Memory Protection

OS GUEST LECTURE

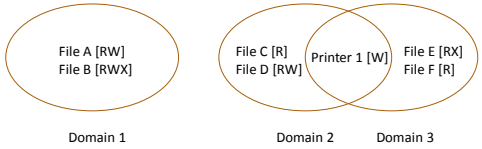
XIAOWAN DONG

11/15/2018

Operating Systems Protection

Goal: Ensure data confidentiality + data integrity + systems availability

Protection domain = the set of accessible objects + access rights



Private Virtual Address Space

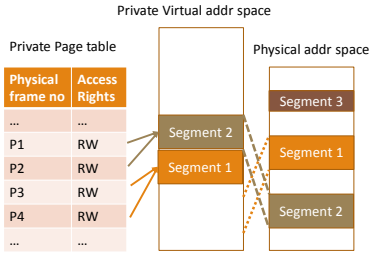
Most common memory protection mechanism in current OS

Each process has a private virtual address space

- *Set of accessible objects*: virtual pages mapped
- *Access rights*: access permissions to each virtual page

Recorded in per-process page table

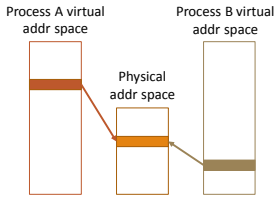
- Virtual-to-physical translation + access permissions



Challenges of Sharing Memory

Difficult to share pointer-based data structures

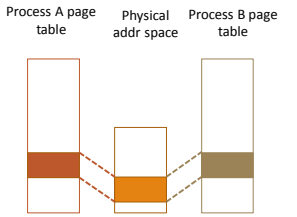
- *Data may map to different virtual addresses in different address spaces*



Challenges of Sharing Memory

Potential duplicate virtual-to-physical translation information for shared memory

- Page table is per-process at page granularity
- *Single copy of the physical memory, multiple copies of the mapping info (even if identical)*

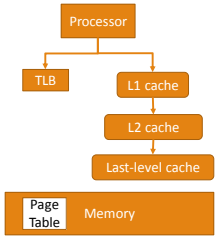


5

Challenges for Memory Sharing

Potential duplicate virtual-to-physical translation information for shared memory

- *Single copy of the physical memory, multiple copies of the mapping info (even if identical)*
- *Duplication in Translation Lookaside Buffer (TLB) and memory hierarchy (caches, main memory)*



6

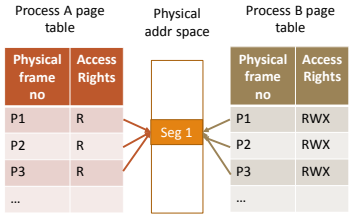
Challenges for Changing Access Permissions of Memory Regions

Changing access permissions of an entire memory region is expensive

- E.g., disabling writes to a memory region across processes

Page table is at page granularity on per-process basis

Required to traverse each page table entry of each virtual address space



7

Are there any other memory protection mechanisms besides private virtual address space?

8

Outline

Single address space
- Domain-page model
- Page-group model

State-of-art memory protection mechanisms
- ARM protection domains
- Intel Memory Protection Keys

Outline

Single address space
- Domain-page model
- Page-group model

State-of-art memory protection mechanisms
- ARM protection domains
- Intel Memory Protection Keys

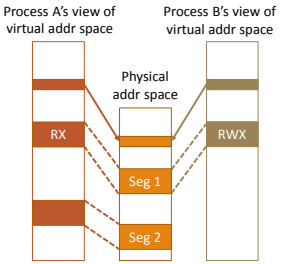
Single Address Space

One single virtual address space shared across processes

One virtual address is mapped to a *unique* physical address

Simplifies memory sharing compared to private virtual address space

Proposed in the 90s
- Emerging 64-bit address space

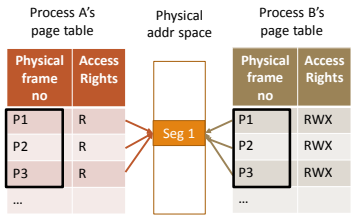


Single Address Space

Can we use page table like in private virtual address space
- Recorded at page granularity per process
- Virtual-to-physical translation information
- Access permissions

Duplicate translation information
- Translation of a virtual page is unique across processes

Other protection domain models
- Domain-page model
- Page-group model



Outline

Single address space
- **Domain-page model**
- Page-group model

State-of-art memory protection mechanisms
- ARM protection domains
- Intel Memory Protection Keys

Single Address Space: Domain-Page Model

Protection domain = set of accessible pages + access permissions

Capability list

Each (domain, page) pair is unique
- Access rights associated with (domain, page)

