Contents

[/etc/services 1](#_Toc312939220)

[/etc/nsswitch.conf 3](#_Toc312939221)

[/etc/hosts 6](#_Toc312939222)

[/etc/resolv.conf 8](#_Toc312939223)

[/etc/fstab 11](#_Toc312939224)

/etc/services: **services** is a plain ASCII file providing a mapping between friendly textual names for internet services, and their underlying assigned port numbers and protocol types. Every networking program should look into this file to get the port number (and protocol) for its service. The C library routines **[getservent](http://linux.about.com/library/cmd/blcmdl3_getservent.htm)**(3), [**getservbyname**](http://linux.about.com/library/cmd/blcmdl3_getservbyname.htm)(3), [**getservbyport**](http://linux.about.com/library/cmd/blcmdl3_getservbyport.htm)(3), [**setservent**](http://linux.about.com/library/cmd/blcmdl3_setservent.htm)(3), and **[endservent](http://linux.about.com/library/cmd/blcmdl3_endservent.htm)**(3) support querying this file from programs.

For example, a POP3 email daemon would do a getportbyname ("pop3") in order to retrieve the number 110 that pop3 runs at. The idea is that if all POP3 daemons use getportbyname(), then no matter what POP3 daemon you run, you can always reconfigure its port number by editing / etc/services.

Port numbers are assigned by the IANA (Internet Assigned Numbers Authority), and their current policy is to assign both TCP and UDP protocols when assigning a port number. Therefore, most entries will have two entries, even for TCP only services.

Port numbers below 1024 (so-called 'low numbered' ports) can only be bound to by root (see [**bind**](http://linux.about.com/library/cmd/blcmdl2_bind.htm)(2), [**tcp**](http://linux.about.com/library/cmd/blcmdl7_tcp.htm)(7), and [**udp**](http://linux.about.com/library/cmd/blcmdl7_udp.htm)(7)). This is so clients connecting to low numbered ports can trust that the service running on the port is the standard implementation, and not a rogue service run by a user of the machine. Well-known port numbers specified by the IANA are normally located in this root-only space.

The presence of an entry for a service in the **services** file does not necessarily mean that the service is currently running on the machine. See **inetd.conf**(5) for the configuration of Internet services offered. Note that not all networking services are started by **inetd**(8), and so won't appear in [**inetd.conf**](http://linux.about.com/library/cmd/blcmdl8_xinetd.htm)(5). In particular, news (NNTP) and mail (SMTP) servers are often initialized from the system boot scripts.

The location of the **services** file is defined by **\_PATH\_SERVICES** in */usr/include/netdb.h*. This is usually set to */etc/services*.

Each line describes one service, and is of the form:

*service-name   port***/***protocol*[*aliases ...*]

where:

*service-name*

is the friendly name the service is known by and looked up under. It is case sensitive. Often, the client program is named after the *service-name*.

*port*

is the port number (in decimal) to use for this service.

*protocol*

is the type of protocol to be used. This field should match an entry in the **[protocols](http://linux.about.com/library/cmd/blcmdl5_protocols.htm)**(5) file. Typical values include **tcp** and **udp**.

*aliases*

is an optional space or tab separated list of other names for this service (but see the BUGS section below). Again, the names are case sensitive.

Either spaces or tabs may be used to separate the fields.

Comments are started by the hash sign (#) and continue until the end of the line. Blank lines are skipped.

The *service-name* should begin in the first column of the file, since leading spaces are not stripped. *service-names* can be any printable characters excluding space and tab. However, a conservative choice of characters should be used to minimize inter-operability problems. E.g., a-z, 0-9, and hyphen (-) would seem a sensible choice.

Lines not matching this format should not be present in the file. (Currently, they are silently skipped by [**getservent**](http://linux.about.com/library/cmd/blcmdl3_getservent.htm)(3), [**getservbyname**](http://linux.about.com/library/cmd/blcmdl3_getservbyname.htm)(3), and [**getservbyport**](http://linux.about.com/library/cmd/blcmdl3_getservbyport.htm)(3). However, this behaviour should not be relied on.)

