CSci 5271
Introduction to Computer Security
Day 11: Advanced access control

Stephen McCamant

University of Minnesota, Computer Science & Engineering

Outline

Unix-style access control, cont'd

Capability-based access control

Announcements intermission

Multilevel and mandatory access control

Side and covert channels

Non-checks

- File permissions on `stat`
- File permissions on link, unlink, rename
- File permissions on read, write
- Parent directory permissions generally
  - Except traversal
  - I.e., permissions not automatically recursive

"POSIX" ACLs

- Based on a withdrawn standardization
- More flexible permissions, still fairly Unix-like
- Multiple user and group entries
  - Decision still based on one entry
- Default ACLs: generalize group inheritance
- Command line: `getfacl`, `setfacl`

ACL legacy interactions

- Hard problem: don't break security of legacy code
  - Suggests: "fail closed"
- Contrary pressure: don't want to break functionality
  - Suggests: "fail open"
- POSIX ACL design: old group permission bits are a mask on all novel permissions

"POSIX" "capabilities"

- Divide root privilege into smaller (~35) pieces
- Note: not real capabilities
- First runtime only, then added to FS similar to setuid
- Motivating example: `ping`
- Also allows permanent disabling

Privilege escalation dangers

- Many pieces of the root privilege are enough to regain the whole thing
  - Access to files as UID 0
  - `CAP_DAC_OVERRIDE`
  - `CAP_FOWNER`
  - `CAP_SYS_MODULE`
  - `CAP_MKNOD`
  - `CAP_PTRACE`
  - `CAP_SYS_ADMIN (mount)`

Legacy interaction dangers

- Former bug: take away capability to drop privileges
- Use of temporary files by no-longer setuid programs
- For more details: "Exploiting capabilities", Emeric Nasi

Outline

Unix-style access control, cont'd

Capability-based access control

Announcements intermission

Multilevel and mandatory access control

Side and covert channels

ACLs: no fine-grained subjects

- Subjects are a list of usernames maintained by a sysadmin
- Unusual to have a separate subject for an application
- Cannot easily subset access (sandbox)

ACLs: ambient authority

- All authority exists by virtue of identity
- Kernel automatically applies all available authority
- Authority applied incorrectly leads to attacks

Confused deputy problem

- Compiler writes to billing database
- Compiler can produce debug output to user-specified file
- Specify debug output to billing file, disrupt billing

(Object) capabilities

- A *capability* both designates a resource and provides authority to access it
- Similar to an object reference
  - Unforgeable, but can copy and distribute
- Typically still managed by the kernel

Capability slogans (Miller et al.)

- No designation without authority
- Dynamic subject creation
- Subject-aggregated authority mgmt.
- No ambient authority
- Composability of authorities
- Access-controlled delegation
- Dynamic resource creation

Partial example: Unix FDs

- Authority to access a specific file
- Managed by kernel on behalf of process
- Can be passed between processes
  - Though rare other than parent to child
- Unix not designed to use pervasively

Distinguish: password capabilities

- Bit pattern itself is the capability
  - No centralized management
- Modern example: authorization using cryptographic certificates

Revocation with capabilities

- Use indirection: give real capability via a pair of middlemen
- $A \to B$ via $A \to F \to R \to B$
- Retain capability to tell R to drop capability to B
- Depends on composability

Confinement with capabilities

- A cannot pass a capability to B if it cannot communicate with A at all
- Disconnected parts of the capability graph cannot be reconnected
- Depends on controlled delegation and data/capability distinction

OKL4 and seL4

- Commercial and research microkernels
- Recent versions of OKL4 use capability design from seL4
- Used as a hypervisor, e.g. underneath paravirtualized Linux
- Shipped on over 1 billion cell phones

Joe-E and Caja

- Dialects of Java and JavaScript (resp.) using capabilities for confined execution
- E.g., of JavaScript in an advertisement
- Note reliance on Java and JavaScript type safety

Outline

Unix-style access control, cont'd

Capability-based access control

Announcements intermission

Multilevel and mandatory access control

Side and covert channels

Logistics reminders/updates

- Project progress reports are due tonight
- Exercise set 2 is coming soon
  - Will be due, but not graded, before the midterm
  - Watch Piazza for the latest news

Outline

Unix-style access control, cont'd

Capability-based access control

Announcements intermission

Multilevel and mandatory access control

Side and covert channels

MAC vs. DAC

- Discretionary access control (DAC)
  - Users mostly decide permissions on their own files
  - If you have information, you can pass it on to anyone
  - E.g., traditional Unix file permissions
- Mandatory access control (MAC)
  - Restrictions enforced regardless of subject choices
  - Typically specified by an administrator

Motivation: it's classified

- Government defense and intelligence agencies use *classification* to restrict access to information
- E.g.: Unclassified, Confidential, Secret, Top Secret
- Multilevel Secure (MLS) systems first developed to support mixing classification levels under timesharing

Motivation: system integrity

- Limit damage if a network server application is compromised
  - Unix DAC is no help if server is root
- Limit damage from browser-downloaded malware
  - Windows DAC is no help if browser is "administrator" user

Bell-LaPadula, linear case

- State-machine-like model developed for US DoD in 1970s

1. A subject at one level may not read a resource at a higher level
   - Simple security property, "no read up"
2. A subject at one level may not write a resource at a lower level
   - * property, "no write down"

High watermark property

- Dynamic implementation of BLP
- Process has security level equal to highest file read
- Written files inherit this level

