shadow

THE MAN-PAGES BOOK

Maintainers:

Alejandro Colomar <alx@kernel.org> 2023-present (4.14 stable)
Iker Pedrosa <ipedrosa@redhat.com> 2022-present
Christian Brauner <christian@brauner.io> 2019-present
Serge E. Hallyn <serge@hallyn.com> 2014-present
Nicolas François <nicolas.francois@centraliens.net> 2007-2014
Tomasz Kłoczko <kloczek@pld.org.pl> 2000-2007
Marek Michałkiewicz <marekm72@gmail.com> 1995-2000

NAME

chage − change user password expiry information

SYNOPSIS

chage [*options*] *LOGIN*

DESCRIPTION

The **chage** command changes the number of days between password changes and the date of the last password change. This information is used by the system to determine when a user must change their password.

OPTIONS

The options which apply to the **chage** command are:

−d, **−−lastday** *LAST_DAY*
Set the number of days since January 1st, 1970 when the password was last changed. The date may also be expressed in the format YYYY−MM−DD (or the format more commonly used in your area). If the *LAST_DAY* is set to *0* the user is forced to change his password on the next log on.

−E, **−−expiredate** *EXPIRE_DATE*
Set the date or number of days since January 1, 1970 on which the user's account will no longer be accessible. The date may also be expressed in the format YYYY−MM−DD (or the format more commonly used in your area). A user whose account is locked must contact the system administrator before being able to use the system again.

For example the following can be used to set an account to expire in 180 days:

```
chage −E $(date −d +180days +%Y−%m−%d)
```

Passing the number *−1* as the *EXPIRE_DATE* will remove an account expiration date.

−h, **−−help**
Display help message and exit.

−i, **−−iso8601**
When printing dates, use YYYY−MM−DD format.

−I, **−−inactive** *INACTIVE*
Set the number of days of inactivity after a password has expired before the account is locked. The *INACTIVE* option is the number of days of inactivity. A user whose account is locked must contact the system administrator before being able to use the system again.

Passing the number *−1* as the *INACTIVE* will remove an account's inactivity.

−l, **−−list**
Show account aging information.

−m, **−−mindays** *MIN_DAYS*
Set the minimum number of days between password changes to *MIN_DAYS*. A value of zero for this field indicates that the user may change their password at any time.

−M, **−−maxdays** *MAX_DAYS*
Set the maximum number of days during which a password is valid. When *MAX_DAYS* plus *LAST_DAY* is less than the current day, the user will be required to change their password before being able to use their account. This occurrence can be planned for in advance by use of the **−W** option, which provides the user with advance warning.

Passing the number *−1* as *MAX_DAYS* will remove checking a password's validity.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the

CHROOT_DIR directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes to configuration files under the root filesystem found under the directory *PREFIX_DIR*. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−W, **−−warndays** *WARN_DAYS*
Set the number of days of warning before a password change is required. The *WARN_DAYS* option is the number of days prior to the password expiring that a user will be warned their password is about to expire.

If none of the options are selected, **chage** operates in an interactive fashion, prompting the user with the current values for all of the fields. Enter the new value to change the field, or leave the line blank to use the current value. The current value is displayed between a pair of *[ ]* marks.

NOTE

The **chage** program requires a shadow password file to be available.

The chage program will report only the information from the shadow password file. This implies that configuration from other sources (e.g. LDAP or empty password hash field from the passwd file) that affect the user's login will not be shown in the chage output.

The **chage** program will also not report any inconsistency between the shadow and passwd files (e.g. missing x in the passwd file). The **pwck** can be used to check for this kind of inconsistencies.

The **chage** command is restricted to the root user, except for the **−l** option, which may be used by an unprivileged user to determine when their password or account is due to expire.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

EXIT VALUES

The **chage** command exits with the following values:

0
success

1
permission denied

2
invalid command syntax

15
can't find the shadow password file

SEE ALSO

passwd(5), *shadow*(5).

NAME

chfn − change real user name and information

SYNOPSIS

chfn [*options*] [*LOGIN*]

DESCRIPTION

The **chfn** command changes user fullname, office room number, office phone number, and home phone number information for a user's account. This information is typically printed by *finger*(1) and similar programs. A normal user may only change the fields for her own account, subject to the restrictions in /etc/login.defs. (The default configuration is to prevent users from changing their fullname.) The superuser may change any field for any account. Additionally, only the superuser may use the **−o** option to change the undefined portions of the GECOS field.

These fields must not contain any colons. Except for the *other* field, they should not contain any comma or equal sign. It is also recommended to avoid non−US−ASCII characters, but this is only enforced for the phone numbers. The *other* field is used to store accounting information used by other applications.

OPTIONS

The options which apply to the **chfn** command are:

−f, **−−full−name** *FULL_NAME*
Change the user's full name.

−h, **−−home−phone** *HOME_PHONE*
Change the user's home phone number.

−o, **−−other** *OTHER*
Change the user's other GECOS information. This field is used to store accounting information used by other applications, and can be changed only by a superuser.

−r, **−−room** *ROOM_NUMBER*
Change the user's room number.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−u, **−−help**
Display help message and exit.

−w, **−−work−phone** *WORK_PHONE*
Change the user's office phone number.

If none of the options are selected, **chfn** operates in an interactive fashion, prompting the user with the current values for all of the fields. Enter the new value to change the field, or leave the line blank to use the current value. The current value is displayed between a pair of **[ ]** marks. Without options, **chfn** prompts for the current user account.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

CHFN_AUTH (boolean)
If *yes*, the **chfn** program will require authentication before making any changes, unless run by the superuser.

CHFN_RESTRICT (string)
This parameter specifies which values in the *gecos* field of the /etc/passwd file may be changed by regular users using the **chfn** program. It can be any combination of letters *f*, *r*, *w*, *h*, for Full name, Room number, Work phone, and Home phone, respectively. For backward compatibility, *yes* is equivalent to *rwh* and *no* is equivalent to *frwh*. If not specified, only the superuser can make any changes. The most restrictive setting is better achieved by not installing **chfn** SUID.

LOGIN_STRING (string)
The string used for prompting a password. The default is to use "Password: ", or a translation of that string. If you set this variable, the prompt will not be translated.

If the string contains *%s*, this will be replaced by the user's name.

FILES

/etc/login.defs
: Shadow password suite configuration.

/etc/passwd
: User account information.

SEE ALSO

chsh(1), *login.defs*(5), *passwd*(5).

NAME

chsh − change login shell

SYNOPSIS

chsh [*options*] [*LOGIN*]

DESCRIPTION

The **chsh** command changes the user login shell. This determines the name of the user's initial login command. A normal user may only change the login shell for her own account; the superuser may change the login shell for any account.

OPTIONS

The options which apply to the **chsh** command are:

−h, **−−help**
Display help message and exit.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−s, **−−shell** *SHELL*
The name of the user's new login shell. Setting this field to blank causes the system to select the default login shell.

If the **−s** option is not selected, **chsh** operates in an interactive fashion, prompting the user with the current login shell. Enter the new value to change the shell, or leave the line blank to use the current one. The current shell is displayed between a pair of *[ ]* marks.

NOTE

The only restriction placed on the login shell is that the command name must be listed in /etc/shells, unless the invoker is the superuser, and then any value may be added. An account with a restricted login shell may not change her login shell. For this reason, placing /bin/rsh in /etc/shells is discouraged since accidentally changing to a restricted shell would prevent the user from ever changing her login shell back to its original value.

For this reason, placing /bin/rsh in /etc/shells is discouraged since accidentally changing to a restricted shell would prevent the user from ever changing her login shell back to its original value.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

CHSH_AUTH (boolean)
If *yes*, the **chsh** program will require authentication before making any changes, unless run by the superuser.

LOGIN_STRING (string)
The string used for prompting a password. The default is to use "Password: ", or a translation of that string. If you set this variable, the prompt will not be translated.

If the string contains *%s*, this will be replaced by the user's name.

FILES

/etc/passwd
User account information.

/etc/shells
List of valid login shells.

/etc/login.defs
Shadow password suite configuration.

SEE ALSO

chfn(1), *login.defs*(5), *passwd*(5).

NAME

expiry − check and enforce password expiration policy

SYNOPSIS

expiry *option*

DESCRIPTION

The **expiry** command checks (**−c**) the current password expiration and forces (**−f**) changes when required. It is callable as a normal user command.

OPTIONS

The options which apply to the **expiry** command are:

−c, **−−check**
Check the password expiration of the current user.

−f, **−−force**
Force a password change if the current user has an expired password.

−h, **−−help**
Display help message and exit.

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

SEE ALSO

passwd(5), *shadow*(5).

NAME

getsubids − get the subordinate id ranges for a user

SYNOPSIS

getsubids [*options*] *USER*

DESCRIPTION

The **getsubids** command lists the subordinate user ID ranges for a given user. The subordinate group IDs can be listed using the **−g** option.

OPTIONS

The options which apply to the **getsubids** command are:

−g
List the subordinate group ID ranges.

−h
Display help message and exit.

EXAMPLE

For example, to obtain the subordinate UIDs of the testuser:

```
$ getsubids testuser
0: testuser 100000 65536
```

This command output provides (in order from left to right) the list index, username, UID range start, and number of UIDs in range.

SEE ALSO

login.defs(5), *newgidmap*(1), *newuidmap*(1), *subgid*(5), *subuid*(5), *useradd*(8), *userdel*(8). *usermod*(8),

NAME

gpasswd − administer /etc/group and /etc/gshadow

SYNOPSIS

gpasswd [*option*] *group*

DESCRIPTION

The **gpasswd** command is used to administer /etc/group, and /etc/gshadow. Every group can have administrators, members and a password.

System administrators can use the **−A** option to define group administrator(s) and the **−M** option to define members. They have all rights of group administrators and members.

gpasswd called by a group administrator with a group name only prompts for the new password of the *group*.

If a password is set the members can still use *newgrp*(1) without a password, and non−members must supply the password.

Notes about group passwords

Group passwords are an inherent security problem since more than one person is permitted to know the password. However, groups are a useful tool for permitting co−operation between different users.

OPTIONS

Except for the **−A** and **−M** options, the options cannot be combined.

The options which apply to the **gpasswd** command are:

−a, **−−add** *user*
Add the *user* to the named *group*.

−d, **−−delete** *user*
Remove the *user* from the named *group*.

−h, **−−help**
Display help message and exit.

−Q, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−r, **−−remove−password**
Remove the password from the named *group*. The group password will be empty. Only group members will be allowed to use **newgrp** to join the named *group*.

−R, **−−restrict**
Restrict the access to the named *group*. The group password is set to "!". Only group members with a password will be allowed to use **newgrp** to join the named *group*.

−A, **−−administrators** *user*,...
Set the list of administrative users.

−M, **−−members** *user*,...
Set the list of group members.

CAVEATS

This tool only operates on the /etc/group and /etc/gshadow files. Thus you cannot change any NIS or LDAP group. This must be performed on the corresponding server.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

ENCRYPT_METHOD (string)
This defines the system default encryption algorithm for encrypting passwords (if no algorithm are specified on the command line).

It can take one of these values: *DES* (default), *MD5*, *SHA256*, *SHA512*. MD5 and DES should not be used for new hashes, see crypt(5) for recommendations.

Note: this parameter overrides the **MD5_CRYPT_ENAB** variable.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

MD5_CRYPT_ENAB (boolean)
Indicate if passwords must be encrypted using the MD5−based algorithm. If set to *yes*, new passwords will be encrypted using the MD5−based algorithm compatible with the one used by recent releases of FreeBSD. It supports passwords of unlimited length and longer salt strings. Set to *no* if you need to copy encrypted passwords to other systems which don't understand the new algorithm. Default is *no*.

This variable is superseded by the **ENCRYPT_METHOD** variable or by any command line option used to configure the encryption algorithm.

This variable is deprecated. You should use **ENCRYPT_METHOD**.

SHA_CRYPT_MIN_ROUNDS (number), **SHA_CRYPT_MAX_ROUNDS** (number)
When **ENCRYPT_METHOD** is set to *SHA256* or *SHA512*, this defines the number of SHA rounds used by the encryption algorithm by default (when the number of rounds is not specified on the command line).

With a lot of rounds, it is more difficult to brute force the password. But note also that more CPU resources will be needed to authenticate users.

If not specified, the libc will choose the default number of rounds (5000), which is orders of magnitude too low for modern hardware.

The values must be inside the 1000−999,999,999 range.

If only one of the **SHA_CRYPT_MIN_ROUNDS** or **SHA_CRYPT_MAX_ROUNDS** values is set, then this value will be used.

If **SHA_CRYPT_MIN_ROUNDS** > **SHA_CRYPT_MAX_ROUNDS**, the highest value will be used.

FILES

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

SEE ALSO

newgrp(1), *groupadd*(8), *groupdel*(8), *groupmod*(8), *grpck*(8), *group*(5), **gshadow**(5).

NAME

groups − display current group names

SYNOPSIS

groups [*user*]

DESCRIPTION

The **groups** command displays the current group names or ID values. If the value does not have a corresponding entry in /etc/group, the value will be displayed as the numerical group value. The optional *user* parameter will display the groups for the named user.

NOTE

Systems which do not support supplementary groups (see *initgroups*(3)) will have the information from /etc/group reported. The user must use **newgrp** or **sg** to change his current real and effective group ID.

FILES

/etc/group
: Group account information.

SEE ALSO

newgrp(1), *getgid*(2), *getgroups*(2), *getuid*(2), *initgroups*(3).

NAME

login − begin session on the system

SYNOPSIS

login [−p] [−h *host*] [*username*] [*ENV=VAR...*]

login [−p] [−h *host*] −f *username*

login [−p] −r *host*

DESCRIPTION

The **login** program is used to establish a new session with the system. It is normally invoked automatically by responding to the *login:* prompt on the user's terminal. **login** may be special to the shell and may not be invoked as a sub−process. When called from a shell, **login** should be executed as **exec login** which will cause the user to exit from the current shell (and thus will prevent the new logged in user to return to the session of the caller). Attempting to execute **login** from any shell but the login shell will produce an error message.

The user is then prompted for a password, where appropriate. Echoing is disabled to prevent revealing the password. Only a small number of password failures are permitted before **login** exits and the communications link is severed.

If password aging has been enabled for your account, you may be prompted for a new password before proceeding. You will be forced to provide your old password and the new password before continuing. Please refer to *passwd*(1) for more information.

After a successful login, you will be informed of any system messages and the presence of mail. You may turn off the printing of the system message file, /etc/motd, by creating a zero−length file .hushlogin in your login directory. The mail message will be one of "*You have new mail.*", "*You have mail.*", or "*No Mail.*" according to the condition of your mailbox.

Your user and group ID will be set according to their values in the /etc/passwd file. The value for **$HOME**, **$SHELL**, **$PATH**, **$LOGNAME**, and **$MAIL** are set according to the appropriate fields in the password entry. Ulimit, umask and nice values may also be set according to entries in the GECOS field.

On some installations, the environmental variable **$TERM** will be initialized to the terminal type on your tty line, as specified in /etc/ttytype.

An initialization script for your command interpreter may also be executed. Please see the appropriate manual section for more information on this function.

A subsystem login is indicated by the presence of a "*" as the first character of the login shell. The given home directory will be used as the root of a new file system which the user is actually logged into.

The **login** program is NOT responsible for removing users from the utmp file. It is the responsibility of *getty*(8) and *init*(8) to clean up apparent ownership of a terminal session. If you use **login** from the shell prompt without **exec**, the user you use will continue to appear to be logged in even after you log out of the "subsession".

OPTIONS

−f
: Do not perform authentication, user is preauthenticated.

  Note: In that case, *username* is mandatory.

−h
: Name of the remote host for this login.

−p
: Preserve environment.

−r
: Perform autologin protocol for rlogin.

The **−r**, **−h** and **−f** options are only used when **login** is invoked by root.

CAVEATS

This version of **login** has many compilation options, only some of which may be in use at any particular site.

The location of files is subject to differences in system configuration.

The **login** program is NOT responsible for removing users from the utmp file. It is the responsibility of *getty*(8) and *init*(8) to clean up apparent ownership of a terminal session. If you use **login** from the shell prompt without **exec**, the user you use will continue to appear to be logged in even after you log out of the "subsession".

As with any program, **login**'s appearance can be faked. If non−trusted users have physical access to a machine, an attacker could use this to obtain the password of the next person coming to sit in front of the machine. Under Linux, the SAK mechanism can be used by users to initiate a trusted path and prevent this kind of attack.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

CONSOLE (string)
: If defined, either full pathname of a file containing device names (one per line) or a ":" delimited list of device names. Root logins will be allowed only upon these devices.

  If not defined, root will be allowed on any device.

  The device should be specified without the /dev/ prefix.

CONSOLE_GROUPS (string)
: List of groups to add to the user's supplementary groups set when logging in on the console (as determined by the CONSOLE setting). Default is none. Use with caution − it is possible for users to gain permanent access to these groups, even when not logged in on the console.

DEFAULT_HOME (boolean)
: Indicate if login is allowed if we can't cd to the home directory. Default is no.

  If set to *yes*, the user will login in the root (/) directory if it is not possible to cd to her home directory.

ENV_HZ (string)
: If set, it will be used to define the HZ environment variable when a user login. The value must be preceded by *HZ=*. A common value on Linux is *HZ=100*.

ENV_PATH (string)
: If set, it will be used to define the PATH environment variable when a regular user login. The value is a colon separated list of paths (for example */bin:/usr/bin*) and can be preceded by *PATH=*. The default value is *PATH=/bin:/usr/bin*.

ENV_SUPATH (string)
: If set, it will be used to define the PATH environment variable when the superuser login. The value is a colon separated list of paths (for example */sbin:/bin:/usr/sbin:/usr/bin*) and can be preceded by *PATH=*. The default value is *PATH=/sbin:/bin:/usr/sbin:/usr/bin*.

ENV_TZ (string)
: If set, it will be used to define the TZ environment variable when a user login. The value can be the name of a timezone preceded by *TZ=* (for example *TZ=CST6CDT*), or the full path to the file containing the timezone specification (for example /etc/tzname).

  If a full path is specified but the file does not exist or cannot be read, the default is to use *TZ=CST6CDT*.

ENVIRON_FILE (string)
: If this file exists and is readable, login environment will be read from it. Every line should be in the form name=value.

  Lines starting with a # are treated as comment lines and ignored.

ERASECHAR (number)
Terminal ERASE character (*010* = backspace, *0177* = DEL).

The value can be prefixed "0" for an octal value, or "0x" for an hexadecimal value.

FAIL_DELAY (number)
Delay in seconds before being allowed another attempt after a login failure.

FAILLOG_ENAB (boolean)
Enable logging and display of /var/log/faillog login failure info.

FAKE_SHELL (string)
If set, **login** will execute this shell instead of the users' shell specified in /etc/passwd.

FTMP_FILE (string)
If defined, login failures will be logged in this file in a utmp format.

HUSHLOGIN_FILE (string)
If defined, this file can inhibit all the usual chatter during the login sequence. If a full pathname is specified, then hushed mode will be enabled if the user's name or shell are found in the file. If not a full pathname, then hushed mode will be enabled if the file exists in the user's home directory.

