CSE 120: Principles of Operating Systems
Lecture 13: Protection

Prof. Joseph Pasquale
University of California, San Diego

March 4, 2019

Protection

- Processes access resources
- Resources are shared, need to be protected
  - from processes without permission
  - from improper access by a process
- What is the right protection model?
- What are the mechanisms?

The Kernel Enforces Protection

- To protect resources, have kernel “own” them
  - kernel can then allow access (temporarily)
- To access a resource, a process must ask for it
  - kernel can test whether access should be given
- Once a process is given access
  - kernel can prevent others for gaining access
  - kernel may/may not be able to take away access
- This assumes the kernel operates correctly

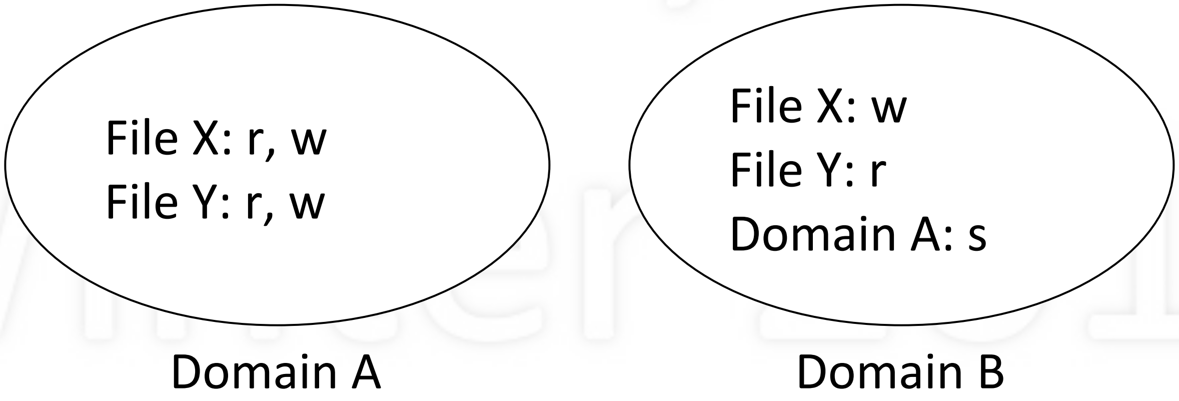
Protecting the Kernel

- The kernel itself must be protected!
- Mechanisms
  - Memory protection
  - Protected mode of operation: kernel vs. user
  - Clock interrupt, so kernel eventually gets control
- Notice, mechanisms are hardware supported
- Protected kernel can protect other resources

Goals Supported by Kernel

- Allow range of permissions
- Allow user to set/get them
- Be fast/simple for common case
- Support user expressing complex permissions

A Formal Model of Protection



- Protection: how to limit access to a resource
- Resource: object that requires protection
- Domain: set of (resource, permission) pairs
- Process: accesses resources within domain

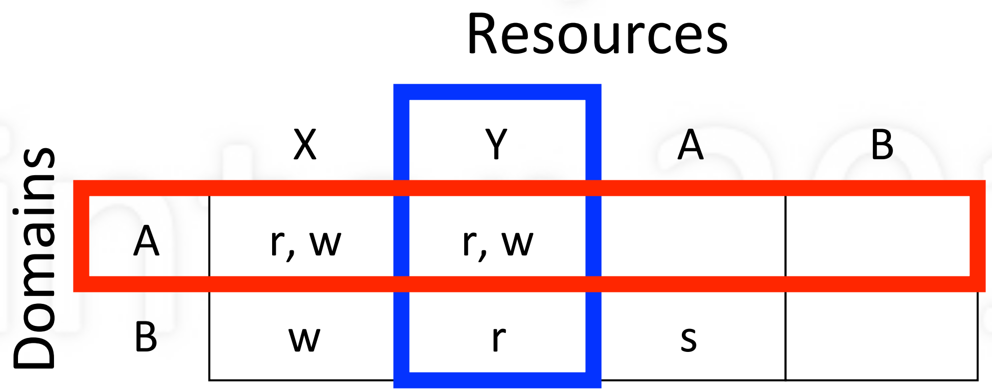
Protection Matrix

Resources

Domains	X	Y	A	B
A	r, w	r, w		
B	w	r	s	

- Can describe all domains as a matrix
  - – Rows are domains
  - – Columns are resources
  - – Matrix entry [d, r] contains permissions/rights

