NAME

autofs − Format of the automounter maps

DESCRIPTION

The automounter maps are FILE, NIS, NISPLUS or LDAP maps referred to by the master map of the automounter (see **auto.master**(5)). These maps describe how file systems below the mount point of the map (given in the master map) are to be mounted. This page describes the **sun** map format; if another map format, other than **amd ,** is specified (e.g. **hesiod**), this documentation does not apply.

Indirect maps, except for the internal hosts map, can be changed on the fly and the automouter will recognize those changes on the next operation it performs on that map. Direct maps require a HUP signal be sent to the daemon to refresh their contents as does the master map.

SUN FORMAT

This is a description of the text file format. Other methods of specifying these files may exist. All empty lines or lines beginning with # are ignored. The basic format of one line in such maps is:

key [**-options**] **location**

key

For indirect mounts this is the part of the path name between the mount point and the path into the filesystem when it is mounted. Usually you can think about the key as a sub-directory name below the autofs managed mount point.

For direct mounts this is the full path of each mount point. This map is always associated with the /- mount point in the master map.

options

Zero or more options may be given. Options can also be given in the **auto.master** file in which case both values are cumulative (this is a difference from SunOS). The options are a list of comma separated options as customary for the **mount**(8) command.

There are several special options

-fstype=

is used to specify a filesystem type if the filesystem is not of the default NFS type. This option is processed by the automounter and not by the mount command.

-strict

is used to treat errors when mounting file systems as fatal. This is important when multiple file systems should be mounted (‘multi-mounts’). If this option is given, no file system is mounted at all if at least one file system can’t be mounted.

-use-weight-only

is used to make the weight the sole factor in selecting a server when multiple servers are present in a map entry. and

-no-use-weight-only

can be used to negate the option if it is present in the master map entry for the map but is not wanted for the given mount.

location

The location specifies from where the file system is to be mounted. In the most cases this will be an NFS volume and the usual notation *host:pathname* is used to indicate the remote filesystem and path to be mounted. If the filesystem to be mounted begins with a / (such as local */dev* entries or smbfs shares) a : needs to be prefixed (e.g. *:/dev/sda1*).

EXAMPLE

Indirect map:

```
kernel          -ro,soft,intr                   ftp.kernel.org:/pub/linux
```

```
boot        -fstype=ext2            :/dev/hda1
windoze     -fstype=smbfs           ://windoze/c
removable   -fstype=ext2            :/dev/hdd
cd          -fstype=iso9660,ro      :/dev/hdc
floppy      -fstype=auto            :/dev/fd0
server      -rw,hard,intr           / -ro myserver.me.org:/ \
                                    /usr myserver.me.org:/usr \
                                    /home myserver.me.org:/home
```

In the first line we have a NFS remote mount of the kernel directory on *ftp.kernel.org*. This is mounted read-only. The second line mounts an ext2 volume from a local ide drive. The third makes a share exported from a Windows machine available for automounting. The rest should be fairly self-explanatory. The last entry (the last three lines) is an example of a multi-map (see below).

If you use the automounter for a filesystem without access permissions (like **vfat**), users usually can't write on such a filesystem because it is mounted as user **root**. You can solve this problem by passing the option *gid=<gid>*, e.g. *gid=floppy*. The filesystem is then mounted as group **floppy** instead of **root**. Then you can add the users to this group, and they can write to the filesystem. Here's an example entry for an autofs map:

```
floppy-vfat  -fstype=vfat,sync,gid=floppy,umask=002  :/dev/fd0
```

Direct map:

```
/nfs/apps/mozilla bogus:/usr/local/moxill
/nfs/data/budgets tiger:/usr/local/budgets
/tst/sbin                                   bogus:/usr/sbin
```

FEATURES

Map Key Substitution

An & character in the **location** is expanded to the value of the **key** field that matched the line (which probably only makes sense together with a wildcard key).

Wildcard Key

A map key of * denotes a wild-card entry. This entry is consulted if the specified key does not exist in the map. A typical wild-card entry looks like this:

```
*           server:/export/home/&
```

The special character '&' will be replaced by the provided key. So, in the example above, a lookup for the key 'foo' would yield a mount of server:/export/home/foo.

Variable Substitution

The following special variables will be substituted in the location field of an automounter map entry if prefixed with $ as customary from shell scripts (curly braces can be used to separate the field name):

ARCH	Architecture (uname -m)
CPU	Processor Type
HOST	Hostname (uname -n)
OSNAME	Operating System (uname -s)
OSREL	Release of OS (uname -r)
OSVERS	Version of OS (uname -v)

autofs provides additional variables that are set based on the user requesting the mount:

USER	The user login name

UID	The user login ID
GROUP	The user group name
GID	The user group ID
HOME	The user home directory
SHOST	Short hostname (domain part removed if present)

If a program map is used these standard environment variables will have a prefix of "AUTOFS_" to prevent interpreted languages like python from being able to load and execute arbitray code from a user home directory.