As a backwards compatibility feature, the slash (/) between the *port* number and *protocol* name can in fact be either a slash or a comma (,). Use of the comma in modern installations is depreciated.

This file might be distributed over a network using a network-wide naming service like Yellow Pages/NIS or BIND/Hesiod.

A sample **services** file might look like this:

netstat 15/tcp

qotd 17/tcp quote

msp 18/tcp # message send protocol

msp 18/udp # message send protocol

chargen 19/tcp ttytst source

chargen 19/udp ttytst source

ftp 21/tcp

# 22 - unassigned

telnet 23/tcp

/etc/nsswitch.conf - System Databases and Name Service Switch configuration file

The sources for the "databases" and their lookup order are specified in the **/etc/nsswitch.conf** file.

The following databases are available in the NSS:

**aliases**

Mail aliases, used by [**sendmail**](http://linux.about.com/library/cmd/blcmdl8_sendmail.htm)(8). Presently ignored.

**ethers**

Ethernet numbers.

**group**

Groups of users, used by [**getgrent**](http://linux.about.com/library/cmd/blcmdl3_getgrent.htm)(3) functions.

**hosts**

Host names and numbers, used by [**gethostbyname**](http://linux.about.com/library/cmd/blcmdl3_gethostbyname.htm)(3) and similar functions.

**netgroup**

Network wide list of hosts and users, used for access rules. C libraries before glibc 2.1 only support netgroups over NIS.

**network**

Network names and numbers, used by [**getnetent**](http://linux.about.com/library/cmd/blcmdl3_getnetent.htm)(3) functions.

**passwd**

User passwords, used by [**getpwent**](http://linux.about.com/library/cmd/blcmdl3_getpwent.htm)(3) functions.

**protocols**

Network protocols, used by [**getprotoent**](http://linux.about.com/library/cmd/blcmdl3_getprotoent.htm)(3) functions.

**publickey**

Public and secret keys for Secure\_RPC used by NFS and NIS+.

**rpc**

Remote procedure call names and numbers, used by [**getrpcbyname**](http://linux.about.com/library/cmd/blcmdl3_getrpcbyname.htm)(3) and similar functions.

**services**

Network services, used by [**getservent**](http://linux.about.com/library/cmd/blcmdl3_getservent.htm)(3) functions.

**shadow**

Shadow user passwords, used by [**getspnam**](http://linux.about.com/library/cmd/blcmdl3_getspnam.htm)(3).

An example **/etc/nsswitch.conf** (namely, the default used when **/etc/nsswitch.conf** is missing):

passwd:

compat

group:

compat

shadow:

compat

hosts:

dns [!UNAVAIL=return] files

networks:

nis [NOTFOUND=return] files

ethers:

nis [NOTFOUND=return] files

protocols:

nis [NOTFOUND=return] files

rpc:

nis [NOTFOUND=return] files

services:

nis [NOTFOUND=return] files

The first column is the database. The rest of the line specifies how the lookup process works. You can specify the way it works for each database individually.

The configuration specification for each database can contain two different items:

\* The service specification like `files', `db', or `nis'.

\* The reaction on lookup result like `[NOTFOUND=return]'.

For libc5 with NYS, the allowed service specifications are `files', `nis', and `nisplus'. For hosts, you could specify `dns' as extra service, for passwd and group `compat', but not for shadow.

For glibc, you must have a file called **/lib/libnss\_SERVICE.so.***X* for every SERVICE you are using. On a standard installation, you could use `files', `db', `nis', and `nisplus'. For hosts, you could specify `dns' as extra service, for passwd, group, and shadow `compat'. These services will not be used by libc5 with NYS. The version number *X* is 1 for glibc 2.0 and 2 for glibc 2.1.

