- [x] 1. Last time
- [x] 2. Protection and security in Unix
  - [x] Intro
  - [x] Setuid
  - [x] TOCTTOU
  - [x] Other thoughts

---

2. Protection and security in Unix

A. Intro

UIDs, GIDs (user id, group id)

A process has one user id and one or more group ids

Files and directories are access-controlled

- Saw this in lab 2 (ls)
- System stores w/ each file who owns it
- Where is the info stored?

Special user: uid 0, called root, treated by kernel as the administrator.

(...) permissions: can

UID 0 (root) has all permissions

read any file, do anything.

certain things only root can do:

- set clock
- halt machine
- mount filesystems
- change process's user or group id

B. setuid

ex: how do users change their password?

/etc/passwd
/etc/shadow

$ passwd

A prog. can be "setuid"

$ ls

real uid: mwalfish
effective uid: mwalfish

/sbin/passwd

$ passwd

real uid: mwalfish
effective uid: root

"su": change to new userid if correct passwd is typed.

Example attacks

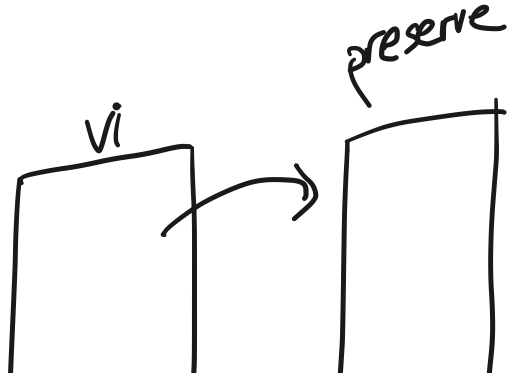
(a)

```
close(2);
exec("/usr/bin/passwd");
```

```
passwd:
    main()                          3
    {reality: 2   fd = open("/etc/passwd", ...)
                  .
                  .
                  .
                  fprintf(stderr, "Err msg\n");
                                    2
```

(b) old days: "preserve"

setuid root



attacker:
"
Create `bin, which does:

```
cp /bin/sh ./attack
chown root attack
chmod 4755 attack
```

attacker redefines IFS='/',
and runs vi in that environment.

preserve:

```
system("/bin/mail");
```

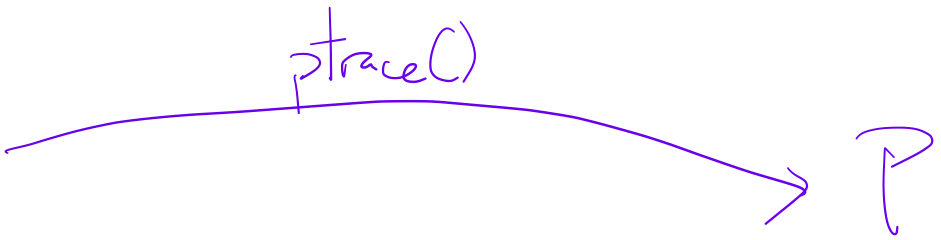
parsed as `bin mail'

---

ptrace()

setuid prog: P

Attacker



TOCTTOU

setuid program: P <logfile>

⋮

user input

fd = open(logfile,
O_CREAT | O_WRONLY |
O_TRUNC,
0666)

access(): check whether the real id, not effective id, is allowed to access the file.

```
if (access (logfile, W_OK) < 0)
    return ERROR;
fd = open (logfile, ... )   /* as above */
```

attacker runs "P /tmp/X"

P

attacker

creat("/tmp/X");

checks access("/tmp/X") → OK

unlink("/tmp/X");

symlink("/etc/passwd", "/tmp/X");

open("/tmp/X", O_TRUNC | O_WRONLY);