Security and Privacy

Access Control

30.04.2019

slide credits: Cristina Basescu, Ph. Oechslin

EPFL

Outline

- Access control
  - Definitions
  - Role based
  - Discretionary
    - ACLs, capabilities in Linux
    - ACLs, capabilities in Windows
  - Mandatory Access Control
    - MAC Linux & Windows

Definitions

Definitions

- Access control defines and enforces the operations that subjects can do on objects
  - e.g does bob (subject) have permission to read (operation) from a socket (object)
- Implies that the subject has bee authenticated first (later today)
- Access rights (aka permissions, privileges) describe which subjects can do which operations on which objects
  - They define a security policy and can be represented as an access control matrix
- Security mechanisms try to prevent operations that are not authorized by the security policy

Definitions

- Principle of least privilege
  - Subjects should have the minimum rights (operations on subjects) necessary to do their job
  - This limits the impact if anything should go wrong
  - Most important principle of access control!
- The challenge in access control is to have a system that is simple to implement and manage and that is close to the principle of least privilege
- There is no 'one-size fits all' solution and often different approaches to access control are combined to achieve the best results.

Multiple levels of access control

- Network level access control
  - subjects are connections or data packets
  - they are identified by their source/destination IP addresses and protocol ports
  - typical operations are pass, block, tag
  - Example: a database server only accepts traffic from inside EPFL (source addr 128.*.*.*) connecting to TCP port 3306 (mysql).
- Typically enforced with
  - network equipment: firewalls
  - the servers:
    - local firewall on the server (try gufw in Linux)
    - configuration of the server software

Multiple levels of access control

- Operating system access control
  - Which user can start/stop the DB engine?
  - Who can read/modify the files of the DB?
- Access control in the application
  - Which user of the application can edit user profiles?
  - Who can see financial data?
- Access control within the enterprise
  - Which employees can access the application?
  - Which applications are limited to human resources, which to marketing?

Multiple approaches to access control

Three common varieties:

- Role-based Access Contorl (RBAC)
- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC[1])

[1] not message authentication code

Role based Access Control

Role-based Access Control (RBAC)

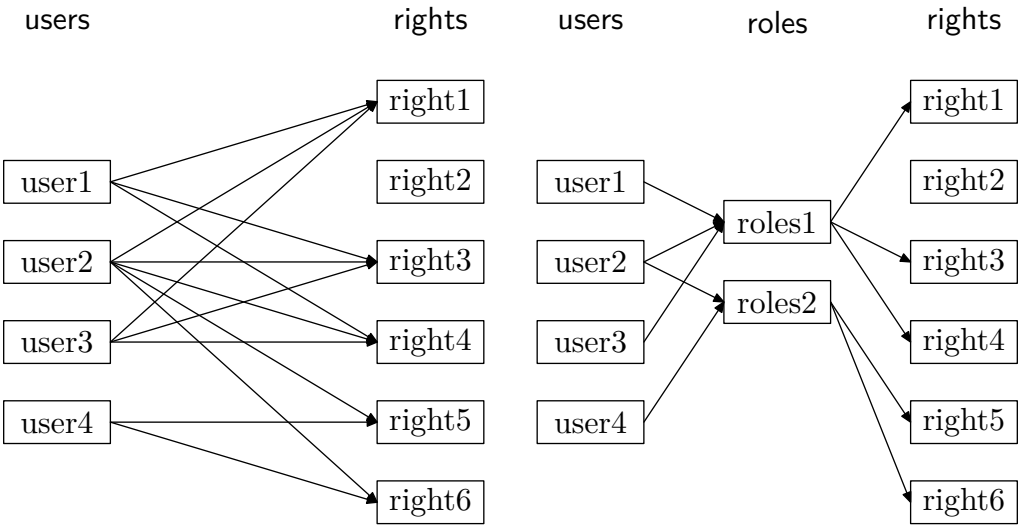
- Simplifies the specification of permissions by grouping users into roles
- Centered on user roles
  - a role can contain multiple permissions

2 Moodle Course Roles

Access to individual Courses	View Courseware	Participate in activities	View Personal records	View student records	Add and edit Courseware	Edit and Course Setting	Set Course Roles
Course Administrator	✓	✓	✓	✓	✓	✓	✓
Teacher	✓	✓	✓	✓	✓		
Non-editing Teacher	✓	✓	✓	✓			
Student	✓	✓	✓				
Guest Read only access	✓						

source: **wisenet**

RBAC: user mgmt within the company



Implementing RBAC

- JAVA:
  - Java Authentication and Authorization Service (JAAS)
  - Uses plug-ins to authenticate users with existing methods (Unix, Windows, LDAP)
  - Uses a policy file to define the permissions
- Operating systems
  - Most operating systems have the notion of groups
  - Groups can be given a set of permissions
  - Users can be added to groups
  - examples
    - Debian/Ubuntu: audio group can access mic and loudspeakers, wireshark group can sniff network traffic
    - Windows: "Remote desktop user" group can access desktop remotely

