MORPHEUS

A Vulnerability-Tolerant Secure Architecture Based on Ensembles of Moving Target Defenses with Churn

M. Gallagher, L. Biernacki, S. Chen, Z.B. Aweke, S.F. Yitbarek, M.T. Aga, A. Harris, Z. Xu, B. Kasikci, V. Bertacco, S. Malik, M. Tiwari, T. Austin

UNIVERSITY OF MICHIGAN
TEXAS The University of Texas at Austin
PRINCETON UNIVERSITY

MORPHEUS

A Vulnerability-Tolerant Secure Architecture Based on Ensembles of Moving Target Defenses with Churn

M. Gallagher, L. Biernacki, S. Chen, Z.B. Aweke, S.F. Yitbarek, M.T. Aga, A. Harris, Z. Xu, B. Kasikci, V. Bertacco, S. Malik, M. Tiwari, T. Austin

UNIVERSITY OF MICHIGAN
TEXAS The University of Texas at Austin
PRINCETON UNIVERSITY

Secure System Design *Now*

- Secure design “loop”:

  For-each vulnerability:

    Attackers exploit vulnerability

    Defenders patch vulnerability

- List of vulnerabilities increasing…

- Not typically possible to prove security against *all* vulnerabilities

Characteristics of Exploits

- Benign programs may have vulnerabilities →

  Defenses need to be *vulnerability-tolerant*

Vulnerabilities + Information Assets = Exploit

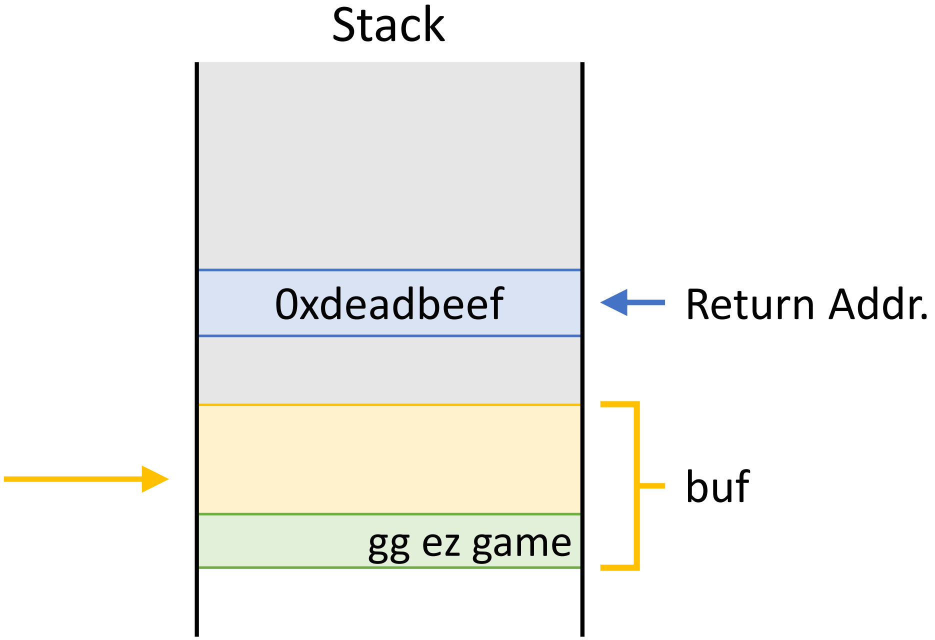
- Attackers use *internal program assets*:
  - Byproduct of system implementation
  - Usually not relied-on by programmers

Exploits: Abusing Program Assets

Benign Use-Case

```
char buf[30];
strcpy(buf, arg);
```

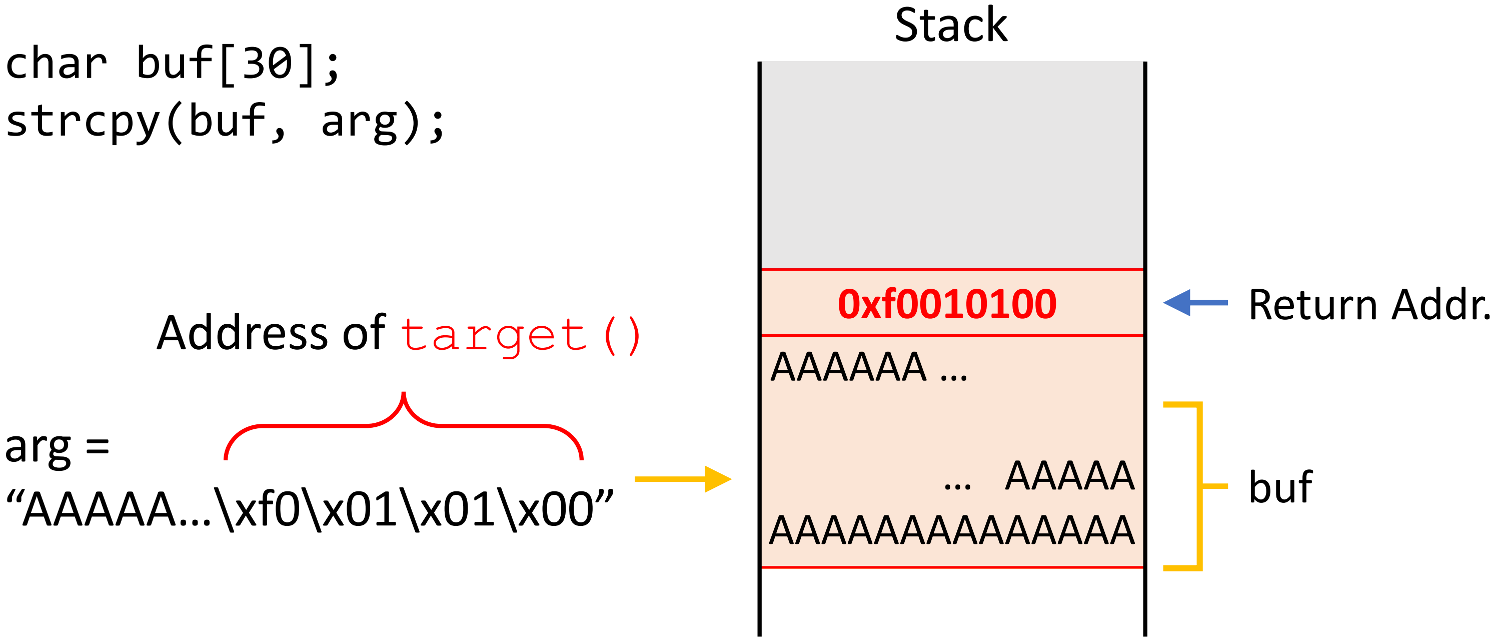
arg = “gg ez game”



Exploits: Abusing Program Assets

Malicious Use-Case

```
char buf[30];
strcpy(buf, arg);
```



Information Assets:

- Location of `target()`
- Pointer Representation

Protecting Information Assets

An Approach:

Randomize assets

Moving Target Defenses (MTDs)

Load-time MTDs: 64-bit ASLR, ISR, …

Attackers defeat load-time MTDs with ***Derandomization Attacks***

➔ Load-Time MTDs have *LOW* durability

Protecting Information Assets

An Approach:

