

10

FERMILAB

NOV 16 1999

LIBRARY

LNF-99-020-NT



LABORATORI NAZIONALI DI FRASCATI SIS-Pubblicazioni

LNF-99/020 (NT)
21 Luglio 1999

Allocate a device in Unix Environment, V1.0.

Massimo Carboni¹ email:Massimo.Carboni@lnf.infn.it,
Sandro Angius¹ email:Sandro.Angius@lnf.infn.it

¹)INFN, Laboratori Nazionali di Frascati, P.O. Box 13, I-00044 Frascati, Italy

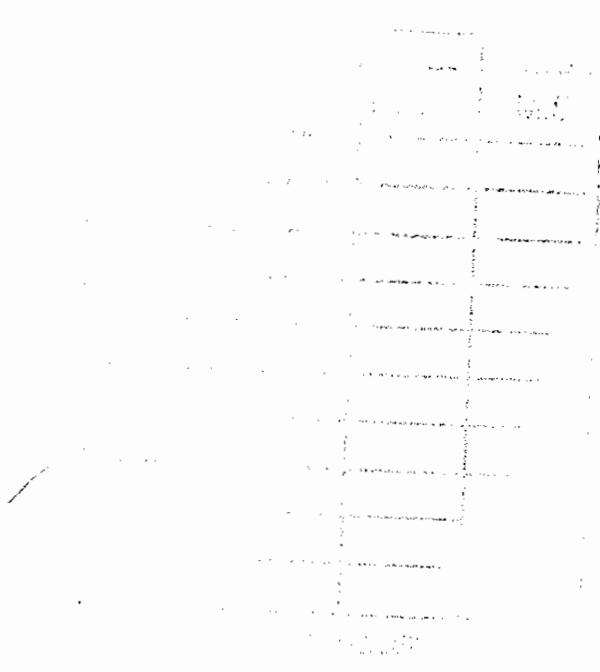


Abstract

In the past, in our computing environment there was the possibility of exclusive using of tape drives, since VMS system offers natively an allocating facility.

Nowadays, we have both VMS and Unix environments, used by many users of the various research groups. Due to the absence of a native Unix utility to allocate devices, we decided to write something to perform this operation.

The development of this tool is still in progress: the distribution structure and the installation details may be changed in future releases.



1 Introduction

The purpose of this software is to give an allocating/deallocating mechanism for devices in Unix Environment. Typical devices which need to be reserved are tape drives. This S/W can help those who need to give the possibility to their users of sharing tape drives without generating conflicts. This S/W works with any kind of file, so those who need to share other resource related files can use it. For this reason, in the following sections we shall use the words file and device, in (de)allocation context, as synonymous. This S/W also allows to restrict the access to a device to one or more groups.

2 How does it work ?

2.1 What does "allocate/deallocate" operation mean ?

The allocation is performed by changing the ownership of the requested file to the requester uid. The deallocation of a device is performed by restoring ownership and access permission of the involved file to the default value as defined in the program. The allocation is related to the session; a daemon will automatically deallocate the device when the process terminates.

Note: The allocation and deallocation must be invoked by the same process, but no exclusive access is granted, i.e. every other process which has the same uid can access the allocated file.

2.2 allocate and deallocate are suid root programs!

Due to the kind of operation needed to accomplish the work, the (de)allocate program need to run as root. A suid root program is always a possible security hole, and you must be aware of this. However, the program switches to the requester uid when root privileges are not necessary (-list and -owned operation).

2.3 Grouping of devices

On some operating systems, a physical device is controlled by more than one device file, each one determining a specific use of the tape (i.e. density to use, compression on/off). To avoid an incorrect use, all the same related device-files are grouped into an unique nickname, defined as a common string name. This behaviour avoids the allocation of different files connected to the same physical device by two or more users. It also prevents the use of a file different from the allocated one, even if refers to the same device.