Advantages of RBAC

- ✓ Easy to grasp the idea of roles
- ✓ Easy to manage
  - Roles decouple digital entities from permissions
  - Simply assign roles to a new subject
    - no need decide for each resource
  - Easy to revoke authorizations by removing role
- ✓ Easy to tell through roles which permissions a subject has and why
  - Typically centrally managed

Disadvantages of RBAC

✗ Difficult to decide on the granularity of roles
- create separate roles for modifying client information and for deleting client, or not?
- Leads either to role explosion or roles that are too broad (not least privilege)

✗ Role meaning is fuzzy
- Employee position in company may be different from RBAC role (think developers in same team working on different subjects)

✗ Unclear if roles can be shared across different departments
- Is a finance IT manager the same as a marketing IT manager?

Discrectionary Access Control

Discretionary Access Control (DAC)

- Access control is at at the discretion of the object owner
  - owner specifies policies to access resources it owns
- Access control matrix represents rules
  - stored by column: access control list (ACL) stored with resource

	/stud/grades.txt	/hw1/grade.sh	/sensitive
stud1	r–	–x	—
TA1	rw-	rwx	r-x

  - stored by row: capabilities stored with subjects

	/stud/grades.txt	/hw1/grade.sh	/sensitive
stud1	r–	–x	—
TA1	rw-	rwx	r-x

ACL vs Capabilities

- Think door protected by a bouncer vs a lock
- ACL (bouncer):
  - the bouncer know exactly who can get in
  - people don't know where they will get in and where they wont
- Capabilities (key)
  - doors don't know who will show up with a key
  - people know exactly for which door they have a key
- ACL is practical when you often have to create or modify rights on objects
- Capabilities, when you often create or change rights of subjects or roles

ACLs in Unix

- Typically done with ACL
- Stored in the target object, e.g in the metadata of files in the file system
- Subjects are grouped in three categories : owner, group, other
- Three access rights: (r)ead, (w)rite, (e)xecute
  - for directories:
    - r: directory can be listed
    - w: directory can be modified (create, delete, rename files)
    - x: directory can be accessed by the `cd` command
- The three rights and three groups are stored in 9 bits
  - represented as three octal digits
  - owner `rwx`, group `rx`, others `r` : `rwx|r-x|r--` = 754

ACLs in Unix: example

Remember the ubuntu local firewall GUI gufw:

- only root can read or write the files and the directory:

```
pho:/etc$ ls -l gufw/
total 44
drwxr-xr-x 2 root root 28672 avr 24 08:17 app_profiles
-rw------- 1 root root    73 avr 23 15:10 gufw.cfg
-rw------- 1 root root  1079 avr 23 11:09 Home.profile
-rw------- 1 root root    76 avr 18 11:40 Office.profile
-rw------- 1 root root    78 avr 18 11:40 Public.profile
```

- everybody can read the application profiles, but only root can modify them:

```
pho:/etc$ ls -l gufw/app_profiles/ssh.gufw_service
-rw-r--r-- 1 root root 213 mai 24  2017 gufw/app_profiles/ssh.gufw_service
```

ACLs in Unix: setuid/setgid

- If a program has `setuid` bit set, it will be run with the permissions of the owner of the file instead of the permissions of the user running the program.
- Very useful to give users more privileges in specific cases
- Example: the `passwd` command allows a user to modify the `/etc/passwd` and `/etc/shadow` files.
  - `passwd` can be read by all but only modified by `root`
  - `shadow` can be read by `root` and `shadow` and modified by `root`

```
pho:~$ ls -l /etc/{passwd,shadow}
-rw-r--r-- 1 root root   2786 avr 23 11:09 /etc/passwd
-rw-r----- 1 root shadow 1489 avr 23 11:09 /etc/shadow
```

  - the program `/usr/bin/passwd` has the setuid bit

```
pho:~$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 59640 jan 25  2018 /usr/bin/passwd
```

ACLs in Unix: setuid/setgid

- When user Jane runs the program `passwd`, the process is run as root:

```
pho:~$ ps -ef | grep passwd
root     16003 26651  0 09:26 pts/1    00:00:00 passwd
```

- The setgid bit does the same for groups: the group of the process running the program is set the group of the owner of the program.
- setuid and setgid is displayed as `s` instead of x in the access rights of the file
- Example: this program has both setuid and setgid bits set:

```
pho:~$ ls -l test
-rwsrwsr-x 1 philippe philippe 0 avr 25 08:36 test
```

ACLs in Unix: setuid/setgid

- Finding files that have the setuid bit set:

```
pho:~$ find / -perm /u=s 2>/dev/null
/bin/fusermount
/bin/mount
/bin/su
/bin/umount
/bin/ping
...
```

- Setuid is very practical, because it lets simple users execute some well defined privileged actions

QUIZ

- setuid can be very dangerous, why?

