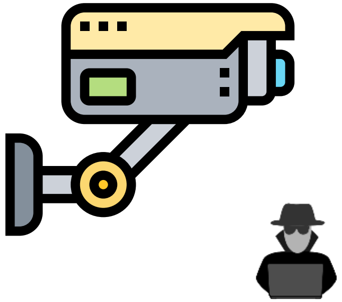
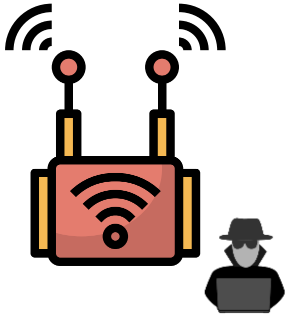
Randezvous: Making Randomization Effective on MCUs

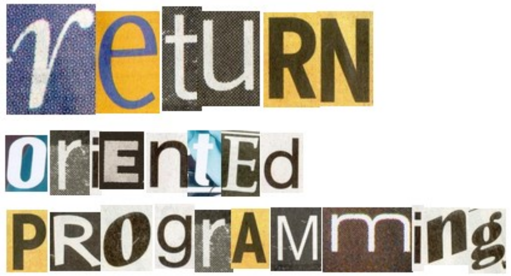
Zhuojia Shen, Komail Dharsee, and John Criswell

University of Rochester

Embedded Systems are Everywhere







Small, resource-constrained microcontrollers (MCUs)

- Limited memory (KBs to MBs)
- No virtual memory and MMU
- Running memory-unsafe C code
- Execute in privileged mode

Vulnerabilities => control-flow hijacking attacks (e.g., ROP)

=> entire system controlled

Prior Solutions & Limitations

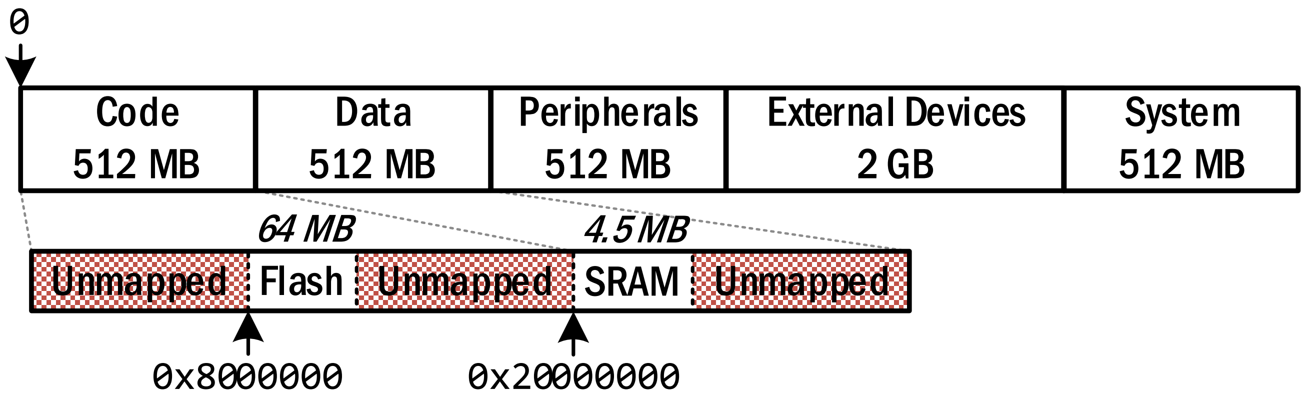
Control-flow integrity (CFI)

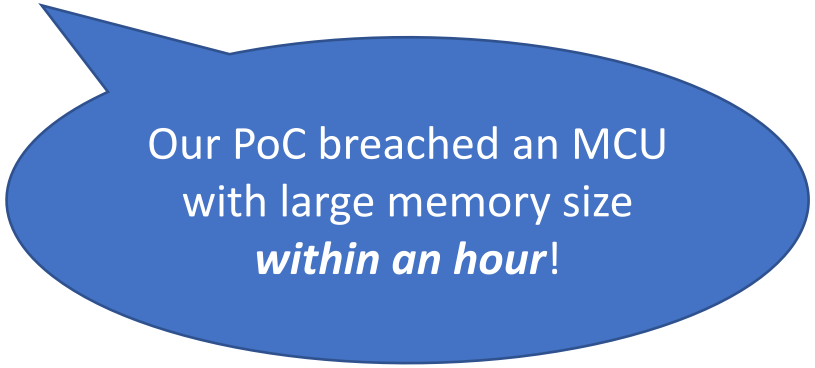
- Still susceptible to advanced attacks [CFBending@UsenixSec'15, CtrlJujutsu@CCS'15]
- High overhead (8.1% - 513%) [CaRE@RAID'17, µRAI@NDSS'20, Silhouette@UsenixSec'20]

Prior Solutions & Limitations

Randomization + execute-only memory (XOM)

