

Security Mechanisms
CS 239
Computer Security
Peter Reiher
January 21, 2004

CS 239, Winter 2004

Lecture 3
Page 1

Outline

- Security tools
- Access control

CS 239, Winter 2004

Lecture 3
Page 2

Tools for Security

- Physical security
- Access control
- Encryption
- Authentication
- Encapsulation
- Intrusion detection
- Common sense

CS 239, Winter 2004

Lecture 3
Page 3

Physical Security

- Lock up your computer
 - Actually, sometimes a good answer
- But what about networking?
 - Networks poke a hole in the locked door
- In any case, lack of physical security often makes other measures pointless

CS 239, Winter 2004

Lecture 3
Page 4

Access Controls

- Only let authorized parties access the system
- A lot trickier than it sounds
- Particularly in a network environment
- Once data is outside your system, how can you continue to control it?
 - Again, of concern in network environments

CS 239, Winter 2004

Lecture 3
Page 5

Encryption

- Algorithms to hide the content of data or communications
- Only those knowing a secret can decrypt the protection
- One of the most important tools in computer security

CS 239, Winter 2004

Lecture 3
Page 6

Encryption is Not a Panacea

- Encryption is usually breakable
 - Given enough time and resources
- Encryption can't protect everything
- Encryption is only as good as the security measures that use it

CS 239, Winter 2004

Lecture 3
Page 7

Authentication

- Methods of ensuring that someone is who they say they are
- Vital for access control
- But also vital for many other purposes
- Often (but not always) based on encryption

CS 239, Winter 2004

Lecture 3
Page 8

Encapsulation

- Methods of allowing outsiders limited access to your resources
- Let them use or access some things
 - But not everything
- Simple, in concept
- Extremely challenging, in practice

CS 239, Winter 2004

Lecture 3
Page 9

Intrusion Detection

- All security methods sometimes fail
- When they do, notice that something is wrong
- And take steps to correct the problem
- Reactive, not preventative
 - But unrealistic to believe any prevention is certain
- Must be automatic to be really useful

CS 239, Winter 2004

Lecture 3
Page 10

Common Sense

- A lot of problems arise because people don't like to think
- The best security tools generally fail if people use them badly
- If the easiest way in is to fool people, that's what attackers will do

CS 239, Winter 2004

Lecture 3
Page 11

The Depressing Truth

- Ultimately, computer security is a losing battle
- Nothing will ever work 100%
- Nothing will work forever
- All your efforts will eventually be undone
- It's like housework – doing it doesn't make the house clean tomorrow, but not doing it guarantees the house is dirty today

CS 239, Winter 2004

Lecture 3
Page 12

Access Control

- Security could be easy
 - If we didn't want anyone to get access to anything
- The trick is giving access to only the right people
- How do we ensure that a given resource can only be accessed by the proper people?

CS 239, Winter 2004

Lecture 3
Page 13

Goals for Access Control

- Complete mediation
- Least privilege
- Useful in a networked environment
- Scalability
- Cost and usability

CS 239, Winter 2004

Lecture 3
Page 14

Access Control Mechanisms

- Directories
- Access control lists
- Capabilities
- Access control matrices

CS 239, Winter 2004

Lecture 3
Page 15

The Language of Access Control

- *Subjects* are active entities that want to gain access to something
 - E.g., users or programs
- *Objects* represent things that can be accessed
 - E.g., files, devices, database records
- *Access* is any form of interaction with an object
- An entity can be both subject and object

CS 239, Winter 2004

Lecture 3
Page 16

Directories

- Each user has a list of the items he can access
 - With the associated rights
- When a user wants to access an item, look it up in his directory

CS 239, Winter 2004

Lecture 3
Page 17

Problems With the Directory Approach

- Per-user directories get very large
 - Overhead and performance problems
- Universal revocation of access
- Pseudonym problems
- Works poorly in networks
- This method is not widely used

CS 239, Winter 2004

Lecture 3
Page 18

Access Control Lists

- For each protected resource, maintain a single list
- Each list entry specifies a user who can access the resource
 - And the allowable modes of access
- When a user requests access to a resource, check the access control list (ACL)

CS 239, Winter 2004

Lecture 3
Page 19

ACL Objects and Subjects

- In ACL terminology, the resources being protected are *objects*
- The entities attempting to access them are *subjects*
 - Allowing finer granularity of control than per-user

CS 239, Winter 2004

Lecture 3
Page 20

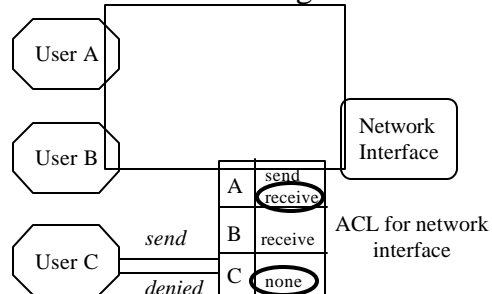
ACL Example

- An operating system example:
 - Using ACLs to protect a network interface device
- User A is allowed to receive from and send to the device
- User B may only receive from it
- User C may not access it

CS 239, Winter 2004

Lecture 3
Page 21

An ACL Protecting a Device



CS 239, Winter 2004

Lecture 3
Page 22

Issues for Access Control Lists

- How do you know the requestor is who he says he is?
- How do you protect the access control list from modification?
- How do you determine what resources a user can access?

