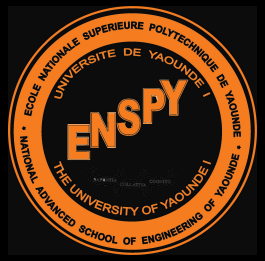


Scientific Days Cameroon

December 20, 2022

Efficient Buffer Overflow Mitigation In Virtualized Clouds Using Intel EPT-based Sub-Page Write Protection Support

Stella Bitchebe

Ph.D. Candidate, Université Côte d'Azur & ENS Lyon, France

Advisor: Pr. Alain Tchana

bitchebe@i3s.unice.fr

Context: Buffer Overflow in Virtualized Clouds

Why the Cloud?

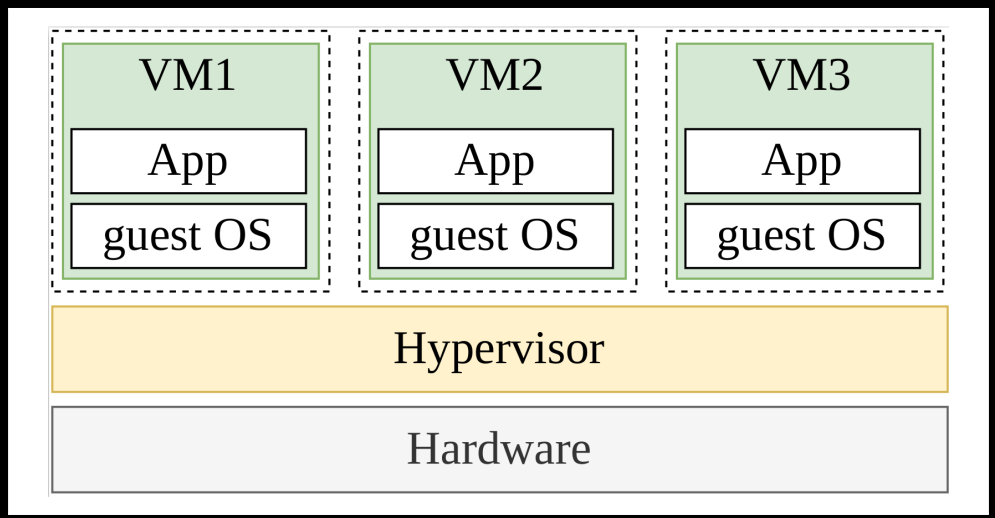
- Attractive costs (e.g., *Faas*)
- Management tasks simplification (e.g., *SaaS, IaaS*)
- Efficiency of Cloud computing

Context: Buffer Overflow in Virtualized Clouds

Why the Cloud?

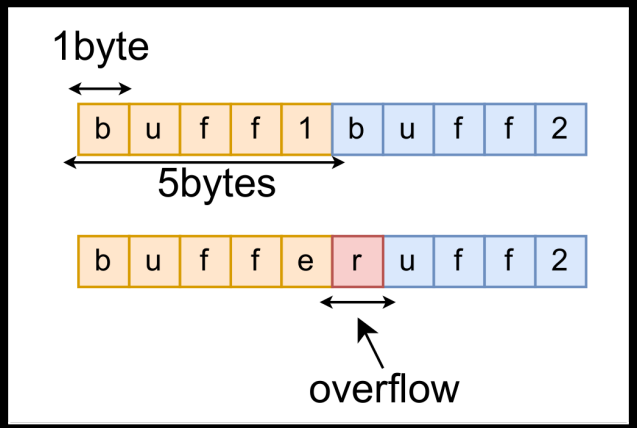
- Attractive costs (e.g., *Faas*)
- Management tasks simplification (e.g., *SaaS*, *IaaS*)
- Efficiency of Cloud computing

Virtualized Cloud Infrastructure



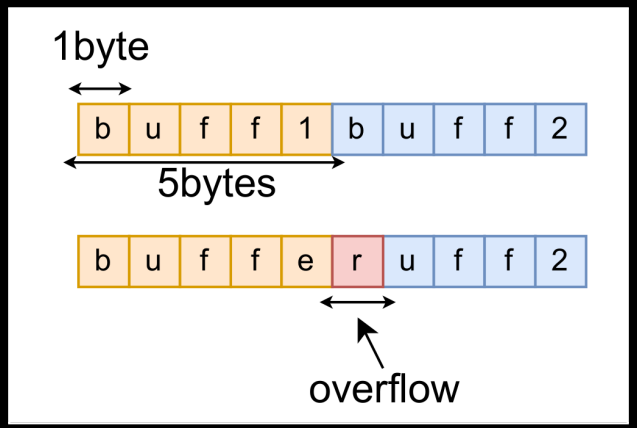
Context: Buffer Overflow in Virtualized Clouds

What is buffer overflow?



Context: Buffer Overflow in Virtualized Clouds

What is buffer overflow?

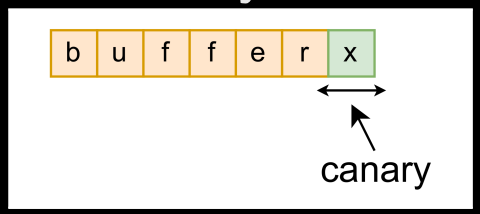


Importance of Buffer Overflow

- 70% Google Chrome's bugs
- 70% of Microsoft vulnerabilities
- Top vulnerability in 2022