- Limited address space => cannot withstand brute force attacks
- Defeated by control data disclosure & spraying attacks





Our Solution: Randezvous

Holistic diversification-based control-flow hijacking defense for MCUs

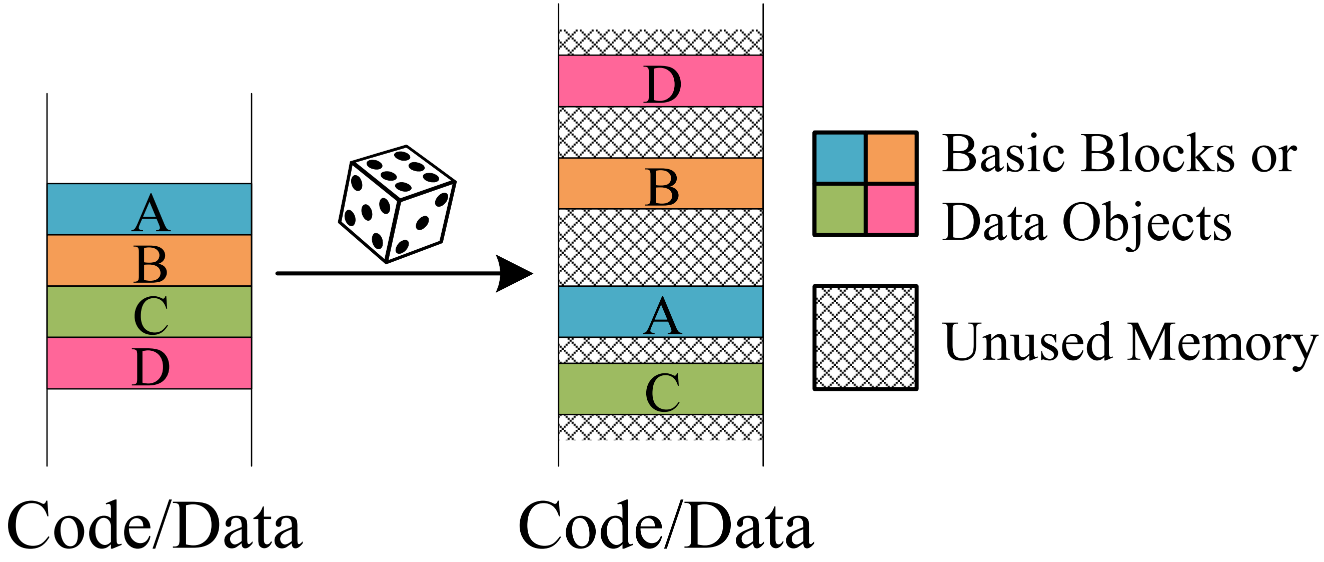
- Targets popular ARMv7-M & ARMv8-M
- Based on randomization + XOM
- Protect in-memory control data from leakage & tampering
- Improve limited entropy against brute force & spraying attacks

Outline

- Design & Implementation
  - Randomization + XOM
  - Control Data Protection
  - Entropy Improvement
- Security Evaluation
- Performance Evaluation
- Summary

Randomization + XOM

- Compile-time code/data layout randomization[EPOXY@Oakland'17]
  - Fill unused code memory w/ trap instructions (`udf` on ARM)



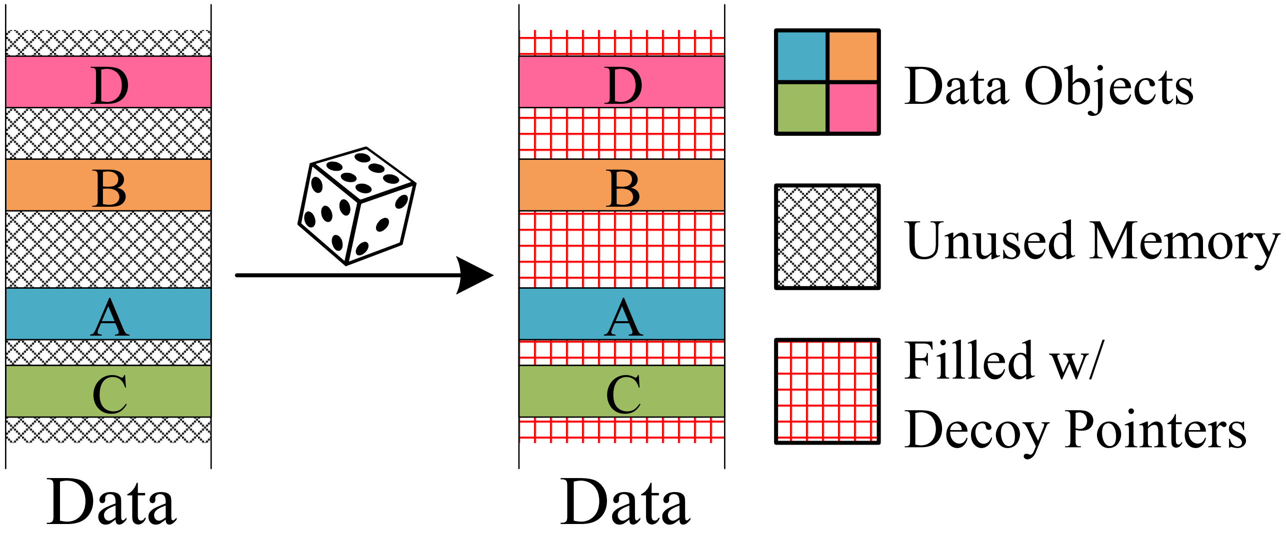
- PicoXOM[SecDev'20]: efficient XOM using ARM's MPU & DWT