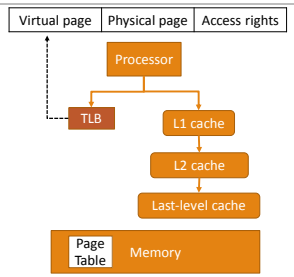
	Page 1	Page 2	Page 3	Page 4	Page 5	Page 6
Domain A	R		RW		RWX	
Domain B		R-X		R	R	R
Domain C	RWX	R-X	RW			

Protection Lookaside Buffer

One implementation of domain-page model

Translation lookaside buffer (TLB)
- On-chip cache of page table
- Virtual-to-physical translation information + access permissions

Protection Lookaside buffer (PLB)
- Only records access permissions
- Translation information is saved separately



Protection Lookaside Buffer (PLB)

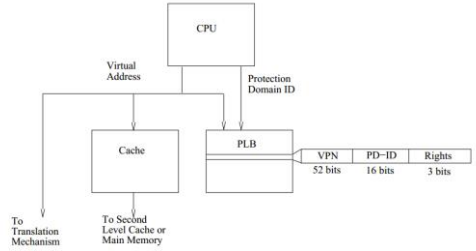


Image Source: ASPLOS , "Architecture support for single address space operating systems",1992.

PLB Advantages

No duplicate translation information
- Each page has a single translation entry in TLB and memory hierarchy

Changing access permission of a memory region is cheaper
- Use a single PLB entry for the entire memory region (stack, code segment and etc.)
- Only required to modify one PLB entry

17

Outline

Single address space
- Domain-page model
- **Page-group model**

State-of-art memory protection mechanisms
- ARM protection domains
- Intel Memory Protection Keys

18

Single Address Space: Page-Group Model

Page group is a set of pages
- Each page belongs to a single page group

Access permissions associated with each page
- *As opposed to (domain, page) pair in the domain-page group*

Protection domain = set of accessible page groups + access permissions

	Group A	Group B	Group C
Process I	√	√	
Process II		√	√
Process III	√		

19

PA-RISC

An architecture of page-group model designed by HP

Each process has 4 page-group registers (PID) for accessible page group IDs
- One additional write-disable bit that disables writes to the entire page group

Each process runs in one of the 4 privilege levels
- 0 (the highest) to 3

	Privilege level	PID1	PID2	PID3	PID4
Process I	0	3	11	6	9
Process II	3	11	12	13	14
Process III	2	3	6	9	5

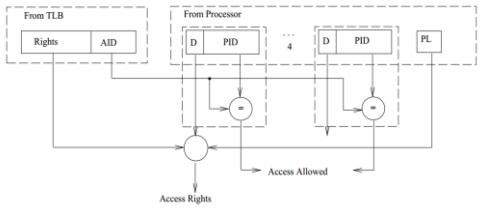
20

PA-RISC

Each page has unique translation information and access permissions
- Recorded in page table/TLB
- Access permission = read, write, execute and the corresponding privilege levels
  - E.g., writable from privilege level 0, readable from level 0, 1 and 2, and inaccessible from level 3
- *No duplicate translation information*

21

PA-RISC



Whether accessible
- Determined by PID registers

Access permission = Rights in TLB based on privilege level + write-disable bit

Image source: ASPLOS , "Architecture support for single address space operating systems",1992.

22

Outline

Single address space
- Domain-page model
- Page-group model

State-of-art memory protection mechanisms
- ARM protection domains
- Intel Memory Protection Keys

23

State-of-Art Memory Protection

Modern architectures use private virtual address spaces
- As supported in modern OSs such as Linux and FreeBSD

However, they also provide other memory protection models
- Like ARM and Intel

24

Outline

Single address space
- Domain-page model
- Page-group model

State-of-art memory protection mechanisms
- **ARM protection domains**
- Intel Memory Protection Keys

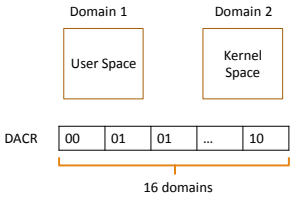
ARM Protection Domain

Domain = set of accessible pages

32-bit ARM supports 16 domains

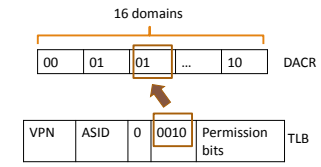
Domain access control register (DACR)
- Defines the access permissions of current process to 16 domains (2 bits per domain)
- Saved in process control block when the current process is context switched off

Each page belongs to a domain
- Identified by the *domain* field in page table entry

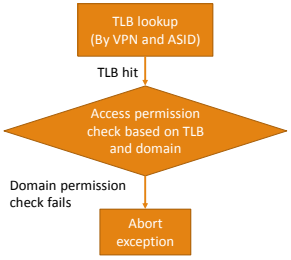


00: No access permission
01: Based on permission bits listed in page table
10: RWX permissions

