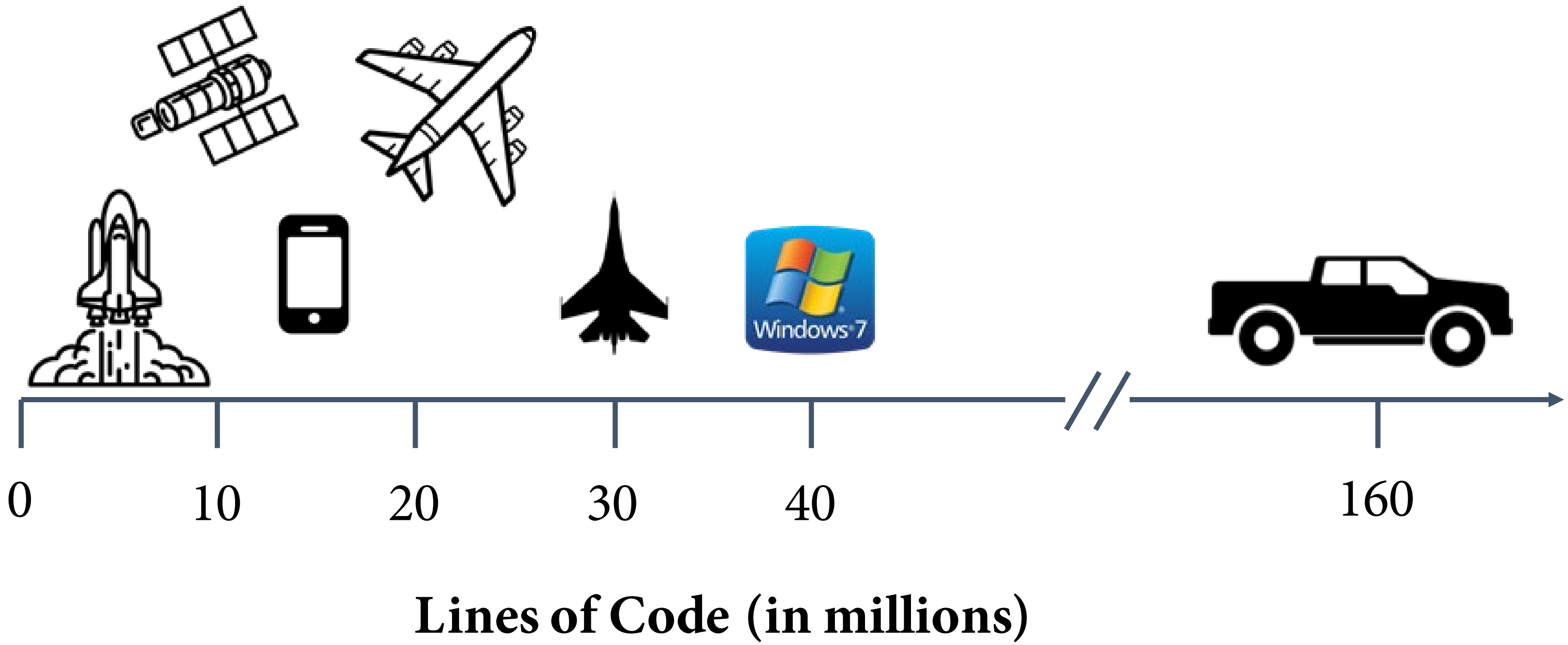


Automotive electronics cost as a share of total car cost.

Source: *Deloitte, Semiconductors – The Next Wave 2019*

Why is CPS security important?

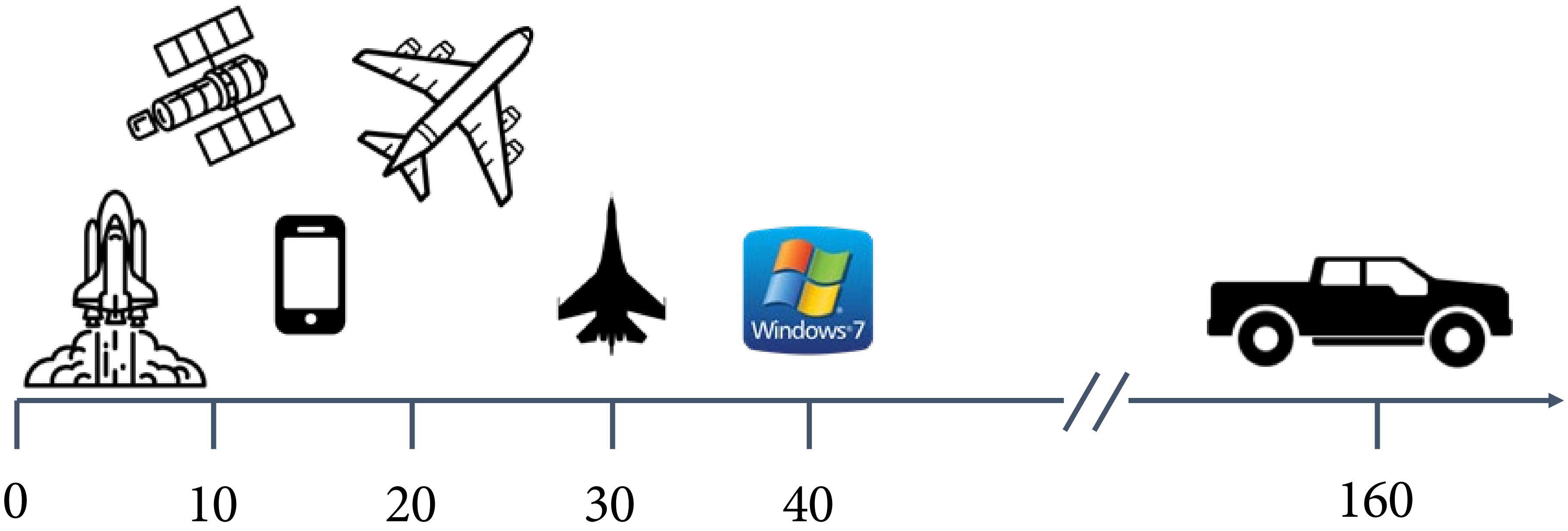
CPS software has become increasingly complex.



Source: *http://bit.ly/KIB_linescode*

Why is CPS security important?

CPS software has become increasingly complex.



~~Lines of Code (in millions)~~

Number of Bugs

Source: *http://bit.ly/KIB_linescode*

Thesis Statement

Security can be efficiently integrated by leveraging fundamental **physical properties**, & tailoring and extending **age-old abstractions** in computing.

My contributions to CPS security

My contributions to CPS security

A two-pronged approach against software threats

Leveraging Physical Properties

- May help reduce performance overheads.
- Makes retrofitting into existing systems more practical.

Revisiting Computing Abstractions

- Allows for design space exploration tailored to CPSs.
- Can be applicable in the broader general-purpose domain.

My contributions to CPS security

An overview of publications

1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

CPS security is challenging.

What makes CPS security challenging?

Resource constraints & strict requirements leave little room for security.

What makes CPS security challenging?

Resource constraints & strict requirements leave little room for security.

Heterogeneous
A wide variety of processors, sensors, and actuators leads to patchy software support.

What makes CPS security challenging?

Resource constraints & strict requirements leave little room for security.

Heterogeneous
A wide variety of processors, sensors, and actuators leads to patchy software support.

Real Time
Actions must be taken within a maximum specified deadline leaving little extra time for other tasks.

What makes CPS security challenging?

Resource constraints & strict requirements leave little room for security.

Heterogeneous
A wide variety of processors, sensors, and actuators leads to patchy software support.

Real Time
Actions must be taken within a maximum specified deadline leaving little extra time for other tasks.

Limited Resources
Systems have smaller memory, limited data-processing capabilities, and stripped-down functionality.

What makes CPS security challenging?

Resource constraints & strict requirements leave little room for security.

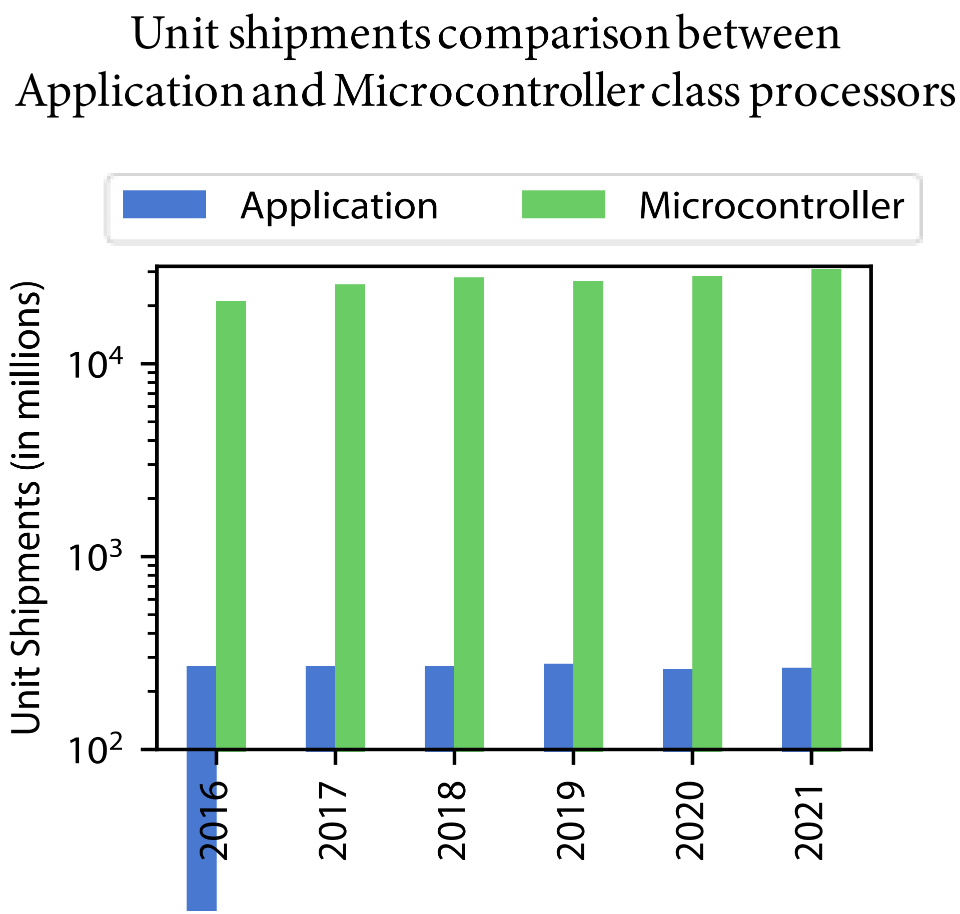
Heterogeneous
A wide variety of processors, sensors, and actuators leads to patchy software support.

Real Time
Actions must be taken within a maximum specified deadline leaving little extra time for other tasks.

Limited Resources
Systems have smaller memory, limited data-processing capabilities, and stripped-down functionality.

What makes CPS security challenging?

CPSs predominantly rely on microcontroller class processors.

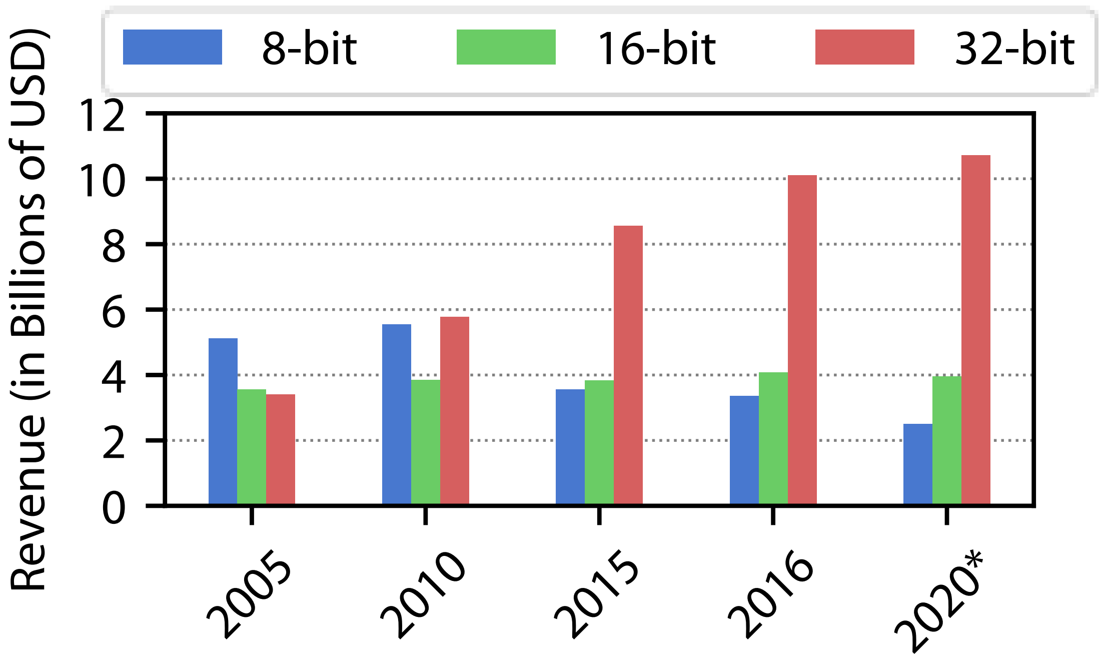


Source: *Statista [67-69]*

What makes CPS security challenging?

Microcontroller class processors used by CPSs have limited performance and functionality.

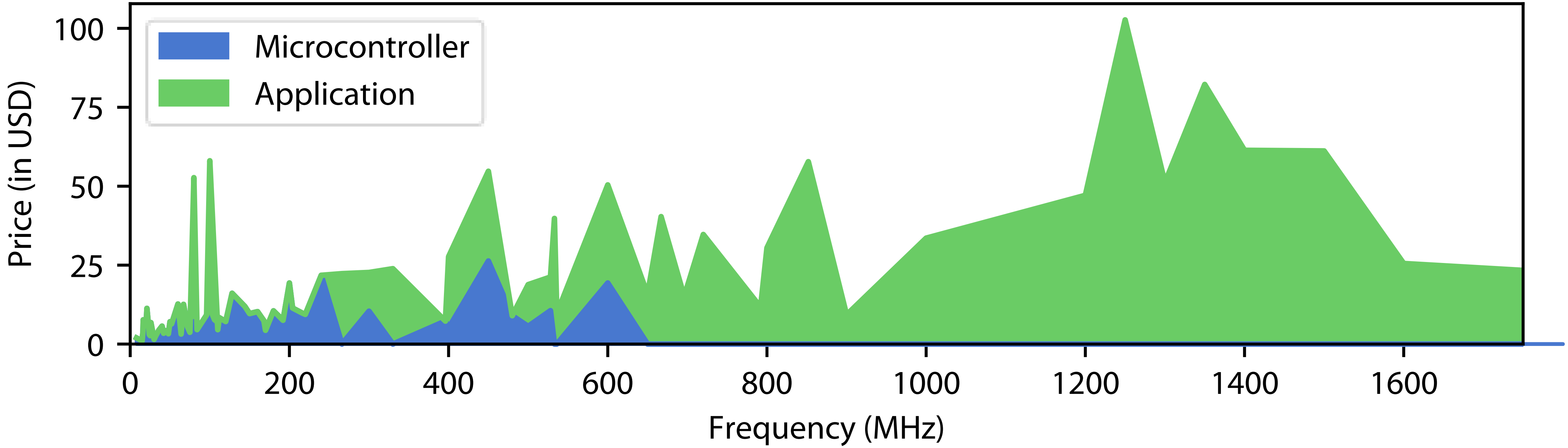
Microcontroller revenue by processor architecture



Source: *Statista – MCU Market Revenues from 2005 to 2020*

What makes CPS security challenging?

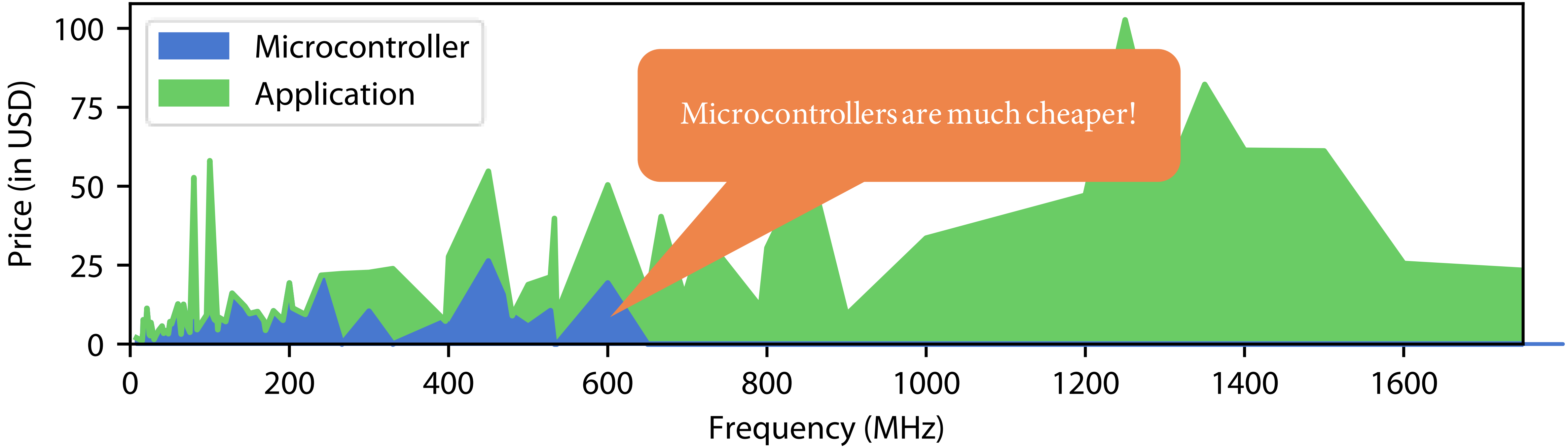
The cost sensitivity of CPS is an important factor in selection of microcontroller class processors.



Source: *Digi-Key*

What makes CPS security challenging?

The cost sensitivity of CPS is an important factor in selection of microcontroller class processors.

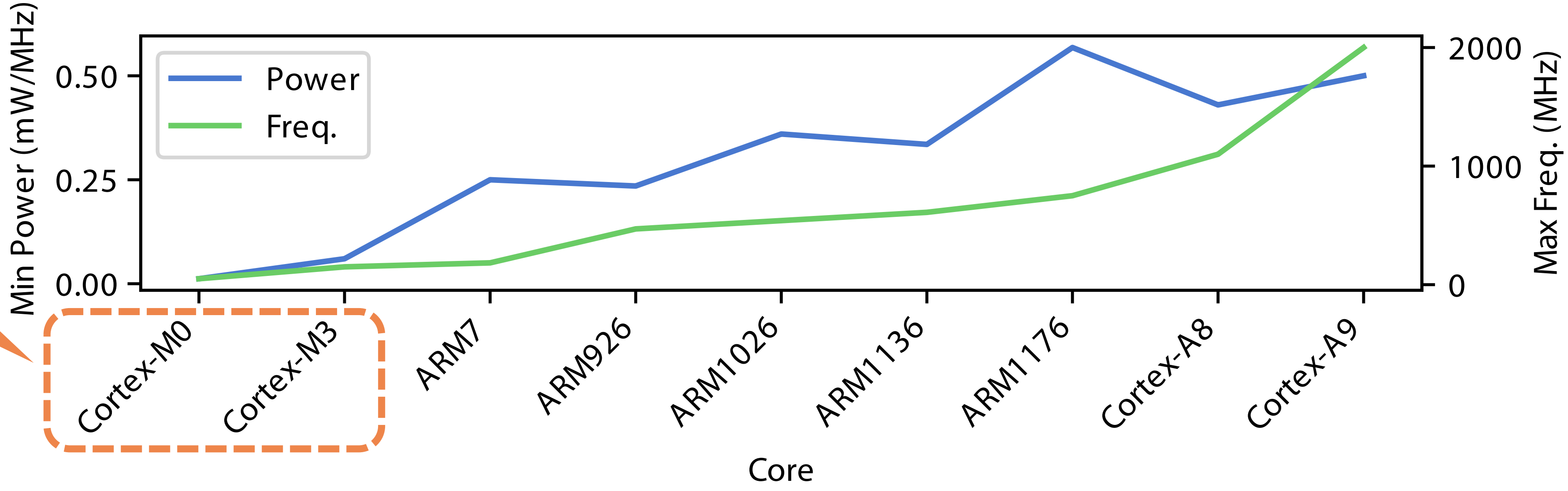


Source: *Digi-Key*

What makes CPS security challenging?

The energy sensitivity of CPS is an important factor in selection of microcontroller class processors.

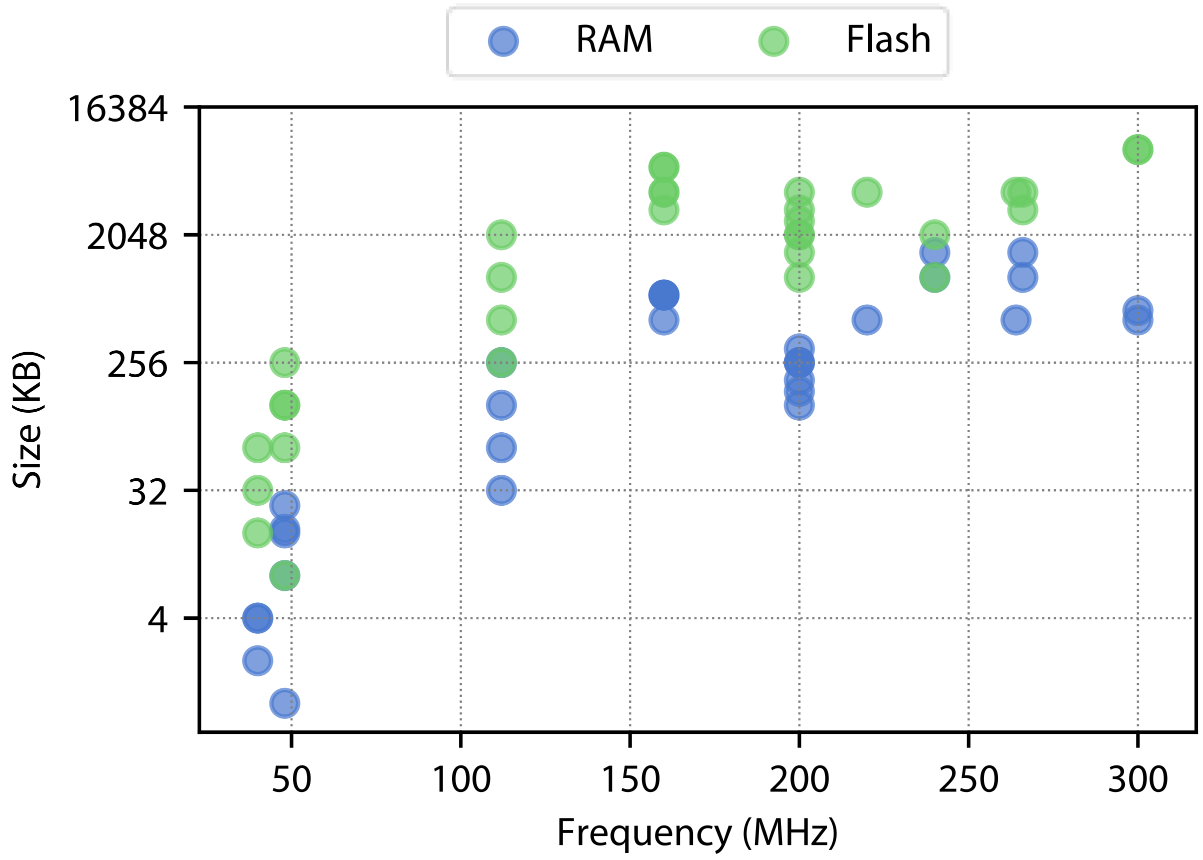
Microcontrollers sip power compared to even low-end application processors.



Source: *Bungo – The ARM Architecture 2015*

What makes CPS security challenging?

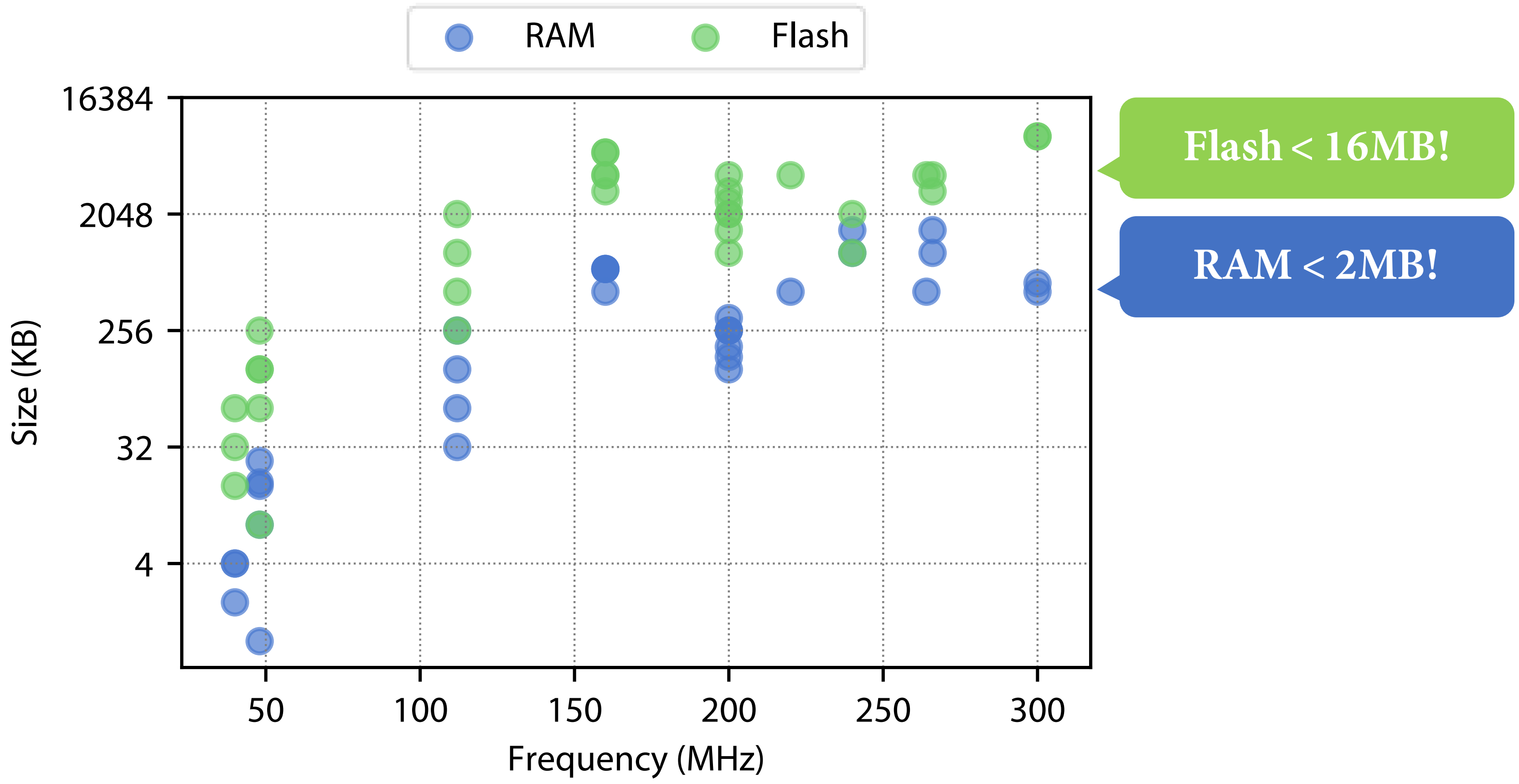
Memory is a scarce resource across the spectrum of microcontrollers.



Source: *NXP*

What makes CPS security challenging?

Memory is a scarce resource across the spectrum of microcontrollers.



Source: *NXP*

CPSs need low overhead deployable security.

My contributions to CPS security

An overview of publications

1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

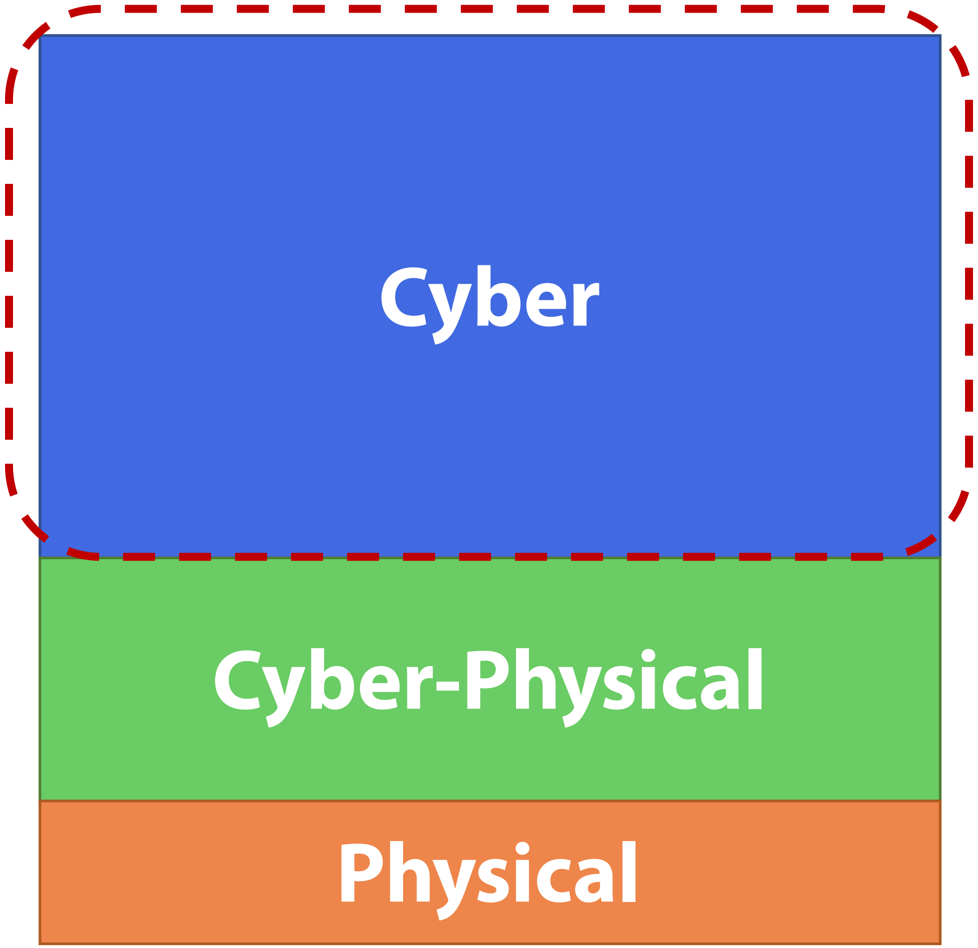
3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

We focus on software threats.

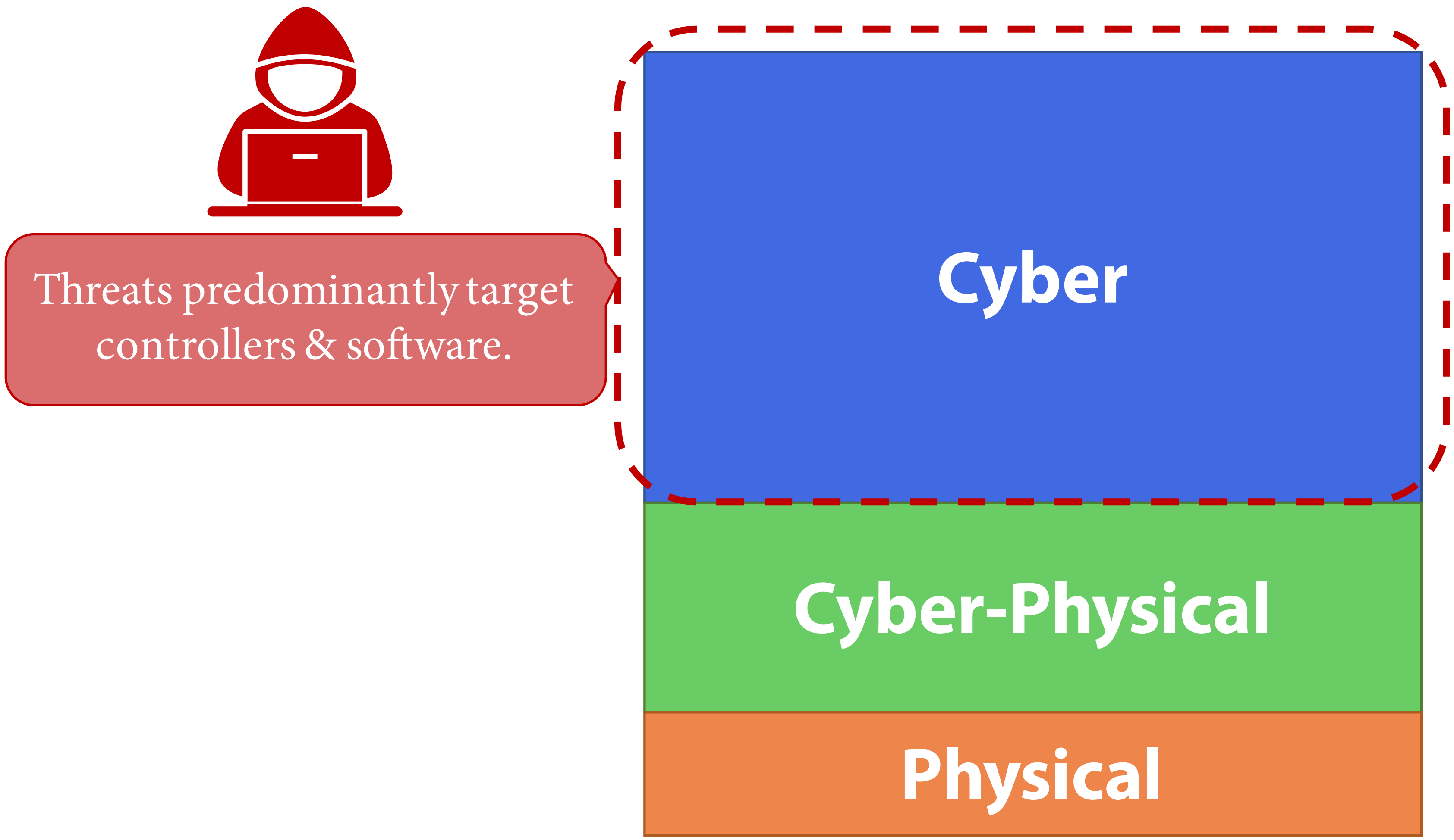
Why focus on software threats?