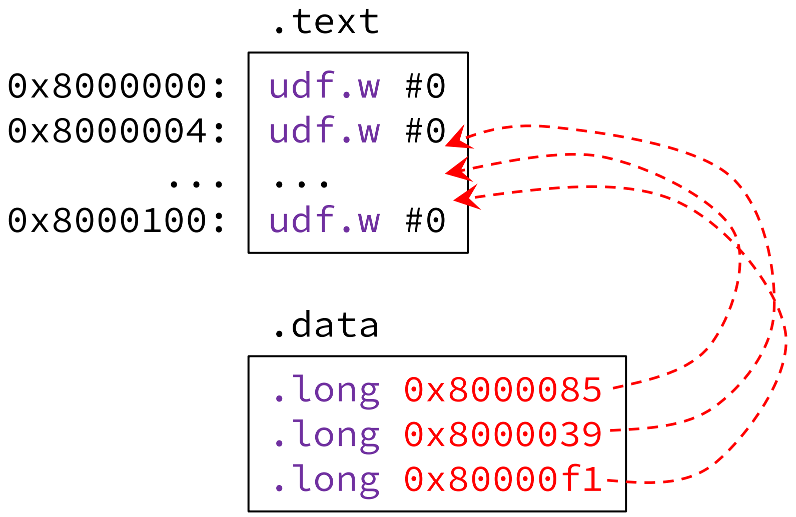
Outline

- Design & Implementation
  - Randomization + XOM
  - Control Data Protection
  - Entropy Improvement
- Security Evaluation
- Performance Evaluation
- Summary

Decoy Pointers

- Pointers to *randomly selected trap instructions*
- Key insight: real control data unidentifiable when camouflaged among numerous decoy pointers

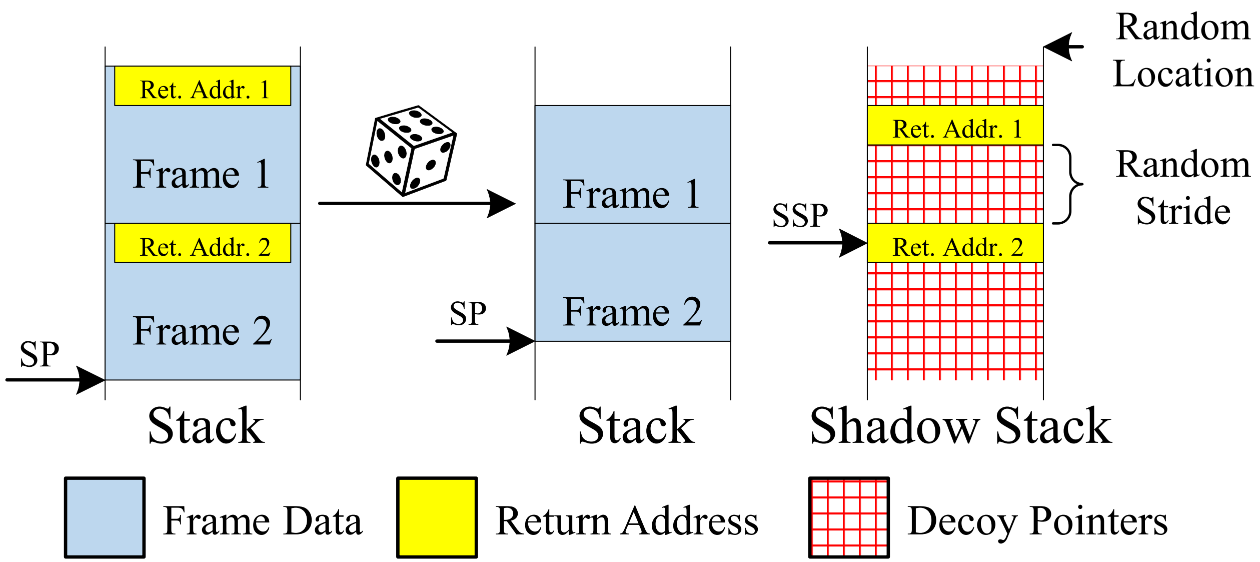




Example of decoy pointers

Diversified Shadow Stack

- Random location in `.data` section
- Filled w/ decoy pointers
- Random strides between return addresses