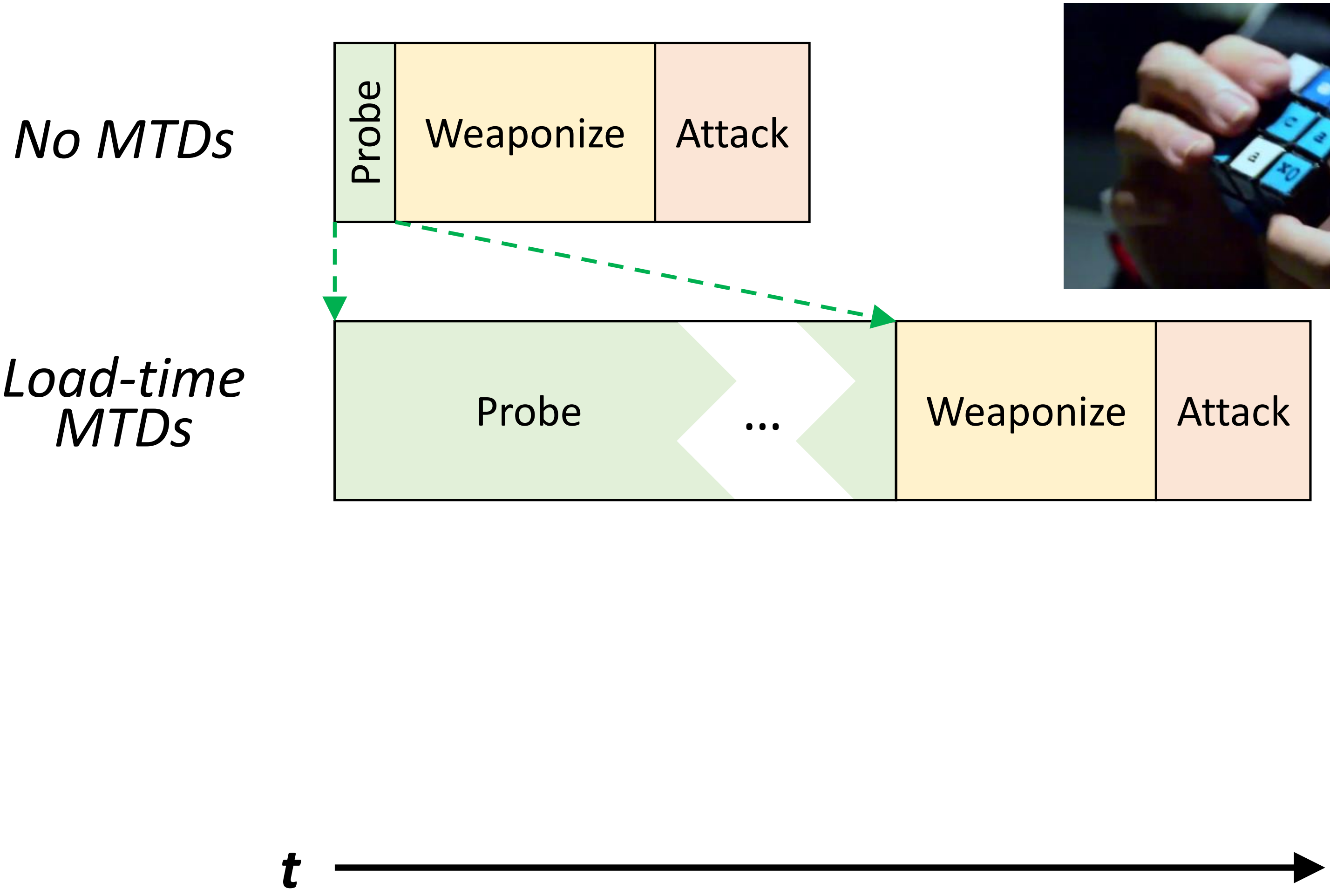
Randomize assets

Morpheus uses H/W-supported re-randomization *(Churn)* to give high-entropy MTDs better durability

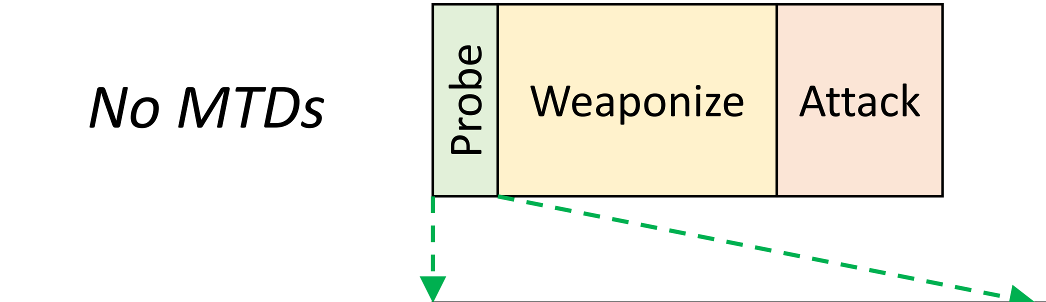
Attackers defeat load-time MTDs with ***Derandomization Attacks***

➔ **Load-Time MTDs have *LOW* durability**

Attacks vs. (Re-)Randomization



Attacks vs. (Re-)Randomization

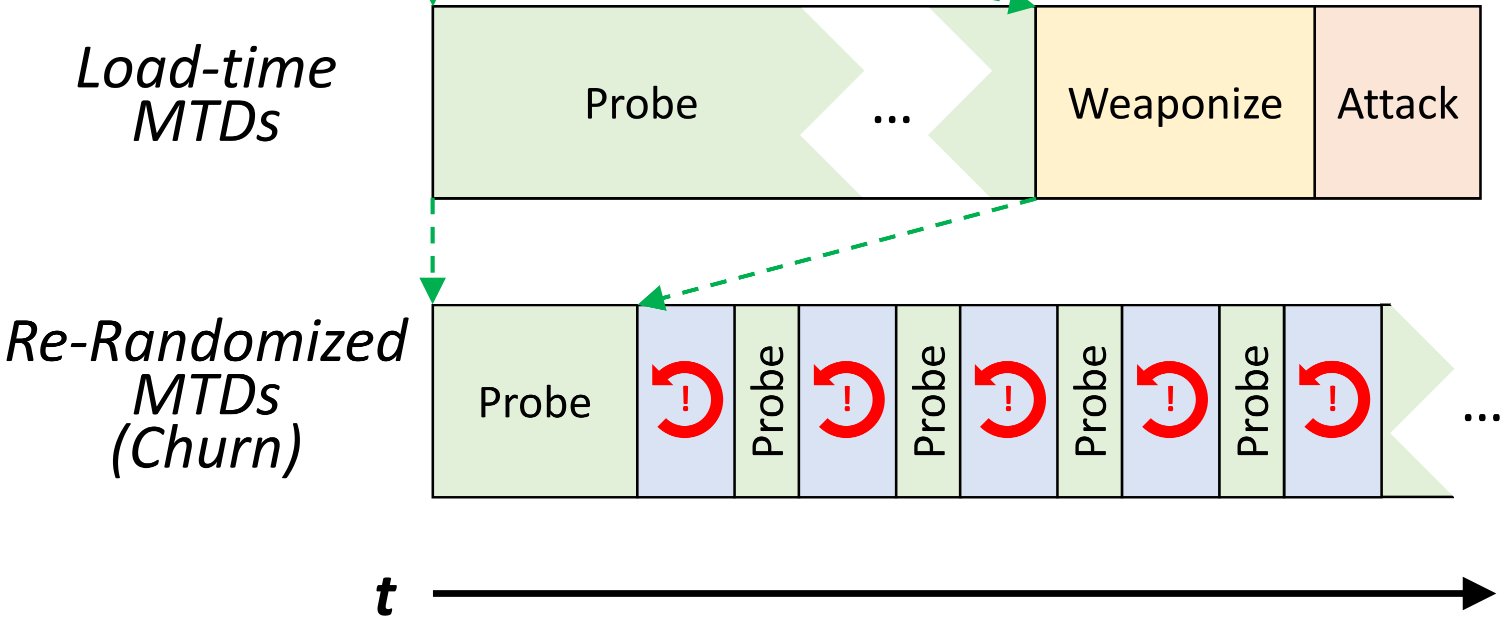


No MTDs

Probe

Weaponize

Attack



Load-time MTDs

Probe

...

Weaponize

Attack

Re-Randomized MTDs (Churn)

Probe

!

Probe

!

Probe

!

Probe

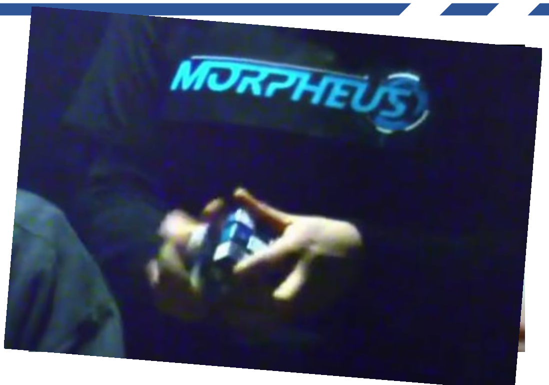
!

Probe

!

...

