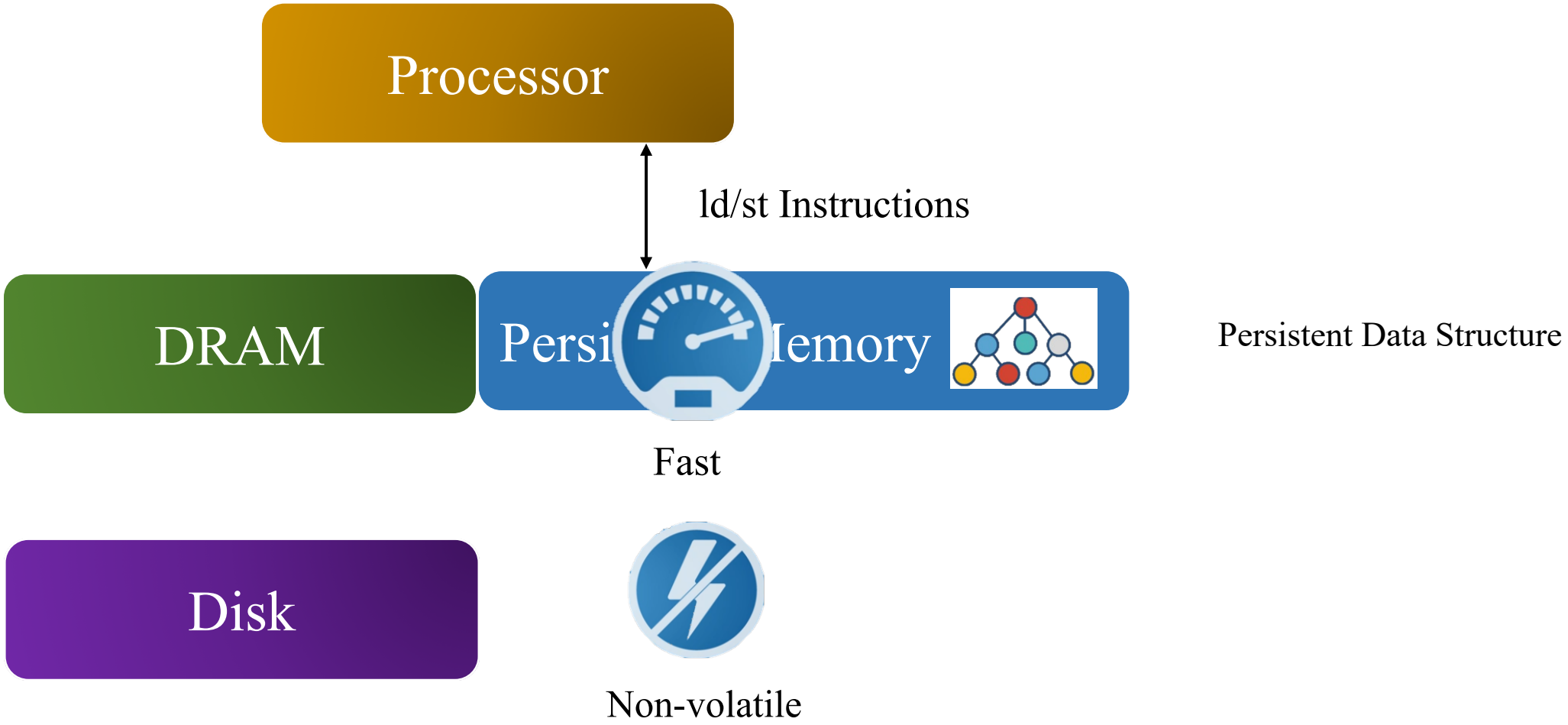
# Hardware-Based Domain Virtualization for Intra-Process Isolation of Persistent Memory Objects

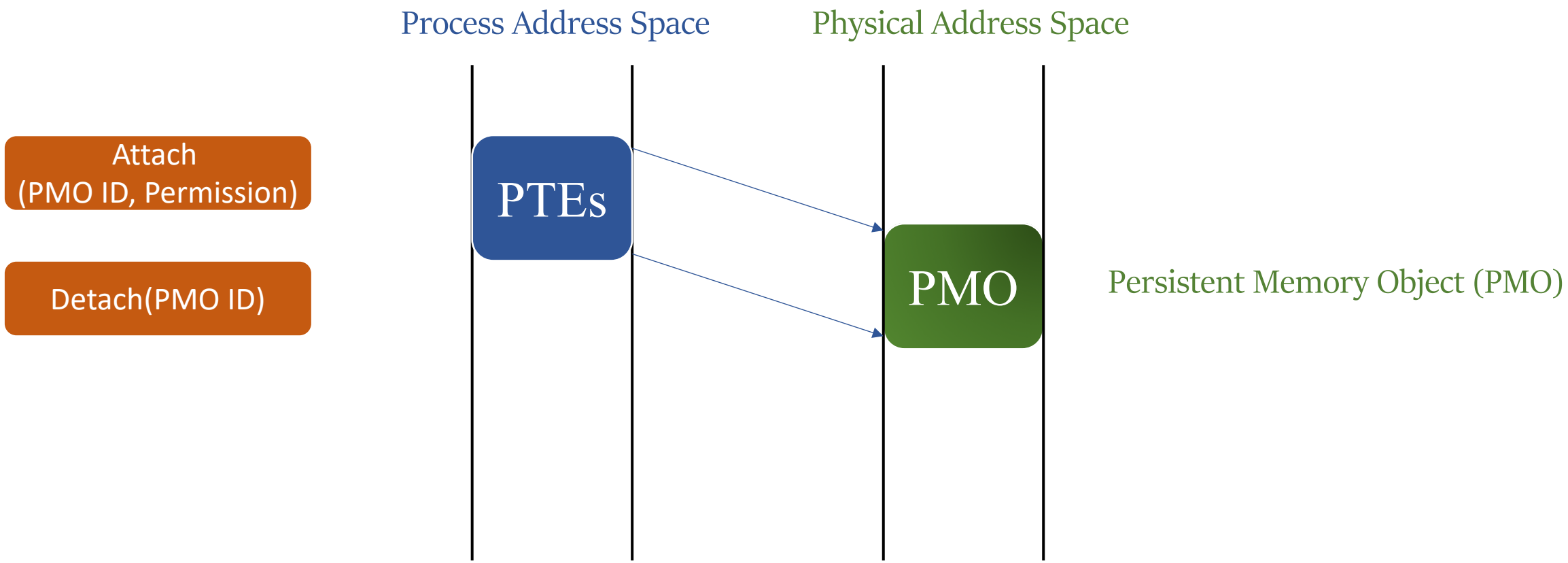
**Yuanchao Xu**, ChenCheng Ye, Yan Solihin, Xipeng Shen



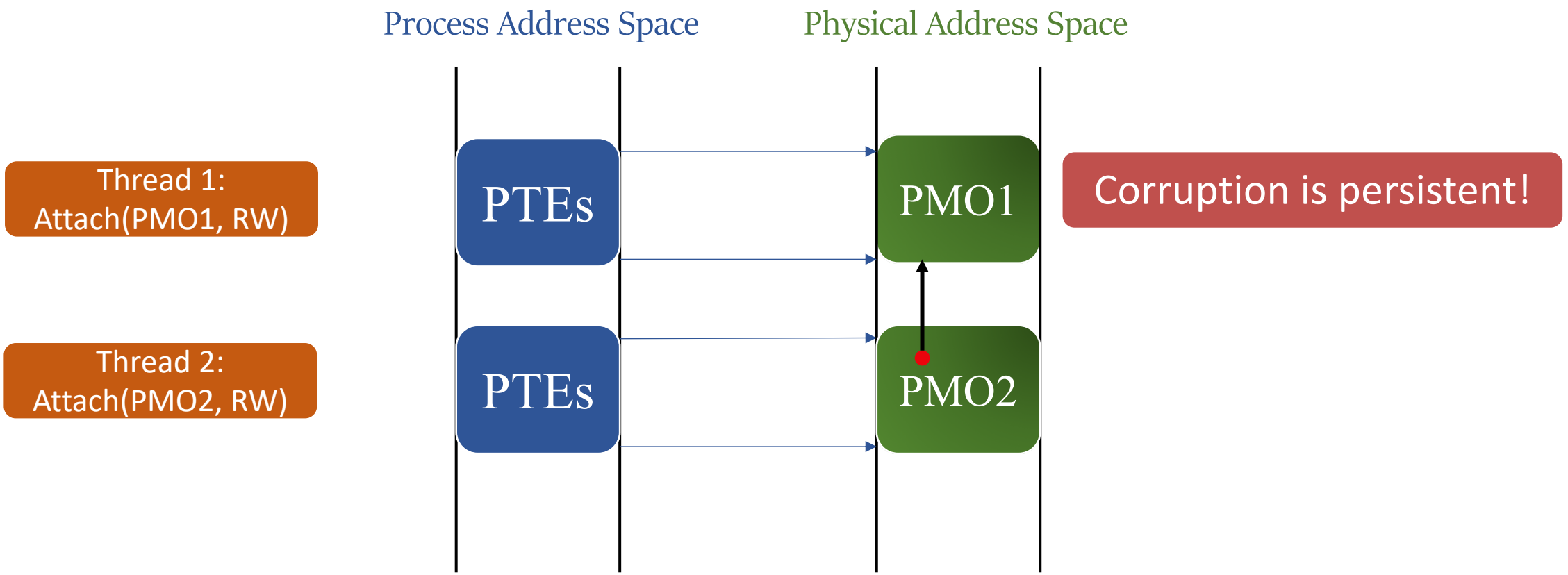
# Persistent Memory (PM)



