Multi-User Environments

Hands-On UNIX System Administration DeCal

Lecture 3 — 6 February 2012

Administrivia

- Wait-list woes.
- INST class accounts.
- Homework submissions.

Shell odds-and-ends

- Some shell shorthand:
  ~ (tilde) expands to your homedir
  ~user expands to **user**'s homedir
  . (period) expands to your current dir
- When you type a command and hit Enter, your shell looks in your **PATH** (a variable) for folders which contain binaries. E.g.:

  /usr/bin:/usr/sbin:/bin:/sbin:/usr/local/bin

Shell odds-and-ends

- To execute a program that can't be found in your PATH variable, you need to specify its full path. From the previous slide, **the following are equivalent** on the OCF:
  - `/home/j/jo/jordan/bin/dtach`
  - `~jordan/bin/dtach`
  - `./bin/dtach` (if your CWD is `~jordan`)

`ls` odds-and-ends

- `ls -l`: long listing form. (More shortly.)

```
-rwxr-xr-x 1 jordan ocf 35567 2009-02-08 17:04 dtach
```

- `ls -lh`: show human-readable filesizes.

```
-rwxr-xr-x 1 jordan ocf 35K 2009-02-08 17:04 dtach
```

- `ls -ld`: show directories, not contents.

```
drwxr-xr-x 22 root root 4096 2010-12-06 21:21 /
```

- `ls -F`: "classify" files with indicators.

```
executable*     file                folder/
symlink@
```

`ls -l`: long listing

- `ls` long listing form includes…

  `-rwxr-xr-x 1 jordan ocf 35567 2009-02-08 17:04 dtach`

  - **File type.** (File, directory, symlink, FIFO, character/block special, or socket.)
  - **Permissions** — user, group, others.
  - Number of **hard links.**
  - **Owner**, **group**, **size**, **date**, and **name.**

UNIX permissions

- There are three basic permissions: **read**, **write**, and **execute.** (cd'ing to a directory requires the execute permission.)
- (passwd: usernames. shadow: passwords.)

```
-rw-r--r-- 1 root root   1341 2011-01-17 02:32 /etc/passwd
-rw-r----- 1 root shadow  954 2011-01-17 02:32 /etc/shadow
```

- passwd needs to be world-readable, shadow secret, and both root-modifiable.

UNIX permissions

- Change ownership with **chown/chgrp.**

  ```
  chown jordan:root file
  chown jordan file; chgrp root file
  ```

- Set permissions with **chmod.**

  If you have a file with the permissions "rw-------", these commands are equivalent:

  ```
  chmod u=rwx,g+w,o-rwx file
  chmod 720 file
  ```

UNIX permissions

- Octal notation is a more compact way to handle file permissions. Interpret "rwx" as a binary number — "rwx-w----" == 720.

user			group			other		
r	w	x	-	w	-	-	-	-
1	1	1	0	1	0	0	0	0
4+	2+	1	0+	2+	0	0+	0+	0
7			2			0		

UNIX permissions

- Your **umask** governs what permissions new files you create have. Common umasks include:
  - 022 — mask out write permissions for group/other, leaving your permissions unaffected.
  - 077 — mask out *all* permissions for group/other; this is the default on INST.

UNIX permissions

- **ACLs** allow for fine-grained permissions control. (This is filesystem-dependent.)
- **Setgid/setuid bits.** A 'setuid root' binary runs with root privileges — this is how programs like sudo and ping can work.
- **Sticky bit.** In a world-writable folder with the sticky bit set, only you can rename or delete your files. Used for /tmp, /var/mail.

Users and Groups

- On Linux, user accounts live in two files:
  /etc/passwd — logins, UIDs, real names
  /etc/shadow — encrypted passwords
- On FreeBSD, logins and passwords both live in /etc/master.passwd (from which a world-readable /etc/passwd is generated).
- Groups are recorded in /etc/group.

Users and Groups

- By the way, manpages aren't just limited to program help! Try "`man 5 passwd`".
- You can look up entries in the passwd and group files with **getent.** (shadow or master.passwd data is visible only to root.)
- E.g., `% getent group dead_parrots`
  `dead_parrots:*:12345:psb,gwh,cgd`

Name Service Switch

- You don't *have* to store passwd, shadow, and group databases locally. (The same goes for other databases, like /etc/hosts and mail aliases.)
- **NSS**, the **Name Service Switch**, can be configured to look up usernames and passwords in network servers.

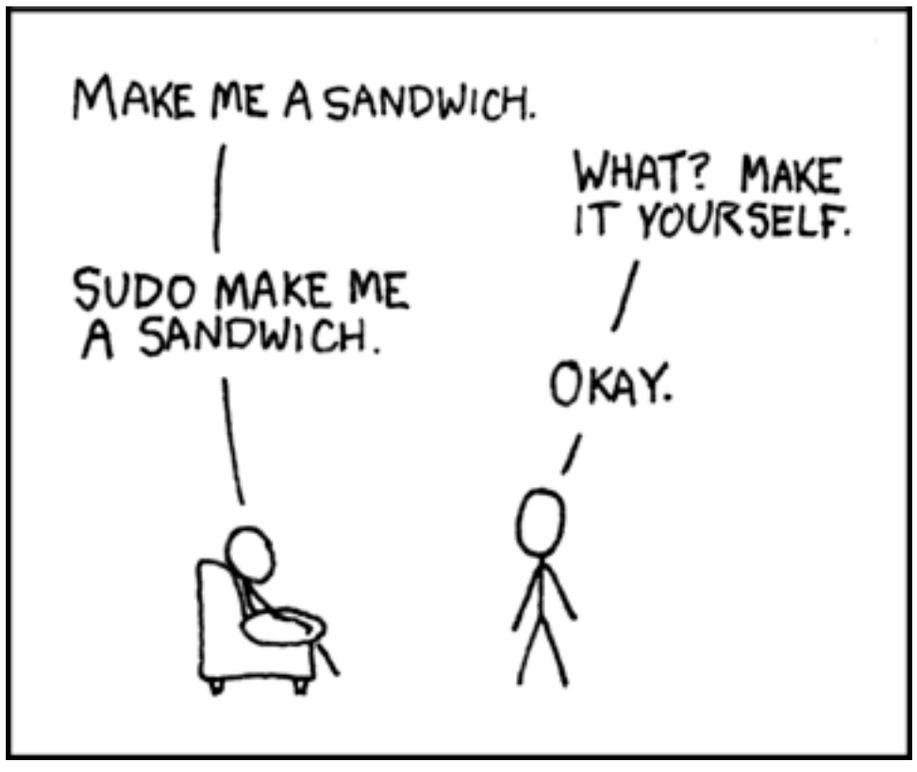
Authentication

- Authentication is handled by the **PAM stack** — PAM stands for **Pluggable Authentication Modules.**
- PAM can work with any conceivable authentication method: fingerprint scanners, voice recognition, smartcards…
- PAM also works with network servers. (Requires configuring both PAM *and* NSS.)

LDAP/Kerberos

- **LDAP**, the **Lightweight Directory Access Protocol**, is a network database system. You can put anything in it — the OCF uses it for user accounts, groups, automatic NFS mount configuration, server IPs…
- You can also store passwords in LDAP, but **Kerberos** offers a really cool centralized auth system. (CalNet uses Kerberos.)

su and sudo



source: http://xkcd.com/149/

su and sudo

- su: “substitute user”/”spoof user.”
  `/bin/su -`: become root (need hyphen!).
- sudo: “substitute user do.”

```
% sudo make sandwich
Password:
make: don't know how to make
sandwich. Stop
```