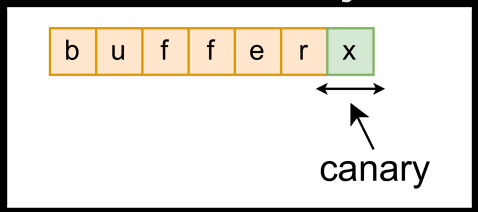
State-of-the-art Techniques

- Canary

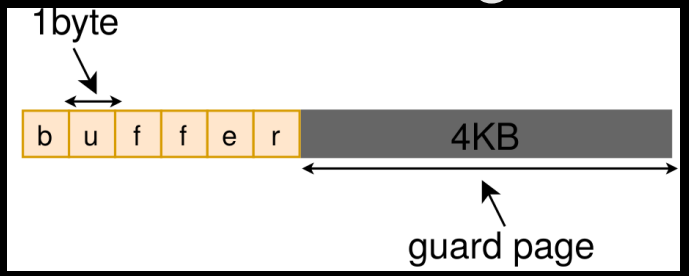


State-of-the-art Techniques

- Canary

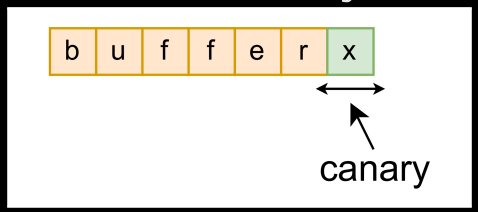


- Guard Page

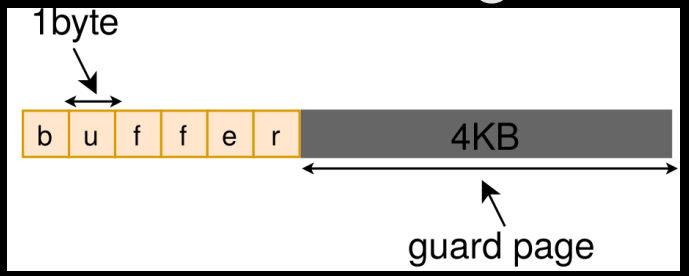


State-of-the-art Techniques

- Canary



- Guard Page



- Hardware Capabilities

CHERI: Capabilities Hardware Enhance RISC-V Instructions

Secure Allocators: SlimGuard

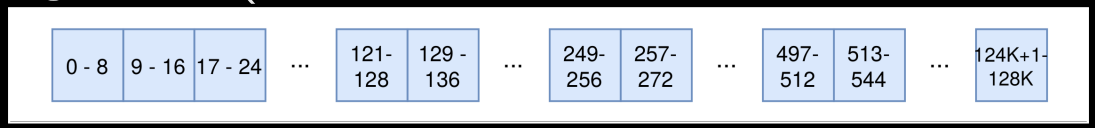
Overview

- State-of-the-art BIBOP (Big Bag Of Pages) secure allocator
- Published in 2019
- Has proven more memory-efficient than other state-of-the-art BIBOP allocators (Guarder, FreeGuard, etc.)
- Available and functional code

Secure Allocators: SlimGuard

Overview

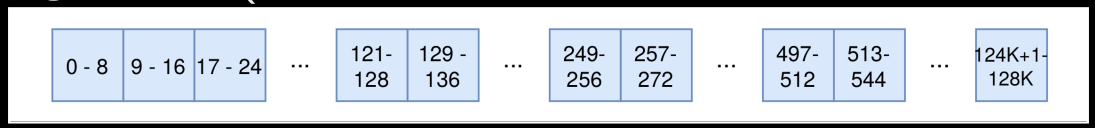
- State-of-the-art BIBOP (Big Bag Of Pages) secure allocator
- Published in 2019
- Has proven more memory-efficient than other state-of-the-art BIBOP allocators (Guarder, FreeGuard, etc.)
- Available and functional code
- Fine-grained bag sizes (not size-of-two -2, 8, 16, ...- like others)



Secure Allocators: SlimGuard

Overview

- State-of-the-art BIBOP (Big Bag Of Pages) secure allocator
- Published in 2019
- Has proven more memory-efficient than other state-of-the-art BIBOP allocators (Guarder, FreeGuard, etc.)
- Available and functional code
- Fine-grained bag sizes (not size-of-two -2, 8, 16, ...- like others)



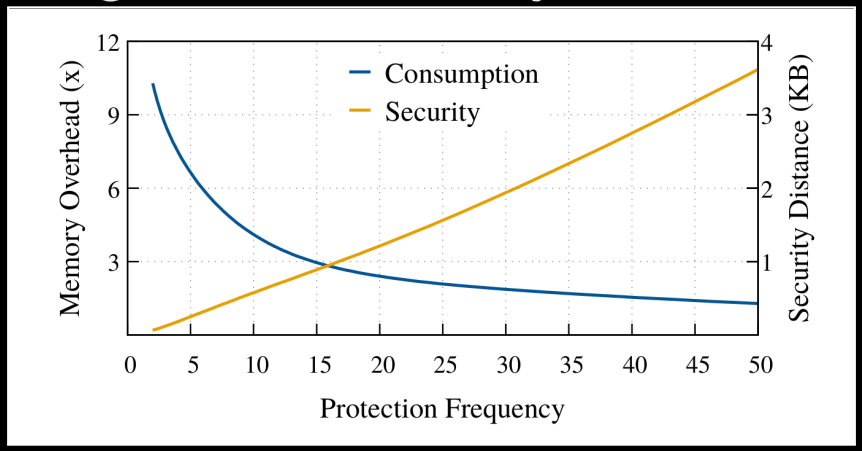
Protection policy: applied at the bag scale

- Configurable proportion of guard pages P
- 1/P allocation pages between 2 guard pages

