

RHSA (Admin 1) Red Hat Enterprise Linux 8 Essentials

Prepared by Eng. Taghreed Elsayed

Lecture 4

جامعة الجميع الذكية

تسعى لتكون الكيان الأول الذي يتبنى مبدأ المجتمع التعليمي
التعاوني ويقدم العلم مقابل العمل لتنمية وبناء المجتمعات.

MANAGING LOCAL LINUX USERS AND GROUPS



جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

❖ WHAT IS A USER?

- ❑ Every process (running program) on the system runs as a particular user. Every file is owned by a particular user. Access to files and directories are restricted by user.
- ❑ The user associated with a running process determines the files and directories accessible to that process.
- ❑ The id command is used to show information about the current logged-in user.
- ❑ Basic information about another user can also be requested by passing in the username of that user as the first argument to the **id command**.

```
[student@desktopX ~]$ id
uid=1000(student) gid=1000(student) groups=1000(student),10(wheel)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```



- ❑ To view the user associated with a file or directory, use the `ls -l` command. The third column shows the username:

```
[student@serverX ~]$ ls -l /tmp
drwx----- . 2 gdm      gdm      4096 Jan 24 13:05 orbit-gdm
drwx----- . 2 student student 4096 Jan 25 20:40 orbit-student
-rw-r--r-- . 1 root     root     23574 Jan 24 13:05 postconf
```

- ❑ To view process information, use the **ps command**. The default is to show only processes in the current shell.
- ❑ Add the `a` option to view all processes with a terminal. To view the user associated with a process, include the `u` option. **The first column shows the username:**



```
[student@serverX ~]$ ps au
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	428	0.0	0.7	152768	14400	tty1	Ss+	Feb03	0:04	/usr/bin/Xorg
root	511	0.0	0.0	110012	812	ttyS0	Ss+	Feb03	0:00	/sbin/agetty
root	1805	0.0	0.1	116040	2580	pts/0	Ss	Feb03	0:00	-bash
root	2109	0.0	0.1	178468	2200	pts/0	S	Feb03	0:00	su - student
student	2110	0.0	0.1	116168	2864	pts/0	S	Feb03	0:00	-bash
student	3690	0.0	0.0	123368	1300	pts/0	R+	11:42	0:00	ps au

- ❑ The output of the previous commands displays users by name, but internally the operating system tracks users by a **UID number**.
- ❑ The mapping of names to numbers is defined in databases of account information. By default, systems use a simple **"flat file," the /etc/passwd file**, to store information about local users.
- ❑ The format of **/etc/passwd** follows (seven colon-separated fields):

① username: ② password: ③ UID: ④ GID: ⑤ GECOS: ⑥ /home/dir: ⑦ shell

- ① *username* is a mapping of a UID to a name for the benefit of human users.
- ② *password* is where, historically, passwords were kept in an encrypted format. Today, they are stored in a separate file called **/etc/shadow**.
- ③ *UID* is a user ID, a number that identifies the user at the most fundamental level.
- ④ *GID* is the user's primary group ID number. Groups will be discussed in a moment.
- ⑤ *GECOS* field is arbitrary text, which usually includes the user's real name.
- ⑥ */home/dir* is the location of the user's personal data and configuration files.
- ⑦ *shell* is a program that runs as the user logs in. For a regular user, this is normally the program that provides the user's command line prompt.



WHAT IS A GROUP?

- ❑ Like users, groups have a name and a number (GID). Local groups are defined in `/etc/group`.

➤ Primary groups

- ❑ • Every user has exactly one primary group.
- ❑ • For local users, the primary group is defined by the GID number of the group listed in the fourth field of `/etc/passwd`.
- ❑ • Normally, the primary group owns new files created by the user.
- ❑ • Normally, the primary group of a newly created user is a newly created group with the same name as the user. The user is the only member of this User Private Group (UPG).



