Enrico RUSSO

Andrea VALENZA

Università di Genova

UNIX file permission

CYBER CHALLENGE.IT

CYBERSECURITY NATIONAL LABORATORY

https://cybersecnatlab.it

License & Disclaimer

License Information

Disclaimer

- We disclaim any warranties or representations as to the accuracy or completeness of this material.
- Materials are provided “as is” without warranty of any kind, either express or implied, including without limitation, warranties of merchantability, fitness for a particular purpose, and non-infringement.
- Under no circumstances shall we be liable for any loss, damage, liability or expense incurred or suffered which is claimed to have resulted from use of this material.

Outline

- Unix access control
  - Principals, subjects and objects
  - Permissions
  - Access control decisions

Principals and Subjects

- A **subject** is a program executing on behalf of some **principals** and operating on system resources (**objects**)
- A **principal** may at any time be idle or have one or more **subjects** associated

Linux Principals

- Principals are **users** and **groups**
- Users have a **username** and a **user identifier** (UID)
- Groups have a **group name** and a **group identifier** (GID)
- UID and GID are 32bit numbers (16 before kernel 2.4)

UID	username
0	root
1	daemon
2	bin

GID	group name
0	root
4	adm
8	mail

User accounts

- User account are stored in `/etc/passwd`

```
user:   $6$y4QdtE/[…]:   1000:   1000:   User,,,,:   /home/user:    /bin/bash
```

1 2 3 4 5 6 7

1. **Username**
2. **Password**: stored as digest (hashed)
3. **User ID**
4. **Group ID**: user's primary group
5. **ID string**: user's full name
6. **Home directory**
7. **Login shell**: the program started after successful log in

Superuser

- The superuser is a special privileged principal with UID 0 (usually associated with the username **root**).
- The superuser can become any other user.
- All security checks are turned off for the superuser, who can do almost everything with a few exceptions, e.g.
  - cannot write to a filesystem mounted as read-only (but can remount it as writable)
  - cannot read passwords (they are digested using a hash function)

Groups

- Users belong to one **or more** groups

```
adm:   x:   4:   syslog,user
```

1 2 3 4

- User account are stored in `/etc/group`
  1. **Group name**
  2. **Password**: lets the user change to a new group ID (`newgrp`)
  3. **Group ID**
  4. **List of users** (user's primary group is stored in `/etc/passwd`)
- Only superuser can add groups and members

Linux Subjects

- Subject are **processes**
- Each process is associated with
  - a **real** UID/GID (**ruid/rgid**): the real UID is inherited from the parent process. Typically it is the UID of the user that launched the process.
  - an **effective** UID/GID (**euid/egid**): the effective UID is inherited from the parent process or from the file being executed. This UID is used to grant access rights to a process.
  - a **saved** UID/GID: this allows a process to switch between the effective UID and real UID, vice versa.

Linux Objects

- The objects of access control include files, directories, memory devices, and I/O devices.
- In Linux almost all objects are <u>modeled as files</u> (organized in a tree-structured file system).
- Each object has
  - owner
  - group
  - 12 permission bits: rwx for owner, rwx for group,and rwx for others and suid, sgid, sticky
    - In this example, 000 for suid, sgid, sticky; 110 100 100 for users

```
-rw-r--r-- 1 root adm  9 mag  4 15:43 flag.txt
```

Permissions on directories

- Read (r): find which files are in the directory, e.g. for executing ls
- Write (w): add files and delete files
- Execute (x): make the directory the current directory (cd) and open files inside
- Nothing (-): permission unavailable for that group

Permissions on files (non-dir)

- Read: reading the content of a file
- Write: changing the content of a file
- Execute: loading the file in memory and execute

The suid, sgid, sticky bits

	suid	sgid	sticky bit
Non-executable files	-	-	not used anymore
Executable files	change euid when executing the file	change egid when executing the file	not used anymore
directories	-	new files inherit group of the directory	only a owner of a file can delete

Other issues on objects

- Only the owner can change the permission bits
- Only the superuser can change the owner

SUID to root

- When root owns an executable file and the SUID bit is set, the process will get superuser status during execution.
- Important SUID programs
  - `/bin/passwd` (change password)
  - `/bin/login` (user login program)
  - `/bin/su` (change UID program)
- SUID programs need to be written very carefully so that their privileges cannot be misused, and they only do what is intended.

Access control decisions

Access control uses the effective UID/GID.

- If the subject’s UID owns the file, it checks the owner permission bits
- If the subject’s UID does not own the file but its GID does, it checks the group permission bits
- If the subject’s UID and GID do not own the file, it checks the “others” permission bits (also called world)

Managing Linux Permissions

- Check permissions on current directory

```
user@host:~$ ls -l
drwxr--r-- 1 root adm   9 mag  4 15:43 documents
-rw-r--r-- 1 root root 9 mag  4 14:32 flag.txt
-r-xr-xr-x 1 root adm   9 mag  4 15:46 random
```

Managing Linux Permissions

- Check permissions on current directory

```
root@host:~$ ls -l
drwxr--r-- 1 root adm   9 mag   4 15:43 documents
-rw-r--r-- 1 root root  9 mag   4 14:32 flag.txt
-r-xr-xr-x 1 root adm   9 mag   4 15:46 random
```

- Changing ownership with chown

```
root@host:~$ chown user.group flag.txt
drwxr--r-- 1 root adm    9 mag 4 15:43 documents
-rw-r--r-- 1 user group  9 mag 4 14:32 flag.txt
-r-xr-xr-x 1 root adm    9 mag 4 15:46 random
```

Managing Linux Permissions

- Changing permissions with chmod
  - Remove execution from everyone

```
root@host:~$ chmod -x random
-r--r--r-- 1 root adm    9 mag 4 15:46 random
```

  - Add execution to group (g) and world (o), but NOT user (u)

```
root@host:~$ chmod go+x random
-r--r-xr-x 1 root adm    9 mag 4 15:46 random
```

Managing Linux Permissions

- Changing permissions with chmod
  - Setting explicit permissions (sum of bits, r=4, w=2, x=1)
    - Owner gets read and write (4 + 2 + 0) = 6
    - Group gets read (4 + 0 + 0) = 4
    - World gets execution (0 + 0 + 1) = 1

```
root@host:~$ chmod 641 random
-rw-r----x 1 root adm    9 mag 4 15:46 random
```

Enrico RUSSO

Andrea VALENZA

Università di Genova

UNIX file permission

https://cybersecnatlab.it