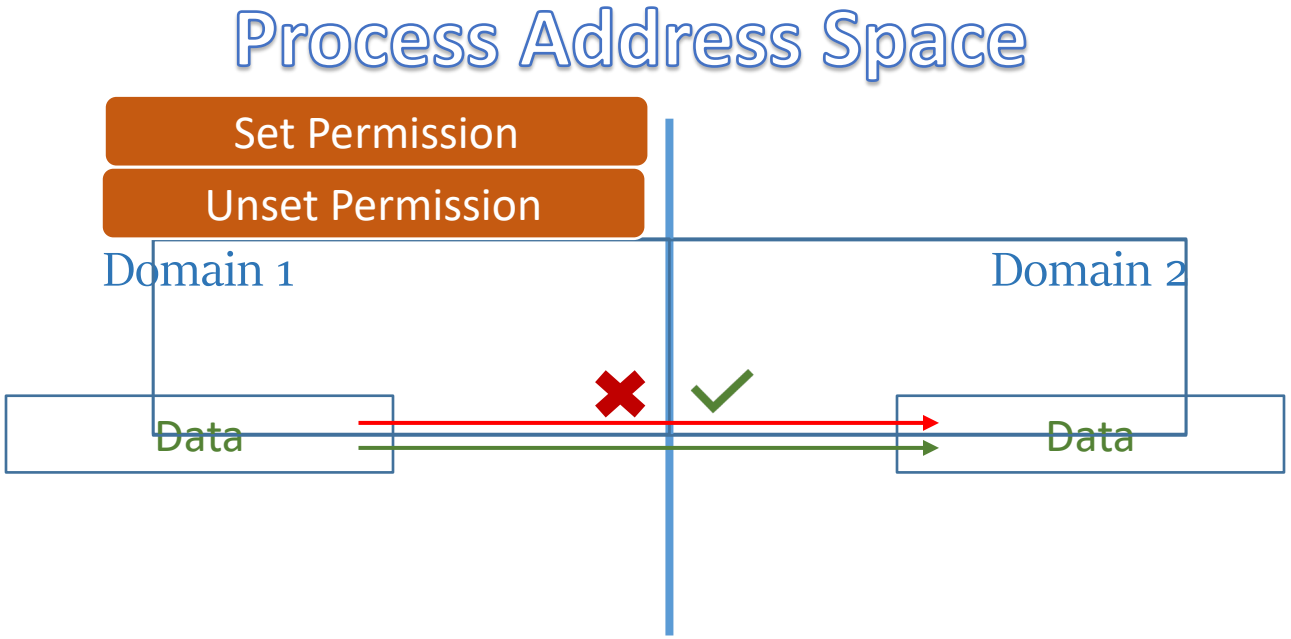
# Attach & Detach Programing Model



# Security is more Important for PM



# Intra-process Isolation



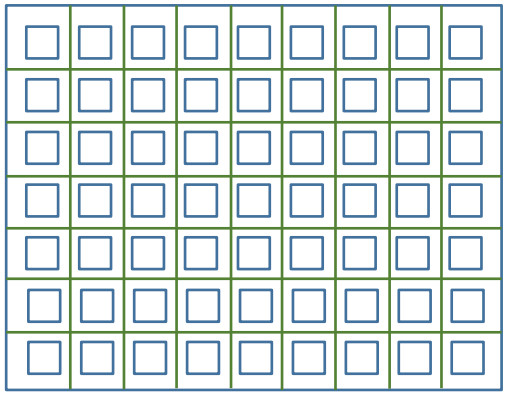

Intentional or accidental access

# Intra-process Isolation for PM

Domain

(unique ID =PMO ID)

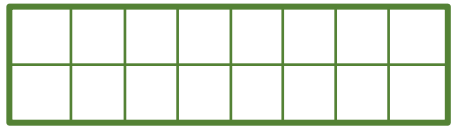

Persistent Memory



Domains→PKeys

Intel Memory Protection Keys

16 protection keys



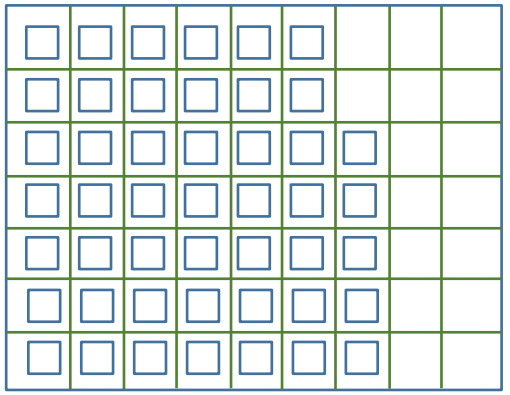
# Intra-process Isolation for PM

Domain

(unique ID =PMO ID)

PMO

Persistent Memory



Application (process)