Supplementary groups

- Users may be a member of zero or more supplementary groups.
- The users that are supplementary members of local groups are listed in the last field of the group's entry in /etc/group.
 - For local groups, user membership is determined by a comma separated list of users found in the last field of the group's entry in /etc/group:

```
groupname:password:GID:list, of, users, in, this, group
```



USER AND GROUP CONCEPTS

DESCRIPTION	KEYWORD
A number that identifies the user at the most fundamental level	UID
The program that provides the user's command line prompt	login shell
Location of local group information	/etc/group
Location of the user's personal files	home directory
A number that identifies the group at the most fundamental level	GID
Location of local user account information	/etc/passwd
The fourth field of /etc/passwd	primary group

1-200 → UID number range for "system users" assigned statically to system processes by red hat

201 → UID number range for "system users" used by system processes that do not own files on the file system.

❑ MANAGING LOCAL USERS

- ❑ A number of command-line tools can be used to manage local user accounts.
- ❑ useradd creates users
- ❑ • **useradd** username sets reasonable defaults for all fields in **/etc/passwd** when run without options.
- ❑ The **useradd command** does not set any valid password by default, and the user cannot log in until a password is set.
- ❑ • **useradd --help** will display the basic options that can be used to override the defaults. In most cases, the same options can be used with the **usermod command to modify an existing user.**

- ❑ • Some defaults, such as the range of valid UID numbers and default password aging rules, are read from the **/etc/login.defs file**. Values in this file are only used when creating new users.
- ❑ • A change to this file will not have an effect on any existing users. **usermod modifies** existing users
- ❑ • **usermod --help** will display the basic options that can be used to modify an account. Some common options include:

USERMOD OPTIONS:	
-c, --comment COMMENT	Add a value, such as a full name, to the GECOS field.
-g, --gid GROUP	Specify the primary group for the user account.
-G, --groups GROUPS	Specify a list of supplementary groups for the user account.
-a, --append	Used with the -G option to append the user to the supplemental groups mentioned without removing the user from other groups.
-d, --home HOME_DIR	Specify a new home directory for the user account.
-m, --move-home	Move a user home directory to a new location. Must be used with the -d option.
-s, --shell SHELL	Specify a new login shell for the user account.
-L, --lock	Lock a user account.
-U, --unlock	Unlock a user account.

- ❑ **userdel** deletes users
 - ❑ • **userdel username** → removes the user from `/etc/passwd`, but leaves the home directory intact by default.
 - ❑ • **userdel -r** → username removes the user and the user's home directory.
- ❑ **In Red Hat Enterprise Linux 7 :**
 - ❑ **the useradd command** assigns new users the first free UID number available in the range starting from UID 1000 or above. (unless one is explicitly specified with the `-u` UID option).
 - ❑ This is how information leakage can occur: If the first free UID number had been previously assigned to a user account which has since been removed from the system, the old user's UID number will get reassigned to the new user, giving the new user ownership of the old user's remaining files.



❑ The following scenario demonstrates this situation:

```
[root@serverX ~]# useradd prince
[root@serverX ~]# ls -l /home
drwx----- . 3 prince prince 74 Feb 4 15:22 prince
[root@serverX ~]# userdel prince
[root@serverX ~]# ls -l /home
drwx----- . 3 1000 1000 74 Feb 4 15:22 prince
[root@serverX ~]# useradd bob
[root@serverX ~]# ls -l /home
drwx----- . 3 bob bob 74 Feb 4 15:23 bob
drwx----- . 3 bob bob 74 Feb 4 15:22 prince
```



passwd sets passwords

- • `passwd username` can be used to either set the user's initial password or change that user's password.
- • **The root user** can set a password to any value.
- A message will be displayed if the password does not meet the minimum recommended criteria, but is followed by a prompt to retype the new password and all tokens are updated successfully.

```
[root@serverX ~]# passwd student
Changing password for user student.
New password: redhat123
BAD PASSWORD: The password fails the dictionary check - it is based on a
dictionary word
Retype new password: redhat123
passwd: all authentication tokens updated successfully.
```



- ❑ A regular user must choose a password which is at least 8 characters in length and is not based on a dictionary word, the username, or the previous password.