CS 239, Winter 2004

Lecture 3
Page 23

ACLs in Practice

- Unix file permissions are a limited form of an ACL
 - Only owner, group, and all can have ACL entries
 - Only read/write/execute controls are available
- Other systems (like modern Windows systems) have more general ACL mechanisms

CS 239, Winter 2004

Lecture 3
Page 24

ACLs and Wildcards

- Can specify a whole range of subjects who share same access rights to object
- E.g., “all members of the software development team can read this file”
- Shortens the lists
- But leads to questions of conflicts

CS 239, Winter 2004

Lecture 3
Page 25

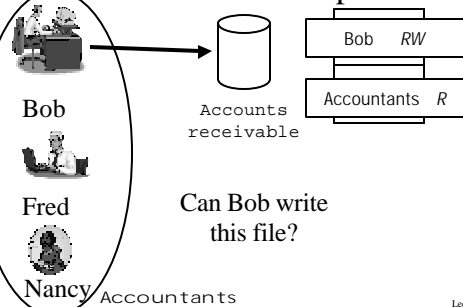
Conflicts in ACLs

- What if a given subject matches more than one rule in an ACL?

CS 239, Winter 2004

Lecture 3
Page 26

ACL Conflict Example



CS 239, Winter 2004

Lecture 3
Page 27

How To Handle ACL Conflicts

- Give most liberal rights
- Give most restrictive rights
- Deal with list in order
 - Giving first rights found
 - Or last rights found

CS 239, Winter 2004

Lecture 3
Page 28

An Example System

- In Unix file access permissions, determine identity
 - Owner, group member, other
- Test only rights for the highest group
- If I own the file, test owner rights
 - Even if I’m in the group and group rights are more liberal

CS 239, Winter 2004

Lecture 3
Page 29

Pros and Cons of ACLs

- + Easy to figure out who can access a resource
- + Easy to revoke or change access permissions
- Hard to figure out what a subject can access
- Changing access rights requires getting to the object

CS 239, Winter 2004

Lecture 3
Page 30

Capabilities

- Each subject keeps a set of data items that specify his allowable accesses
- Essentially, a set of tickets
- Possession of the capability for an object implies that access is allowed

CS 239, Winter 2004

Lecture 3
Page 31

Properties of Capabilities

- Must be unforgeable
 - In single machine, keep under control of OS
 - What about in a networked system?
- In most systems, some capabilities allow creation of other capabilities
 - Process can pass restricted set of capabilities to a subprocess

CS 239, Winter 2004

Lecture 3
Page 32

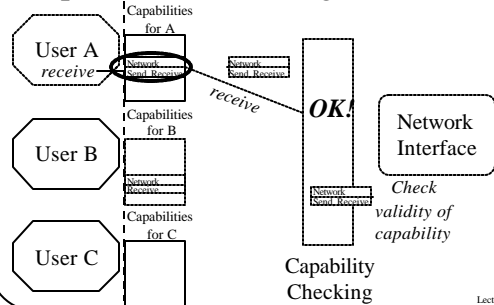
Capabilities and Domains

- The set of objects a subject can access at a given moment is its domain
 - The subject has a capability for each object in its domain
- Domains can be expanded by obtaining new capabilities
- New domains can be created for subprocesses
- Where do we keep capabilities?

CS 239, Winter 2004

Lecture 3
Page 33

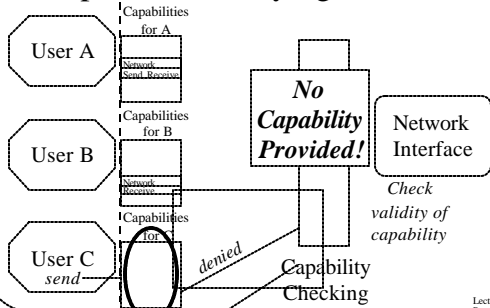
Capabilities Protecting a Device



CS 239, Winter 2004

Lecture 3
Page 34

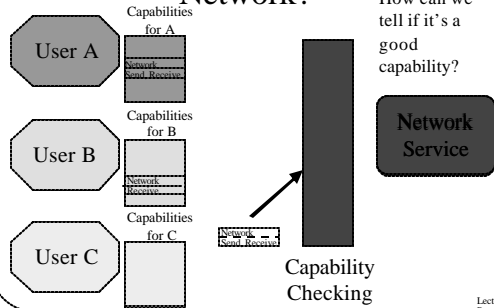
Capabilities Denying Access



CS 239, Winter 2004

Lecture 3
Page 35

How Will This Work in a Network?