The cyber layer is the most complex part of a CPS.



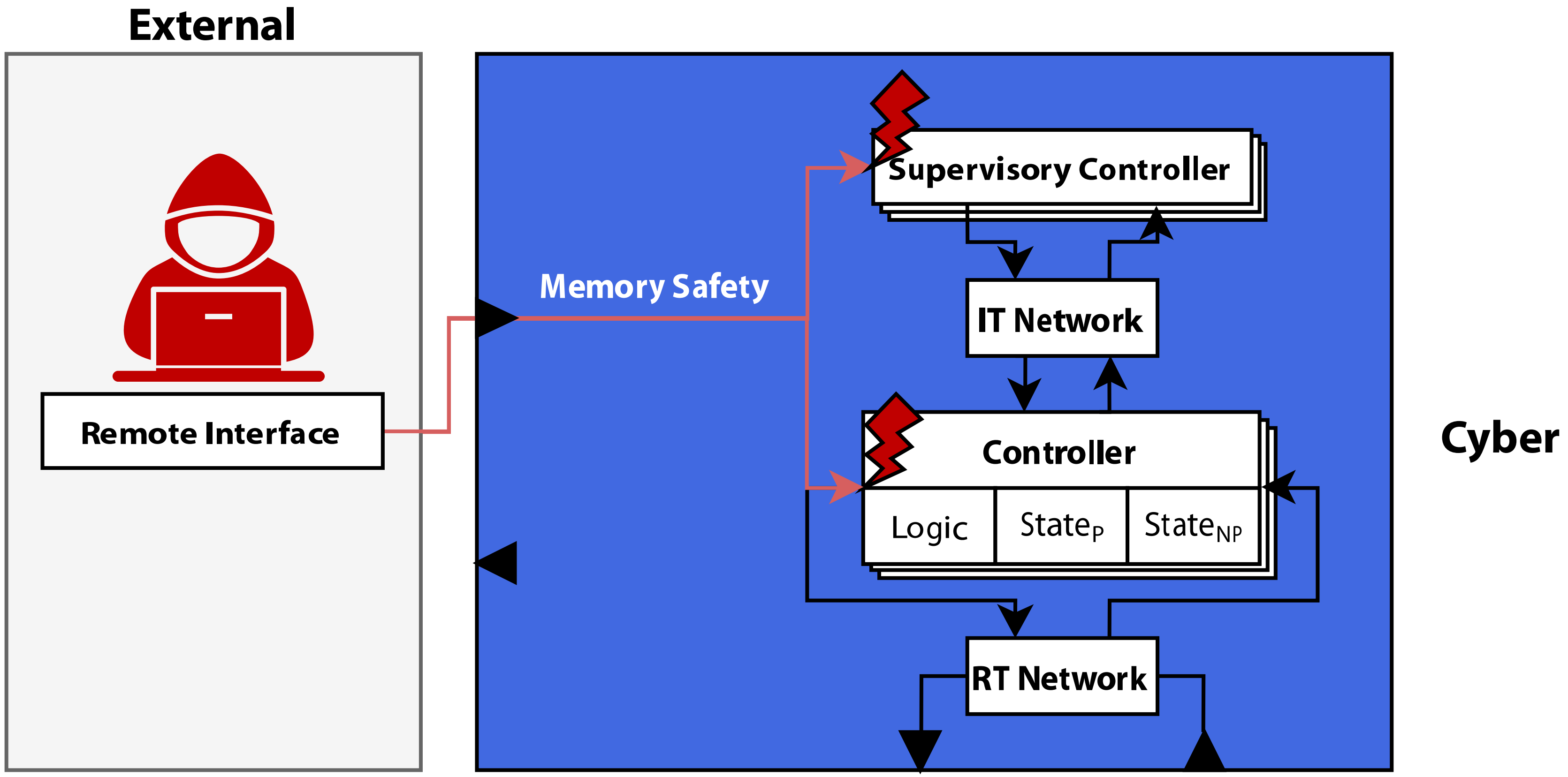
Why focus on software threats?

The cyber layer is the most complex part of a CPS.



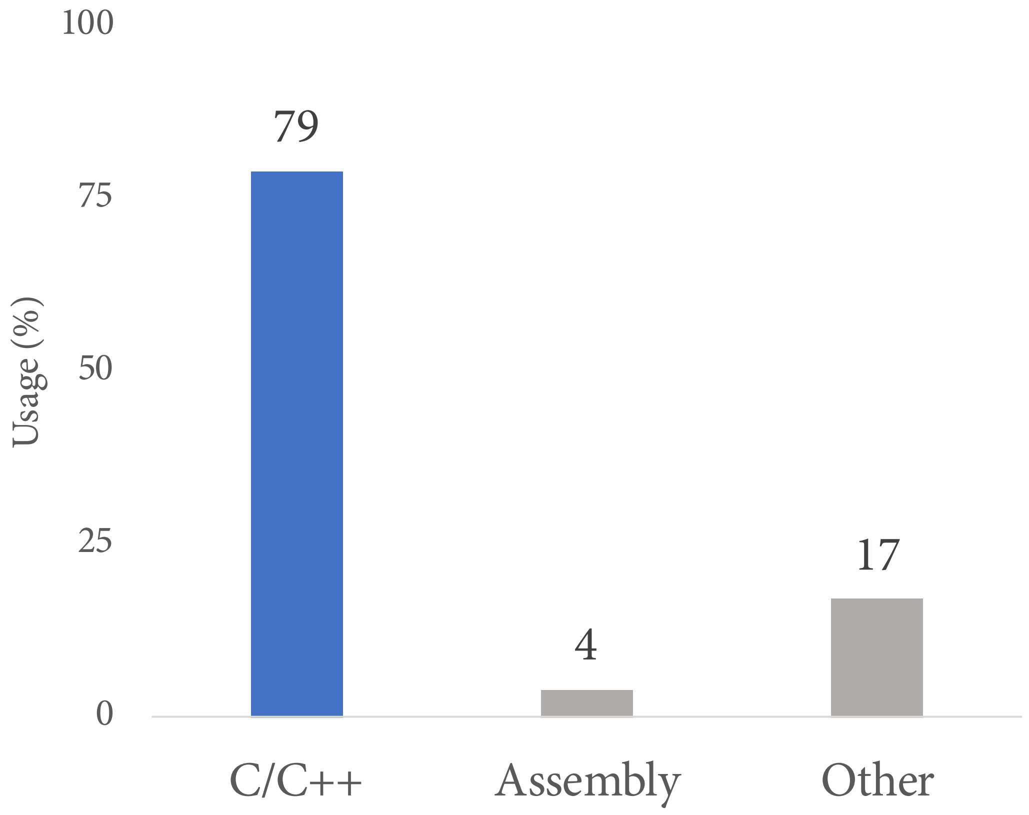
Why focus on software threats?

Memory safety vulnerabilities can target both the supervisory and main controllers.



Why focus on software threats?

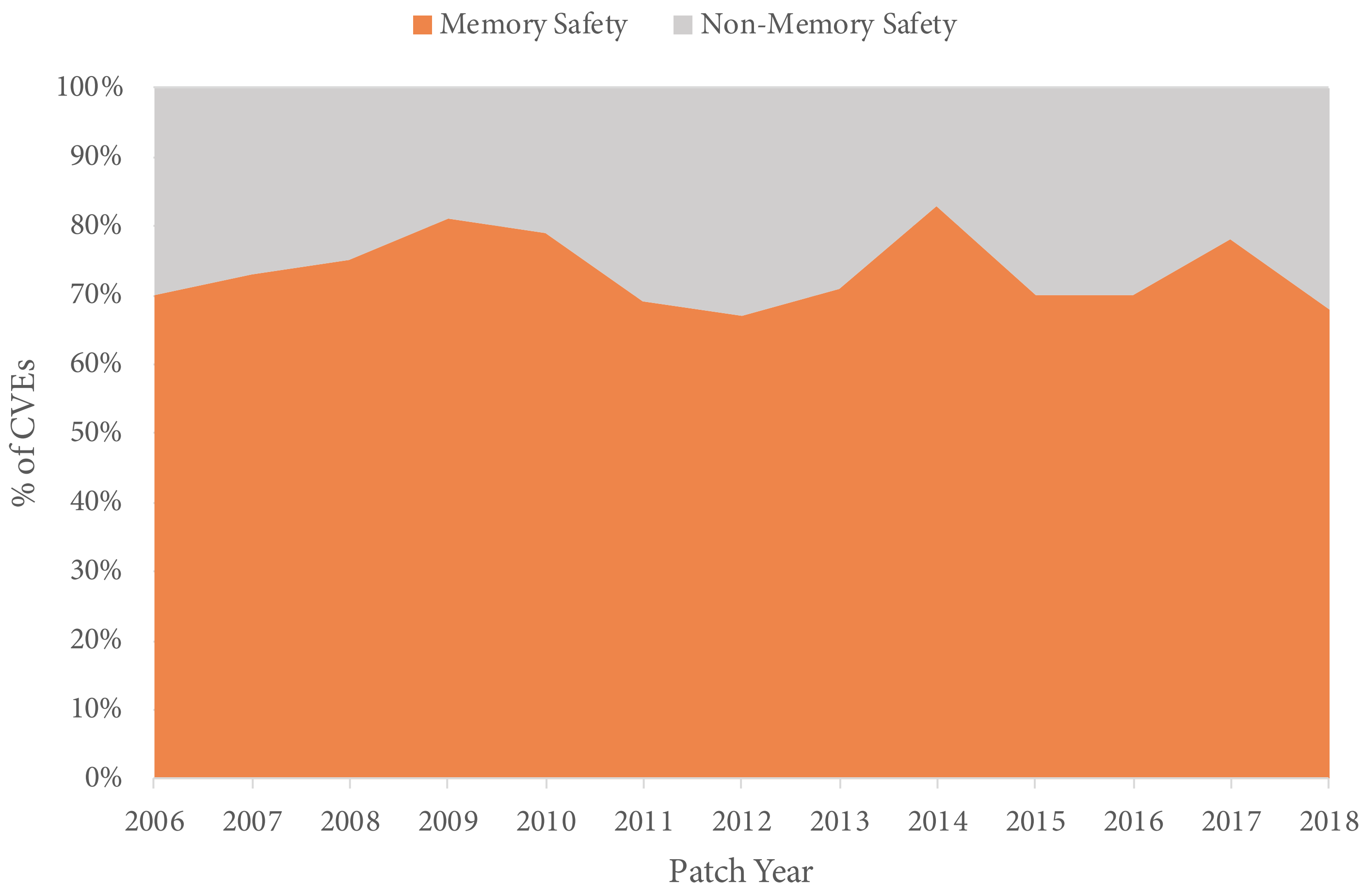
CPSs are predominantely written in memory unsafe languages.



Source: *Embedded Insights Survey 2019*

Why focus on software threats?

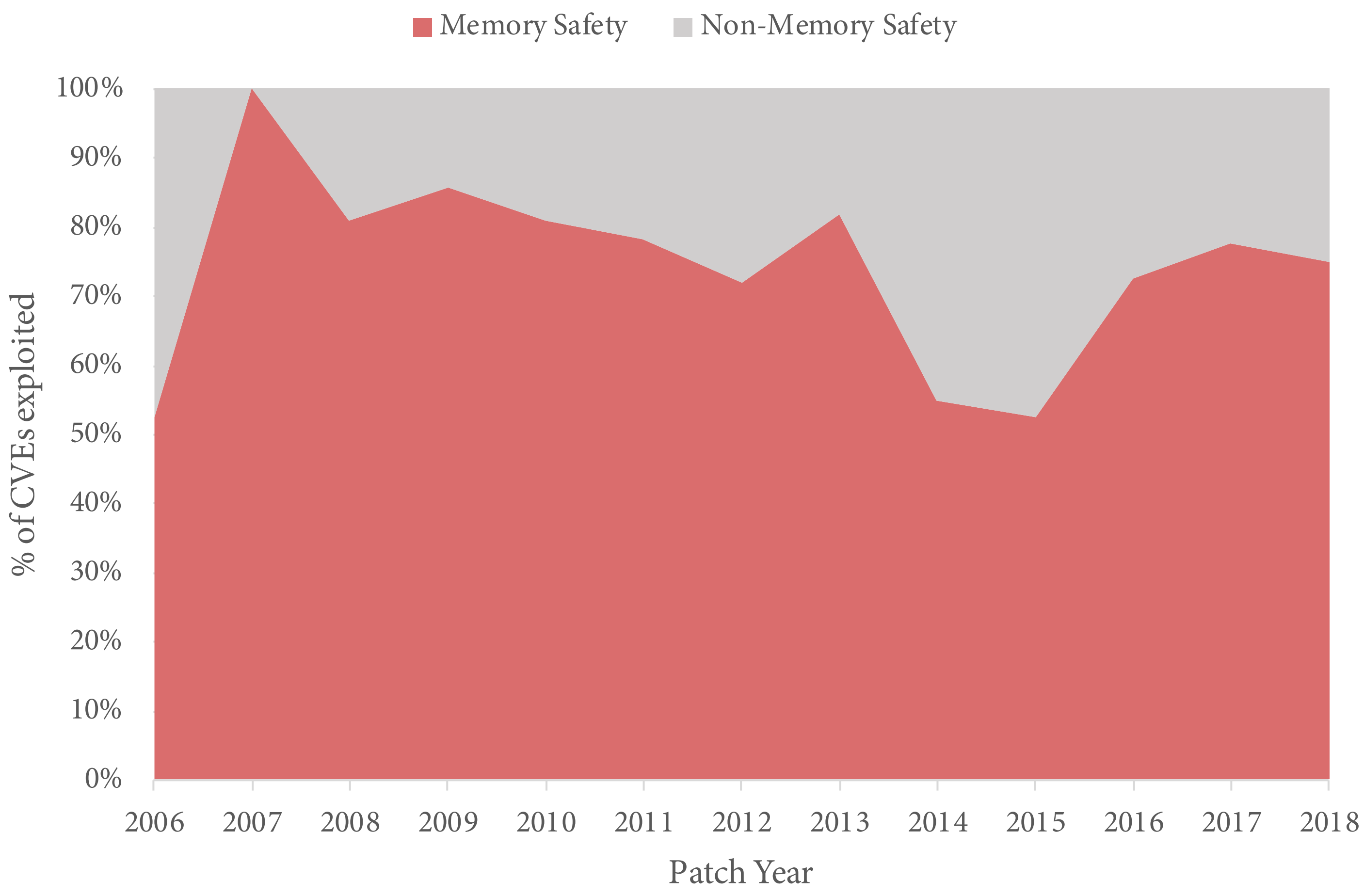
Memory Safety is the predominant source of vulnerabilities (ie. CVEs).



Source: *Matt Miller, Microsoft Security Response Center (MSRC) - BlueHat 2019*

Why focus on software threats?

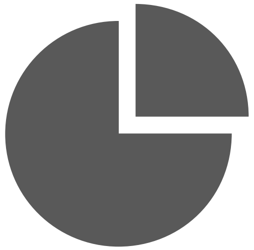
Memory Safety CVEs are heavily exploited.



Source: *Matt Miller, Microsoft Security Response Center (MSRC) - BlueHat 2019*

Why focus on software threats?

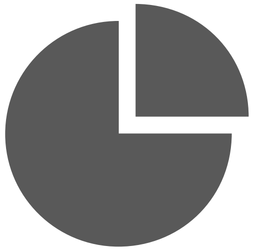
Software provides many entrypoints for an attacker.



SW is a large portion of a CPS.

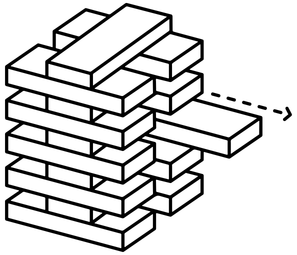
Why focus on software threats?

Software provides many entrypoints for an attacker.



SW is a large portion of a CPS.

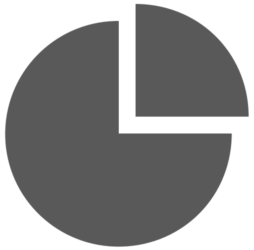
+



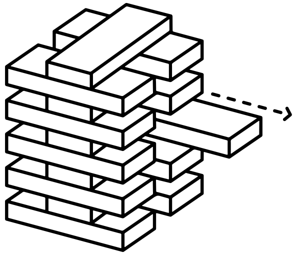
SW is increasingly complex.

Why focus on software threats?

Software provides many entrypoints for an attacker.



SW is a large portion of a CPS.

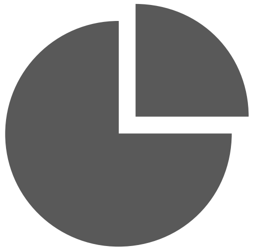


SW is increasingly complex.

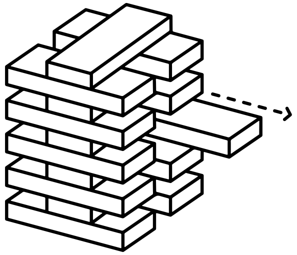
CPSs are pervasive.

Why focus on software threats?

Software provides many entrypoints for an attacker.

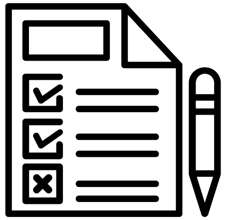


SW is a large portion of a CPS.



SW is increasingly complex.

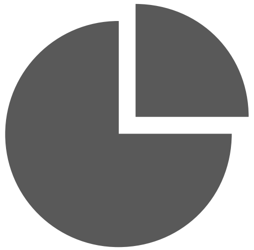
CPSs are pervasive.



Microcontroller features are limited.

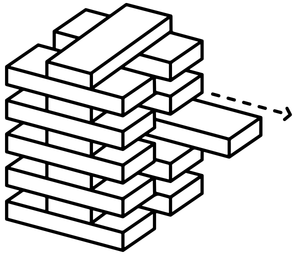
Why focus on software threats?

Software provides many entrypoints for an attacker.



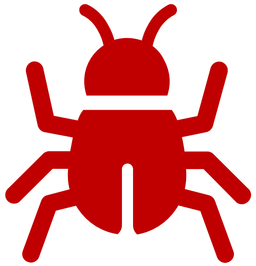
SW is a large portion of a CPS.

+



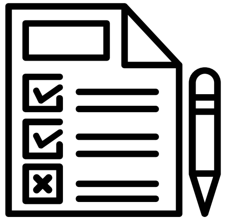
SW is increasingly complex.

=



More Bugs

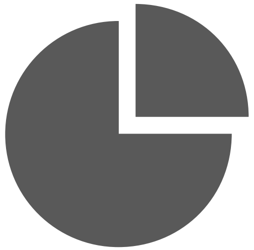
CPSs are pervasive.



Microcontroller features are limited.

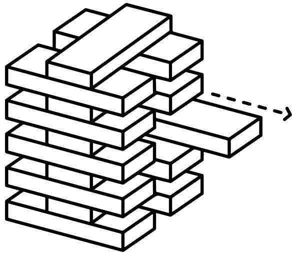
Why focus on software threats?

Software provides many entrypoints for an attacker.



SW is a large portion of a CPS.

+

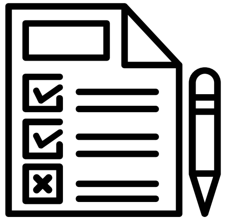


SW is increasingly complex.

=

More Vulns

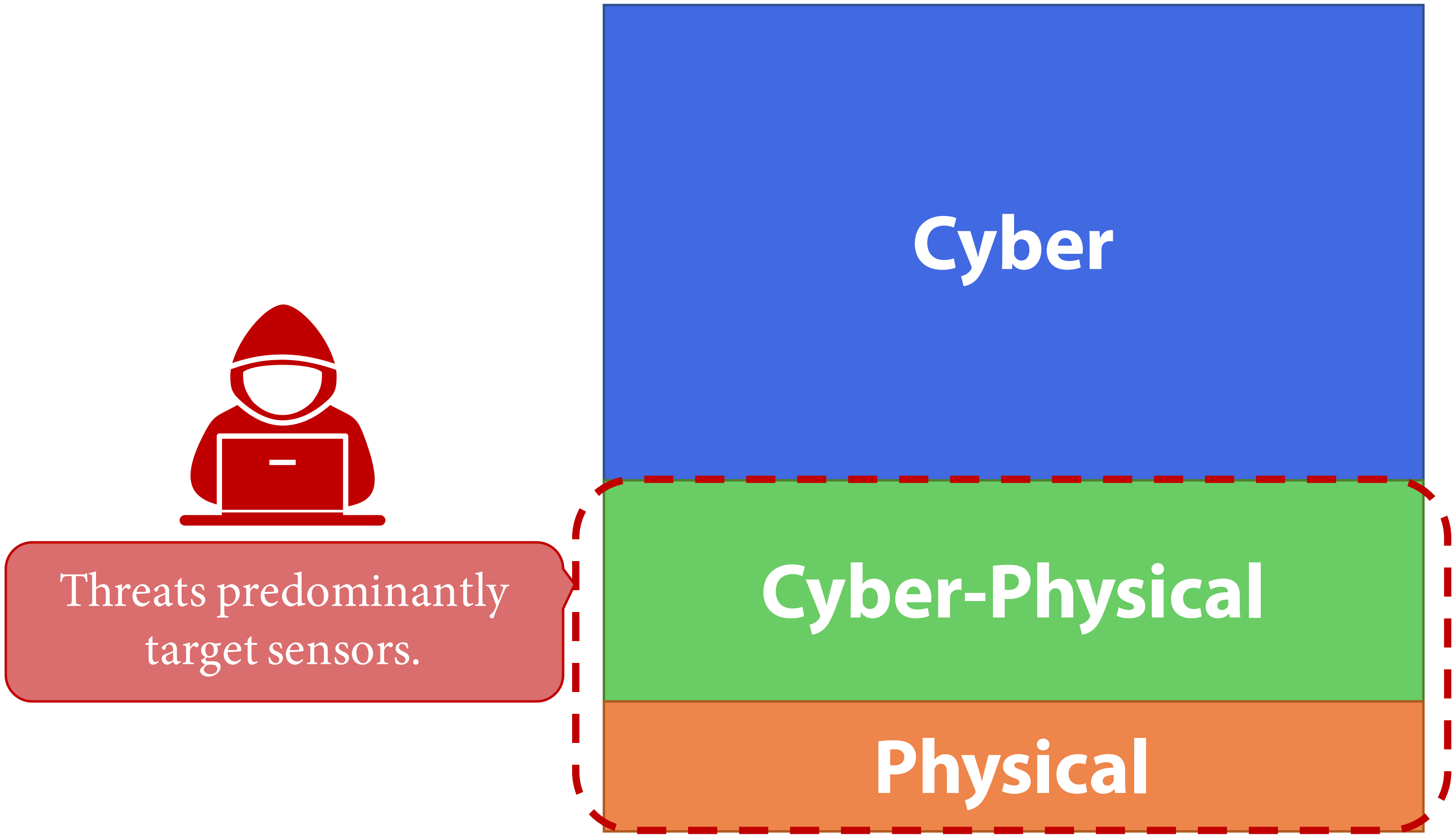
CPSs are pervasive.



Microcontroller features are limited.

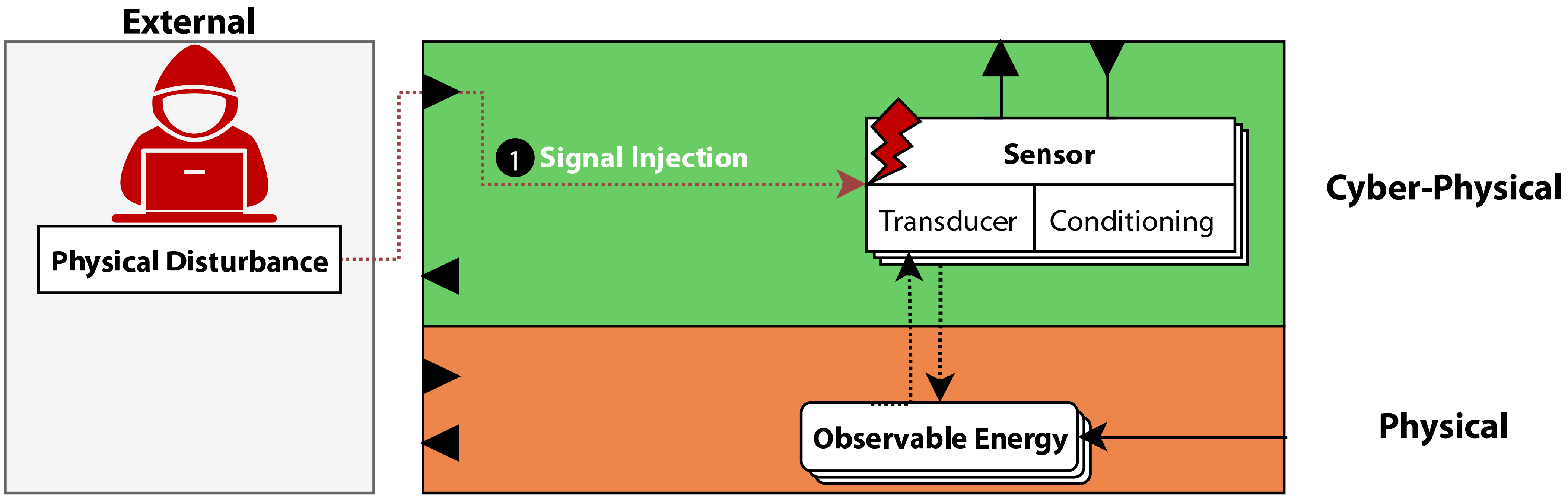
Why focus on software threats?

(Cyber-)Physical threats are more limited in scope.



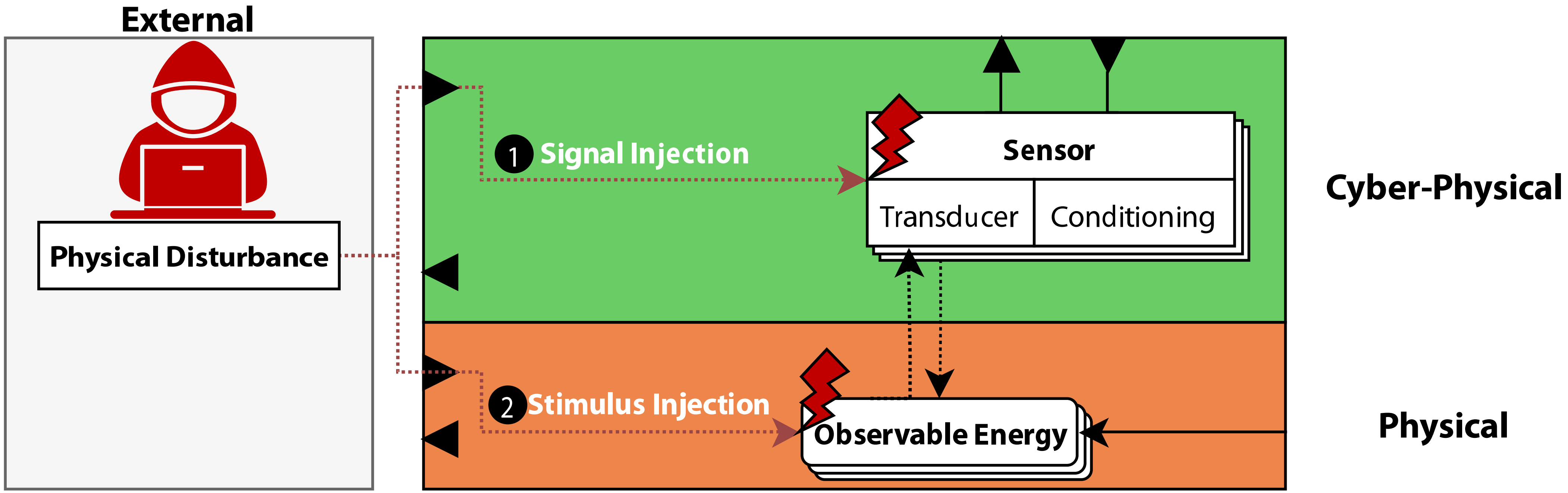
Why focus on software threats?

Sensors can be manipulated by exploiting how they observe physical phenomena.



Why focus on software threats?

Sensors can be manipulated by exploiting how they observe physical phenomena.



Why focus on software threats?

Software vulnerabilites are more flexibile for an attacker.

Software Vulnerabilities

Access

Physical Vulnerabilities

Why focus on software threats?

Software vulnerabilites are more flexibile for an attacker.

Why focus on software threats?

Software vulnerabilites are more flexibile for an attacker.

Software Vulnerabilities		Physical Vulnerabilities
	Access	
	Coverage	
	Privacy	

Why focus on software threats?

Software vulnerabilites are more flexibile for an attacker.

Software Vulnerabilities

Physical Vulnerabilities

Access

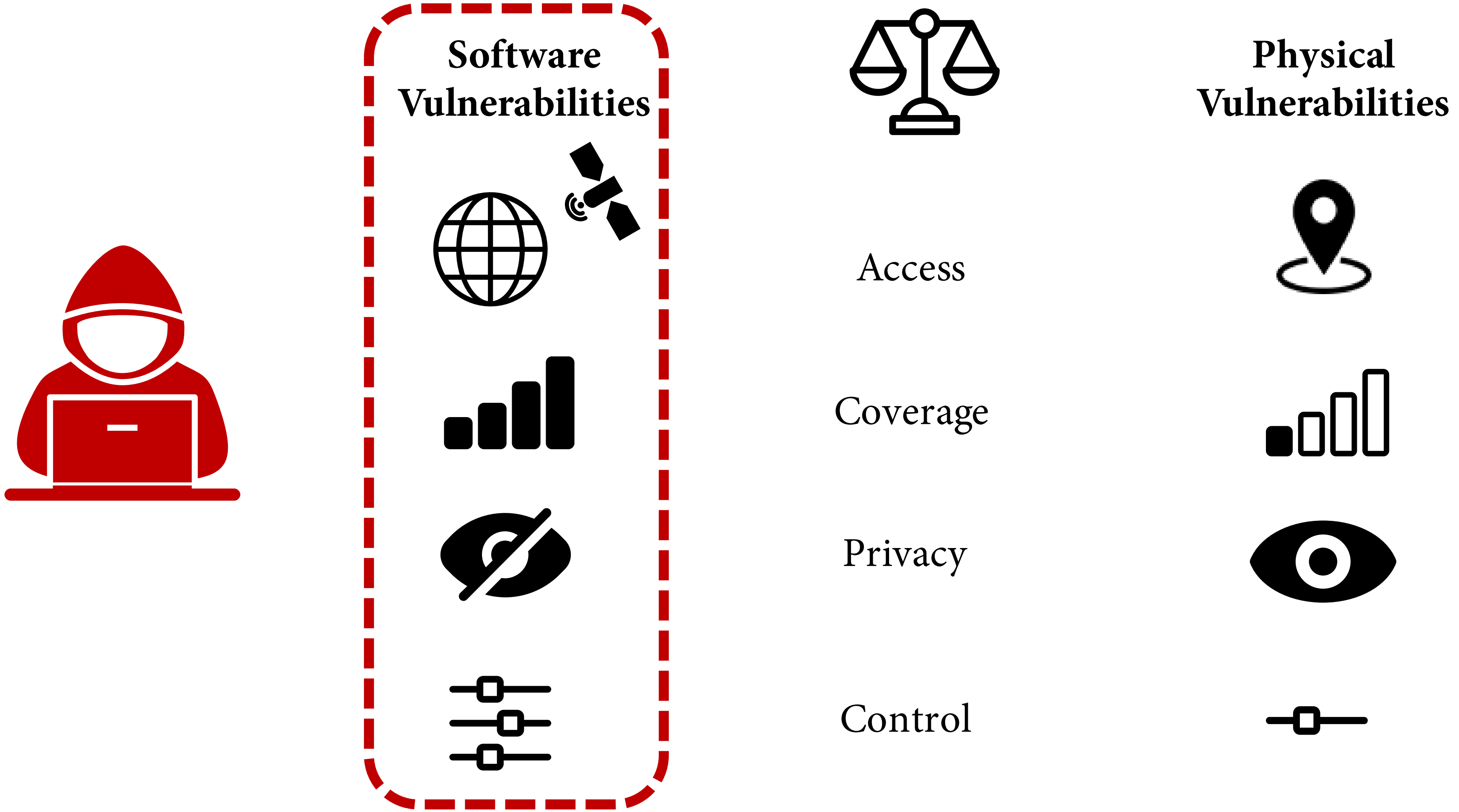
Coverage

Privacy

Control

Why focus on software threats?