ISSUE_FILE (string)
If defined, this file will be displayed before each login prompt.

KILLCHAR (number)
Terminal KILL character (*025* = CTRL/U).

The value can be prefixed "0" for an octal value, or "0x" for an hexadecimal value.

LASTLOG_ENAB (boolean)
Enable logging and display of /var/log/lastlog login time info.

LOGIN_RETRIES (number)
Maximum number of login retries in case of bad password.

LOGIN_STRING (string)
The string used for prompting a password. The default is to use "Password: ", or a translation of that string. If you set this variable, the prompt will not be translated.

If the string contains *%s*, this will be replaced by the user's name.

LOGIN_TIMEOUT (number)
Max time in seconds for login.

LOG_OK_LOGINS (boolean)
Enable logging of successful logins.

LOG_UNKFAIL_ENAB (boolean)
Enable display of unknown usernames when login failures are recorded.

Note: logging unknown usernames may be a security issue if an user enter her password instead of her login name.

MAIL_CHECK_ENAB (boolean)
Enable checking and display of mailbox status upon login.

You should disable it if the shell startup files already check for mail ("mailx −e" or equivalent).

MAIL_DIR (string)
The mail spool directory. This is needed to manipulate the mailbox when its corresponding user account is modified or deleted. If not specified, a compile−time default is used. The parameter CREATE_MAIL_SPOOL in /etc/default/useradd determines whether the mail spool should be created.

MAIL_FILE (string)
Defines the location of the users mail spool files relatively to their home directory.

The **MAIL_DIR** and **MAIL_FILE** variables are used by **useradd**, **usermod**, and **userdel** to create,

move, or delete the user's mail spool.

If **MAIL_CHECK_ENAB** is set to *yes*, they are also used to define the **MAIL** environment variable.

MOTD_FILE (string)
If defined, ":" delimited list of "message of the day" files to be displayed upon login.

NOLOGINS_FILE (string)
If defined, name of file whose presence will inhibit non−root logins. The contents of this file should be a message indicating why logins are inhibited.

PORTTIME_CHECKS_ENAB (boolean)
Enable checking of time restrictions specified in /etc/porttime.

QUOTAS_ENAB (boolean)
Enable setting of resource limits from /etc/limits and ulimit, umask, and niceness from the user's passwd gecos field.

TTYGROUP (string), **TTYPERM** (string)
The terminal permissions: the login tty will be owned by the **TTYGROUP** group, and the permissions will be set to **TTYPERM**.

TTYGROUP can be either the name of a group or a numeric group identifier.

If TTYGROUP is not defined, then the group ownership of the terminal is set to the user's primary group. If TTYPERM is not defined, then the permissions are set to *0600*.

If you have a **write** program which is "setgid" to a special group which owns the terminals, define TTYGROUP to the group number and TTYPERM to 0620. Otherwise leave TTYGROUP commented out and assign TTYPERM to either 622 or 600.

TTYTYPE_FILE (string)
If defined, file which maps tty line to TERM environment parameter. Each line of the file is in a format something like "vt100 tty01".

ULIMIT (number)
Default **ulimit** value.

UMASK (number)
The file mode creation mask is initialized to this value. If not specified, the mask will be initialized to 022.

useradd and **newusers** use this mask to set the mode of the home directory they create if **HOME_MODE** is not set.

It is also used by **login** to define users' initial umask. Note that this mask can be overridden by the user's GECOS line (if **QUOTAS_ENAB** is set) or by the specification of a limit with the *K* identifier in *limits*(5).

USERGROUPS_ENAB (boolean)
Enable setting of the umask group bits to be the same as owner bits (examples: 022 −> 002, 077 −> 007) for non−root users, if the uid is the same as gid, and username is the same as the primary group name.

If set to *yes*, **userdel** will remove the user's group if it contains no more members, and **useradd** will create by default a group with the name of the user.

FILES

/var/run/utmp
List of current login sessions.

/var/log/wtmp
List of previous login sessions.

/etc/passwd
User account information.

/etc/shadow
: Secure user account information.

/etc/motd
: System message of the day file.

/etc/nologin
: Prevent non−root users from logging in.

/etc/ttytype
: List of terminal types.

$HOME/.hushlogin
: Suppress printing of system messages.

/etc/login.defs
: Shadow password suite configuration.

SEE ALSO

mail(1), *passwd*(1), *sh*(1), *su*(1), *login.defs*(5), *nologin*(5), *passwd*(5), *securetty*(5), *getty*(8).

NAME

newgidmap − set the gid mapping of a user namespace

SYNOPSIS

newgidmap *pid gid lowergid count* [*gid lowergid count* [...]]

DESCRIPTION

The **newgidmap** sets /proc/[pid]/gid_map based on its command line arguments and the gids allowed. Subgid delegation can either be managed via /etc/subgid or through the configured NSS subid module. These options are mutually exclusive.

Note that the root group is not exempted from the requirement for a valid /etc/subgid entry.

After the pid argument, **newgidmap** expects sets of 3 integers:

gid
: Beginning of the range of GIDs inside the user namespace.

lowergid
: Beginning of the range of GIDs outside the user namespace.

count
: Length of the ranges (both inside and outside the user namespace).

newgidmap verifies that the caller is the owner of the process indicated by **pid** and that for each of the above sets, each of the GIDs in the range [lowergid, lowergid+count) is allowed to the caller according to /etc/subgid before setting /proc/[pid]/gid_map.

Note that newgidmap may be used only once for a given process.

Instead of an integer process id, the first argument may be specified as *fd:N*, where the integer N is the file descriptor number for the calling process's opened file for /proc/[pid[. In this case, **newgidmap** will use openat(2) to open the gid_map file under that directory, avoiding a TOCTTOU in case the process exits and the pid is immediately reused.

OPTIONS

There currently are no options to the **newgidmap** command.

FILES

/etc/subgid
: List of user's subordinate group IDs.

/proc/[pid]/gid_map
: Mapping of gids from one between user namespaces.

SEE ALSO

login.defs(5), *newusers*(8), *subgid*(5), *useradd*(8), *userdel*(8), *usermod*(8).

NAME

newgrp − log in to a new group

SYNOPSIS

newgrp [−] [*group*]

DESCRIPTION

The **newgrp** command is used to change the current group ID during a login session. If the optional − flag is given, the user's environment will be reinitialized as though the user had logged in, otherwise the current environment, including current working directory, remains unchanged.

newgrp changes the current real group ID to the named group, or to the default group listed in /etc/passwd if no group name is given. **newgrp** also tries to add the group to the user groupset. If not root, the user will be prompted for a password if she does not have a password (in /etc/shadow if this user has an entry in the shadowed password file, or in /etc/passwd otherwise) and the group does, or if the user is not listed as a member and the group has a password. The user will be denied access if the group password is empty and the user is not listed as a member.

If there is an entry for this group in /etc/gshadow, then the list of members and the password of this group will be taken from this file, otherwise, the entry in /etc/group is considered.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

SYSLOG_SG_ENAB (boolean)
Enable "syslog" logging of **sg** activity.

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

SEE ALSO

id(1), *login*(1), *su*(1), *sg*(1), *gpasswd*(1), *group*(5), **gshadow**(5).

NAME

newuidmap − set the uid mapping of a user namespace

SYNOPSIS

newuidmap *pid uid loweruid count* [*uid loweruid count* [...]]

DESCRIPTION

The **newuidmap** sets /proc/[pid]/uid_map based on its command line arguments and the uids allowed. Subuid delegation can either be managed via /etc/subuid or through the configured NSS subid module. These options are mutually exclusive.

Note that the root user is not exempted from the requirement for a valid /etc/subuid entry.

After the pid argument, **newuidmap** expects sets of 3 integers:

uid
: Beginning of the range of UIDs inside the user namespace.

loweruid
: Beginning of the range of UIDs outside the user namespace.

count
: Length of the ranges (both inside and outside the user namespace).

newuidmap verifies that the caller is the owner of the process indicated by **pid** and that for each of the above sets, each of the UIDs in the range [loweruid, loweruid+count) is allowed to the caller according to /etc/subuid before setting /proc/[pid]/uid_map.

Note that newuidmap may be used only once for a given process.

Instead of an integer process id, the first argument may be specified as *fd:N*, where the integer N is the file descriptor number for the calling process's opened file for /proc/[pid[. In this case, **newuidmap** will use openat(2) to open the uid_map file under that directory, avoiding a TOCTTOU in case the process exits and the pid is immediately reused.

OPTIONS

There currently are no options to the **newuidmap** command.

FILES

/etc/subuid
: List of user's subordinate user IDs.

/proc/[pid]/uid_map
: Mapping of uids from one between user namespaces.

SEE ALSO

login.defs(5), *newusers*(8), *subuid*(5), *useradd*(8), *usermod*(8), *userdel*(8).

NAME

passwd − change user password

SYNOPSIS

passwd [*options*] [*LOGIN*]

DESCRIPTION

The **passwd** command changes passwords for user accounts. A normal user may only change the password for their own account, while the superuser may change the password for any account. **passwd** also changes the account or associated password validity period.

Password Changes

The user is first prompted for their old password, if one is present. This password is then encrypted and compared against the stored password. The user has only one chance to enter the correct password. The superuser is permitted to bypass this step so that forgotten passwords may be changed.

After the password has been entered, password aging information is checked to see if the user is permitted to change the password at this time. If not, **passwd** refuses to change the password and exits.

The user is then prompted twice for a replacement password. The second entry is compared against the first and both are required to match in order for the password to be changed.

Then, the password is tested for complexity. **passwd** will reject any password which is not suitably complex. Care must be taken not to include the system default erase or kill characters.

Hints for user passwords

The security of a password depends upon the strength of the encryption algorithm and the size of the key space. The legacy *UNIX* System encryption method is based on the NBS DES algorithm. More recent methods are now recommended (see **ENCRYPT_METHOD**). The size of the key space depends upon the randomness of the password which is selected.

Compromises in password security normally result from careless password selection or handling. For this reason, you should not select a password which appears in a dictionary or which must be written down. The password should also not be a proper name, your license number, birth date, or street address. Any of these may be used as guesses to violate system security.

As a general guideline, passwords should be long and random. It's fine to use simple character sets, such as passwords consisting only of lowercase letters, if that helps memorizing longer passwords. For a password consisting only of lowercase English letters randomly chosen, and a length of 32, there are 26^32 (approximately 2^150) different possible combinations. Being an exponential equation, it's apparent that the exponent (the length) is more important than the base (the size of the character set).

You can find advice on how to choose a strong password on http://en.wikipedia.org/wiki/Password_strength

OPTIONS

The options which apply to the **passwd** command are:

−a, **−−all**
This option can be used only with **−S** and causes show status for all users.

−d, **−−delete**
Delete a user's password (make it empty). This is a quick way to disable a password for an account. It will set the named account passwordless.

−e, **−−expire**
Immediately expire an account's password. This in effect can force a user to change their password at the user's next login.

−h, **−−help**
Display help message and exit.

−i, **−−inactive** *INACTIVE*
This option is used to disable an account after the password has been expired for a number of days. After a user account has had an expired password for *INACTIVE* days, the user may no longer sign on to the account.

−k, **−−keep−tokens**
Indicate password change should be performed only for expired authentication tokens (passwords). The user wishes to keep their non−expired tokens as before.

−l, **−−lock**
Lock the password of the named account. This option disables a password by changing it to a value which matches no possible encrypted value (it adds a ´!´ at the beginning of the password).

Note that this does not disable the account. The user may still be able to login using another authentication token (e.g. an SSH key). To disable the account, administrators should use **usermod −−expiredate 1** (this set the account's expire date to Jan 2, 1970).

Users with a locked password are not allowed to change their password.

−n, **−−mindays** *MIN_DAYS*
Set the minimum number of days between password changes to *MIN_DAYS*. A value of zero for this field indicates that the user may change their password at any time.

−q, **−−quiet**
Quiet mode.

−r, **−−repository** *REPOSITORY*
change password in *REPOSITORY* repository

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes to configuration files under the root filesystem found under the directory *PREFIX_DIR*. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−S, **−−status**
Display account status information. The status information consists of 7 fields. The first field is the user's login name. The second field indicates if the user account has a locked password (L), has no password (NP), or has a usable password (P). The third field gives the date of the last password change. The next four fields are the minimum age, maximum age, warning period, and inactivity period for the password. These ages are expressed in days.

−u, **−−unlock**
Unlock the password of the named account. This option re−enables a password by changing the password back to its previous value (to the value before using the **−l** option).

−w, **−−warndays** *WARN_DAYS*
Set the number of days of warning before a password change is required. The *WARN_DAYS* option is the number of days prior to the password expiring that a user will be warned that their password is about to expire.

−x, **−−maxdays** *MAX_DAYS*
Set the maximum number of days a password remains valid. After *MAX_DAYS*, the password is required to be changed.

Passing the number *−1* as *MAX_DAYS* will remove checking a password's validity.

−s, **−−stdin**
This option is used to indicate that passwd should read the new password from standard input, which can be a pipe.

CAVEATS

Password complexity checking may vary from site to site. The user is urged to select a password as complex as he or she feels comfortable with.

Users may not be able to change their password on a system if NIS is enabled and they are not logged into the NIS server.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

ENCRYPT_METHOD (string)
: This defines the system default encryption algorithm for encrypting passwords (if no algorithm are specified on the command line).

  It can take one of these values: *DES* (default), *MD5*, *SHA256*, *SHA512*. MD5 and DES should not be used for new hashes, see crypt(5) for recommendations.

  Note: this parameter overrides the **MD5_CRYPT_ENAB** variable.

MD5_CRYPT_ENAB (boolean)
: Indicate if passwords must be encrypted using the MD5−based algorithm. If set to *yes*, new passwords will be encrypted using the MD5−based algorithm compatible with the one used by recent releases of FreeBSD. It supports passwords of unlimited length and longer salt strings. Set to *no* if you need to copy encrypted passwords to other systems which don't understand the new algorithm. Default is *no*.

  This variable is superseded by the **ENCRYPT_METHOD** variable or by any command line option used to configure the encryption algorithm.

  This variable is deprecated. You should use **ENCRYPT_METHOD**.

OBSCURE_CHECKS_ENAB (boolean)
: Enable additional checks upon password changes.

PASS_ALWAYS_WARN (boolean)
: Warn about weak passwords (but still allow them) if you are root.

PASS_CHANGE_TRIES (number)
: Maximum number of attempts to change password if rejected (too easy).

PASS_MAX_LEN (number), **PASS_MIN_LEN** (number)
: Number of significant characters in the password for crypt(). **PASS_MAX_LEN** is 8 by default. Don't change unless your crypt() is better. This is ignored if **MD5_CRYPT_ENAB** set to *yes*.

SHA_CRYPT_MIN_ROUNDS (number), **SHA_CRYPT_MAX_ROUNDS** (number)
: When **ENCRYPT_METHOD** is set to *SHA256* or *SHA512*, this defines the number of SHA rounds used by the encryption algorithm by default (when the number of rounds is not specified on the command line).

  With a lot of rounds, it is more difficult to brute force the password. But note also that more CPU resources will be needed to authenticate users.

  If not specified, the libc will choose the default number of rounds (5000), which is orders of magnitude too low for modern hardware.

  The values must be inside the 1000−999,999,999 range.

  If only one of the **SHA_CRYPT_MIN_ROUNDS** or **SHA_CRYPT_MAX_ROUNDS** values is set, then this value will be used.

  If **SHA_CRYPT_MIN_ROUNDS** > **SHA_CRYPT_MAX_ROUNDS**, the highest value will be used.

FILES

/etc/passwd
: User account information.

/etc/shadow
: Secure user account information.

/etc/login.defs
: Shadow password suite configuration.

EXIT VALUES

The **passwd** command exits with the following values:

0
: success

1
: permission denied

2
: invalid combination of options

3
: unexpected failure, nothing done

4
: unexpected failure, passwd file missing

5
: passwd file busy, try again

6
: invalid argument to option

SEE ALSO

chpasswd(8), *makepasswd*(1), *passwd*(5), *shadow*(5), *login.defs*(5), *usermod*(8).

The following web page comically (yet correctly) compares the strength of two different methods for choosing a password: "https://xkcd.com/936/"

NAME

sg − execute command as different group ID

SYNOPSIS

sg [−] [group [−c] command]

DESCRIPTION

The **sg** command works similar to **newgrp** but accepts a command. The command will be executed with the /bin/sh shell. With most shells you may run **sg** from, you need to enclose multi−word commands in quotes. Another difference between **newgrp** and **sg** is that some shells treat **newgrp** specially, replacing themselves with a new instance of a shell that **newgrp** creates. This doesn't happen with **sg**, so upon exit from a **sg** command you are returned to your previous group ID.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

SYSLOG_SG_ENAB (boolean)
Enable "syslog" logging of **sg** activity.

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

SEE ALSO

id(1), *login*(1), *newgrp*(1), *su*(1), *gpasswd*(1), *group*(5), **gshadow**(5).

NAME

su − change user ID or become superuser

SYNOPSIS

su [*options*] [−] [*username* [*args*]]

DESCRIPTION

The **su** command is used to become another user during a login session. Invoked without a **username**, **su** defaults to becoming the superuser. The − option may be used to provide an environment similar to what the user would expect had the user logged in directly. The **−c** option may be used to treat the next argument as a command by most shells.

Options are recognized everywhere in the argument list. You can use the −− argument to stop option parsing. The − option is special: it is also recognized after −−, but has to be placed before **username**.

The user will be prompted for a password, if appropriate. Invalid passwords will produce an error message. All attempts, both valid and invalid, are logged to detect abuse of the system.

The current environment is passed to the new shell. The value of **$PATH** is reset to /bin:/usr/bin for normal users, or /sbin:/bin:/usr/sbin:/usr/bin for the superuser. This may be changed with the **ENV_PATH** and **ENV_SUPATH** definitions in /etc/login.defs.

A subsystem login is indicated by the presence of a "*" as the first character of the login shell. The given home directory will be used as the root of a new file system which the user is actually logged into.

OPTIONS

The options which apply to the **su** command are:

−c, **−−command** *COMMAND*
Specify a command that will be invoked by the shell using its **−c**.

The executed command will have no controlling terminal. This option cannot be used to execute interactive programs which need a controlling TTY.

−, **−l**, **−−login**
Provide an environment similar to what the user would expect had the user logged in directly.

When − is used, it must be specified before any **username**. For portability it is recommended to use it as last option, before any **username**. The other forms (**−l** and **−−login**) do not have this restriction.

−s, **−−shell** *SHELL*
The shell that will be invoked.

The invoked shell is chosen from (highest priority first):

The shell specified with −−shell.

If **−−preserve−environment** is used, the shell specified by the **$SHELL** environment variable.

The shell indicated in the /etc/passwd entry for the target user.

/bin/sh if a shell could not be found by any above method.

If the target user has a restricted shell (i.e. the shell field of this user's entry in /etc/passwd is not listed in /etc/shells), then the **−−shell** option or the **$SHELL** environment variable won't be taken into account, unless **su** is called by root.

−m, **−p**, **−−preserve−environment**
Preserve the current environment, except for:

$PATH
reset according to the /etc/login.defs options **ENV_PATH** or **ENV_SUPATH** (see below);

$IFS

reset to “<space><tab><newline>”, if it was set.