t



Introduction

Morpheus Architecture

Evaluations

Parting Thoughts

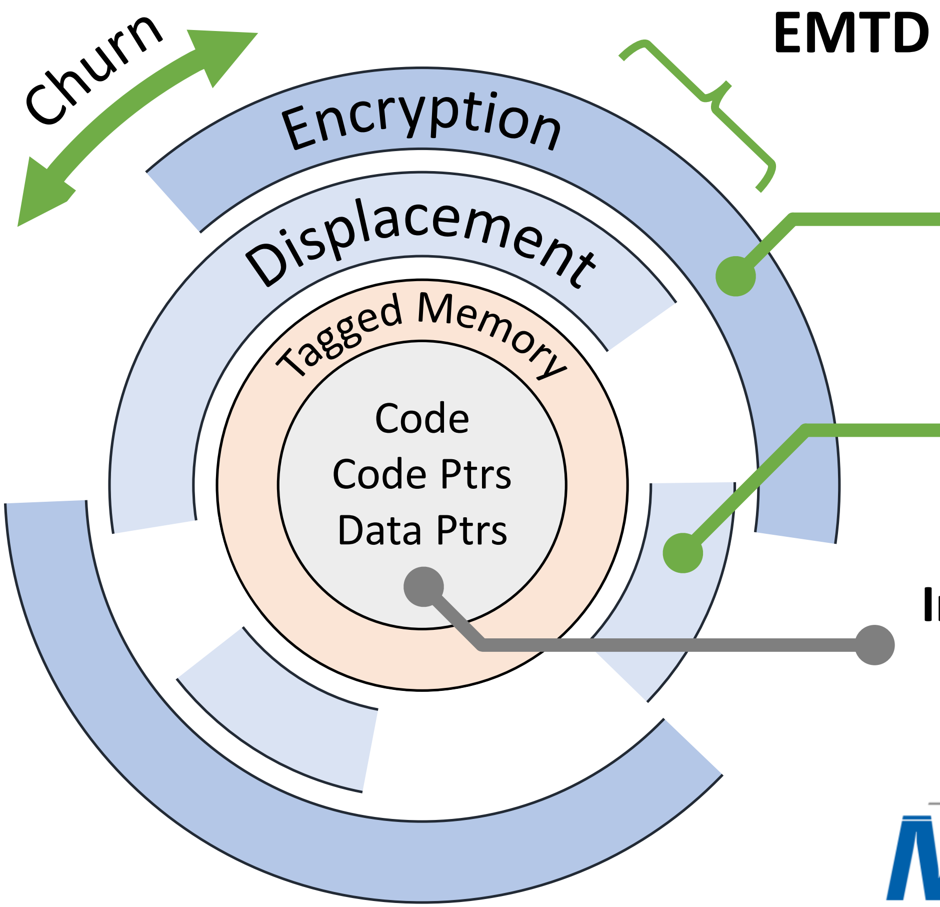
Introduction

Morpheus Architecture

Evaluations

Parting Thoughts

Morpheus: Ensemble of MTDs



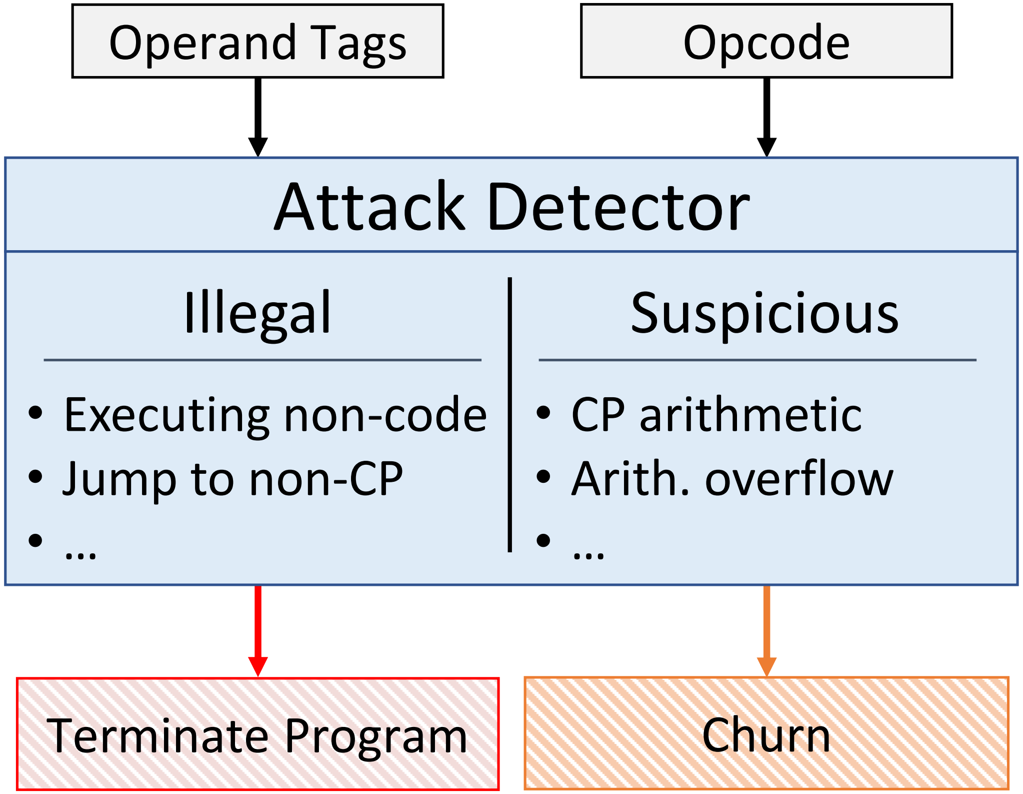
Encryption:
Obfuscate information assets

Displacement:
Shift code & data segments

Information Assets

Tagging & Attack Detection

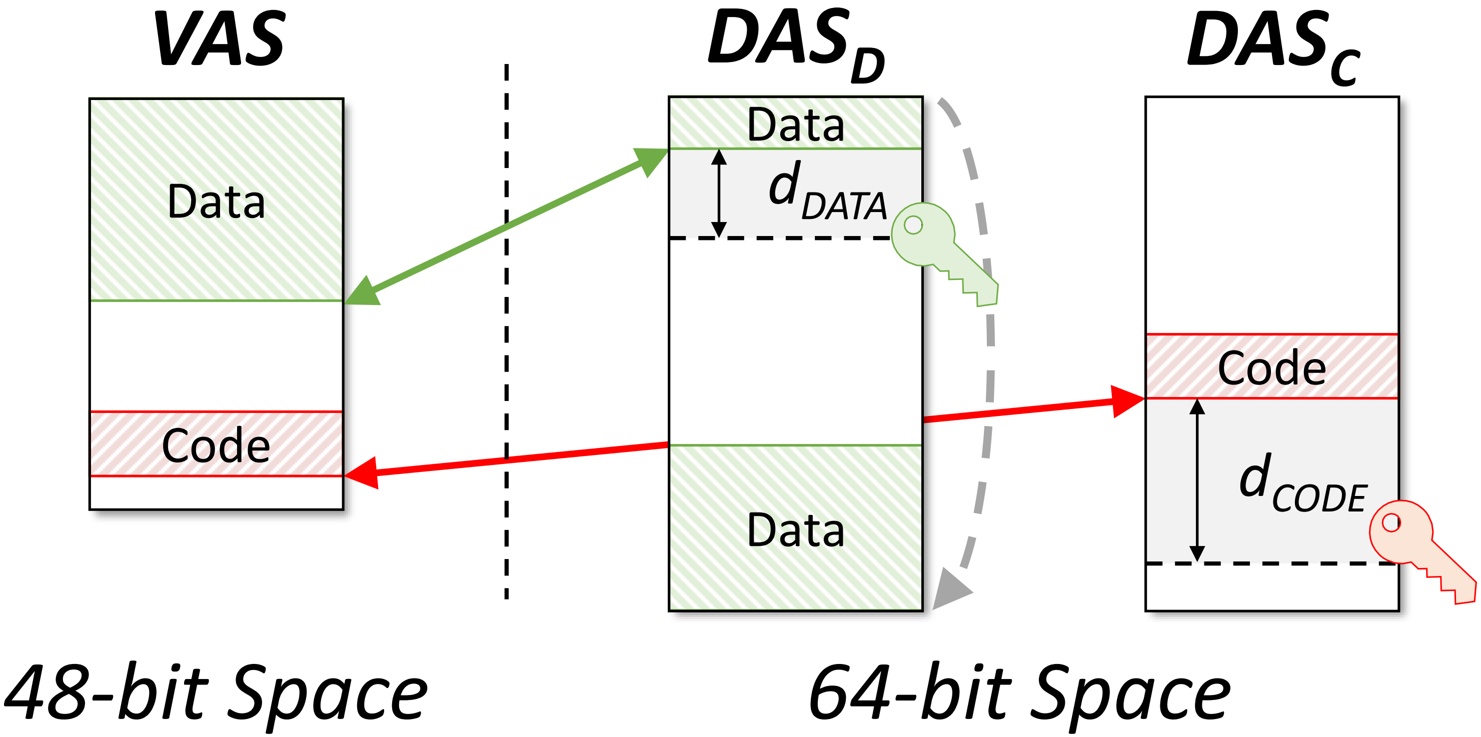
- Tags enable behavior tracking
- Illegal Ops
  - Clearly dangerous
- Suspicious Ops
  - Normal programs may perform
  - May be probes or attacks



Otherwise, churn every 50ms

Displacement

- Introduces entropy to Code & Data *location*
- Shift address space into 2 independent spaces
  - Add *d*, a 60 bit displacement, to pointers