Efficient Representations



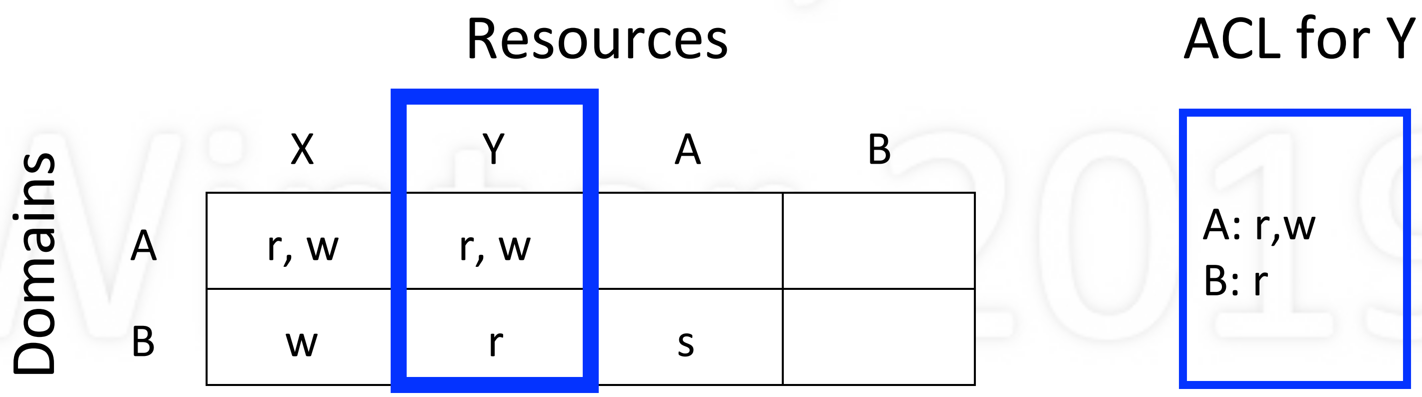
Resources

Domains

	X	Y	A	B
A	r, w	r, w		
B	w	r	s	

- Access Control Lists
  - – For each resource, list (domain, permissions) pairs
- Capability Lists
  - – For each domain, list (resource, permissions) pairs

Access Control Lists



Resources

Domains

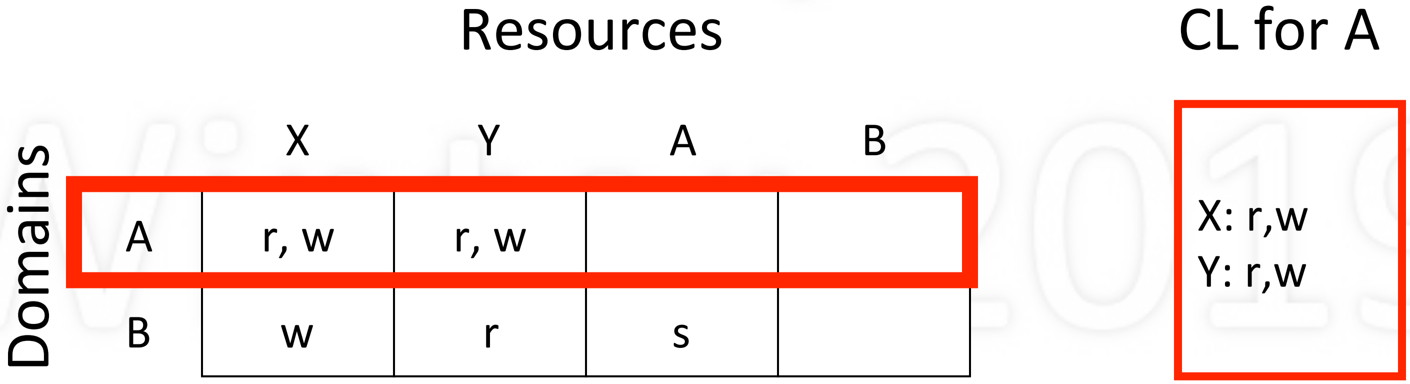
	X	Y	A	B
A	r, w	r, w		
B	w	r	s	

ACL for Y

A: r,w
B: r

- ACL is associated with resource
- Like a registry: if name is on list, ok to access
- Can be inefficient: must lookup on each access
- Revocation is easy; just remove from list