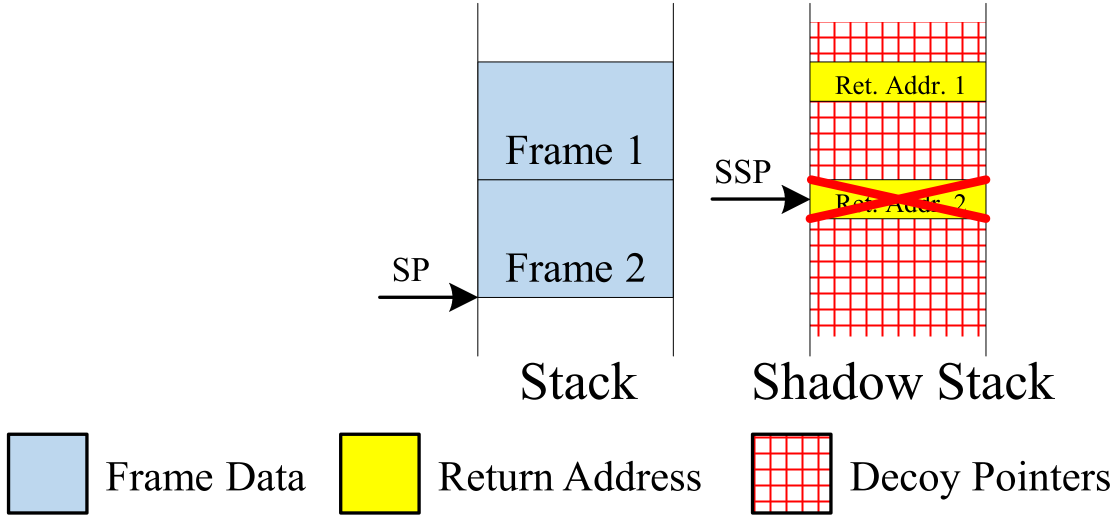

```
push {r4,lr}
...
pop  {r4,pc}
```

```
str  lr,[r8],#32
add  r8,r8,r9
push {r4}
...
pop  {r4}
sub  r8,r8,r9
ldr  pc,[r8,#-32]!
```

Example of diversified shadow stack transformation w/ r8 as SSP, dynamic stride in r9, and static stride of 32

Return Address Nullification

- Nullify stale return address w/ decoy pointer before returning




```
str  lr,[r8],#32
add  r8,r8,r9
push {r4}
...
pop  {r4}
sub  r8,r8,r9
ldr  pc,[r8,#-32]!
```

```
str  lr,[r8],#32
add  r8,r8,r9
push {r4}
...
pop  {r4}
sub  r8,r8,r9
ldr  lr,[r8,#-32]!
movw ip,#decoy-lo16
movt ip,#decoy-hi16
str  ip,[r8]
bx   lr
```

Example of return address nullification transformation

Local-to-Global Variable Promotion

- Promote local function pointers to globals in `.data` section

```
foo() {
  void (*fp)();
  ...
}
```

```
static void (*fp)();

foo() {
  ...
}
```

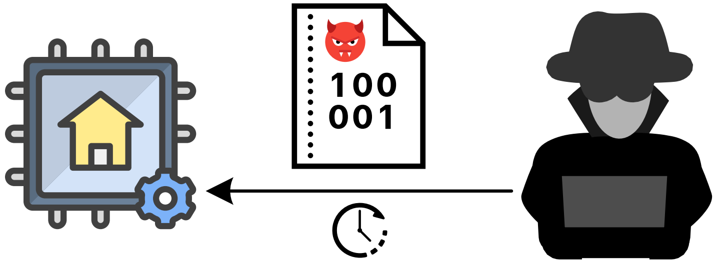
Example of local-to-global variable promotion transformation

Outline

- Design & Implementation
  - Randomization + XOM
  - Control Data Protection
  - Entropy Improvement
- Security Evaluation
- Performance Evaluation
- Summary