Secure Allocators: SlimGuard

Limits

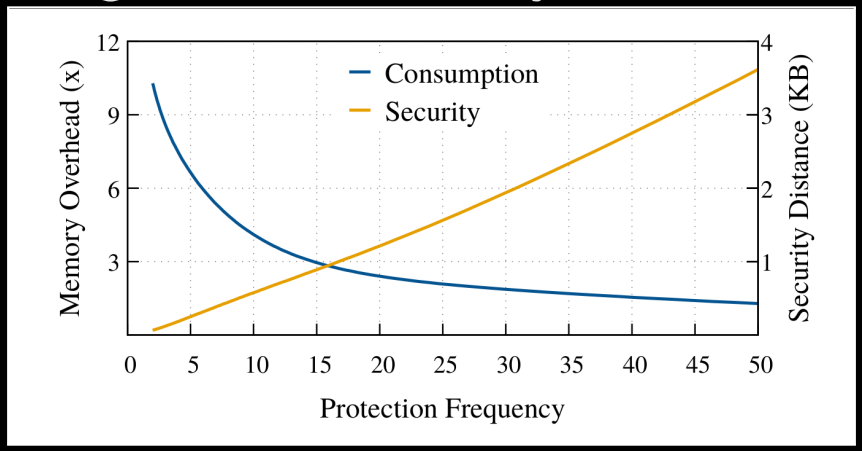
Significant memory overhead



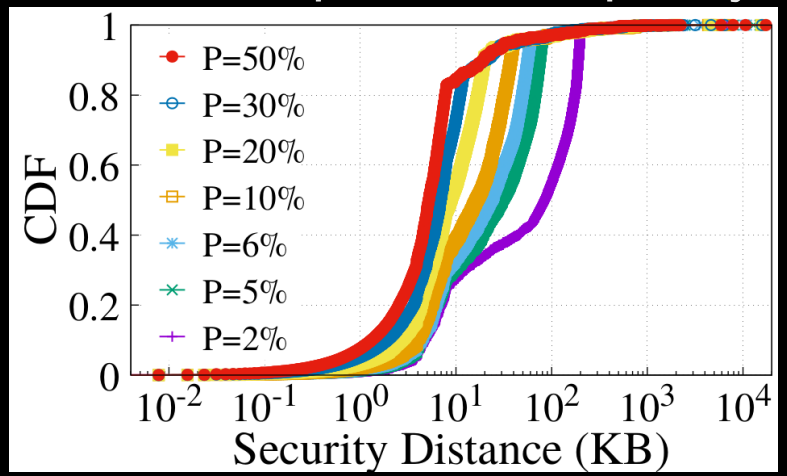
Secure Allocators: SlimGuard

Limits

Significant memory overhead

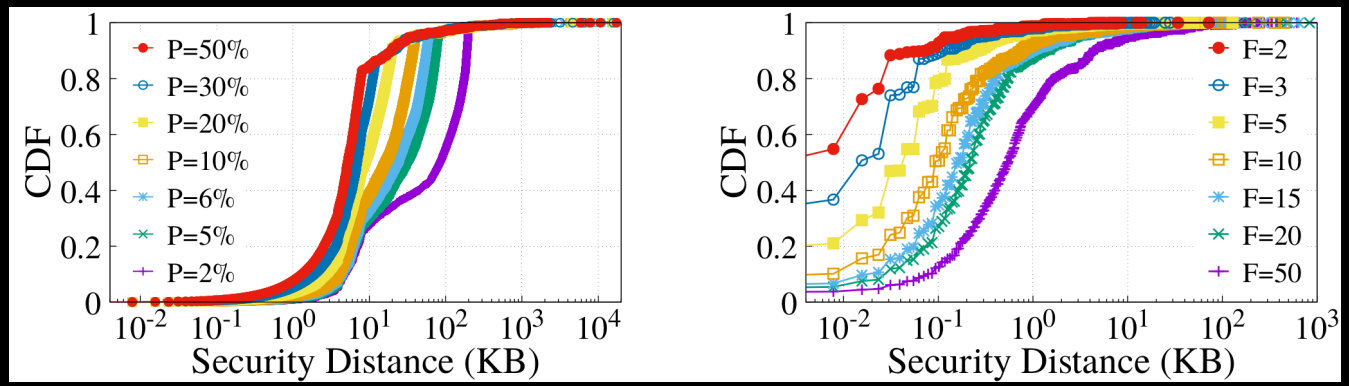


Inefficient protection policy



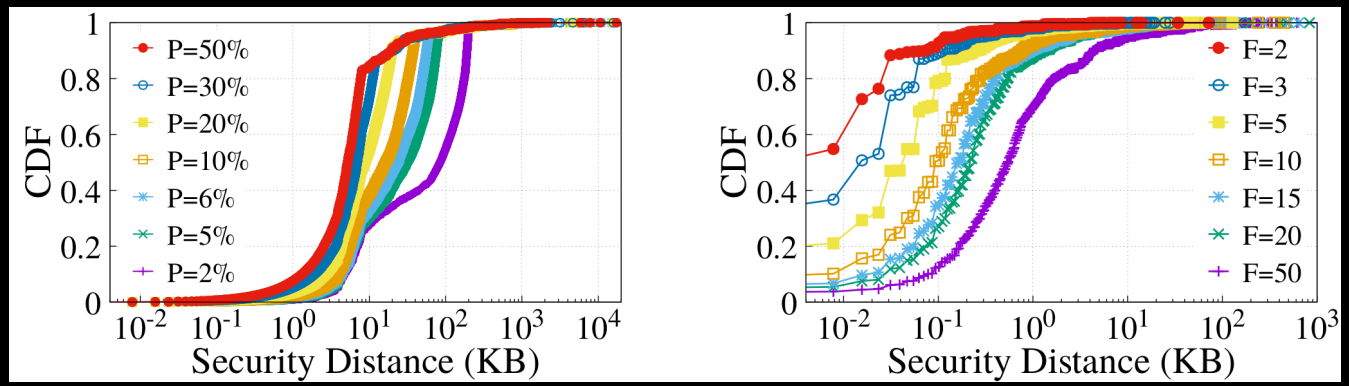
Secure Allocators: SlimGuard

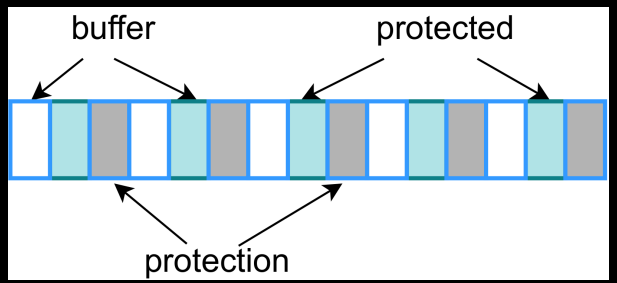
Custom Protection Policy



Secure Allocators: SlimGuard

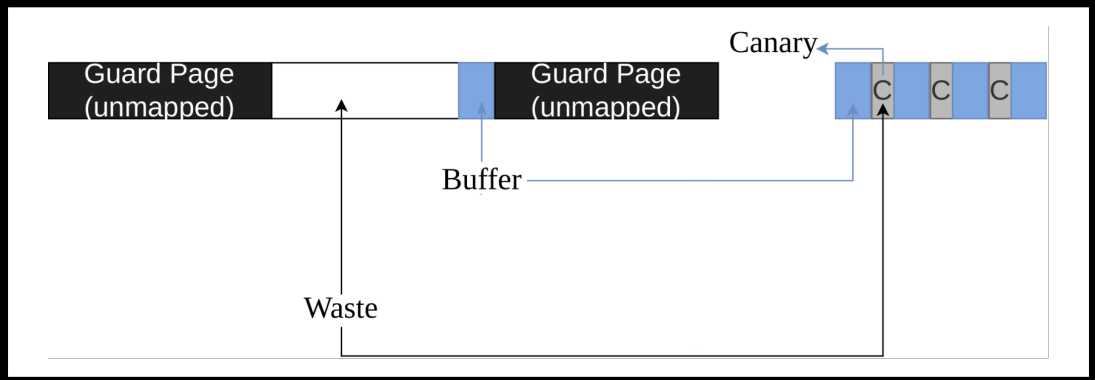
Custom Protection Policy