Additional entries can be defined with the -Dvariable=Value map-option to **automount**(8).

Executable Maps

A map can be marked as executable. A **program** map will be called with the key as an argument. It may return no lines of output if there's an error, or one or more lines containing a map entry (with \ quoting line breaks). The map entry corresponds to what would normally follow a map key.

An executable map can return an error code to indicate the failure in addition to no output at all. All output sent to stderr is logged into the system logs.

Multiple Mounts

A **multi-mount map** can be used to name multiple filesystems to mount. It takes the form:

key [-*options*] **[mount-point [**-*options*] **location...]...**

This may extend over multiple lines, quoting the line-breaks with `\´. If present, the per-mountpoint mount-options are appended to the default mount-options.

Replicated Server

Multiple replicated hosts, same path:
<path> host1,host2,hostn:/path/path

Multiple hosts, some with same path, some with another
<path> host1,host2:/blah host3:/some/other/path

Multiple replicated hosts, different (potentially) paths:
<path> host1:/path/pathA host2:/path/pathB

Mutliple weighted, replicated hosts same path:
<path> host1(5),host2(6),host3(1):/path/path

Multiple weighted, replicated hosts different (potentially) paths:
<path> host1(3):/path/pathA host2(5):/path/pathB

Anything else is questionable and unsupported, but these variations will also work:
<path> host1(3),host:/blah

UNSUPPORTED

This version of the automounter supports direct maps stored in FILE, NIS, NISPLUS and LDAP only.

AMD FORMAT

This is a description of the text file format. Other methods of specifying mount map entries may be required for different map sources. All empty lines or lines beginning with # are ignored. The basic format of one line in such maps is:

key location-list

key

A **key** is a path (or a single path component alone) that may end in the wildcard key, "*", or the wildcard key alone and must not begin with the "/" character.

location-list

Following the **key** is a mount **location-list**.

A **location-list** list has the following syntax:

location[location[...]] [|| location[location[...]]

A mount **location-list** can use the cut operator, ||, to specify locations that should be tried if none of the locations to the left of it where selected for a mount attempt.

A mount **location** consists of an optional colon seperated list of **selectors**, followed by a colon seperated list of **option:=value** pairs.

The **selectors** that may be used return a value or boolean result. Those that return a value may be to used with the comparison operators **==** and **!=** and those that return a boolean result may be negated with the **!**.

For a **location** to be selected for a mount attempt all of its **selectors** must evaluate to true. If a **location** is selected for a mount attempt and succeeds the lookup is completed and returns success. If the mount attempt fails the proceedure continues with the next **location** until they have all been tried.

In addition some **selectors** take no argumenets, some one argument and others optionally take two arguments.

The **selectors** that take no arguments are:

arch

The machine architecture which, if not set in the confugration, is obtained using uname(2).

karch

The machine kernel architecture which, if not set in the confugration, is obtained using uname(2).

os

The operating system name, if not set in the confugration, is obtained using uname(2).

osver

The operating system version, if not set in the confugration, is obtained using uname(2).

full_os

The full operating system name, if not set in the confugration this selector has no value.

vendor

The operating system vendor name, if not set in the confugration this selector has the value "unknown".

byte

The endianess of the hardware.

cluster

The name of the local cluster. It has a value only if it is set in the configuration.

autodir

The base path under which external mounts are done if they are needed. Most mounts are done in place but some can’t be and this is the base path under which those mounts will be done.

domain
The local domain name. It is set to the value of the configuration option **sub_domain**. If sub_domain is not given in the configuration it is set to the domain part of the local host name, as given by gethostname(2).

host
The local host name, without the domain part, as given by gethostname(2).

hostd
The full host name. If **sub_domain** is given in the configuration this is set to the contatenation of **host** and **sub_domain** deperated by a **.**. If **sub_domain** is not set in the configuration the value of **domain** is used instead of **sub_domain**.

uid
The numeric value of the uid of the user that first requested the mount. Note this is usual the same as that used by amd but can be different within autofs.

gid
The numeric value of the gid of the user that first requested the mount. Note this is usual the same as that used by amd but can be different within autofs.

key
The string value of the key being looked up.

map
The string value of the map name used to lookup **key**s.

path
The string value of the full path to the mount being requested.

dollar
Evaluates to the string "$".

The **selectors** that take one argument are:

in_network(network) , network(network) , netnumber(network) , wire(network)
These **selectors** are all the same. **in_network()** is the preferred usage. The **network** argument is an address (which may include a subnet mask) or network name. The function compares **network** against each interface and returns true if **network** belongs to the network the interface is connected to.

xhost(hostname)
The **xhost()** selector compares **hostname** to the **${host}** and if it doesn't match it attempts to lookup the cannonical name of **hostname** and compares it to {host} as well.

exists(filename)
Returns true if **filename** exits as determined by lstat(2).

true()
Evaluates to true, the argument is ignored and may be empty.

false()
Evaluates to false, the argument is ignored and may be empty.