Capability Lists



Resources

Domains	X	Y	A	B
A	r, w	r, w		
B	w	r	s	

CL for A

X: r,w
Y: r,w

- Capability list associated with each domain
- Like key/ticket: if you have it, you get access
- Efficient: on access, just produce capability
- Hard to revoke

UNIX Protection

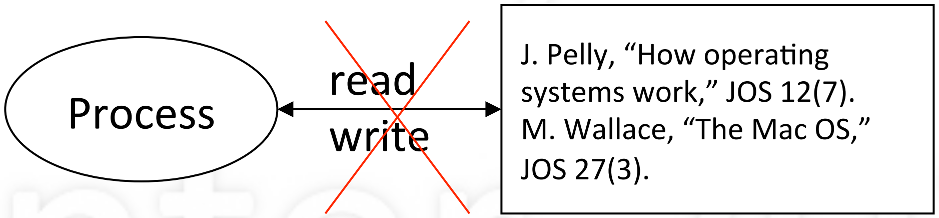
- Associated with each file is set of permissions
  - Permission bits r/w/x for owner, group, world
  - Limited form of access control list
- Protection domain: UID (user account ID) + …
  - A process is always in some domain
- When process opens file, check permission
- If ok, provide process with a capability
  - Future operations then carried out efficiently

Extending Protection in UNIX

- For common case, r/w/x for o/g/w adequate
- For special cases, can extend via user program
- SETUID mechanism: causes domain switch
- If executable file has SETUID bit set
  - Process runs in domain of owner (of executable)
  - Therefore, it runs with all the rights of the owner

CSE 120, UCSD Winter 2019

Example Use of SETUID bit



Process

~~read~~
~~write~~

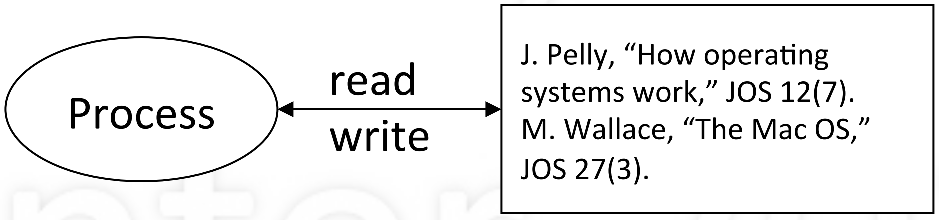
J. Pelly, "How operating systems work," JOS 12(7).
M. Wallace, "The Mac OS," JOS 27(3).

Exec of editor by Chris
Domain: Chris

Name: Bib
Owner: Pat
Perm: rw-------

- Pat has a file "Bib" of bibliographic references
- Chris wants to read *and add entries*
- But, Chris lacks permissions (only Pat can r/w)
- Pat wishes to allow append access, but how?

Example Use of SETUID bit



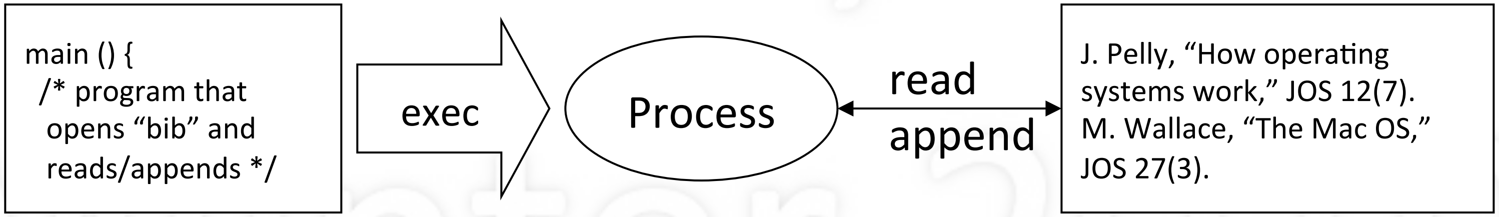
J. Pelly, “How operating systems work,” JOS 12(7).
M. Wallace, “The Mac OS,” JOS 27(3).