If the target user has a restricted shell, this option has no effect (unless **su** is called by root).

Note that the default behavior for the environment is the following:

The **$HOME**, **$SHELL**, **$USER**, **$LOGNAME**, **$PATH**, and **$IFS** environment variables are reset.

If **−−login** is not used, the environment is copied, except for the variables above.

If **−−login** is used, the **$TERM**, **$COLORTERM**, **$DISPLAY**, and **$XAUTHORITY** environment variables are copied if they were set.

If **−−login** is used, the **$TZ**, **$HZ**, and **$MAIL** environment variables are set according to the /etc/login.defs options **ENV_TZ**, **ENV_HZ**, **MAIL_DIR**, and **MAIL_FILE** (see below).

If **−−login** is used, other environment variables might be set by the **ENVIRON_FILE** file (see below).

CAVEATS

This version of **su** has many compilation options, only some of which may be in use at any particular site.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

CONSOLE (string)
If defined, either full pathname of a file containing device names (one per line) or a ":" delimited list of device names. Root logins will be allowed only upon these devices.

If not defined, root will be allowed on any device.

The device should be specified without the /dev/ prefix.

CONSOLE_GROUPS (string)
List of groups to add to the user's supplementary groups set when logging in on the console (as determined by the CONSOLE setting). Default is none. Use with caution − it is possible for users to gain permanent access to these groups, even when not logged in on the console.

DEFAULT_HOME (boolean)
Indicate if login is allowed if we can't cd to the home directory. Default is no.

If set to *yes*, the user will login in the root (/) directory if it is not possible to cd to her home directory.

ENV_HZ (string)
If set, it will be used to define the HZ environment variable when a user login. The value must be preceded by *HZ=*. A common value on Linux is *HZ=100*.

ENVIRON_FILE (string)
If this file exists and is readable, login environment will be read from it. Every line should be in the form name=value.

Lines starting with a # are treated as comment lines and ignored.

ENV_PATH (string)
If set, it will be used to define the PATH environment variable when a regular user login. The value is a colon separated list of paths (for example */bin:/usr/bin*) and can be preceded by *PATH=*. The default value is *PATH=/bin:/usr/bin*.

ENV_SUPATH (string)
If set, it will be used to define the PATH environment variable when the superuser login. The value is a colon separated list of paths (for example */sbin:/bin:/usr/sbin:/usr/bin*) and can be preceded by *PATH=*. The default value is *PATH=/sbin:/bin:/usr/sbin:/usr/bin*.

ENV_TZ (string)

If set, it will be used to define the TZ environment variable when a user login. The value can be the name of a timezone preceded by *TZ=* (for example *TZ=CST6CDT*), or the full path to the file containing the timezone specification (for example /etc/tzname).

If a full path is specified but the file does not exist or cannot be read, the default is to use *TZ=CST6CDT*.

LOGIN_STRING (string)
The string used for prompting a password. The default is to use "Password: ", or a translation of that string. If you set this variable, the prompt will not be translated.

If the string contains *%s*, this will be replaced by the user's name.

MAIL_CHECK_ENAB (boolean)
Enable checking and display of mailbox status upon login.

You should disable it if the shell startup files already check for mail ("mailx −e" or equivalent).

MAIL_DIR (string)
The mail spool directory. This is needed to manipulate the mailbox when its corresponding user account is modified or deleted. If not specified, a compile−time default is used. The parameter CREATE_MAIL_SPOOL in /etc/default/useradd determines whether the mail spool should be created.

MAIL_FILE (string)
Defines the location of the users mail spool files relatively to their home directory.

The **MAIL_DIR** and **MAIL_FILE** variables are used by **useradd**, **usermod**, and **userdel** to create, move, or delete the user's mail spool.

If **MAIL_CHECK_ENAB** is set to *yes*, they are also used to define the **MAIL** environment variable.

QUOTAS_ENAB (boolean)
Enable setting of resource limits from /etc/limits and ulimit, umask, and niceness from the user's passwd gecos field.

SULOG_FILE (string)
If defined, all su activity is logged to this file.

SU_NAME (string)
If defined, the command name to display when running "su −". For example, if this is defined as "su" then a "ps" will display the command is "−su". If not defined, then "ps" would display the name of the shell actually being run, e.g. something like "−sh".

SU_WHEEL_ONLY (boolean)
If *yes*, the user must be listed as a member of the first gid 0 group in /etc/group (called *root* on most Linux systems) to be able to **su** to uid 0 accounts. If the group doesn't exist or is empty, no one will be able to **su** to uid 0.

SYSLOG_SU_ENAB (boolean)
Enable "syslog" logging of **su** activity − in addition to sulog file logging.

USERGROUPS_ENAB (boolean)
Enable setting of the umask group bits to be the same as owner bits (examples: 022 −> 002, 077 −> 007) for non−root users, if the uid is the same as gid, and username is the same as the primary group name.

If set to *yes*, **userdel** will remove the user's group if it contains no more members, and **useradd** will create by default a group with the name of the user.

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/login.defs

Shadow password suite configuration.

EXIT VALUES

On success, **su** returns the exit value of the command it executed.

If this command was terminated by a signal, **su** returns the number of this signal plus 128.

If su has to kill the command (because it was asked to terminate, and the command did not terminate in time), **su** returns 255.

Some exit values from **su** are independent from the executed command:

0
success (**−−help** only)

1
System or authentication failure

126
The requested command was not found

127
The requested command could not be executed

SEE ALSO

login(1), *login.defs*(5), *sg*(1), *sh*(1).

NAME

shadow, getspnam − encrypted password file routines

SYNTAX

#include <shadow.h>

*struct spwd *getspent();*

*struct spwd *getspnam(char *name);*

void setspent();

void endspent();

*struct spwd *fgetspent(FILE *fp);*

*struct spwd *sgetspent(char *cp);*

*int putspent(struct spwd *p, FILE *fp);*

int lckpwdf();

int ulckpwdf();

DESCRIPTION

shadow manipulates the contents of the shadow password file, /etc/shadow. The structure in the *#include* file is:

```
struct spwd {
    char          *sp_namp; /* user login name */
    char          *sp_pwdp; /* encrypted password */
    long int      sp_lstchg; /* last password change */
    long int      sp_min; /* days until change allowed. */
    long int      sp_max; /* days before change required */
    long int      sp_warn; /* days warning for expiration */
    long int      sp_inact; /* days before account inactive */
    long int      sp_expire; /* date when account expires */
    unsigned long int sp_flag; /* reserved for future use */
}
```

The meanings of each field are:

- sp_namp − pointer to null−terminated user name
- sp_pwdp − pointer to null−terminated password
- sp_lstchg − days since Jan 1, 1970 password was last changed
- sp_min − days before which password may not be changed
- sp_max − days after which password must be changed
- sp_warn − days before password is to expire that user is warned of pending password expiration
- sp_inact − days after password expires that account is considered inactive and disabled
- sp_expire − days since Jan 1, 1970 when account will be disabled
- sp_flag − reserved for future use

DESCRIPTION

getspent, *getspname*, *fgetspent*, and *sgetspent* each return a pointer to a *struct spwd*. *getspent* returns the next entry from the file, and *fgetspent* returns the next entry from the given stream, which is assumed to be a file of the proper format. *sgetspent* returns a pointer to a *struct spwd* using the provided string as input. *getspnam* searches from the current position in the file for an entry matching *name*.

setspent and *endspent* may be used to begin and end, respectively, access to the shadow password file.

The *lckpwdf* and *ulckpwdf* routines should be used to insure exclusive access to the /etc/shadow file. *lckpwdf* attempts to acquire a lock using *pw_lock* for up to 15 seconds. It continues by attempting to

acquire a second lock using *spw_lock* for the remainder of the initial 15 seconds. Should either attempt fail after a total of 15 seconds, *lckpwdf* returns −1. When both locks are acquired 0 is returned.

DIAGNOSTICS

Routines return NULL if no more entries are available or if an error occurs during processing. Routines which have *int* as the return value return 0 for success and −1 for failure.

CAVEATS

These routines may only be used by the superuser as access to the shadow password file is restricted.

FILES

/etc/shadow
: Secure user account information.

SEE ALSO

getpwent(3), *shadow*(5).

NAME

faillog − login failure logging file

DESCRIPTION

/var/log/faillog maintains a count of login failures and the limits for each account.

The file contains fixed length records, indexed by numerical UID. Each record contains the count of login failures since the last successful login; the maximum number of failures before the account is disabled; the line on which the last login failure occurred; the date of the last login failure; and the duration (in seconds) during which the account will be locked after a failure.

The structure of the file is:

```
struct    faillog {
          short   fail_cnt;
          short   fail_max;
          char    fail_line[12];
          time_t  fail_time;
          long    fail_locktime;
};
```

FILES

/var/log/faillog
: Failure logging file.

SEE ALSO

faillog(8)

NAME

gshadow − shadowed group file

DESCRIPTION

/etc/gshadow contains the shadowed information for group accounts.

This file must not be readable by regular users if password security is to be maintained.

Each line of this file contains the following colon−separated fields:

group name
It must be a valid group name, which exist on the system.

encrypted password
Refer to *crypt*(3) for details on how this string is interpreted.

If the password field contains some string that is not a valid result of *crypt*(3), for instance ! or *, users will not be able to use a unix password to access the group (but group members do not need the password).

The password is used when a user who is not a member of the group wants to gain the permissions of this group (see *newgrp*(1)).

This field may be empty, in which case only the group members can gain the group permissions.

A password field which starts with an exclamation mark means that the password is locked. The remaining characters on the line represent the password field before the password was locked.

This password supersedes any password specified in /etc/group.

administrators
It must be a comma−separated list of user names.

Administrators can change the password or the members of the group.

Administrators also have the same permissions as the members (see below).

members
It must be a comma−separated list of user names.

Members can access the group without being prompted for a password.

You should use the same list of users as in /etc/group.

FILES

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

SEE ALSO

gpasswd(5), *group*(5), *grpck*(8), *grpconv*(8), *newgrp*(1).

NAME

limits − resource limits definition

DESCRIPTION

The *limits* file (/etc/limits by default or LIMITS_FILE defined config.h) describes the resource limits you wish to impose. It should be owned by root and readable by root account only.

By default no quota is imposed on 'root'. In fact, there is no way to impose limits via this procedure to root−equiv accounts (accounts with UID 0).

Each line describes a limit for a user in the form:

user LIMITS_STRING

or in the form:

@group LIMITS_STRING

The *LIMITS_STRING* is a string of a concatenated list of resource limits. Each limit consists of a letter identifier followed by a numerical limit.

The valid identifiers are:

- A: max address space (KB)
- C: max core file size (KB)
- D: max data size (KB)
- F: maximum file size (KB)
- K: file creation mask, set by *umask*(2).
- I: max nice value (0..39 which translates to 20..−19)
- L: max number of logins for this user
- M: max locked−in−memory address space (KB)
- N: max number of open files
- O: max real time priority
- P: process priority, set by *setpriority*(2).
- R: max resident set size (KB)
- S: max stack size (KB)
- T: max CPU time (MIN)
- U: max number of processes

For example, *L2D2048N5* is a valid *LIMITS_STRING*. For reading convenience, the following entries are equivalent:

```
username L2D2048N5
username L2 D2048 N5
```

Be aware that after *username* the rest of the line is considered a limit string, thus comments are not allowed. An invalid limits string will be rejected (not considered) by the **login** program.

The default entry is denoted by username "*". If you have multiple *default* entries in your *LIMITS_FILE*, then the last one will be used as the default entry.

The limits specified in the form "*@group*" apply to the members of the specified *group*.

If more than one line with limits for a user exist, only the first line for this user will be considered.

If no lines are specified for a user, the last *@group* line matching a group whose the user is a member of will be considered, or the last line with default limits if no groups contain the user.

To completely disable limits for a user, a single dash "−" will do.

To disable a limit for a user, a single dash "−" can be used instead of the numerical value for this limit.

Also, please note that all limit settings are set PER LOGIN. They are not global, nor are they

permanent. Perhaps global limits will come, but for now this will have to do ;)

FILES

/etc/limits

SEE ALSO

login(1), *setpriority*(2), *setrlimit*(2).

NAME

login.access − login access control table

DESCRIPTION

The *login.access* file specifies (user, host) combinations and/or (user, tty) combinations for which a login will be either accepted or refused.

When someone logs in, the *login.access* is scanned for the first entry that matches the (user, host) combination, or, in case of non−networked logins, the first entry that matches the (user, tty) combination. The permissions field of that table entry determines whether the login will be accepted or refused.

Each line of the login access control table has three fields separated by a ":" character:

permission:*users*:*origins*

The first field should be a "+" (access granted) or "−" (access denied) character. The second field should be a list of one or more login names, group names, or *ALL* (always matches). The third field should be a list of one or more tty names (for non−networked logins), host names, domain names (begin with "."), host addresses, internet network numbers (end with "."), *ALL* (always matches) or *LOCAL* (matches any string that does not contain a "." character). If you run NIS you can use @netgroupname in host or user patterns.

The *EXCEPT* operator makes it possible to write very compact rules.

The group file is searched only when a name does not match that of the logged−in user. Only groups are matched in which users are explicitly listed: the program does not look at a user's primary group id value.

FILES

/etc/login.defs
: Shadow password suite configuration.

SEE ALSO

login(1).

NAME

login.defs − shadow password suite configuration

DESCRIPTION

The /etc/login.defs file defines the site−specific configuration for the shadow password suite. This file is required. Absence of this file will not prevent system operation, but will probably result in undesirable operation.

This file is a readable text file, each line of the file describing one configuration parameter. The lines consist of a configuration name and value, separated by whitespace. Blank lines and comment lines are ignored. Comments are introduced with a "#" pound sign and the pound sign must be the first non−white character of the line.

Parameter values may be of four types: strings, booleans, numbers, and long numbers. A string is comprised of any printable characters. A boolean should be either the value *yes* or *no*. An undefined boolean parameter or one with a value other than these will be given a *no* value. Numbers (both regular and long) may be either decimal values, octal values (precede the value with *0*) or hexadecimal values (precede the value with *0x*). The maximum value of the regular and long numeric parameters is machine−dependent.

The following configuration items are provided:

CHFN_AUTH (boolean)
If *yes*, the **chfn** program will require authentication before making any changes, unless run by the superuser.

CHFN_RESTRICT (string)
This parameter specifies which values in the *gecos* field of the /etc/passwd file may be changed by regular users using the **chfn** program. It can be any combination of letters *f*, *r*, *w*, *h*, for Full name, Room number, Work phone, and Home phone, respectively. For backward compatibility, *yes* is equivalent to *rwh* and *no* is equivalent to *frwh*. If not specified, only the superuser can make any changes. The most restrictive setting is better achieved by not installing **chfn** SUID.

CHSH_AUTH (boolean)
If *yes*, the **chsh** program will require authentication before making any changes, unless run by the superuser.

CONSOLE (string)
If defined, either full pathname of a file containing device names (one per line) or a ":" delimited list of device names. Root logins will be allowed only upon these devices.

If not defined, root will be allowed on any device.

The device should be specified without the /dev/ prefix.

CONSOLE_GROUPS (string)
List of groups to add to the user's supplementary groups set when logging in on the console (as determined by the CONSOLE setting). Default is none. Use with caution − it is possible for users to gain permanent access to these groups, even when not logged in on the console.

CREATE_HOME (boolean)
Indicate if a home directory should be created by default for new users.

This setting does not apply to system users, and can be overridden on the command line.

DEFAULT_HOME (boolean)
Indicate if login is allowed if we can't cd to the home directory. Default is no.

If set to *yes*, the user will login in the root (/) directory if it is not possible to cd to her home directory.

ENCRYPT_METHOD (string)
This defines the system default encryption algorithm for encrypting passwords (if no algorithm are specified on the command line).

It can take one of these values: *DES* (default), *MD5*, *SHA256*, *SHA512*. MD5 and DES should not

be used for new hashes, see crypt(5) for recommendations.

Note: this parameter overrides the **MD5_CRYPT_ENAB** variable.

ENV_HZ (string)
If set, it will be used to define the HZ environment variable when a user login. The value must be preceded by *HZ=*. A common value on Linux is *HZ=100*.

ENV_PATH (string)
If set, it will be used to define the PATH environment variable when a regular user login. The value is a colon separated list of paths (for example */bin:/usr/bin*) and can be preceded by *PATH=*. The default value is *PATH=/bin:/usr/bin*.

ENV_SUPATH (string)
If set, it will be used to define the PATH environment variable when the superuser login. The value is a colon separated list of paths (for example */sbin:/bin:/usr/sbin:/usr/bin*) and can be preceded by *PATH=*. The default value is *PATH=/sbin:/bin:/usr/sbin:/usr/bin*.

ENV_TZ (string)
If set, it will be used to define the TZ environment variable when a user login. The value can be the name of a timezone preceded by *TZ=* (for example *TZ=CST6CDT*), or the full path to the file containing the timezone specification (for example /etc/tzname).

If a full path is specified but the file does not exist or cannot be read, the default is to use *TZ=CST6CDT*.

ENVIRON_FILE (string)
If this file exists and is readable, login environment will be read from it. Every line should be in the form name=value.

Lines starting with a # are treated as comment lines and ignored.

ERASECHAR (number)
Terminal ERASE character (*010* = backspace, *0177* = DEL).

The value can be prefixed "0" for an octal value, or "0x" for an hexadecimal value.

FAIL_DELAY (number)
Delay in seconds before being allowed another attempt after a login failure.

FAILLOG_ENAB (boolean)
Enable logging and display of /var/log/faillog login failure info.

FAKE_SHELL (string)
If set, **login** will execute this shell instead of the users' shell specified in /etc/passwd.

FTMP_FILE (string)
If defined, login failures will be logged in this file in a utmp format.

GID_MAX (number), **GID_MIN** (number)
Range of group IDs used for the creation of regular groups by **useradd**, **groupadd**, or **newusers**.

The default value for **GID_MIN** (resp. **GID_MAX**) is 1000 (resp. 60000).

HMAC_CRYPTO_ALGO (string)
Used to select the HMAC cryptography algorithm that the pam_timestamp module is going to use to calculate the keyed−hash message authentication code.

Note: Check hmac(3) to see the possible algorithms that are available in your system.

HOME_MODE (number)
The mode for new home directories. If not specified, the **UMASK** is used to create the mode.

useradd and **newusers** use this to set the mode of the home directory they create.

HUSHLOGIN_FILE (string)
If defined, this file can inhibit all the usual chatter during the login sequence. If a full pathname is

specified, then hushed mode will be enabled if the user's name or shell are found in the file. If not a full pathname, then hushed mode will be enabled if the file exists in the user's home directory.

ISSUE_FILE (string)
If defined, this file will be displayed before each login prompt.

KILLCHAR (number)
Terminal KILL character (*025* = CTRL/U).

The value can be prefixed "0" for an octal value, or "0x" for an hexadecimal value.

LASTLOG_ENAB (boolean)
Enable logging and display of /var/log/lastlog login time info.

LASTLOG_UID_MAX (number)
Highest user ID number for which the lastlog entries should be updated. As higher user IDs are usually tracked by remote user identity and authentication services there is no need to create a huge sparse lastlog file for them.