The **selectors** that take up to two arguments are:

netgrp(netgroup[,hostname])
The **netgrp()** selector returns true if **hostname** is a member of the netgroup **netgroup**. If **hostname** is not given **${host}** is used for the comparison.

netgrpd(netgroup[,hostname])
The **netgrpd()i** selector behaves the same as **netgrp()** except that if **hostname** is not given **${hostd}**, the fully qualified hostname, is used instead of **${host}**.

The **options** that may be used are:

type
This is the mount filesystem **type**. It can have a value of **auto**, **link**, **linkx**, **host**, **lofs**, **ext2-4**, **xfs**, **nfs**, **nfsl** or **cdfs**. Other types that are not yet implemented or are not available iin autofs are **nfsx**, **lustre**, **jfs**, **program**, **cachefs** and **direct**.

maptype
The **maptype** option specifies the type of the map source and can have a value of **file**, **nis**, **nisplus**, **exec**, **ldap** or **hesiod**. Map sources either not yet implemented or not available in autofs are **sss**, **ndbm**, **passwd** and **union**.

fs
The option **fs** is used to specify the local filesystem. The meaning of this option (and whether or not it is used) is dependent on the mount filesystem **type**.

rhost
The remote host name for network mount requests.

rfs
The remote host filesystem path for network mount requests.

dev
Must resolve to the device file for local device mount requests.

sublink
The **sublink** option is used to specify a subdirectory within the mount location to which this entry will point.

pref
The **pref** option is used to specify a prefix that is prepended to the lookup key before looking up the map entry key.

opts
The **opts** option is used to specify mount options to be used for the mount. If a "**-**" is given it is ignored. Options that may be used are dependend on the mount filesystem.

addopts
The **addopts** option is used to specify additional mount options used in addition to the default mount options for the mount location.

remopts
The **addopts** option is used to specify mount options used instead the options given in **opts** when the mount location is on a remote retwork.

A number of **options** aren't available or aren't yet implemented
within autofs, these are:

cache
The **cache option** isn't used by autofs. The map entry cache is continually updated and stale entries cleaned on re-load when map changes are detected so these configuration entries are not used. The **regex** map key matching is not implemented and may not be due to the potential overhead of the full map scans needed on every key lookup.

cachedir
The **cachefs** filesystem is not available on Linux, a different implementation is used for caching network mounted file systems.

mount , unmount , umount
These **options** are used by the amd **program** mount type which is not yet implemented.

delay
This **option** is not used by the autofs implementation and is ignored.

FEATURES

Key Matching

The amd parser key matching is unusual.

The key string to be looked up is constructed by prepending the prefix, if there is one.

The resulting relative path string is matched by first trying the sting itself. If no match is found the last component of the key string is replaced with the wilcard match cahracter ("*") and a wildcard match is attemted. This process continues until a match is found or until the last match, against the wilcard match key alone, fails to match a map entry and the key lookup fails.

Macro Usage

Macros are used a lot in the autofs amd implementation.

Many of the option values are set as macro variables corresponding to the option name during the map entry parse. So they may be used in subsequent option values. Beware though, the order in which option values is not necessarily left to right so you may get unexpected results.

EXAMPLE

Example NFS mount map:

Assuming we have the autofs master map entry:

```
/test           file,amd:/etc/amd.test
```

And the following map in /etc/amd.test:

```
/defaults       type:=nfs;rhost:=bilbo
apps            rfs:=/autofs
util            rhost:=zeus;rfs:=/work/util
local           rfs:=/shared;sublink:=local
```

In the first line we have an NFS remote mount of the exported directory /autofs from host bilbo which would be mounted on /test/apps. Next another nfs mount for the exported directory /work/util from host zeus. This would be mounted on /test/util.

Finally we have an example of the use of the **sublink** option. In this case the filesystem bilbo:/shared would be mounted on a path external the automount directory (under the direcory given by configuration option auto_dir) and the path /test/local either symlinked or bind mounted (depending on the setting autofs_use_lofs) to the "local" subdirectory of the external mount.

SEE ALSO

automount(8), **auto.master**(5), **autofs**(8), **autofs.conf**(5), **mount**(8). **autofs_ldap_auth.conf**(5)

AUTHOR

This manual page was written by Christoph Lameter <chris@waterf.org>, for the Debian GNU/Linux system. Edited by H. Peter Avian <hpa@transmeta.com>, Jeremy Fitzhardinge <jeremy@goop.org> and Ian Kent <raven@themaw.net>.