Software vulnerabilites are more flexibile for an attacker.

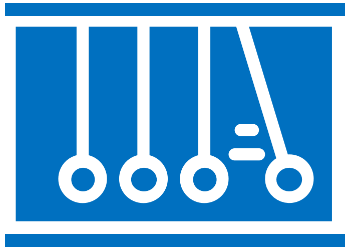


CPSs need
low overhead deployable *software* security.

My contributions to CPS security

My contributions to CPS security

An overview of publications



1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

YOLO
You Only Live Once

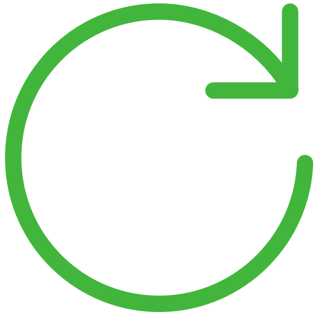
Appears as

YOLO: Frequently Resetting Cyber-Physical Systems for Security
Arroyo, M., Tarek Ibn Ziad, M., Kobayashi, H., Yang, J., Sethumadhavan, S.
SPIE Defense & Commercial Sensing 2019
(DOI: 10.1117/12.2518909)

Patent
US10417425

YOLO in a nutshell

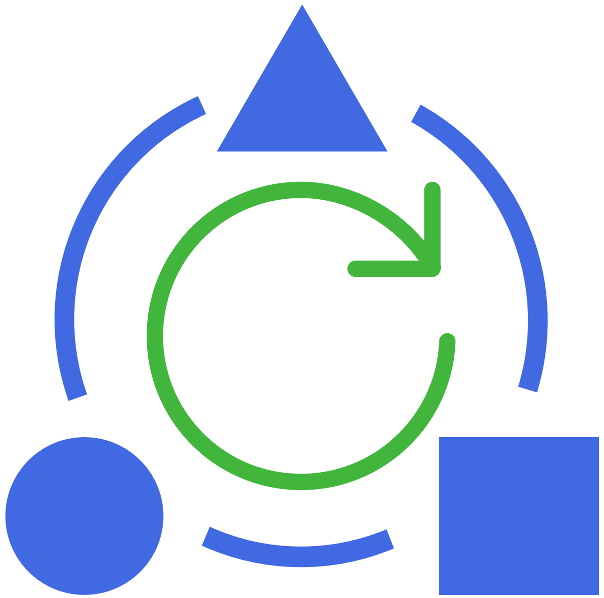
A flexible framework for wiping an attacker from a system.



1. Reset

YOLO in a nutshell

A flexible framework for wiping an attacker from a system.



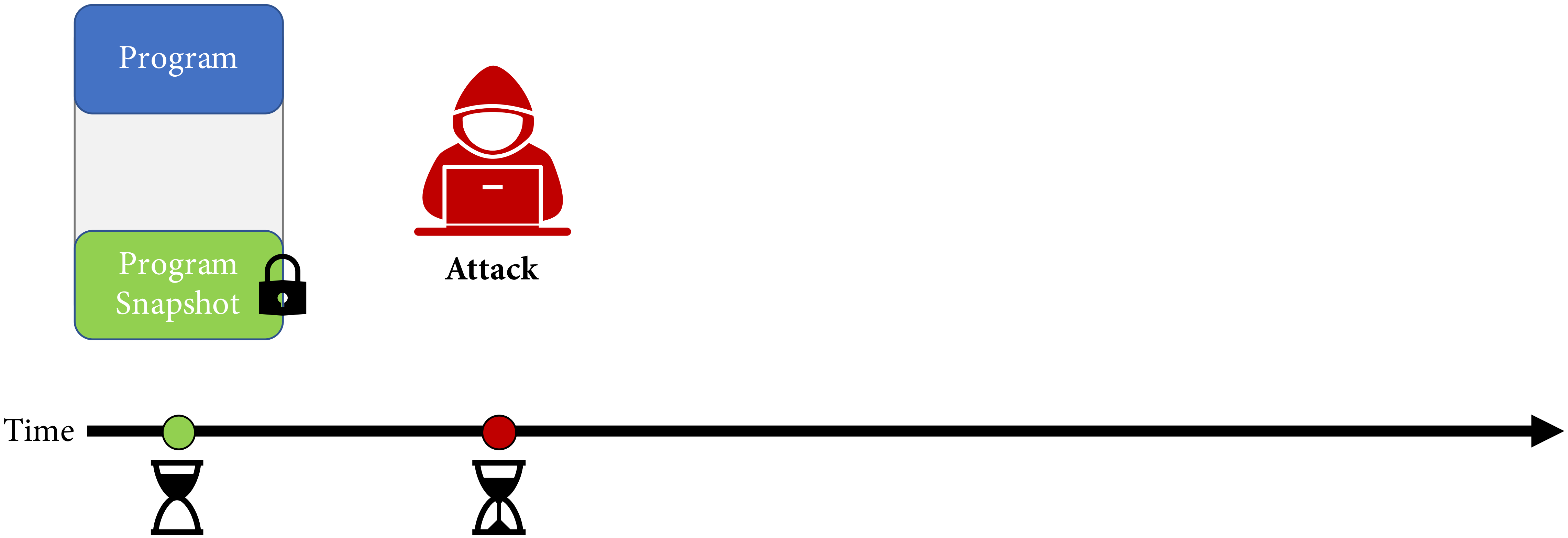
1. Reset
2. Diversify

Why does YOLO perform resets?

Bounds the time an adversary can affect the system.

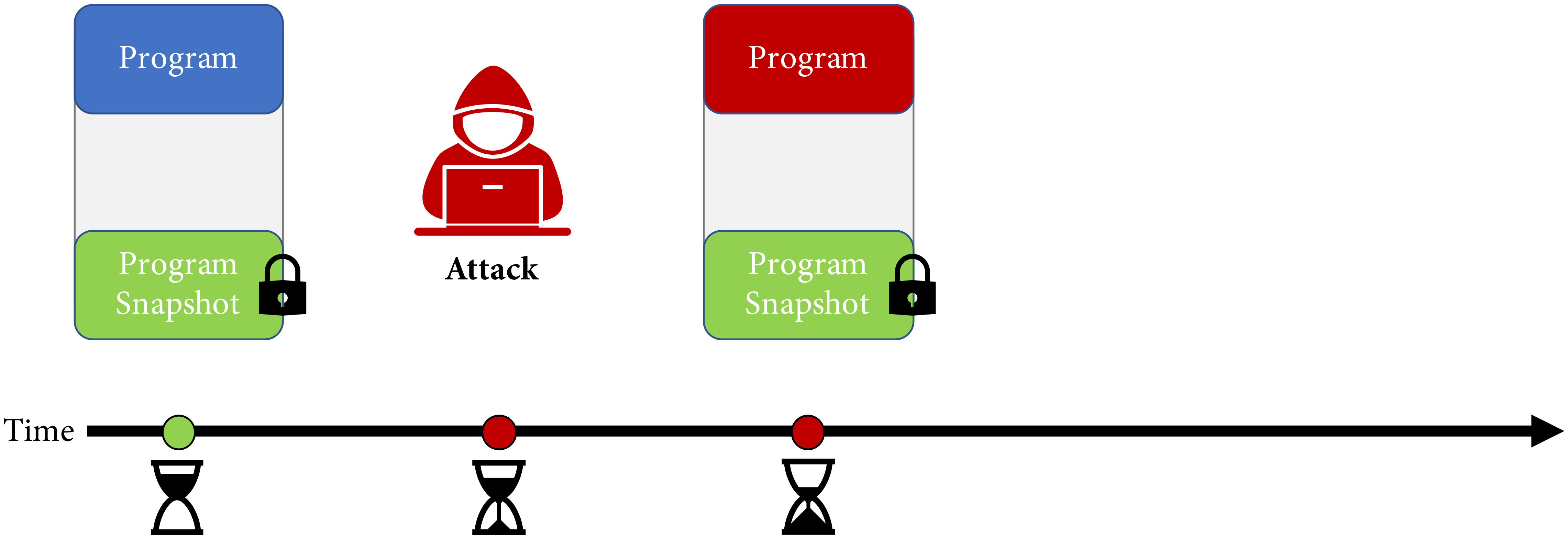
Why does YOLO perform resets?

Bounds the time an adversary can affect the system.



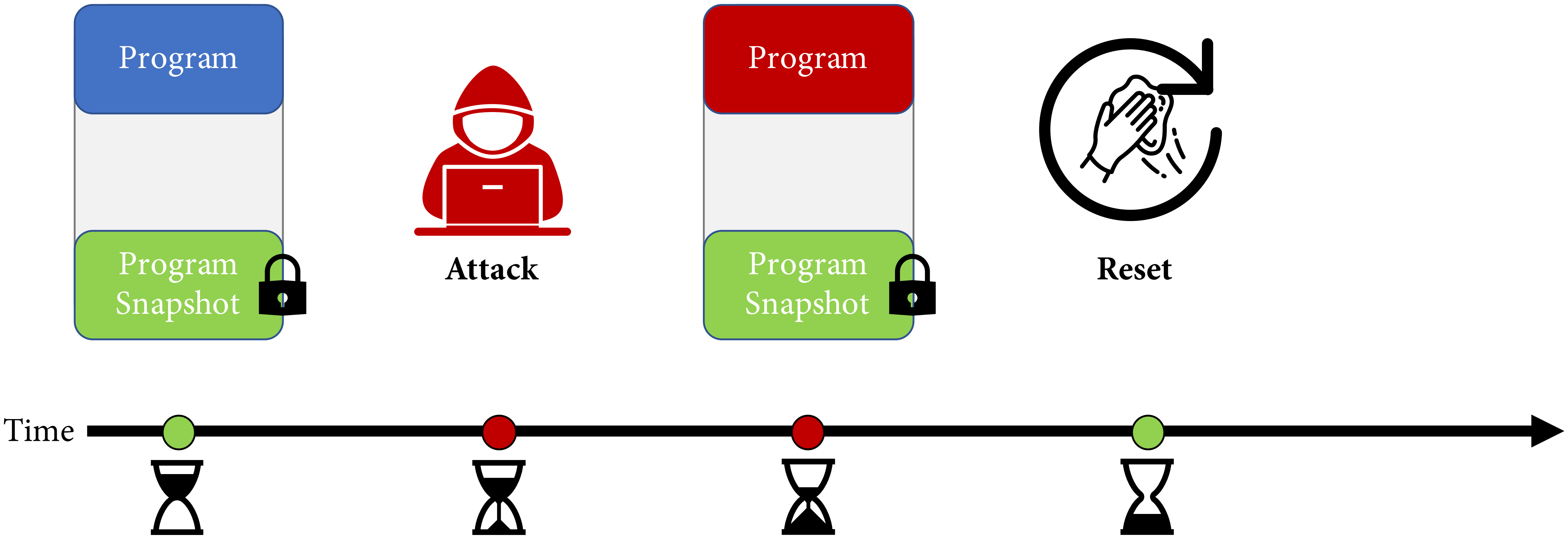
Why does YOLO perform resets?

Bounds the time an adversary can affect the system.



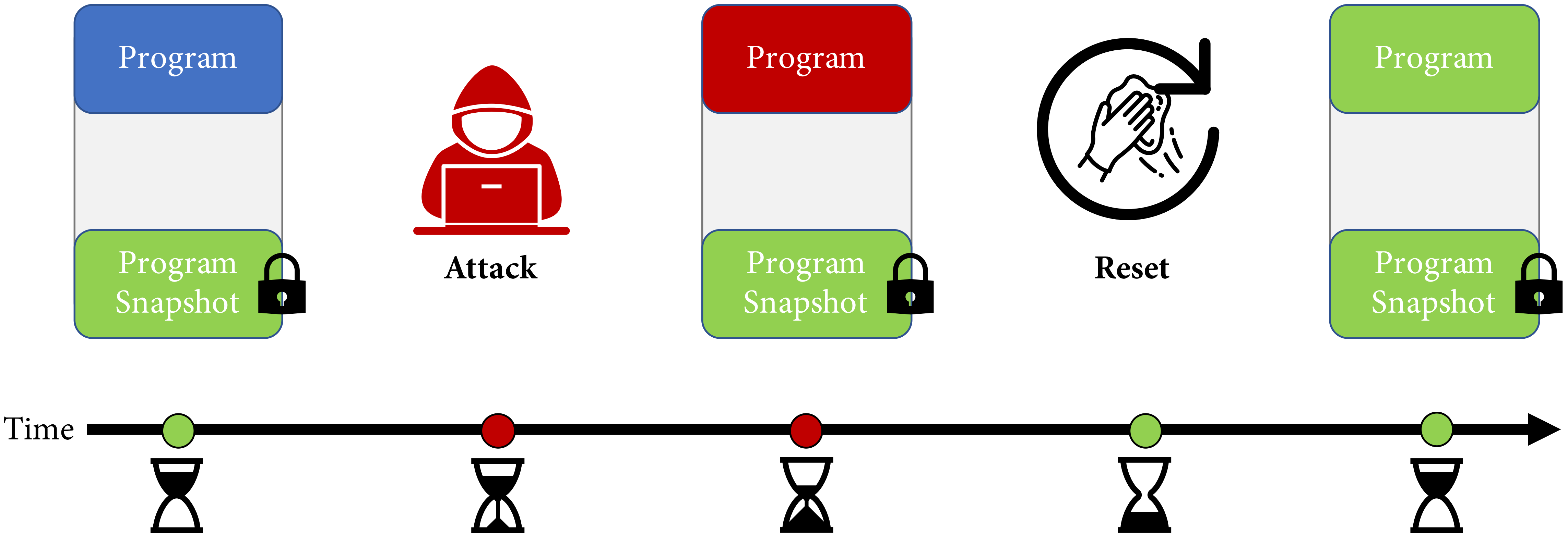
Why does YOLO perform resets?

Bounds the time an adversary can affect the system.



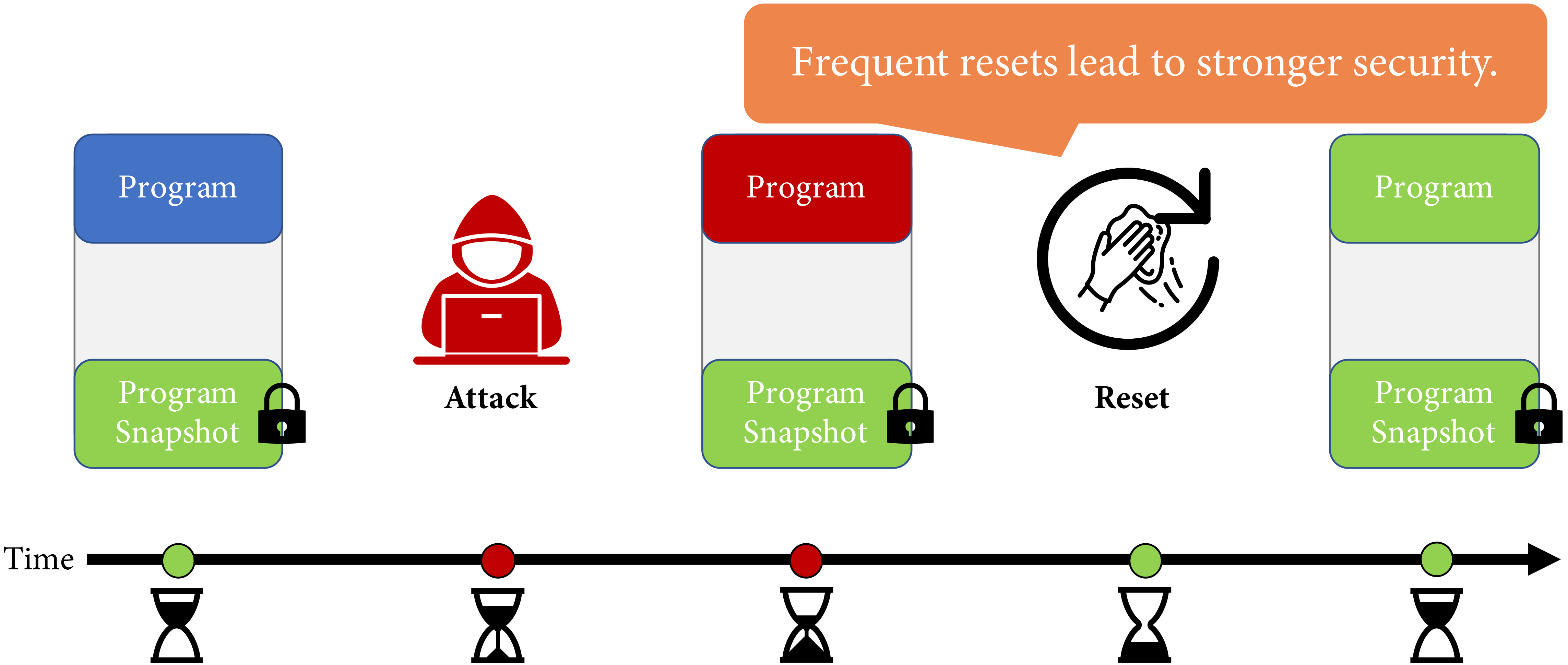
Why does YOLO perform resets?

Bounds the time an adversary can affect the system.



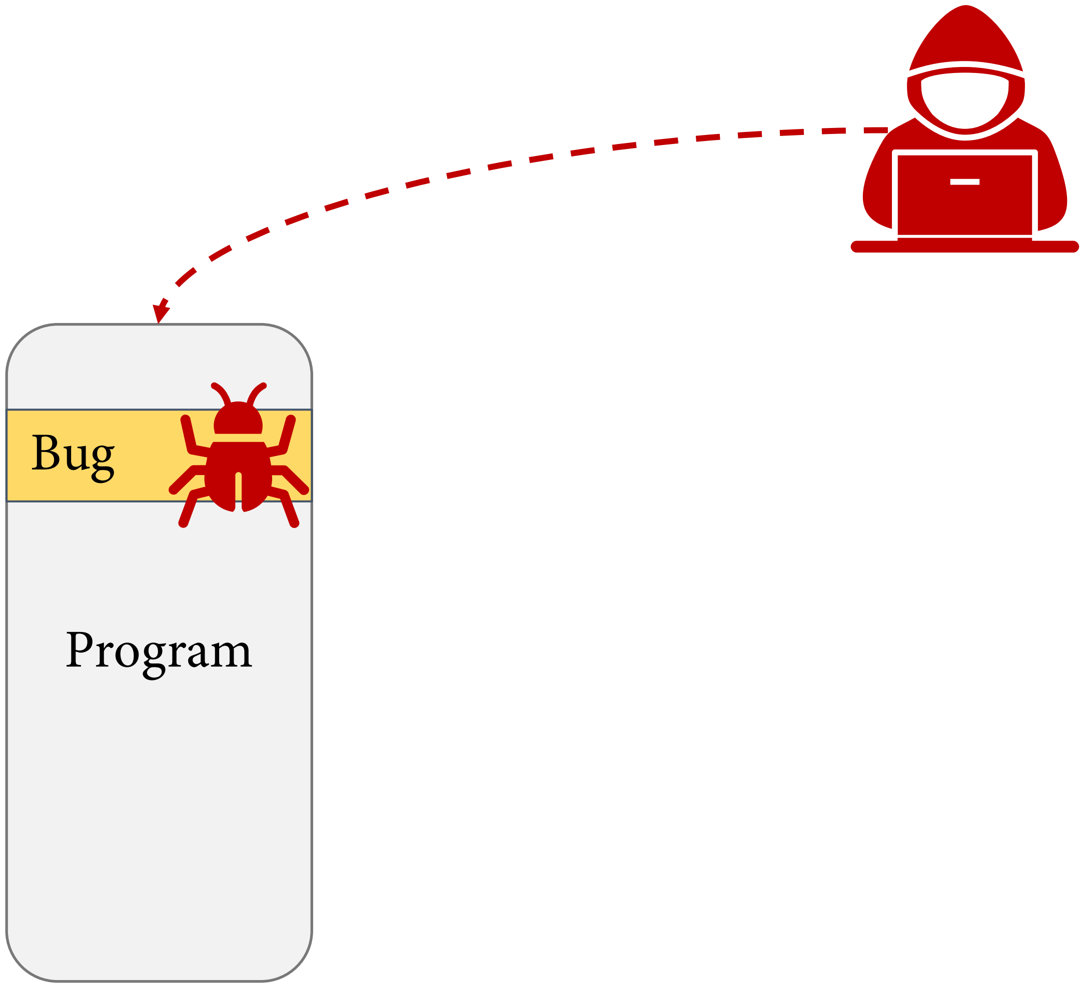
Why does YOLO perform resets?

Bounds the time an adversary can affect the system.



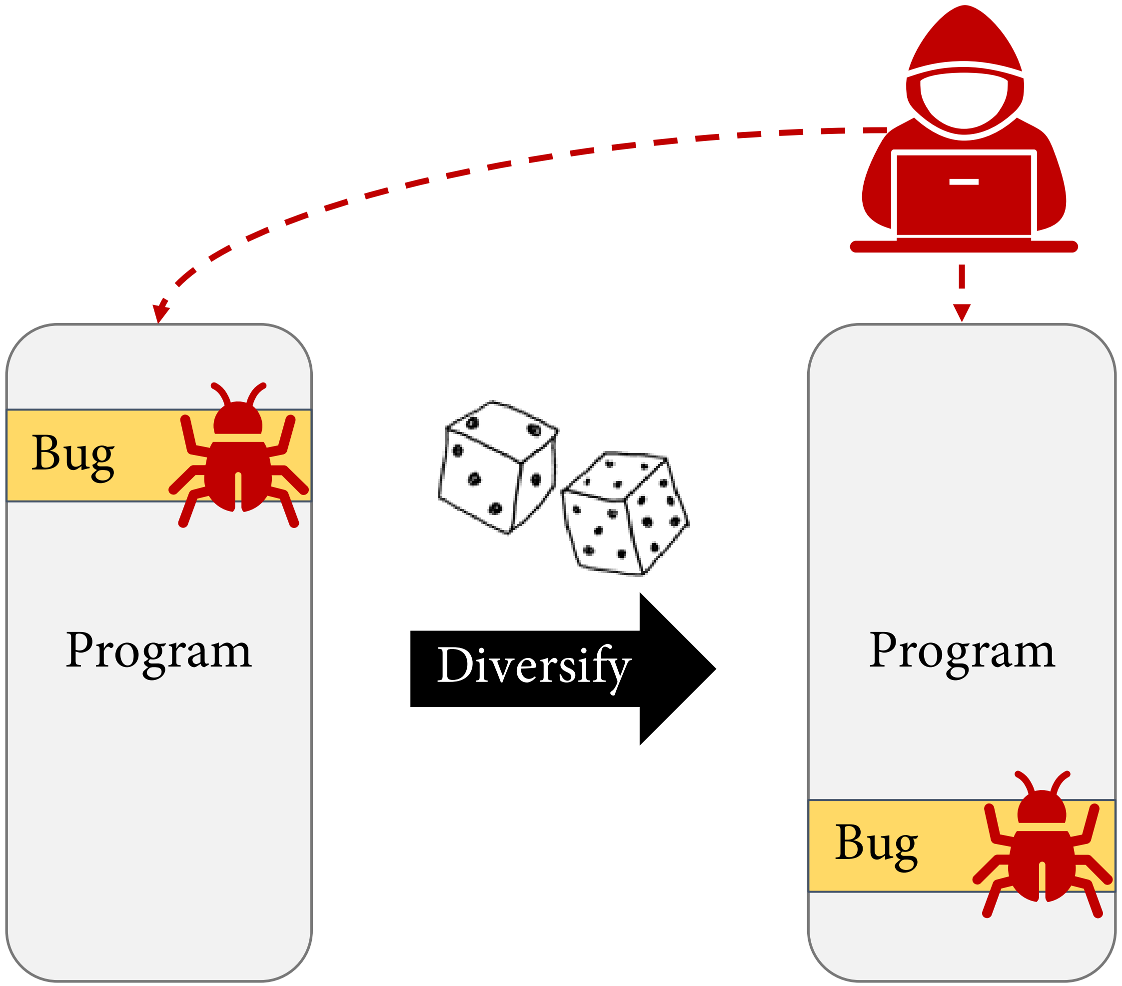
Why does YOLO need to diversify?

Prevents a system from compromise by the same method continuously.



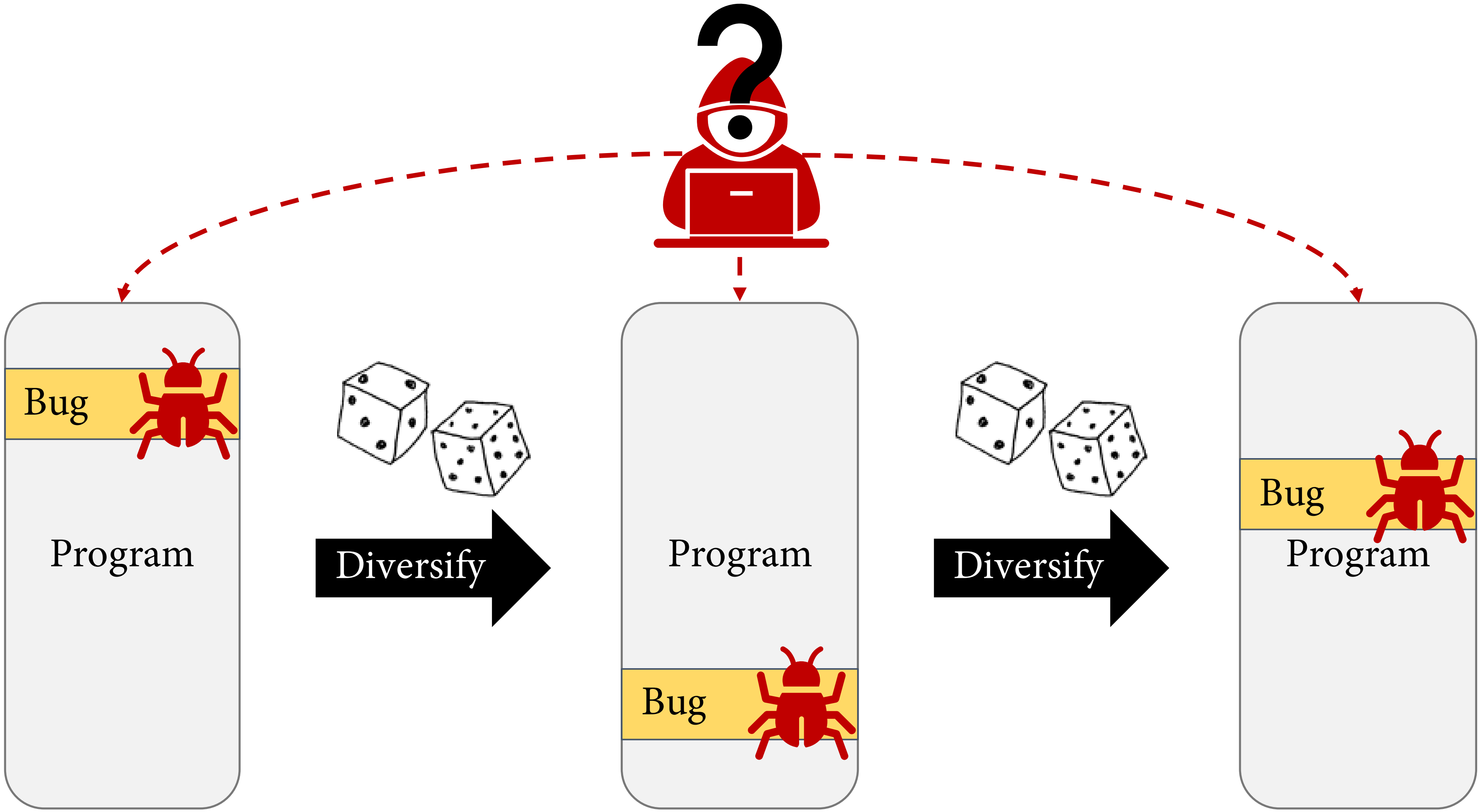
Why does YOLO need to diversify?

Prevents a system from compromise by the same method continuously.



Why does YOLO need to diversify?

Prevents a system from compromise by the same method continuously.



Why does YOLO work for CPSs?

Leverages physical properties to cope with reset & diversify mechanisms.

Inertia
Allows for operation during resets & diversification.

Why does YOLO work for CPSs?

Leverages physical properties to cope with reset & diversify mechanisms.

Inertia
Allows for operation during resets & diversification.

Observability
Feedback loop allows state to be learned by reobserving state.

Why does YOLO work for CPSs?

Leverages physical properties to cope with reset & diversify mechanisms.

Inertia
Allows for operation during resets & diversification.

Observability
Feedback loop allows state to be learned by reobserving state.

Physically Bounded
An attacker needs time to affect the behavior of the system.

How was YOLO evaluated?

We used two real-world systems.

Engine Control Unit (ECU)

Flight Control Unit (FCU)

Special thanks to Columbia FSAE!

How was YOLO evaluated?

Engine Control Unit (ECU)
Reset every 125ms

Flight Control Unit (FCU)
Reset every 1s

Why is YOLO well suited for constrained devices?

Inertia is used in place of additional resources to maintain system stability.

Cost Savings

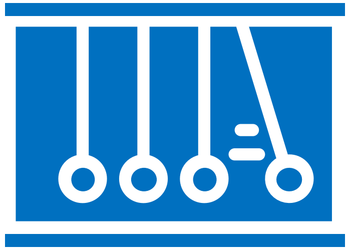
YOLO does not require redundant resources to maintain system stability & security.

Legacy Systems

YOLO can be easily retrofitted into existing systems.

My contributions to CPS security

An overview of publications



1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

My contributions to CPS security

An overview of publications

1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

PAS
Phantom Address Space

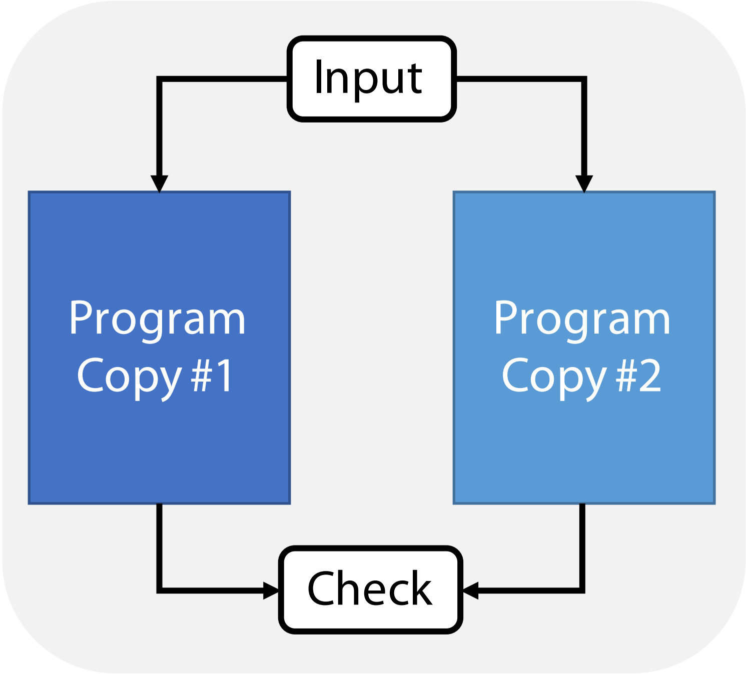
Currently under submission.

