**User and Group Management**

1. [Create, delete, and modify local user accounts](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#create-delete-and-modify-local-user-accounts)
2. [Create, delete, and modify local groups and group memberships](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#create-delete-and-modify-local-groups-and-group-memberships)
3. [Manage system-wide environment profiles](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#manage-system-wide-environment-profiles)
4. [Manage template user environment](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#manage-template-user-environment)
5. [Configure user resource limits](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#configure-user-resource-limits)
6. [Manage user privileges](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#manage-user-privileges)
7. [Configure PAM](https://github.com/StenlyTU/LFCS-official/blob/main/stuff/UserandGroupManagement.md#configure-pam)

**Create, delete, and modify local user accounts**

***useradd***

* Add users
* useradd -D - print the default configuration used by useradd command.

GROUP=100

HOME=/home

INACTIVE=-1

EXPIRE=

SHELL=/bin/bash

SKEL=/etc/skel

CREATE\_MAIL\_SPOOL=yes

*GROUP=100* -> default group

*HOME=/home* -> base for home directory

*INACTIVE=-1* -> user password won't expire

*EXPIRE=* -> user account won't expire

*SHELL=/bin/bash* -> default shell

*SKEL=/etc/skel* -> skeleton directory. It's content will be copied in new user home directory.

*CREATE\_MAIL\_SPOOL=yes* -> User will have a mail spool to receive email

* This configuration is saved in /etc/default/useradd
* Also /etc/login.defs parameter are evaluated during user add.
* Some parameter of /etc/login.defs will overwrite /etc/default/useradd parameters.
* /etc/login.defs contains:
  + Location of mail spool
  + Settings about password
  + *CREATE\_HOME yes* -> create home directory
  + *USERGROUPS\_ENAB yes* -> means that a group with same name of user must be created. This group will become default user group. This means that value of GROUP in /etc/default/useradd is overwritten.
* useradd parameters:
  + -c -> Any text string. It is generally a short description of the login, and is currently used as the field for the user's full name.
  + -e -> date after which the user will be disabled.
  + -g -> primary group. NOTE: if not specified it will create new group with same name of user that will become user's primary group.
  + -G -> secondary groups.
  + -m -> create home directory. Useless because CREATE\_HOME is yes.
  + -p -> configure password. **NOTE**: value must be provided encrypted.
    - Normally password is not provided during user add.
  + -s -> shell to use.
  + -N -> Do not create a group with the same name as the user.
* When a user is created two file will be changed:
  + /etc/passwd It contains users information, no passwords.
    - Syntax:
      * user name
      * x: means that password isn't stored here
      * userid: user id (UID)
      * groupid: primary group id (GID)
      * User Info: The comment field
      * home: home directory
      * shell
    - To edit file: vipw
  + /etc/shadow It contains passwords plus passwords properties.
    - To edit file: vipw -s

***usermod***

* used to modify a user
* usermod parameters:
  + -L -> Lock the user account
  + -U -> Unlock the user account
  + -c -> Add comment for the user
  + -s -> Change the default user shell
  + -e -> Expire the user
* usermod -e 1 user - disable user.
* usermod -e "" user - enable user.
* usermod -e 2021-11-27 <username> -> ***Expire the user in one year.***

***userdel***

* remove user
* userdel -r user
  + -r remove home and email spool. **NOTE**: if not used, when new user with the same name is created there will be conflict.
  + -f force. Delete user though he is logged.

***passwd***

* Change password of current user
* passwd user -> Change password of user
* passwd -l user -> Lock password of user
* passwd -u user -> Unlock password for user
* passwd --expire user -> ***Force the user to change it's password into the next login***.
* echo newpass | passwd --stdin brenda
  + It will change password of brenda
  + Can be used in a script
  + **NOTE**: Dangerous, password is in clear text.

chage -> Give Password information for the user.

* Change user password expiry information.
* If used without parameters will prompt for information.
* It will permit to change date when the password was last changed.
* chage -l user
* chage -E 2014-09-11 user
  + Set a date after which user will be locked
* pwunconv/pwconv - Move users password to **/etc/passwd** file.

**Create, delete, and modify local groups and group memberships**

***groupadd***

* add group
* When a group is created /etc/group file will be changed
  + Syntax:
    - group\_name: It is the name of group. If you run ls -l command, you will see this name printed in the group field.
    - Password: Generally password is not used, hence it is empty/blank. It can store encrypted password. This is useful to implement privileged groups.
    - Group ID (GID): group id
      * For each user must be assigned a group ID. You can see this number in your /etc/passwd file.
    - Group List: It is a list of user names of users who are members of the group. The user names, must be separated by commas.
      * **NOTE**: The groups without group list are used as primary group for some users

***groupdel***

* delete group

***groupmod/usermod***

* modify group
  + groupmod -n group\_new group\_old - Rename Linux group.
* usermod -aG group user - Add group to user.
  + -G list of secondary groups
  + -a append. **NOTE**: If not specified new group list will override current value
* groups <username> - Give list of groups for the user.
* gpasswd -d <username> <group> - Delete user from the group.
  + With ***gpasswd*** you can add user to group and specify all group members.
* newgrp wheel - Used to change the current group ID during a login session, NOT creating new group.