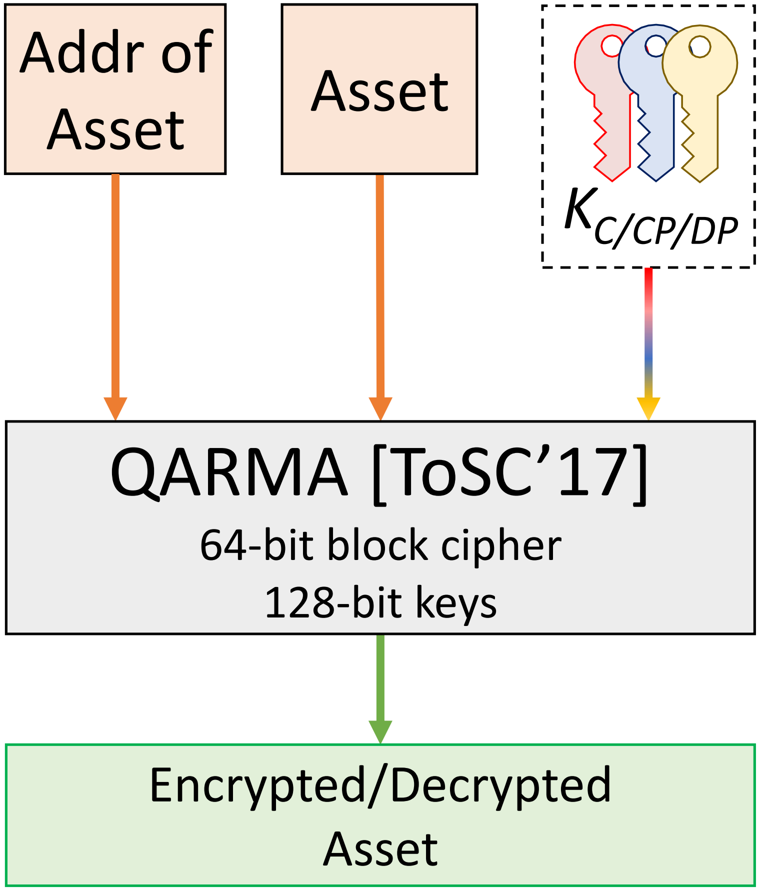


Encryption

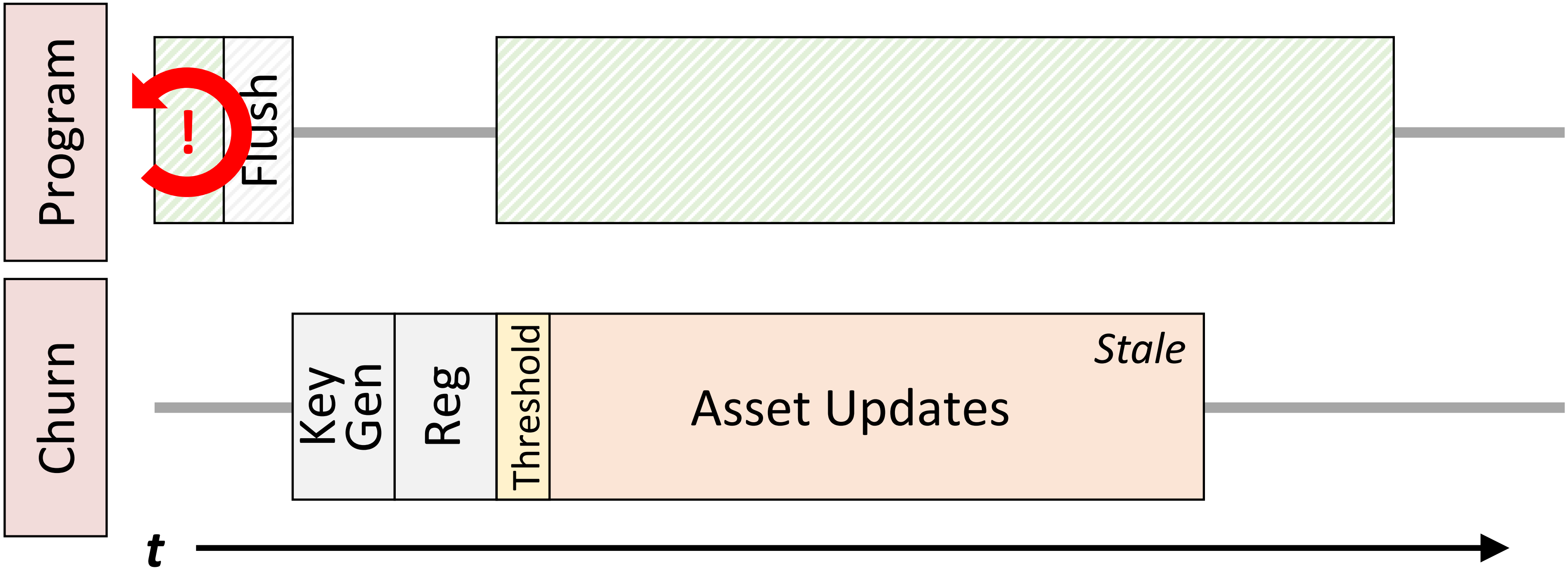
- Introduces entropy to Code & Pointer *values*

- Encrypt domains under own keys
  - Code
  - Code Pointer
  - Data Pointer

- QARMA Block Cipher
  - Fast cipher used in Arm's PAC
  - Used in *counter-mode* here