Client Client Client

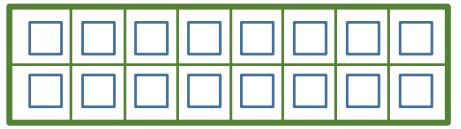
Client Client Client

…

Intel Memory Protection Keys

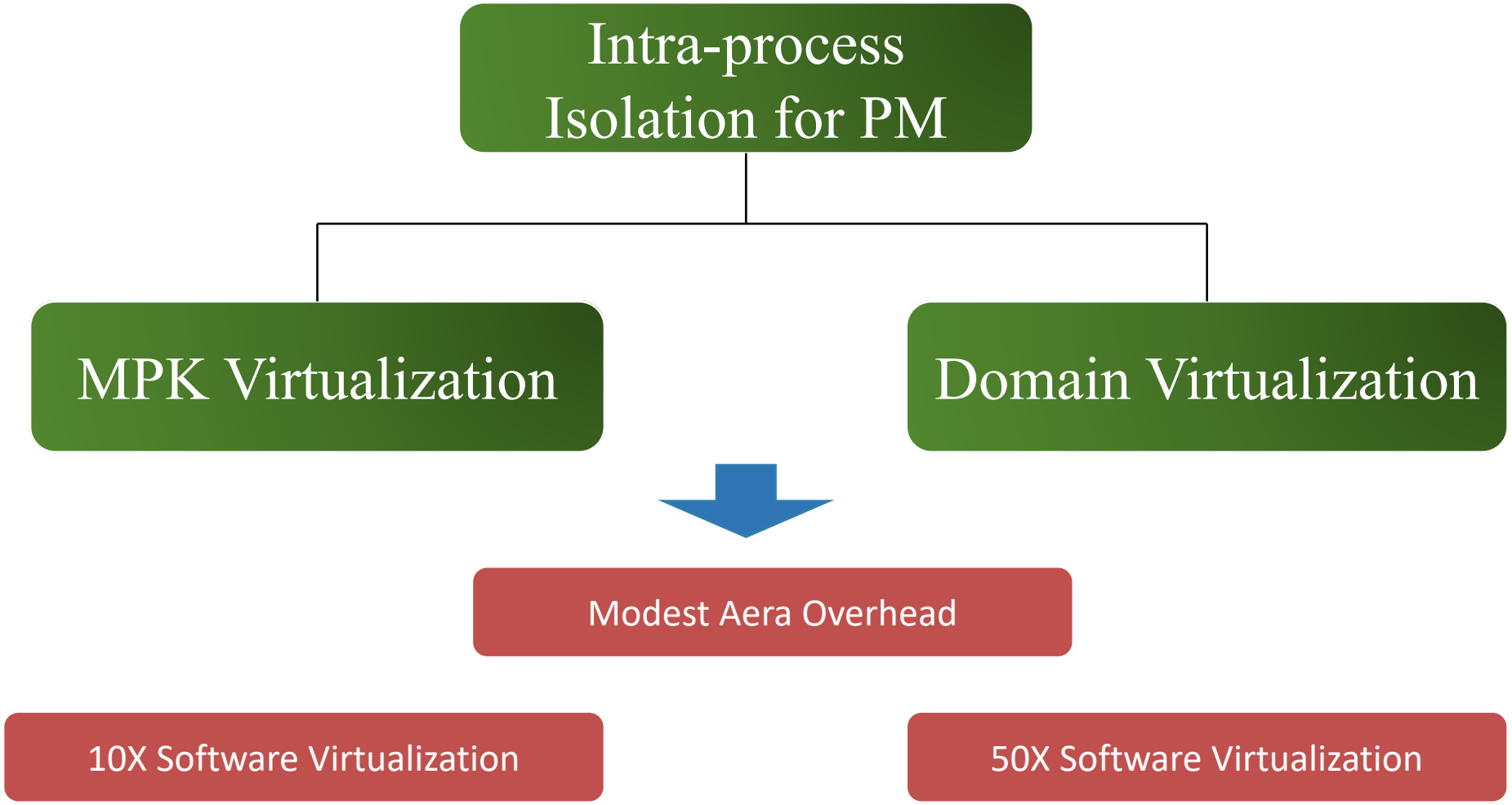
16 protection keys

Domains→Keys



Hundreds of active domains

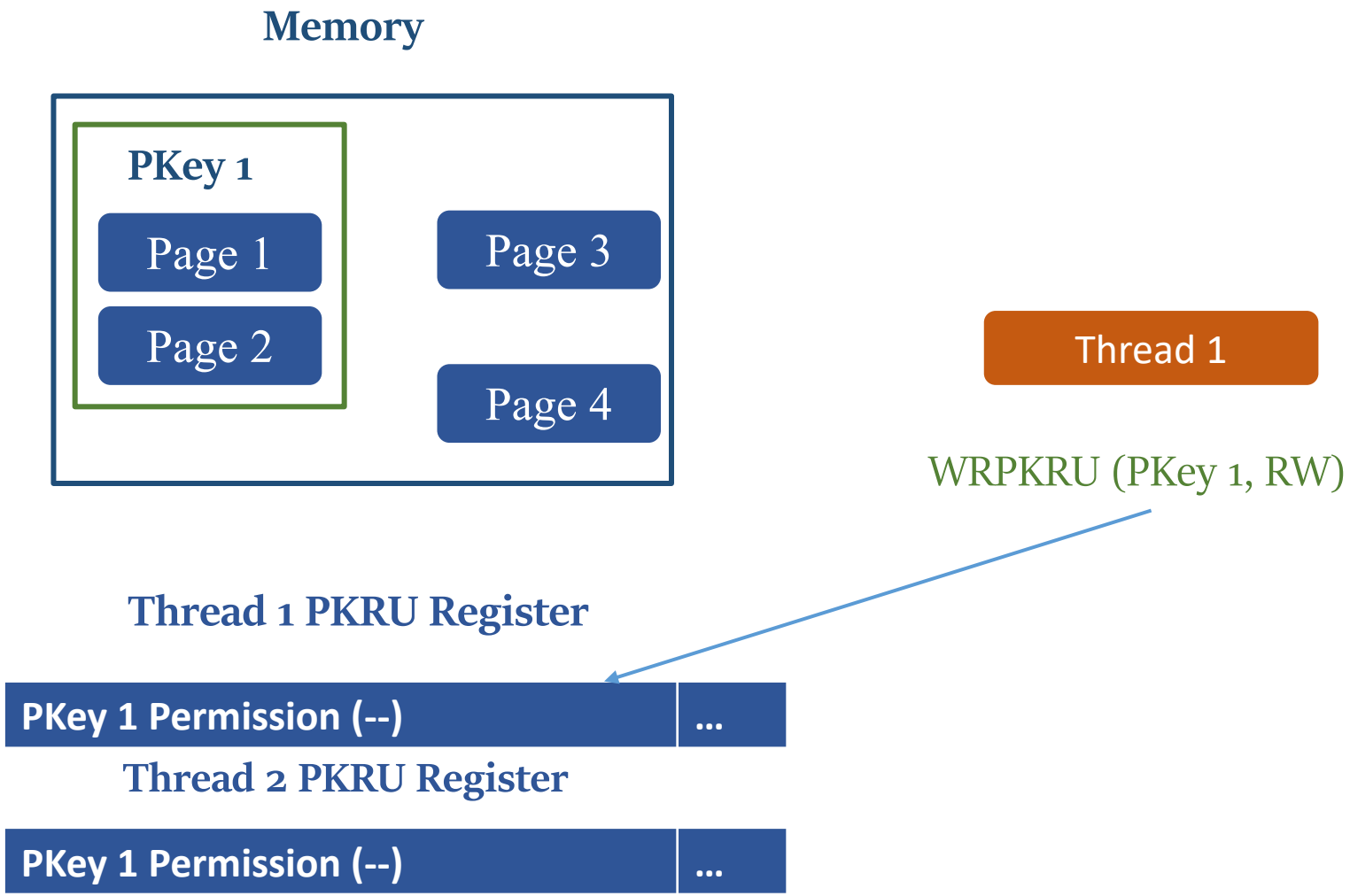
# Contribution



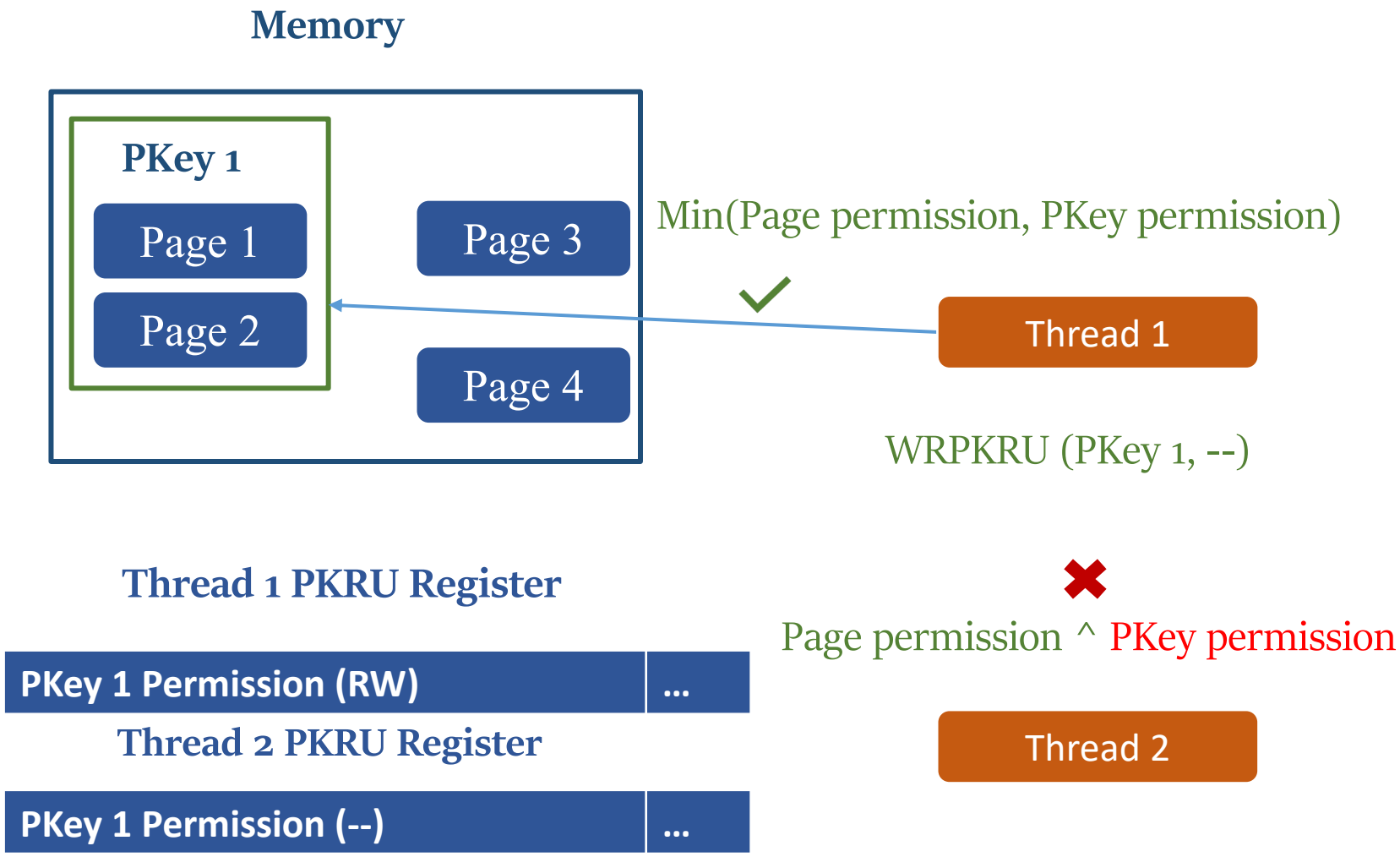
# Outline

- Intel Memory Protection Keys (MPK)
- Virtualization Analysis
- MPK Virtualization
- Domain Virtualization
- Evaluation

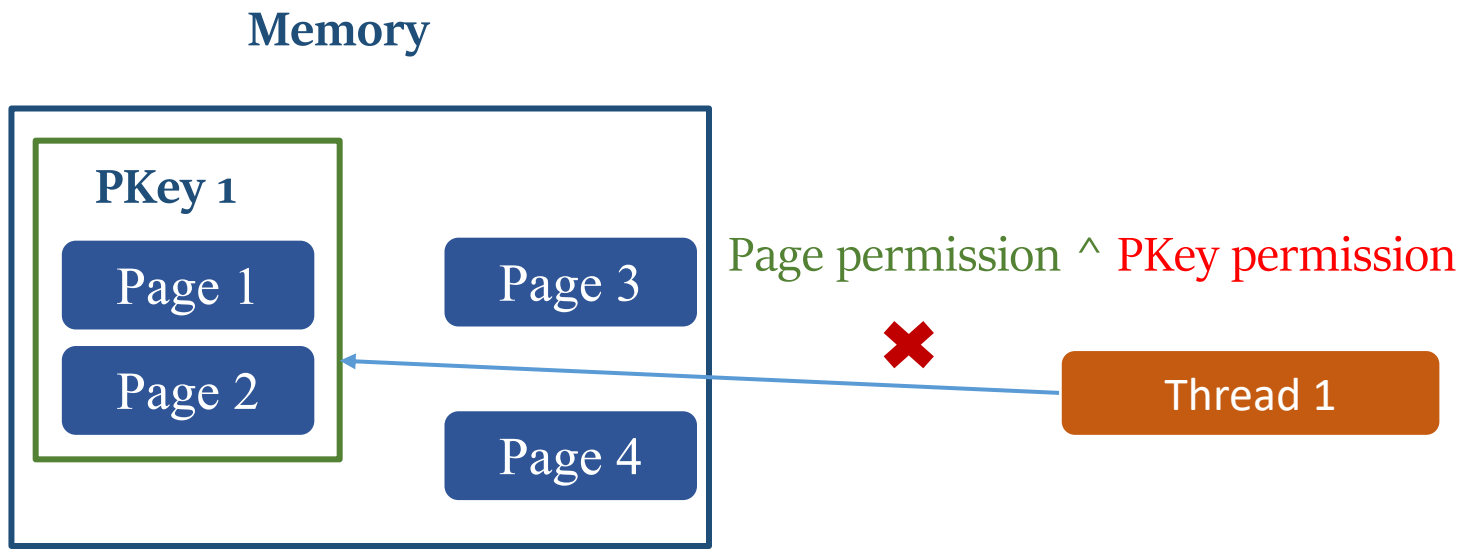
# Intel Memory Protection Keys (MPK)



# Intel Memory Protection Keys (MPK)



# Intel Memory Protection Keys (MPK)



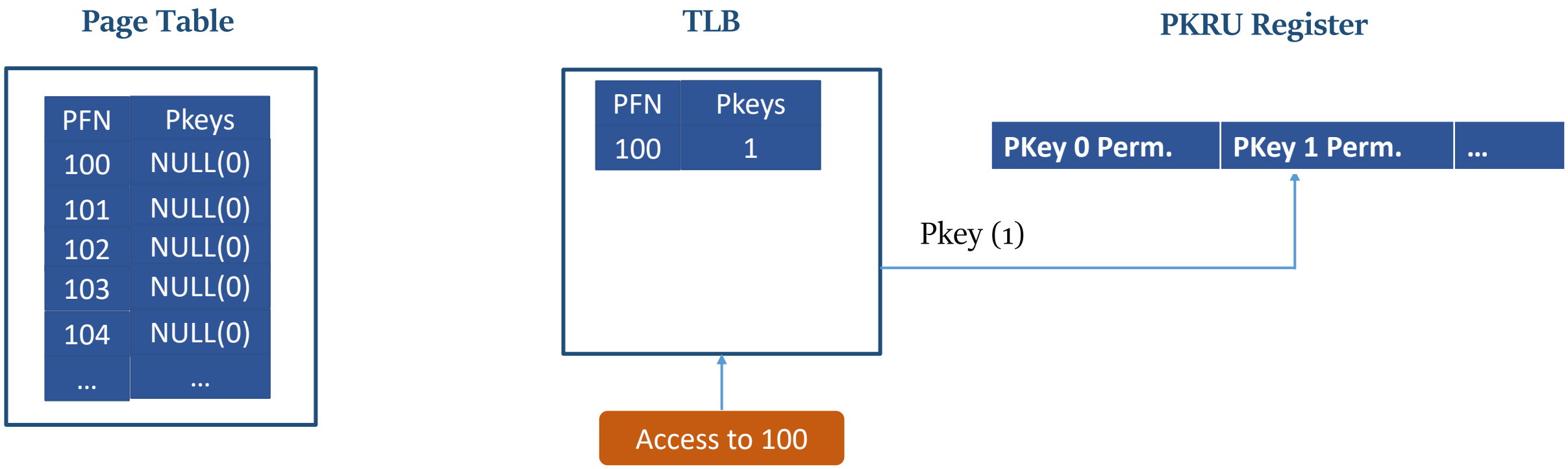

**Thread 1 PKRU Register**

| PKey 1 Permission (--) | ... |