Patent
US62904887

PAS in a nutshell

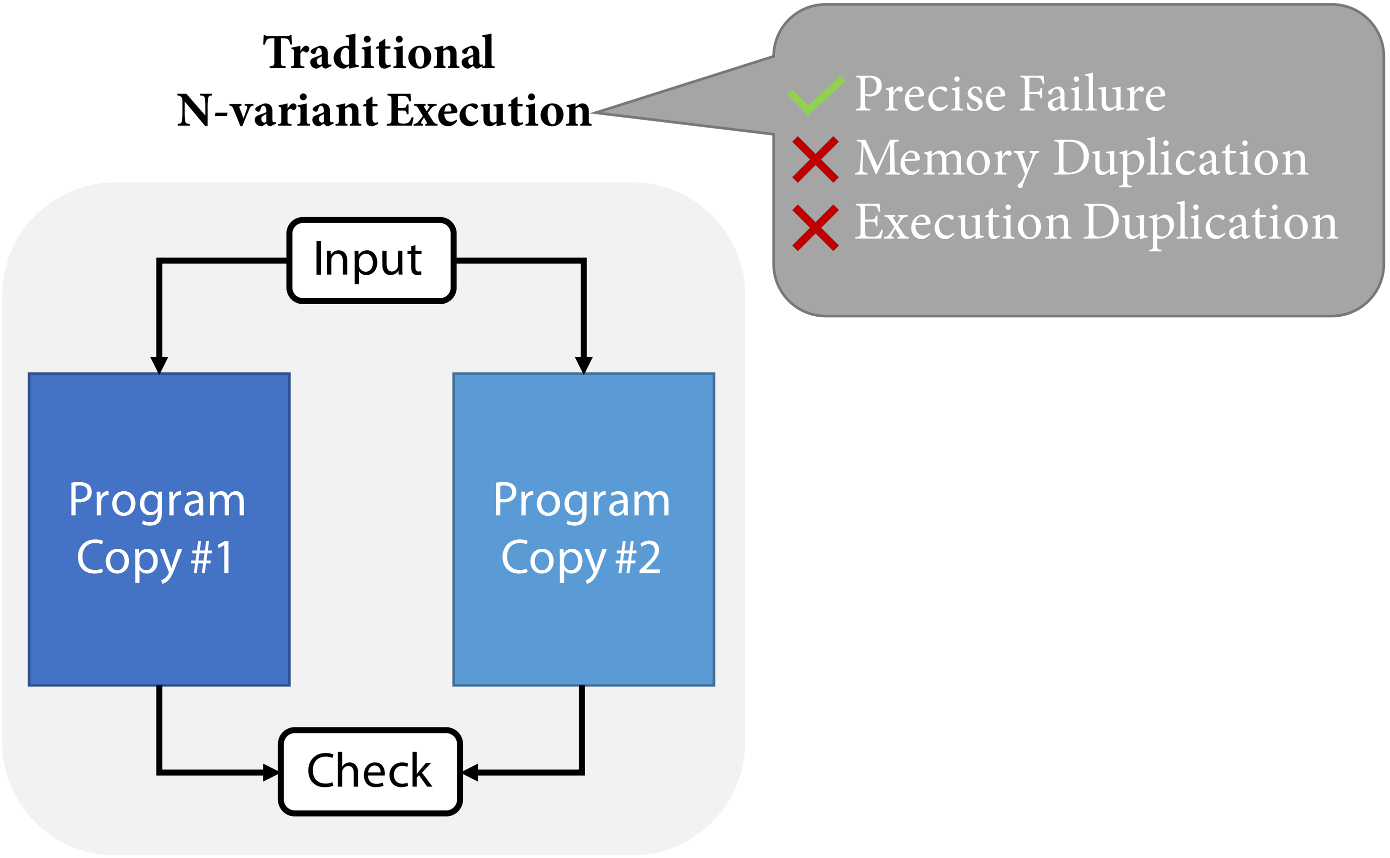
An architectural concept to efficiently enable N-variant execution.

Traditional N-variant Execution



PAS in a nutshell

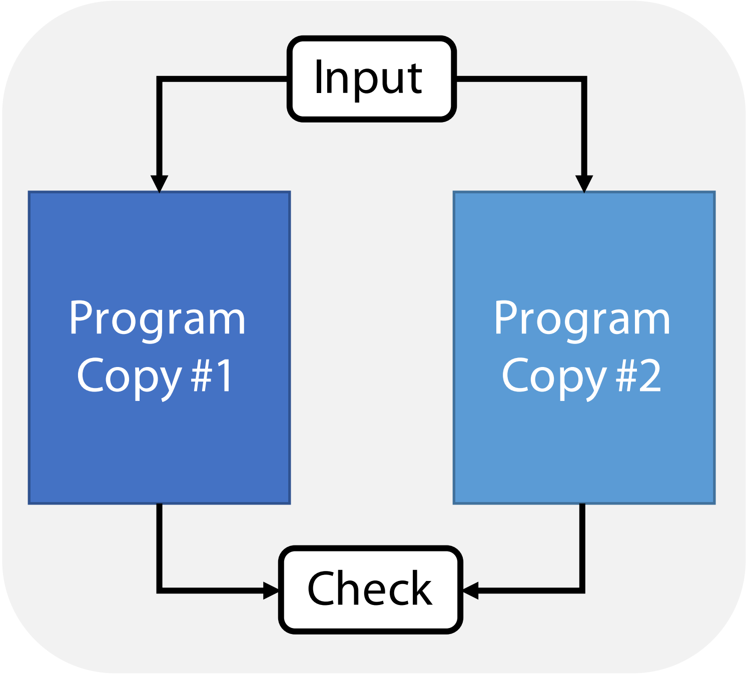
An architectural concept to efficiently enable N-variant execution.



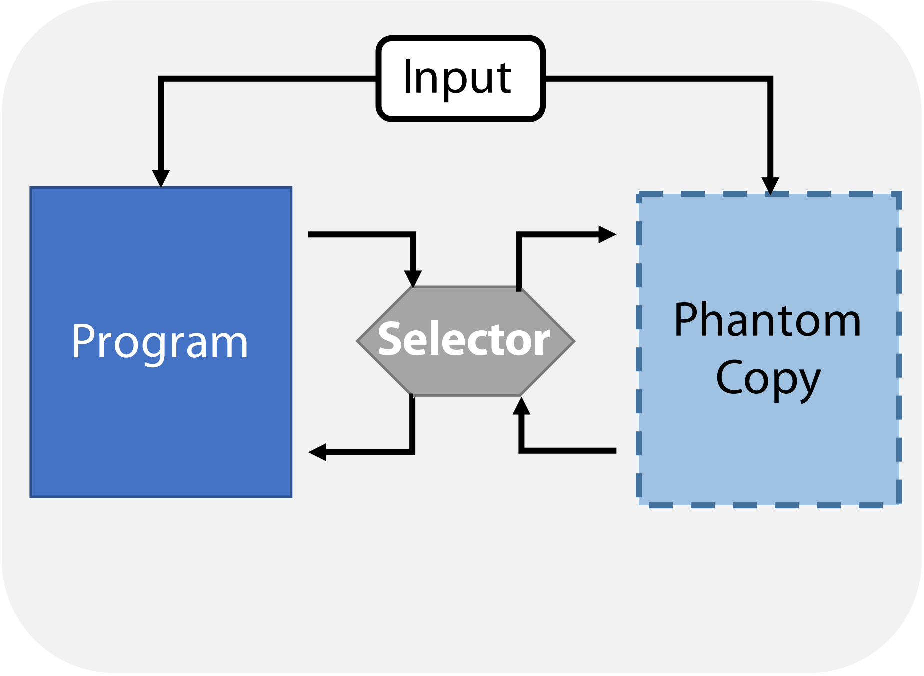
PAS in a nutshell

An architectural concept to efficiently enable N-variant execution.

Traditional N-variant Execution

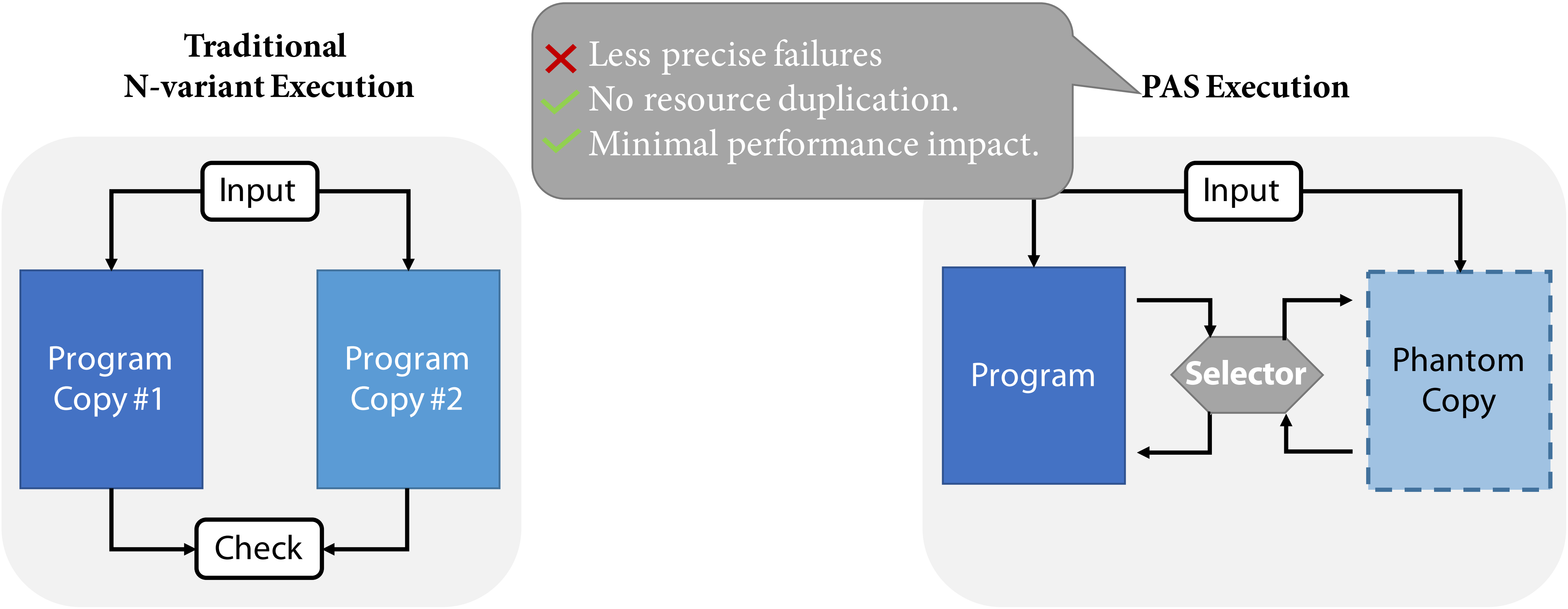


PAS Execution



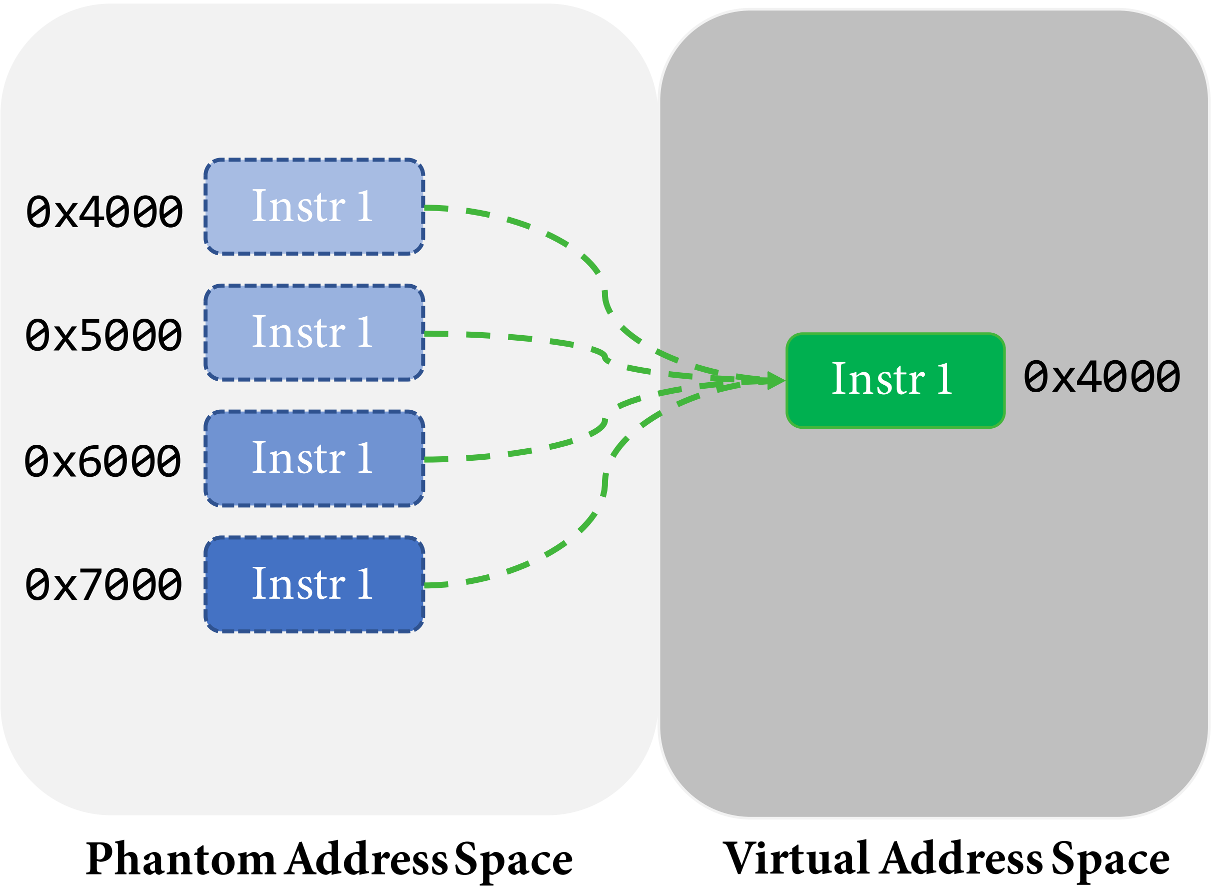
PAS in a nutshell

An architectural concept to efficiently enable N-variant execution.



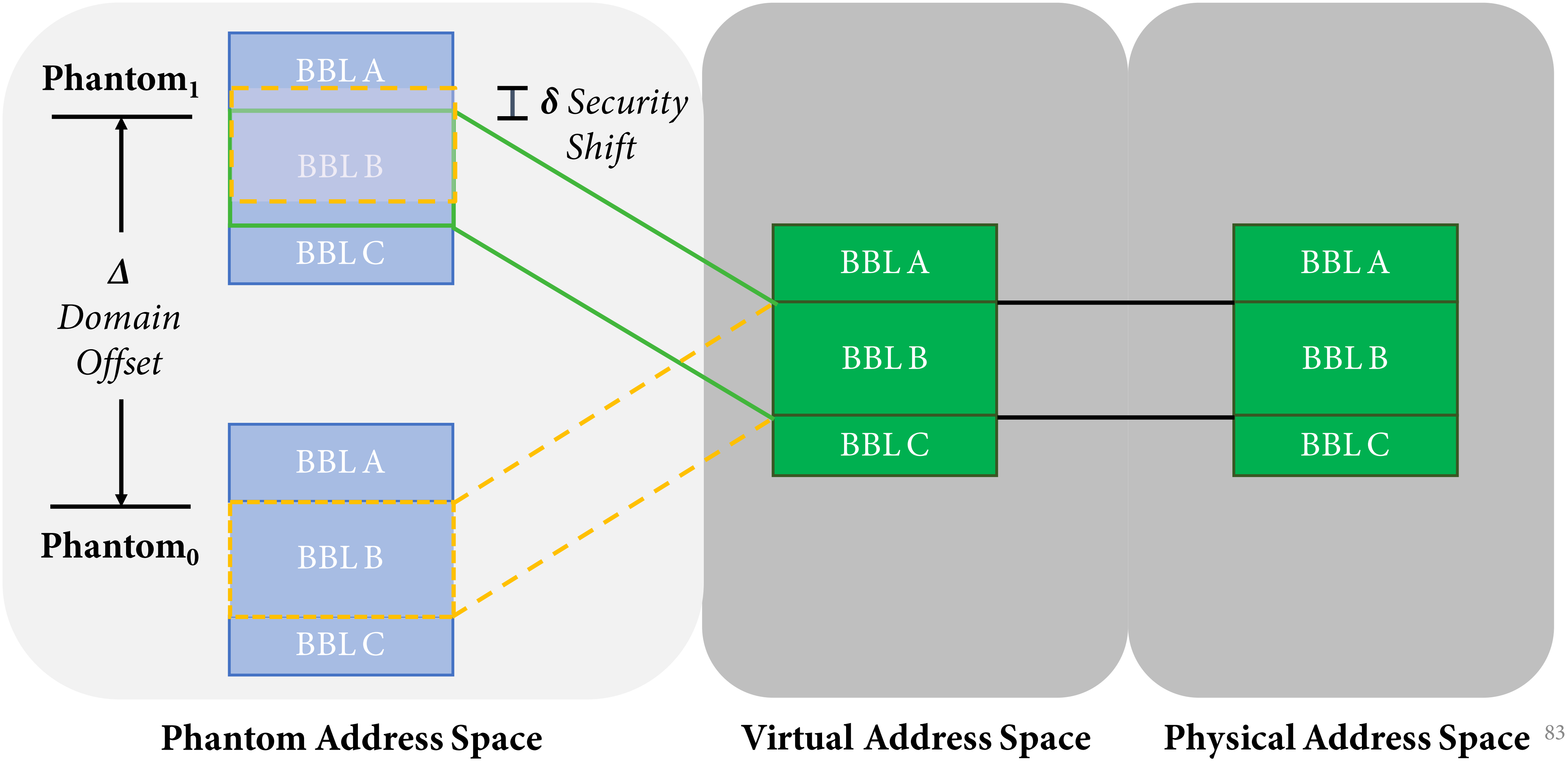
PAS in a nutshell

Multiple phantom addresses alias to an instruction.



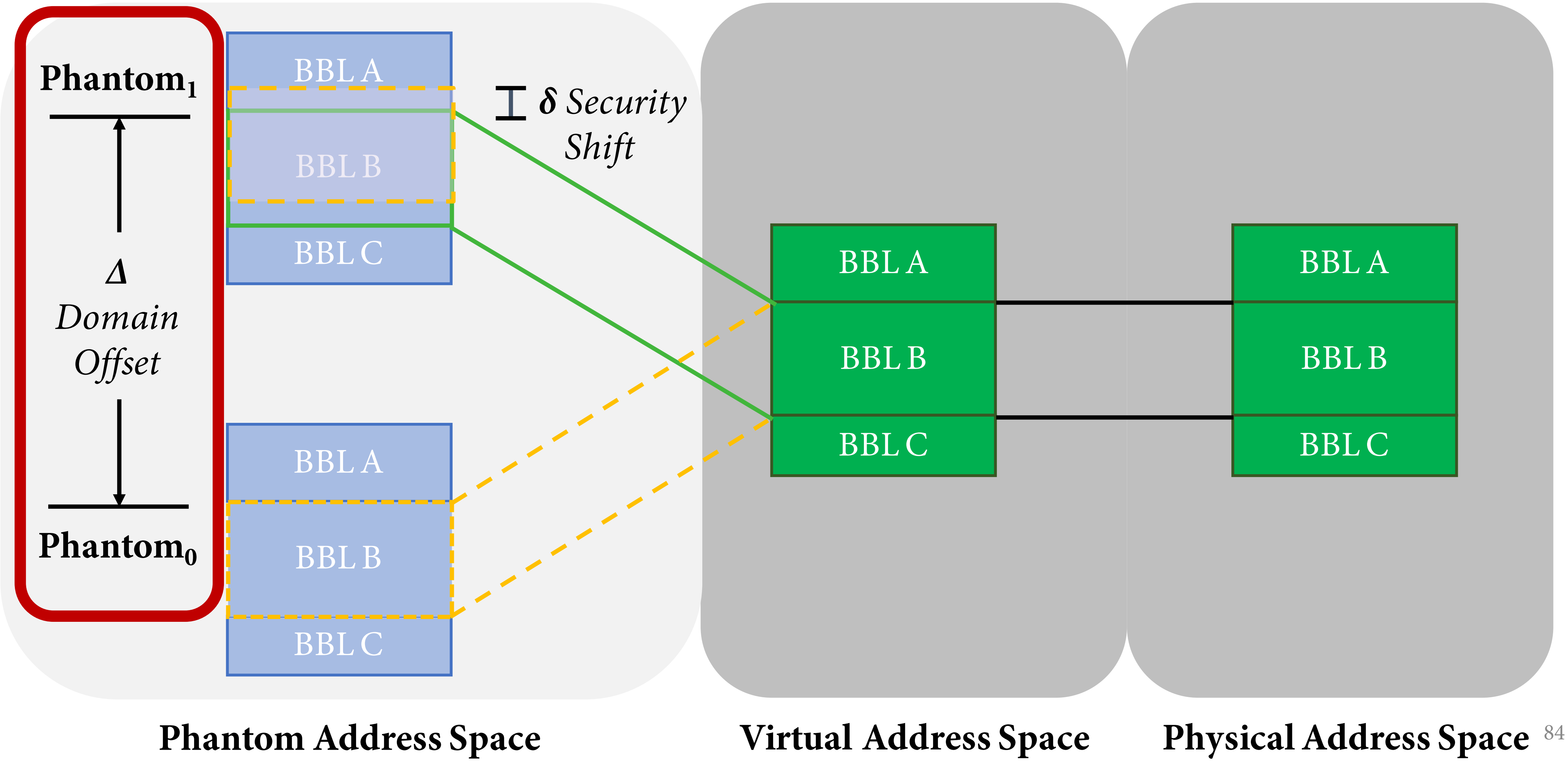
How are phantoms constructed?

Phantoms are logically displaced relative to the original program.



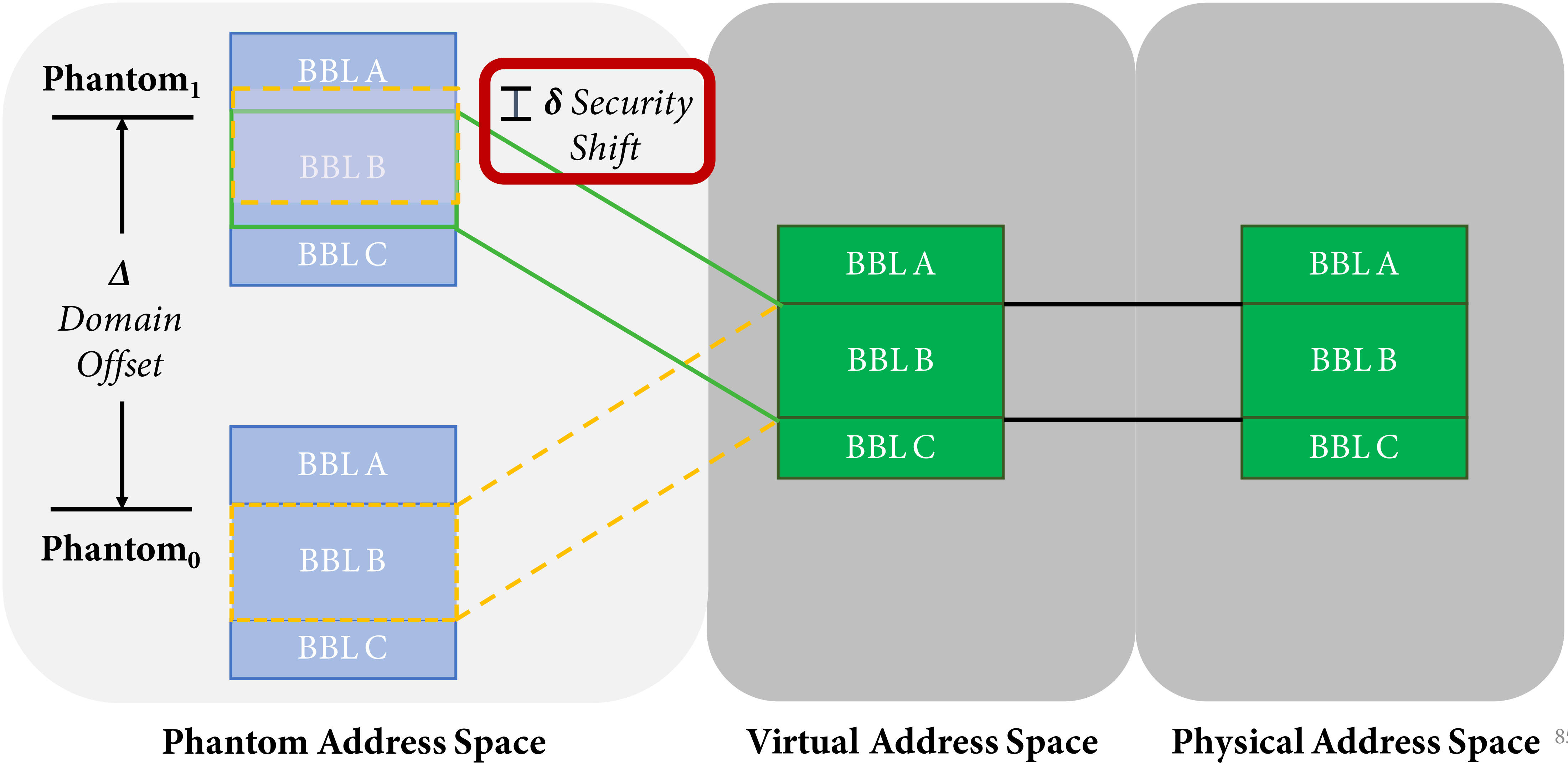
How are phantoms constructed?

Phantoms are logically displaced relative to the original program.



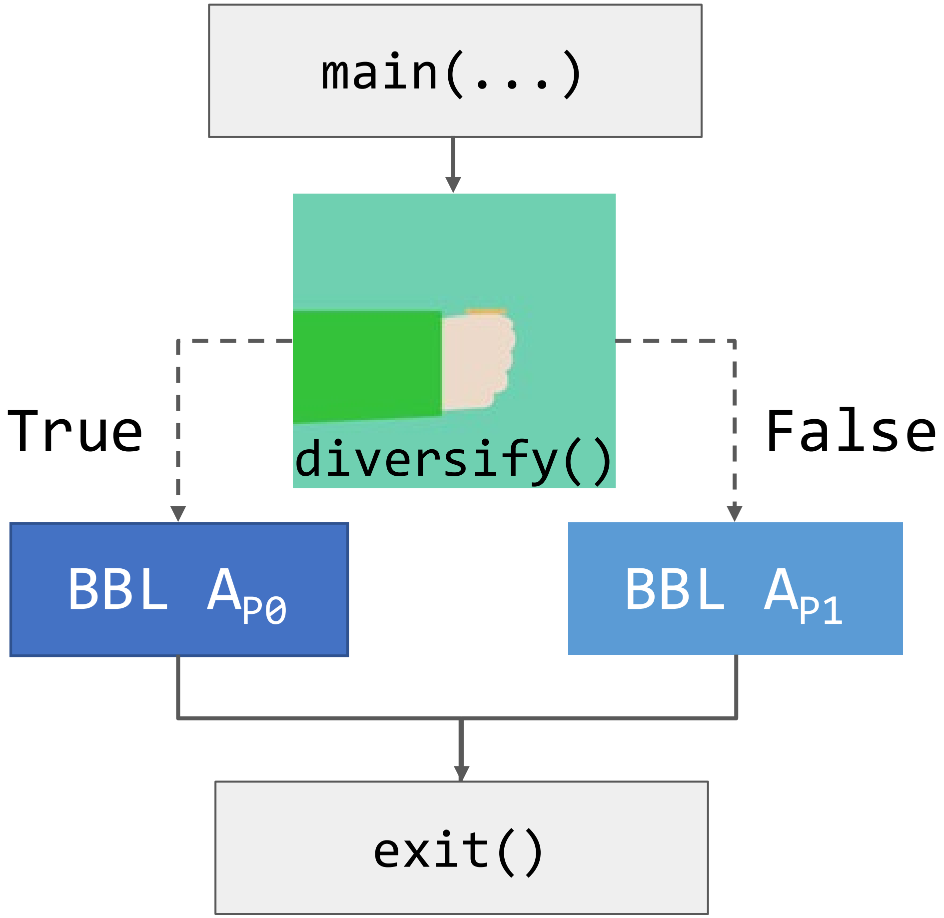
How are phantoms constructed?

Phantoms are logically displaced relative to the original program.



How does PAS diversify execution?

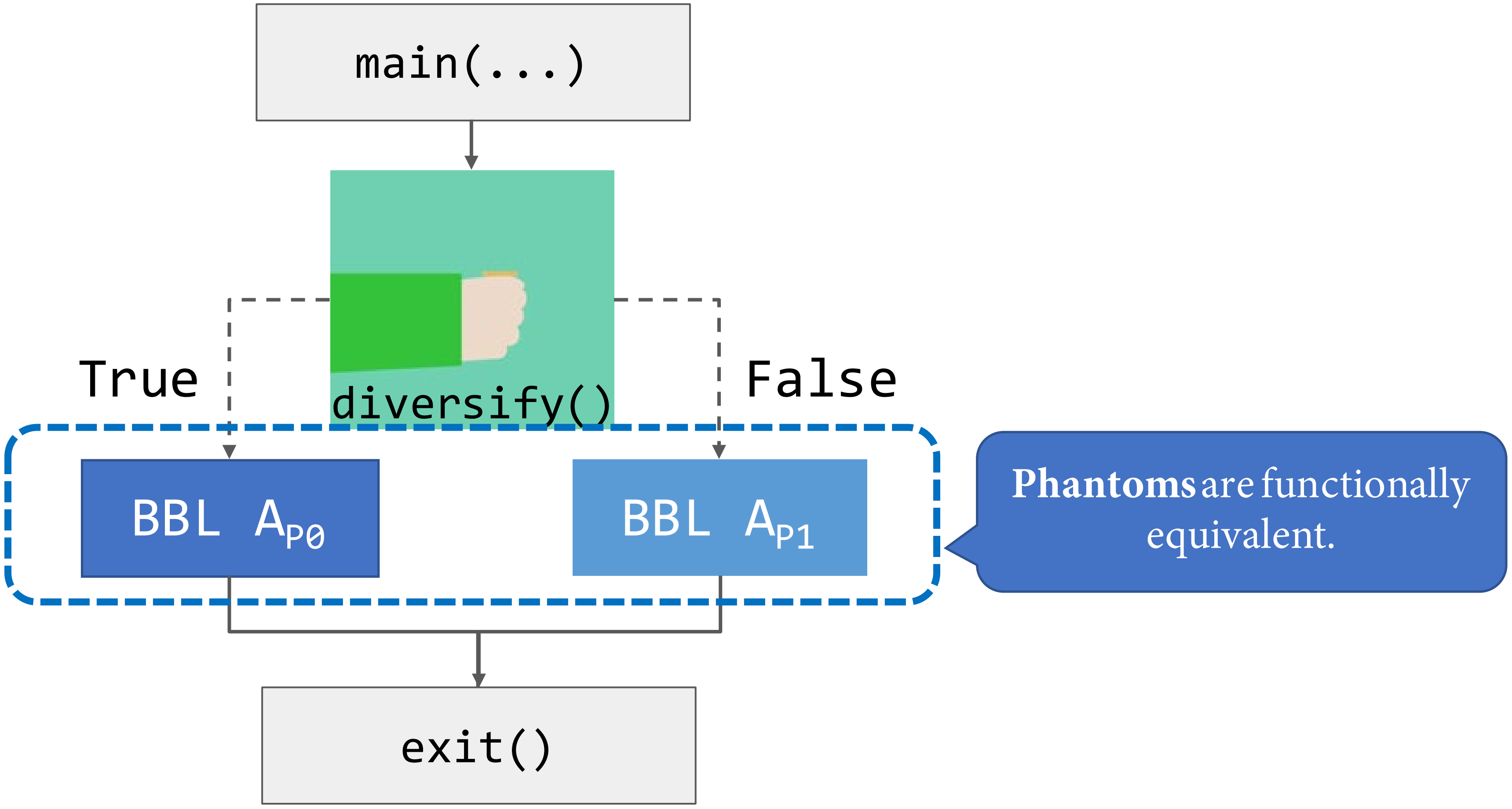
It diversifies the *path* of execution at every basic block.



Program Control Flow Graph

How does PAS diversify execution?