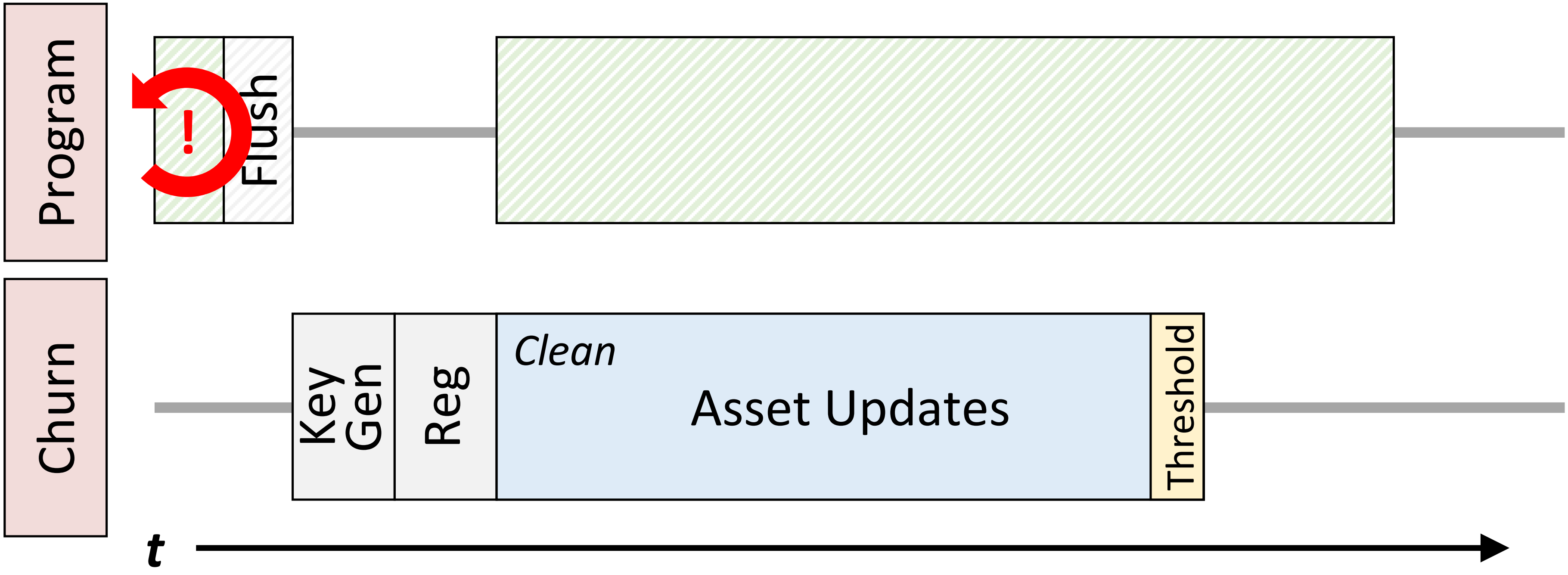
Churning EMTDs



Stale: Under OLD key

Clean: Updated to NEW key

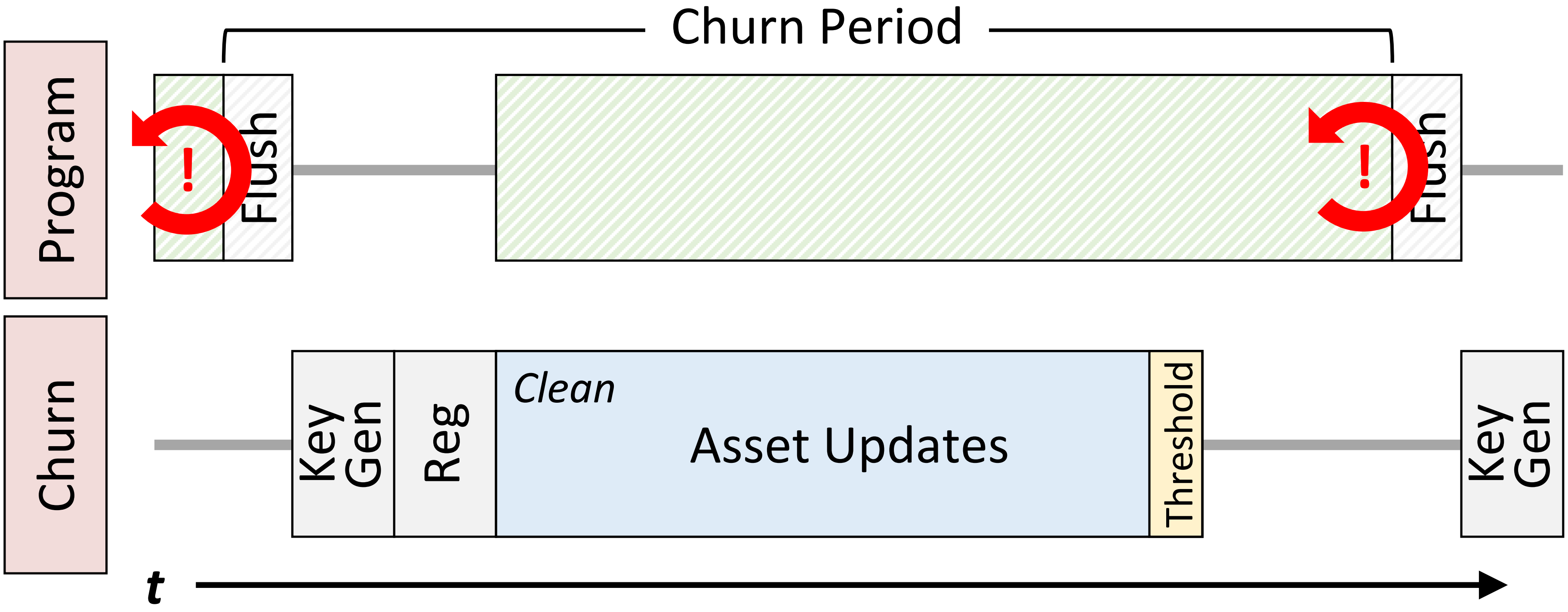
Churning EMTDs



Stale: Under OLD key

Clean: Updated to NEW key

Churning EMTDs



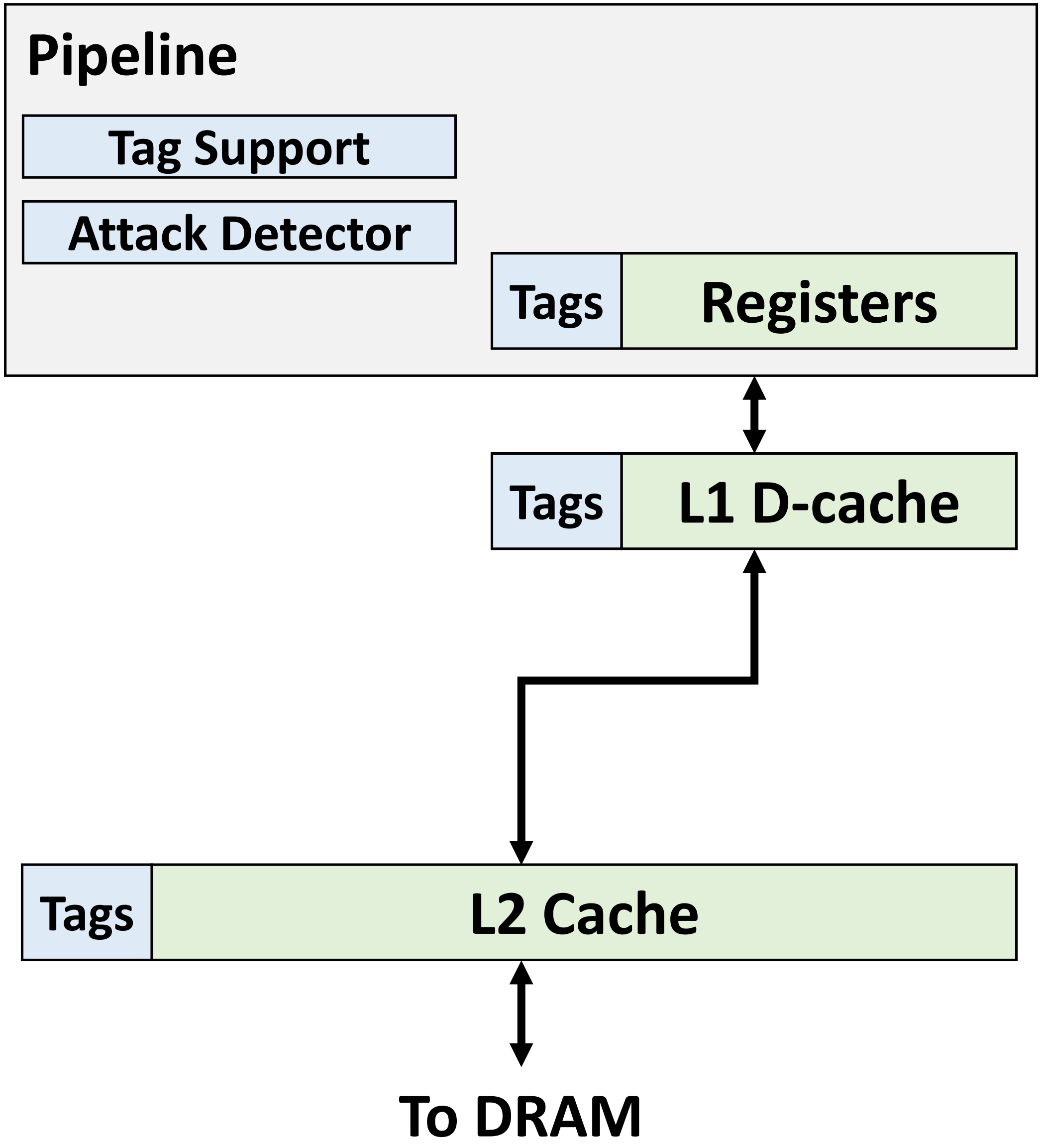
Stale: Under OLD key

Clean: Updated to NEW key

µArch Additions

Tagged Memory

- Tag Propagation
- Attack Detector



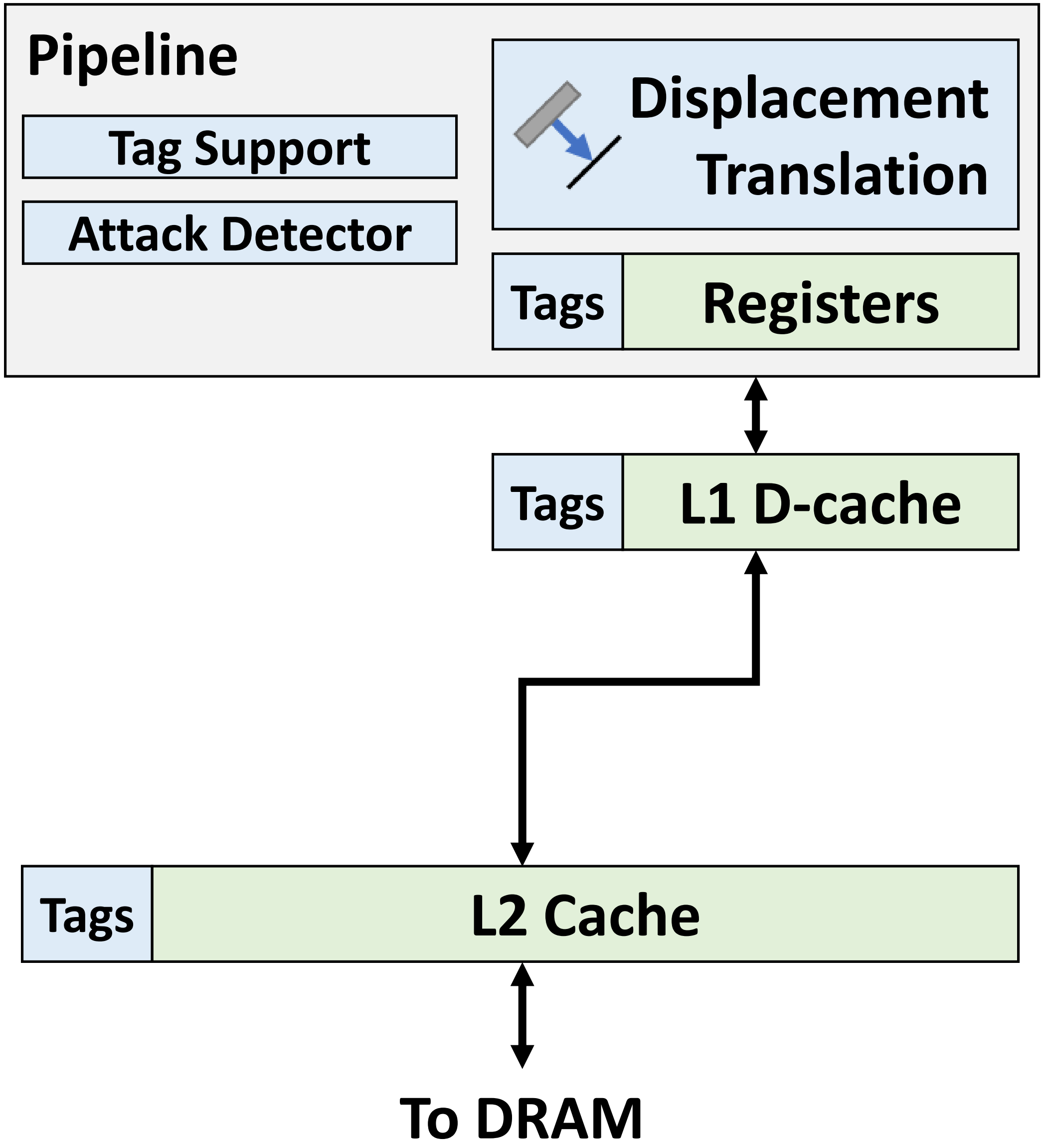
µArch Additions

Tagged Memory

- Tag Propagation
- Attack Detector

Displacement

- Translate DAS→VAS



µArch Additions

Tagged Memory

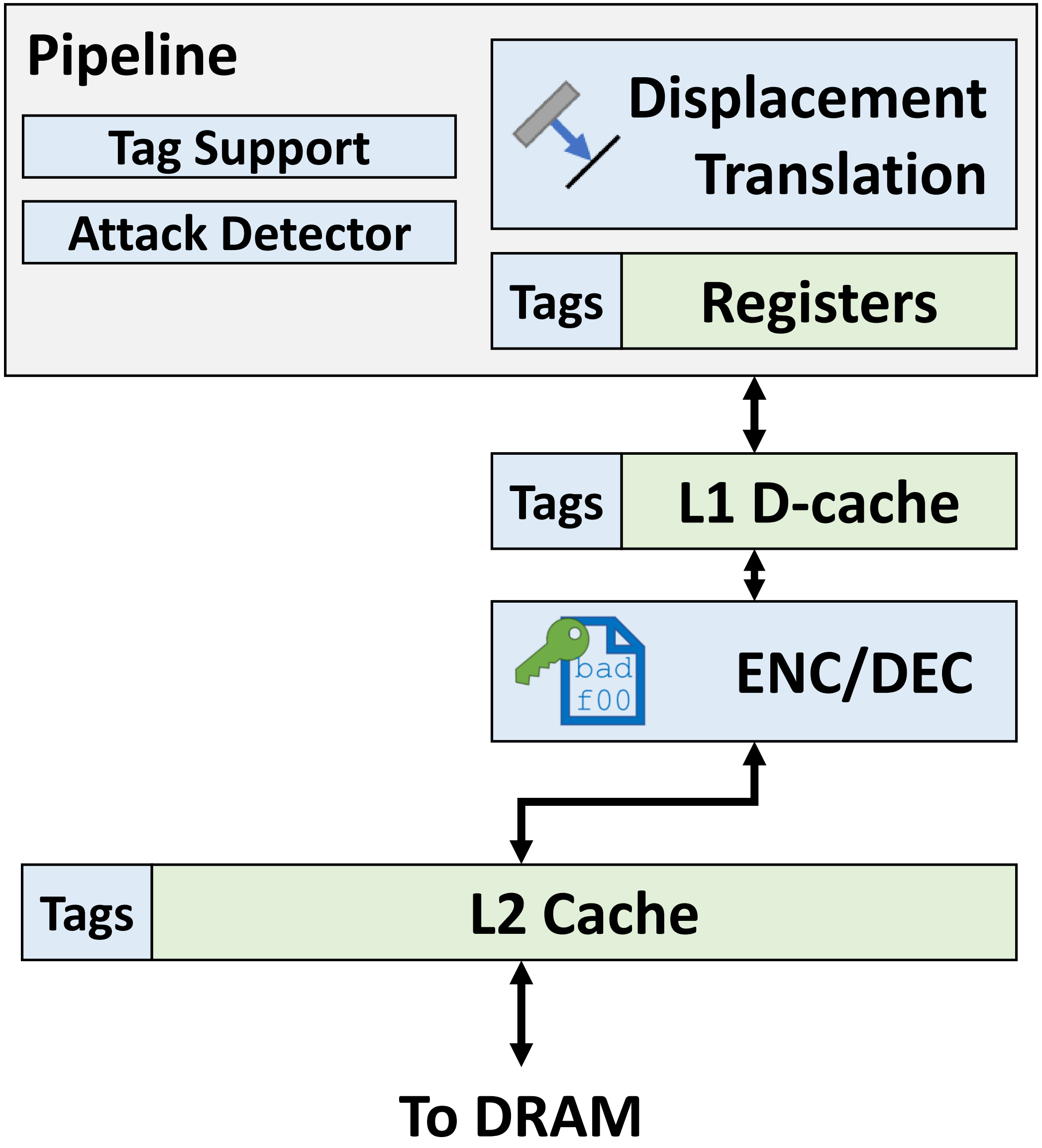
- Tag Propagation
- Attack Detector

Displacement

- Translate DAS→VAS

Encryption

- QARMA Engines



µArch Additions