|---|---|

**Thread 2 PKRU Register**

| PKey 1 Permission (--) | ... |
|---|---|

# Intel MPK Workflow



# Inefficiency of Software Virtualization

Clean Pkeys | Set Pkeys | TLB invalidations | Store and restore permissions

**Page Table**

**PMO 1**

| PFN | Pkeys |
|---|---|
| 100 | NULL(0) |
| 101 | NULL(0) |
| 102 | NULL(0) |
| 103 | NULL(0) |
| 104 | NULL(0) |
| … | … |

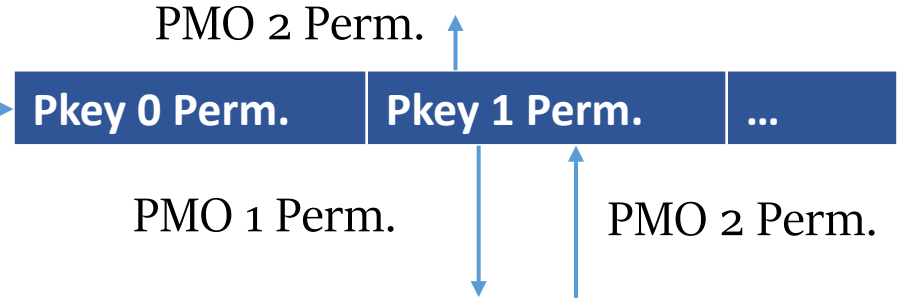
**PMO 2**

| PFN | Pkeys |
|---|---|
| 200 | NULL(0) |
| 201 | NULL(0) |
| 202 | NULL(0) |
| 203 | NULL(0) |
| 204 | NULL(0) |
| … | … |

**TLB**

| PFN | Pkeys |
|---|---|
| 100 | 1 |
| 103 | 1 |

**PKRU Register**




All steps are needed when access evicted domain/PMO!