UID ranges Specific UID numbers and ranges of numbers are used for specific purposes by Red Hat Enterprise Linux.

- UID 0 is always assigned to the super user account, root.
- UID 1-200 is a range of "system users" assigned statically to system processes by Red Hat.
- UID 201-999 is a range of "system users" used by system processes that do not own files on the file system.

- ❑ They are typically assigned dynamically from the available pool when the software that needs them is installed



- ❑ Programs run as these "unprivileged" system users in order to limit their access to just the resources they need to function.
- UID 1000+ is the range available for assignment to regular users.



NOTE

Prior to Red Hat Enterprise Linux 7, the convention was that UID 1-499 was used for system users and UID 500+ for regular users. Default ranges used by **useradd** and **groupadd** can be changed in the **/etc/login.defs** file.

CREATING USERS USING COMMANDLINE TOOLS

1. will create a number of users on your server X system, setting and recording an initial password for each user.

2. **OUTCOMES** A system with additional user accounts.

Reset your server X system.

1. Log into the GNOME desktop on serverX as student with a password of student.

2. Open a window with a Bash prompt.
Select Applications → Utilities → Terminal.

3. Become the root user at the shell prompt.



```
[student@serverX ~]$  
su -  
Password:  
redhat
```

4. Add the user *juliet*.

```
[root@serverX ~]#  
useradd juliet
```

5. Confirm that *juliet* has been added by examining the **/etc/passwd** file.

```
[root@serverX ~]# tail -2 /etc/passwd  
tcpdump:x:72:72:::/sbin/nologin  
juliet:x:1001:1001::/home/juliet:/bin/bash
```

6. Use the **passwd** command to initialize *juliet*'s password.

```
[root@serverX ~]# passwd juliet
Changing password for user juliet.
New password: juliet
```

```
BAD PASSWORD: The password is shorter than 8 characters
Retype new password: juliet
passwd: all authentication tokens updated successfully.
```

7. Continue adding the remaining users in the steps below and set initial passwords.

7.1. romeo

```
[root@serverX ~]# useradd romeo [root@serverX ~]# passwd romeo
Changing password for user romeo.
New password: romeo
BAD PASSWORD: The password is shorter than 8 characters
Retype new password: romeo
passwd: all authentication tokens updated successfully.
```

7.2. hamlet

```
[root@serverX ~]#  
useradd hamlet  
[root@serverX ~]#  
passwd hamlet
```

7.3. reba

```
[root@serverX ~]#  
useradd reba  
[root@serverX ~]#  
passwd reba
```

7.4. dolly

```
[root@serverX ~]#  
useradd dolly  
[root@serverX ~]#  
passwd dolly
```

7.5. elvis

```
[root@serverX ~]#  
useradd elvis  
[root@serverX ~]#  
passwd elvis
```



MANAGING LOCAL GROUP ACCOUNTS

MANAGING SUPPLEMENTARY GROUPS

- ❑ A group must exist before a user can be added to that group. Several command-line tools are used to manage local group accounts.
- ❑ `groupadd` creates groups
- ❑ • `groupadd groupname` → without options uses the next available GID from the range specified in the `/etc/login.defs` file.
- ❑ • The `-g` GID option is used to specify a specific GID.

```
[student@serverX ~]$ sudo groupadd -g 5000 ateam
```



- ❑ The **-r option** will create a system group using a GID from the range of valid system GID numbers listed in the `/etc/login.defs` file.

```
[student@serverX ~]$ sudo groupadd -r appusers
```

- **groupmod** modifies existing groups
- • **The groupmod command** is used to change a group name to a GID mapping. The **-n option** is used to specify a new name.

```
[student@serverX ~]$ sudo groupmod -n javaapp appusers
```



- The **-g** option is used to specify a new GID.

```
[student@serverX ~]$ sudo groupmod -g 6000 ateam
```

groupdel deletes a group

- The **groupdel** command will remove a group.

```
[student@serverX ~]$ sudo groupdel javaapp
```

- ❑ A group may not be removed if it is the primary group of any existing user. As with **userdel**, check all file systems to ensure that no files remain owned by the group.