The second item in the specification gives the user much finer control on the lookup process. Action items are placed between two service names and are written within brackets. The general form is

`[' ( `!'? STATUS `=' ACTION )+ `]'

where

STATUS => success | notfound | unavail | tryagain

ACTION => return | continue

The case of the keywords is insignificant. The STATUS values are the results of a call to a lookup function of a specific service. They mean:

**success**

No error occurred and the wanted entry is returned. The default action for this is `return'.

**notfound**

The lookup process works ok but the needed value was not found. The default action is `continue'.

**unavail**

The service is permanently unavailable. This can either mean the needed file is not available, or, for DNS, the server is not available or does not allow queries. The default action is `continue'.

**tryagain**

The service is temporarily unavailable. This could mean a file is locked or a server currently cannot accept more connections. The default action is `continue'.

hosts - The static table lookup for host names

# /etc/hosts

DESCRIPTION

This manual page describes the format of the */etc/hosts* file. This file is a simple text file that associates IP addresses with hostnames, one line per IP address. For each host a single line should be present with the following information:

IP\_address canonical\_hostname aliases

Fields of the entry are separated by any number of blanks and/or tab characters. Text from a "#" character until the end of the line is a comment, and is ignored. Host names may contain only alphanumeric characters, minus signs ("-"), and periods ("."). They must begin with an alphabetic character and end with an alphanumeric character. Aliases provide for name changes, alternate spellings, shorter hostnames, or generic hostnames (for example, *localhost*). The format of the host table is described in RFC 952.

The Berkeley Internet Name Domain (BIND) Server implements the Internet name server for UNIX systems. It augments or replaces the */etc/hosts* file or host name lookup, and frees a host from relying on */etc/hosts* being up to date and complete.

In modern systems, even though the host table has been superseded by DNS, it is still widely used for:

**bootstrapping**

Most systems have a small host table containing the name and address information for important hosts on the local network. This is useful when DNS is not running, for example during system bootup.

**NIS**

Sites that use NIS use the host table as input to the NIS host database. Even though NIS can be used with DNS, most NIS sites still use the host table with an entry for all local hosts as a backup.

**isolated nodes**

Very small sites that are isolated from the network use the host table instead of DNS. If the local information rarely changes, and the network is not connected to the Internet, DNS offers little advantage.

EXAMPLE

127.0.0.1 localhost

192.168.1.10 foo.mydomain.org foo

192.168.1.13 bar.mydomain.org bar

216.234.231.5 master.debian.org master

205.230.163.103 www.opensource.org

resolver - resolver configuration file

# /etc/resolv.conf

DESCRIPTION

The **resolver** is a set of routines in the C library (resolve(3) ) that provide access to the Internet Domain Name System. The **resolver** configuration file contains information that is read by the **resolver** routines the first time they are invoked by a process. The file is designed to be human readable and contains a list of keywords with values that provide various types of **resolver** information.

On a normally configured system, this file should not be necessary. The only name server to be queried will be on the local machine, the domain name is determined from the host name, and the domain search path is constructed from the domain name.

The different configuration directives are:

**nameserver**

Internet address (in dot notation) of a name server that the **resolver** should query. Up to **MAXNS** (see <[resolv.h](file:///\\\\usr\\include\\resolv.h)> name servers may be listed, one per keyword. If there are multiple servers, the **resolver** library queries them in the order listed. If no **nameserver** entries are present, the default is to use the name server on the local machine. (The algorithm used is to try a name server, and if the query times out, try the next, until out of name servers, then repeat trying all the name servers until a maximum number of retries are made).

**domain**

Local domain name. Most queries for names within this domain can use short names relative to the local domain. If no **domain** entry is present, the domain is determined from the local host name returned by [gethostname](http://linux.about.com/library/cmd/blcmdl2_gethostname.htm)(2); the domain part is taken to be everything after the first `.' Finally, if the host name does not contain a domain part, the root domain is assumed.