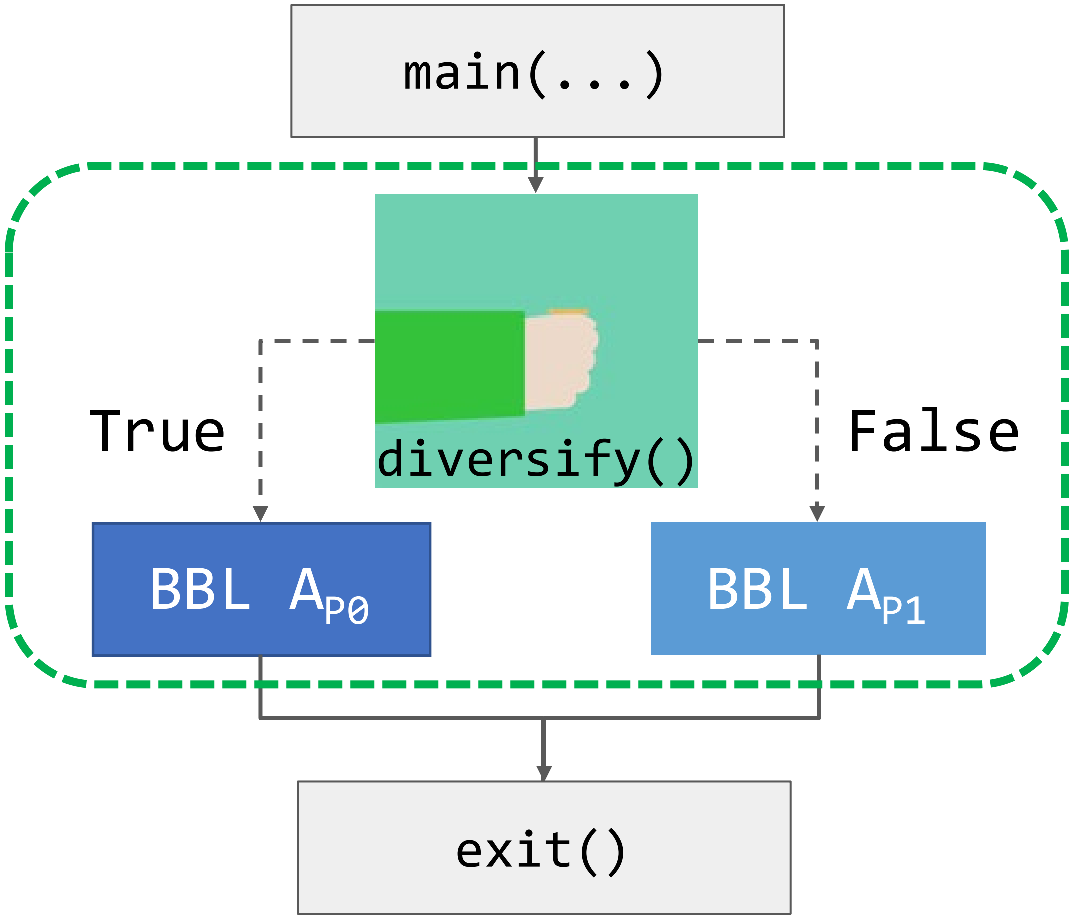
It diversifies the *path* of execution at every basic block.



Program Control Flow Graph

How does PAS diversify execution?

It diversifies the *path* of execution at every basic block.



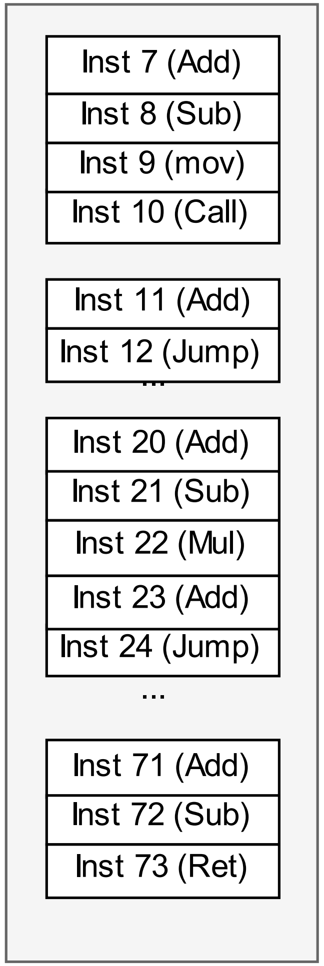
Program Control Flow Graph

Thwarts many Code-Reuse Attacks (CRAs).

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.

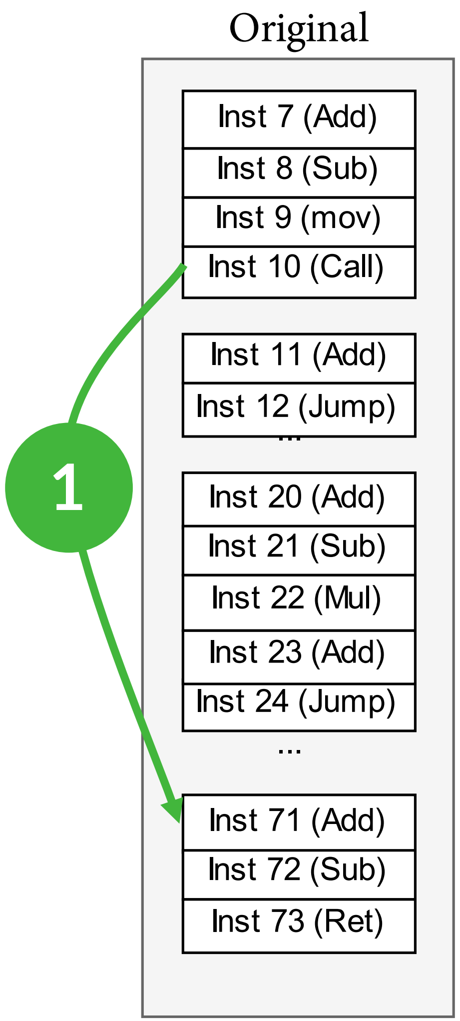
Original



Normal Execution

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.

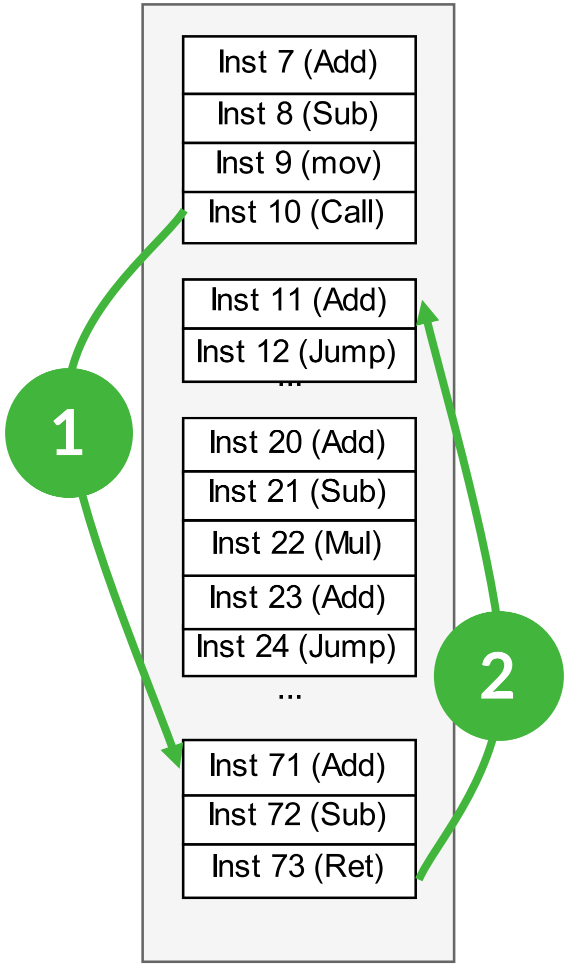


Normal Execution

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.

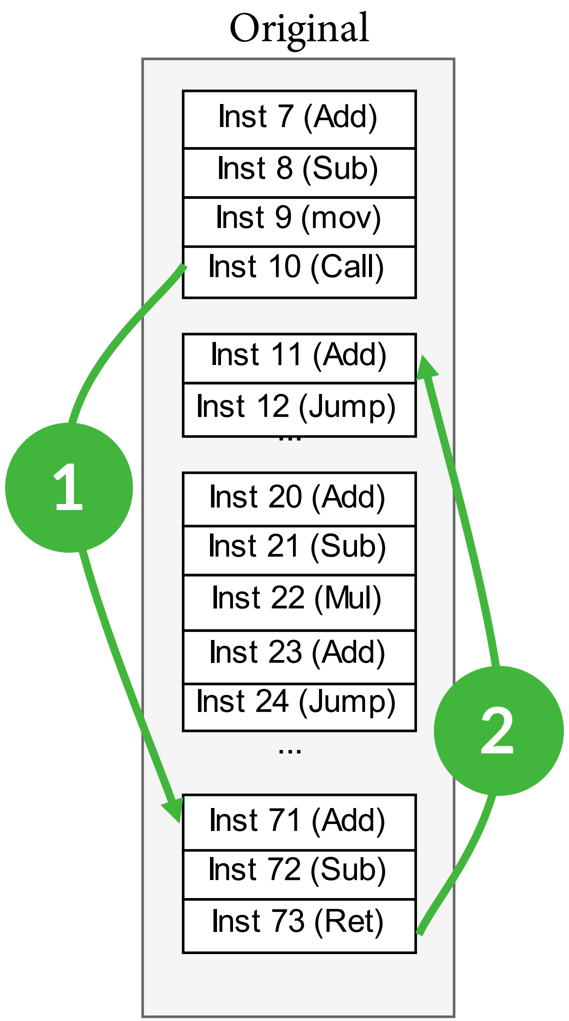
Original



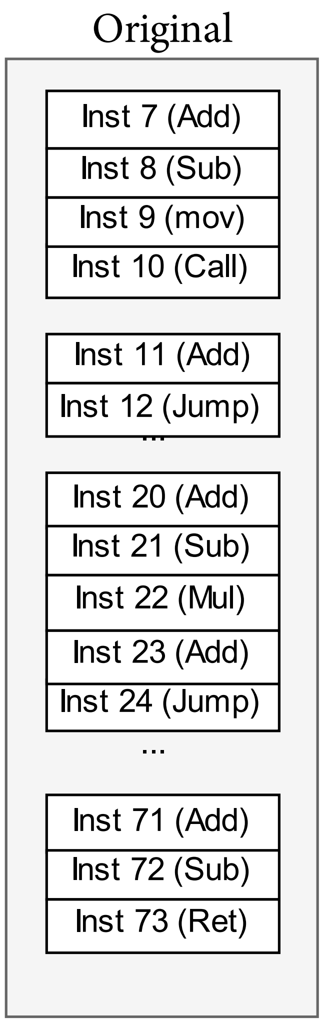
Normal Execution

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



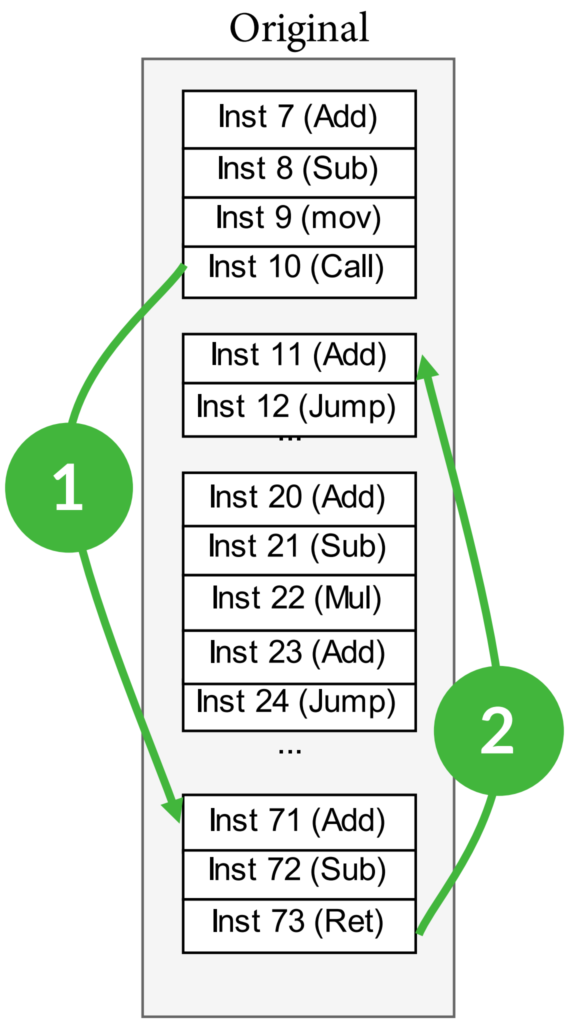
Normal Execution



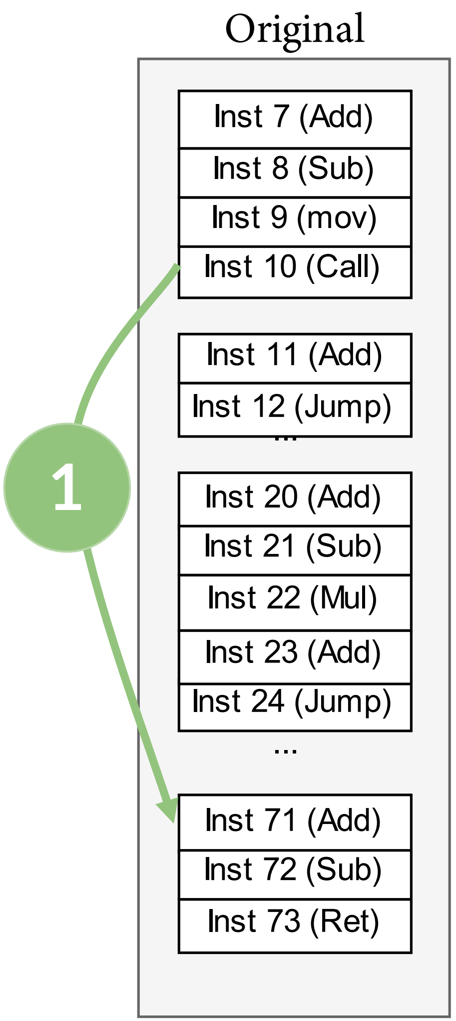
Execution with CRA

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



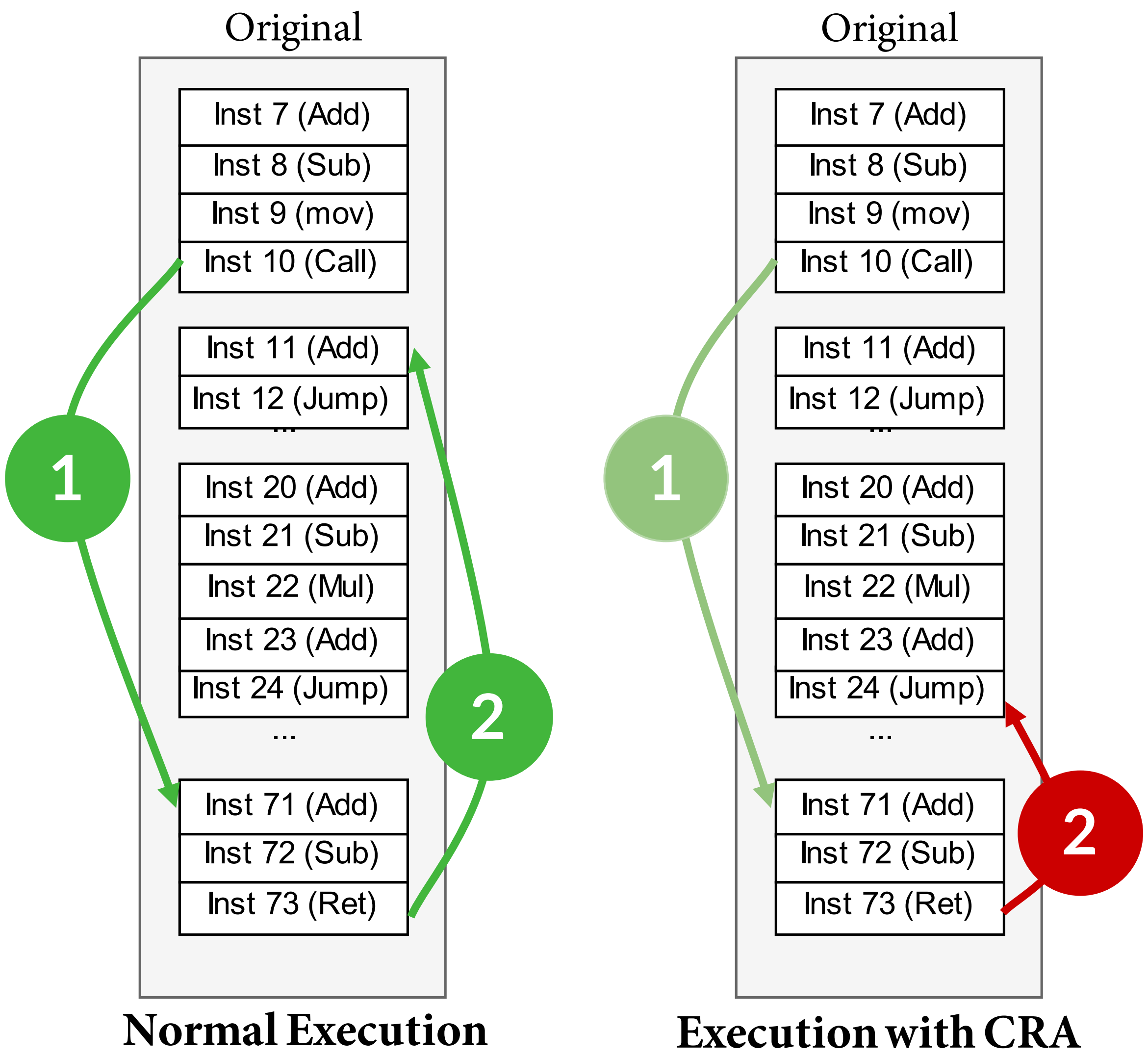
Normal Execution



Execution with CRA

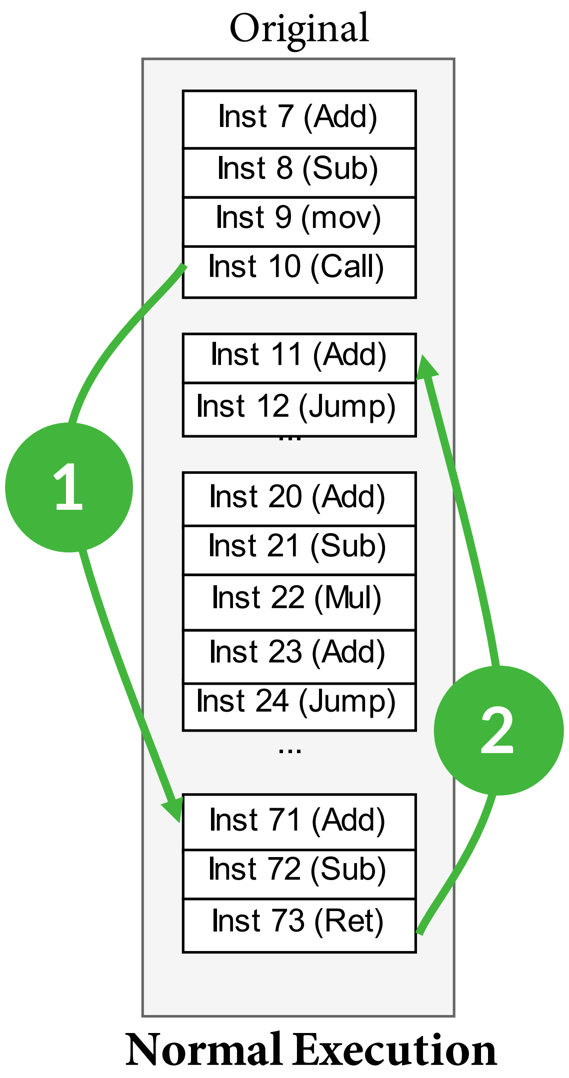
How does PAS protect against CRAs?

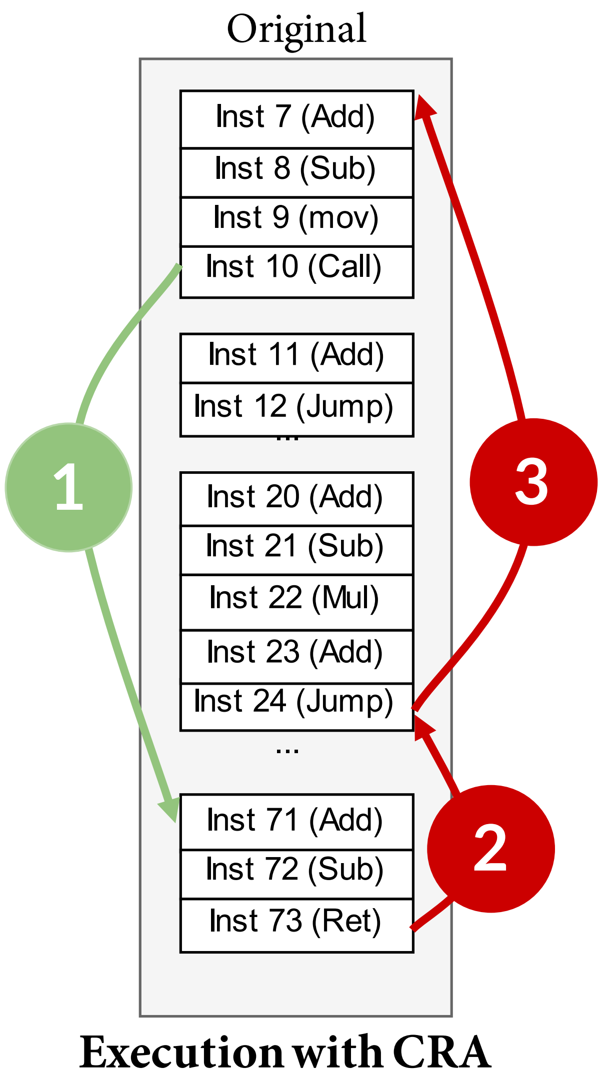
Phantoms force an adversary to guess the execution path.



How does PAS protect against CRAs?

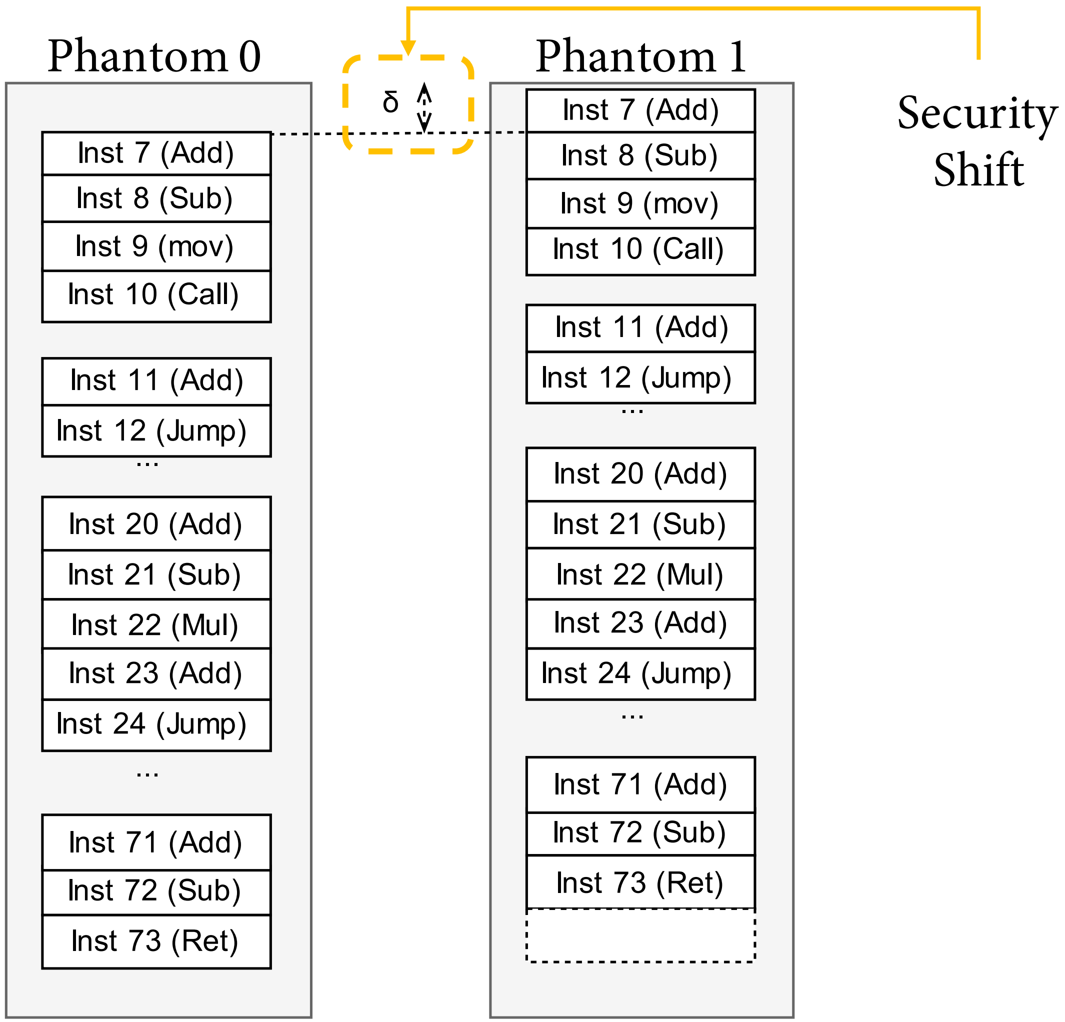
Phantoms force an adversary to guess the execution path.





How does PAS protect against CRAs?

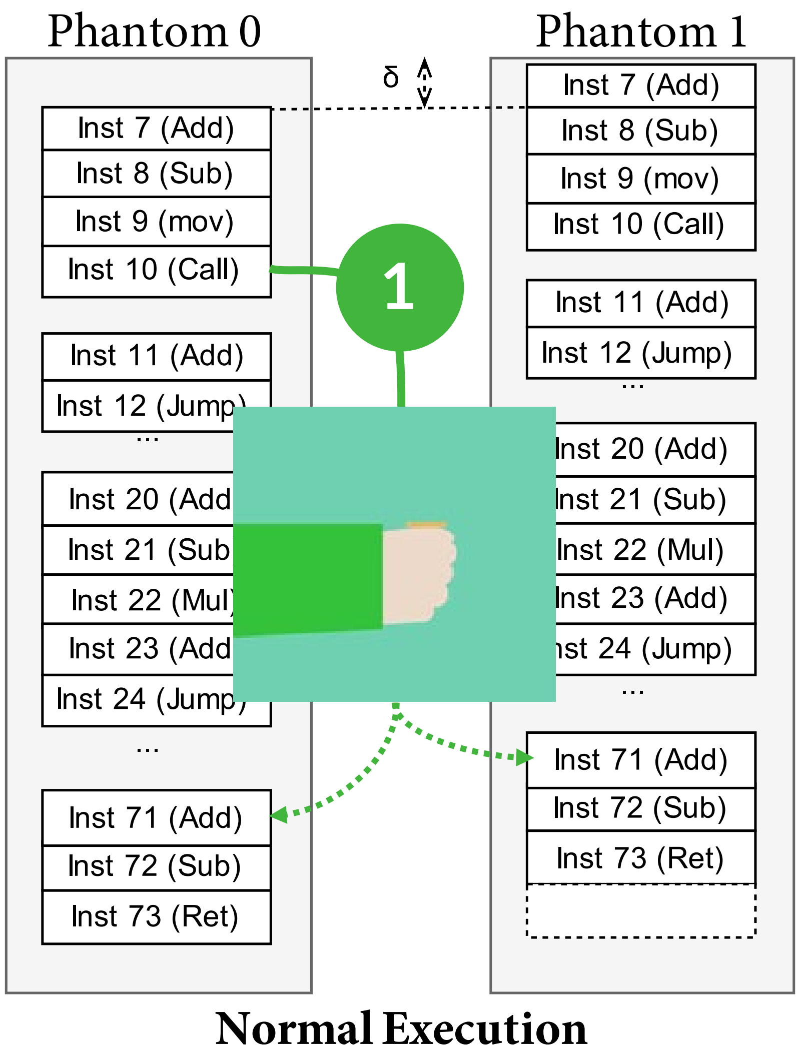
Phantoms force an adversary to guess the execution path.



Normal Execution

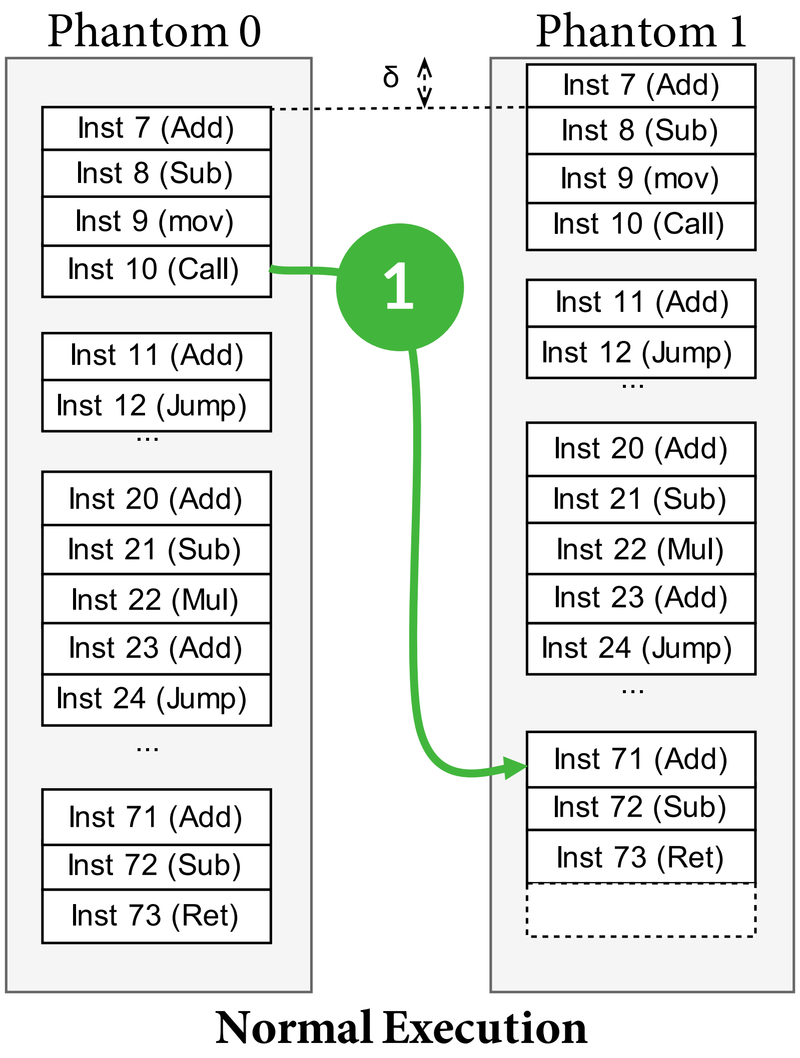
How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



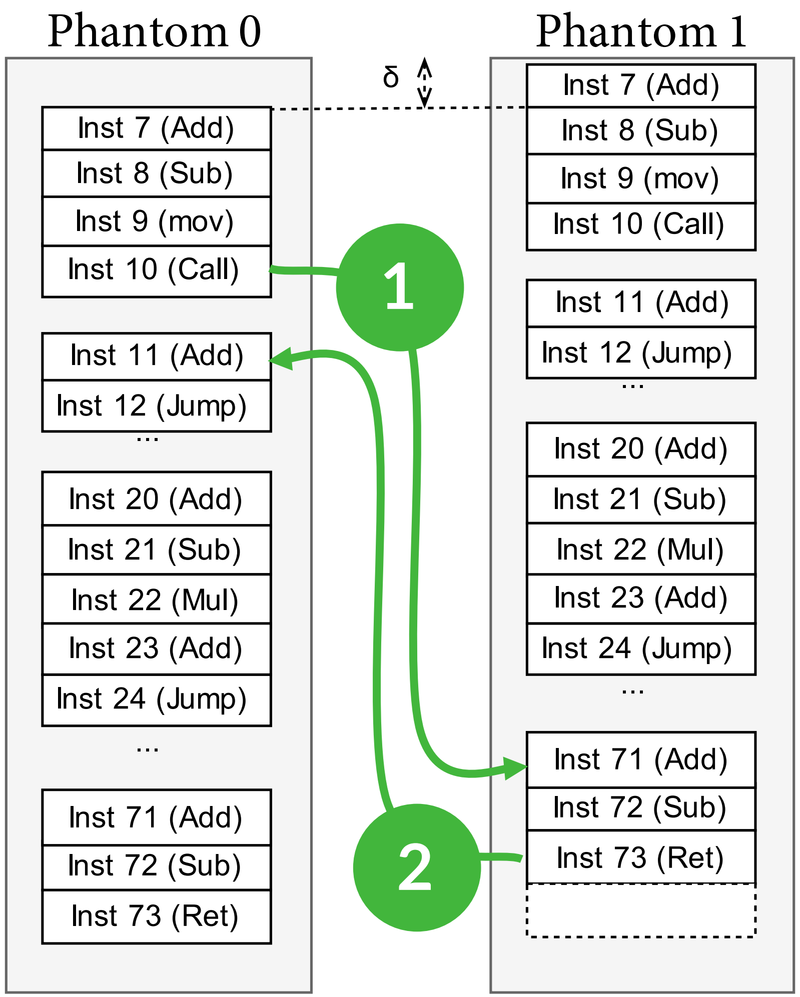
How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



How does PAS protect against CRAs?

