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

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

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

\[ \text{Vulnerabilities + Information Assets = Exploit} \]

• Attackers use \textit{internal program assets}:
  • Byproduct of system implementation
  • Usually not relied-on by programmers
Exploits: Abusing Program Assets

**Benign Use-Case**

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

arg = "gg ez game"
```

![Diagram showing stack with variables and function call](image)
Exploits: Abusing Program Assets

Malicious Use-Case

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

arg = “AAAAA...\xf0\x01\x01\x00”

Address of `target()`

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
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

- No MTDs
  - Probe
  - Weaponize
  - Attack

- Load-time MTDs
  - Probe
  - ...
  - Weaponize
  - Attack

$t$
Attacks vs. (Re-)Randomization

- **No MTDs**
  - Probe → Weaponize → Attack

- **Load-time MTDs**
  - Probe → ... → Weaponize → Attack

- **Re-Randomized MTDs (Churn)**
  - Probe → Probe → Probe → ... → Weaponize → Attack

$t$
Introduction

Morpheus Architecture

Evaluations

Parting Thoughts
Introduction

Parting Thoughts

Morpheus Architecture

Evaluations

Parting Thoughts
Morpheus: Ensemble of MTDs

Encryption: Obfuscate information assets

Displacement: Shift code & data segments
Tagging & Attack Detection

• Tags enable behavior tracking

• Illegal Ops
  • Clearly dangerous

• Suspicious Ops
  • Normal programs may perform
  • May be probes or attacks

Operand Tags
- Opcode
- Illegal Tags
  • Executing non-code
  • Jump to non-CP
  • ...
- Suspicious Tags
  • CP arithmetic
  • Arith. overflow
  • ...

Attack Detector
- Terminate Program
- Churn

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

QARMA [ToSC’17]
64-bit block cipher
128-bit keys

Addr of Asset
Asset

Encrypted/Decrypted Asset

15
Churning EMTDs

Stale: Under OLD key
Clean: Updated to NEW key
Churning EMTDs

Program

Flush

Churn

Key Gen

Reg

Asset Updates

Clean

Threshold

Stale: Under OLD key

Clean: Updated to NEW key
Churning EMTDs

Churn Period

Program

Flush

Churn

Key Gen

Reg

Clean: Asset Updates

Stale: Under OLD key

Clean: Updated to NEW key
μArch Additions

Tagged Memory
- Tag Propagation
- Attack Detector

Pipeline
- Tag Support
- Attack Detector

Registers
- Tags

L1 D-cache
- Tags

L2 Cache
- Tags

To DRAM
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

Pipeline
- Tag Support
- Attack Detector

Displacement Translation

Tags

Registers

L1 D-cache

ENC/DEC

L2 Cache

To DRAM
μArch Additions

Tagged Memory
- Tag Propagation
- Attack Detector

Displacement
- Translate DAS → VAS

Encryption
- QARMA Engines

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

<table>
<thead>
<tr>
<th>Core Type</th>
<th>MinorCPU (InO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Freq.</td>
<td>2.5GHz</td>
</tr>
<tr>
<td>L1 I$</td>
<td>32KB 2-cycle</td>
</tr>
<tr>
<td>L1 D$</td>
<td>32KB 2-cycle</td>
</tr>
<tr>
<td>L2 Unified</td>
<td>256KB 20-cycle</td>
</tr>
<tr>
<td>Tag Cache</td>
<td>4KB</td>
</tr>
</tbody>
</table>
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

- AnC Address De-randomization
- High bit probes
- Code search
- Blind code search

251s to penetrate a Morpheus system with high entropy & no churn!
Effects of Churn Period

![Graph showing the effects of churn period on percent slowdown. The graph compares SPEC'06 Worst-Case (403.gcc), SPEC'06 Average, MiBench Worst-Case (dijkstra), and MiBench Average. The x-axis represents churn period with values: None, 200ms (1255x), 100ms (2510x), 50ms (5020x), 25ms (10040x), 10ms (25100x), and Cont. (90320x). The y-axis represents percent slowdown with values ranging from 0% to 35%. The graph indicates that the average slowdown is less than 1%.](image-url)
Evaluation Summary

Keys change $5020\times$ faster than time-to-penetrate with advanced probes

Low performance impact (<1%) on system

With network latencies of $\sim 1\text{ms}/36\text{miles}$, churn invalidates information before attackers can use it
Introduction

Morpheus Architecture

Parting Thoughts

Evaluations
Introduction

Morpheus Architecture

Evaluations

Parting Thoughts
Limitations of Morpheus

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

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

Future Work

Churn relative distance

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
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

<table>
<thead>
<tr>
<th></th>
<th>SiFive U54-MC</th>
<th>Morpheus</th>
</tr>
</thead>
<tbody>
<tr>
<td>U54 w/ Caches</td>
<td>2.249 mm²</td>
<td>2.249 mm²</td>
</tr>
<tr>
<td>+ Tagged Memory</td>
<td>-</td>
<td>0.084 mm²</td>
</tr>
<tr>
<td>+ QARMA</td>
<td>-</td>
<td>0.044 mm²</td>
</tr>
<tr>
<td>+ Churn Support</td>
<td>-</td>
<td>0.082 mm²</td>
</tr>
<tr>
<td>Total</td>
<td>2.249 mm²</td>
<td>2.459 mm²</td>
</tr>
</tbody>
</table>
Full μArch