No **LASTLOG_UID_MAX** option present in the configuration means that there is no user ID limit for writing lastlog entries.

LOG_OK_LOGINS (boolean)
Enable logging of successful logins.

LOG_UNKFAIL_ENAB (boolean)
Enable display of unknown usernames when login failures are recorded.

Note: logging unknown usernames may be a security issue if an user enter her password instead of her login name.

LOGIN_RETRIES (number)
Maximum number of login retries in case of bad password.

LOGIN_STRING (string)
The string used for prompting a password. The default is to use "Password: ", or a translation of that string. If you set this variable, the prompt will not be translated.

If the string contains *%s*, this will be replaced by the user's name.

LOGIN_TIMEOUT (number)
Max time in seconds for login.

MAIL_CHECK_ENAB (boolean)
Enable checking and display of mailbox status upon login.

You should disable it if the shell startup files already check for mail ("mailx −e" or equivalent).

MAIL_DIR (string)
The mail spool directory. This is needed to manipulate the mailbox when its corresponding user account is modified or deleted. If not specified, a compile−time default is used. The parameter CREATE_MAIL_SPOOL in /etc/default/useradd determines whether the mail spool should be created.

MAIL_FILE (string)
Defines the location of the users mail spool files relatively to their home directory.

The **MAIL_DIR** and **MAIL_FILE** variables are used by **useradd**, **usermod**, and **userdel** to create, move, or delete the user's mail spool.

If **MAIL_CHECK_ENAB** is set to *yes*, they are also used to define the **MAIL** environment variable.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

MD5_CRYPT_ENAB (boolean)
Indicate if passwords must be encrypted using the MD5−based algorithm. If set to *yes*, new passwords will be encrypted using the MD5−based algorithm compatible with the one used by recent releases of FreeBSD. It supports passwords of unlimited length and longer salt strings. Set to *no* if you need to copy encrypted passwords to other systems which don't understand the new algorithm. Default is *no*.

This variable is superseded by the **ENCRYPT_METHOD** variable or by any command line option used to configure the encryption algorithm.

This variable is deprecated. You should use **ENCRYPT_METHOD**.

MOTD_FILE (string)
If defined, ":" delimited list of "message of the day" files to be displayed upon login.

NOLOGINS_FILE (string)
If defined, name of file whose presence will inhibit non−root logins. The contents of this file should be a message indicating why logins are inhibited.

NONEXISTENT (string)
If a system account intentionally does not have a home directory that exists, this string can be provided in the /etc/passwd entry for the account to indicate this. The result is that pwck will not emit a spurious warning for this account.

OBSCURE_CHECKS_ENAB (boolean)
Enable additional checks upon password changes.

PASS_ALWAYS_WARN (boolean)
Warn about weak passwords (but still allow them) if you are root.

PASS_CHANGE_TRIES (number)
Maximum number of attempts to change password if rejected (too easy).

PASS_MAX_DAYS (number)
The maximum number of days a password may be used. If the password is older than this, a password change will be forced. If not specified, −1 will be assumed (which disables the restriction).

PASS_MIN_DAYS (number)
The minimum number of days allowed between password changes. Any password changes attempted sooner than this will be rejected. If not specified, 0 will be assumed (which disables the restriction).

PASS_WARN_AGE (number)
The number of days warning given before a password expires. A zero means warning is given only upon the day of expiration, a value of −1 means no warning is given. If not specified, no warning will be provided.

PASS_MAX_DAYS, **PASS_MIN_DAYS** and **PASS_WARN_AGE** are only used at the time of account creation. Any changes to these settings won't affect existing accounts.

PASS_MAX_LEN (number), **PASS_MIN_LEN** (number)
Number of significant characters in the password for crypt(). **PASS_MAX_LEN** is 8 by default. Don't change unless your crypt() is better. This is ignored if **MD5_CRYPT_ENAB** set to *yes*.

PORTTIME_CHECKS_ENAB (boolean)
Enable checking of time restrictions specified in /etc/porttime.

QUOTAS_ENAB (boolean)

Enable setting of resource limits from /etc/limits and ulimit, umask, and niceness from the user's passwd gecos field.

SHA_CRYPT_MIN_ROUNDS (number), **SHA_CRYPT_MAX_ROUNDS** (number)
When **ENCRYPT_METHOD** is set to *SHA256* or *SHA512*, this defines the number of SHA rounds used by the encryption algorithm by default (when the number of rounds is not specified on the command line).

With a lot of rounds, it is more difficult to brute force the password. But note also that more CPU resources will be needed to authenticate users.

If not specified, the libc will choose the default number of rounds (5000), which is orders of magnitude too low for modern hardware.

The values must be inside the 1000−999,999,999 range.

If only one of the **SHA_CRYPT_MIN_ROUNDS** or **SHA_CRYPT_MAX_ROUNDS** values is set, then this value will be used.

If **SHA_CRYPT_MIN_ROUNDS** > **SHA_CRYPT_MAX_ROUNDS**, the highest value will be used.

SULOG_FILE (string)
If defined, all su activity is logged to this file.

SU_NAME (string)
If defined, the command name to display when running "su −". For example, if this is defined as "su" then a "ps" will display the command is "−su". If not defined, then "ps" would display the name of the shell actually being run, e.g. something like "−sh".

SU_WHEEL_ONLY (boolean)
If *yes*, the user must be listed as a member of the first gid 0 group in /etc/group (called *root* on most Linux systems) to be able to **su** to uid 0 accounts. If the group doesn't exist or is empty, no one will be able to **su** to uid 0.

SUB_GID_MIN (number), **SUB_GID_MAX** (number), **SUB_GID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate group IDs) allocate **SUB_GID_COUNT** unused group IDs from the range **SUB_GID_MIN** to **SUB_GID_MAX** for each new user.

The default values for **SUB_GID_MIN**, **SUB_GID_MAX**, **SUB_GID_COUNT** are respectively 100000, 600100000 and 65536.

SUB_UID_MIN (number), **SUB_UID_MAX** (number), **SUB_UID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate user IDs) allocate **SUB_UID_COUNT** unused user IDs from the range **SUB_UID_MIN** to **SUB_UID_MAX** for each new user.

The default values for **SUB_UID_MIN**, **SUB_UID_MAX**, **SUB_UID_COUNT** are respectively 100000, 600100000 and 65536.

SYS_GID_MAX (number), **SYS_GID_MIN** (number)
Range of group IDs used for the creation of system groups by **useradd**, **groupadd**, or **newusers**.

The default value for **SYS_GID_MIN** (resp. **SYS_GID_MAX**) is 101 (resp. **GID_MIN**−1).

SYS_UID_MAX (number), **SYS_UID_MIN** (number)
Range of user IDs used for the creation of system users by **useradd** or **newusers**.

The default value for **SYS_UID_MIN** (resp. **SYS_UID_MAX**) is 101 (resp. **UID_MIN**−1).

SYSLOG_SG_ENAB (boolean)
Enable "syslog" logging of **sg** activity.

SYSLOG_SU_ENAB (boolean)

Enable "syslog" logging of **su** activity − in addition to sulog file logging.

TTYGROUP (string), **TTYPERM** (string)
The terminal permissions: the login tty will be owned by the **TTYGROUP** group, and the permissions will be set to **TTYPERM**.

TTYGROUP can be either the name of a group or a numeric group identifier.

If TTYGROUP is not defined, then the group ownership of the terminal is set to the user's primary group. If TTYPERM is not defined, then the permissions are set to *0600*.

If you have a **write** program which is "setgid" to a special group which owns the terminals, define TTYGROUP to the group number and TTYPERM to 0620. Otherwise leave TTYGROUP commented out and assign TTYPERM to either 622 or 600.

TTYTYPE_FILE (string)
If defined, file which maps tty line to TERM environment parameter. Each line of the file is in a format something like "vt100 tty01".

UID_MAX (number), **UID_MIN** (number)
Range of user IDs used for the creation of regular users by **useradd** or **newusers**.

The default value for **UID_MIN** (resp. **UID_MAX**) is 1000 (resp. 60000).

ULIMIT (number)
Default **ulimit** value.

UMASK (number)
The file mode creation mask is initialized to this value. If not specified, the mask will be initialized to 022.

useradd and **newusers** use this mask to set the mode of the home directory they create if **HOME_MODE** is not set.

It is also used by **login** to define users' initial umask. Note that this mask can be overridden by the user's GECOS line (if **QUOTAS_ENAB** is set) or by the specification of a limit with the *K* identifier in *limits*(5).

USERDEL_CMD (string)
If defined, this command is run when removing a user. It should remove any at/cron/print jobs etc. owned by the user to be removed (passed as the first argument).

The return code of the script is not taken into account.

Here is an example script, which removes the user's cron, at and print jobs:

```
#! /bin/sh
# Check for the required argument.
if [ $# != 1 ]; then
        echo "Usage: $0 username"
        exit 1
fi
# Remove cron jobs.
crontab −r −u $1
# Remove at jobs.
# Note that it will remove any jobs owned by the same UID,
# even if it was shared by a different username.
AT_SPOOL_DIR=/var/spool/cron/atjobs
find $AT_SPOOL_DIR −name "[^.]*" −type f −user $1 −delete \;
# Remove print jobs.
lprm $1
# All done.
```

```
exit 0
```

USERGROUPS_ENAB (boolean)
Enable setting of the umask group bits to be the same as owner bits (examples: 022 −> 002, 077 −> 007) for non−root users, if the uid is the same as gid, and username is the same as the primary group name.

If set to *yes*, **userdel** will remove the user's group if it contains no more members, and **useradd** will create by default a group with the name of the user.

CROSS REFERENCES

The following cross references show which programs in the shadow password suite use which parameters.

chfn
CHFN_AUTH CHFN_RESTRICT LOGIN_STRING

chgpasswd
ENCRYPT_METHOD MAX_MEMBERS_PER_GROUP MD5_CRYPT_ENAB SHA_CRYPT_MAX_ROUNDS SHA_CRYPT_MIN_ROUNDS

chpasswd
ENCRYPT_METHOD MD5_CRYPT_ENAB SHA_CRYPT_MAX_ROUNDS SHA_CRYPT_MIN_ROUNDS

chsh
CHSH_AUTH LOGIN_STRING

gpasswd
ENCRYPT_METHOD MAX_MEMBERS_PER_GROUP MD5_CRYPT_ENAB SHA_CRYPT_MAX_ROUNDS SHA_CRYPT_MIN_ROUNDS

groupadd
GID_MAX GID_MIN MAX_MEMBERS_PER_GROUP SYS_GID_MAX SYS_GID_MIN

groupdel
MAX_MEMBERS_PER_GROUP

groupmems
MAX_MEMBERS_PER_GROUP

groupmod
MAX_MEMBERS_PER_GROUP

grpck
MAX_MEMBERS_PER_GROUP

grpconv
MAX_MEMBERS_PER_GROUP

grpunconv
MAX_MEMBERS_PER_GROUP

lastlog
LASTLOG_UID_MAX

login
CONSOLE CONSOLE_GROUPS DEFAULT_HOME ENV_HZ ENV_PATH ENV_SUPATH ENV_TZ ENVIRON_FILE ERASECHAR FAIL_DELAY FAILLOG_ENAB FAKE_SHELL FTMP_FILE HUSHLOGIN_FILE ISSUE_FILE KILLCHAR LASTLOG_ENAB LASTLOG_UID_MAX LOGIN_RETRIES LOGIN_STRING LOGIN_TIMEOUT LOG_OK_LOGINS LOG_UNKFAIL_ENAB MAIL_CHECK_ENAB MAIL_DIR MAIL_FILE MOTD_FILE NOLOGINS_FILE PORTTIME_CHECKS_ENAB QUOTAS_ENAB TTYGROUP TTYPERM TTYTYPE_FILE ULIMIT UMASK USERGROUPS_ENAB

newgrp / sg
SYSLOG_SG_ENAB

newusers
: ENCRYPT_METHOD GID_MAX GID_MIN MAX_MEMBERS_PER_GROUP MD5_CRYPT_ENAB HOME_MODE PASS_MAX_DAYS PASS_MIN_DAYS PASS_WARN_AGE SHA_CRYPT_MAX_ROUNDS SHA_CRYPT_MIN_ROUNDS SUB_GID_COUNT SUB_GID_MAX SUB_GID_MIN SUB_UID_COUNT SUB_UID_MAX SUB_UID_MIN SYS_GID_MAX SYS_GID_MIN SYS_UID_MAX SYS_UID_MIN UID_MAX UID_MIN UMASK

passwd
: ENCRYPT_METHOD MD5_CRYPT_ENAB OBSCURE_CHECKS_ENAB PASS_ALWAYS_WARN PASS_CHANGE_TRIES PASS_MAX_LEN PASS_MIN_LEN SHA_CRYPT_MAX_ROUNDS SHA_CRYPT_MIN_ROUNDS

pwck
: PASS_MAX_DAYS PASS_MIN_DAYS PASS_WARN_AGE

pwconv
: PASS_MAX_DAYS PASS_MIN_DAYS PASS_WARN_AGE

su
: CONSOLE CONSOLE_GROUPS DEFAULT_HOME ENV_HZ ENVIRON_FILE ENV_PATH ENV_SUPATH ENV_TZ LOGIN_STRING MAIL_CHECK_ENAB MAIL_DIR MAIL_FILE QUOTAS_ENAB SULOG_FILE SU_NAME SU_WHEEL_ONLY SYSLOG_SU_ENAB USERGROUPS_ENAB

sulogin
: ENV_HZ ENV_TZ

useradd
: CREATE_HOME GID_MAX GID_MIN HOME_MODE LASTLOG_UID_MAX MAIL_DIR MAX_MEMBERS_PER_GROUP PASS_MAX_DAYS PASS_MIN_DAYS PASS_WARN_AGE SUB_GID_COUNT SUB_GID_MAX SUB_GID_MIN SUB_UID_COUNT SUB_UID_MAX SUB_UID_MIN SYS_GID_MAX SYS_GID_MIN SYS_UID_MAX SYS_UID_MIN UID_MAX UID_MIN UMASK

userdel
: MAIL_DIR MAIL_FILE MAX_MEMBERS_PER_GROUP USERDEL_CMD USERGROUPS_ENAB

usermod
: LASTLOG_UID_MAX MAIL_DIR MAIL_FILE MAX_MEMBERS_PER_GROUP

SEE ALSO

login(1), *passwd*(1), *su*(1), *passwd*(5), *shadow*(5), *pam*(8).

NAME

passwd − the password file

DESCRIPTION

/etc/passwd contains one line for each user account, with seven fields delimited by colons (":"). These fields are:

- login name
- optional encrypted password
- numerical user ID
- numerical group ID
- user name or comment field
- user home directory
- optional user command interpreter

If the *password* field is a lower−case "x", then the encrypted password is actually stored in the *shadow*(5) file instead; there *must* be a corresponding line in the /etc/shadow file, or else the user account is invalid.

The encrypted *password* field may be empty, in which case no password is required to authenticate as the specified login name. However, some applications which read the /etc/passwd file may decide not to permit *any* access at all if the *password* field is blank.

A *password* field which starts with an exclamation mark means that the password is locked. The remaining characters on the line represent the *password* field before the password was locked.

Refer to *crypt*(3) for details on how this string is interpreted.

If the password field contains some string that is not a valid result of *crypt*(3), for instance ! or *, the user will not be able to use a unix password to log in (but the user may log in the system by other means).

The comment field, also known as the gecos field, is used by various system utilities, such as *finger*(1). The use of an ampersand here will be replaced by the capitalised login name when the field is used or displayed by such system utilities.

The home directory field provides the name of the initial working directory. The **login** program uses this information to set the value of the **$HOME** environmental variable.

The command interpreter field provides the name of the user's command language interpreter, or the name of the initial program to execute. The **login** program uses this information to set the value of the **$SHELL** environmental variable. If this field is empty, it defaults to the value /bin/sh.

FILES

/etc/passwd
: User account information.

/etc/shadow
: optional encrypted password file

/etc/passwd−
: Backup file for /etc/passwd.

  Note that this file is used by the tools of the shadow toolsuite, but not by all user and password management tools.

SEE ALSO

crypt(3), *getent*(1), *getpwnam*(3), *login*(1), *passwd*(1), *pwck*(8), *pwconv*(8), *pwunconv*(8), *shadow*(5), *su*(1), *sulogin*(8).

NAME

porttime − port access time file

DESCRIPTION

porttime contains a list of tty devices, user names, and permitted login times.

Each entry consists of three colon separated fields. The first field is a comma separated list of tty devices, or an asterisk to indicate that all tty devices are matched by this entry. The second field is a comma separated list of user names, or an asterisk to indicated that all user names are matched by this entry. The third field is a comma separated list of permitted access times.

Each access time entry consists of zero or more days of the week, abbreviated *Su*, *Mo*, *Tu*, *We*, *Th*, *Fr*, and *Sa*, followed by a pair of times separated by a hyphen. The abbreviation *Wk* may be used to represent Monday thru Friday, and *Al* may be used to indicate every day. If no days are given, *Al* is assumed.

EXAMPLES

The following entry allows access to user **jfh** on every port during weekdays from 9am to 5pm.

*:jfh:Wk0900−1700

The following entries allow access only to the users *root* and *oper* on /dev/console at any time. This illustrates how the /etc/porttime file is an ordered list of access times. Any other user would match the second entry which does not permit access at any time.

```
console:root,oper:Al0000−2400
console:*:
```

The following entry allows access for the user *games* on any port during non−working hours.

*:games:Wk1700−0900,SaSu0000−2400

FILES

/etc/porttime
: File containing port access.

SEE ALSO

login(1).

NAME

shadow − shadowed password file

DESCRIPTION

shadow is a file which contains the password information for the system's accounts and optional aging information.

This file must not be readable by regular users if password security is to be maintained.

Each line of this file contains 9 fields, separated by colons (“:”), in the following order:

login name
It must be a valid account name, which exist on the system.

encrypted password
This field may be empty, in which case no passwords are required to authenticate as the specified login name. However, some applications which read the /etc/shadow file may decide not to permit any access at all if the password field is empty.

A password field which starts with an exclamation mark means that the password is locked. The remaining characters on the line represent the password field before the password was locked.

Refer to *crypt*(3) for details on how this string is interpreted.

If the password field contains some string that is not a valid result of *crypt*(3), for instance ! or *, the user will not be able to use a unix password to log in (but the user may log in the system by other means).

date of last password change
The date of the last password change, expressed as the number of days since Jan 1, 1970 00:00 UTC.

The value 0 has a special meaning, which is that the user should change her password the next time she will log in the system.

An empty field means that password aging features are disabled.

minimum password age
The minimum password age is the number of days the user will have to wait before she will be allowed to change her password again.

An empty field and value 0 mean that there is no minimum password age.

maximum password age
The maximum password age is the number of days after which the user will have to change her password.

After this number of days is elapsed, the password may still be valid. The user should be asked to change her password the next time she will log in.

An empty field means that there are no maximum password age, no password warning period, and no password inactivity period (see below).

If the maximum password age is lower than the minimum password age, the user cannot change her password.

password warning period
The number of days before a password is going to expire (see the maximum password age above) during which the user should be warned.