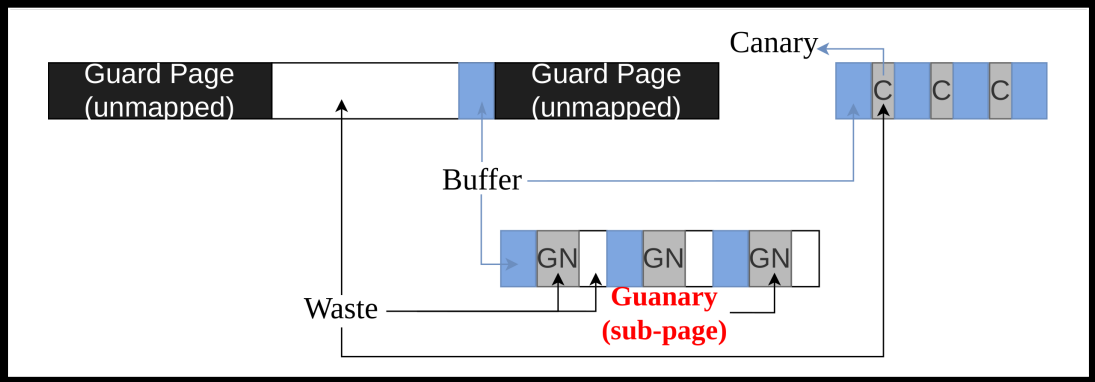
Synchronous Detection vs Memory Overhead

Canary: **modest memory overhead** + asynchronous overflow detection
Guard page: significant memory consumption + **synchronous overflow detection**



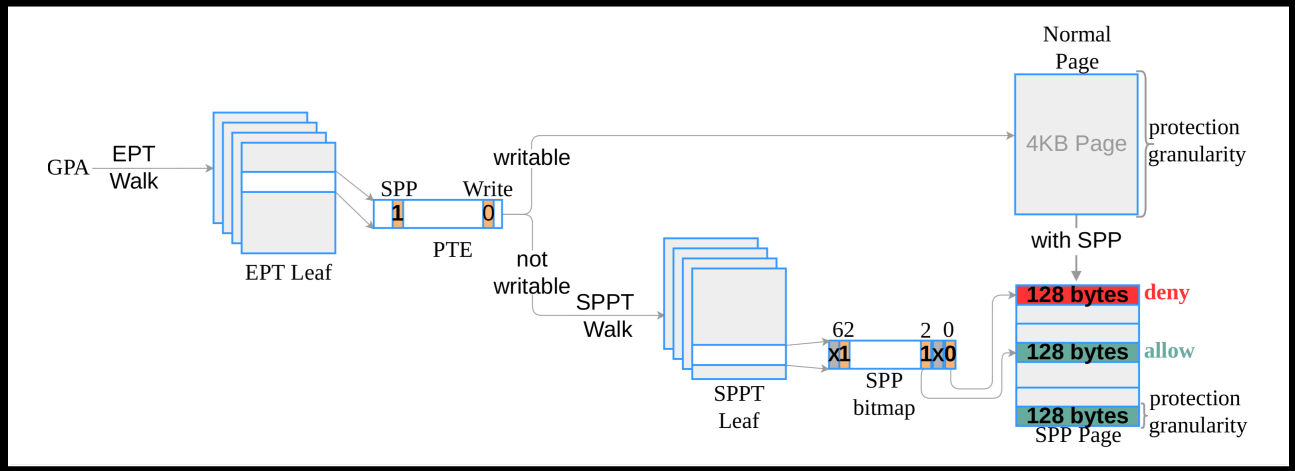
Synchronous Detection vs Memory Overhead

Canary: modest memory overhead + asynchronous overflow detection
Guard page: significant memory consumption + synchronous overflow detection
GuaNary: **modest memory overhead + synchronous overflow detection**



