Users and Groups

From ArchWiki

Users and groups are used on GNU/Linux for access control — that is, to control access to the system's files, directories, and peripherals. Linux offers relatively simple/coarse access control mechanisms by default. For more advanced options, see ACL and LDAP Authentication.

Summary

This article describes how to manage user and group accounts. Whilst certain desktop environments provide graphical user/group management tools, they are not discussed here.

Overview

Users and groups are used on GNU/Linux for access control. The superuser (root) has complete access to the operating system and its configuration; it is intended for administrative use only. Unprivileged users can use the su and sudo programs for controlled privilege escalation.

Related

DeveloperWiki:UID / GID Database

PolicyKit

chmod

Change username

Contents

Overview

A *user* is anyone who uses a computer. In this case, we are describing the names which represent those users. It may be Mary or Bill, and they may use the names Dragonlady or Pirate in place of their real name. All that matters is that the computer has a name for each account it creates, and it is this name by which a person gains access to use the computer. Some system services also run using restricted or privileged user accounts.

Managing users is done for the purpose of security by limiting access in certain specific ways.

Any individual may have more than one account, as long as they use a different name for each account they create. Further, there are some reserved names which may not be used such as "root".

Users may be grouped together into a "group," and users may choose to join an existing group to utilize the privileged access it grants.

Note: The beginner should use these tools carefully and stay away from having anything to do with any other *existing* user account, other than their own.

Permissions and ownership

From In UNIX Everything is a File (http://ph7spot.com/musings/in-unix-everything-is-a-file) :

> *The UNIX operating system crystallizes a couple of unifying ideas and concepts that shaped its design, user interface, culture and evolution. One of the most important of these is probably the mantra: "everything is a file," widely regarded as one of the defining points of UNIX.*
>
> *This key design principle consists of providing a unified paradigm for accessing a wide range of input/output resources: documents, directories, hard-drives, CD-ROMs, modems, keyboards, printers, monitors, terminals and even some inter-process and network communications. The trick is to provide a common abstraction for all of these resources, each of which the UNIX fathers called a "file." Since every "file" is exposed through the same API, you can use the same set of basic commands to read/write to a disk, keyboard, document or network device.*

From Extending UNIX File Abstraction for General-Purpose Networking (http://www.intel-research.net/Publications/Pittsburgh/101220041324_277.pdf) :

> *A fundamental and very powerful, consistent abstraction provided in UNIX and compatible operating systems is the file abstraction. Many OS services and device interfaces are implemented to provide a file or file system metaphor to applications. This enables new uses for, and greatly increases the power of, existing applications — simple tools designed with specific uses in mind can, with UNIX file abstractions, be used in novel ways. A simple tool, such as cat, designed to read one or more files and output the contents to standard output, can be used to read from I/O devices through special device files, typically found under the* `/dev` *directory. On many systems, audio recording and playback can be done simply with the commands,* "`cat /dev/audio > myfile`" *and* "`cat myfile > /dev/audio`," *respectively.*

Every file on a GNU/Linux system is owned by a user and a group. In addition, there are three types of access permissions: read, write, and execute. Different access permissions can be applied to a file's owning user, owning group, and others (those without ownership). One can determine a file's owners and permissions by viewing the long listing format of the `ls` command:

```
$ ls /boot/ -l
```

```
total 13740
drwxr-xr-x 2 root root    4096 Jan 12 00:33 grub
-rw-r--r-- 1 root root 8570335 Jan 12 00:33 kernel26-fallback.img
-rw-r--r-- 1 root root 1821573 Jan 12 00:31 kernel26.img
-rw-r--r-- 1 root root 1457315 Jan  8 08:19 System.map26
-rw-r--r-- 1 root root 2209920 Jan  8 08:19 vmlinuz26
```

The first column displays the file's permissions (for example, the file `kernel26.img` has permissions `-rw-r--r--`). The third and fourth columns display the file's owning user and group, respectively. In this example, all files are owned by the *root* user and the *root* group.

```
$ ls /media/ -l
```

```
total 16
drwxrwx--- 1 root vboxsf 16384 Jan 29 11:02 sf_Shared
```

In this example, the `sf_Shared` directory is owned by the *root* user and the *vboxsf* group. It is also possible to determine a file's owners and permissions using the stat command:

Owning user:

```
$ stat -c %U /media/sf_Shared/
```

```
root
```

Owning group:

```
$ stat -c %G /media/sf_Shared/
```

```
vboxsf
```

Access rights:

```
$ stat -c %A /media/sf_Shared/
```

```
drwxrwx---
```

Access permissions are displayed in three groups of characters, representing the permissions of the owning user, owning group, and others, respectively. For example, the characters `-rw-r--r--` indicate that the file's owner has read and write permission, but not execute (`rw-`), whilst users belonging to the owning group and other users have only read permission (`r--` and `r--`). Meanwhile, the characters `drwxrwx---` indicate that the file's owner and users belonging to the owning group all have read, write, and execute permissions (`rwx` and `rwx`), whilst other users are denied access (`---`). The first character represents the file's type.