2.4 Software components

The S/W consists of two programs:

- `allocate/deallocate`
- `deallocated`

The former performs all the operations to allocate or deallocate a device. The name used to invoke the command determines the specific operation. Due to this feature, if you use a link to reference the `allocate/deallocate` commands (normally two hard links to the same file), you need to call it `allocate` or `deallocate`.

The latter is a daemon that starts at boot time and periodically checks if a process ends without deallocating a device which was previously allocated by it. In this case the daemon will deallocate the device.

3 Availability

The latest software version can be found at this URL:

<http://www.lnf.infn.it/computing/src/TapeTools.tar.gz>

4 Installation guidelines and configuration

4.1 Installation

1. Untar and unzip the kit. It will create a subdirectory named `tapetools` in your current directory.
2. Change directory to `tapetools/src`, edit the `Makefile` to adapt it to what your system needs, possibly changing the definitions that you want to customize (see section 4.2 (Software configuration) for more information).
3. Issue the command `“make”`.
4. Login as `root` and do `“make install”`. The installing phase will copy the executable programs in the area defined by `Makefile`, also setting their ownership and permissions.
5. Set your system to start the `deallocated` daemon at boot time.
6. Finally, set up your devices and write your `Configuration` file as explained in section 4.3 (Configuration file).

4.2 Software configuration

You can choose to change some definitions to customize the software to your own preferences. In the following we list what you can configure editing the Makefile:

- **LOG_FILE**
Log file for all allocate/deallocate operations, see section 5 (Usage) for description (default is: `/var/log/allocate.log`).
- **ROOT_DEV**
Device files directory (default is: `/dev`).
- **ROOT_LOCK**
Lock files directory (default is: `/var/spool/tapes`), the directory must be owned by root and have 755 access permission.
Note: the lock files must be owned by root and have 644 access permission.
- **LIST_OF_DEVICES**
Configuration file defining managed files, see next subsection (default is: `/var/spool/tapes/DEVICES.ALLOWED`). This file must be owned by root and have 644 access permission.
- **RUN_FILE**
File containing the pid of deallocated daemon (default is: `/var/run/deallocated.pid`).
- **SLEEPTIME**
Sleep time (in seconds) for deallocated daemon (default is: 60).

4.3 Configuration file

The LIST_OF_DEVICES file defines the managed files, how they are grouped and group permissions. Empty line, or lines beginning with '#' char, are ignored. A valid line must have the following syntax:

```
<nickname>;<file>,[file, ...];[group, ...]
```

Where nickname is a word that identifies a device (always mandatory). File is the basename of device file (the dirname is **ROOT_DEV**), at least one occurrence must be present. Group is the name of group that can use the device.

For example, if you have a DAT and you have two device files to use it in rewind and not rewind manner (`/dev/dat0`, `/dev/ndat0`) the line to be written is:

```
dat;dat0,ndat0
```

If you want to restrict the use of this DAT to exp and soft groups, you have just to write:

```
dat;dat0,ndat0;exp,soft
```

4.4 Device configuration

Before using this S/W, you need to create the files that you want to manage in the directory specified by **ROOT_DEV**.

For example, if you want to manage tape devices in Digital Unix environment, you can create a directory named /dev/shared owned by root and with rx permission for every user (remember to set **ROOT_DEV** to /dev/shared before invoking make). Then, in this directory, create the device files that you want, using the mknod command and set them with ownership to root and 600 access permission.

Note: if you set other ownership and/or access permission when the file is deallocated you will lose your settings.

5 Usage

In allocation/deallocation operation, the allocate and deallocate commands require as parameter only the basename of the device file to manage. Upon success, the command will show the complete device file name. You can also use these commands to list the managed devices (i.e. allocate -list), or to see the present situation of devices (i.e. allocate -owned).

Refer to man pages for further information and examples.