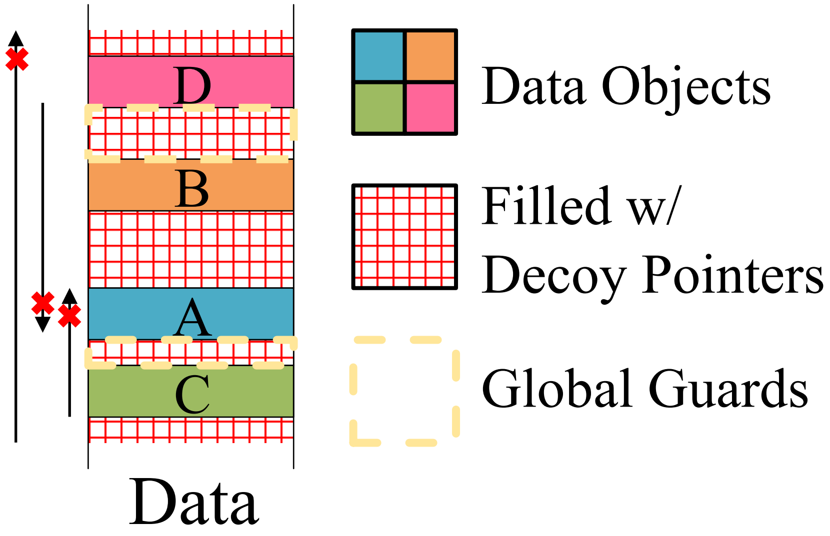
Delayed Reboot

- Make each successive reboot caused by a trap take longer
- For i-th reboot, delay D_i increases as i increases until a threshold
- Artificially reduce # of attack attempts in a give time period
- Exchange availability for security



Global Guards

- Adaptation of guard memory
- Set random pieces of unused memory in `.data` section as read-only
- Mitigate control data spraying attacks



Outline

- Design & Implementation
  - Randomization + XOM
  - Control Data Protection
  - Entropy Improvement
- Security Evaluation
- Performance Evaluation
- Summary

Security Evaluation

Statistical modeling

- Brute force return-into-libc attacks w/ control data disclosure
- Calculate entropy & expected time to resist the attacks w/ equations
- On 3 different-sized MCUs
- **Randezvous is able to resist the attacks**

Exploit analysis

- PoC exploit
- Real-world CVE exploit based on CVE-2021-27421
- **Randezvous withstood persistent attacks for longer than 3 days**

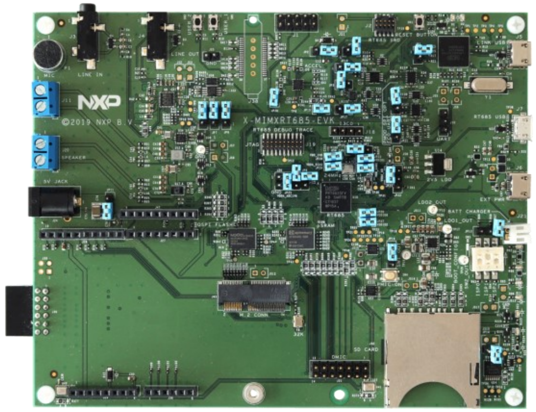
Outline

- Design & Implementation
  - Randomization + XOM
  - Control Data Protection
  - Entropy Improvement
- Security Evaluation
- Performance Evaluation
- Summary

Experimental Setup

Hardware

- NXP MIMXRT685-EVK board
- ARM Cortex-M33 processor @ 300 MHz
- 4.5 MB SRAM & 64 MB flash memory
- TRNG & SD card slot



