CSE543 - Introduction to Computer and Network Security

Module: Capability Systems

Professor Trent Jaeger
Confused Deputy

• Imagine a **multi-client server** (e.g., web server)
  ‣ Clients have a different set of objects that they can access
• In an ACL system, the **server always has access to all the objects**
  ‣ What happens if a client tricks the server into accessing into another client’s objects or sensitive server objects?
  ‣ Shouldn’t the server only have access to that client’s objects for its requests?
• Leads to **confused deputy**...
Action-specific Permissions

• Ideally, we would like the permissions of a request to only those available to the requesting client.

• How do we *change the permissions* of a process in an ACL system?
Capabilities

• Each row lists the access rights of a subject
  ‣ Capabilities

• In theory, a server could use capabilities to change permissions
  ‣ Choose among subjects with the desired permissions
  ‣ Or choose to activate the permissions to apply for the specific case

• How can programmers use such a concept?
  • Will they?
Capabilities

• A capability is the tuple \( (\text{object}, \text{rights}) \)

• A capability system implements access control by checking if the process has an appropriate capability
  ‣ Simple, right? Analogous to keys for a lock
  • Can use or delegate (copy) for others
  ‣ This is a little like a ticket in the Kerberos system

• Q: Does this eliminate the need for authentication?
Capabilities

• A: Well, yes and no …

• Capabilities remove the overhead of managing per-object rights, but add the overhead of managing capabilities

• Moreover, to get any real security, they have to be unforgeable

  ‣ Hardware tags (to protect capabilities)
  ‣ Protected address space/registers
  ‣ Language based techniques
    • Enforce access restrictions on caps.

  ‣ Cryptography
    • Make them unforgeable
User space capability?

• Well, what are the requirements?
  ‣ Authenticity/integrity - do not want malicious process to forge capabilities

• Start with the data itself: [object, rights]
  ‣ Object is typically encoded with identifier, or by some other tag (capabilities are sometimes known as tags)
  ‣ Rights are often fixed (read, modify, write, execute, etc.)

• Now, do what you with any other data (assume the kernel has a secret key k)

\[ E(k, [O_i, r_1, r_2, \ldots r_n]) \]

• What’s wrong with this construction (I got it from the website of one of the experts in the area)?
The right construction

- Encryption does not provide authenticity/integrity, it provides confidentiality instead

\[[O_i, r_1, r_2, \ldots r_n], \text{HMAC}(k, [O_i, r_1, r_2, \ldots r_n])\]

- So how would you attack the preceding construction?
The OS kernel manages capabilities in the process table, out of reach of the process.

Capabilities added by user requests (delegation)
A (fictional) Capability Example

• We use the “ls -lt” command to view the contents of our home directory in a OS implementing capabilities:
  ‣ Initially, our shell process has RWX capabilities for our home directory, and RX capabilities for all the directories to the root.
  ‣ The “ls -lt” command is exec()ed, and the shell delegates the directory permissions by giving “ls” the capabilities
    • Note that the capabilities are _not_ tied to any subject - copied to “ls” C-list
  ‣ The “ls -lt” process exercises the rights to read the directory structure
  ‣ Of course, the “ls -lt” process now needs to obtain read rights to the files (to get their specific meta-information), and obtains them by appealing to the security manager (in kernel) -- the request fulfills the policy, and they are added and exercised
    • The “ls -lt” uses access rights given to the terminal to write output
• Note: there are many ways that the policy can be implemented, rights handed off, etc.
Capabilities Between Procs

- So, how do we use capabilities to prevent confused deputy attacks on web servers?
So, how do we use capabilities to prevent confused deputy attacks on web servers?

- Client sends a request to a web server
- Request includes client’s capabilities for web server
  - E.g., files that may be read (served) to the client
- Web server uses the client’s capabilities to process the request
  - Provides capabilities in system call
  - Like file descriptors
- How does this prevent confused deputy attacks on web servers?
  - ???
Capabilities Between Procs

- So, how do we use capabilities to **prevent confused deputy attacks** on web servers?
  - Client sends a request to a web server
  - Request includes client’s capabilities for web server
    - E.g., files that may be read (served) to the client
  - Web server uses the client’s capabilities to process the request
- How does this prevent confused deputy attacks on web servers?
  - Client cannot gain unauthorized access if only accesses are made using the client’s capabilities
Capabilities Between Procs