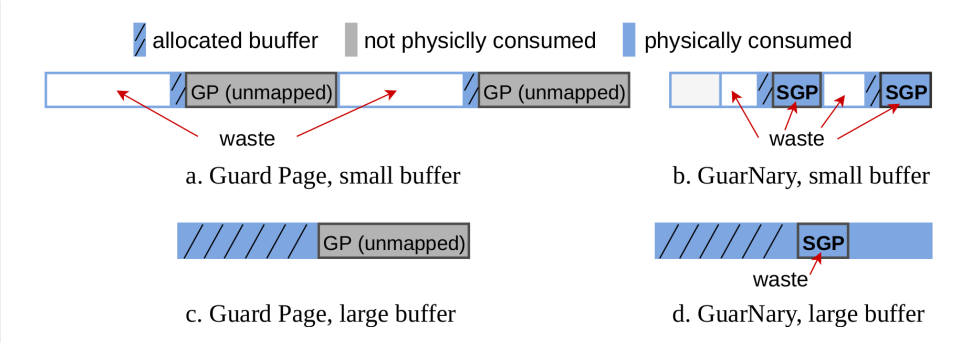
GuaNary: Canary and Guard Page Countermeasure

Intel Sub-Page write Permission (SPP)



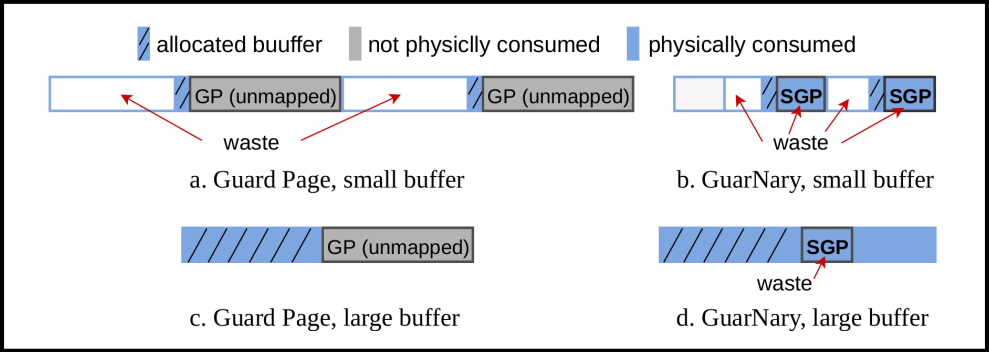
GuaNary: Challenges

C_1: One size does not fit all



GuaNary: Challenges

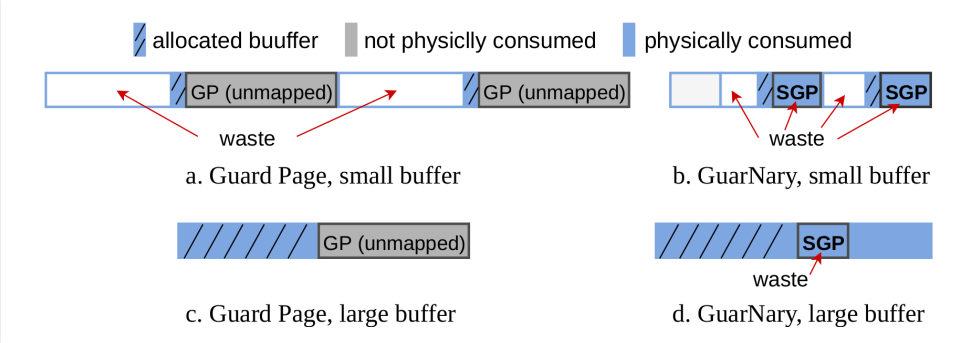
C_1: One size does not fit all



C_2: Costly hypercalls

GuaNary: Challenges

C_1: One size does not fit all

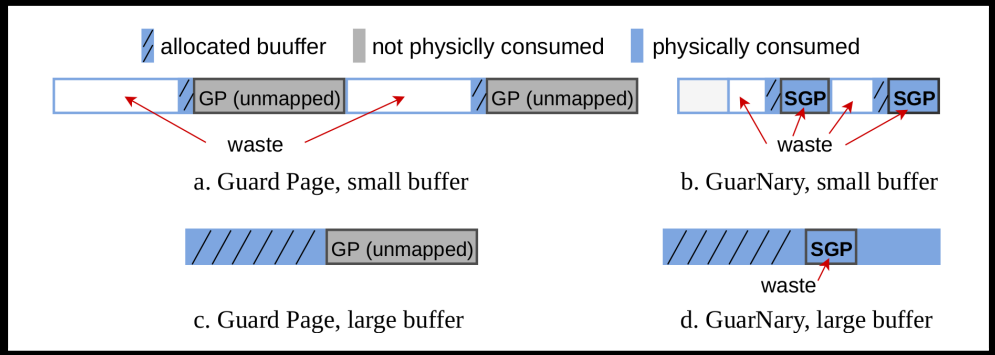


C_2: Costly hypercalls

C_3: Page heterogeneity

GuaNary: Challenges

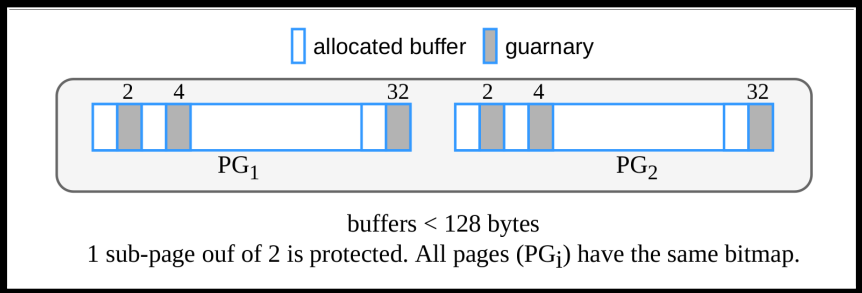
C_1: One size does not fit all



C_2: Costly hypercalls

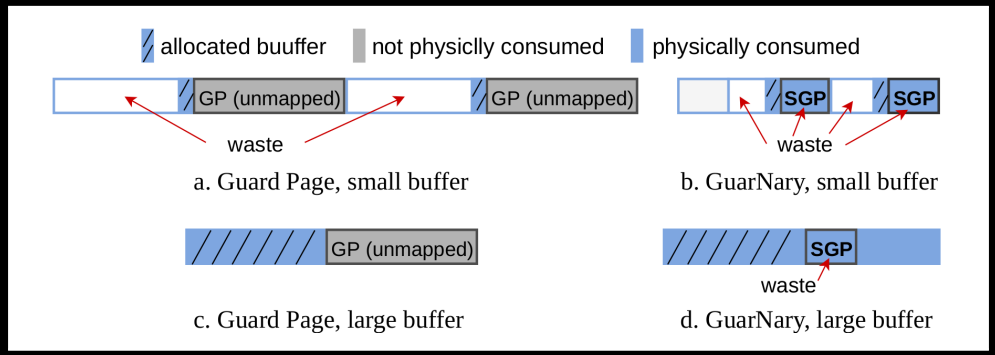
C_3: Page heterogeneity

C_4: SPP Page heterogeneity