List files owned by a user or group with the `find` command:

```
# find / -group [group]
```

```
# find / -user [user]
```

A file's owning user and group can be changed with the `chown` (change owner) command. A file's access permissions can be changed with the `chmod` (change mode) command.

See man chown (http://linux.die.net/man/1/chown) , man chmod (http://linux.die.net/man/1/chmod) , and Linux file permissions (http://www.tuxfiles.org/linuxhelp/filepermissions.html) for additional detail.

File list

Warning: Do not edit these files by hand. There are utilities that properly handle locking and avoid invalidating the format of the database. See #User management and #Group management for an overview.

File	Purpose
`/etc/shadow`	Secure user account information
`/etc/passwd`	User account information
`/etc/gshadow`	Contains the shadowed information for group accounts
`/etc/group`	Defines the groups to which users belong
`/etc/sudoers`	List of who can run what by sudo
`/home/*`	Home directories

User management

To list users currently logged on the system, the `who` command can be used.

To add a new user, use the `useradd` command:

```
# useradd -m -g [initial_group] -G [additional_groups] -s [login_shell] [username]
```

- `-m` creates the user home directory as `/home/[username]` ; within their home directory, a non-root user can write files, delete them, install programs, and so on.
- `-g` defines the group name or number of the user's initial login group; the group name must exist; if a group number is provided, it must refer to an already existing group; if not specified, the behavior of useradd will depend on the USERGROUPS_ENAB variable contained in `/etc/login.defs` .
- `-G` introduces a list of supplementary groups which the user is also a member of; each group is separated from the next by a comma, with no intervening spaces; the default is for the user to belong only to the initial group.
- `-s` defines the path and filename of the user's default login shell; Arch Linux init scripts use Bash; after the boot process is complete, the default login shell is the one specified here; ensure the chosen shell package is installed if choosing something other than Bash.

A typical desktop system example, adding a user named *archie* specifying bash as the login shell:

```
# useradd -m -g users -s /bin/bash archie
```

To enter user information for the *GECOS* field (e.g. the full user name), type:

```
# chfn [username]
```

(this way `chfn` runs in interactive mode).

To specify the user's password, type:

```
# passwd [username]
```

User accounts may be deleted with the `userdel` command.

```
# userdel -r [username]
```

The `-r` option specifies that the user's home directory and mail spool should also be deleted.

User database

Local user information is stored in the `/etc/passwd` file. To list all user accounts on the system:

```
$ cat /etc/passwd
```

There is one line per account, and each is of the format:

```
account:password:UID:GID:GECOS:directory:shell
```

where:

- `account` is the user name
- `password` is the user password
- `UID` is the numerical user ID
- `GID` is the numerical primary group ID for the user
- `GECOS` is an optional field used for informational purposes; usually it contains the full user name
- `directory` is the user's `$HOME` directory
- `shell` is the user command interpreter (defaults to `/bin/sh`)

Note: Arch Linux uses *shadowed* passwords. The `passwd` file is world-readable, so storing passwords (hashed or otherwise) in this file would be insecure. Instead, the `password` field will contain a placeholder character (`x`) indicating that the hashed password is saved in the access-restricted file `/etc/shadow`.

Group management

`/etc/group` is the file that defines the groups on the system (`man group` for details).

Display group membership with the `groups` command:

```
$ groups [user]
```

If `user` is omitted, the current user's group names are displayed.

The `id` command provides additional detail, such as the user's UID and associated GIDs:

```
$ id [user]
```

To list all groups on the system:

```
$ cat /etc/group
```

Create new groups with the `groupadd` command:

```
# groupadd [group]
```

Add users to a group with the `gpasswd` command:

```
# gpasswd -a [user] [group]
```

To delete existing groups:

```
# groupdel [group]
```

To remove users from a group:

```
# gpasswd -d [user] [group]
```

If the user is currently logged in, he/she must log out and in again for the change to have effect.

Group list

User groups

Note:

- Some of these may not be needed when running a system with systemd. See Supplementary information section in Systemd.
- None of these groups is needed for standard desktop permissions like sound, 3D, printing, mounting, etc. as long as the *logind* session isn't broken (for example by starting X on a different VT than where you logged in).