• Another way to use capabilities
  • The client has a capability to communicate with a server (i.e., write messages to the server)
  • A client can provide a write capability to the server for a file
  • The client has read-write access (capability) to this file

• Is there a problem with this situation?
Earl Boebert found the following problem:

So, consider a high secrecy Trojan horse program

- Low secrecy programs can write to high secrecy programs
- Suppose the low secrecy program writes a capability to write a low secrecy file to the high secrecy program
  - E.g., User-space capabilities are just data

Then, the high secrecy Trojan horse has a capability to write to a low secrecy file

- Can use this capability in an operation to violate the *-property

What would you do to prevent this?
Capability Systems

• **Solutions** to this problem have been produced
  ‣ Compare capability to MLS before use
  ‣ Compare capability to MLS before propagation

• **Not really holding up** use of capability systems
  ‣ Security people like capability systems
  ‣ In general, prevents all kinds of confused deputy attacks
    • If programmers use the right capabilities at the right time

• Problem is that **programmers won’t use capability systems**
  ‣ And programmers often make mistakes programming simple security code, so ...
Procedure-Level Protection Domains

- **HYDRA (CMU, 1970s)**
  - Each procedure defines a new protection domain

- **Procedure**
  - Code
  - Data
  - Capabilities to other objects
    - Caller-independent
    - Caller-dependent templates

- **Local Name Space (Stack Frame)**
  - Capabilities are bound here
  - Record of a procedure invocation (procedure instance)

- **Process**
  - Stack of LNSs
How HYDRA works

- Q: Which object defines the protection domain?
Implications of Fine-Grained Protection

• Programmer
  ‣ Must define *templates* for procedure
  ‣ Connect the procedure rights together

• Programmer is *responsible* for
  ‣ Functionality
  ‣ And security
  ‣ At the same time
Programming for Capabilities

• Programmer
  ‣ Must use the right capabilities for security-sensitive operations
  ‣ Determine which capabilities to delegate to other processes

• Programmer is responsible for
  ‣ Instantiation
  ‣ Use
  ‣ Delegation
  ‣ Revocation

• Proven difficult to convince programmers to do these tasks
Can the System Help?

• Can the system enforce capabilities without programmer modifications?
  • How about for name resolution attacks?
  • File squat

• Remember, we have figure out what the programmer expects for each system call?
  ‣ Will the programmer tell us?
  ‣ Can we estimate?
Can the System Help?

• Can the system enforce capabilities without programmer modifications?
  • How about for name resolution attacks?
• Suppose a system call is only supposed to open a protected file (SSL private key)
  • Attacks may be possible: E.g., File squat
• For a system call, ensure that only protected files are retrieved
  • How to achieve this defense?
Can the System Help?

- Can the system enforce capabilities without program modifications?
  - How about for name resolution attacks?

- Suppose a system call is only supposed to open a protected file (SSL private key)
  - Attacks may be possible: E.g., File squat

- For a system call, ensure that only protected files are retrieved
  - How to achieve this defense?
  - For that specific open (library) call, restrict the set of permissions available to the process (protected files)
Process Firewall

**UserSpace**
- Students
- Serve HTML File
- Check Passwd
- Introspect

**Operating System**
- Access Control
- Process Firewall
- File
- Passwd
- Dir
- IPC
- Signal

**Process Context:** Entrypoint, Call Stack, etc.
Process Firewall

• This defense can be implemented efficiently
  • Much more efficiently than program-internal defenses
• However, the system needs to know the programmer intent
• May be easier to specify declaratively to system rather than modifying program code
  • Still an open research problem
Program to System

- For name resolution defenses to prevent confused deputy attacks
- (1) Programmer may use POSIX API (stat, lstat, fstat - don’t prevent all attacks)
- (2) Programmer may leverage capability systems (programmer needs to reason about the system)
- (3) System may enforce context-specific rules (introspection into the process - e.g., Process Firewall)
- (4) System enforces access control (even MAC allows confused deputy attacks)
Take Away

• There are some vulnerabilities MAC cannot prevent
  ‣ Could trick the victim into using its permissions for the unauthorized operations for the adversaries
  ‣ Confused Deputy vulnerabilities
• These vulnerabilities are still common (and latent) in many programs
• Can prevent confused deputy attacks using capability systems
  ‣ A capability system enables programmers to determine permissions for individual functions (Hydra)
  ‣ Requires more effort from programmers
  ‣ Have system infer capabilities per syscall (Process Firewall)