**search**

Search list for host-name lookup. The search list is normally determined from the local domain name; by default, it contains only the local domain name. This may be changed by listing the desired domain search path following the **search** keyword with spaces or tabs separating the names. Most **resolver** queries will be attempted using each component of the search path in turn until a match is found. Note that this process may be slow and will generate a lot of network traffic if the servers for the listed domains are not local, and that queries will time out if no server is available for one of the domains.

The search list is currently limited to six domains with a total of 256 characters.

**sortlist**

Allows addresses returned by gethostbyname to be sorted. A **sortlist** is specified by IP address netmask pairs. The netmask is optional and defaults to the natural netmask of the net. The IP address and optional network pairs are separated by slashes. Up to 10 pairs may be specified. For example:

sortlist 130.155.160.0/255.255.240.0 130.155.0.0

**options**

Allows certain internal **resolver** variables to be modified. The syntax is where *option* is one of the following:

**debug**

sets **RES\_DEBUG** in Ft \_res.options .

**ndots:** *n*

sets a threshold for the number of dots which must appear in a name given to Fn res\_query (see [resolver](http://linux.about.com/library/cmd/blcmdl3_resolver.htm)(3)) before an *initial absolute query* will be made. The default for *n* is ``1 , '' meaning that if there are *any* dots in a name, the name will be tried first as an absolute name before any *search list* elements are appended to it.

**timeout:** *n*

sets the amount of time the resolver will wait for a response from a remote name server before retrying the query via a different name server. Measured in seconds, the default is **RES\_TIMEOUT** (see <[resolv.h](file:///\\\\usr\\include\\resolv.h)> ).

**attempts:** *n*

sets the number of times the resolver will send a query to its name servers before giving up and returning an error to the calling application. The default is **RES\_DFLRETRY** (see <[resolv.h](file:///\\\\usr\\include\\resolv.h)> ).

**rotate**

sets **RES\_ROTATE** in Ft \_res.options , which causes round robin selection of nameservers from among those listed. This has the effect of spreading the query load among all listed servers, rather than having all clients try the first listed server first every time.

**no-check-names**

sets **RES\_NOCHECKNAME** in Ft \_res.options , which disables the modern BIND checking of incoming host names and mail names for invalid characters such as underscore (\_), non-ASCII, or control characters.

**inet6**

sets **RES\_USE\_INET6** in Ft \_res.options . This has the effect of trying a AAAA query before an A query inside the Ft gethostbyname function, and of mapping IPv4 responses in IPv6 ``tunnelled form'' if no AAAA records are found but an A record set exists.

The **domain** and **search** keywords are mutually exclusive. If more than one instance of these keywords is present, the last instance wins.

The **search** keyword of a system's resolv.conf file can be overridden on a per-process basis by setting the environment variable ``**LOCALDOMAIN** '' to a space-separated list of search domains.

The **options** keyword of a system's resolv.conf file can be amended on a per-process basis by setting the environment variable ``**RES\_OPTIONS to a space-separated list of** '' **resolver** options as explained above under **options**

The keyword and value must appear on a single line, and the keyword (e.g., **nameserver )** must start the line. The value follows the keyword, separated by white space.

/etc/fstab - static information about the filesystems

DESCRIPTION

The file **fstab** contains descriptive information about the various file systems. **fstab** is only read by programs, and not written; it is the duty of the system administrator to properly create and maintain this file. Each filesystem is described on a separate line; fields on each line are separated by tabs or spaces. Lines starting with '#' are comments. The order of records in **fstab** is important because **[fsck](http://linux.about.com/library/cmd/blcmdl8_fsck.htm)**(8), [**mount**](http://linux.about.com/library/cmd/blcmdl8_mount.htm)(8), and **[umount](http://linux.about.com/library/cmd/blcmdl8_umount.htm)**(8) sequentially iterate through **fstab** doing their thing.

The first field, (*fs\_spec*), describes the block special device or remote filesystem to be mounted.