Workstation/desktop users often add their non-root user to some of following groups to allow access to peripherals and other hardware and facilitate system administration:

Group	Affected files	Purpose
audio	`/dev/audio`, `/dev/snd/*`, `/dev/rtc0`	Direct access to sound hardware, for all sessions (requirement is imposed by both ALSA and OSS). Local sessions already have the ability to play sound and access mixer controls.
camera		Access to Digital Cameras.
disk	`/dev/sda[1-9]`, `/dev/sdb[1-9]`	Access to block devices not affected by other groups such as *optical*, *floppy*, and *storage*.
floppy	`/dev/fd[0-9]`	Access to floppy drives.
games	`/var/games`	Access to some game software.
locate	`/usr/bin/locate`, `/var/lib/locate`, `/var/lib/mlocate`, `/var/lib/slocate`	Right to use updatedb command.
lp	`/etc/cups`, `/var/log/cups`, `/var/cache/cups`, `/var/spool/cups`	Access to printer hardware; enables the user to manage print jobs.
network		Right to change network settings such as when using NetworkManager.
networkmanager		Requirement for your user to connect wirelessly with NetworkManager. This group is not included with Arch by default so it must be added manually.
optical	`/dev/sr[0-9]`, `/dev/sg[0-9]`	Access to optical devices such as CD and DVD drives.
power		Right to use Pm-utils (suspend, hibernate...) and power management controls.
scanner	`/var/lock/sane`	Access to scanner hardware.
storage		Access to removable drives such as USB hard drives, flash/jump drives, MP3 players; enables the user to mount storage devices.
sys		Right to admin printers in CUPS.
users		Standard users group.
uucp	`/dev/ttyS[0-9]`, `/dev/tts/[0-9]`	Serial and USB devices such as modems, handhelds, RS-232/serial ports.
video	`/dev/fb/0`, `/dev/misc/agpgart`	Access to video capture devices, 2D/3D hardware acceleration, framebuffer (X can be used *without* belonging to this group). Local sessions already have the ability to use hardware acceleration and video capture.
wheel		Administration group, commonly used to give access to the sudo and su commands (neither uses it by default). Will be used in the future by systemd to allow starting/stopping services as non-root.[1] (http://cgit.freedesktop.org/systemd/systemd/tree/TODO#n79)

System groups

The following groups are used for system purposes and are not likely to be used by novice Arch users:

Group	Affected files	Purpose
avahi		
bin	`/usr/bin/*`	Read-only access to the binary files in `/usr/bin/`
clamav	`/var/lib/clamav/*`, `/var/log/clamav/*`	Used by Clam AntiVirus.
daemon		
dbus	`/var/run/dbus/*`	
ftp	`/srv/ftp`	used by FTP servers like Proftpd
gdm	X server authorization directory (ServAuthDir)	GDM group.
hal	`/var/run/hald`, `/var/cache/hald`	
http		

kmem	`/dev/port`, `/dev/mem`, `/dev/kmem`	
log	`/var/log/*`	Access to log files in `/var/log`.
mail	`/usr/bin/mail`	
mem		
mpd	`/var/lib/mpd/*`, `/var/log/mpd/*`, `/var/run/mpd/*`, optionally music directories	MPD group.
nobody		Unprivileged group.
ntp	`/var/lib/ntp/*`	NTPd group.
policykit		PolicyKit group.
root	`/*`	Complete system administration and control (root, admin).
smmsp		sendmail group.
systemd-journal	`/var/log/journal/*`	Provides access to the complete systemd logs. Otherwise, only user generated messages are displayed.
tty	`/dev/tty`, `/dev/vcc`, `/dev/vc`, `/dev/ptmx`	Eg. to acces `/dev/ACMx`
vboxsf	virtual machines' shared folders	Used by VirtualBox.
fuse		Used by fuse to allow user mounts.

Software groups

These groups allow its members to use specific software:

Group	Affected files	Purpose
adbusers	devices nodes under `/dev/`	Right to access Android Debugging Bridge.
cdemu	`/dev/vhba_ctl`	Right to use cdemu drive emulation.
kvm	`/dev/kvm`	Benefit from KVM's-Hardware-assisted virtualization speed if your Processor features either Intel's VT-x or AMD's AMD-V (http://www.linux-kvm.org/page/FAQ#What_do_I_need_to_use_KVM.3F) extension.
thinkpad	`/dev/misc/nvram`	Used by ThinkPad users for access to tools such as tpb.
vboxusers	`/dev/vboxdrv`	Right to use VirtualBox software.
vmware		Right to use VMware software.
ssh		Sshd can be configured to only allow members of this group to login.
wireshark		Right to capture packets with Wireshark.

Deprecated or unused groups

Following groups are currently of no use for anyone:

Group	Purpose
rfkill	**Unused!** Right to control wireless devices power state (probably should be used by `rfkill (https://www.archlinux.org/packages/?name=rfkill)`).
stb-admin	**Unused!** Right to access system-tools-backends (http://system-tools-backends.freedesktop.org/)

Retrieved from "https://wiki.archlinux.org/index.php?title=Users_and_Groups&oldid=263336"
Category: Security