GuaNary: Challenges

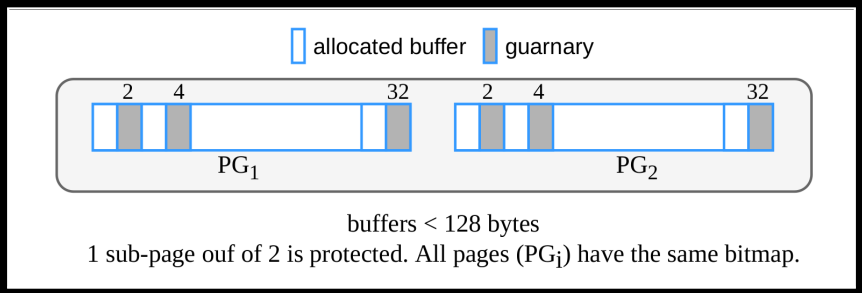
C_1: One size does not fit all

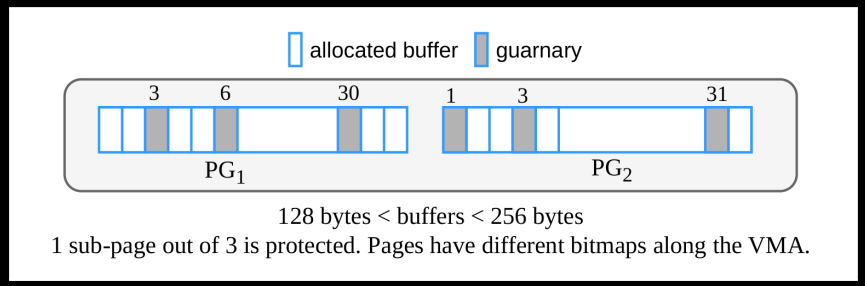


C_2: Costly hypercalls

C_3: Page heterogeneity

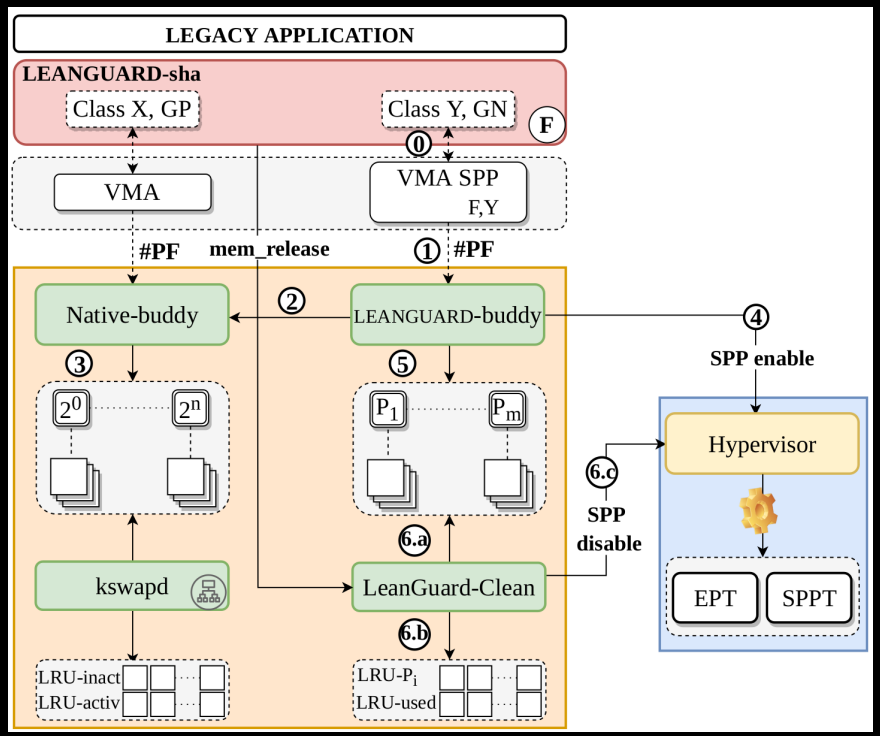
C_4: SPP Page heterogeneity





LeanGuard: GuaNary-based Secure Allocator

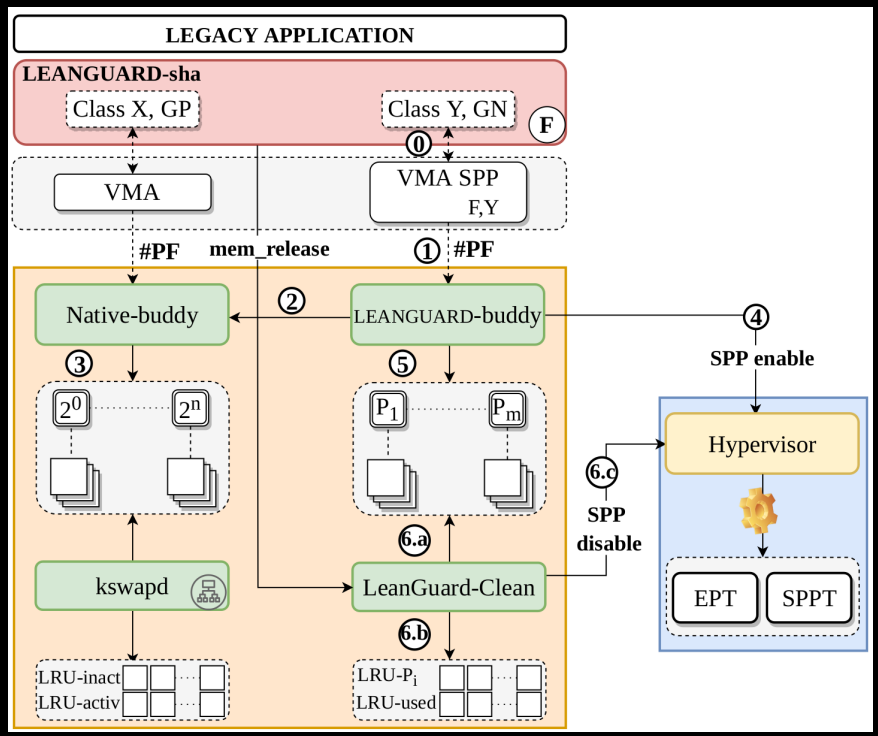
Overview



c_1: use GuaNary or guard page according to the size of buffers

LeanGuard: GuaNary-based Secure Allocator

Overview



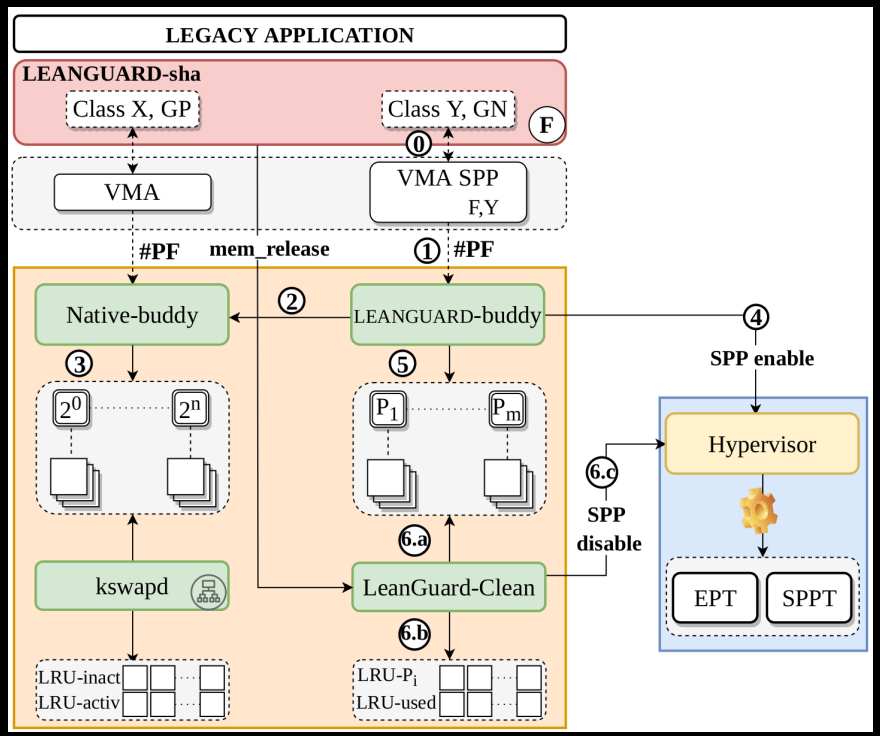
c_1: use GuaNary or guard page according to the size of buffers

c_2: batching hypercalls for optimal performance

c_3: new buddy allocator (LeanGuard-buddy)

LeanGuard: GuaNary-based Secure Allocator

Overview



c_1: use GuaNary or guard page according to the size of buffers

c_2: batching hypercalls for optimal performance

c_3: new buddy allocator (LeanGuard-buddy)

c_4: formula to determine which pattern to use; 202 pools, one per pattern

LeanGuard: GuaNary-based Secure Allocator

Implementation

- Xen hypervisor version 4.10: 600LOCs addition

LeanGuard: GuaNary-based Secure Allocator

Implementation

- Xen hypervisor version 4.10: 600LOCs addition
- Linux OS version 5.11.14: 750LOCs addition

LeanGuard: GuaNary-based Secure Allocator

Implementation

- Xen hypervisor version 4.10: 600LOCs addition
- Linux OS version 5.11.14: 750LOCs addition
- SlimGuard secure allocator: 100LOCs addition