Capabilities in Linux

- Remember: capabilities are permissions that are related to a subject, not to an object
- Linux supports capabilities for processes. Some examples are
  - `CAP_CHOWN`: make arbitrary changes to file user ID and group ID
  - `CAP_DAC_OVERRIDE`: Bypass file read, write, and execute permission checks
  - `CAP_SYS_BOOT`: use reboot or kexec (load new kernel)
- Example: `dumpcap` is the program used by wireshark to sniff network traffic.
  - It can only be run by user `root` and members of the `wireshark` group
  - It does not have the setuid bit

```
pho:~$ ls -l /usr/bin/dumpcap
-rwxr-xr-- 1 root wireshark 104688 jan 19 06:23 /usr/bin/dumpcap
```

Capabilities in Linux

- Let's check the capabilities of dumpcap:

```
pho:~$ getcap /usr/bin/dumpcap
/usr/bin/dumpcap = cap_net_admin,cap_net_raw+eip
```

  - It has the `net_admin` and and `net_raw` capabilities:
  - ➡ It can read and write to all network interfaces

- The program can do this while running in the name of the user:

```
pho:~$ ps -ef | grep dumpcap
philippe 24342 26651  0 13:10 pts/1    00:00:00 /usr/bin/dumpcap#
```

- This is much safer than using setuid
  - If there was a bug in dumpcap allowing to execute arbitrary commands, these commands would be run as root!

ACLs in Windows

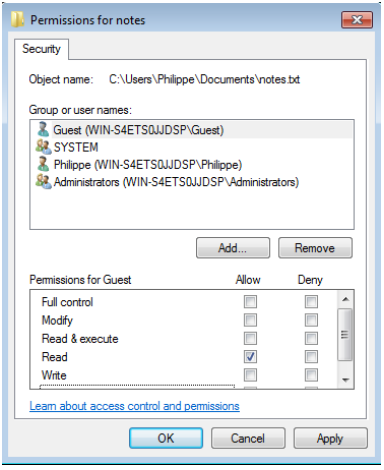
- Windows has more access rights than unix:

source: **Microsoft**

- For example there is a different right for writing and deleting
- The ACL is not limited to three types of subjects (user, group, other)
  - Objects can have a list of ACLs for different users and groups

ACLs in Windows

- Example: the rights for document notes are set for four different subjects:



ACLs in Windows

- The Windows administrator account is not the highest privileged account
- The system (`NT Authority\system`) account is the one that runs the system and launches services
- Thus you can configure anti-virus software in a way that even administrators can not remove it

Capabilities in Windows

- Windows has privileges that act like capabilities
  - SeTimeZonePrivilege: can change timezone
  - SeSystemtimePrivilege: can change the system time
  - SeShutdownPrivilege: can shutdown the system
  - SeDebugPrivilege: can debug and access memory of other processes
- You can check your privileges with

```
C:\Users\user>whoami /priv
Informations de privilèges----------------------

Nom de privilège               Description                                État
============================== ========================================== =========
SeShutdownPrivilege            Arrêter le système                         Activé
SeChangeNotifyPrivilege        Contourner la vérification de parcours     Activé
SeUndockPrivilege              Retirer l'ordinateur de la station d'accueil Désactivé
SeIncreaseWorkingSetPrivilege  Augmenter une plage de travail de processus  Désactivé
SeTimeZonePrivilege            Changer le fuseau horaire                  Désactivé
```

DAC Pros and Cons

- Advantages
  - Flexible
  - Easy to manage (owners get to set the permissions themselves)
  - Intuitive
- Disadvantage
  - Depends on the owners judgment
  - Only works if programs are not malicious and users make no mistakes
  - Vulnerable to the "Trojan" problem
    - A malicious program run by an authorized user can read a protected file and write an unprotected copy of that file

    ➡ Anybody can now read the file

Mandatory Access Control (MAC)

Mandatory Access Control

Tries to ensure that even someone with access cannot leak the data.