For ordinary mounts it will hold (a link to) a block special device node (as created by [**mknod**](http://linux.about.com/library/cmd/blcmdl1_mknod.htm)(8)) for the device to be mounted, like `/dev/cdrom' or `/dev/sdb7'. For NFS mounts one will have host:dir, e.g., `knuth.aeb.nl:/'. For procfs, use `proc'.

Instead of giving the device explicitly, one may indicate the (ext2 or xfs) filesystem that is to be mounted by its UUID or volume label (cf. [**e2label**](http://linux.about.com/library/cmd/blcmdl8_e2label.htm)(8) or **xfs\_admin**(8)), writing LABEL=label or UUID=uuid, e.g., `LABEL=Boot' or `UUID=3e6be9de-8139-11d1-9106-a43f08d823a6'. This will make the system more robust: adding or removing a SCSI disk changes the disk device name but not the filesystem volume label.

The second field, (*fs\_file*), describes the mount point for the filesystem. For swap partitions, this field should be specified as `none'. If the name of the mount point contains spaces these can be escaped as `\040'.

The third field, (*fs\_vfstype*), describes the type of the filesystem. Linux supports lots of filesystem types, such as *adfs*, *affs*, *autofs*, *coda*, *coherent*, *cramfs*, *devpts*, *efs*, *ext2*, *ext3*, *hfs*, *hpfs*, *iso9660*, *jfs*, *minix*, *msdos*, *ncpfs*, *nfs*, *ntfs*, *proc*, *qnx4*, *reiserfs*, *romfs*, *smbfs*, *sysv*, *tmpfs*, *udf*, *ufs*, *umsdos*, *vfat*, *xenix*, *xfs*, and possibly others. For more details, see [**mount**](http://linux.about.com/library/cmd/blcmdl8_mount.htm)(8). For the filesystems currently supported by the running kernel, see */proc/filesystems*. An entry *swap* denotes a file or partition to be used for swapping, cf. [**swapon**](http://linux.about.com/library/cmd/blcmdl8_swapon.htm)(8). An entry *ignore* causes the line to be ignored. This is useful to show disk partitions which are currently unused.

The fourth field, (*fs\_mntops*), describes the mount options associated with the filesystem.

It is formatted as a comma separated list of options. It contains at least the type of mount plus any additional options appropriate to the filesystem type. For documentation on the available options for non-nfs file systems, see [**mount**](http://linux.about.com/library/cmd/blcmdl8_mount.htm)(8). For documentation on all nfs-specific options have a look at [**nfs**](http://linux.about.com/library/cmd/blcmdl5_nfs.htm)(5). Common for all types of file system are the options ``noauto'' (do not mount when "mount -a" is given, e.g., at boot time), ``user'' (allow a user to mount), and ``owner'' (allow device owner to mount), and ``\_netdev'' (device requires network to be available). The ``owner'' and ``\_netdev'' options are Linux-specific. For more details, see [**mount**](http://linux.about.com/library/cmd/blcmdl8_mount.htm)(8).

The fifth field, (*fs\_freq*), is used for these filesystems by the [**dump**](http://linux.about.com/library/cmd/blcmdl8_dump.htm)(8) command to determine which filesystems need to be dumped. If the fifth field is not present, a value of zero is returned and **dump** will assume that the filesystem does not need to be dumped.

The sixth field, (*fs\_passno*), is used by the [**fsck**](http://linux.about.com/library/cmd/blcmdl8_fsck.htm)(8) program to determine the order in which filesystem checks are done at reboot time. The root filesystem should be specified with a *fs\_passno* of 1, and other filesystems should have a *fs\_passno* of 2. Filesystems within a drive will be checked sequentially, but filesystems on different drives will be checked at the same time to utilize parallelism available in the hardware. If the sixth field is not present or zero, a value of zero is returned and **fsck** will assume that the filesystem does not need to be checked.