An empty field and value 0 mean that there are no password warning period.

password inactivity period
The number of days after a password has expired (see the maximum password age above) during which the password should still be accepted (and the user should update her password during the

next login).

After expiration of the password and this expiration period is elapsed, no login is possible for the user. The user should contact her administrator.

An empty field means that there are no enforcement of an inactivity period.

account expiration date
The date of expiration of the account, expressed as the number of days since Jan 1, 1970 00:00 UTC.

Note that an account expiration differs from a password expiration. In case of an account expiration, the user shall not be allowed to login. In case of a password expiration, the user is not allowed to login using her password.

An empty field means that the account will never expire.

The value 0 should not be used as it is interpreted as either an account with no expiration, or as an expiration on Jan 1, 1970.

reserved field
This field is reserved for future use.

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/shadow−
Backup file for /etc/shadow.

Note that this file is used by the tools of the shadow toolsuite, but not by all user and password management tools.

SEE ALSO

chage(1), *login*(1), *passwd*(1), *passwd*(5), *pwck*(8), *pwconv*(8), *pwunconv*(8), *su*(1), *sulogin*(8).

NAME

suauth − detailed su control file

SYNOPSIS

/etc/suauth

DESCRIPTION

The file /etc/suauth is referenced whenever the su command is called. It can change the behaviour of the su command, based upon:

1) the user su is targeting

2) the user executing the su command (or any groups he might be a member of)

The file is formatted like this, with lines starting with a # being treated as comment lines and ignored;

to−id:from−id:ACTION

Where to−id is either the word *ALL*, a list of usernames delimited by "," or the words *ALL EXCEPT* followed by a list of usernames delimited by ",".

from−id is formatted the same as to−id except the extra word *GROUP* is recognized. *ALL EXCEPT GROUP* is perfectly valid too. Following *GROUP* appears one or more group names, delimited by ",". It is not sufficient to have primary group id of the relevant group, an entry in **/etc/group**(5) is necessary.

Action can be one only of the following currently supported options.

DENY
: The attempt to su is stopped before a password is even asked for.

NOPASS
: The attempt to su is automatically successful; no password is asked for.

OWNPASS
: For the su command to be successful, the user must enter his or her own password. They are told this.

Note there are three separate fields delimited by a colon. No whitespace must surround this colon. Also note that the file is examined sequentially line by line, and the first applicable rule is used without examining the file further. This makes it possible for a system administrator to exercise as fine control as he or she wishes.

EXAMPLE

```
# sample /etc/suauth file
#
# A couple of privileged usernames may
# su to root with their own password.
#
root:chris,birddog:OWNPASS
#
# Anyone else may not su to root unless in
# group wheel. This is how BSD does things.
#
root:ALL EXCEPT GROUP wheel:DENY
#
# Perhaps terry and birddog are accounts
# owned by the same person.
# Access can be arranged between them
# with no password.
#
terry:birddog:NOPASS
```

```
birddog:terry:NOPASS
#
```

FILES

/etc/suauth

BUGS

There could be plenty lurking. The file parser is particularly unforgiving about syntax errors, expecting no spurious whitespace (apart from beginning and end of lines), and a specific token delimiting different things.

DIAGNOSTICS

An error parsing the file is reported using *syslogd*(8) as level ERR on facility AUTH.

SEE ALSO

su(1).

NAME

subgid − the configuration for subordinate group ids

DESCRIPTION

Subgid authorizes a group id to map ranges of group ids from its namespace into child namespaces.

The delegation of the subordinate gids can be configured via the *subid* field in /etc/nsswitch.conf file. Only one value can be set as the delegation source. Setting this field to *files* configures the delegation of gids to /etc/subgid. Setting any other value treats the delegation as a plugin following with a name of the form *libsubid_$value.so*. If the value or plugin is missing, then the subordinate gid delegation falls back to *files*.

Note, that **groupadd** will only create entries in /etc/subgid if subid delegation is managed via subid files.

LOCAL SUBORDINATE DELEGATION

Each line in /etc/subgid contains a user name and a range of subordinate group ids that user is allowed to use. This is specified with three fields delimited by colons (“:”). These fields are:

- login name or UID
- numerical subordinate group ID
- numerical subordinate group ID count

This file specifies the group IDs that ordinary users can use, with the **newgidmap** command, to configure gid mapping in a user namespace.

Multiple ranges may be specified per user.

When large number of entries (10000−100000 or more) are defined in /etc/subgid, parsing performance penalty will become noticeable. In this case it is recommended to use UIDs instead of login names. Benchmarks have shown speed−ups up to 20x.

FILES

/etc/subgid
: Per user subordinate group IDs.

/etc/subgid−
: Backup file for /etc/subgid.

SEE ALSO

login.defs(5), *newgidmap*(1), *newuidmap*(1), *newusers*(8), *subuid*(5), *useradd*(8), *userdel*(8), *usermod*(8), *user_namespaces*(7).

NAME

subuid − the configuration for subordinate user ids

DESCRIPTION

Subuid authorizes a user id to map ranges of user ids from its namespace into child namespaces.

The delegation of the subordinate uids can be configured via the *subid* field in /etc/nsswitch.conf file. Only one value can be set as the delegation source. Setting this field to *files* configures the delegation of uids to /etc/subuid. Setting any other value treats the delegation as a plugin following with a name of the form *libsubid_$value.so*. If the value or plugin is missing, then the subordinate uid delegation falls back to *files*.

Note, that **useradd** will only create entries in /etc/subuid if subid delegation is managed via subid files.

LOCAL SUBORDINATE DELEGATION

Each line in /etc/subuid contains a user name and a range of subordinate user ids that user is allowed to use. This is specified with three fields delimited by colons (“:”). These fields are:

- login name or UID
- numerical subordinate user ID
- numerical subordinate user ID count

This file specifies the user IDs that ordinary users can use, with the **newuidmap** command, to configure uid mapping in a user namespace.

Multiple ranges may be specified per user.

When large number of entries (10000−100000 or more) are defined in /etc/subuid, parsing performance penalty will become noticeable. In this case it is recommended to use UIDs instead of login names. Benchmarks have shown speed−ups up to 20x.

FILES

/etc/subuid
: Per user subordinate user IDs.

/etc/subuid−
: Backup file for /etc/subuid.

SEE ALSO

login.defs(5), *newgidmap*(1), *newuidmap*(1), *newusers*(8), *subgid*(5), *useradd*(8), *userdel*(8), *usermod*(8), *user_namespaces*(7).

NAME

chgpasswd − update group passwords in batch mode

SYNOPSIS

chgpasswd [*options*]

DESCRIPTION

The **chgpasswd** command reads a list of group name and password pairs from standard input and uses this information to update a set of existing groups. Each line is of the format:

group_name:*password*

By default the supplied password must be in clear−text, and is encrypted by **chgpasswd**.

The default encryption algorithm can be defined for the system with the **ENCRYPT_METHOD** variable of /etc/login.defs, and can be overwritten with the **−e**, **−m**, or **−c** options.

This command is intended to be used in a large system environment where many accounts are created at a single time.

OPTIONS

The options which apply to the **chgpasswd** command are:

−c, **−−crypt−method**
Use the specified method to encrypt the passwords.

The available methods are *DES*, *MD5*, *SHA256*, *SHA512* and *NONE* if your libc supports these methods.

−e, **−−encrypted**
Supplied passwords are in encrypted form.

−h, **−−help**
Display help message and exit.

−m, **−−md5**
Use MD5 encryption instead of DES when the supplied passwords are not encrypted.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−s, **−−sha−rounds**
Use the specified number of rounds to encrypt the passwords.

You can only use this option with crypt method: *SHA256 SHA512*

By default, the number of rounds for SHA256 or SHA512 is defined by the SHA_CRYPT_MIN_ROUNDS and SHA_CRYPT_MAX_ROUNDS variables in /etc/login.defs.

A minimal value of 1000 and a maximal value of 999,999,999 will be enforced for SHA256 and SHA512. The default number of rounds is 5000.

CAVEATS

Remember to set permissions or umask to prevent readability of unencrypted files by other users.

You should make sure the passwords and the encryption method respect the system's password policy.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

ENCRYPT_METHOD (string)
This defines the system default encryption algorithm for encrypting passwords (if no algorithm are specified on the command line).

It can take one of these values: *DES* (default), *MD5*, *SHA256*, *SHA512*. MD5 and DES should not be used for new hashes, see crypt(5) for recommendations.

Note: this parameter overrides the **MD5_CRYPT_ENAB** variable.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

MD5_CRYPT_ENAB (boolean)
Indicate if passwords must be encrypted using the MD5−based algorithm. If set to *yes*, new passwords will be encrypted using the MD5−based algorithm compatible with the one used by recent releases of FreeBSD. It supports passwords of unlimited length and longer salt strings. Set to *no* if you need to copy encrypted passwords to other systems which don't understand the new algorithm. Default is *no*.

This variable is superseded by the **ENCRYPT_METHOD** variable or by any command line option used to configure the encryption algorithm.

This variable is deprecated. You should use **ENCRYPT_METHOD**.

SHA_CRYPT_MIN_ROUNDS (number), **SHA_CRYPT_MAX_ROUNDS** (number)
When **ENCRYPT_METHOD** is set to *SHA256* or *SHA512*, this defines the number of SHA rounds used by the encryption algorithm by default (when the number of rounds is not specified on the command line).

With a lot of rounds, it is more difficult to brute force the password. But note also that more CPU resources will be needed to authenticate users.

If not specified, the libc will choose the default number of rounds (5000), which is orders of magnitude too low for modern hardware.

The values must be inside the 1000−999,999,999 range.

If only one of the **SHA_CRYPT_MIN_ROUNDS** or **SHA_CRYPT_MAX_ROUNDS** values is set, then this value will be used.

If **SHA_CRYPT_MIN_ROUNDS** > **SHA_CRYPT_MAX_ROUNDS**, the highest value will be used.

FILES

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

/etc/login.defs
Shadow password suite configuration.

SEE ALSO

gpasswd(1), *groupadd*(8), *login.defs*(5).

NAME

chpasswd − update passwords in batch mode

SYNOPSIS

chpasswd [*options*]

DESCRIPTION

The **chpasswd** command reads a list of user name and password pairs from standard input and uses this information to update a group of existing users. Each line is of the format:

user_name:*password*

By default the passwords must be supplied in clear−text, and are encrypted by **chpasswd**. Also the password age will be updated, if present.

The default encryption algorithm can be defined for the system with the **ENCRYPT_METHOD** or **MD5_CRYPT_ENAB** variables of /etc/login.defs, and can be overwritten with the **−e**, **−m**, or **−c** options.

chpasswd first updates all the passwords in memory, and then commits all the changes to disk if no errors occurred for any user.

This command is intended to be used in a large system environment where many accounts are created at a single time.

OPTIONS

The options which apply to the **chpasswd** command are:

−c, **−−crypt−method** *METHOD*
Use the specified method to encrypt the passwords.

The available methods are *DES*, *MD5*, *SHA256*, *SHA512* and *NONE* if your libc supports these methods.

By default (if none of the **−c**, **−m**, or **−e** options are specified), the encryption method is defined by the **ENCRYPT_METHOD** or **MD5_CRYPT_ENAB** variables of /etc/login.defs.

−e, **−−encrypted**
Supplied passwords are in encrypted form.

−h, **−−help**
Display help message and exit.

−m, **−−md5**
Use MD5 encryption instead of DES when the supplied passwords are not encrypted.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes to configuration files under the root filesystem found under the directory *PREFIX_DIR*. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−s, **−−sha−rounds** *ROUNDS*
Use the specified number of rounds to encrypt the passwords.

You can only use this option with crypt method: *SHA256 SHA512*

By default, the number of rounds for SHA256 or SHA512 is defined by the SHA_CRYPT_MIN_ROUNDS and SHA_CRYPT_MAX_ROUNDS variables in /etc/login.defs.

A minimal value of 1000 and a maximal value of 999,999,999 will be enforced for SHA256 and SHA512. The default number of rounds is 5000.

CAVEATS

Remember to set permissions or umask to prevent readability of unencrypted files by other users.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

ENCRYPT_METHOD (string)

This defines the system default encryption algorithm for encrypting passwords (if no algorithm are specified on the command line).

It can take one of these values: *DES* (default), *MD5*, *SHA256*, *SHA512*. MD5 and DES should not be used for new hashes, see crypt(5) for recommendations.

Note: this parameter overrides the **MD5_CRYPT_ENAB** variable.

MD5_CRYPT_ENAB (boolean)

Indicate if passwords must be encrypted using the MD5−based algorithm. If set to *yes*, new passwords will be encrypted using the MD5−based algorithm compatible with the one used by recent releases of FreeBSD. It supports passwords of unlimited length and longer salt strings. Set to *no* if you need to copy encrypted passwords to other systems which don't understand the new algorithm. Default is *no*.

This variable is superseded by the **ENCRYPT_METHOD** variable or by any command line option used to configure the encryption algorithm.

This variable is deprecated. You should use **ENCRYPT_METHOD**.

SHA_CRYPT_MIN_ROUNDS (number), **SHA_CRYPT_MAX_ROUNDS** (number)

When **ENCRYPT_METHOD** is set to *SHA256* or *SHA512*, this defines the number of SHA rounds used by the encryption algorithm by default (when the number of rounds is not specified on the command line).

With a lot of rounds, it is more difficult to brute force the password. But note also that more CPU resources will be needed to authenticate users.

If not specified, the libc will choose the default number of rounds (5000), which is orders of magnitude too low for modern hardware.

The values must be inside the 1000−999,999,999 range.

If only one of the **SHA_CRYPT_MIN_ROUNDS** or **SHA_CRYPT_MAX_ROUNDS** values is set, then this value will be used.

If **SHA_CRYPT_MIN_ROUNDS** > **SHA_CRYPT_MAX_ROUNDS**, the highest value will be used.

FILES

/etc/passwd

User account information.

/etc/shadow

Secure user account information.

/etc/login.defs

Shadow password suite configuration.

SEE ALSO

passwd(1), *newusers*(8), *login.defs*(5), *useradd*(8).

NAME

faillog − display faillog records or set login failure limits

SYNOPSIS

faillog [*options*]

DESCRIPTION

faillog displays the contents of the failure log database (/var/log/faillog). It can also set the failure counters and limits. When **faillog** is run without arguments, it only displays the faillog records of the users who had a login failure.

OPTIONS

The options which apply to the **faillog** command are:

−a, **−−all**
Display (or act on) faillog records for all users having an entry in the faillog database.

The range of users can be restricted with the **−u** option.

In display mode, this is still restricted to existing users but forces the display of the faillog entries even if they are empty.

With the **−l**, **−m**, **−r**, **−t** options, the users' records are changed, even if the user does not exist on the system. This is useful to reset records of users that have been deleted or to set a policy in advance for a range of users.

−h, **−−help**
Display help message and exit.

−l, **−−lock−secs** *SEC*
Lock account for *SEC* seconds after failed login.

Write access to /var/log/faillog is required for this option.

−m, **−−maximum** *MAX*
Set the maximum number of login failures after the account is disabled to *MAX*.

Selecting a *MAX* value of 0 has the effect of not placing a limit on the number of failed logins.

The maximum failure count should always be 0 for *root* to prevent a denial of services attack against the system.

Write access to /var/log/faillog is required for this option.

−r, **−−reset**
Reset the counters of login failures.

Write access to /var/log/faillog is required for this option.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−t, **−−time** *DAYS*
Display faillog records more recent than *DAYS*.

−u, **−−user** *LOGIN|RANGE*
Display faillog record or maintains failure counters and limits (if used with **−l**, **−m** or **−r** options) only for the specified user(s).

The users can be specified by a login name, a numerical user ID, or a *RANGE* of users. This *RANGE* of users can be specified with a min and max values (*UID_MIN−UID_MAX*), a max value (*−UID_MAX*), or a min value (*UID_MIN−*).

When none of the **−l**, **−m**, or **−r** options are used, **faillog** displays the faillog record of the specified user(s).

CAVEATS

faillog only prints out users with no successful login since the last failure. To print out a user who has had a successful login since their last failure, you must explicitly request the user with the **−u** flag, or print out all users with the **−a** flag.

FILES

/var/log/faillog
: Failure logging file.

SEE ALSO

login(1), *faillog*(5).

NAME

groupadd − create a new group

SYNOPSIS

groupadd [*OPTIONS*] *NEWGROUP*

DESCRIPTION

The **groupadd** command creates a new group account using the values specified on the command line plus the default values from the system. The new group will be entered into the system files as needed.

Groupnames may contain only lower and upper case letters, digits, underscores, or dashes. They can end with a dollar sign. Dashes are not allowed at the beginning of the groupname. Fully numeric groupnames and groupnames . or .. are also disallowed.

Groupnames may only be up to 32 characters long.

OPTIONS

The options which apply to the **groupadd** command are:

−f, **−−force**
This option causes the command to simply exit with success status if the specified group already exists. When used with **−g**, and the specified GID already exists, another (unique) GID is chosen (i.e. **−g** is turned off).

−g, **−−gid** *GID*
The numerical value of the group's ID. *GID* must be unique, unless the **−o** option is used. The value must be non−negative. The default is to use the smallest ID value greater than or equal to **GID_MIN** and greater than every other group.

See also the **−r** option and the **GID_MAX** description.

−h, **−−help**
Display help message and exit.

−K, **−−key** *KEY=VALUE*
Overrides /etc/login.defs defaults (GID_MIN, GID_MAX and others). Multiple **−K** options can be specified.

Example: **−K** *GID_MIN=100* **−K** *GID_MAX=499*

Note: **−K** *GID_MIN=10*,*GID_MAX=499* doesn't work yet.

−o, **−−non−unique**
permits the creation of a group with an already used numerical ID. As a result, for this *GID*, the mapping towards group *NEWGROUP* may not be unique.

−p, **−−password** *PASSWORD*
defines an initial password for the group account. PASSWORD is expected to be encrypted, as returned by **crypt** (3).

Without this option, the group account will be locked and with no password defined, i.e. a single exclamation mark in the respective field of ths system account file /etc/group or /etc/gshadow.

Note: This option is not recommended because the password (or encrypted password) will be visible by users listing the processes.

You should make sure the password respects the system's password policy.

−r, **−−system**
Create a system group.

The numeric identifiers of new system groups are chosen in the **SYS_GID_MIN**−**SYS_GID_MAX** range, defined in login.defs, instead of **GID_MIN**−**GID_MAX**.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes to configuration files under the root filesystem found under the directory *PREFIX_DIR*. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−U, **−−users**
A list of usernames to add as members of the group.

The default behavior (if the **−g**, **−N**, and **−U** options are not specified) is defined by the **USERGROUPS_ENAB** variable in /etc/login.defs.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

GID_MAX (number), **GID_MIN** (number)
Range of group IDs used for the creation of regular groups by **useradd**, **groupadd**, or **newusers**.

The default value for **GID_MIN** (resp. **GID_MAX**) is 1000 (resp. 60000).

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

SYS_GID_MAX (number), **SYS_GID_MIN** (number)
Range of group IDs used for the creation of system groups by **useradd**, **groupadd**, or **newusers**.