Exec of editor by Chris
Domain: Chris

Name: Bib
Owner: Pat
Perm: rw-rw-rw-

- Pat can set permissions so Chris can r/w
- But this gives Chris too much power
  - Chris can modify the file arbitrarily
  - Pat would like to give Chris only append access

Example Use of SETUID bit



```
main () {
 /* program that
  opens “bib” and
  reads/appends */
```

J. Pelly, “How operating systems work,” JOS 12(7).
M. Wallace, “The Mac OS,” JOS 27(3).

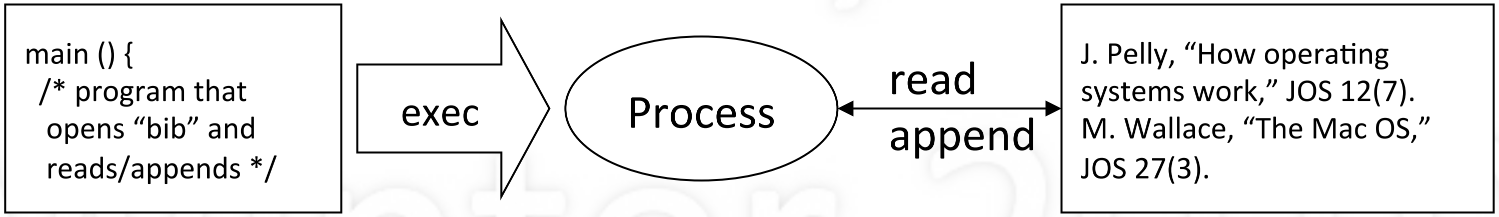
Name: EditBib
Owner: Pat
Perm: r-sr-xr-x

Exec of EditBib by Chris
Domain: Chris→Pat

Name: Bib
Owner: Pat
Perm: rw-------

- Pat provides program: only reads/appends
- Sets permissions
  - of program: execute (for Chris), and SETUID on
  - of Bib file: read/write only for Pat, not Chris

Example Use of SETUID bit



```
main () {
 /* program that
  opens “bib” and
  reads/appends */
```

J. Pelly, “How operating systems work,” JOS 12(7). M. Wallace, “The Mac OS,” JOS 27(3).

Name: EditBib
Owner: Pat
Perm: r-sr-xr-x

Exec of EditBib by Chris
Domain: Chris→Pat

Name: Bib
Owner: Pat
Perm: rw-------

- When Chris executes EditBib, runs as Pat
  - SETUID causes a domain switch to Pat’s domain
- Since Pat has r/w access, file can be modified
- Limited to what program does: read/append

Summary

- Protection
- Formal Model
- Access Control Lists
- Capability Lists

Textbook

- Chapter 17 (on Protection)
  - Lecture-related: 17.1-17.2, 17.4-17.7, 17.13
  - Recommended: 17.3, 17.8-17.12

Review & Research

- What is meant by “protection” in operating systems?*
- What are the reasons require protection?*
- In what way does the kernel enforce protection?**
- How is the kernel itself protected?**
- Why is hardware support needed to protect the kernel?**

R&R

- What are the goals for the kernel to support protection?*
- In the formal model of protection, what is the definition of each of the following: protection, resource, domain, process?

R&R

- What is a protection matrix?*
- What is represented by the matrix rows?
- What is represented by the matrix columns?
- What is contained in a matrix cell?
- What is the “s” right/permission?*
- Why is the protection matrix inefficient to represent as full matrix?**

R&R

- What is an access control list (ACL)?
- In what way is an ACL like a registry?*
- In what way is inefficiency a disadvantage of ACLs?*
- What is revocation, and why is it an advantage of ACLs?*

R&R

- What is a capability list?
- In what way is a capability like a key/ticket?*
- In what way is efficiency an advantage of a capability?*
- Why is revocation difficult when using capability lists?*

R&R

- What is the design for protection in UNIX?
- What is a protection domain in UNIX?*
- How is the protection model integrated with the UNIX file system?**
- How does UNIX use ACLs and capabilities?**
- What is the SETUID mechanism in UNIX?**

R&R

- If an executable file has the SETUID bit set, how does this affect a process that executes that file?**
- On slide 13, what happens if Chris tries to read the Bib file, and why?*
- What would happen if Pat sets the permission bits to rw-rw-rw, how would it affect your previous answer?*

R&R

- On slide 15, what is the purpose of the EditBib program?*
- Why is it that Pat would be the owner of this file (rather than Chris)?**
- Why is it that the SETUID bit must be set for this file (and not for the Bib file)?**
- Why would it not be good to change the permissions of Bib to rw-rw-rw-?*

R&R

- When Chris's process executes the EditBib file, what does it mean for the process to run in Pat's domain?*
- By running in Pat's domain, doesn't this mean that Chris can now impersonate Pat: why or why not?**