Biba and low watermark

- Inverting a confidentiality policy gives an integrity one
- Biba: no write up, no read down
- Low watermark policy
- BLP $\wedge$ Biba $\Rightarrow$ levels are isolated

Information-flow perspective

- Confidentiality: secret data should not flow to public sinks
- Integrity: untrusted data should not flow to critical sinks
- Watermark policies are process-level conservative abstractions

Covert channels

- Problem: conspiring parties can misuse other mechanisms to transmit information
- Storage channel: writable shared state
  - E.g., screen brightness on mobile phone
- Timing channel: speed or ordering of events
  - E.g., deliberately consume CPU time

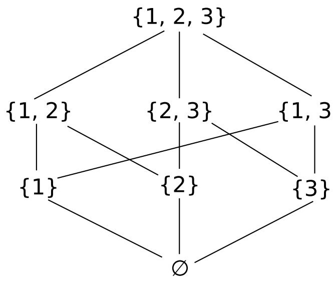
Multilateral security / compartments

- In classification, want finer divisions based on need-to-know
- Also, selected wider sharing (e.g., with allied nations)
- Many other applications also have this character
  - Anderson's example: medical data
- How to adapt BLP-style MAC?

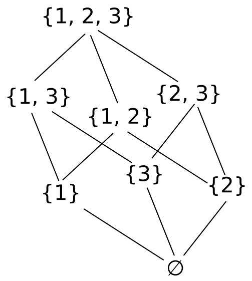
Partial orders and lattices

- $\leq$ on integers is a *total order*
  - Reflexive, antisymmetric, transitive, $a \leq b$ or $b \leq a$
- Dropping last gives a *partial order*
- A *lattice* is a partial order plus operators for:
  - Least upper bound or join $\sqcup$
  - Greatest lower bound or meet $\sqcap$
- Example: subsets with $\subseteq$, $\cup$, $\cap$

Subset lattice example



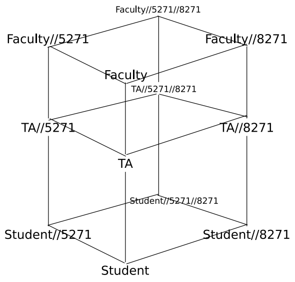
Subset lattice example



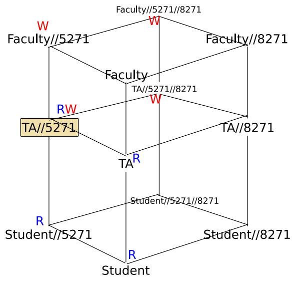
Lattice model

- Generalize MLS levels to elements in a lattice
- BLP and Biba work analogously with lattice ordering
- No access to incomparable levels
- Potential problem: combinatorial explosion of compartments

Classification lattice example



Lattice BLP example



Another notation

Faculty
→ (Faculty, $\emptyset$)
Faculty//5271
→ (Faculty, $\{5271\}$)
Faculty//5271//8271
→ (Faculty, $\{5271, 8271\}$)

MLS operating systems

- 1970s timesharing, including Multics
- "Trusted" versions of commercial Unix (e.g. Solaris)
- SELinux (called "type enforcement")
- Integrity protections in Windows Vista and later

Multi-VM systems

- One (e.g., Windows) VM for each security level
- More trustworthy OS underneath provides limited interaction
- E.g., NSA NetTop: VMWare on SELinux
- Downside: administrative overhead

Air gaps, pumps, and diodes

- The lack of a connection between networks of different levels is called an *air gap*
- A *pump* transfers data securely from one network to another
- A *data diode* allows information flow in only one direction

Chelsea Manning cables leak

- Manning was an intelligence analyst deployed to Iraq
- PC in a T-SCIF connected to SIPRNet (Secret), air gapped
- CD-RWs used for backup and software transfer
- Contrary to policy: taking such a CD-RW home in your pocket `http://www.fas.org/sgp/jud/manning/022813-statement.pdf`

Outline

Unix-style access control, cont'd

Capability-based access control

Announcements intermission

Multilevel and mandatory access control

Side and covert channels

Unintentional information flow

- Generalizing from the last section, want to secure all ways information can get revealed
- It is important to consider all the ways this can happen, even unintentional
- This is a never-ending area of security research, and sometimes a serious vulnerability

Side channel

- A *side channel* is an unexpected way in which a system reveals information
  - Different from how information is intentionally output
- These can pop up in many different ways

Analog side channels

- Mediated by the physical world outside the machine:
  - Sound of the hard-disk running
  - Power usage
  - E-M radiation

Digital side channels

- Reveal information while staying inside the computer abstraction:
  - You can't read a file, but the error message reveals that it exists
  - Running time of an operation depends on what else is running

Covert channels

- In a side channel, the source of information is an unsuspecting victim
- In a covert channel, the source and receive work together to transmit information (contrary to a policy)
- Sometimes the channel can be the same, it's just a matter of usage

Exam analogy

- Side channel: the sound of many people erasing indicates that an exam question is difficult
- Covert channel: cough once if the answer is “true”, twice if it is “false”

Timing channels

- One common source of side/covert channels is effects on the amount of time operations take
- Lots of factors affect performance of computer operations
- There are many ways to measure the passage of time
  - E.g., with parallel operations even without a clock

Classic: SSH keystroke timing

- When typing your password, keys are sent one by one but encrypted
- Longer delays may mean that keys are farther apart
- Statistics and machine learning are often used in decoding