usermod alters group membership

- The membership of a group is controlled with user management. Change a user's primary group with **usermod -g *groupname***.

```
[student@serverX ~]$ sudo usermod -g student student
```

- Add a user to a supplementary group with **usermod -aG *groupname username***.

```
[student@serverX ~]$ sudo usermod -aG wheel elvis
```

MANAGING GROUPS USING COMMANDLINE TOOLS

- you will add users to newly created supplementary groups.
- OUTCOMES The shake speare group consists of juliet, romeo, and hamlet. The artists group contains reba, dolly, and elvis.
- **Perform the following steps on serverX unless directed otherwise.**

1. Become the **root** user at the shell prompt.

```
[student@serverX ~]$  
su -  
Password:  
redhat
```

2. Create a supplementary group called **shakespeare** with a group ID of **30000**.

```
[root@serverX ~]#  
groupadd -g 30000 shakespeare
```

3. Create a supplementary group called **artists**.

```
[root@serverX ~]#  
groupadd artists
```

4. Confirm that *shakespeare* and *artists* have been added by examining the **/etc/group** file.

```
[root@serverX ~]# tail -5 /etc/group  
reba:x:1004:  
dolly:x:1005:  
elvis:x:1006:  
shakespeare:x:30000:  
artists:x:30001:
```

5. Add the *juliet* user to the *shakespeare* group as a supplementary group.

```
[root@serverX ~]#  
usermod -G shakespeare juliet
```

6. Confirm that *juliet* has been added using the **id** command.

```
[root@serverX ~]# id juliet  
uid=1001(juliet) gid=1001(juliet) groups=1001(juliet),30000(shakespeare)
```



7. Continue adding the remaining users to groups as follows:

7.1. Add *romeo* and *hamlet* to the *shakespeare* group.

```
[root@serverX ~]#  
usermod -G shakespeare romeo  
[root@serverX ~]#  
usermod -G shakespeare hamlet
```

7.2. Add *reba*, *dolly*, and *elvis* to the *artists* group.

```
[root@serverX ~]#  
usermod -G artists reba  
[root@serverX ~]#  
usermod -G artists dolly  
[root@serverX ~]#  
usermod -G artists elvis
```



7.3. Verify the supplemental group memberships by examining the **/etc/group** file.

```
[root@serverX ~]# tail -5 /etc/group
reba:x:1004:
dolly:x:1005:
elvis:x:1006:
shakespeare:x:30000:juliet,romeo,hamlet
artists:x:30001:reba,dolly,elvis
```



SUPERUSER ACCESS

- ❑ THE root USER Most operating systems have some sort of superuser, a user that has all power over the system.
- ❑ This user in Red Hat Enterprise Linux is the root user. This user has the power to override normal privileges on the file system, and is used to manage and administer the system.
- ❑ In order to perform tasks such as installing or removing software and to manage system files and directories, a user must escalate privileges to the root user. Most devices can only be controlled by root, but there are a few exceptions.
- ❑ For instance, removable devices, such as USB devices, are allowed to be controlled by a normal user. Thus, a non-root user is allowed to add and remove files and otherwise manage a removable device, but only root is allowed to manage "fixed" hard drives by default.

- ❑ Root has unlimited power to damage the system: remove files and directories, remove user accounts, add backdoors, etc.
- ❑ If the root account is compromised, someone else would have administrative control of the system.
- ❑ Throughout this course, administrators will be encouraged to log in as a normal user and escalate privileges to root only when needed.
- ❑ The root account on Linux is roughly equivalent to the local Administrator account on Windows. In Linux, most system administrators log into an unprivileged user account and use various tools to temporarily gain root privileges.

- One common practice on Windows in the past is for the local Administrator user to log in directly to perform system administrator duties.
- However, on Linux, it is recommended that system administrators should not log in directly as root. Instead, system administrators should log in as a non-root user, and use other mechanisms (**su, sudo, or PolicyKit, for example**) to temporarily gain **superuser privileges**.
- In recent versions of Microsoft Windows, Administrator disabled by default, and features such as User Account Control (UAC) are used to limit administrative privileges for users until actually needed. In Linux, the PolicyKit system is the nearest equivalent to UAC.