Tagged Memory

- Tag Propagation
- Attack Detector

Displacement

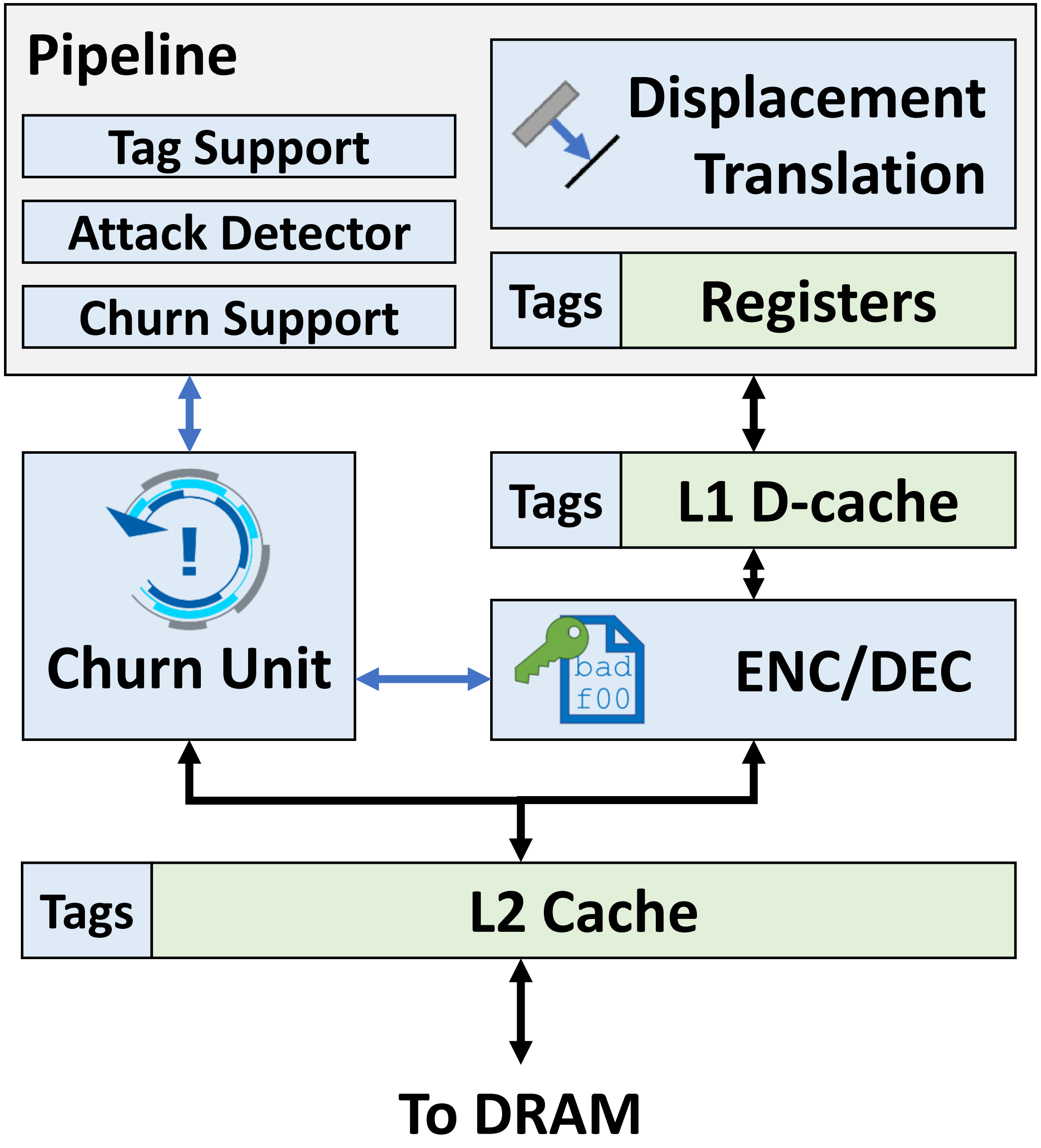
- Translate DAS→VAS

Encryption

- QARMA Engines

Churn Unit

- State Machine
- RNG (Key-Gen)
- Threshold Register



Introduction

Morpheus Architecture

Evaluations

Parting Thoughts

Introduction

Morpheus Architecture

Evaluations

Parting Thoughts

Evaluation Framework

- gem5 + DRAMSim2
  - RISC-V – RV64IMA ISA
  - Implements churn unit
  - Simulate tag fetch & Tag$
- Benchmarks:
  - SPEC 2006, INT+FP, C-only
  - Subset of MiBench

Core Type	MinorCPU (InO)
CPU Freq.	2.5GHz
L1 I$	32KB 2-cycle
L1 D$	32KB 2-cycle
L2 Unified	256KB 20-cycle
Tag Cache	4KB

Security in Morpheus

How long to penetrate Morpheus defenses?