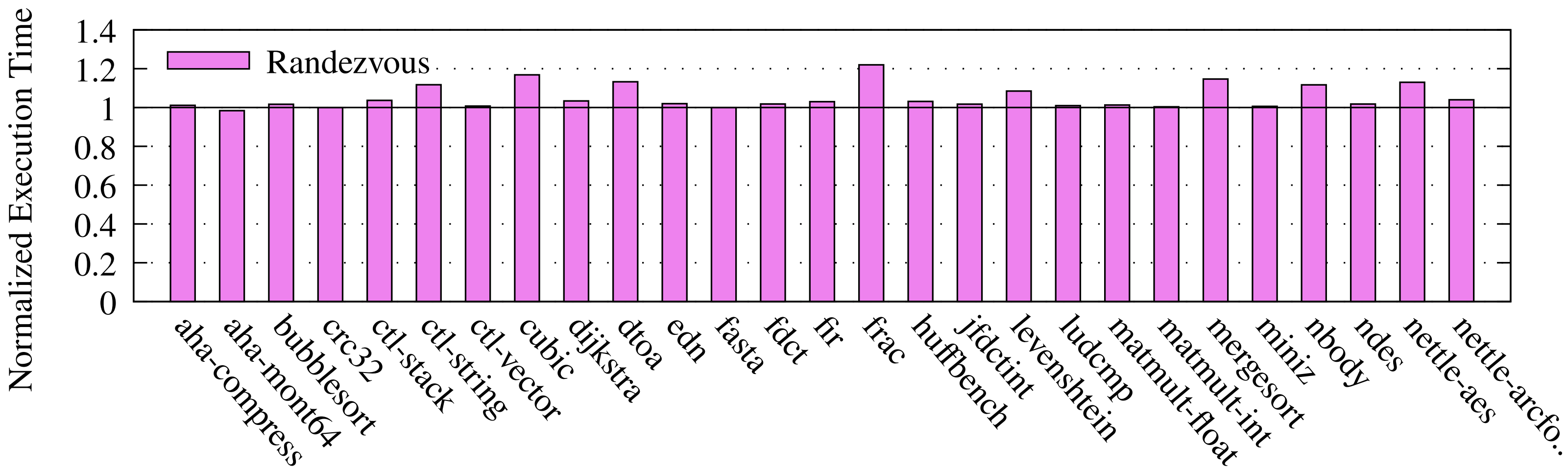
Benchmarks & applications

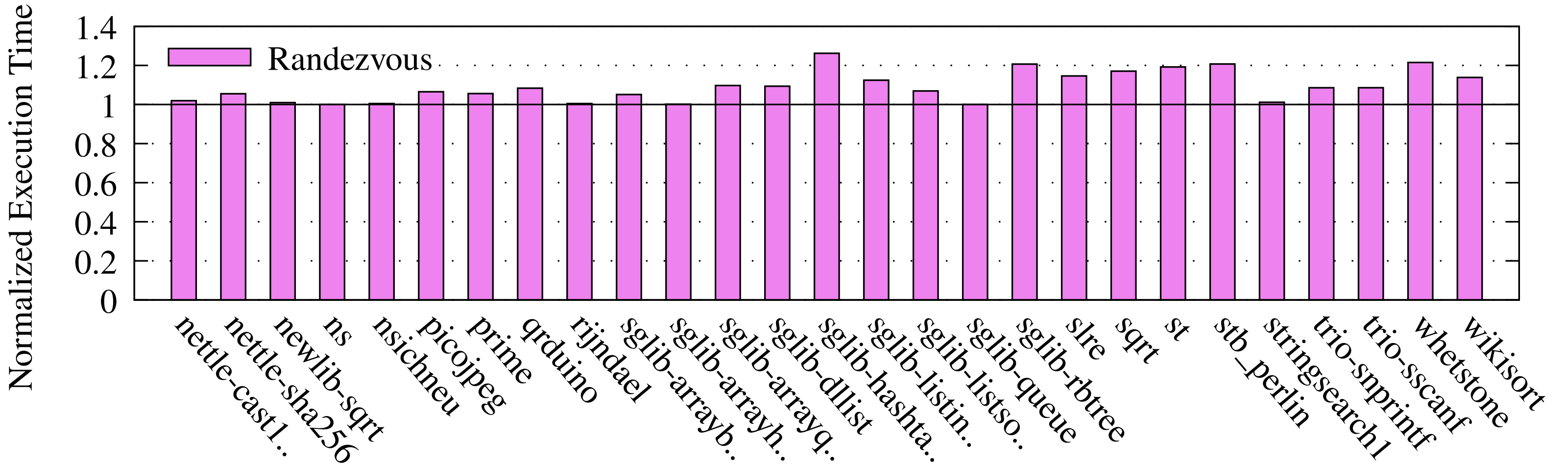
- BEEBS
- CoreMark-Pro
- MbedTLS-Benchmark
- PinLock
- FatFs-SD

Configurations

- Baseline: LLVM/Clang 11.0.1 `-Os -flto`
- Randezvous: Baseline + all Randezvous features w/ all seeds set to zero