**Manage system-wide environment profiles**

* The variable for all users are stored in /etc/environment
* The variable for a user are stored in his home directory in file .bash\_profile
  + **NOTE**: It is an hidden file, it is visible only running ls -la
* su - Create entirely new session in comparison with su where only user and group id are substituted. *$USER* is not set.
* When executing su - the file .bash\_profile is sourced which sources .bashrc
* When executing su only the file .bashrc is sourced.
* -rw-r--r-- /etc/profile Explanation: When invoked interactively with the --login option or when invoked as sh, Bash reads the /etc/profile instructions. These usually set the shell variables PATH, USER, MAIL, HOSTNAME and HISTSIZE. On some systems, the umask value is configured in /etc/profile; on other systems this file holds pointers to other configuration files.
* Scripts in /etc/profile.d are going to be executed with every new session.

**Manage template user environment**

* /etc/skel -> skeleton directory. It's content will be copied into the new users home directory.

**Configure user resource limits**

***ulimit***

* It limits the use of system-wide resources.
* /etc/security/limits.d/ folder is used to add restrictions interactively. The file must end with .conf
* Limits can be configured changing file /etc/security/limits.conf
* Typical configuration:
* 1. @student hard nproc 20
* 2. @faculty soft nproc 20
* 3. ftp hard nproc 0
* 4. @student - maxlogins 4
* 5. %engineers - nproc 10

6. eric soft nproc unlimited

* 1. Members of student group can run only 20 processes
  2. Members of faculty group will receive notification that more than 20 processes were run (soft limit)
  3. ftp user cannot run any process
  4. Members of student group can have maximum 4 logged user.
  5. "-" means both hard and soft
* Default process limits are stored in /etc/security/limits.d/20-nproc.conf
* # Default limit for number of user's processes to prevent
* # accidental fork bombs.
* # See rhbz #432903 for reasoning.
* \* soft nproc 4096

root soft nproc unlimited

* man limits.conf - for manual.
* Limits will be enforced in next opened session.
* ulimit -a -> List all limits for the current user.
* ulimit - command can also be used to change limits.
* **NOTE**: **individual limits have priority over group limits**, so if you impose no limits for admin group, but one of the members in this group have a limits line, the user will have its limits set according to this line.
* **NOTE**: The file /etc/security/limits.conf **does not affect system services**.

**Manage user privileges**

Refer to sudo configuration

**Configure PAM**

* PAM = plugable authentication modules
* A command/program can be PAM aware
* PAM can be used to configure e.g. login to use Active Directory or LDAP
* Use ldd to see if command use PAM libraries
  + ldd /usr/bin/passwd | grep pam
* The main configuration file for PAM is /etc/pam.conf and the /etc/pam.d/ directory contains the PAM configuration files for each PAM-aware application/services. PAM will ignore the file if the directory exists.
* Each command that will use PAM will have an entry in /etc/pam.d with its PAM configuration.
* A good example of PAM configuration is showed in pam\_tally2 module man page.
  + pam\_tally2: The login counter (tallying) module
  + At the end of man page there is an example to configure login to lock the account after 4 failed logins.
  + man pam\_tally2
* Example from /etc/pam.d/chsh
* #%PAM-1.0
* auth sufficient pam\_rootok.so
* auth include system-auth
* account include system-auth
* password include system-auth

session include system-auth

Each configuration line has three fields: a function type, control argument and module.

* + **Function type** - The function that a user application asks PAM to perform. Here, it’s auth, the task of authenticating the user.
    1. ***auth*** - Authenticate a user (see if the user is who they say they are).
    2. ***account*** - Check user account status (whether the user is authorized to do something, for example).
    3. ***session*** - Perform something only for the user’s current session (such as displaying a message of the day).
    4. ***password*** - Change a user’s password or other credentials.

When a hyphen appears before the type, PAM will not record to the system log if the module cannot be loaded because it could not be found in the system.

* + **Control argument** - This setting controls what PAM does after success or failure of its action for the current line
    1. ***requisite*** - If this rule succeeds, PAM proceeds to additional rules. If the rule fails, the authentication is unsuccessful, and PAM does not need to look at any more rules.
    2. ***required*** - If this rule succeeds, PAM proceeds to additional rules. If the rule fails, PAM proceeds to additional rules but will always return an unsuccessful authentication regardless of the end result of the additional rules.
    3. ***sufficient*** - If this rule succeeds, the authentication is successful, and PAM does not need to look at any more rules. If the rule fails, PAM proceeds to additional rules(failure of this module is ignored).
    4. ***optional*** - if the authentication via this module fails or succeeds, nothing happens unless this is the only module of its type defined for this service.
    5. ***include*** - means that the lines of the given type should be read from another file.
    6. ***substack*** - is similar to includes but authentication failures or successes do not cause the exit of the complete module, but only of the substack.
  + **Module** - The authentication module that runs for this line, determining what the line actually does.
    1. PAM modules can take arguments after the module name.

auth sufficient pam\_unix.so nullok

* + The @include syntax loads an entire configuration file, but you can also use a control argument to load only the configuration for a particular function.
  + For more info check man pam
* In /etc/security you can find config files related to PAM:
  + pwscore -> accept as input password and gives you rating from 0 to 100 how strong is the password.
  + Time restriction can be configured in /etc/security/time.conf
  + Password complexity policies can be configured in /etc/security/pwquality.conf