The default value for **SYS_GID_MIN** (resp. **SYS_GID_MAX**) is 101 (resp. **GID_MIN**−1).

FILES

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

/etc/login.defs
Shadow password suite configuration.

CAVEATS

You may not add a NIS or LDAP group. This must be performed on the corresponding server.

If the groupname already exists in an external group database such as NIS or LDAP, **groupadd** will deny the group creation request.

EXIT VALUES

The **groupadd** command exits with the following values:

0
success

2
invalid command syntax

3
invalid argument to option

4
GID is already used (when called without **−o**)

9
group name is already used

10
can't update group file

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *gpasswd*(8), *groupdel*(8), *groupmod*(8), *login.defs*(5), *useradd*(8), *userdel*(8), *usermod*(8).

NAME

groupdel − delete a group

SYNOPSIS

groupdel [*options*] *GROUP*

DESCRIPTION

The **groupdel** command modifies the system account files, deleting all entries that refer to *GROUP*. The named group must exist.

OPTIONS

The options which apply to the **groupdel** command are:

−f, **−−force**
This option forces the removal of the group, even if there's some user having the group as the primary one.

−h, **−−help**
Display help message and exit.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes in the *PREFIX_DIR* directory and use the configuration files from the *PREFIX_DIR* directory. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

CAVEATS

You may not remove the primary group of any existing user. You must remove the user before you remove the group.

You should manually check all file systems to ensure that no files remain owned by this group.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

FILES

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

EXIT VALUES

The **groupdel** command exits with the following values:

0
success

2

invalid command syntax

6
specified group doesn't exist

8
can't remove user's primary group

10
can't update group file

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *gpasswd*(8), *groupadd*(8), *groupmod*(8), *useradd*(8), *userdel*(8), *usermod*(8).

NAME

groupmems − administer members of a user's primary group

SYNOPSIS

groupmems −a *user_name* | −d *user_name* | [−g *group_name*] | −l | −p

DESCRIPTION

The **groupmems** command allows a user to administer their own group membership list without the requirement of superuser privileges. The **groupmems** utility is for systems that configure its users to be in their own name sake primary group (i.e., guest / guest).

Only the superuser, as administrator, can use **groupmems** to alter the memberships of other groups.

OPTIONS

The options which apply to the **groupmems** command are:

−a, **−−add** *user_name*
Add a user to the group membership list.

If the /etc/gshadow file exist, and the group has no entry in the /etc/gshadow file, a new entry will be created.

−d, **−−delete** *user_name*
Delete a user from the group membership list.

If the /etc/gshadow file exist, the user will be removed from the list of members and administrators of the group.

If the /etc/gshadow file exist, and the group has no entry in the /etc/gshadow file, a new entry will be created.

−g, **−−group** *group_name*
The superuser can specify which group membership list to modify.

−h, **−−help**
Display help message and exit.

−l, **−−list**
List the group membership list.

−p, **−−purge**
Purge all users from the group membership list.

If the /etc/gshadow file exist, and the group has no entry in the /etc/gshadow file, a new entry will be created.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

SETUP

The **groupmems** executable should be in mode 2710 as user *root* and in group *groups*. The system administrator can add users to group *groups* to allow or disallow them using the **groupmems** utility to manage their own group membership list.

```
$ groupadd −r groups
$ chmod 2710 groupmems
$ chown root:groups groupmems
$ groupmems −g groups −a gk4
```

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is

started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

FILES

/etc/group
: Group account information.

/etc/gshadow
: secure group account information

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *groupadd*(8), *groupdel*(8), *useradd*(8), *userdel*(8), *usermod*(8).

NAME

groupmod − modify a group definition on the system

SYNOPSIS

groupmod [*options*] *GROUP*

DESCRIPTION

The **groupmod** command modifies the definition of the specified *GROUP* by modifying the appropriate entry in the group database.

OPTIONS

The options which apply to the **groupmod** command are:

−a, **−−append** *GID*
If group members are specified with −U, append them to the existing member list, rather than replacing it.

−g, **−−gid** *GID*
The group ID of the given *GROUP* will be changed to *GID*.

The value of *GID* must be a non−negative decimal integer. This value must be unique, unless the **−o** option is used.

Users who use the group as primary group will be updated to keep the group as their primary group.

Any files that have the old group ID and must continue to belong to *GROUP*, must have their group ID changed manually.

No checks will be performed with regard to the **GID_MIN**, **GID_MAX**, **SYS_GID_MIN**, or **SYS_GID_MAX** from /etc/login.defs.

−h, **−−help**
Display help message and exit.

−n, **−−new−name** *NEW_GROUP*
The name of the group will be changed from *GROUP* to *NEW_GROUP* name.

−o, **−−non−unique**
When used with the **−g** option, allow to change the group *GID* to a non−unique value.

−p, **−−password** *PASSWORD*
The encrypted password, as returned by *crypt*(3).

Note: This option is not recommended because the password (or encrypted password) will be visible by users listing the processes.

You should make sure the password respects the system's password policy.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes in the *PREFIX_DIR* directory and use the configuration files from the *PREFIX_DIR* directory. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−U, **−−users**
A list of usernames to add as members of the group.

The default behavior (if the **−g**, **−N**, and **−U** options are not specified) is defined by the **USERGROUPS_ENAB** variable in /etc/login.defs.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

MAX_MEMBERS_PER_GROUP (number)
: Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

  The default value is 0, meaning that there are no limits in the number of members in a group.

  This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

  If you need to enforce such limit, you can use 25.

  Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

FILES

/etc/group
: Group account information.

/etc/gshadow
: Secure group account information.

/etc/login.defs
: Shadow password suite configuration.

/etc/passwd
: User account information.

EXIT VALUES

The **groupmod** command exits with the following values:

0
: E_SUCCESS: success

2
: E_USAGE: invalid command syntax

3
: E_BAD_ARG: invalid argument to option

4
: E_GID_IN_USE: group id already in use

6
: E_NOTFOUND: specified group doesn't exist

9
: E_NAME_IN_USE: group name already in use

10
: E_GRP_UPDATE: can't update group file

11
: E_CLEANUP_SERVICE: can't setup cleanup service

12
: E_PAM_USERNAME: can't determine your username for use with pam

13
: E_PAM_ERROR: pam returned an error, see syslog facility id groupmod for the PAM error message

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *gpasswd*(8), *groupadd*(8), *groupdel*(8), *login.defs*(5), *useradd*(8), *userdel*(8), *usermod*(8).

NAME

grpck − verify integrity of group files

SYNOPSIS

grpck [options] [*group* [*shadow*]]

DESCRIPTION

The **grpck** command verifies the integrity of the groups information. It checks that all entries in /etc/group and /etc/gshadow have the proper format and contain valid data. The user is prompted to delete entries that are improperly formatted or which have other uncorrectable errors.

Checks are made to verify that each entry has:

- the correct number of fields
- a unique and valid group name
- a valid group identifier (/etc/group only)
- a valid list of members and administrators
- a corresponding entry in the /etc/gshadow file (respectively /etc/group for the gshadow checks)

The checks for correct number of fields and unique group name are fatal. If an entry has the wrong number of fields, the user will be prompted to delete the entire line. If the user does not answer affirmatively, all further checks are bypassed. An entry with a duplicated group name is prompted for deletion, but the remaining checks will still be made. All other errors are warnings and the user is encouraged to run the **groupmod** command to correct the error.

The commands which operate on the /etc/group and /etc/gshadow files are not able to alter corrupted or duplicated entries. **grpck** should be used in those circumstances to remove the offending entries.

OPTIONS

The **−r** and **−s** options cannot be combined.

The options which apply to the **grpck** command are:

−h, **−−help**
Display help message and exit.

−r, **−−read−only**
Execute the **grpck** command in read−only mode. This causes all questions regarding changes to be answered *no* without user intervention.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−s, **−−sort**
Sort entries in /etc/group and /etc/gshadow by GID.

−S, **−−silence−warnings**
Suppress more controversial warnings, in particular warnings about inconsistency between group members listed in /etc/group and /etc/ghadow.

By default, **grpck** operates on /etc/group and /etc/gshadow. The user may select alternate files with the *group* and *shadow* parameters.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

FILES

/etc/group
: Group account information.

/etc/gshadow
: Secure group account information.

/etc/passwd
: User account information.

EXIT VALUES

The **grpck** command exits with the following values:

0
: success

1
: invalid command syntax

2
: one or more bad group entries

3
: can't open group files

4
: can't lock group files

5
: can't update group files

SEE ALSO

group(5), *groupmod*(8), *gshadow*(5), *passwd*(5), *pwck*(8), *shadow*(5).

NAME

lastlog − reports the most recent login of all users or of a given user

SYNOPSIS

lastlog [*options*]

DESCRIPTION

lastlog formats and prints the contents of the last login log /var/log/lastlog file. The *login−name*, *port*, and *last login time* will be printed. The default (no flags) causes lastlog entries to be printed, sorted by their order in /etc/passwd.

OPTIONS

The options which apply to the **lastlog** command are:

−b, **−−before** *DAYS*
Print only lastlog records older than *DAYS*.

−C, **−−clear**
Clear lastlog record of a user. This option can be used only together with **−u** (**−−user**)).

−h, **−−help**
Display help message and exit.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−S, **−−set**
Set lastlog record of a user to the current time. This option can be used only together with **−u** (**−−user**)).

−t, **−−time** *DAYS*
Print the lastlog records more recent than *DAYS*.

−u, **−−user** *LOGIN|RANGE*
Print the lastlog record of the specified user(s).

The users can be specified by a login name, a numerical user ID, or a *RANGE* of users. This *RANGE* of users can be specified with a min and max values (*UID_MIN−UID_MAX*), a max value (*−UID_MAX*), or a min value (*UID_MIN−*).

If the user has never logged in the message *** Never logged in*** will be displayed instead of the port and time.

Only the entries for the current users of the system will be displayed. Other entries may exist for users that were deleted previously.

NOTE

The lastlog file is a database which contains info on the last login of each user. You should not rotate it. It is a sparse file, so its size on the disk is usually much smaller than the one shown by "**ls −l**" (which can indicate a really big file if you have in passwd users with a high UID). You can display its real size with "**ls −s**".

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

LASTLOG_UID_MAX (number)
Highest user ID number for which the lastlog entries should be updated. As higher user IDs are usually tracked by remote user identity and authentication services there is no need to create a huge sparse lastlog file for them.

No **LASTLOG_UID_MAX** option present in the configuration means that there is no user ID limit for writing lastlog entries.

FILES

/var/log/lastlog
Database times of previous user logins.

CAVEATS

Large gaps in UID numbers will cause the lastlog program to run longer with no output to the screen (i.e. if in lastlog database there is no entries for users with UID between 170 and 800 lastlog will appear to hang as it processes entries with UIDs 171−799).

Having high UIDs can create problems when handling the <term> /var/log/lastlog</term> with external tools. Although the actual file is sparse and does not use too much space, certain applications are not designed to identify sparse files by default and may require a specific option to handle them.

NAME

logoutd − enforce login time restrictions

SYNOPSIS

logoutd

DESCRIPTION

logoutd enforces the login time and port restrictions specified in /etc/porttime. **logoutd** should be started from /etc/rc. The /var/run/utmp file is scanned periodically and each user name is checked to see if the named user is permitted on the named port at the current time. Any login session which is violating the restrictions in /etc/porttime is terminated.

FILES

/etc/porttime
: File containing port access.

/var/run/utmp
: List of current login sessions.

NAME

newusers − update and create new users in batch

SYNOPSIS

newusers [*options*] [*file*]

DESCRIPTION

The **newusers** command reads a *file* (or the standard input by default) and uses this information to update a set of existing users or to create new users. Each line is in the same format as the standard password file (see *passwd*(5)) with the exceptions explained below:

pw_name:pw_passwd:pw_uid:pw_gid:pw_gecos:pw_dir:pw_shell

pw_name
: This is the name of the user.

  It can be the name of a new user or the name of an existing user (or a user created before by **newusers**). In case of an existing user, the user's information will be changed, otherwise a new user will be created.

pw_passwd
: This field will be encrypted and used as the new value of the encrypted password.

pw_uid
: This field is used to define the UID of the user.

  If the field is empty, a new (unused) UID will be defined automatically by **newusers**.

  If this field contains a number, this number will be used as the UID.

  If this field contains the name of an existing user (or the name of a user created before by **newusers**), the UID of the specified user will be used.

  If the UID of an existing user is changed, the files ownership of the user's file should be fixed manually.

pw_gid
: This field is used to define the primary group ID for the user.

  If this field contains the name of an existing group (or a group created before by **newusers**), the GID of this group will be used as the primary group ID for the user.

  If this field is a number, this number will be used as the primary group ID of the user. If no groups exist with this GID, a new group will be created with this GID, and the name of the user.

  If this field is empty, a new group will be created with the name of the user and a GID will be automatically defined by **newusers** to be used as the primary group ID for the user and as the GID for the new group.

  If this field contains the name of a group which does not exist (and was not created before by **newusers**), a new group will be created with the specified name and a GID will be automatically defined by **newusers** to be used as the primary group ID for the user and GID for the new group.

pw_gecos
: This field is copied in the GECOS field of the user.

pw_dir
: This field is used to define the home directory of the user.

  If this field does not specify an existing directory, the specified directory is created, with ownership set to the user being created or updated and its primary group. Note that *newusers does not create parent directories* of the new user's home directory. The newusers command will fail to create the home directory if the parent directories do not exist, and will send a message to stderr

informing the user of the failure. The newusers command will not halt or return a failure to the calling shell if it fails to create the home directory, it will continue to process the batch of new users specified.

If the home directory of an existing user is changed, **newusers** does not move or copy the content of the old directory to the new location. This should be done manually.

pw_shell
This field defines the shell of the user. No checks are performed on this field.

newusers first tries to create or change all the specified users, and then write these changes to the user or group databases. If an error occurs (except in the final writes to the databases), no changes are committed to the databases.

This command is intended to be used in a large system environment where many accounts are updated at a single time.

OPTIONS

The options which apply to the **newusers** command are:

−−badname
Allow names that do not conform to standards.

−c, **−−crypt−method**
Use the specified method to encrypt the passwords.

The available methods are DES, MD5, NONE, and SHA256 or SHA512 if your libc support these methods.

−h, **−−help**
Display help message and exit.

−r, **−−system**
Create a system account.

System users will be created with no aging information in /etc/shadow, and their numeric identifiers are chosen in the **SYS_UID_MIN−SYS_UID_MAX** range, defined in login.defs, instead of **UID_MIN−UID_MAX** (and their **GID** counterparts for the creation of groups).

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−s, **−−sha−rounds**
Use the specified number of rounds to encrypt the passwords.

You can only use this option with crypt method: *SHA256 SHA512*

By default, the number of rounds for SHA256 or SHA512 is defined by the SHA_CRYPT_MIN_ROUNDS and SHA_CRYPT_MAX_ROUNDS variables in /etc/login.defs.

A minimal value of 1000 and a maximal value of 999,999,999 will be enforced for SHA256 and SHA512. The default is 5000.

CAVEATS

The input file must be protected since it contains unencrypted passwords.

You should make sure the passwords and the encryption method respect the system's password policy.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

ENCRYPT_METHOD (string)
This defines the system default encryption algorithm for encrypting passwords (if no algorithm are specified on the command line).

It can take one of these values: *DES* (default), *MD5*, *SHA256*, *SHA512*. MD5 and DES should not

be used for new hashes, see crypt(5) for recommendations.

Note: this parameter overrides the **MD5_CRYPT_ENAB** variable.

GID_MAX (number), **GID_MIN** (number)
Range of group IDs used for the creation of regular groups by **useradd**, **groupadd**, or **newusers**.

The default value for **GID_MIN** (resp. **GID_MAX**) is 1000 (resp. 60000).

HOME_MODE (number)
The mode for new home directories. If not specified, the **UMASK** is used to create the mode.

useradd and **newusers** use this to set the mode of the home directory they create.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

MD5_CRYPT_ENAB (boolean)
Indicate if passwords must be encrypted using the MD5−based algorithm. If set to *yes*, new passwords will be encrypted using the MD5−based algorithm compatible with the one used by recent releases of FreeBSD. It supports passwords of unlimited length and longer salt strings. Set to *no* if you need to copy encrypted passwords to other systems which don't understand the new algorithm. Default is *no*.

This variable is superseded by the **ENCRYPT_METHOD** variable or by any command line option used to configure the encryption algorithm.

This variable is deprecated. You should use **ENCRYPT_METHOD**.

PASS_MAX_DAYS (number)
The maximum number of days a password may be used. If the password is older than this, a password change will be forced. If not specified, −1 will be assumed (which disables the restriction).

PASS_MIN_DAYS (number)
The minimum number of days allowed between password changes. Any password changes attempted sooner than this will be rejected. If not specified, 0 will be assumed (which disables the restriction).

PASS_WARN_AGE (number)
The number of days warning given before a password expires. A zero means warning is given only upon the day of expiration, a value of −1 means no warning is given. If not specified, no warning will be provided.

SHA_CRYPT_MIN_ROUNDS (number), **SHA_CRYPT_MAX_ROUNDS** (number)
When **ENCRYPT_METHOD** is set to *SHA256* or *SHA512*, this defines the number of SHA rounds used by the encryption algorithm by default (when the number of rounds is not specified on the command line).

With a lot of rounds, it is more difficult to brute force the password. But note also that more CPU resources will be needed to authenticate users.

If not specified, the libc will choose the default number of rounds (5000), which is orders of magnitude too low for modern hardware.

The values must be inside the 1000−999,999,999 range.

If only one of the **SHA_CRYPT_MIN_ROUNDS** or **SHA_CRYPT_MAX_ROUNDS** values is set, then this value will be used.

If **SHA_CRYPT_MIN_ROUNDS** > **SHA_CRYPT_MAX_ROUNDS**, the highest value will be used.

SUB_GID_MIN (number), **SUB_GID_MAX** (number), **SUB_GID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate group IDs) allocate **SUB_GID_COUNT** unused group IDs from the range **SUB_GID_MIN** to **SUB_GID_MAX** for each new user.

The default values for **SUB_GID_MIN**, **SUB_GID_MAX**, **SUB_GID_COUNT** are respectively 100000, 600100000 and 65536.

SUB_UID_MIN (number), **SUB_UID_MAX** (number), **SUB_UID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate user IDs) allocate **SUB_UID_COUNT** unused user IDs from the range **SUB_UID_MIN** to **SUB_UID_MAX** for each new user.

The default values for **SUB_UID_MIN**, **SUB_UID_MAX**, **SUB_UID_COUNT** are respectively 100000, 600100000 and 65536.

SYS_GID_MAX (number), **SYS_GID_MIN** (number)
Range of group IDs used for the creation of system groups by **useradd**, **groupadd**, or **newusers**.

The default value for **SYS_GID_MIN** (resp. **SYS_GID_MAX**) is 101 (resp. **GID_MIN**−1).

SYS_UID_MAX (number), **SYS_UID_MIN** (number)
Range of user IDs used for the creation of system users by **useradd** or **newusers**.

The default value for **SYS_UID_MIN** (resp. **SYS_UID_MAX**) is 101 (resp. **UID_MIN**−1).