5.1 Log file messages

Every allocation and deallocation operation is logged into a file, see section 4.2 (Software configuration) for more information. Here is an example of the log:

```
{1998/04/22 14:46:28}    allocate demoid   expgrp      3889 dat0      dat0h
{1998/04/22 14:56:33}    deallocate demoid expgrp      3889 dat0      dat0h
{1998/04/22 14:56:39}    allocate demoid   expgrp      3889 dat0      dat0h
{1998/05/08 12:25:57}    allocate expmgr   (expgrp)    2064 dlt3      dlt3h
{1998/05/11 10:01:30}    deallocated (root) (daemon)    1669 dlt3      dlt3h
{1998/05/21 11:54:37}    allocate techsupp (software)  30971 dat2      ndat21
```

```

{1998/05/21 17:22:32}    allocate techsupp (software) 10469 exa1    nexa1l
{1998/05/21 17:25:05}    deallocate techsupp (software) 10469 exa1    nexa1l
{1998/05/21 17:25:13}    allocate techsupp (software) 10469 exa1    nexa1h
{1998/05/21 17:31:16}    deallocated (root) (daemon) 24593 dat2    ndat2l
{1998/05/21 18:24:41}    deallocate techsupp (software) 10469 exa1    nexa1h
{1998/05/22 15:16:19}    allocate hoffice (hardware) 27931 exa1    nexa1h
{1998/05/22 16:48:01}    deallocated Error deleting ROOT_LOCK/LOCK..exa1
{1998/05/25 15:16:19}    allocate hoffice (hardware) 27931 exa1    nexa1h
{1998/05/25 15:22:26}    deallocated (root) (daemon) 24593 exa1    nexa1h

```

Where:

- 1st column (between {}) is the date and hour of the event.
- 2nd column is the name of program which logged the information.

In case of error, the 3rd column is the last and it will contain a message giving information about the error type. The message is similar (only in shorter syntax) to the message shown to the user issuing the command. See section 6.1 (Troubleshooting) for more information.

Otherwise, if there are no errors, the other columns are:

- 3rd column is the user who runned the program.
For deallocated, the field is in brackets.
- 4th column is the group of the user who runned the program. It's shown in brackets when the group isn't necessary to allow access to the device and stands for the main group of the user. It is shown without brackets, when the access to a device is restricted to one or more groups and stands for the group (the first) that matches the authorised groups. For deallocated, the field is always in brackets.
- 5th column is the pid of program who logged the information.
- 6th column is the nickname of the device involved.
- 7th column is the basename of the device involved.

6 Appendix

6.1 Troubleshooting

Messages returned by `allocate` or `deallocate` operations:

- `>>>` This command can be named only `allocate` or `deallocate`.
You have renamed the command with a name different from `allocate` or `deallocate`, or you are using a link not named `allocate` or `deallocate`: rename it.
- `>>> allocate`: You are not allowed to allocate devices.
`>>> deallocate`: You are not allowed to deallocate devices.
The program is not `suid` root, or not owned by root, do:


```
chown root:system <file>
chown 4755 <file>
```
- `>>>allocate`:Invalid line in configuration file.
`>>>deallocate`:Invalid line in configuration file.
An incorrect line has been detected in the configuration file, see 4.3 (Configuration file) section for more info.
- `>>> allocate`: you are not belonging to groups
The allocation of this file is reserved to groups specified in configuration file, see 4.3 (Configuration file) section for more info.
- `>>> deallocate`: you are not belonging to groups
The configuration file does not allow you to deallocate the specified file, check if you are in the same process that allocated the file or check if the configuration file has been modified.
- `>>>allocate`: Invalid permission or ownership on file
`>>>deallocate`: Invalid permission or ownership on file
The specified file has a wrong permission or ownership, check if they are the same as specified in section 4.2 (Software configuration) for more information).
- `>>>allocate`: cannot open device list
`>>>deallocate`: cannot open device list
`>>>allocate`: cannot stat device list
`>>>deallocate`: cannot stat device list
Check if the configuration file exists and if it is specified in `Makefile`, see section 4.2 (Software configuration) for more information.