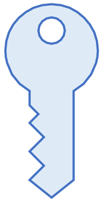
- Difficult to attack a system that is
  - Constantly changing
  - Has high entropy

- Approach: Attack a *weaker* Morpheus

De-featured Morpheus

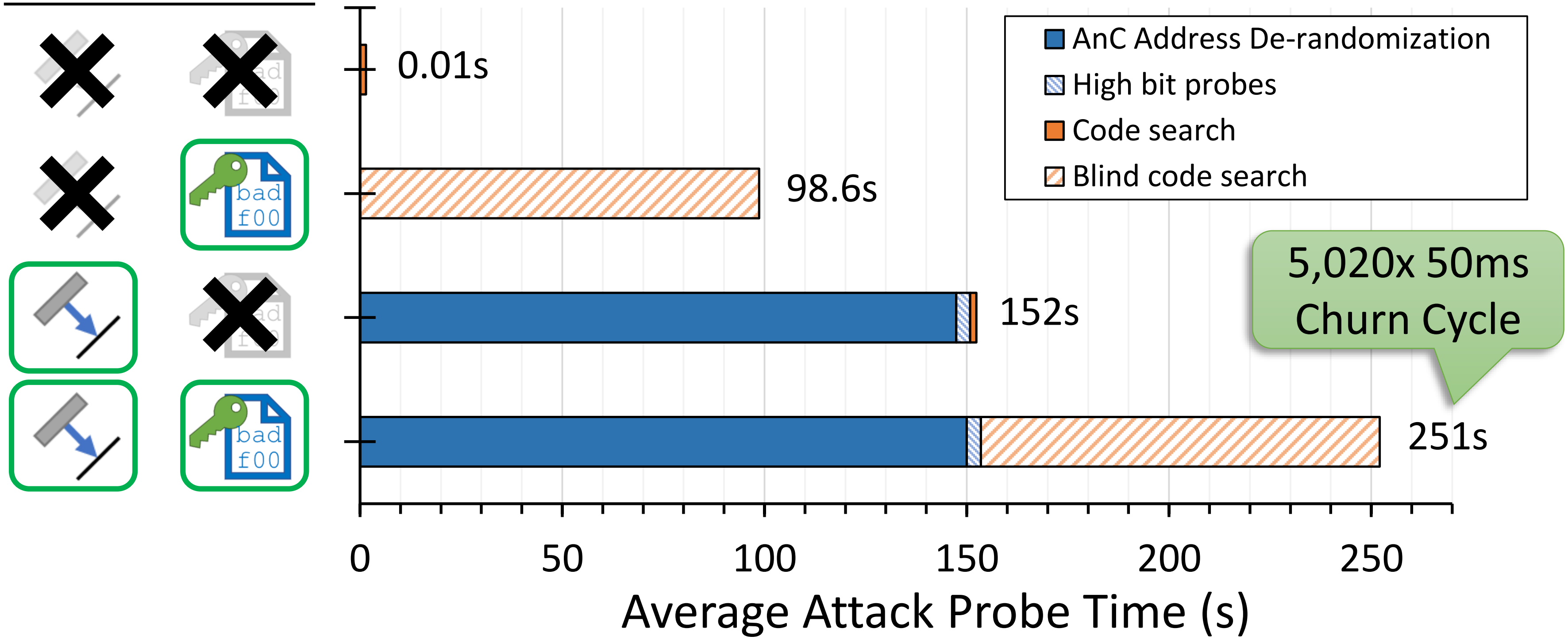
Churn Disabled

Shared Key for Defenses



Attacking a Weakened Morpheus

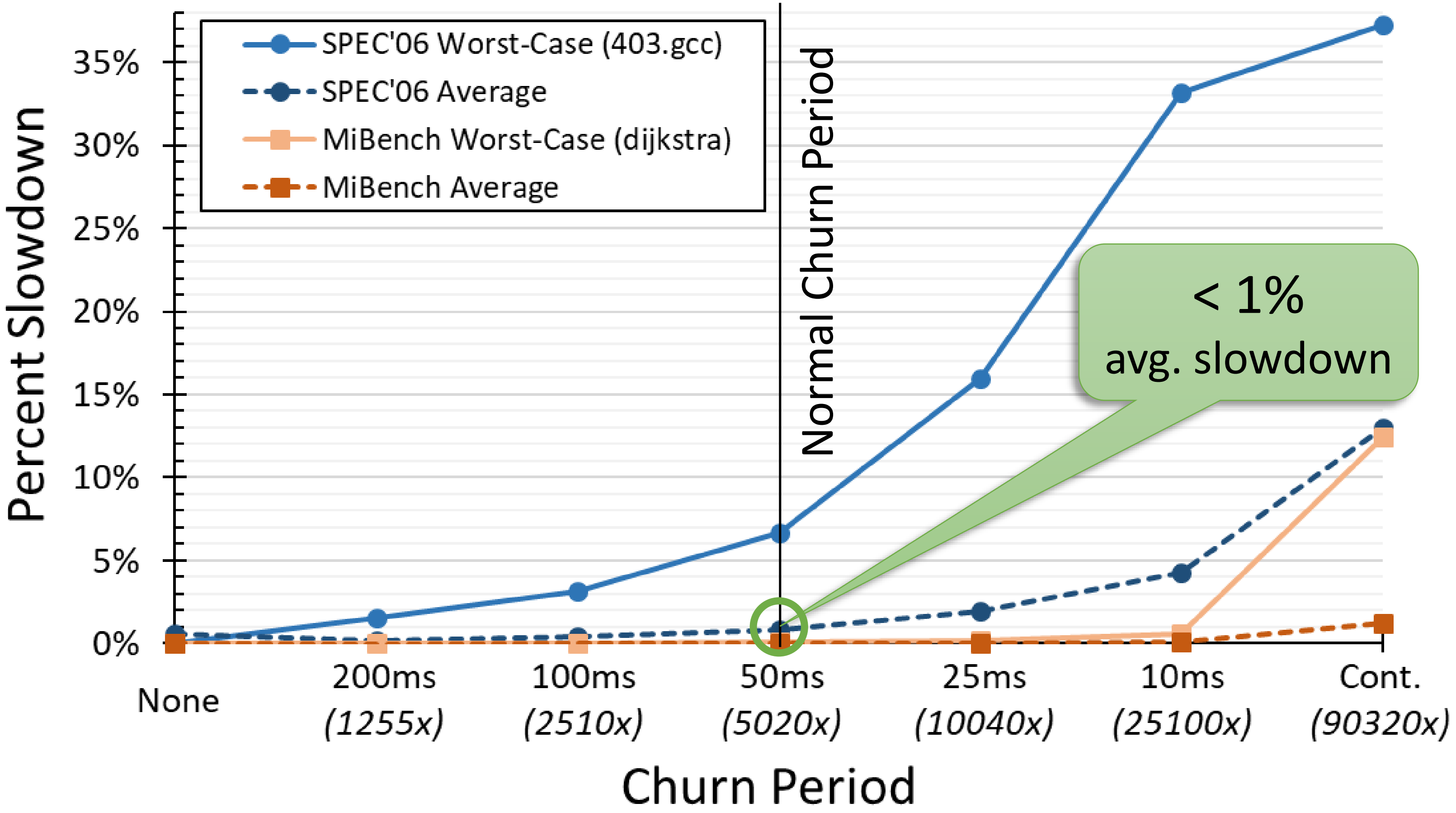
Defenses Enabled



5,020x 50ms Churn Cycle

251s to penetrate a Morpheus system with *high entropy* & *no churn*!

Effects of Churn Period



< 1%
avg. slowdown

Evaluation Summary

*Keys change **5020x faster** than time-to-penetrate with advanced probes*

*Low performance impact (**<1%**) on system*

*With network latencies of **~1ms/36miles**, churn **invalidates** information before attackers can use it*

Introduction

Morpheus Architecture

Evaluations

Parting Thoughts

Introduction

Morpheus Architecture

Evaluations

Parting Thoughts

Limi pheus

Future Work

- Relative Address Attacks
  - Distance between code & data churns
  - Distance *within* segments is preserved

→ Churn relative distance

- Reliance on Tagged Memory
  - Enables powerful EMTDs + Churn
  - Attacks missed by tag-checks are mitigated by EMTDs
  - Additional complexity of tagging

→ Support churn without tags

Conclusions

- EMTDs + Churn provide vulnerability tolerance
  - Attackers exploit vulnerabilities & information assets
  - EMTDs protect assets by churning them to stop derandomization
- Morpheus shows that with H/W support, we achieve:
  - High entropy defenses
  - High durability with churn
  - Low performance overhead (<1%)
- Future directions of EMTDs + Churn
  - Achieve stronger control-flow protections
  - Hinder side-channels
  - Create additional ensemble defenses

MORPHEUS

Conclusions

- EMTDs + Churn provide vulnerability tolerance
  - Attackers exploit vulnerabilities & information assets
  - EMTDs protect assets by churning them to stop derandomization