UID_MAX (number), **UID_MIN** (number)
Range of user IDs used for the creation of regular users by **useradd** or **newusers**.

The default value for **UID_MIN** (resp. **UID_MAX**) is 1000 (resp. 60000).

UMASK (number)
The file mode creation mask is initialized to this value. If not specified, the mask will be initialized to 022.

useradd and **newusers** use this mask to set the mode of the home directory they create if **HOME_MODE** is not set.

It is also used by **login** to define users' initial umask. Note that this mask can be overridden by the user's GECOS line (if **QUOTAS_ENAB** is set) or by the specification of a limit with the *K* identifier in *limits*(5).

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

/etc/login.defs
: Shadow password suite configuration.

/etc/subgid
: Per user subordinate group IDs.

/etc/subuid
: Per user subordinate user IDs.

SEE ALSO

login.defs(5), *passwd*(1), *subgid*(5), **subuid**(5), *useradd*(8).

NAME

nologin − politely refuse a login

SYNOPSIS

nologin

DESCRIPTION

The **nologin** command displays a message that an account is not available and exits non−zero. It is intended as a replacement shell field for accounts that have been disabled.

To disable all logins, investigate *nologin*(5).

If **SSH_ORIGINAL_COMMAND** is populated it will be logged.

SEE ALSO

login(1), *nologin*(5).

HISTORY

The **nologin** command appeared in BSD 4.4.

NAME

pwck − verify the integrity of password files

SYNOPSIS

pwck [options] [*PASSWORDFILE* [*SHADOWFILE*]]

DESCRIPTION

The **pwck** command verifies the integrity of the users and authentication information. It checks that all entries in /etc/passwd and /etc/shadow have the proper format and contain valid data. The user is prompted to delete entries that are improperly formatted or which have other uncorrectable errors.

Checks are made to verify that each entry has:

- the correct number of fields
- a unique and valid user name
- a valid user and group identifier
- a valid primary group
- a valid home directory
- a valid login shell

Checks for shadowed password information are enabled when the second file parameter *SHADOWFILE* is specified or when /etc/shadow exists on the system.

These checks are the following:

- every passwd entry has a matching shadow entry, and every shadow entry has a matching passwd entry
- passwords are specified in the shadowed file
- shadow entries have the correct number of fields
- shadow entries are unique in shadow
- the last password changes are not in the future

The checks for correct number of fields and unique user name are fatal. If the entry has the wrong number of fields, the user will be prompted to delete the entire line. If the user does not answer affirmatively, all further checks are bypassed. An entry with a duplicated user name is prompted for deletion, but the remaining checks will still be made. All other errors are warnings and the user is encouraged to run the **usermod** command to correct the error.

The commands which operate on the /etc/passwd file are not able to alter corrupted or duplicated entries. **pwck** should be used in those circumstances to remove the offending entry.

OPTIONS

The **−r** and **−s** options cannot be combined.

The options which apply to the **pwck** command are:

−−badname
Allow names that do not conform to standards.

−h, **−−help**
Display help message and exit.

−q, **−−quiet**
Report errors only. The warnings which do not require any action from the user won't be displayed.

−r, **−−read−only**
Execute the **pwck** command in read−only mode.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−s, **−−sort**

Sort entries in /etc/passwd and /etc/shadow by UID.

By default, **pwck** operates on the files /etc/passwd and /etc/shadow. The user may select alternate files with the *passwd* and *shadow* parameters.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

NONEXISTENT (string)
If a system account intentionally does not have a home directory that exists, this string can be provided in the /etc/passwd entry for the account to indicate this. The result is that pwck will not emit a spurious warning for this account.

PASS_MAX_DAYS (number)
The maximum number of days a password may be used. If the password is older than this, a password change will be forced. If not specified, −1 will be assumed (which disables the restriction).

PASS_MIN_DAYS (number)
The minimum number of days allowed between password changes. Any password changes attempted sooner than this will be rejected. If not specified, 0 will be assumed (which disables the restriction).

PASS_WARN_AGE (number)
The number of days warning given before a password expires. A zero means warning is given only upon the day of expiration, a value of −1 means no warning is given. If not specified, no warning will be provided.

FILES

/etc/group
Group account information.

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

EXIT VALUES

The **pwck** command exits with the following values:

0
success

1
invalid command syntax

2
one or more bad password entries

3
can't open password files

4
can't lock password files

5
can't update password files

6
can't sort password files

SEE ALSO

group(5), *grpck*(8), *passwd*(5), *shadow*(5), *usermod*(8).

NAME

pwconv, pwunconv, grpconv, grpunconv − convert to and from shadow passwords and groups

SYNOPSIS

pwconv [*options*]

pwunconv [*options*]

grpconv [*options*]

grpunconv [*options*]

DESCRIPTION

The **pwconv** command creates *shadow* from *passwd* and an optionally existing *shadow*.

The **pwunconv** command creates *passwd* from *passwd* and *shadow* and then removes *shadow*.

The **grpconv** command creates *gshadow* from *group* and an optionally existing *gshadow*.

The **grpunconv** command creates *group* from *group* and *gshadow* and then removes *gshadow*.

These four programs all operate on the normal and shadow password and group files: /etc/passwd, /etc/group, /etc/shadow, and /etc/gshadow.

Each program acquires the necessary locks before conversion. **pwconv** and **grpconv** are similar. First, entries in the shadowed file which don't exist in the main file are removed. Then, shadowed entries which don't have ‘x' as the password in the main file are updated. Any missing shadowed entries are added. Finally, passwords in the main file are replaced with ‘x'. These programs can be used for initial conversion as well to update the shadowed file if the main file is edited by hand.

pwconv will use the values of *PASS_MIN_DAYS*, *PASS_MAX_DAYS*, and *PASS_WARN_AGE* from /etc/login.defs when adding new entries to /etc/shadow.

Likewise **pwunconv** and **grpunconv** are similar. Passwords in the main file are updated from the shadowed file. Entries which exist in the main file but not in the shadowed file are left alone. Finally, the shadowed file is removed. Some password aging information is lost by **pwunconv**. It will convert what it can.

OPTIONS

The options which apply to the **pwconv**, **pwunconv**, **grpconv**, and **grpunconv** commands are:

−h, **−−help**
Display help message and exit.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

BUGS

Errors in the password or group files (such as invalid or duplicate entries) may cause these programs to loop forever or fail in other strange ways. Please run **pwck** and **grpck** to correct any such errors before converting to or from shadow passwords or groups.

CONFIGURATION

The following configuration variable in /etc/login.defs changes the behavior of **grpconv** and **grpunconv**:

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

The following configuration variables in /etc/login.defs change the behavior of **pwconv**:

PASS_MAX_DAYS (number)
The maximum number of days a password may be used. If the password is older than this, a password change will be forced. If not specified, −1 will be assumed (which disables the restriction).

PASS_MIN_DAYS (number)
The minimum number of days allowed between password changes. Any password changes attempted sooner than this will be rejected. If not specified, 0 will be assumed (which disables the restriction).

PASS_WARN_AGE (number)
The number of days warning given before a password expires. A zero means warning is given only upon the day of expiration, a value of −1 means no warning is given. If not specified, no warning will be provided.

FILES

/etc/login.defs
Shadow password suite configuration.

SEE ALSO

grpck(8), *login.defs*(5), *pwck*(8).

NAME

useradd − create a new user or update default new user information

SYNOPSIS

useradd [*options*] *LOGIN*

useradd −D

useradd −D [*options*]

DESCRIPTION

When invoked without the **−D** option, the **useradd** command creates a new user account using the values specified on the command line plus the default values from the system. Depending on command line options, the **useradd** command will update system files and may also create the new user's home directory and copy initial files.

By default, a group will also be created for the new user (see **−g**, **−N**, **−U**, and **USERGROUPS_ENAB**).

OPTIONS

The options which apply to the **useradd** command are:

−−badname
Allow names that do not conform to standards.

−b, **−−base−dir** *BASE_DIR*
The default base directory for the system if **−d** *HOME_DIR* is not specified. *BASE_DIR* is concatenated with the account name to define the home directory.

If this option is not specified, **useradd** will use the base directory specified by the **HOME** variable in /etc/default/useradd, or /home by default.

−c, **−−comment** *COMMENT*
Any text string. It is generally a short description of the account, and is currently used as the field for the user's full name.

−d, **−−home−dir** *HOME_DIR*
The new user will be created using *HOME_DIR* as the value for the user's login directory. The default is to append the *LOGIN* name to *BASE_DIR* and use that as the login directory name. If the directory *HOME_DIR* does not exist, then it will be created unless the **−M** option is specified.

−D, **−−defaults**
See below, the subsection "Changing the default values".

−e, **−−expiredate** *EXPIRE_DATE*
The date on which the user account will be disabled. The date is specified in the format *YYYY−MM−DD*.

If not specified, **useradd** will use the default expiry date specified by the **EXPIRE** variable in /etc/default/useradd, or an empty string (no expiry) by default.

−f, **−−inactive** *INACTIVE*
defines the number of days after the password exceeded its maximum age where the user is expected to replace this password. The value is stored in the shadow password file. An input of 0 will disable an expired password with no delay. An input of −1 will blank the respective field in the shadow password file. See *shadow*(5)for more information.

If not specified, **useradd** will use the default inactivity period specified by the **INACTIVE** variable in /etc/default/useradd, or −1 by default.

−F, **−−add−subids−for−system**
Update /etc/subuid and /etc/subgid even when creating a system account with **−r** option.

−g, **−−gid** *GROUP*
The name or the number of the user's primary group. The group name must exist. A group number must refer to an already existing group.

If not specified, the behavior of **useradd** will depend on the **USERGROUPS_ENAB** variable in /etc/login.defs. If this variable is set to *yes* (or **−U/−−user−group** is specified on the command line), a group will be created for the user, with the same name as her loginname. If the variable is set to *no* (or **−N/−−no−user−group** is specified on the command line), useradd will set the primary group of the new user to the value specified by the **GROUP** variable in /etc/default/useradd, or 1000 by default.

−G, **−−groups** *GROUP1*[,*GROUP2*,...[,*GROUPN*]]]
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 whitespace. The groups are subject to the same restrictions as the group given with the **−g** option. The default is for the user to belong only to the initial group. In addition to passing in the −G flag, you can add the option **GROUPS** to the file /etc/default/useradd which in turn will add all users to those supplementary groups.

−h, **−−help**
Display help message and exit.

−k, **−−skel** *SKEL_DIR*
The skeleton directory, which contains files and directories to be copied in the user's home directory, when the home directory is created by **useradd**.

This option is only valid if the **−m** (or **−−create−home**) option is specified.

If this option is not set, the skeleton directory is defined by the **SKEL** variable in /etc/default/useradd or, by default, /etc/skel.

If possible, the ACLs and extended attributes are copied.

−K, **−−key** *KEY=VALUE*
Overrides /etc/login.defs defaults (**UID_MIN**, **UID_MAX**, **UMASK**, **PASS_MAX_DAYS** and others).

Example: **−K** *PASS_MAX_DAYS =−1* can be used when creating an account to turn off password aging. Multiple **−K** options can be specified, e.g.: **−K** *UID_MIN =100* **−K** *UID_MAX=499*

−l, **−−no−log−init**
Do not add the user to the lastlog and faillog databases.

By default, the user's entries in the lastlog and faillog databases are reset to avoid reusing the entry from a previously deleted user.

If this option is not specified, **useradd** will also consult the variable **LOG_INIT** in the /etc/default/useradd if set to no the user will not be added to the lastlog and faillog databases.

−m, **−−create−home**
Create the user's home directory if it does not exist. The files and directories contained in the skeleton directory (which can be defined with the **−k** option) will be copied to the home directory.

By default, if this option is not specified and **CREATE_HOME** is not enabled, no home directories are created.

The directory where the user's home directory is created must exist and have proper SELinux context and permissions. Otherwise the user's home directory cannot be created or accessed.

−M, **−−no−create−home**
Do not create the user's home directory, even if the system wide setting from /etc/login.defs (**CREATE_HOME**) is set to *yes*.

−N, **−−no−user−group**
Do not create a group with the same name as the user, but add the user to the group specified by the **−g** option or by the **GROUP** variable in /etc/default/useradd.

The default behavior (if the **−g**, **−N**, and **−U** options are not specified) is defined by the

USERGROUPS_ENAB variable in /etc/login.defs.

−o, **−−non−unique**
allows the creation of an account with an already existing UID.

This option is only valid in combination with the **−u** option. As a user identity serves as key to map between users on one hand and permissions, file ownerships and other aspects that determine the system's behavior on the other hand, more than one login name will access the account of the given UID.

−p, **−−password** *PASSWORD*
defines an initial password for the account. PASSWORD is expected to be encrypted, as returned by **crypt** (3). Within a shell script, this option allows to create efficiently batches of users.

Without this option, the new account will be locked and with no password defined, i.e. a single exclamation mark in the respective field of /etc/shadow. This is a state where the user won't be able to access the account or to define a password himself.

Note: Avoid this option on the command line because the password (or encrypted password) will be visible by users listing the processes.

You should make sure the password respects the system's password policy.

−r, **−−system**
Create a system account.

System users will be created with no aging information in /etc/shadow, and their numeric identifiers are chosen in the **SYS_UID_MIN**−**SYS_UID_MAX** range, defined in /etc/login.defs, instead of **UID_MIN**−**UID_MAX** (and their **GID** counterparts for the creation of groups).

Note that **useradd** will not create a home directory for such a user, regardless of the default setting in /etc/login.defs (**CREATE_HOME**). You have to specify the **−m** options if you want a home directory for a system account to be created.

Note that this option will not update /etc/subuid and /etc/subgid. You have to specify the **−F** options if you want to update the files for a system account to be created.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes to configuration files under the root filesystem found under the directory *PREFIX_DIR*. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−s, **−−shell** *SHELL*
sets the path to the user's login shell. Without this option, the system will use the **SHELL** variable specified in /etc/default/useradd, or, if that is as well not set, the field for the login shell in /etc/passwd remains empty.

−u, **−−uid** *UID*
The numerical value of the user's ID. This value must be unique, unless the **−o** option is used. The value must be non−negative. The default is to use the smallest ID value greater than or equal to **UID_MIN** and greater than every other user.

See also the **−r** option and the **UID_MAX** description.

−U, **−−user−group**
Create a group with the same name as the user, and add the user to this group.

The default behavior (if the **−g**, **−N**, and **−U** options are not specified) is defined by the

USERGROUPS_ENAB variable in /etc/login.defs.

−Z, **−−selinux−user** *SEUSER*
defines the SELinux user for the new account. Without this option, SELinux uses the default user. Note that the shadow system doesn't store the selinux−user, it uses *semanage*(8) for that.

−−selinux−range *SERANGE*
defines the SELinux MLS range for the new account. Without this option, SELinux uses the default range. Note that the shadow system doesn't store the selinux−range, it uses *semanage*(8) for that.

This option is only valid if the **−Z** (or **−−selinux−user**) option is specified.

Changing the default values
When invoked with only the **−D** option, **useradd** will display the current default values. When invoked with **−D** plus other options, **useradd** will update the default values for the specified options. Valid default−changing options are:

−b, **−−base−dir** *BASE_DIR*
sets the path prefix for a new user's home directory. The user's name will be affixed to the end of *BASE_DIR* to form the new user's home directory name, if the **−d** option is not used when creating a new account.

This option sets the **HOME** variable in /etc/default/useradd.

−e, **−−expiredate** *EXPIRE_DATE*
sets the date on which newly created user accounts are disabled.

This option sets the **EXPIRE** variable in /etc/default/useradd.

−f, **−−inactive** *INACTIVE*
defines the number of days after the password exceeded its maximum age where the user is expected to replace this password. See *shadow*(5)for more information.

This option sets the **INACTIVE** variable in /etc/default/useradd.

−g, **−−gid** *GROUP*
sets the default primary group for newly created users, accepting group names or a numerical group ID. The named group must exist, and the GID must have an existing entry.

This option sets the **GROUP** variable in /etc/default/useradd.

−s, **−−shell** *SHELL*
defines the default login shell for new users.

This option sets the **SHELL** variable in /etc/default/useradd.

NOTES

The system administrator is responsible for placing the default user files in the /etc/skel/ directory (or any other skeleton directory specified in /etc/default/useradd or on the command line).

CAVEATS

You may not add a user to a NIS or LDAP group. This must be performed on the corresponding server.

Similarly, if the username already exists in an external user database such as NIS or LDAP, **useradd** will deny the user account creation request.

Usernames may contain only lower and upper case letters, digits, underscores, or dashes. They can end with a dollar sign. Dashes are not allowed at the beginning of the username. Fully numeric usernames and usernames . or .. are also disallowed. It is not recommended to use usernames beginning with . character as their home directories will be hidden in the **ls** output.

Usernames may only be up to 32 characters long.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

CREATE_HOME (boolean)

Indicate if a home directory should be created by default for new users.

This setting does not apply to system users, and can be overridden on the command line.

GID_MAX (number), **GID_MIN** (number)
Range of group IDs used for the creation of regular groups by **useradd**, **groupadd**, or **newusers**.

The default value for **GID_MIN** (resp. **GID_MAX**) is 1000 (resp. 60000).

HOME_MODE (number)
The mode for new home directories. If not specified, the **UMASK** is used to create the mode.

useradd and **newusers** use this to set the mode of the home directory they create.

LASTLOG_UID_MAX (number)
Highest user ID number for which the lastlog entries should be updated. As higher user IDs are usually tracked by remote user identity and authentication services there is no need to create a huge sparse lastlog file for them.

No **LASTLOG_UID_MAX** option present in the configuration means that there is no user ID limit for writing lastlog entries.

MAIL_DIR (string)
The mail spool directory. This is needed to manipulate the mailbox when its corresponding user account is modified or deleted. If not specified, a compile−time default is used. The parameter CREATE_MAIL_SPOOL in /etc/default/useradd determines whether the mail spool should be created.

MAIL_FILE (string)
Defines the location of the users mail spool files relatively to their home directory.

The **MAIL_DIR** and **MAIL_FILE** variables are used by **useradd**, **usermod**, and **userdel** to create, move, or delete the user's mail spool.

If **MAIL_CHECK_ENAB** is set to *yes*, they are also used to define the **MAIL** environment variable.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

PASS_MAX_DAYS (number)
The maximum number of days a password may be used. If the password is older than this, a password change will be forced. If not specified, −1 will be assumed (which disables the restriction).

PASS_MIN_DAYS (number)
The minimum number of days allowed between password changes. Any password changes attempted sooner than this will be rejected. If not specified, 0 will be assumed (which disables the restriction).

PASS_WARN_AGE (number)
The number of days warning given before a password expires. A zero means warning is given only upon the day of expiration, a value of −1 means no warning is given. If not specified, no warning will be provided.

SUB_GID_MIN (number), **SUB_GID_MAX** (number), **SUB_GID_COUNT** (number)

If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate group IDs) allocate **SUB_GID_COUNT** unused group IDs from the range **SUB_GID_MIN** to **SUB_GID_MAX** for each new user.

The default values for **SUB_GID_MIN**, **SUB_GID_MAX**, **SUB_GID_COUNT** are respectively 100000, 600100000 and 65536.