~4000 cycles, 1 eviction per 1000 instructions ~ 400% overhead

# Two Hardware Virtualization Design

Clean Pkeys

Set Pkeys

TLB invalidations

Store and restore permissions

MPK Virtualization

# Two Hardware Virtualization Design

Clean Pkeys

Set Pkeys

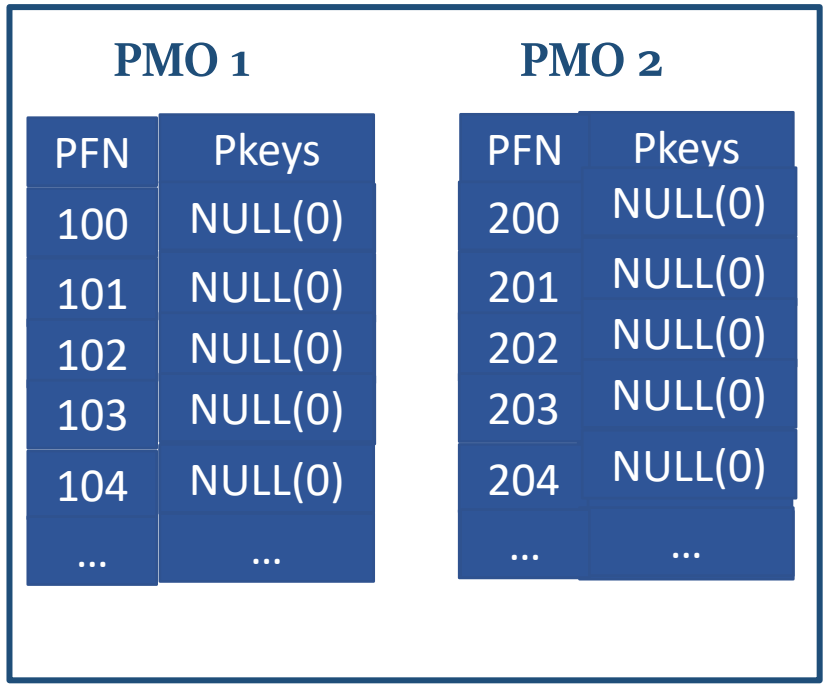
TLB invalidations

Store and restore permissions

Domain Virtualization

# MPK Virtualization

**Page Table**

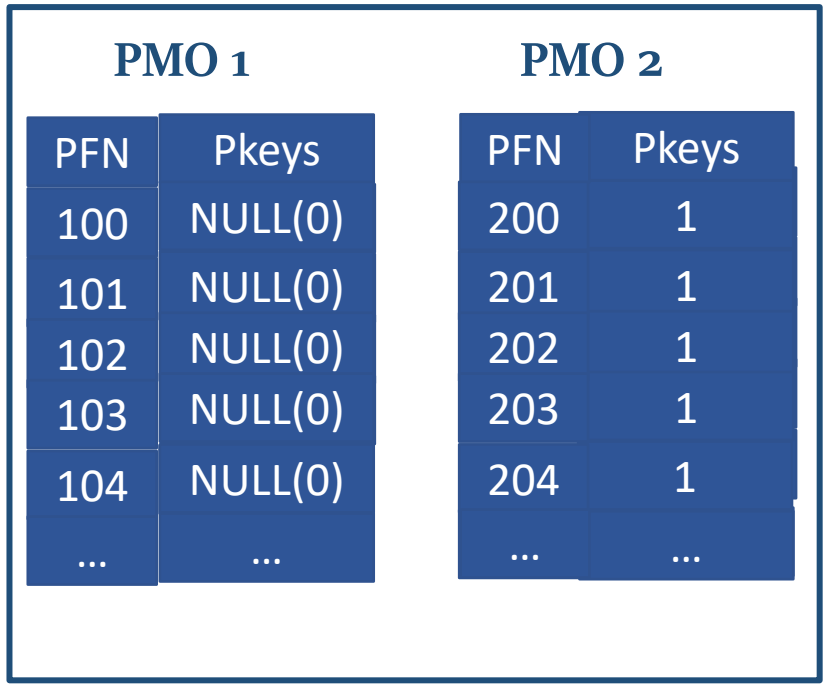



Consecutive Virtual Address

PTE with PKey ⟷ **TLB** —PKey→ **PKRU Register**

# MPK Virtualization

**Page Table**




Consecutive Virtual Address

PTE with PKey ⟷ **TLB** —PKey→ **PKRU Register**

# MPK Virtualization

**Page Table**

PTE with PKey ⟷ **TLB** PKey → **PKRU Register**

**PMO 1**

| PFN | Pkeys |
|---|---|
| 100 | NULL(0) |
| 101 | NULL(0) |
| 102 | NULL(0) |
| 103 | NULL(0) |
| 104 | NULL(0) |
| … | … |

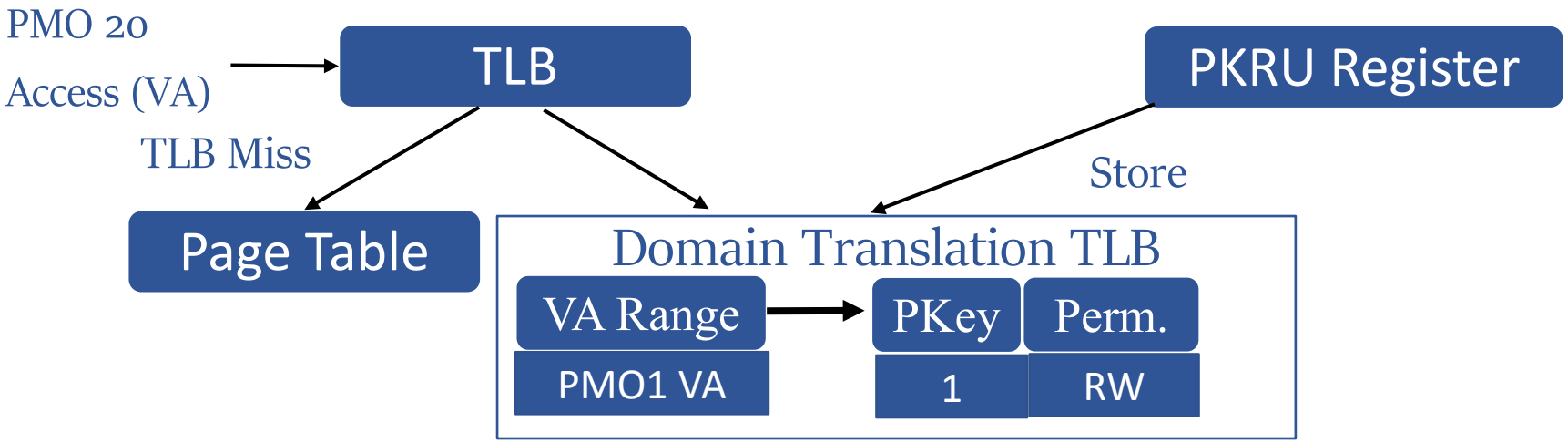
**PMO 2**

| PFN | Pkeys |
|---|---|
| 200 | 1 |
| 201 | 1 |
| 202 | 1 |
| 203 | 1 |
| 204 | 1 |
| … | … |