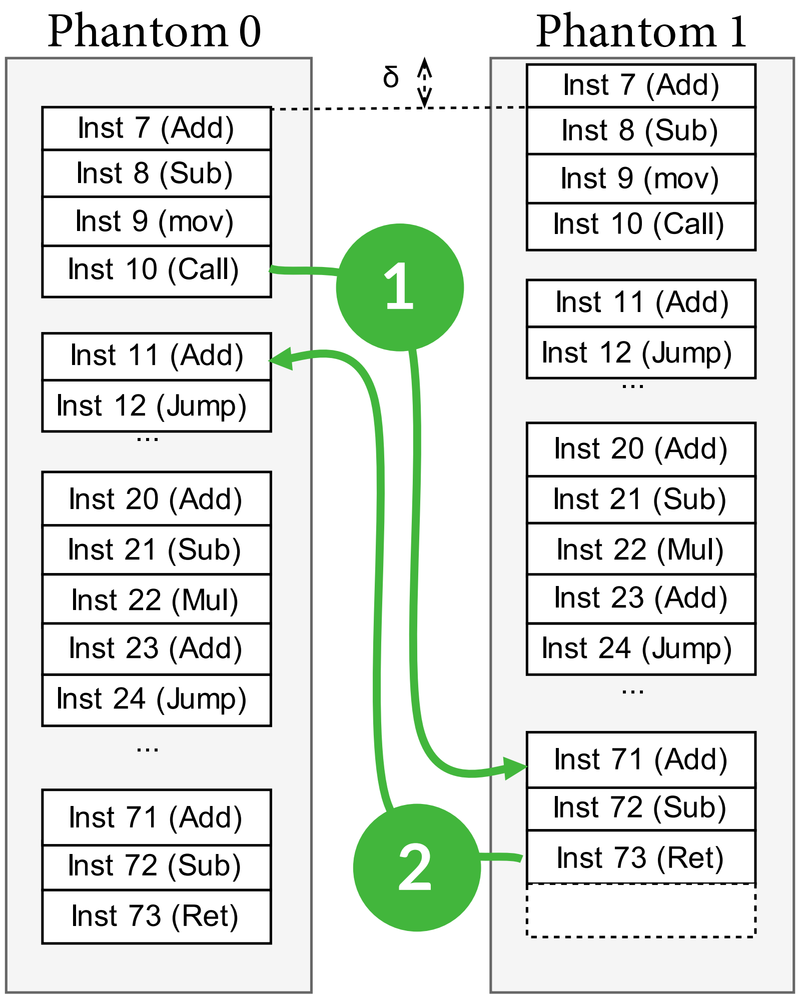
Phantoms force an adversary to guess the execution path.



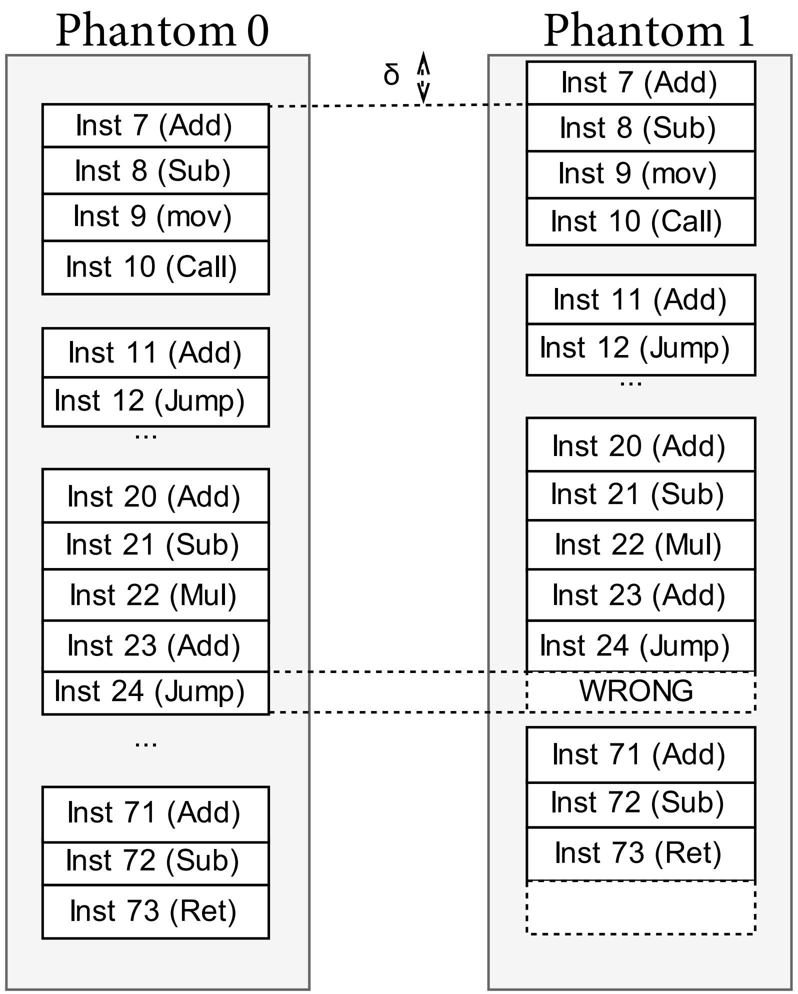
Normal Execution

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



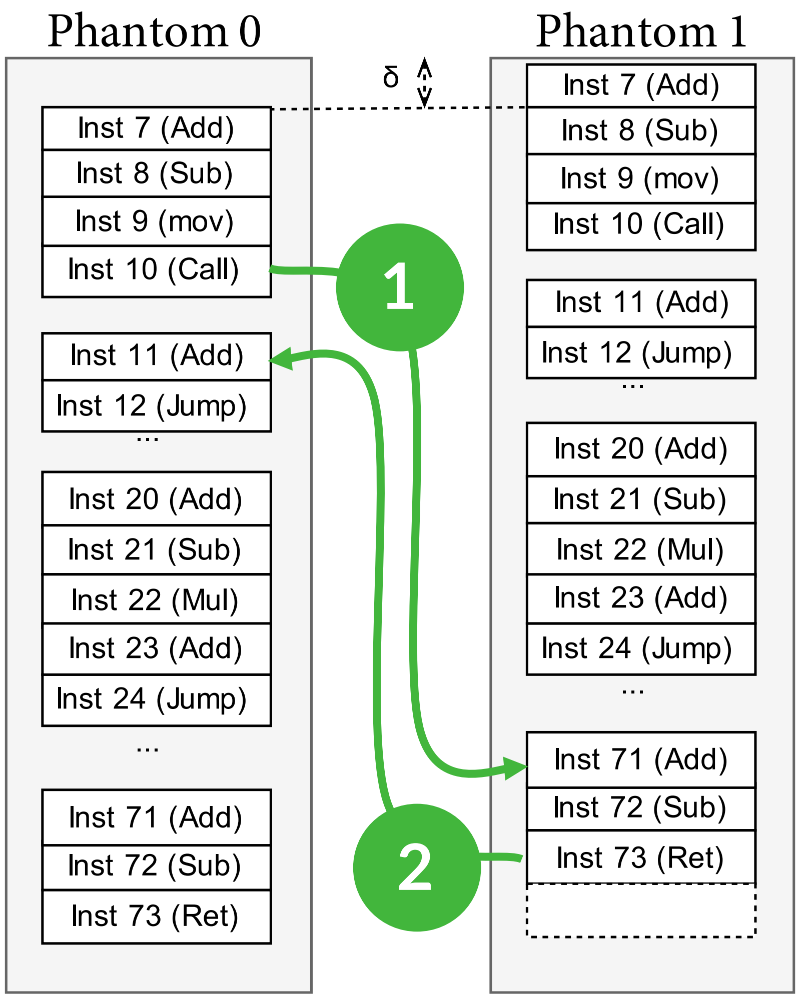
Normal Execution



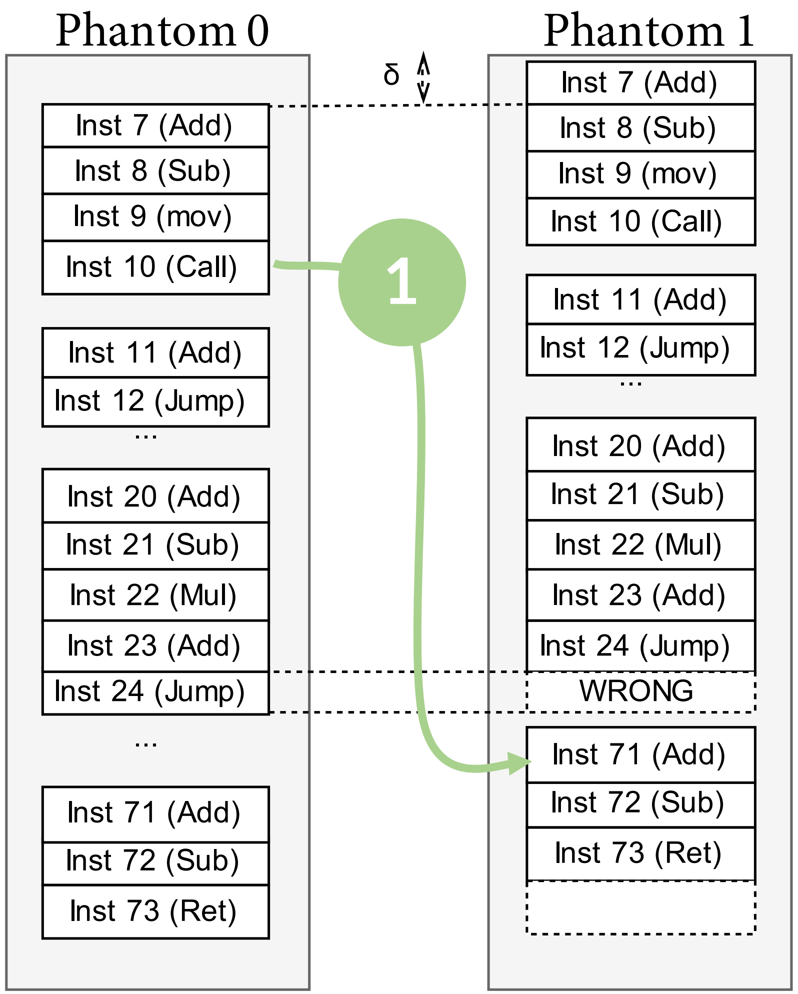
Execution with CRA

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



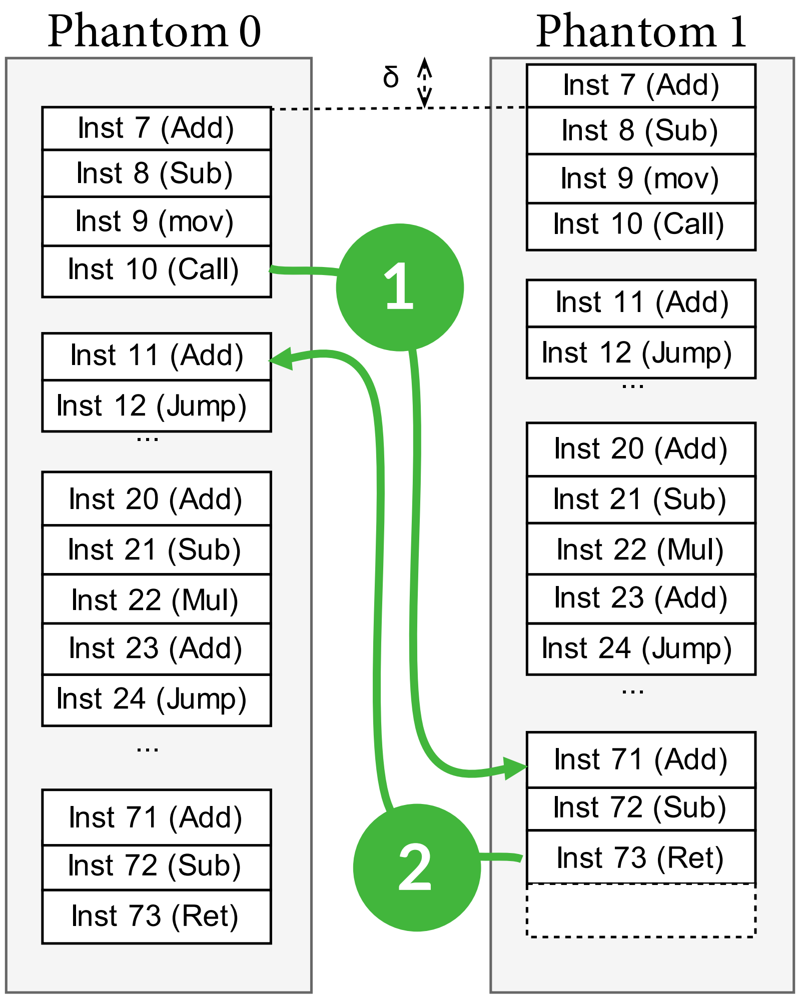
Normal Execution



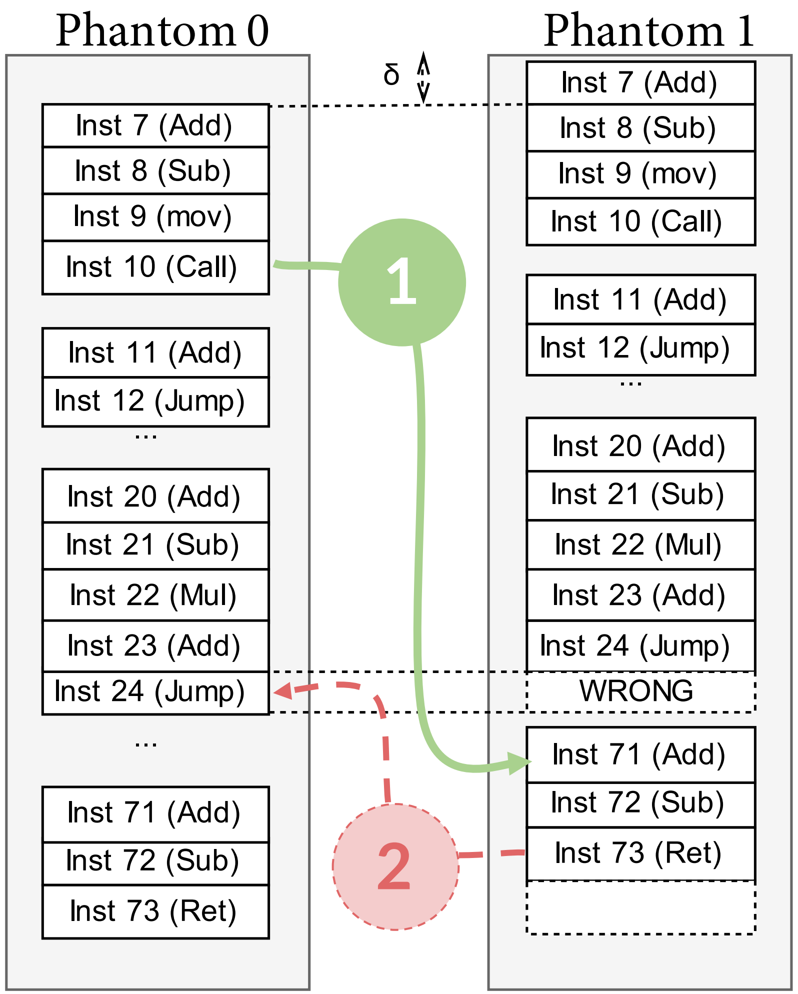
Execution with CRA

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



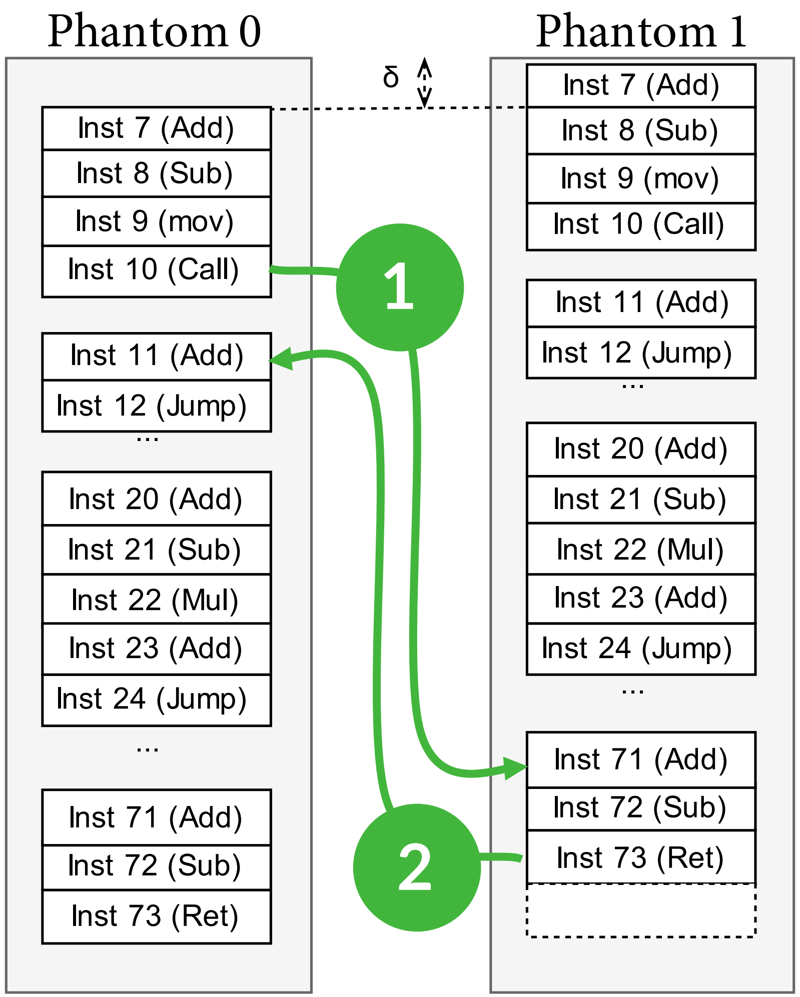
Normal Execution



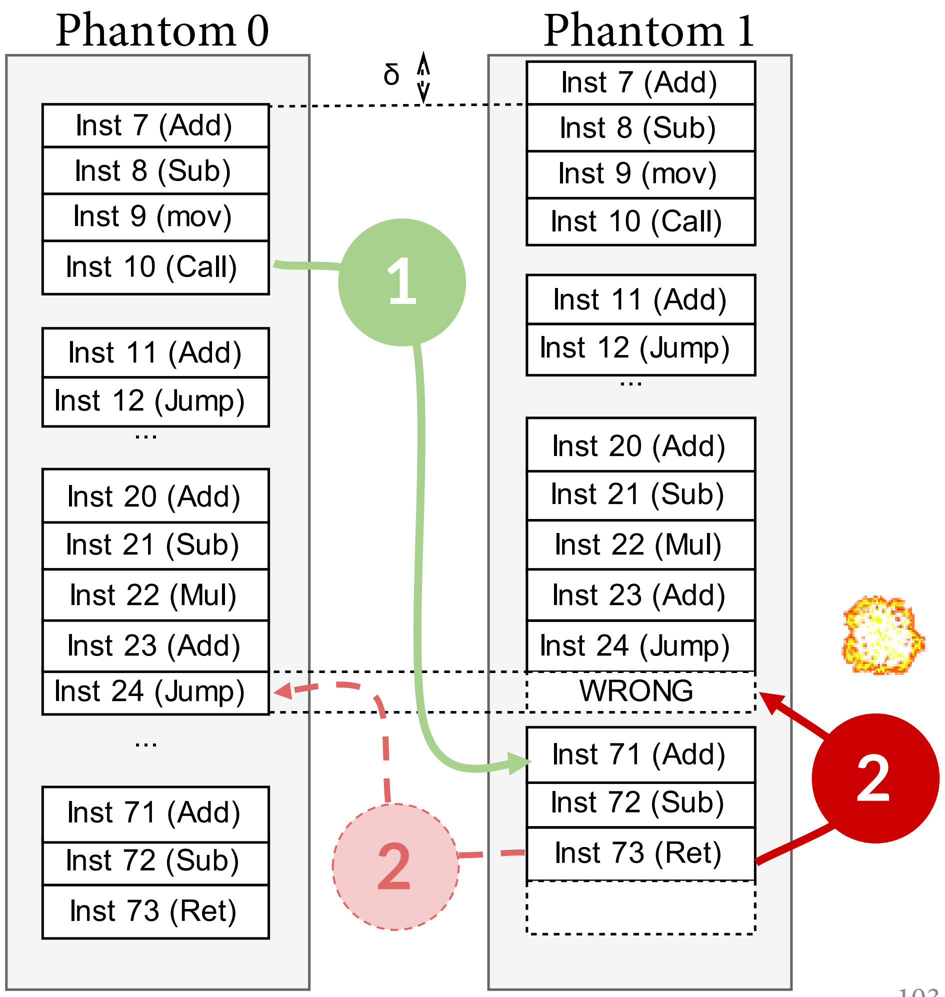
Execution with CRA

How does PAS protect against CRAs?

Phantoms force an adversary to guess the execution path.



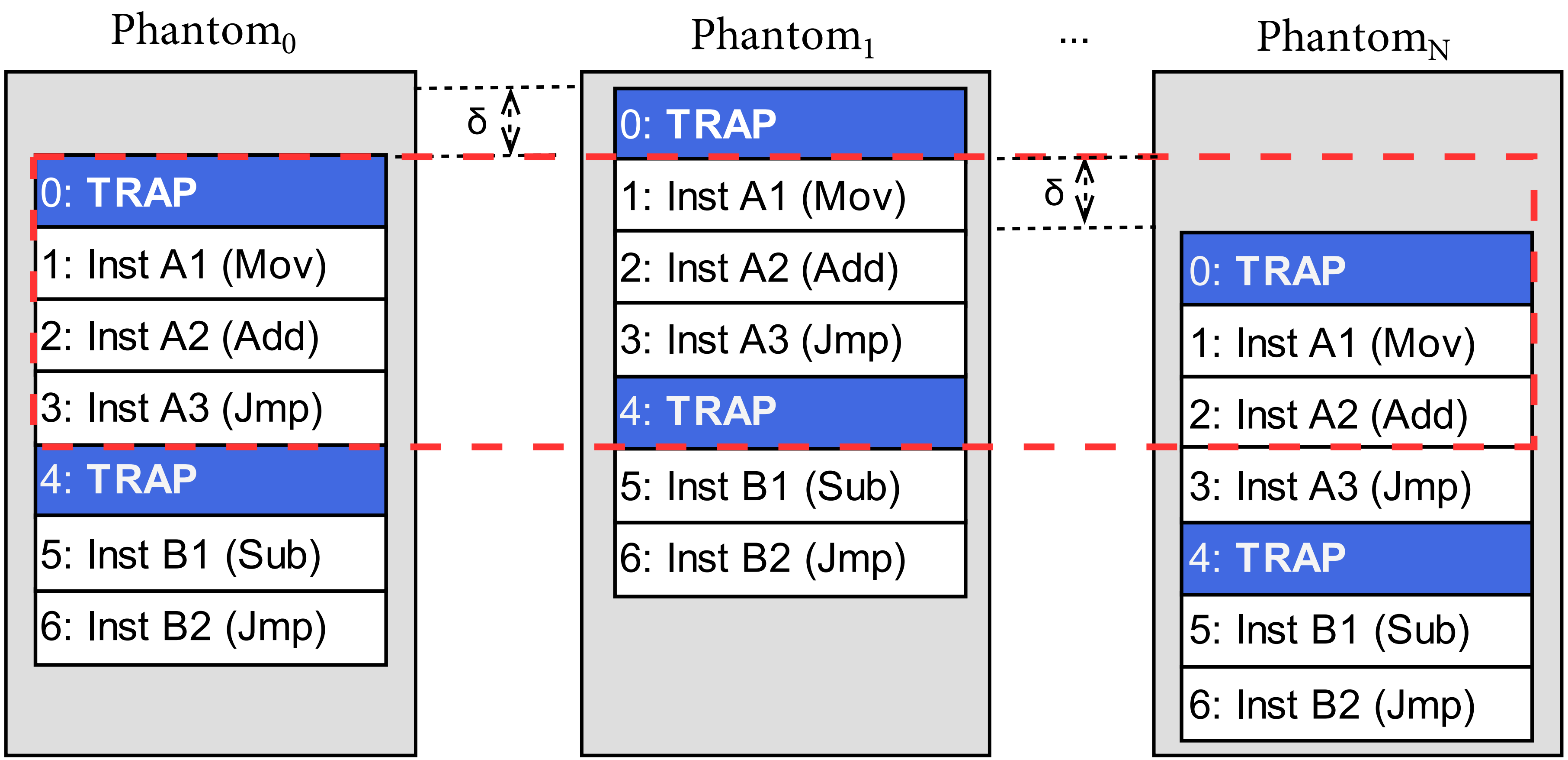
Normal Execution



Execution with CRA

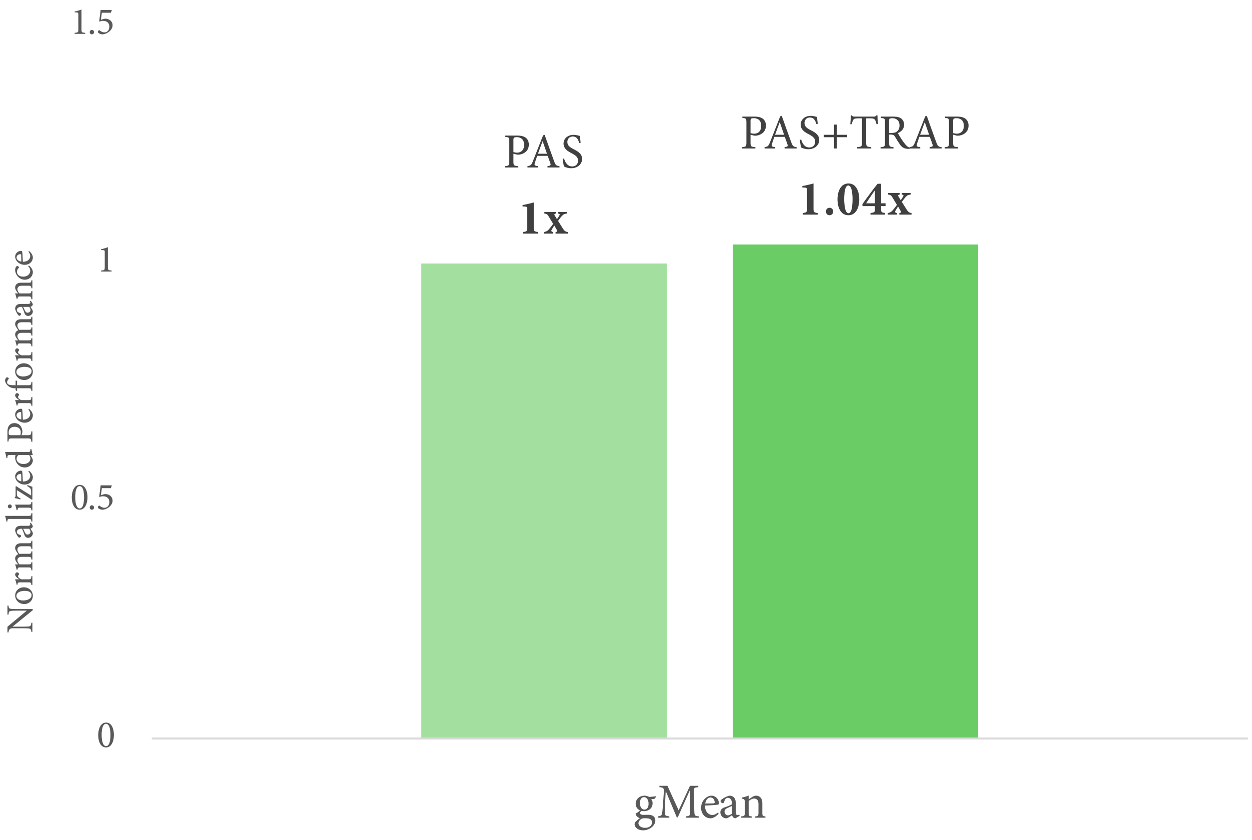
How does PAS precisely trap an attacker?

Code is instrumented with special instructions to throw an exception.



How was PAS evaluated?

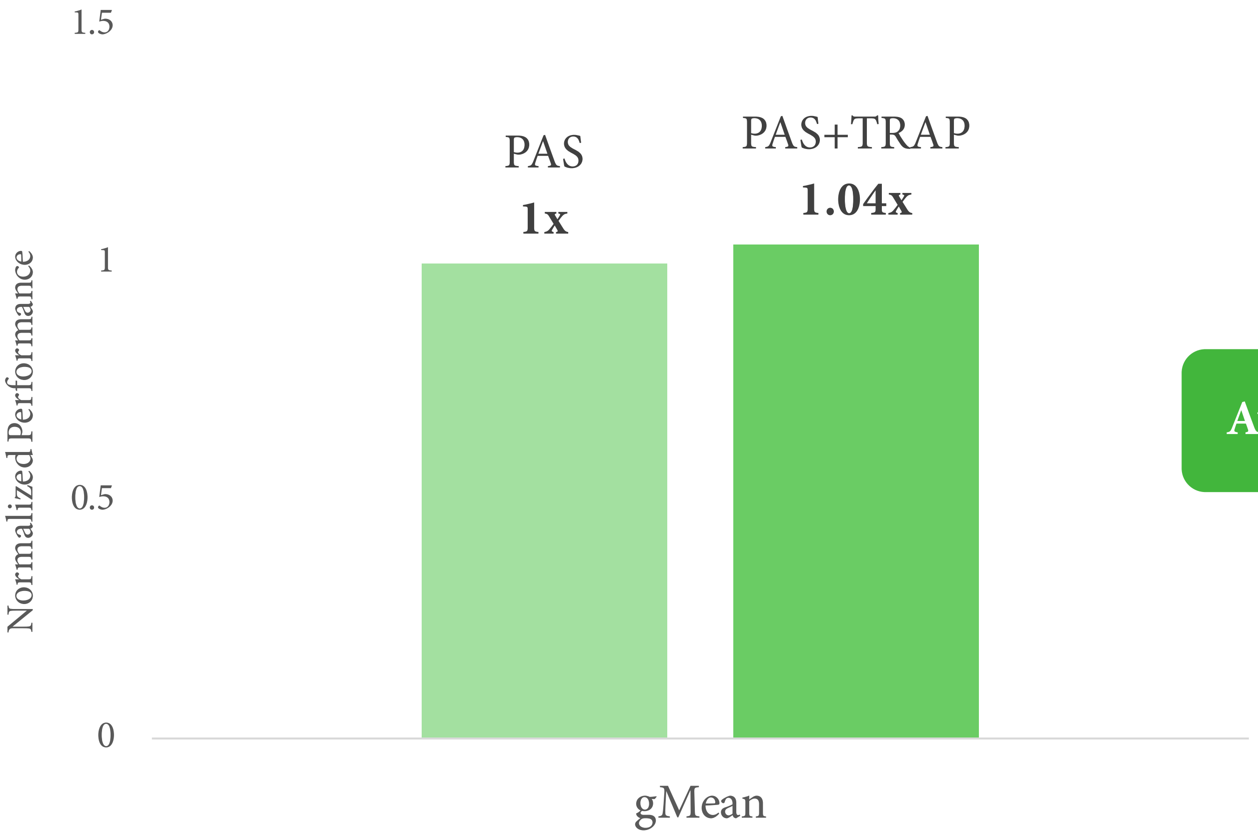
We used the gem5 architectural simulator to validate correctness & performance.



SPEC CPU2017

How was PAS evaluated?

We used the gem5 architectural simulator to validate correctness & performance.



SPEC CPU2017

Average slowdown < 4%!

Why is PAS well suited for constrained devices?

It brings efficient N-variant execution protection with minimal cost.

Minimal Performance Impact

PAS has minimal impact on workload execution.

Memory Savings

PAS cuts down on resource duplication associated with other N-variant execution approaches.