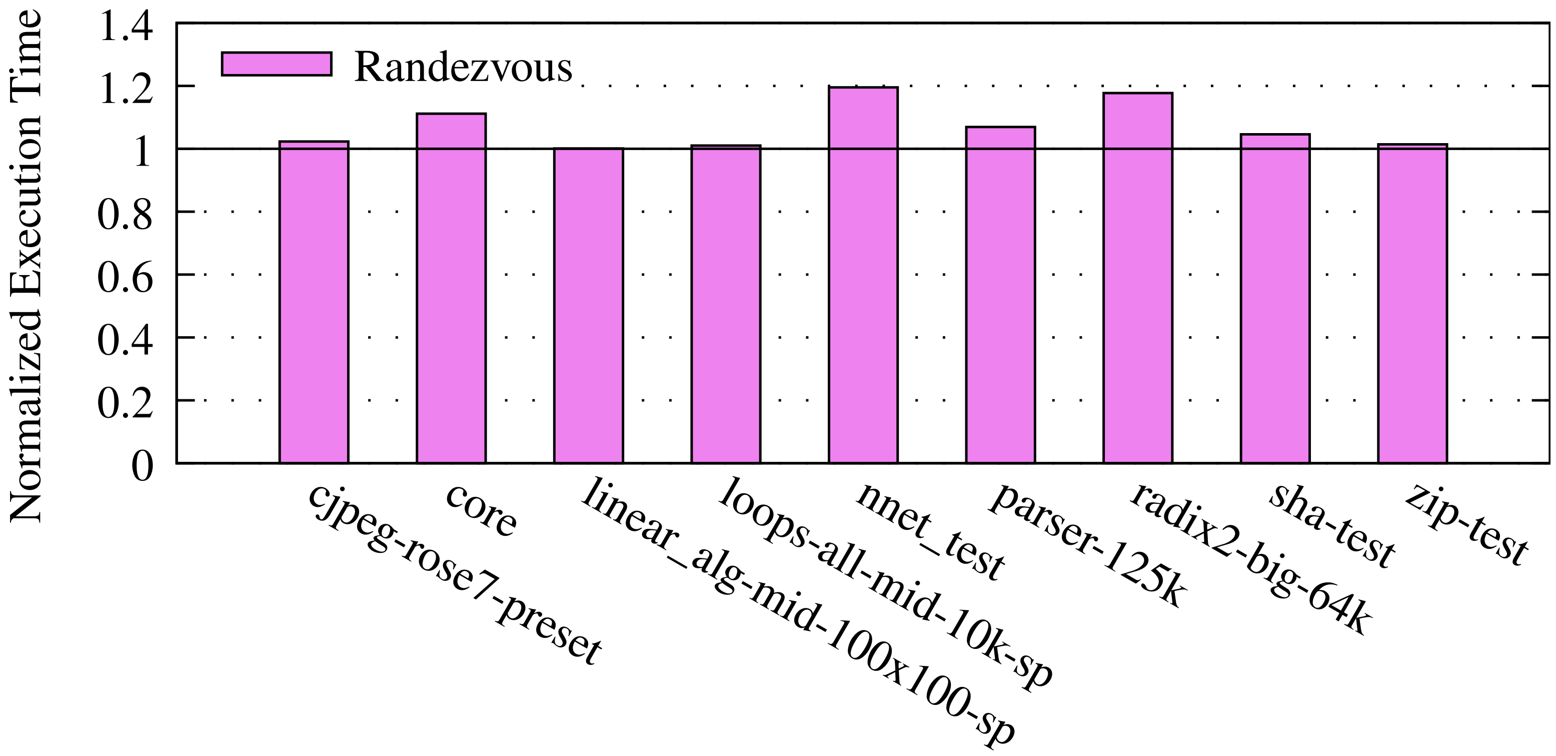
BEEBS Execution Time (Lower is Better)





6.9% overhead

CoreMark-Pro Execution Time (Lower is Better)