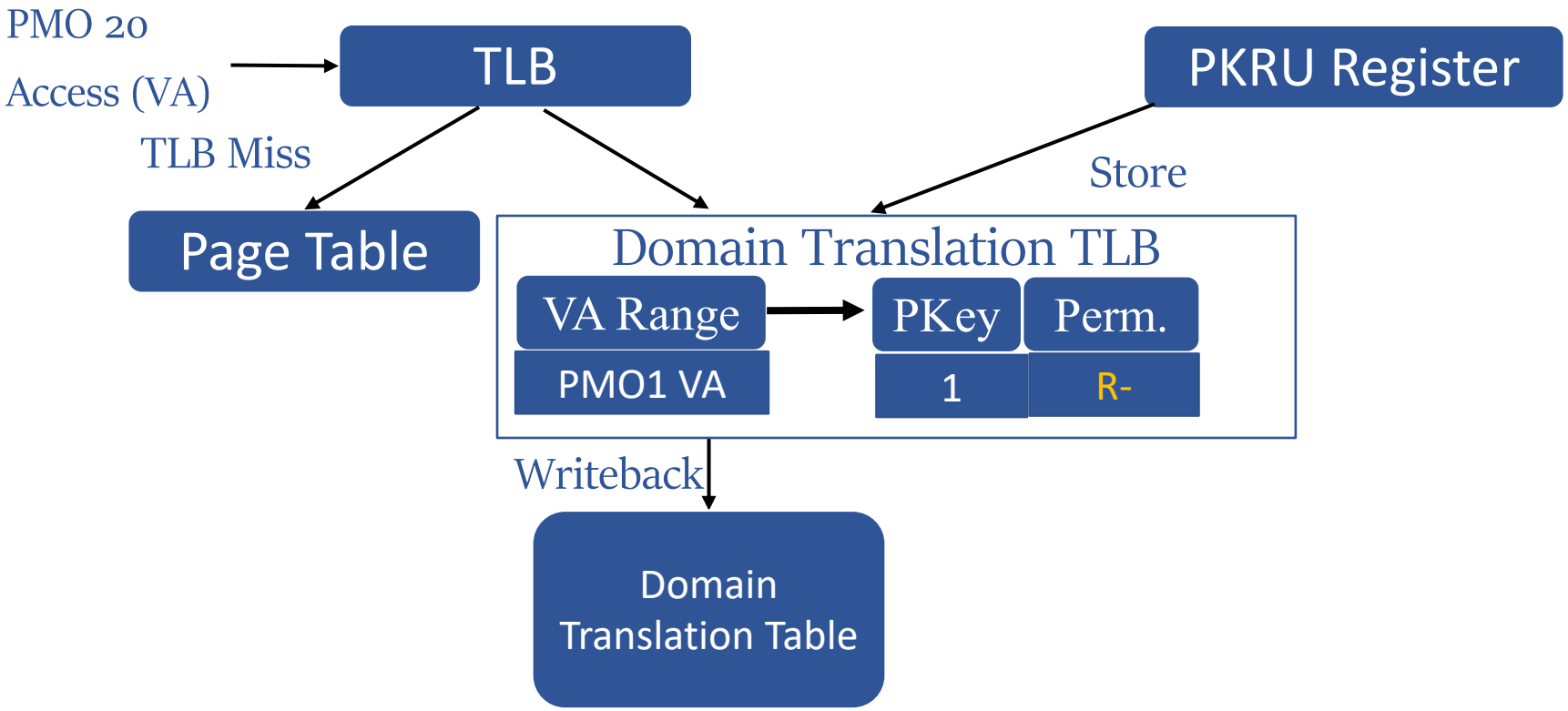
Virtual Address Range | Pkey

Consecutive Virtual Address

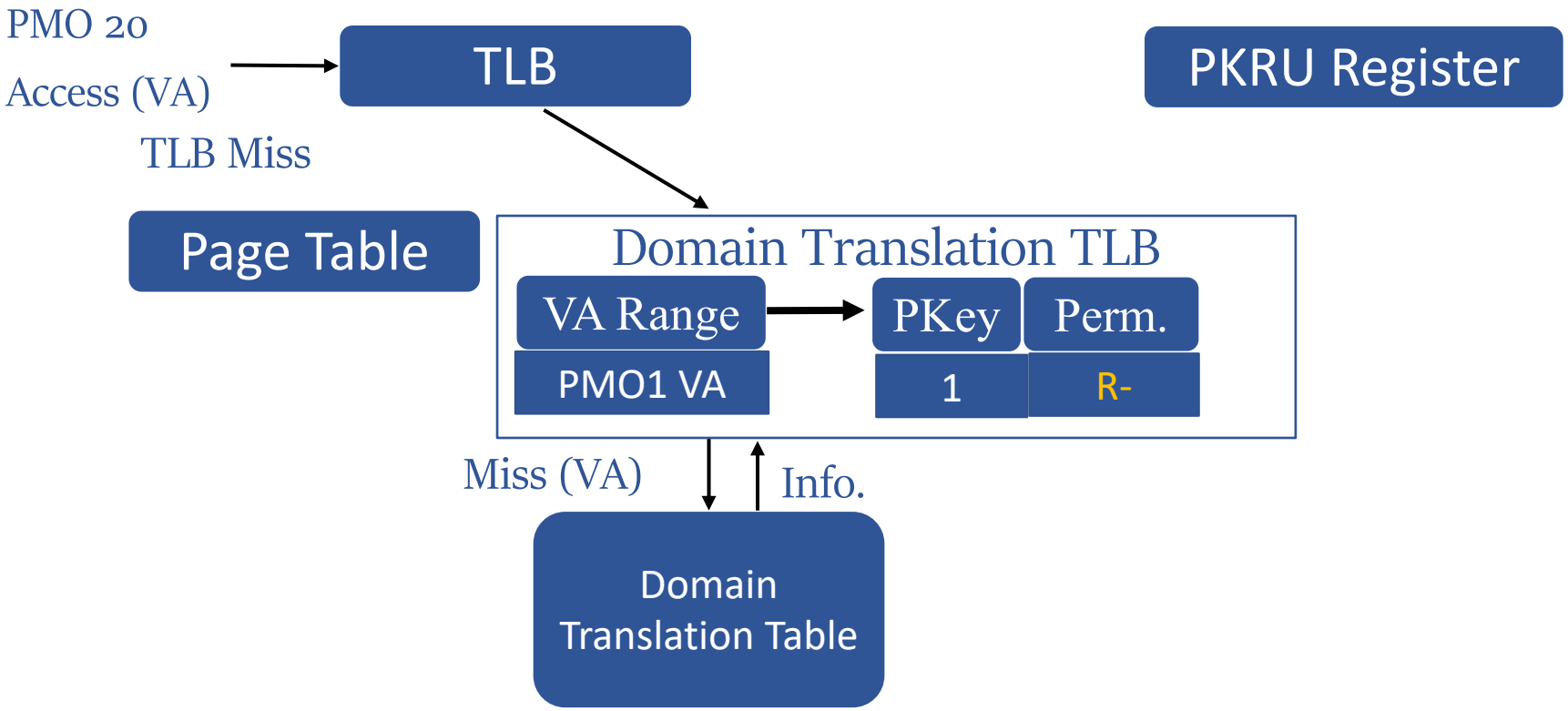
# MPK Virtualization



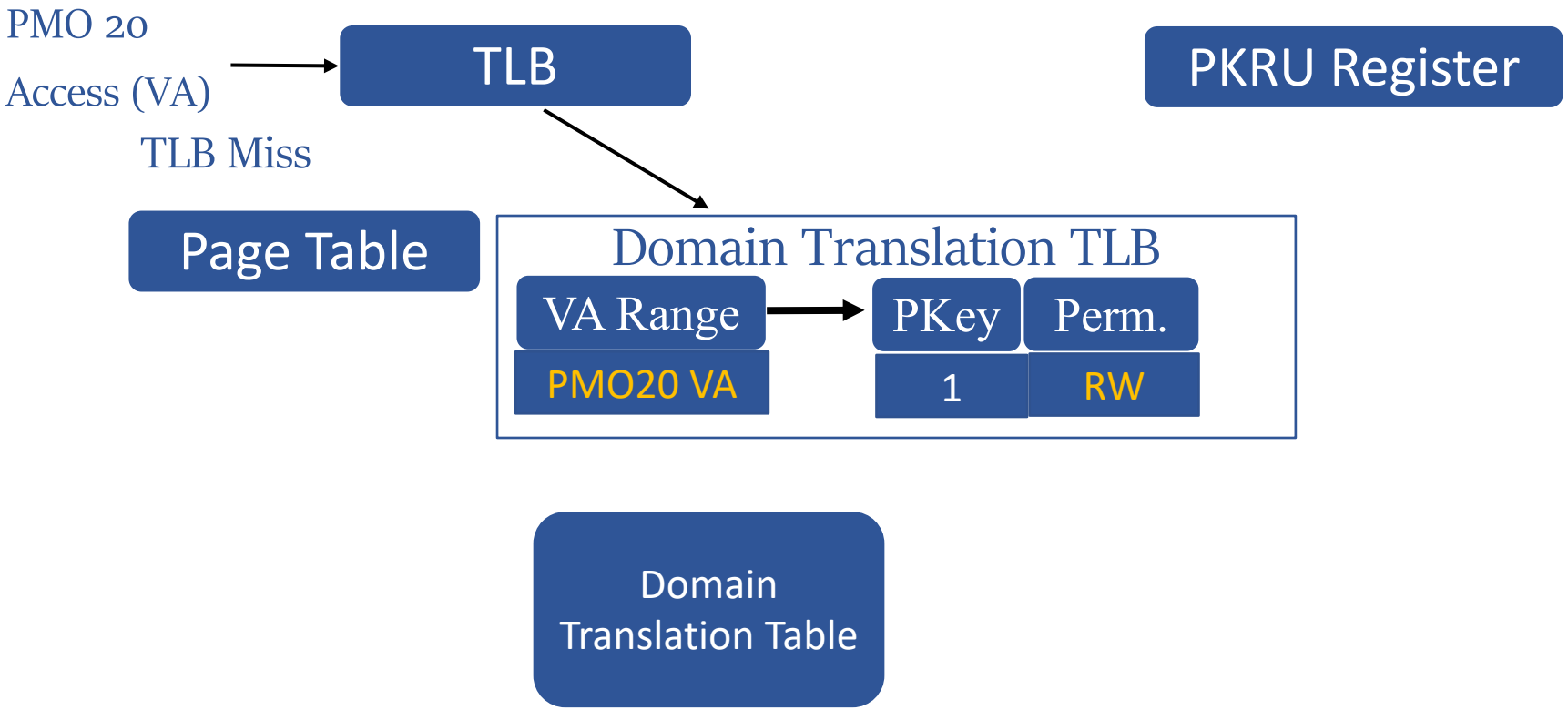
# MPK Virtualization



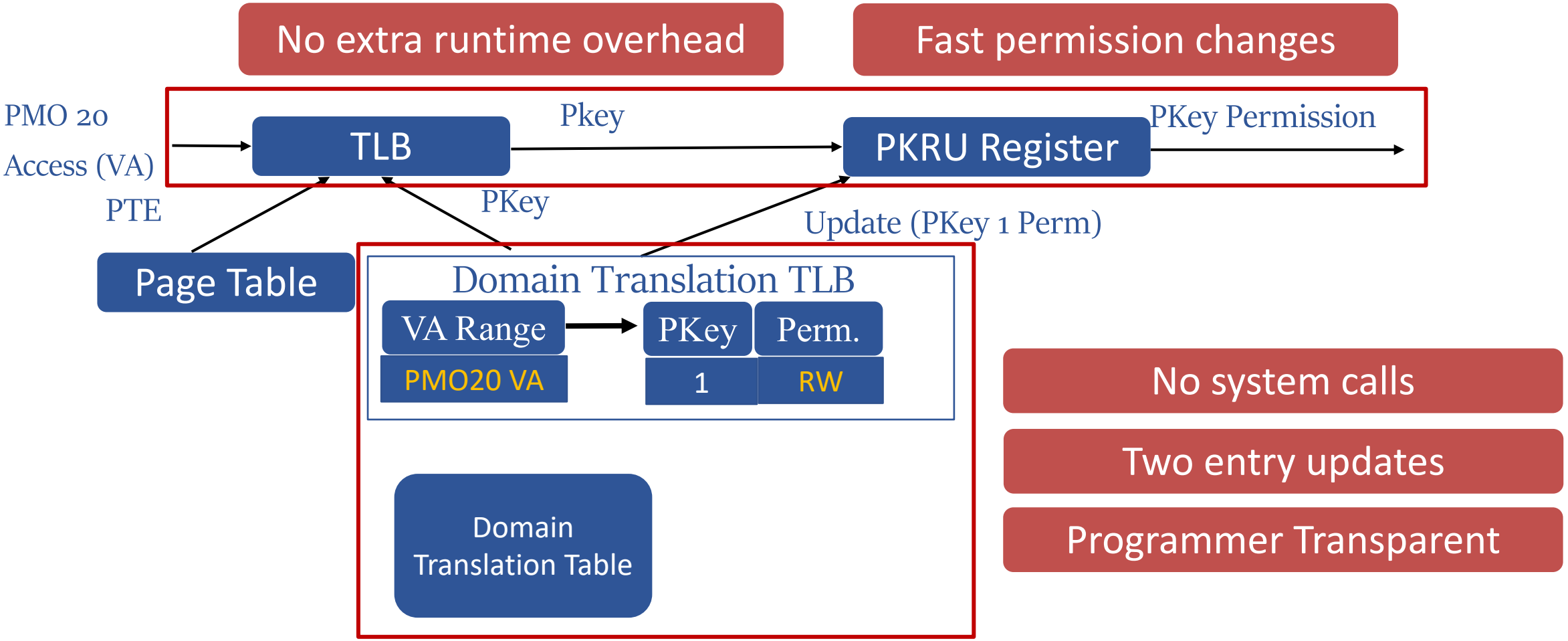
# MPK Virtualization



# MPK Virtualization



# MPK Virtualization



# Second Hardware Virtualization Design

Clean Pkeys

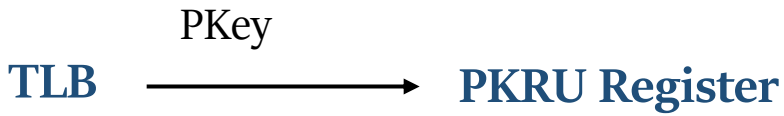
Set Pkeys

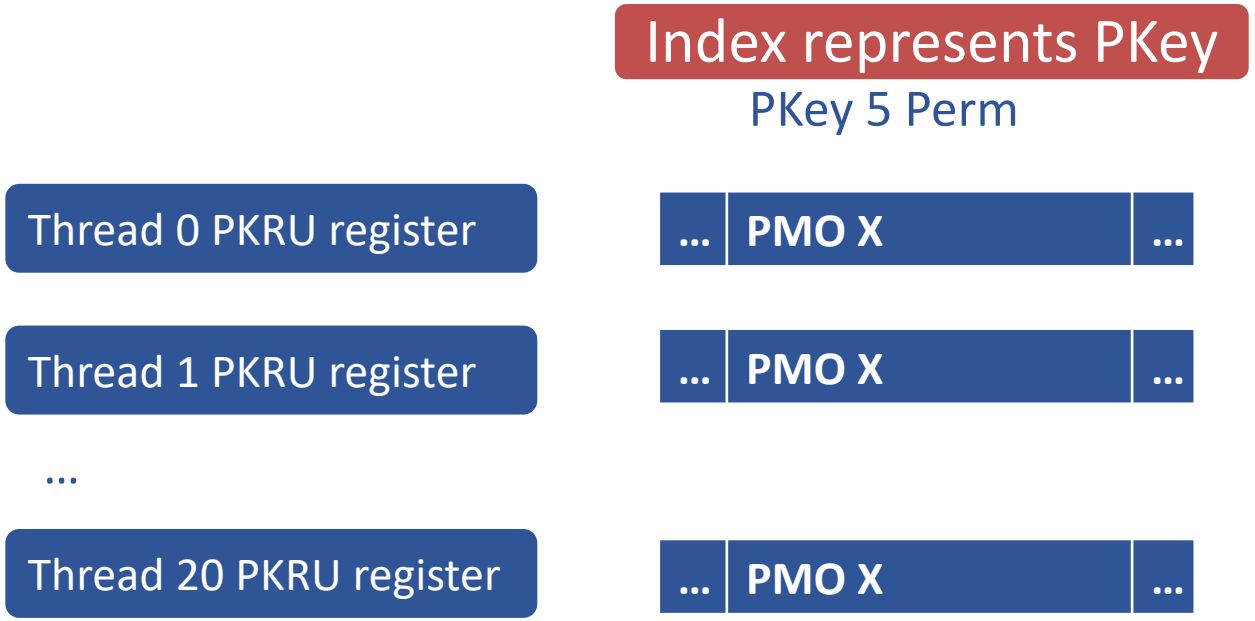
TLB invalidations

Store and restore permissions