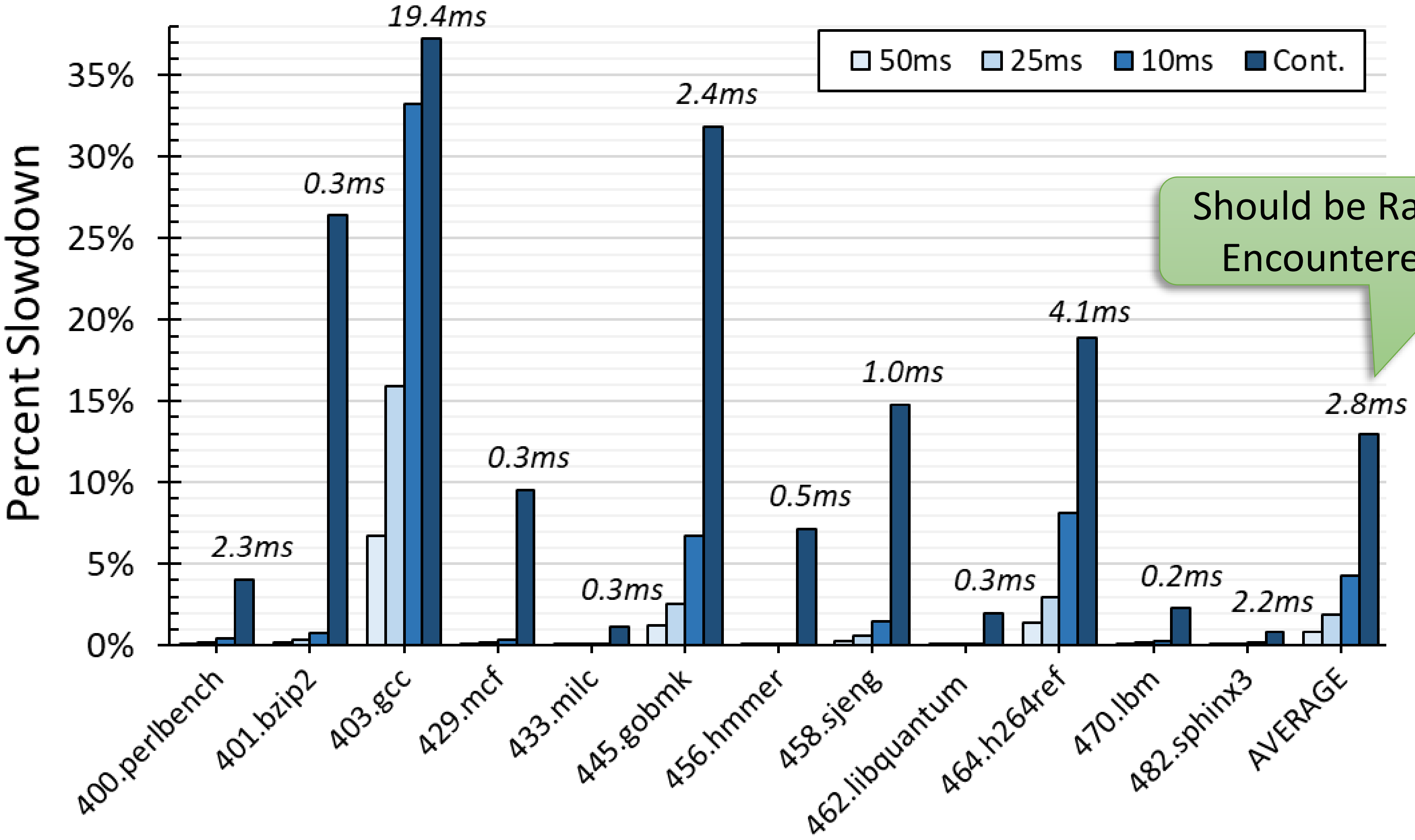
- Morpheus shows that with H/W support, we achieve:
  - High entropy defenses
  - High durability with churn
  - Low performance overhead (<1%)

- Future directions of EMTDs + Churn
  - Achieve stronger control-flow protections
  - Hinder side-channels
  - Create additional ensemble defenses

// BACKUP

Beep Beep

SPEC 2006 Detail



Should be Rarely Encountered!

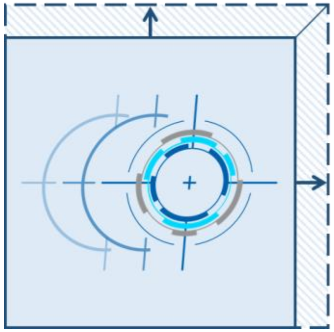
Penetration Testing

- RIPE testing suite
  - Used a subset of attacks ported to RISC-V
  - Code injection
    - Code is encrypted → injected code is invalid
  - Code reuse (ROP)
    - Locations shifted → injected return addresses invalid

- Back-Call-Site Attack (breaks Active-Set CFI)

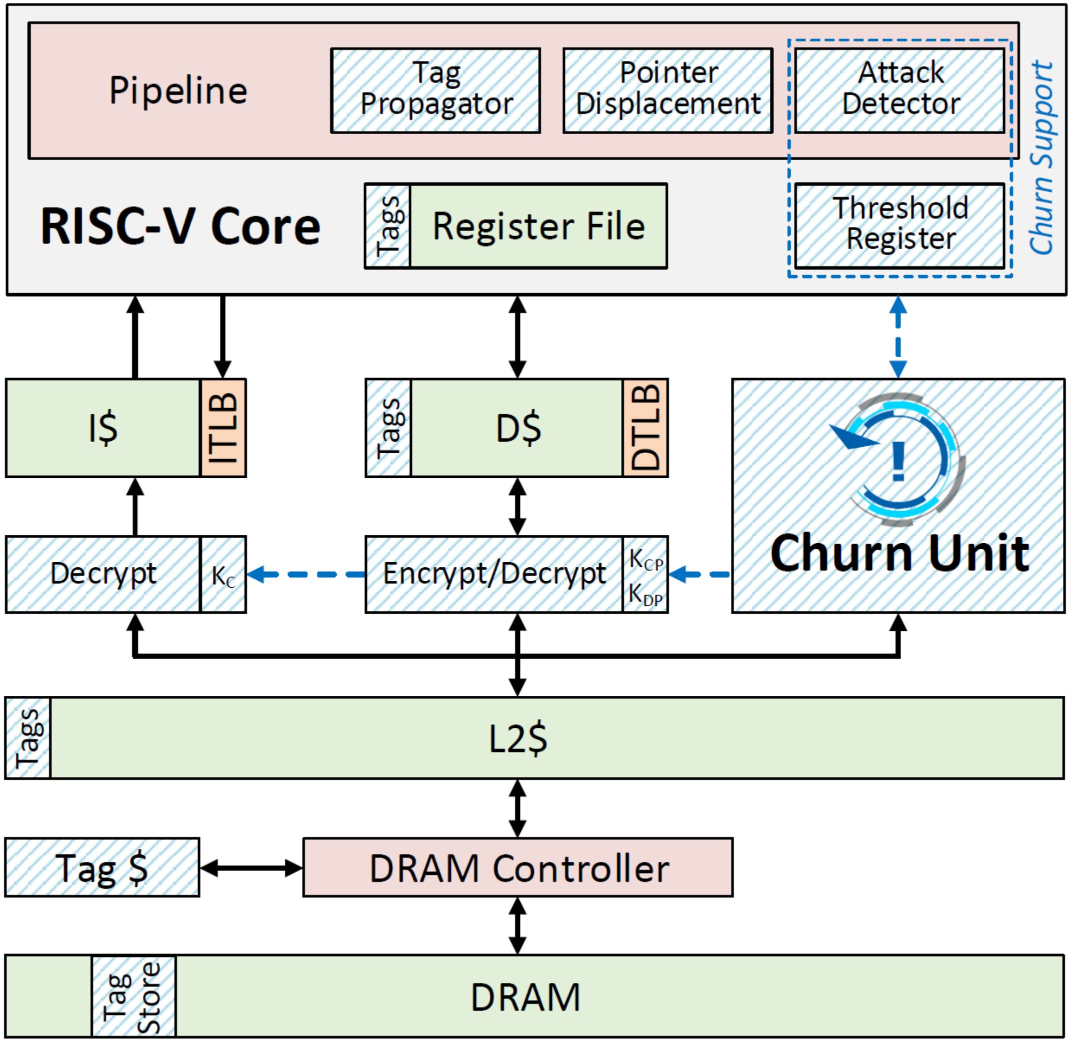
Hardware Area Estimate

- *[Not in paper]*
- Baseline: SiFive U54 - 28nm estimate
  - CACTI 7 for cache sizes
  - QARMA estimated from original work
  - Churn Support → smaller 64-bit core from SiFive



	SiFive U54-MC	Morpheus	
U54 w/ Caches	2.249 mm^2	2.249 mm^2	-
+ Tagged Memory	-	0.084 mm^2	3.74%
+ QARMA	-	0.044 mm^2	1.96%
+ Churn Support	-	0.082 mm^2	3.65%
Total	**2.249 mm^2**	**2.459 mm^2**	**9.34%**

Full µArch


Pipeline
Tag Propagator
Pointer Displacement
Attack Detector
Churn Support
RISC-V Core
Tags
Register File
Threshold Register
I$
ITLB
Tags
D$
DTLB
Churn Unit
Decrypt
K_C
Encrypt/Decrypt
K_{CP}
K_{DP}
Tags
L2$
Tag $
DRAM Controller
Tag Store
DRAM