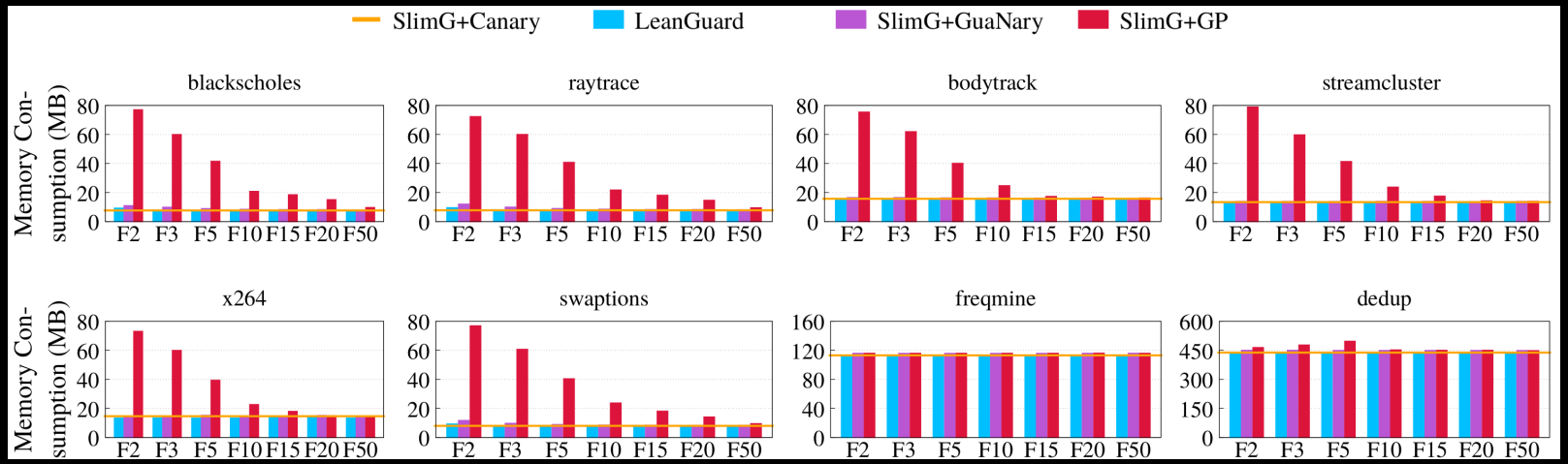
Evaluations

Benchmarks

- Micro-benchmark: 1GB working set application, random buffers allocation from all SlimGuard Classes
- Macro-benchmark: PARSEC suite

Evaluations

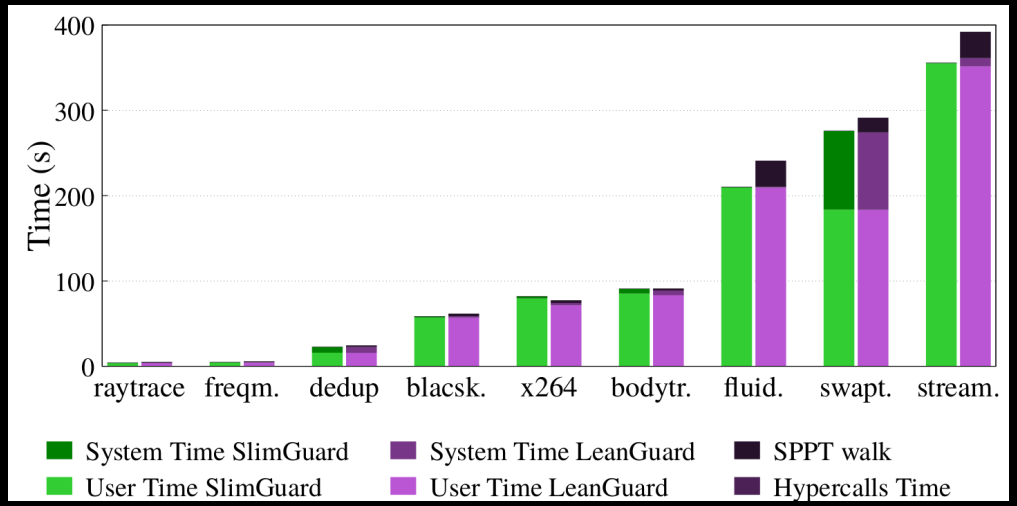
Memory Consumption Results



- GuaNary effectively reduces memory consumption compared to Canary and guard page
- To protect **50%** of the allocated buffers (F=2), LeanGuard on average uses **60%** less memory compared to SlimG+GP
- Using the same amount of memory as SlimG+GP, LeanGuard allows protecting **25**× more buffers than SlimG+GP

Evaluations

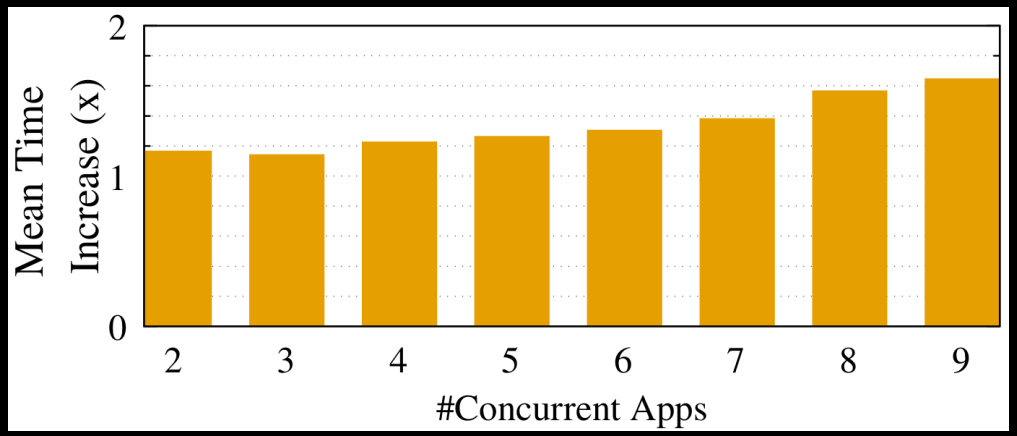
Performance Overhead



- ∼7.7% overhead on average
- The main source of this overhead is the SPP page table walk on TLB miss

Evaluations

Scalability



- LeanGuard-buddy uses a lock mechanism

Conclusion

- We introduce a novel technique to mitigate buffer overflow in virtualized clouds: **GuaNary**
  - based on Intel sub-page write permission (SPP)
  - **modest memory overhead + synchronous overflow detection**
- We propose
  - LeanGuard: a complete software stack that uses GuaNary
  - LeanGuard-secure allocator:
    - consumes **8.3**$\times$ less memory compared to the SlimGuard allocator (up-to-date state-of-the-art)
    - Allows protecting **25**$\times$ more buffers than SlimGuard for the same amount of memory