Domain Virtualization

# Inefficiency from TLB invalidations

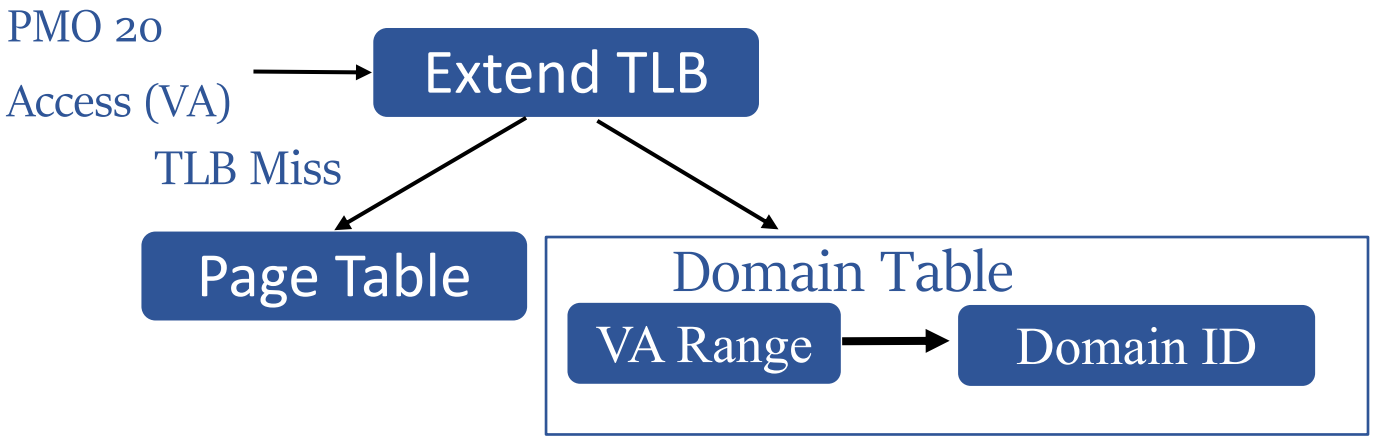


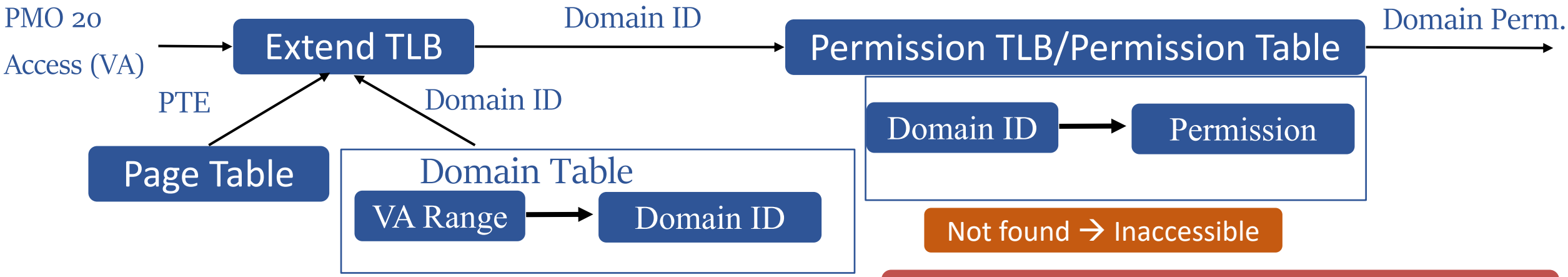
# Inefficiency from TLB invalidations

# Inefficiency from TLB invalidations

# Domain Virtualization



# Domain Virtualization




Not found → Inaccessible

Thread-private permission control

No TLB invalidation

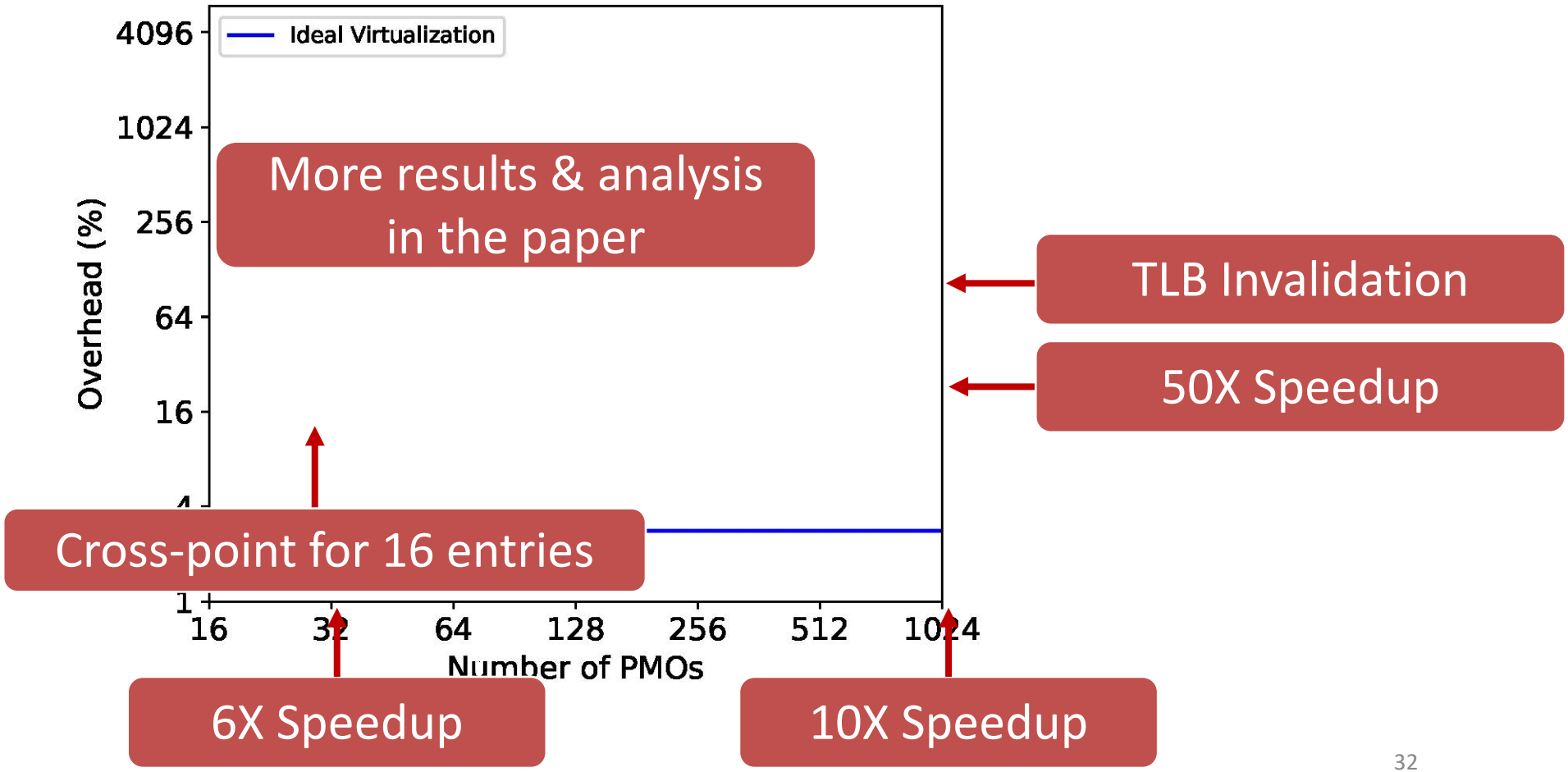
Fast permission grant/change/removal

# Evaluation Methodology

- Workloads:
  - WHISPER benchmarks for 1 PMO
  - Microbenchmarks with multiple PMOs
- Access pattern of multiple PMOs
  - Randomly choose a PMO ID to access
- Architectural Overhead:
  - Sniper Simulator (details in the paper)

# Performance

- Microbenchmarks for multiple PMOs



# Conclusion

- **Proposed protecting PMO by using intra-process isolation**
- Uncovered scalability limitations of software MPK virtualization for PMOs
- Designed the **hardware MPK virtualization** that **builds on top of MPK,** achieving **10X** speedup for 1024 domains
- Designed the **domain virtualization** that achieves **50X** speedup for 1024 domains