SWITCHING USERS WITH su

- ❑ The **su command** allows a user to switch to a different user account. If a username is not specified, the root account is implied. When invoked as a regular user, a prompt will display asking for the password of the account you are switching to; when invoked as root, there is no need to enter the account password.

```
su [-] <username>
```

```
[student@desktopX ~]$  
su -  
Password:  
redhat  
[root@desktopX ~]#
```

- ❖ The command `su username` starts a non-login shell, while the command `su - username` starts a login shell. The main distinction **is `su -`** sets up the shell environment as if this were a clean login as that user, while **`su`** just starts a shell as that user with the current environment settings.
- ❖ **In most cases**, administrators want to run **`su -`** to get the user's normal settings. For more information, see the `bash(1)` man page.



NOTE

The **`su`** command is most frequently used to get a command line interface (shell prompt) which is running as another user, typically **`root`**. However, with the **`-c`** option, it can be used like the Windows utility **`runas`** to run an arbitrary program as another user. See **info `su`** for details.

RUNNING COMMANDS AS root WITH sudo

- ❑ Fundamentally, Linux implements a very coarse-grained permissions model: root can do everything, other users can do nothing (systems-related).
- ❑ The common solution previously discussed is to allow standard users to temporarily “become root” using the su command.
- ❑ The disadvantage is that while acting as root, all the privileges (and responsibilities) of root are granted. Not only can the user restart the web server, but they can also remove the entire /etc directory.
- ❑ Additionally, all users requiring superuser privilege in this manner must know the root password.

- ❑ The **sudo command** allows a user to be permitted to run a command as root, or as another user, based on settings in the **/etc/sudoers file**. Unlike other tools such as **su**, **sudo** requires users to enter their own password for authentication, not the password of the account they are trying to access.
- ❑ This allows an administrator to hand out fine-grained permissions to users to delegate system administration tasks, without having to hand out the root password. For example, when **sudo** has been configured to allow the user **student** to run the command **usermod** as root, **student** could run the following command to lock a user account:

MANAGING LOCAL LINUX USERS AND GROUPS

```
[student@serverX ~]$ sudo usermod -L username  
[sudo] password for student: password
```

One additional benefit to using **sudo** is that all commands executed using **sudo** are logged by default to **/var/log/secure**.

```
[student@serverX ~]$ sudo tail /var/log/secure  
...  
Feb 19 15:23:36 localhost sudo: student : TTY=pts/0 ; PWD=/home/student ;  
USER=root ; COMMAND=/sbin/usermod -L student  
  
Feb 19 15:23:36 localhost usermod[16325]: lock user 'student' password  
Feb 19 15:23:47 localhost sudo: student : TTY=pts/0 ; PWD=/home/student ;  
USER=root ; COMMAND=/bin/tail /var/log/secure
```



- ❑ In Red Hat Enterprise Linux 7, all members of group wheel can use **sudo** to run commands as any user, **including root**. The user will be prompted for their own password.
- ❖ This is a change from Red Hat Enterprise Linux 6 and earlier. Users who were members of group wheel did not get this administrative access by default in RHEL 6 and earlier.
- To enable similar behavior on earlier versions of Red Hat Enterprise Linux, use **visudo** to **edit the configuration file and uncomment the line allowing the group wheel to run all commands**.

```
[root@desktopX ~]# cat /etc/sudoers
...Output omitted...
## Allows people in group wheel to run all commands
%wheel          ALL=(ALL)          ALL

## Same thing without a password
# %wheel  ALL=(ALL)          NOPASSWD: ALL
...Output omitted...
```

- ❑ RHEL 7 automatically grants all members of wheel full sudo privileges. This could lead to unauthorized users getting **superuser access** to RHEL 7 systems.
- ❑ Historically, membership in group wheel has been used by Unix-like systems to grant or control **superuser access**.





جامعة الجميع الذكية
EVERYONE'S SMART UNIVERSITY