- Historically associated with military-grade information security
  - Multilevel security: e.g unclassified, confidential, secret, top-secret
- The system labels both subjects and objects with security labels
  - Can only be modified by trusted administrators via trusted software
- Security policy:
  - Example: Subjects can only access objects of same or lower level

subject \objects	**top-secret**	**secret**	**confidential**	**unclassified**
top-secret	read, write	read	read	read
secret		read, write	read	read
confidential			read, write	read
unclassified				read, write

Mandatory access control

- Depends on trusted software and admins for
  - keeping the system in a protected state
    - preventing operations that violate the rules of the matrix
  - labeling new subjects and objects
  - perform transitions of labels (e.g. when a document is declassified)
- Can be used in conjunction with DAC or RBCA

QUIZ!

- Why is it a bad idea to allow secret subjects to write confidential documents?

MAC confidentiality vs integrity

- MAC and Confidentiality
  - When protecting confidentiality, we don't want users to write to a lower level (no write-down)
    - prevents leaking information from higher levels to lower levels ('trojan' problem)
  - Typical scenario: network access control
    - network split in zones: internet, internal, secret
    - firewalls only allow data to flow from lower zones to higher zones
- MAC and Integrity
  - we don't want users from lower level to write into higher levels (no write-up)
    - prevents unauthorized modification objects
  - Typical scenarios: operating systems
    - users can read and execute programs of the OS, but they can not modify them

MAC: MAC in Windows

- Windows implements MAC for integrity protection (Mandatory Integrity Control)
- There are four integrity levels: low, medium, high, system
- Objects have a label that says whether write-up, read-up or execute-up are allowed
  - Subjects of a lower integrity level can thus not write, read or execute the object
- Internet browsers and other programs processing files received from Internet (Acrobat reader, Ms-Office in "protected view" mode) run in low integrity level
  - They can only write to directories of low integrity level (e.g. AppData/LocalLow)

MAC: Windows example

- You can check the integrity level of a file or directory with the `icacls` command:

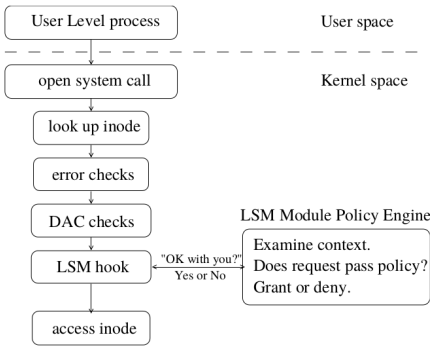
```
C:\Users\user\AppData>icacls LocalLow
LocalLow AUTORITE NT\Système:(OI)(CI)(F)
         BUILTIN\Administrateurs:(OI)(CI)(F)
         os-PC\user:(OI)(CI)(F)
         Étiquette obligatoire\Niveau obligatoire faible:(OI)(CI)(NW)
```

- *Niveau obligatoire faible* means low integrity level
- You can set the integrity level with `icacls`

```
C:\Users\user\>icacls test.txt /setintegritylevel h
```

MAC: Linux examples

- SELinux and AppArmor are two MAC systems for Linux
- They are both based on the generic Linux Security Module (LSM)
- LSM sits in the kernel and is called just after standard DAC checks have been done and before access is given
  - it is used to implement additional security policies



MAC: Linux examples

- SELinux
  - Every user has a context made of name, role and domain
  - Files, ports and other objects can be labeled with name, role and type
  - Rules can be defined to allow certain actions
- SELinux is implemented in Android: better isolation of apps and generic services

MAC: Linux examples

- AppArmor
  - Also based on LSM
  - Uses profiles to define access rights to files, network and capabilities
  - There are no labels or security levels
  - Profiles basically define the same rules that can be defined with DAC, but they can not be modified at the discretion of the owner of the objects or subjects.
  - Profiles can be generated by observing a running application
  - AppArmor is enabled by default in Ubuntu
- AppArmor demo: copy a PDF file into you .ssh directory and try to open it with evince!

MAC Pros and Cons

- ✓ Addresses the limitations of DAC
- ✓ Easy to scale
- ✗ Can be too restrictive, prevent legitimate tasks
- ✗ Not flexible

Conclusions & Questions

Conclusions

- Different types of access control (RBAC, DAC, MAC) are used depending on the situation
  - the goal is to reach least privilege and not be complex
- Modern OSes make use of all of these types
  - DAC with ACL for files and most objects
  - DAC with capabilities for privileged operations
  - Using groups to implement RBAC (users, admins, hr, marketing)
  - MAC for protecting the integrity of the system

Questions

- What is the most important principle of access control?
- What is the difference between ACLs and capabilities?
- What are the disadvantages of RBAC?
- Why are some linux programs setguid `shadow`?
  - check the permissions of `/etc/shadow` to find out
- When MAC is used to protect integrity, can we write up or write down?