7.0% overhead

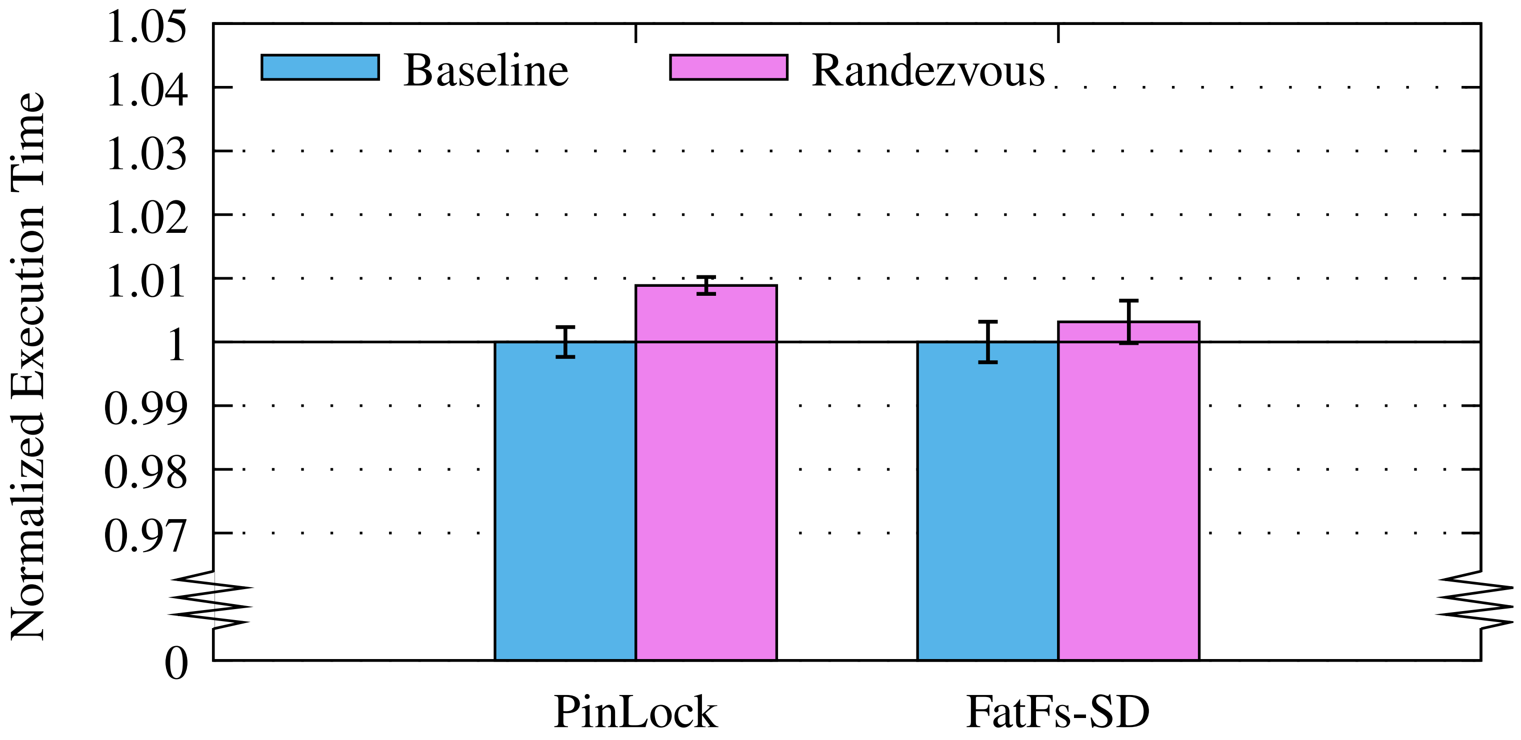
MbedTLS-Benchmark Latency & Throughput

Latency of 21 Cryptographic Algorithms (Lower is Better)	Randezvous (x)
Min	1.009
Max	1.145
Geomean	**1.055**

Throughput of 37 Cryptographic Algorithms (Higher is Better)	Randezvous (x)
Min	0.873
Max	1.011
Geomean	**0.955**

~5% overhead

Application Execution Time (Lower is Better)

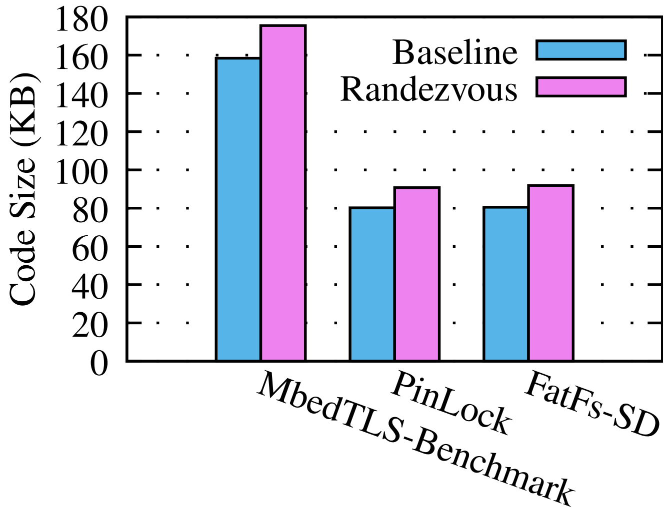


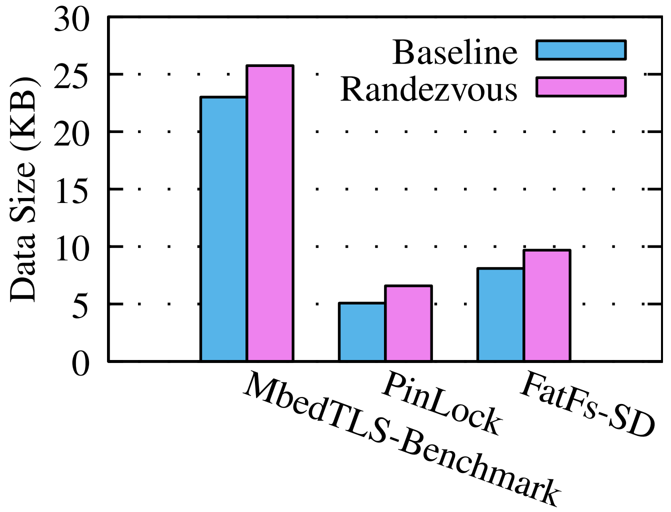
0.6% overhead

Memory Usage (Lower is Better)

BEEBS	Baseline Code (byte)	Baseline Data (x)	Randezvous Code (x)	Randezvous Data (x)
Min	60,474	70,212	1.133	1.079
Max	75,260	85,278	1.162	1.318
Geomean	—	—	**1.158**	**1.212**

CoreMark-Pro	Baseline Code (byte)	Baseline Data (byte)	Randezvous Code (x)	Randezvous Data (x)
Min	72,852	5,887	1.128	1.001
Max	104,978	1,383,731	1.151	3.855
Geomean	—	—	**1.142**	**1.275**





15.4% code size overhead

22.0% data size overhead

Outline

- Design & Implementation
  - Randomization + XOM
  - Control Data Protection
  - Entropy Improvement
- Security Evaluation
- Performance Evaluation
- Summary

Summary

- Randezvous: randomization-based control-flow hijacking defense for ARM MCUs
- Resist control data disclosure & aid MCUs' entropy w/ low costs
- Open source at https://github.com/URSec/Randezvous
- Artifact evaluated