CS 239, Winter 2004

Lecture 3
Page 36

Revoking Capabilities

- A major challenge in capability systems
- Several methods available:
 - 1). Search and destroy
 - 2). Invalidation at use
 - 3). Indirection through a token
 - 4). Generation numbers

CS 239, Winter 2004

Lecture 3
Page 37

Revocation By Destroying Capabilities

- Find the capability you want to revoke
- Destroy it
- Easy if all capabilities live only in system-controlled memory
- But most systems allow storage of capabilities on disk
- And what about networked systems?

CS 239, Winter 2004

Lecture 3
Page 38

Revocation By Invalidation on Use

- Keep a list of revoked capabilities
 - Usually one list per object
- When a capability is presented for use, check it against the list
- Expensive, especially if the list is long or complete mediation is used
 - Not feasible on every access
- And what about networked systems?

CS 239, Winter 2004

Lecture 3
Page 39

Revocation by Indirection Through a Token

- Capability points to token under system control
- Token is set up on first access to object
- To revoke access, destroy the token
- Adds cost to checking access
- Usually hard to provide selective revocation

CS 239, Winter 2004

Lecture 3
Page 40

Revocation By Generation Numbers

- Store a random number in each capability
- Store the same random number with the protected object
- On access, check the numbers
- To revoke access, change the number
- No selective revocation
- Requires some control of capabilities

CS 239, Winter 2004

Lecture 3
Page 41

Pros and Cons of Capabilities

- + Easy to determine what a subject can access
- + Potentially faster than ACLs (in some circumstances)
- + Easy model for transfer of privileges
 - Hard to determine who can access an object
 - Requires extra mechanism to allow revocation
 - In network environment, need cryptographic methods to prevent forgery

CS 239, Winter 2004

Lecture 3
Page 42

Capabilities and Directories

- How are capabilities any different than maintaining a per-subject directory?

CS 239, Winter 2004

Lecture 3
Page 43

ACLs, Capabilities, Complete Mediation, & Performance

- Ideally, every data access should have access control independently applied
- Practicality of doing so depends on the performance costs
- What does it cost to use ACLs?
 - Capabilities?

CS 239, Winter 2004

Lecture 3
Page 44

Performance Issues of Access Control

- What if the status of the access control mechanism changed between when last checked and current access?
- Common case is nothing changes
- Different approaches possible
 - Actually check changeable data structure on each access
 - Give process something cheap and revocable that allows access

CS 239, Winter 2004

Lecture 3
Page 45

Access Control and ACLs

- The ACL is a list
- Initially, checking an ACL involves searching a list
- For later checks, maintain pointer to list entry
- Be sure that changing the permissions changes what's pointed to

CS 239, Winter 2004

Lecture 3
Page 46

Access Controls and Capabilities

- Attach the capability (or pointer to it) to each request
- Use attached information to determine if current access is permissible
- This approach is hard to use with revocation

CS 239, Winter 2004

Lecture 3
Page 47

An Alternate Approach To Using Capabilities

- On first access, use a capability to obtain an access token
 - Using careful, expensive checks to see if capability was revoked
- If revocation required, destroy the access token
- Can also be done with pointers

CS 239, Winter 2004

Lecture 3
Page 48

Access Control in the Distributed World

- ACLs still work OK
 - Provided you have a global namespace for subjects
- Capabilities are more problematic
 - Their security relies on unforgeability

CS 239, Winter 2004

Lecture 3
Page 49

Using Cryptographic Capabilities

- Can cryptography make capabilities unforgeable?
- It can make it impossible to create them from nothing
 - And only usable by their owner
- But it can't make them uncopyable
- So cryptographic capability systems must assume they can be freely copied

CS 239, Winter 2004

Lecture 3
Page 50

Access Control Matrices

- A very general access control concept
- In principle, ACLs are a 1-D list of who is permitted to access one object
- And capabilities are a 1-D list of what one subject can access
- Access control matrices are a 2-D description of access rights

CS 239, Winter 2004

Lecture 3
Page 51

Access Control Matrix Example

	File A	File B	Network	Printer	Objects
User 1	rw	r		w	
User 2	r		sr	w	
Sysadmin	rw	rw	sr	rw configure	
Guest			sr		
Subjects	File B's ACL				

CS 239, Winter 2004

Lecture 3
Page 52

Pros and Cons of Access Control Matrices

- + Makes all access issues explicit and easy to find
- + Easy to tell who can access a resource, and what resources anyone can access
 - Matrix very sparse, so inefficient
 - Hard to achieve good performance
- More important conceptually than in implementations