My contributions to CPS security

An overview of publications

1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

CALIFORMS
Cache Line Formats

Appears as

Practical Byte-Granular Memory Blacklisting using Califorms
Sasaki, H., Arroyo, M., Tarek Ibn Ziad, M., Koustubha, B., Sinha, K., Sethumadhavan, S.
IEEE/ACM International Symposium on Microarchitecture (MICRO) 2019
(DOI: 10.1145/3352460.3358299)

IEEE 2019 Micro Top Picks Honorable Mention

Patent
US16744922

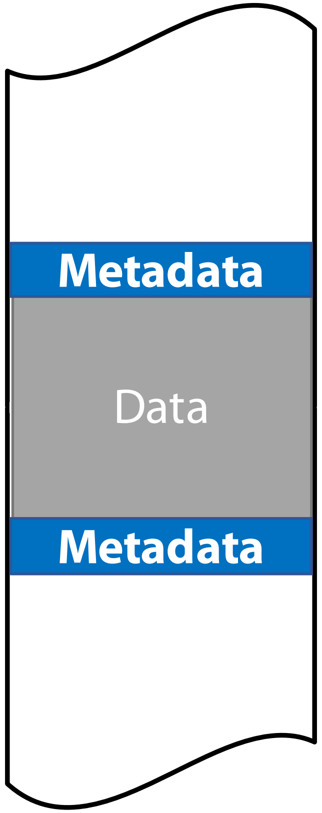
CALIFORMS in a nutshell

A hardware primitive to encode metadata within program data.

Inline

CALIFORMS in a nutshell

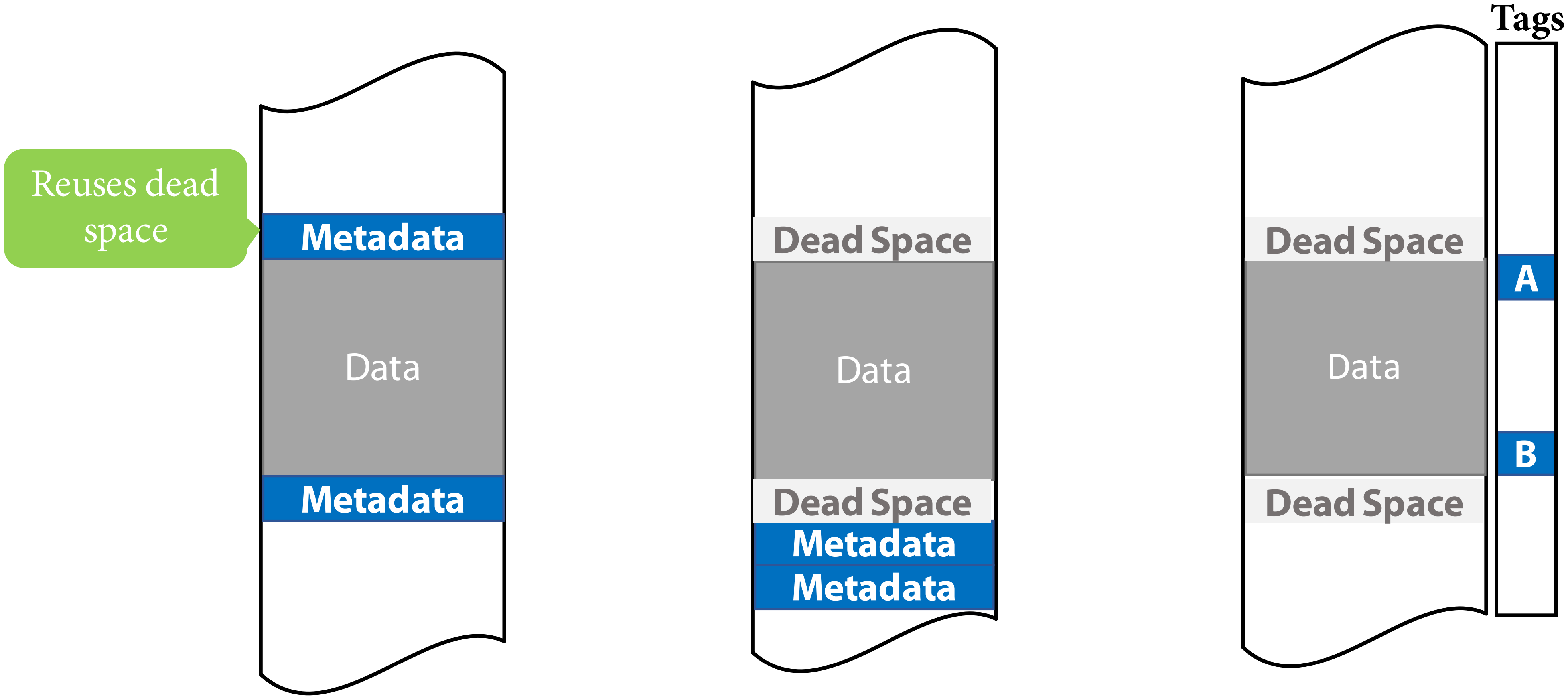
A hardware primitive to encode metadata within program data.



Inline

Why is CALIFORMS' inline metadata useful?

It consumes less memory and requires less memory accesses.



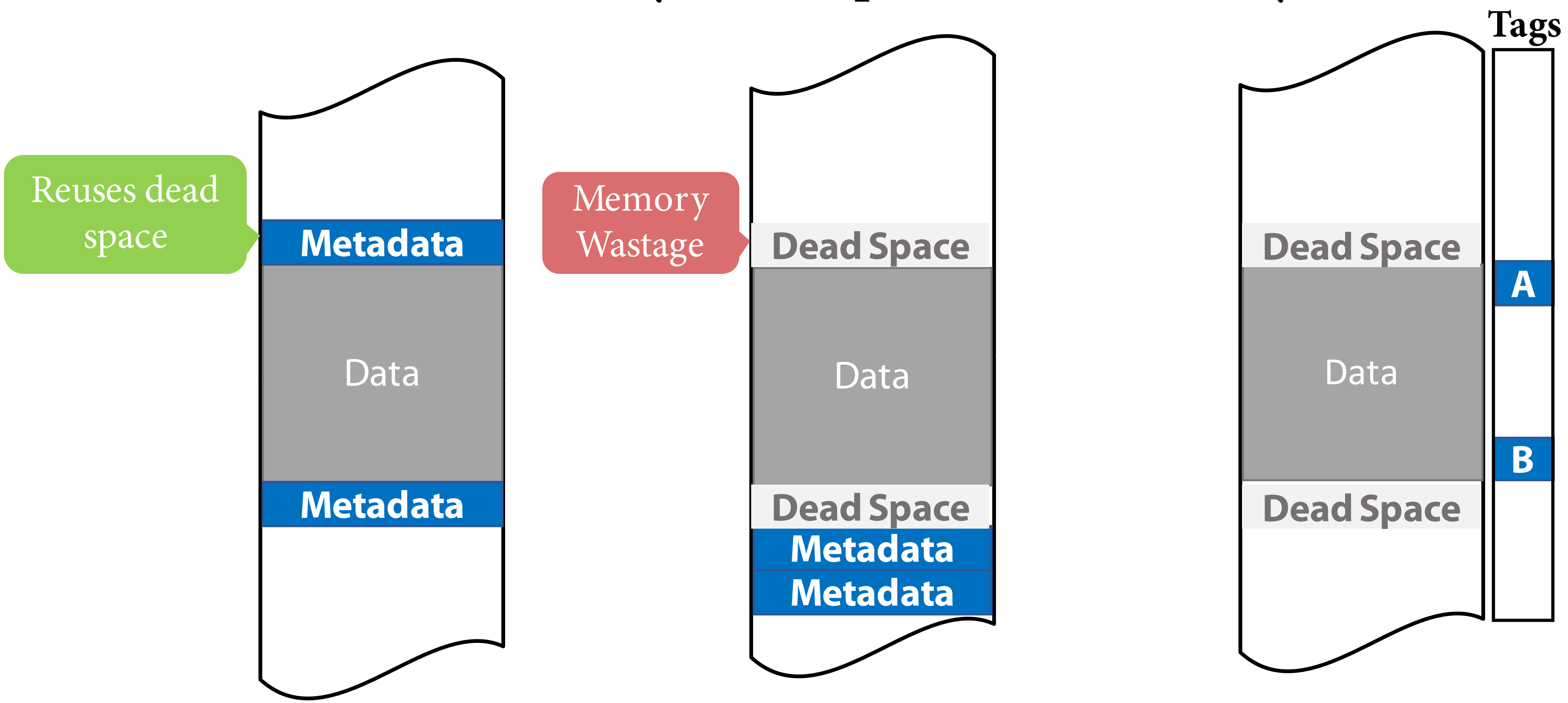
Inline
Metadata is accessed with the data.

Disjoint
Metadata is accessed separately from data.

Co-joined
Memory is extended with metadata attributes.

Why is CALIFORMS' inline metadata useful?

It consumes less memory and requires less memory accesses.



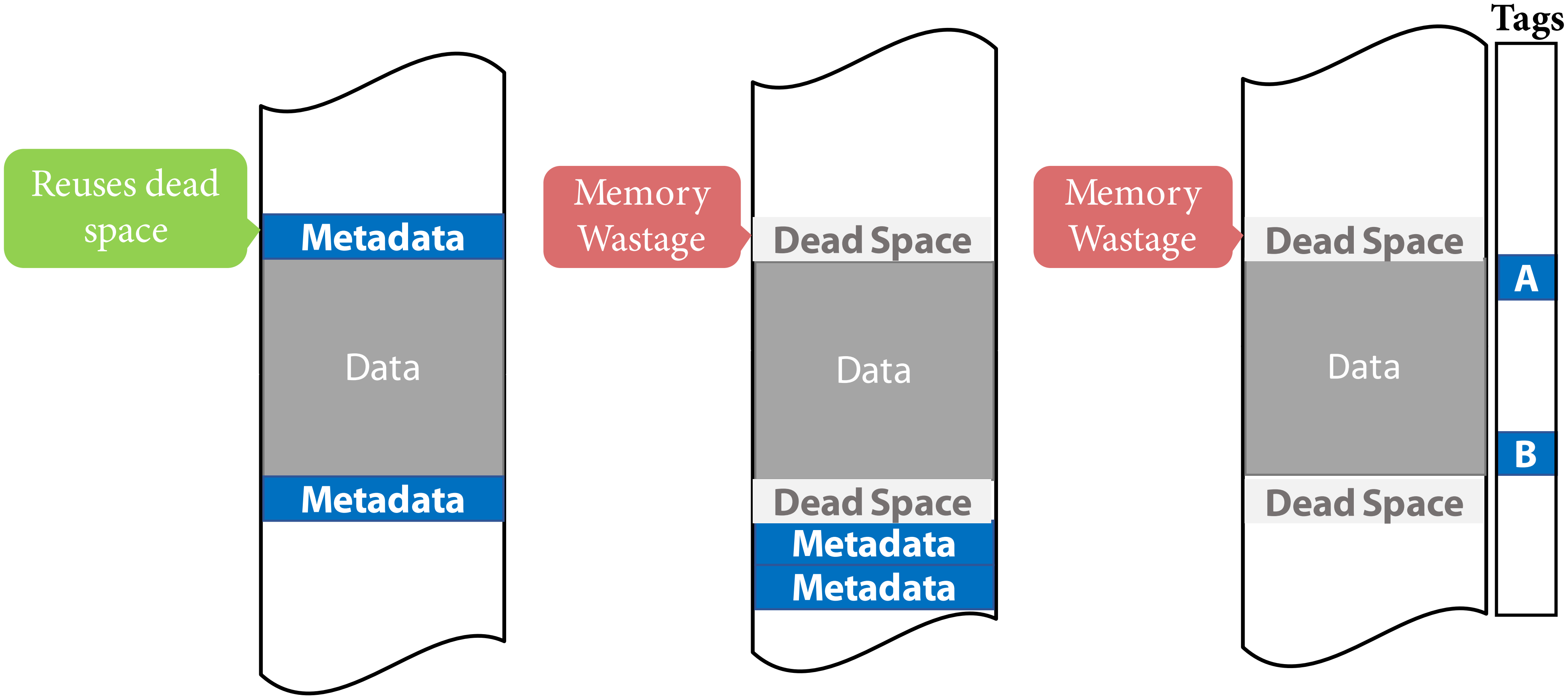
Inline
Metadata is accessed with the data.

Disjoint
Metadata is accessed separately from data.

Co-joined
Memory is extended with metadata attributes.

Why is CALIFORMS' inline metadata useful?

It consumes less memory and requires less memory accesses.



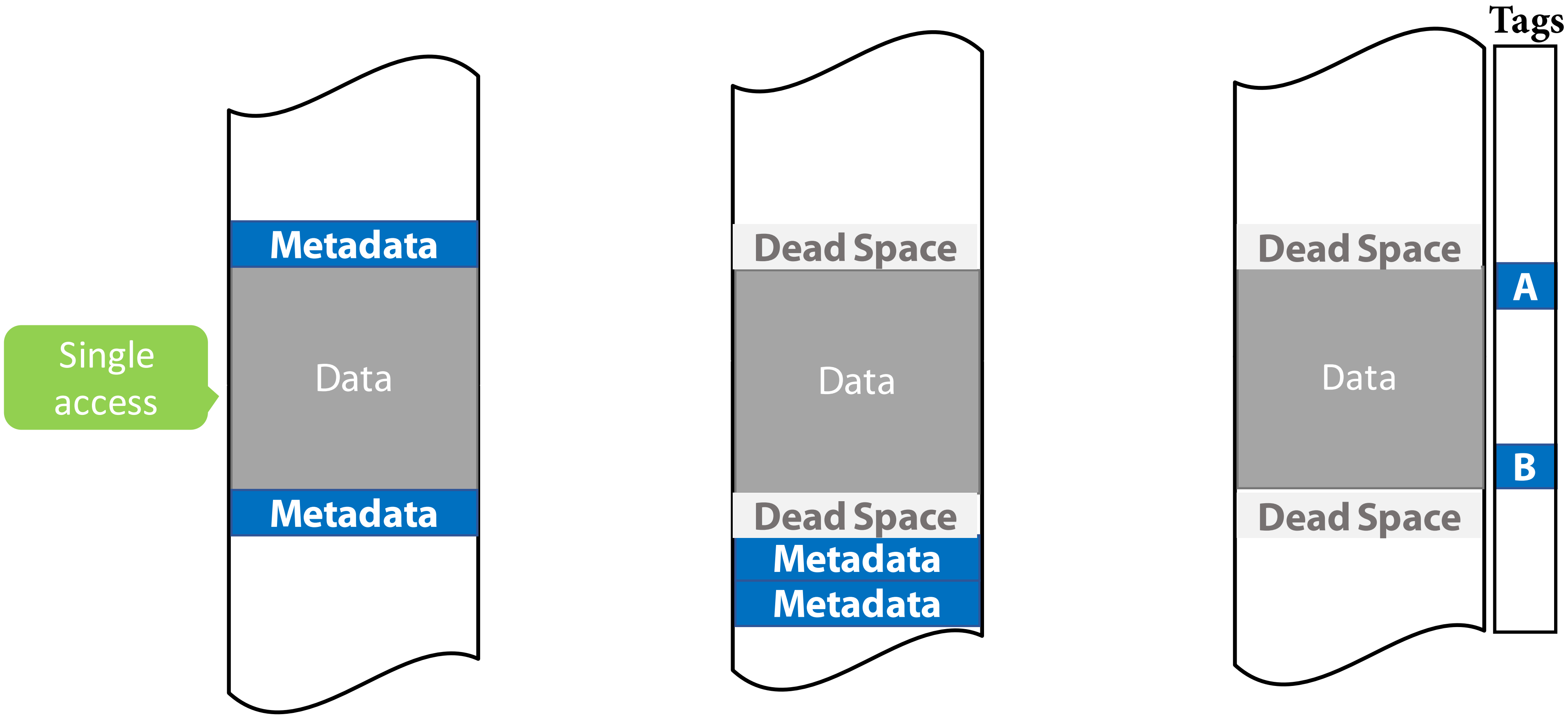
Inline
Metadata is accessed with the data.

Disjoint
Metadata is accessed separately from data.

Co-joined
Memory is extended with metadata attributes.

Why is CALIFORMS' inline metadata useful?

It consumes less memory and requires less memory accesses.



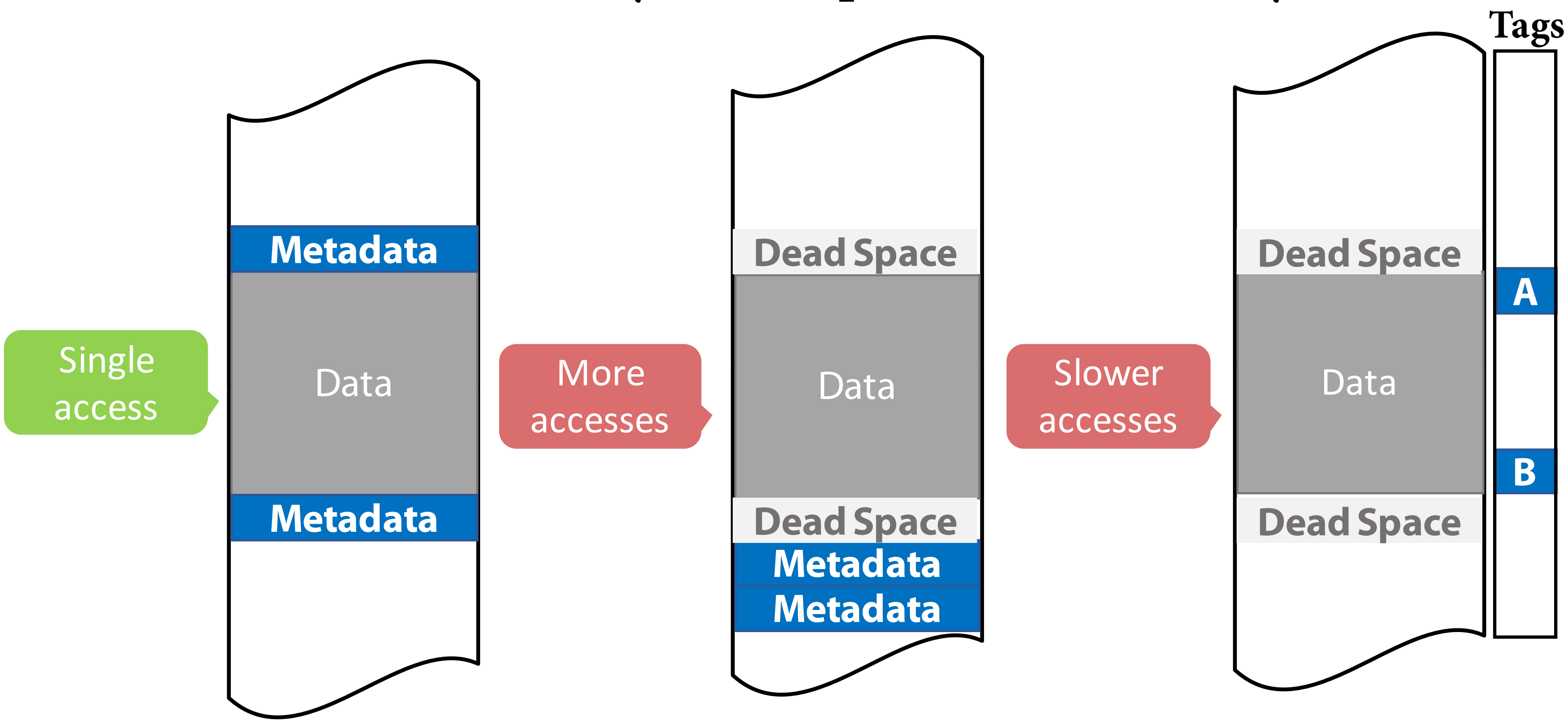
Inline
Metadata is accessed with the data.

Disjoint
Metadata is accessed separately from data.

Co-joined
Memory is extended with metadata attributes.

Why is CALIFORMS' inline metadata useful?

It consumes less memory and requires less memory accesses.



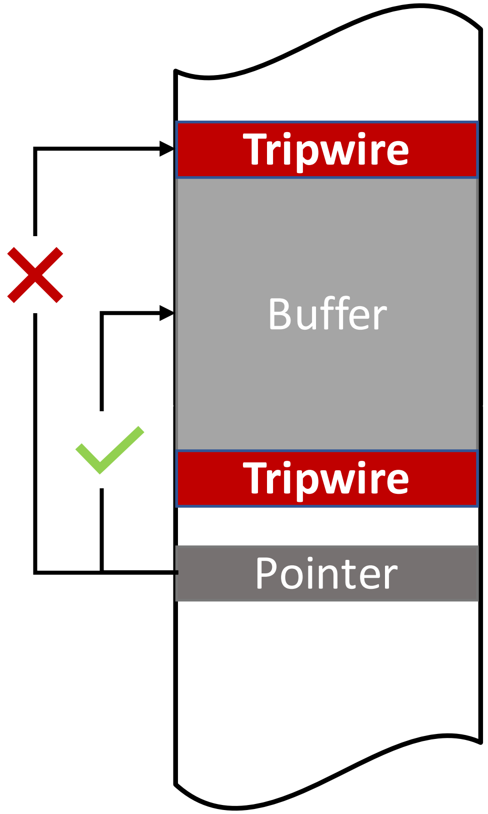
Inline
Metadata is accessed with the data.

Disjoint
Metadata is accessed separately from data.

Co-joined
Memory is extended with metadata attributes.

How is CALIFORMS used for security?

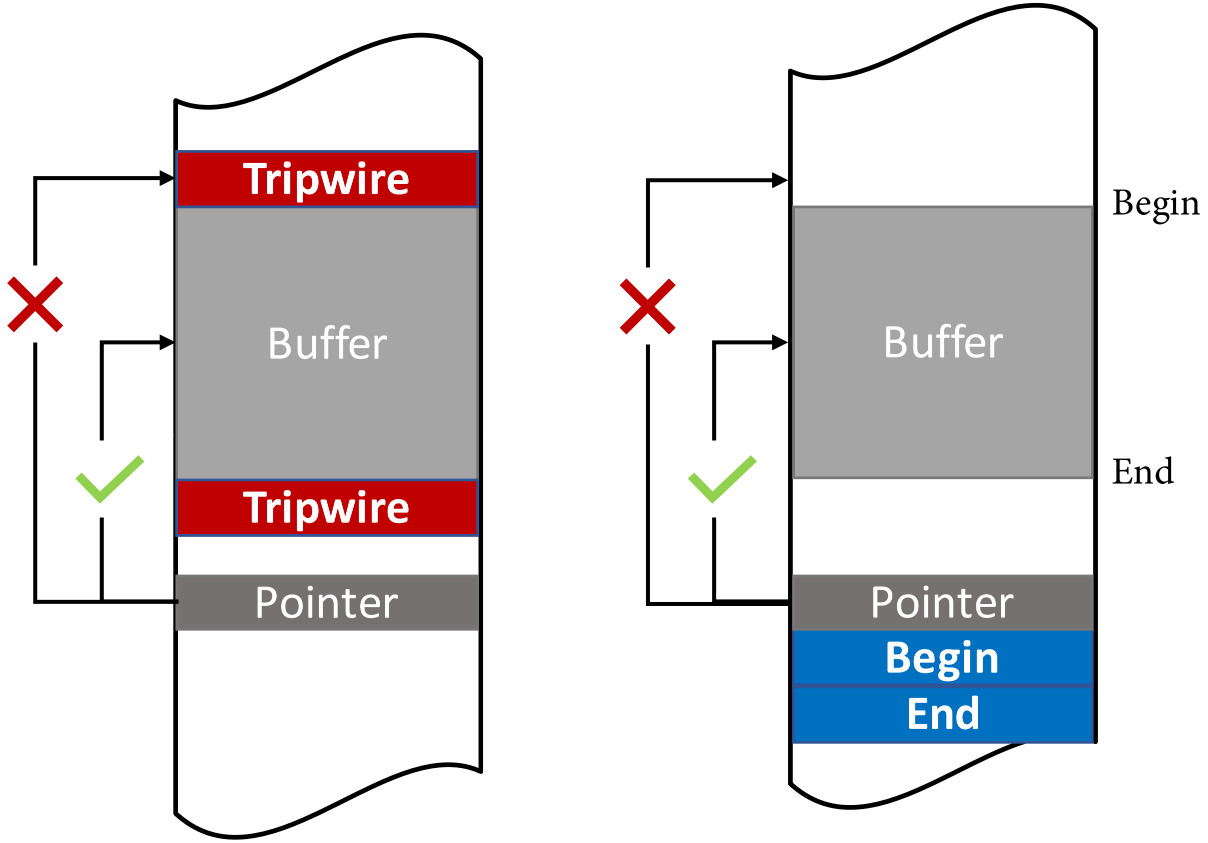
It enables an efficient memory access control mechanism.



Inlined Metadata Blacklisting

How is CALIFORMS used for security?

It enables an efficient memory access control mechanism.

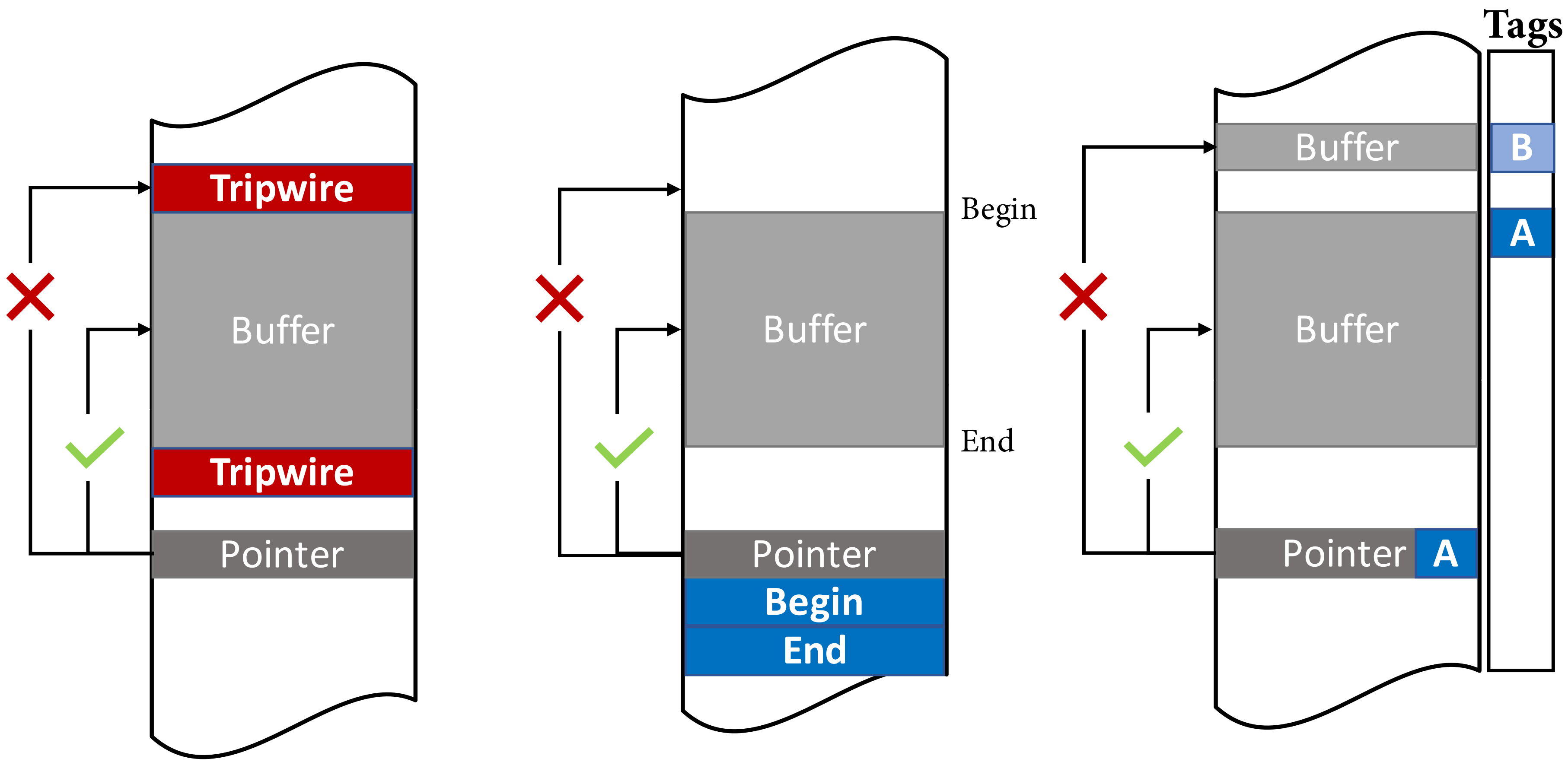


Inlined Metadata Blacklisting

Disjoint Metadata Whitelisting

How is CALIFORMS used for security?

It enables an efficient memory access control mechanism.



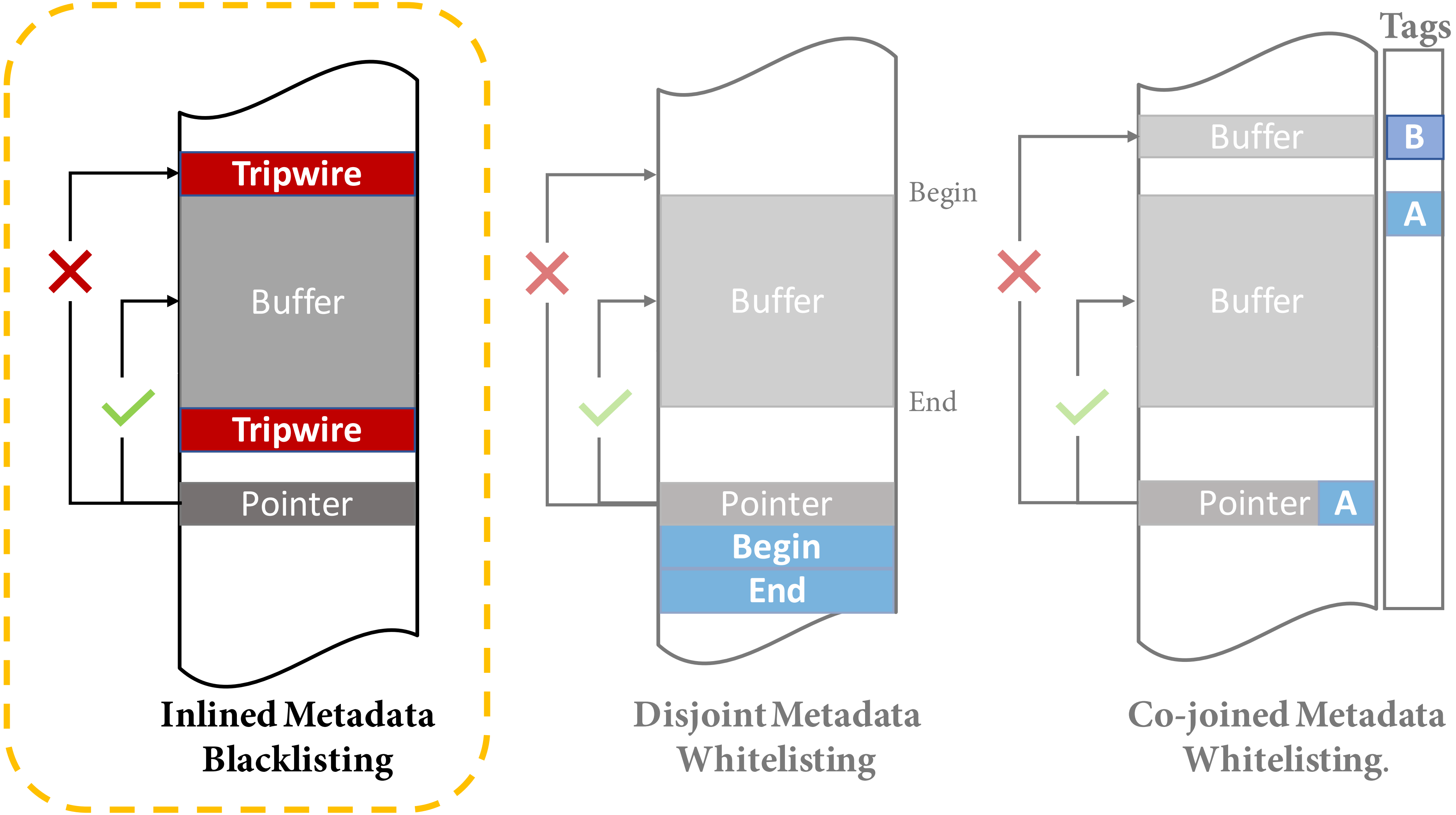
Inlined Metadata Blacklisting

Disjoint Metadata Whitelisting

Co-joined Metadata Whitelisting.