ARM Protection Model



00: No access permission
01: Based on permission bits listed in page table
10: RWX permissions



ARM Protection Domain

A legacy feature that is not in use in reality
- Only domain 1 (user space) and 2 (kernel space) are in use
- Removed from 64-bit ARM architecture

Any other use cases
- Alleviate duplicate translation information on Android

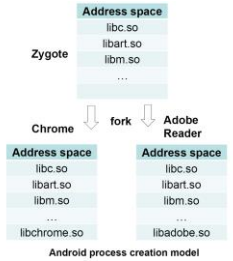
ARM Protection Domain: A Use Case on Android

Duplicate Translation info on Android

Android uses Linux kernel and thus uses private virtual address space

58% duplicate page table pages for shared libraries on Android

All android applications share the same virtual and physical addresses for the preloaded shared libraries
- Due to Android process creation model

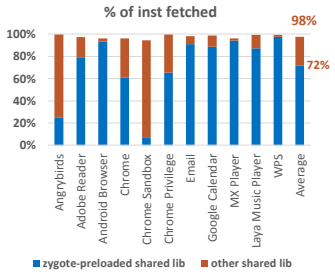


Android process creation model

Android Application Instruction Footprint

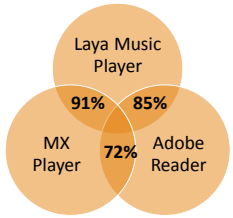
Most of the instructions accessed are from shared libraries preloaded

Number of shared libraries per application:
- Loaded: *88 to 107* (zygote-preloaded: 88)
- Invoked: *24 to 68* (zygote-preloaded: 21 to 46)



Android Application Instruction Footprint

Considerable overlap in the shared library code accessed across different applications
- 46% of total inst pages accessed are in common for each pair of applications
- Zygote-preloaded: 38%



The % of inst footprint overlapped

Sharing TLB for Shared Libraries

To alleviate duplication, we share page table and TLB for preloaded shared libraries across all Android processes [Eurosys'16, IISWC'15]

- *In this talk we only focus on sharing TLB*

We use Global bit and ARM protection domain

Global bit

- Traditionally used for kernel-space translation
- Kernel space mappings are identical and therefore shared across processes
- Overrides Address Space Identifier (ASID) in TLB

33

Sharing TLB for Shared Libraries

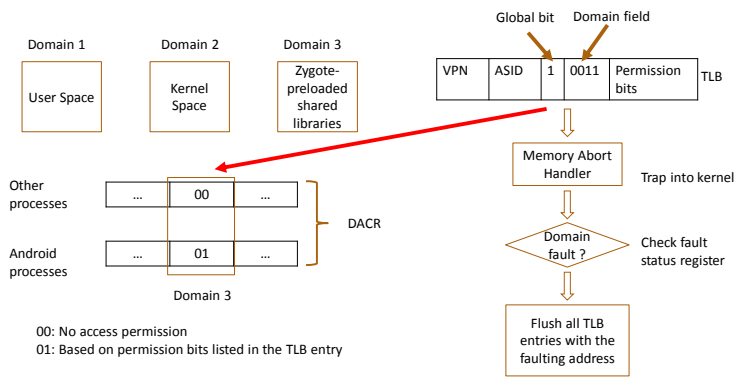
Global bit

- Set the global bit in the page table entries of the preloaded shared libraries
- *To share TLB entries*

ARM protection domain

- There are other process (system services and daemons) not forked from the template
- Prevents them from accessing the shared global TLB entries
- *To unshare TLB entries*

34



35

Outline

Single address space

- Domain-page model
- Page-group model

State-of-art memory protection mechanisms

- ARM protection domains
- **Intel Memory Protection Keys**

36

Intel Memory Protection Keys

Similar to 32-bit ARM protection domain model
- While 64-bit ARM removes it, Intel brings it back

Goal: Applications can efficiently modify access permissions at memory region granularity

Only applied to user-space pages

37

Intel Memory Protection Keys

Intel supports 16 domains

Protection key rights for user pages (PKU) register:
- Specifies the access permissions of current process to 16 domains
- 2 bits per domain (access disable bit + write disable bit)
- Configurable in user space

Each page is associated with a protection key
- Recorded in page table entry

Domain = set of accessible pages with the same protection key

Access permission check: page table entry permissions + protection key permissions (access disable? Write disable?)

38

Conclusions

Per-process private virtual address space interferences with memory sharing

Duplication of address translation information resulted from private virtual address space

Changing access permissions at memory region granularity is expensive with page table

Other memory protection models can be leveraged:
- Single address space
- State-of-art mechanisms (ARM protection domain and Intel MPK)

39