CS 239, Winter 2004

Lecture 3
Page 53

Role Based Access Control

- Not really an alternative to ACLs, capabilities, access control matrix
- Rather, a more complex way of looking at access control subjects
- Commonly used in systems that care about security

CS 239, Winter 2004

Lecture 3
Page 54

The Idea Behind Role Based Access Control

- Each user has certain roles he can take while using the system
- At any given time, the user is performing a certain role
- Give the user access to only those things that are required to fulfill that role

CS 239, Winter 2004

Lecture 3
Page 55

A Simple Example

- Fred is a system administrator
 - Which requires him to install programs, examine logs, etc.
- Fred also reads email, looks at web sites, etc., like any other user
- Fred should operate under one role while doing normal work
 - And a different role while performing administrative tasks

CS 239, Winter 2004

Lecture 3
Page 56

Continuing With the Example

- Fred logs on as “fred”
- He reads his email as “fred”
- He decides to upgrade the C++ compiler
 - So he changes roles to “administrator”
- When he’s done, he returns to the role of “fred”

CS 239, Winter 2004

Lecture 3
Page 57

What Has Been Gained?

- While reading mail and surfing the web, Fred isn’t able to upgrade the C++ compiler
 - He doesn’t have the access rights
- So if he accidentally downloads malicious code, it can’t “upgrade” the compiler

CS 239, Winter 2004

Lecture 3
Page 58

Changing Roles

- Role based access control only helps if changing roles isn’t trivial
 - Otherwise, the malicious code merely changes roles before doing anything else
- Typically requires providing some secure form of authentication
 - Which proves you have the right to change roles

CS 239, Winter 2004

Lecture 3
Page 59

Practical Limitations on Role Based Access Control

- Number of roles per user
- Problems of disjoint role privileges
- System administration overheads

CS 239, Winter 2004

Lecture 3
Page 60

Number of Roles Per User

- Each new role requires new authentication
- Less secure if the authentication is the same for each role
 - E.g., Unix `sudo`, which only requires your basic password
- How many passwords will people remember?
 - And how often will they be happy to type them?

CS 239, Winter 2004

Lecture 3
Page 61

Problems of Disjoint Roles

- Each role should have disjoint privileges
 - More secure if roles aren't supersets of other roles
- May cause difficulties if certain operations require privileges from different roles

CS 239, Winter 2004

Lecture 3
Page 62

Problems of System Administration

- Access control is only useful if the permissions are set correctly for each subject and object
- The more subjects there are, the more work system administrators must do
 - Since each subject needs to get only the proper privileges

CS 239, Winter 2004

Lecture 3
Page 63

Reference Monitors

- Whatever form it takes, access control must be instantiated in actual code
 - That checks if a given attempt to reference an object should be allowed
- That code is called a reference monitor
- Obviously, good reference monitors are critical for system security

CS 239, Winter 2004

Lecture 3
Page 64

Desirable Properties of Reference Monitors

- Correctness
- Proper placement
- Efficiency
- Simplicity
- Flexibility

CS 239, Winter 2004

Lecture 3
Page 65

An Example Reference Monitor

- The Linux code that mediates file access
- Applied on relatively few of the file system calls
 - Open, execute, directory traversal, a few others
 - Not on read and write

CS 239, Winter 2004

Lecture 3
Page 66

Another Example Reference Monitor

- A firewall
- It examines every packet for certain characteristics
- Typically, either any subject can do something or no subject can
- But sometimes packets from particular source addresses can do more
 - Essentially, the source address identifies a privileged subject

CS 239, Winter 2004

Lecture 3
Page 67

Thinking More Broadly About Access Control

- From one perspective, access control is the core of all computer security
- All security is about who can access what
- So where do security problems come from?
 - Not applying access control
 - Not applying access control properly

CS 239, Winter 2004

Lecture 3
Page 68

What Is the Most Common Access Control Mechanism?

- The null mechanism
- Let anyone do anything they want
- Sounds terrible, but it's actually the key to the success of computers and networks

CS 239, Winter 2004

Lecture 3
Page 69

Why Is Null Access Control Ever Good?

- Any user can run an instruction on a CPU without necessarily checking access control
- Any packet can be handled by a router without checking access control
- The trick is to apply access control when it's most important
 - And to apply it properly

CS 239, Winter 2004

Lecture 3
Page 70

Problems Arising From Null Access Control

- Spam
- Distributed denial of service
 - And most other denials of service
- Buffer overflows
- Worms

CS 239, Winter 2004

Lecture 3
Page 71

Conclusion

- Much of security relates to allowing some people access to some resources
- While preventing the same access to others
- Without some method of determining who should access what . . .
You can't do that

CS 239, Winter 2004

Lecture 3
Page 72