How is CALIFORMS used for security?

It enables an efficient memory access control mechanism.



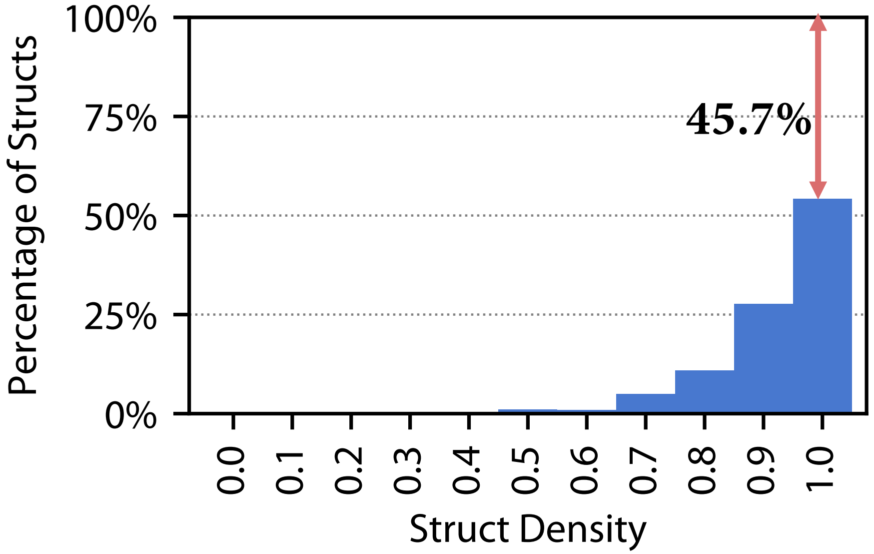
What key insight does CALIFORMS make?

Program data naturally contains inaccessible data (i.e. dead bytes).

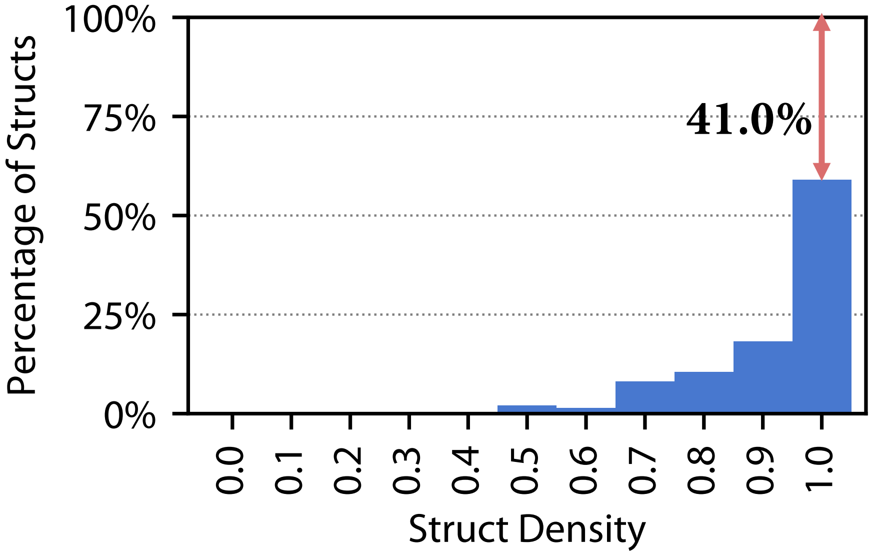
```
struct A {
    char c;
    /* compiler inserts padding
     * bytes for alignment */
    // ie. char dead_bytes[3];
    int i;
    char buf[64];
    void (*fp)();
};
```

How prevalent are dead bytes?

Over 40% of structs have at least one dead byte.



SPEC2006 C/C++ Benchmarks

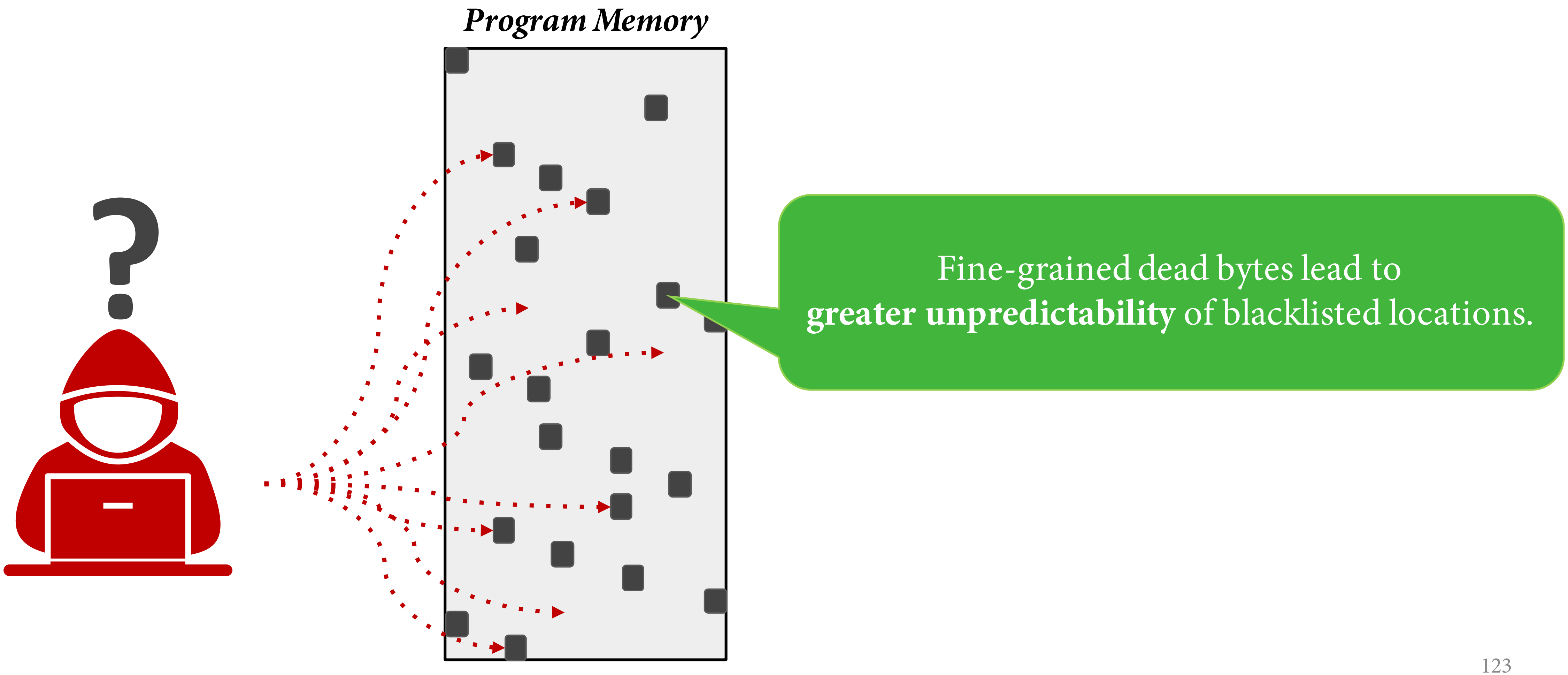


JavaScript V8 Engine

Struct Density = $\sum_i^{\#fields}(\text{sizeof}(\text{field}_i))/\text{sizeof}(\text{struct})$

Why are dead bytes useful for security?

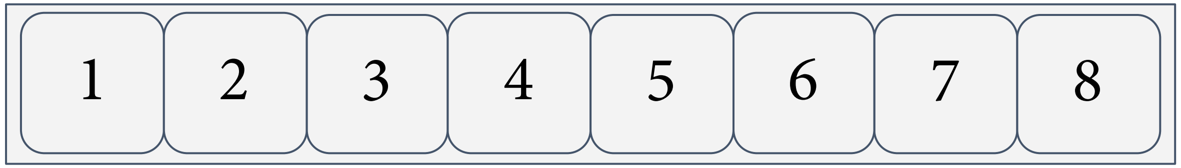
They are naturally inlined with data to provide fine-grained protection.



How do we encode data within dead bytes?

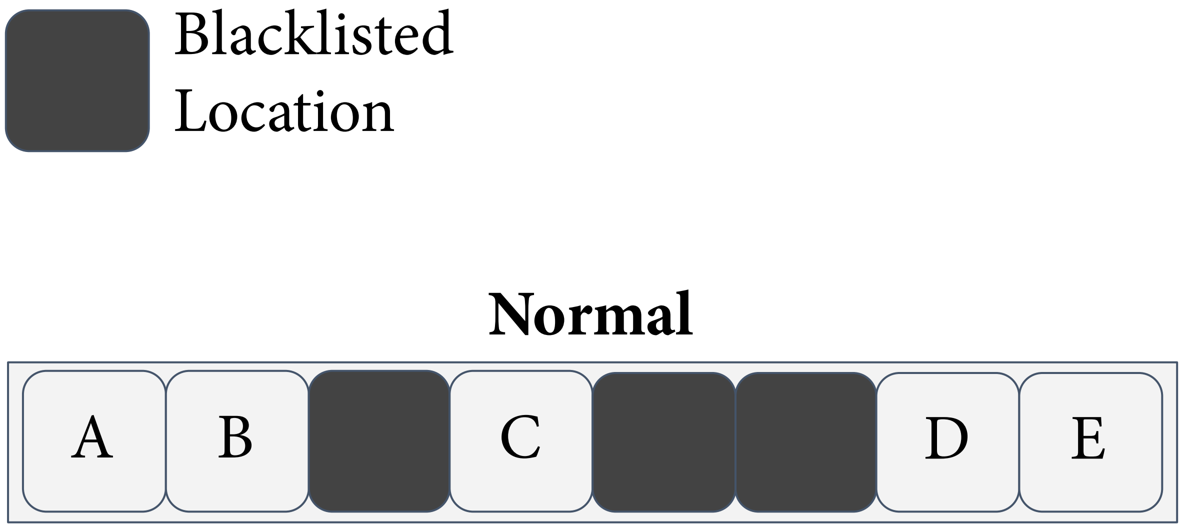
We use a novel cache line based encoding scheme for L1, L2 and beyond.

Normal



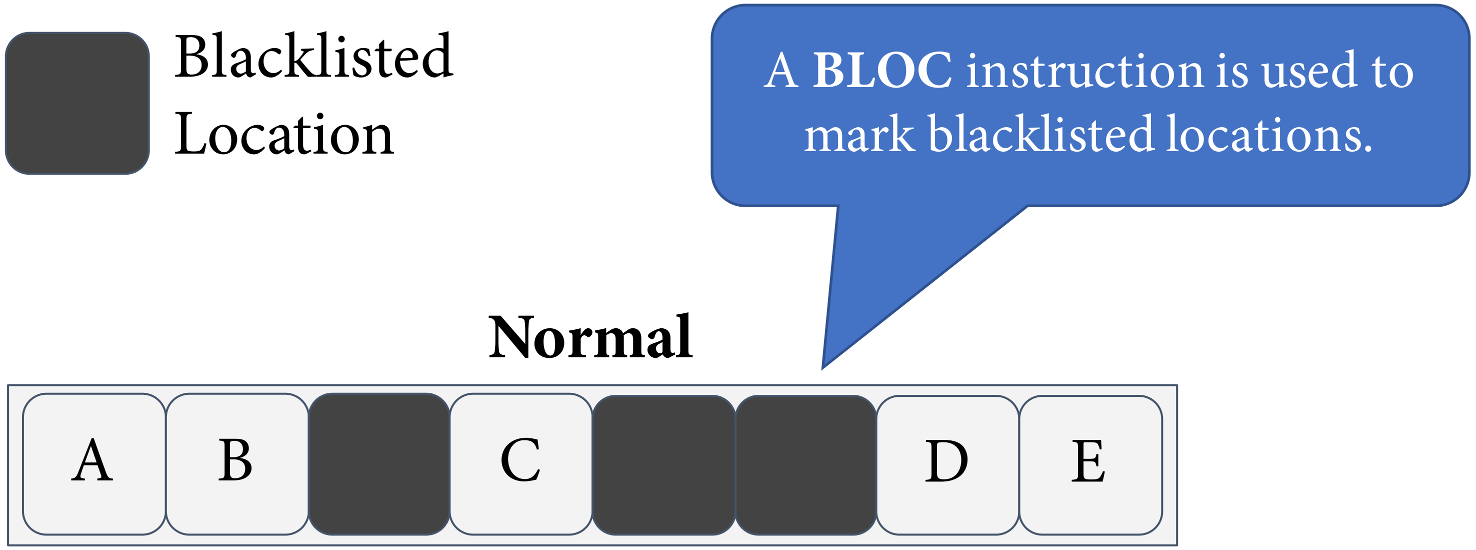
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



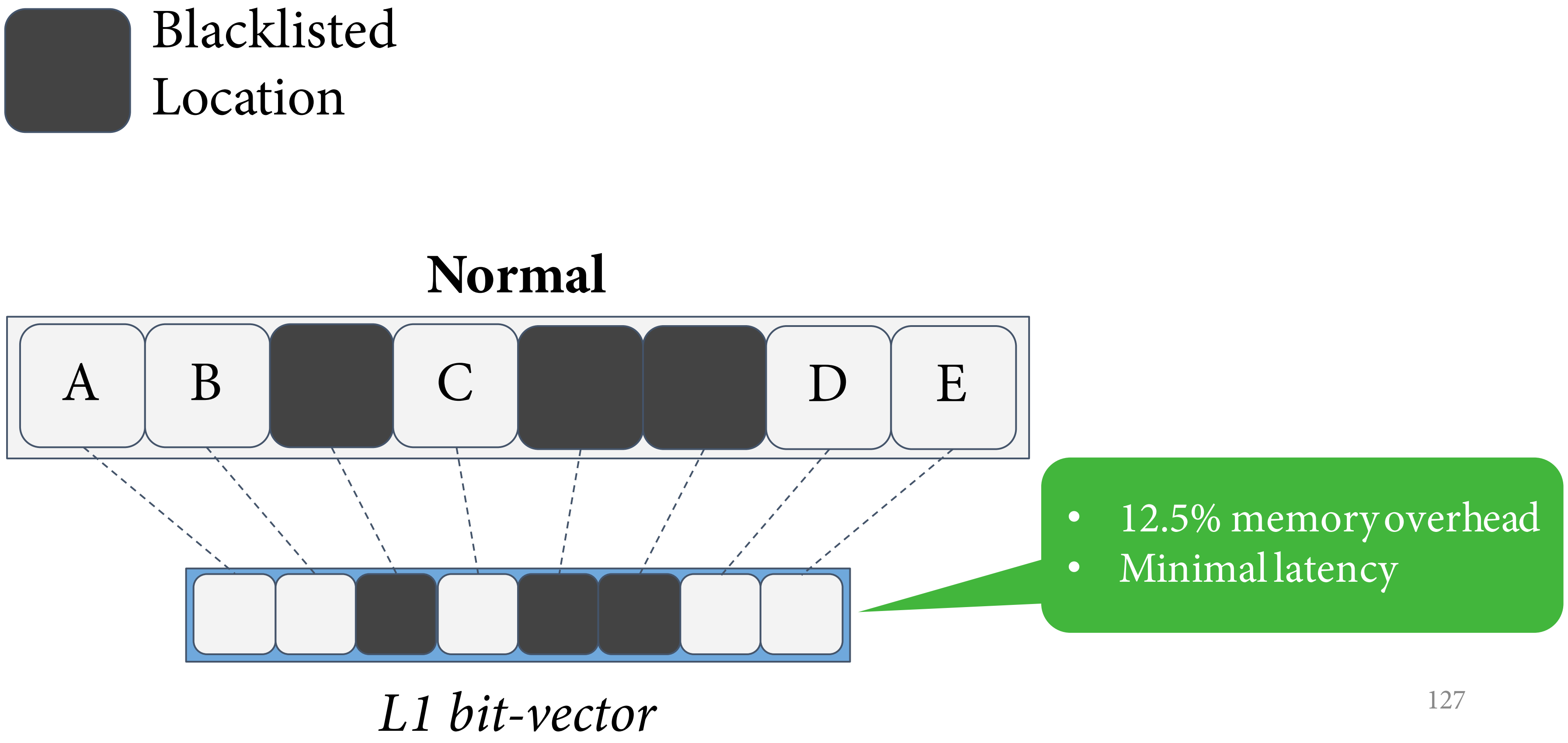
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



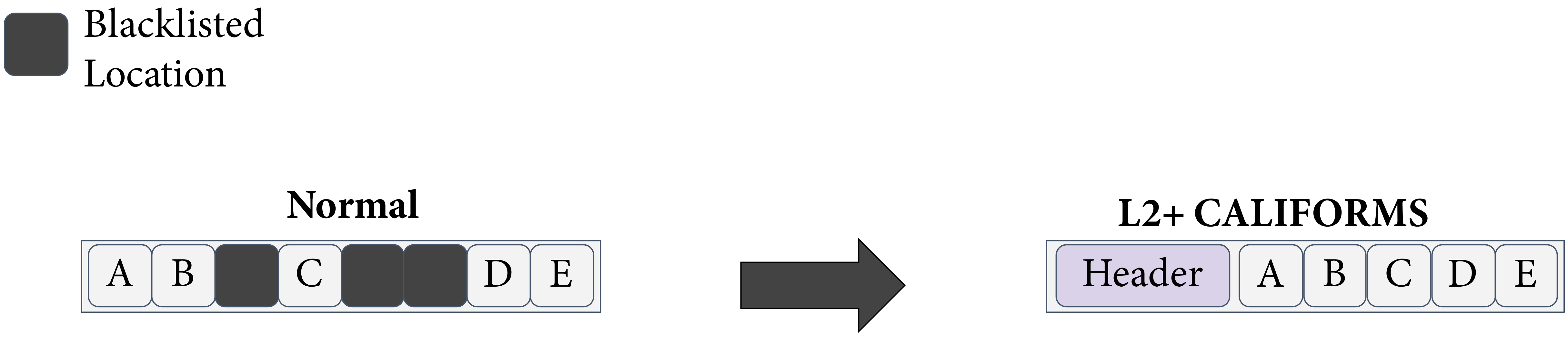
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



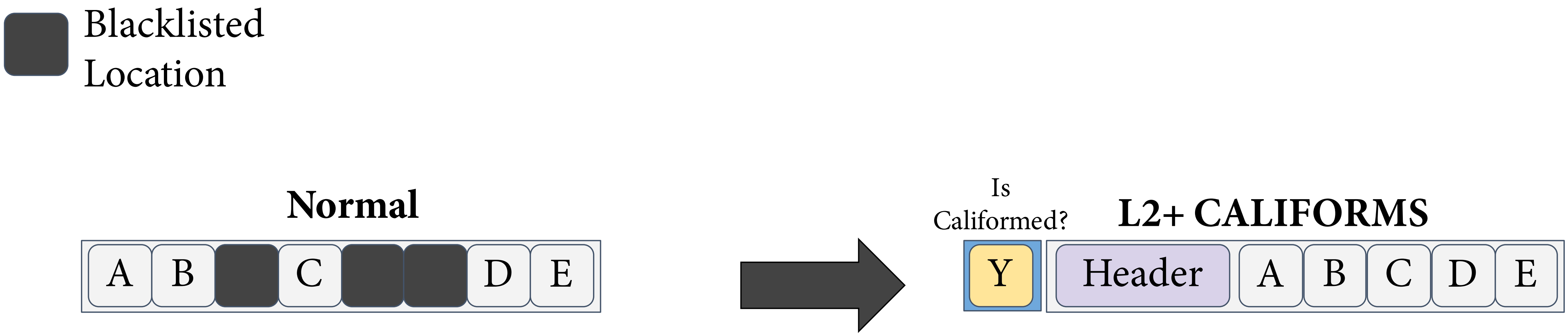
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



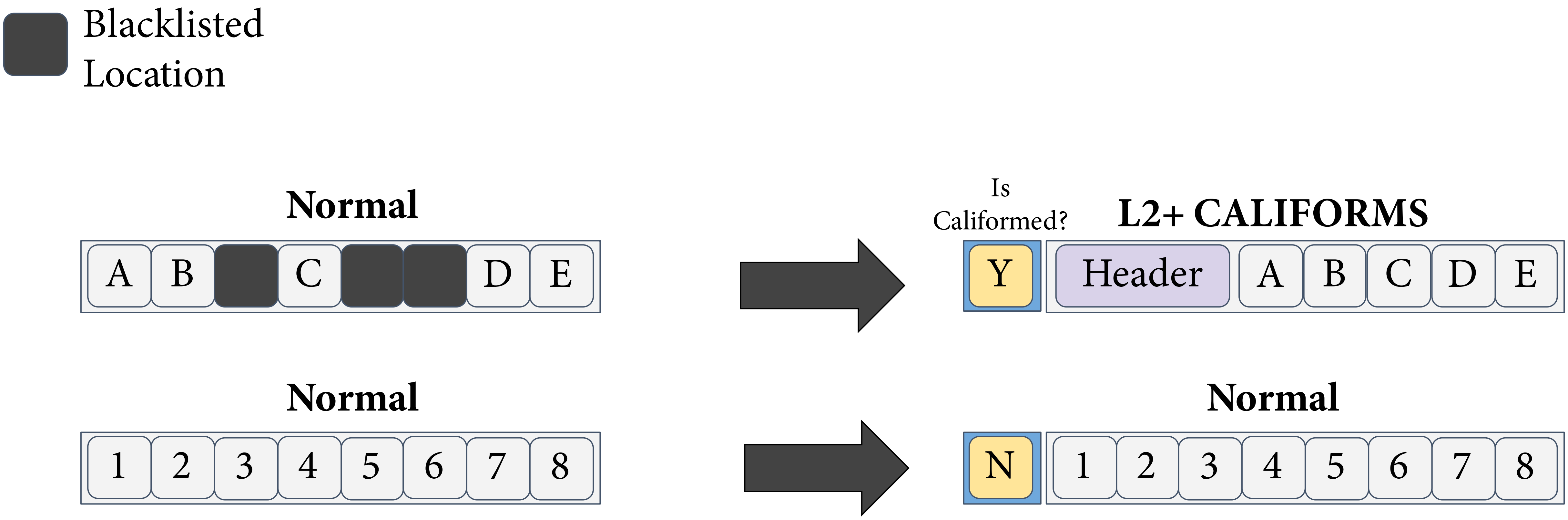
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



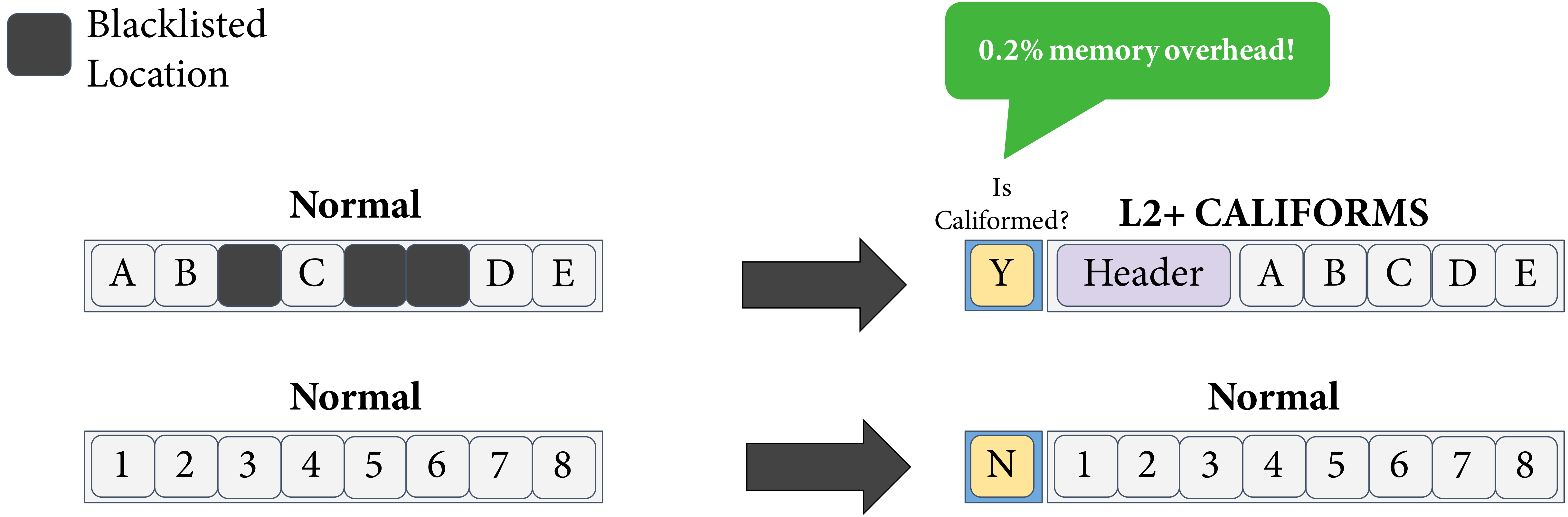
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



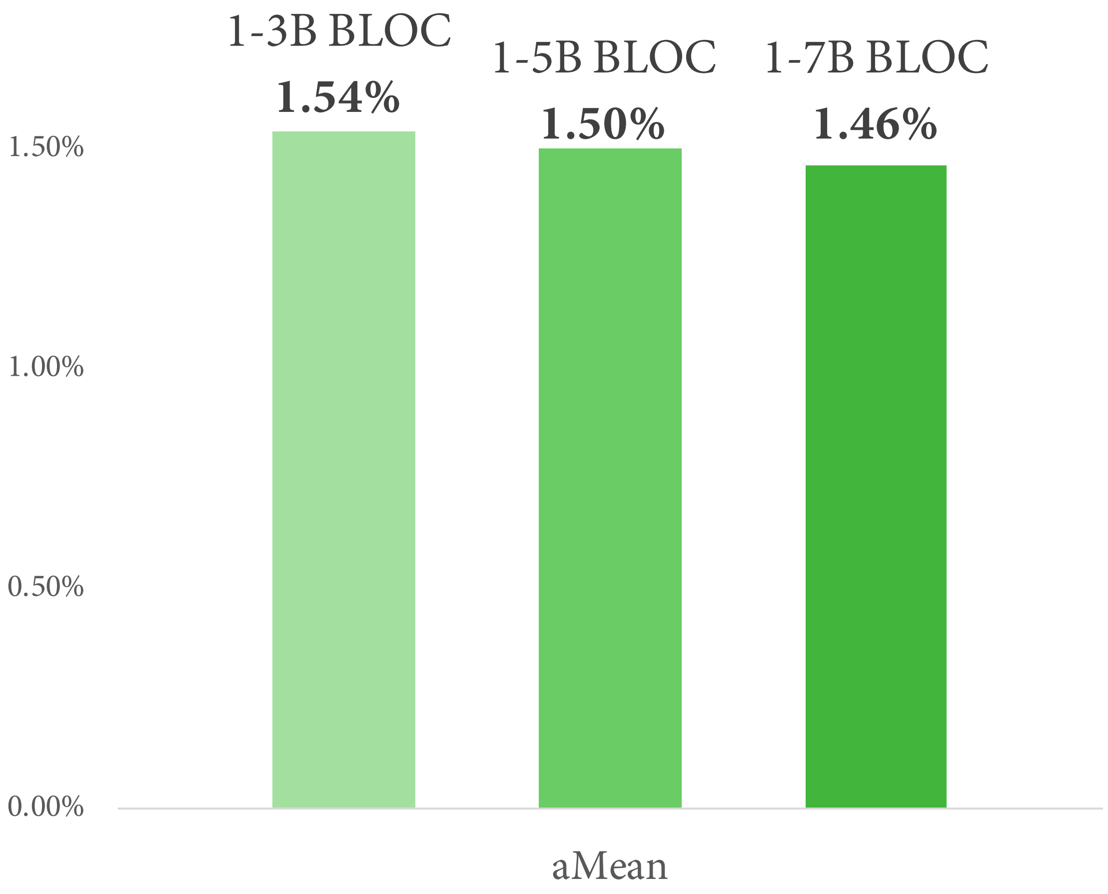
How do we encode data within dead bytes?

We use a novel cache line based encoding scheme for L1, L2 and beyond.



How was CALIFORMS evaluated?

Emulated BLOC instruction effects on real hardware.

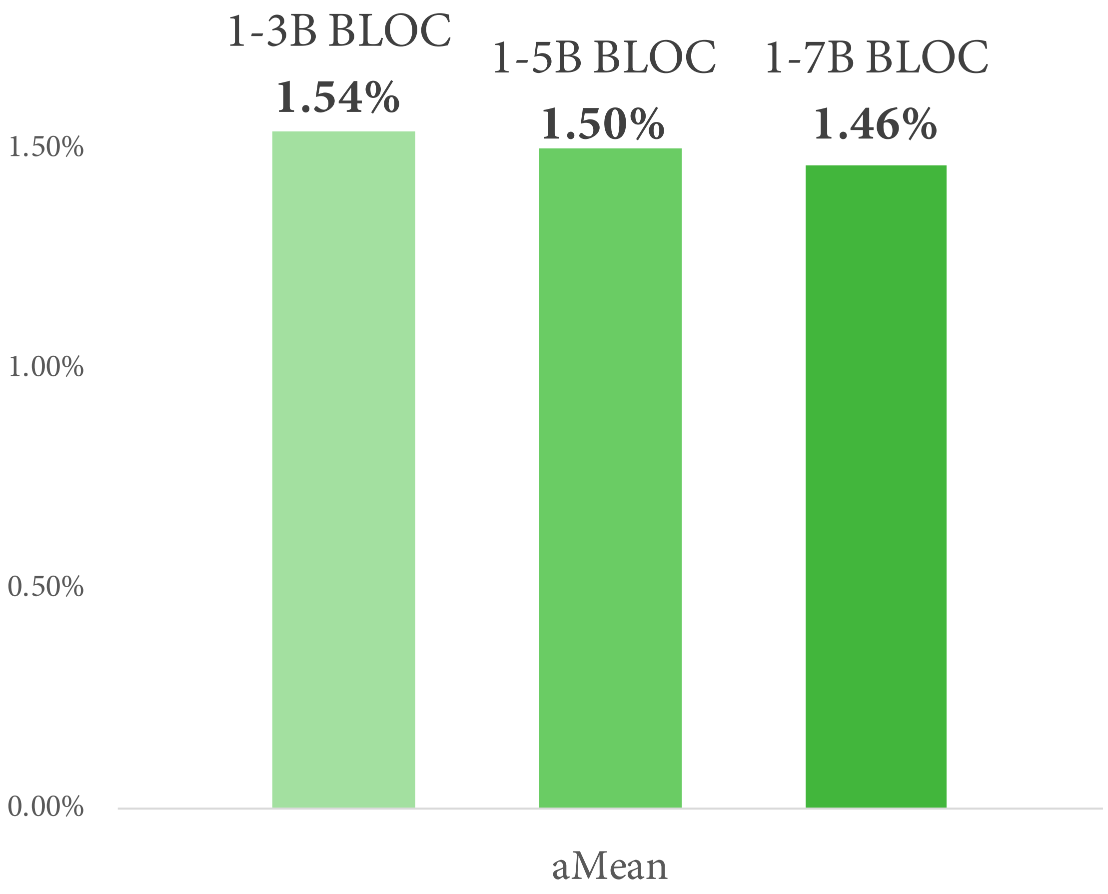


SPEC CPU2006

Intelligent Insertion Policy

How was CALIFORMS evaluated?

Emulated BLOC instruction effects on real hardware.



Average slowdown < 2%!

SPEC CPU2006

Intelligent Insertion Policy

Why is CALIFORMS useful in constrained devices?

It brings memory blacklisting to a new class of devices.

Memory Savings

Uses dead bytes in already allocated memory with minimal impact & reduces memory accesses.

Limited Scope of Changes

Changes are contained to the cache controllers making it portable to any architecture.

My contributions to CPS security

An overview of publications

1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from the system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for inline fine-grained metadata storage.

Thesis Statement

Security can be efficiently integrated by leveraging fundamental **physical properties**, & tailoring and extending **age-old abstractions** in computing.

Why are my contributions well suited for constrained devices?

Memory Savings
Cut down on resource duplication.

Cost Savings
Minimize on redundant resources to maintain system security.

Minimal Performance Impact
Minimal impact on workload execution.

Limited Scope of Changes
Changes are contained to be portable to any architecture.

Many thanks to all I've collaborated with!

Mohamed, Hiroshi, Kanad, Evgeny, Koustubha,
Hidenori, Vasileios, Junfeng

Special thanks to my advisor!

Simha


#BTTF2015

My contributions to CPS security

An overview of publications

1. YOLO: You Only Live Once

A mitigation that leverages *inertia* to periodically wipe an attacker from a system.

2. PAS: Phantom Address Space

An architectural primitive for diversified execution.

3. CALIFORMS: Cache Line Formats

A mechanism for fine-grained inline metadata storage.

IEEE 2019 Micro Top Picks Honorable Mention

Questions?