SUB_UID_MIN (number), **SUB_UID_MAX** (number), **SUB_UID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate user IDs) allocate **SUB_UID_COUNT** unused user IDs from the range **SUB_UID_MIN** to **SUB_UID_MAX** for each new user.

The default values for **SUB_UID_MIN**, **SUB_UID_MAX**, **SUB_UID_COUNT** are respectively 100000, 600100000 and 65536.

SYS_GID_MAX (number), **SYS_GID_MIN** (number)
Range of group IDs used for the creation of system groups by **useradd**, **groupadd**, or **newusers**.

The default value for **SYS_GID_MIN** (resp. **SYS_GID_MAX**) is 101 (resp. **GID_MIN**−1).

SYS_UID_MAX (number), **SYS_UID_MIN** (number)
Range of user IDs used for the creation of system users by **useradd** or **newusers**.

The default value for **SYS_UID_MIN** (resp. **SYS_UID_MAX**) is 101 (resp. **UID_MIN**−1).

UID_MAX (number), **UID_MIN** (number)
Range of user IDs used for the creation of regular users by **useradd** or **newusers**.

The default value for **UID_MIN** (resp. **UID_MAX**) is 1000 (resp. 60000).

UMASK (number)
The file mode creation mask is initialized to this value. If not specified, the mask will be initialized to 022.

useradd and **newusers** use this mask to set the mode of the home directory they create if **HOME_MODE** is not set.

It is also used by **login** to define users' initial umask. Note that this mask can be overridden by the user's GECOS line (if **QUOTAS_ENAB** is set) or by the specification of a limit with the *K* identifier in *limits*(5).

USERGROUPS_ENAB (boolean)
Enable setting of the umask group bits to be the same as owner bits (examples: 022 −> 002, 077 −> 007) for non−root users, if the uid is the same as gid, and username is the same as the primary group name.

If set to *yes*, **userdel** will remove the user's group if it contains no more members, and **useradd** will create by default a group with the name of the user.

FILES

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

/etc/default/useradd
Default values for account creation.

/etc/shadow−maint/useradd−pre.d/*, /etc/shadow−maint/useradd−post.d/*

Run−part files to execute during user addition. The environment variable **ACTION** will be populated with useradd and **SUBJECT** with the **username**. useradd−pre.d will be executed prior to any user addition. useradd−post.d will execute after user addition. If a script exits non−zero then execution will terminate.

/etc/skel/
: Directory containing default files.

/etc/subgid
: Per user subordinate group IDs.

/etc/subuid
: Per user subordinate user IDs.

/etc/login.defs
: Shadow password suite configuration.

EXIT VALUES

The **useradd** command exits with the following values:

0
: success

1
: can't update password file

2
: invalid command syntax

3
: invalid argument to option

4
: UID already in use (and no **−o**)

6
: specified group doesn't exist

9
: username or group name already in use

10
: can't update group file

12
: can't create home directory

14
: can't update SELinux user mapping

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *crypt*(3), *groupadd*(8), *groupdel*(8), *groupmod*(8), *login.defs*(5), *newusers*(8), *subgid*(5), **subuid**(5), *userdel*(8), *usermod*(8).

NAME

userdel − delete a user account and related files

SYNOPSIS

userdel [options] *LOGIN*

DESCRIPTION

The **userdel** command modifies the system account files, deleting all entries that refer to the user name *LOGIN*. The named user must exist.

OPTIONS

The options which apply to the **userdel** command are:

−f, **−−force**
This option forces the removal of the user account, even if the user is still logged in. It also forces **userdel** to remove the user's home directory and mail spool, even if another user uses the same home directory or if the mail spool is not owned by the specified user. If **USERGROUPS_ENAB** is defined to *yes* in /etc/login.defs and if a group exists with the same name as the deleted user, then this group will be removed, even if it is still the primary group of another user.

Note: This option is dangerous and may leave your system in an inconsistent state.

−h, **−−help**
Display help message and exit.

−r, **−−remove**
Files in the user's home directory will be removed along with the home directory itself and the user's mail spool. Files located in other file systems will have to be searched for and deleted manually.

The mail spool is defined by the **MAIL_DIR** variable in the login.defs file.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes in the *PREFIX_DIR* directory and use the configuration files from the *PREFIX_DIR* directory. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−Z, **−−selinux−user**
Remove any SELinux user mapping for the user's login.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

MAIL_DIR (string)
The mail spool directory. This is needed to manipulate the mailbox when its corresponding user account is modified or deleted. If not specified, a compile−time default is used. The parameter CREATE_MAIL_SPOOL in /etc/default/useradd determines whether the mail spool should be created.

MAIL_FILE (string)
Defines the location of the users mail spool files relatively to their home directory.

The **MAIL_DIR** and **MAIL_FILE** variables are used by **useradd**, **usermod**, and **userdel** to create, move, or delete the user's mail spool.

If **MAIL_CHECK_ENAB** is set to *yes*, they are also used to define the **MAIL** environment variable.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

USERDEL_CMD (string)
If defined, this command is run when removing a user. It should remove any at/cron/print jobs etc. owned by the user to be removed (passed as the first argument).

The return code of the script is not taken into account.

Here is an example script, which removes the user's cron, at and print jobs:

```
#! /bin/sh
# Check for the required argument.
if [ $# != 1 ]; then
        echo "Usage: $0 username"
        exit 1
fi
# Remove cron jobs.
crontab −r −u $1
# Remove at jobs.
# Note that it will remove any jobs owned by the same UID,
# even if it was shared by a different username.
AT_SPOOL_DIR=/var/spool/cron/atjobs
find $AT_SPOOL_DIR −name "[^.]*" −type f −user $1 −delete \;
# Remove print jobs.
lprm $1
# All done.
exit 0
```

USERGROUPS_ENAB (boolean)
Enable setting of the umask group bits to be the same as owner bits (examples: 022 −> 002, 077 −> 007) for non−root users, if the uid is the same as gid, and username is the same as the primary group name.

If set to *yes*, **userdel** will remove the user's group if it contains no more members, and **useradd** will create by default a group with the name of the user.

FILES

/etc/group
Group account information.

/etc/login.defs
Shadow password suite configuration.

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

/etc/shadow−maint/userdel−pre.d/*, /etc/shadow−maint/userdel−post.d/*
Run−part files to execute during user deletion. The environment variable **ACTION** will be populated with **userdel** and **SUBJECT** with the username. userdel−pre.d will be executed prior to any user deletion. userdel−post.d will execute after user deletion. If a script exits non−zero then

execution will terminate.

/etc/subgid
: Per user subordinate group IDs.

/etc/subuid
: Per user subordinate user IDs.

EXIT VALUES

The **userdel** command exits with the following values:

0
: success

1
: can't update password file

2
: invalid command syntax

6
: specified user doesn't exist

8
: user currently logged in

10
: can't update group file

12
: can't remove home directory

CAVEATS

userdel will not allow you to remove an account if there are running processes which belong to this account. In that case, you may have to kill those processes or lock the user's password or account and remove the account later. The **−f** option can force the deletion of this account.

You should manually check all file systems to ensure that no files remain owned by this user.

You may not remove any NIS attributes on a NIS client. This must be performed on the NIS server.

If **USERGROUPS_ENAB** is defined to *yes* in /etc/login.defs, **userdel** will delete the group with the same name as the user. To avoid inconsistencies in the passwd and group databases, **userdel** will check that this group is not used as a primary group for another user, and will just warn without deleting the group otherwise. The **−f** option can force the deletion of this group.

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *login.defs*(5), *gpasswd*(8), *groupadd*(8), *groupdel*(8), *groupmod*(8), *subgid*(5), **subuid**(5), *useradd*(8), *usermod*(8).

NAME

usermod − modify a user account

SYNOPSIS

usermod [*options*] *LOGIN*

DESCRIPTION

The **usermod** command modifies the system account files.

OPTIONS

The options which apply to the **usermod** command are:

−a, **−−append**
Add the user to the supplementary group(s). Use only with the **−G** option.

−b, **−−badname**
Allow names that do not conform to standards.

−c, **−−comment** *COMMENT*
update the comment field of the user in /etc/passwd, which is normally modified using the *chfn*(1) utility.

−d, **−−home** *HOME_DIR*
The user's new login directory.

If the **−m** option is given, the contents of the current home directory will be moved to the new home directory, which is created if it does not already exist. If the current home directory does not exist the new home directory will not be created.

−e, **−−expiredate** *EXPIRE_DATE*
The date on which the user account will be disabled. The date is specified in the format *YYYY−MM−DD*. Integers as input are interpreted as days after 1970−01−01.

An input of −1 or an empty string will blank the account expiration field in the shadow password file. The account will remain available with no date limit.

This option requires a /etc/shadow file. A /etc/shadow entry will be created if there were none.

−f, **−−inactive** *INACTIVE*
defines the number of days after the password exceeded its maximum age during which the user may still login by immediately replacing the password. This grace period before the account becomes inactive is stored in the shadow password file. An input of 0 will disable an expired password with no delay. An input of −1 will blank the respective field in the shadow password file. See *shadow*(5) for more information.

This option requires a /etc/shadow file. A /etc/shadow entry will be created if there were none.

−g, **−−gid** *GROUP*
The name or numerical ID of the user's new primary group. The group must exist.

Any file from the user's home directory owned by the previous primary group of the user will be owned by this new group.

The group ownership of files outside of the user's home directory must be fixed manually.

The change of the group ownership of files inside of the user's home directory is also not done if the home dir owner uid is different from the current or new user id. This is a safety measure for special home directories such as /.

−G, **−−groups** *GROUP1*[,*GROUP2*,...[,*GROUPN*]]]
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 whitespace. The groups must exist.

If the user is currently a member of a group which is not listed, the user will be removed from the

group. This behaviour can be changed via the **−a** option, which appends the user to the current supplementary group list.

−l, **−−login** *NEW_LOGIN*
The name of the user will be changed from *LOGIN* to *NEW_LOGIN*. Nothing else is changed. In particular, the user's home directory or mail spool should probably be renamed manually to reflect the new login name.

−L, **−−lock**
Lock a user's password. This puts a '!' in front of the encrypted password, effectively disabling the password. You can't use this option with **−p** or **−U**.

Note: if you wish to lock the account (not only access with a password), you should also set the *EXPIRE_DATE* to *1*.

−m, **−−move−home**
moves the content of the user's home directory to the new location. If the current home directory does not exist the new home directory will not be created.

This option is only valid in combination with the **−d** (or **−−home**) option.

usermod will try to adapt the ownership of the files and to copy the modes, ACL and extended attributes, but manual changes might be needed afterwards.

−o, **−−non−unique**
allows to change the user ID to a non−unique value.

This option is only valid in combination with the **−u** option. As a user identity serves as key to map between users on one hand and permissions, file ownerships and other aspects that determine the system's behavior on the other hand, more than one login name will access the account of the given UID.

−p, **−−password** *PASSWORD*
defines a new password for the user. PASSWORD is expected to be encrypted, as returned by **crypt** (3).

Note: Avoid this option on the command line because the password (or encrypted password) will be visible by users listing the processes.

You should make sure the password respects the system's password policy.

−r, **−−remove**
Remove the user from named supplementary group(s). Use only with the **−G** option.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−P, **−−prefix** *PREFIX_DIR*
Apply changes within the directory tree starting with *PREFIX_DIR* and use as well the configuration files located there. This option does not chroot and is intended for preparing a cross−compilation target. Some limitations: NIS and LDAP users/groups are not verified. PAM authentication is using the host files. No SELINUX support.

−s, **−−shell** *SHELL*
changes the user's login shell. An empty string for SHELL blanks the field in /etc/passwd and logs the user into the system's default shell.

−u, **−−uid** *UID*
The new value of the user's ID.

This value must be unique, unless the **−o** option is used. The value must be non−negative.

The user's mailbox, and any files which the user owns and which are located in the user's home directory will have the file user ID changed automatically.

The ownership of files outside of the user's home directory must be fixed manually.

The change of the user ownership of files inside of the user's home directory is also not done if the home dir owner uid is different from the current or new user id. This is a safety measure for special home directories such as /.

No checks will be performed with regard to the **UID_MIN**, **UID_MAX**, **SYS_UID_MIN**, or **SYS_UID_MAX** from /etc/login.defs.

−U, **−−unlock**
Unlock a user's password. This removes the '!' in front of the encrypted password. You can't use this option with **−p** or **−L**.

Note: if you wish to unlock the account (not only access with a password), you should also set the *EXPIRE_DATE* (for example to *99999*, or to the **EXPIRE** value from /etc/default/useradd).

−v, **−−add−subuids** *FIRST−LAST*
Add a range of subordinate uids to the user's account.

This option may be specified multiple times to add multiple ranges to a user's account.

No checks will be performed with regard to **SUB_UID_MIN**, **SUB_UID_MAX**, or **SUB_UID_COUNT** from /etc/login.defs.

−V, **−−del−subuids** *FIRST−LAST*
Remove a range of subordinate uids from the user's account.

This option may be specified multiple times to remove multiple ranges to a user's account. When both **−−del−subuids** and **−−add−subuids** are specified, the removal of all subordinate uid ranges happens before any subordinate uid range is added.

No checks will be performed with regard to **SUB_UID_MIN**, **SUB_UID_MAX**, or **SUB_UID_COUNT** from /etc/login.defs.

−w, **−−add−subgids** *FIRST−LAST*
Add a range of subordinate gids to the user's account.

This option may be specified multiple times to add multiple ranges to a user's account.

No checks will be performed with regard to **SUB_GID_MIN**, **SUB_GID_MAX**, or **SUB_GID_COUNT** from /etc/login.defs.

−W, **−−del−subgids** *FIRST−LAST*
Remove a range of subordinate gids from the user's account.

This option may be specified multiple times to remove multiple ranges to a user's account. When both **−−del−subgids** and **−−add−subgids** are specified, the removal of all subordinate gid ranges happens before any subordinate gid range is added.

No checks will be performed with regard to **SUB_GID_MIN**, **SUB_GID_MAX**, or **SUB_GID_COUNT** from /etc/login.defs.

−Z, **−−selinux−user** *SEUSER*
defines the SELinux user to be mapped with *LOGIN*. An empty string ("") will remove the respective entry (if any). Note that the shadow system doesn't store the selinux−user, it uses semanage(8) for that.

−−selinux−range *SERANGE*
defines the SELinux MLS range for the new account. Note that the shadow system doesn't store the selinux−range, it uses *semanage*(8) for that.

This option is only valid if the **−Z** (or **−−selinux−user**) option is specified.

CAVEATS

You must make certain that the named user is not executing any processes when this command is being executed if the user's numerical user ID, the user's name, or the user's home directory is being changed. **usermod** checks this on Linux. On other operating systems it only uses utmp to check if the user is logged in.

You must change the owner of any **crontab** files or **at** jobs manually.

You must make any changes involving NIS on the NIS server.

CONFIGURATION

The following configuration variables in /etc/login.defs change the behavior of this tool:

LASTLOG_UID_MAX (number)
Highest user ID number for which the lastlog entries should be updated. As higher user IDs are usually tracked by remote user identity and authentication services there is no need to create a huge sparse lastlog file for them.

No **LASTLOG_UID_MAX** option present in the configuration means that there is no user ID limit for writing lastlog entries.

MAIL_DIR (string)
The mail spool directory. This is needed to manipulate the mailbox when its corresponding user account is modified or deleted. If not specified, a compile−time default is used. The parameter CREATE_MAIL_SPOOL in /etc/default/useradd determines whether the mail spool should be created.

MAIL_FILE (string)
Defines the location of the users mail spool files relatively to their home directory.

The **MAIL_DIR** and **MAIL_FILE** variables are used by **useradd**, **usermod**, and **userdel** to create, move, or delete the user's mail spool.

If **MAIL_CHECK_ENAB** is set to *yes*, they are also used to define the **MAIL** environment variable.

MAX_MEMBERS_PER_GROUP (number)
Maximum members per group entry. When the maximum is reached, a new group entry (line) is started in /etc/group (with the same name, same password, and same GID).

The default value is 0, meaning that there are no limits in the number of members in a group.

This feature (split group) permits to limit the length of lines in the group file. This is useful to make sure that lines for NIS groups are not larger than 1024 characters.

If you need to enforce such limit, you can use 25.

Note: split groups may not be supported by all tools (even in the Shadow toolsuite). You should not use this variable unless you really need it.

SUB_GID_MIN (number), **SUB_GID_MAX** (number), **SUB_GID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate group IDs) allocate **SUB_GID_COUNT** unused group IDs from the range **SUB_GID_MIN** to **SUB_GID_MAX** for each new user.

The default values for **SUB_GID_MIN**, **SUB_GID_MAX**, **SUB_GID_COUNT** are respectively 100000, 600100000 and 65536.

SUB_UID_MIN (number), **SUB_UID_MAX** (number), **SUB_UID_COUNT** (number)
If /etc/subuid exists, the commands **useradd** and **newusers** (unless the user already have subordinate user IDs) allocate **SUB_UID_COUNT** unused user IDs from the range **SUB_UID_MIN** to **SUB_UID_MAX** for each new user.

The default values for **SUB_UID_MIN**, **SUB_UID_MAX**, **SUB_UID_COUNT** are respectively 100000, 600100000 and 65536.

FILES

/etc/group
: Group account information

/etc/gshadow
: Secure group account information

/etc/login.defs
: Shadow password suite configuration

/etc/passwd
: User account information

/etc/shadow
: Secure user account information

/etc/subgid
: Per user subordinate group IDs

/etc/subuid
: Per user subordinate user IDs

SEE ALSO

chfn(1), *chsh*(1), *passwd*(1), *crypt*(3), *gpasswd*(8), *groupadd*(8), *groupdel*(8), *groupmod*(8), *login.defs*(5), *subgid*(5), **subuid**(5), *useradd*(8), *userdel*(8).

NAME

vipw, vigr − edit the password, group, shadow−password or shadow−group file

SYNOPSIS

vipw [*options*]

vigr [*options*]

DESCRIPTION

The **vipw** and **vigr** commands edit the files /etc/passwd and /etc/group, respectively. With the **−s** flag, they will edit the shadow versions of those files, /etc/shadow and /etc/gshadow, respectively. The programs will set the appropriate locks to prevent file corruption. When looking for an editor, the programs will first try the environment variable **$VISUAL**, then the environment variable **$EDITOR**, and finally the default editor, *vi*(1).

OPTIONS

The options which apply to the **vipw** and **vigr** commands are:

−g, **−−group**
Edit group database.

−h, **−−help**
Display help message and exit.

−p, **−−passwd**
Edit passwd database.

−q, **−−quiet**
Quiet mode.

−R, **−−root** *CHROOT_DIR*
Apply changes in the *CHROOT_DIR* directory and use the configuration files from the *CHROOT_DIR* directory. Only absolute paths are supported.

−s, **−−shadow**
Edit shadow or gshadow database.

ENVIRONMENT

VISUAL
Editor to be used.

EDITOR
Editor to be used if **VISUAL** is not set.

FILES

/etc/group
Group account information.

/etc/gshadow
Secure group account information.

/etc/passwd
User account information.

/etc/shadow
Secure user account information.

SEE ALSO

vi(1), *group*(5), *gshadow*(5) , *passwd*(5), , *shadow*(5).