- >>>allocate: Cannot open lock directory
 >>>deallocate: Cannot open lock directory
 Check if the directory for lock files exists and if it is specified in Makefile, see section 4.2 (Software configuration) for more information.
 - >>> deallocate: Error removing lock file. Remove by hand the lock file (look for it in directory **ROOT_LOCK**) and reset permissions on the involved device as specified in section 4.4 (Device Configuration).
 - >>> allocate: device not (de)allocable.
 The specified device exists, but it isn't defined in configuration file see 4.3 (Configuration file) section for more info.
 - >>> deallocate: device not (de)allocable.
 The specified device exists, but it isn't defined in configuration file; check if it has been changed.
 - >>> allocate: Cannot open:
 >>> deallocate: Cannot open:
 The lock files or its directory have wrong permission, they must be owned by root and 644 access mode; see section 4.2 (Software configuration) for more information.
 - >>> deallocate: Deallocation failed due chown error.
 >>> allocate: allocation failed due chown error.
 >>> deallocate: Deallocation failed due chmod error.
 These messages might never occur.
- Messages returned by deallocated daemon:
- >>>deallocated: You are not allowed to run this daemon.
 You are trying to start the daemon, but you are not root.
 - >>>deallocated: Found another daemon exiting.
 The RUN_FILE indicates a daemon already running, remove the RUN_FILE if no deallocated process running exists. See section 4.2 (Software configuration) for more information on RUN_FILE.
 - >>>deallocated: Cannot read exiting.
 >>>deallocated: Cannot write exiting.
 These messages should never occur.

NAME

(de)allocate – (de)allocate tape utility

SYNOPSIS

allocate <tape_device> | -owned | -list [Nickname]

deallocate <tape_device> | -owned | -list [Nickname]

DESCRIPTION

The (de)allocate application allow the user to (release)reserve a physical tape drive. The parameter **tape_device** is the basename that correspond to the real device file. The complete device file is returned at the command completion.

After allocation the device file has the ownership of the requester, a lock file avoid new allocation. Allocation is referred to the current session, when it ends the system will automatically deallocate.

OPTIONS

tape_device Tape device name to be (de)allocate.

-owned Give information on allocated devices.

-list [Nickname] List devices available online.

RETURN VALUES

The (de)allocate command returns a 0 (zero) exit status when the operations were successful, a 1 if the operation failed.

EXAMPLES

The following example allocate tape device /dev/nexa1h:

```
% allocate nexa1h
>>> allocate: /dev/nexa1h successfully allocated
```

To deallocate /dev/nexa1h issue:

```
% deallocate nexa1h
>>> deallocate: /dev/nexa1h successfully deallocated
```

List of managed devices:

```
% allocate -list
AS (Nickname) Device Files available
-----
NA (dat0 ) dat0a dat0h dat0l ndat0a ndat0h ndat0l
YD (exa1 ) exa1a exa1h exa1l nexa1a nexa1h nexa1l
YA (dat2 ) dat2a dat2h dat2l ndat2a ndat2h ndat2l
YD (dlt3 ) dlt3a dlt3h dlt3l ndlt3a ndlt3h ndlt3l
```

Where column A (Availability) tells if device can be allocated by your group (Y) or not (N), and column S (Status) if device is allocated (A) or deallocated (D).

Get information on allocated devices:

```
% allocate -owned
(Nickname) Device Owned by Pid Notes
```

(DE)ALLOCATE(1)

(DE)ALLOCATE(1)

```
(dat0 ) dat0h user1 3889 Owned by yourself
(dat2 ) dat2h user1 2217 Owned by yourself (This session)
(dlt3 ) dlt3a user2 5361
```

FILES

```
/usr/lnf/bin/allocate
/usr/lnf/bin/deallocate
/usr/lnf/sbin/deallocated
/var/spool/tapes/etc/DEVICES.ALLOWED
/var/spool/tapes/lock/
/var/spool/tapes/lock/TapeLog.txt
```

SEE ALSO

mt(1).

HISTORY

Origin: Laboratori Nazionali di Frascati / INFN

Original release. April 1998 – Massimo Carboni & Sandro Angius, LNF/CS

