**NAME**

named − Internet domain name server

**SYNOPSIS**

**named** [ **−d** *debuglevel* ] [ **−p** *port#* ] [{−b} *bootfile* ]

**DESCRIPTION**

*Named* is the Internet domain name server. See RFC883 for more information on the Internet name-domain system. Without any arguments, *named* will read the default boot file */etc/named.boot*, read any initial data and listen for queries.

Options are:

**−d** Print debugging information. A number after the ‘‘d’’ determines the level of messages printed.

**−p** Use a different port number. The default is the standard port number as listed in /etc/services.

**−b** Use an alternate boot file. This is optional and allows you to specify a file with a leading dash.

Any additional argument is taken as the name of the boot file. The boot file contains information about where the name server is to get its initial data. If multiple boot files are specified, only the last is used. Lines in the boot file cannot be continued on subsequent lines. The following is a small example:

```
;
;        boot file for name server
;
directory        /usr/local/domain

; type      domain                source host/file              backup file

cache       .                                                   root.cache
primary     Berkeley.EDU          berkeley.edu.zone
primary     32.128.IN-ADDR.ARPA   ucbhosts.rev
secondary   CC.Berkeley.EDU       128.32.137.8 128.32.137.3     cc.zone.bak
secondary   6.32.128.IN-ADDR.ARPA 128.32.137.8 128.32.137.3     cc.rev.bak
primary     0.0.127.IN-ADDR.ARPA                                localhost.rev
forwarders  10.0.0.78 10.2.0.78
; slave
```

The ‘‘directory’’ line causes the server to change its working directory to the directory specified. This can be important for the correct processing of $INCLUDE files in primary zone files.

The ‘‘cache’’ line specifies that data in ‘‘root.cache’’ is to be placed in the backup cache. Its main use is to specify data such as locations of root domain servers. This cache is not used during normal operation, but is used as ‘‘hints’’ to find the current root servers. The file ‘‘root.cache’’ is in the same format as ‘‘berkeley.edu.zone’’. There can be more than one ‘‘cache’’ file specified. The cache files are processed in such a way as to preserve the time-to-live’s of data dumped out. Data for the root nameservers is kept artificially valid if necessary.

The first ‘‘primary’’ line states that the file ‘‘berkeley.edu.zone’’ contains authoritative data for the ‘‘Berkeley.EDU’’ zone. The file ‘‘berkeley.edu.zone’’ contains data in the master file format described in RFC883. All domain names are relative to the origin, in this case, ‘‘Berkeley.EDU’’ (see below for a more detailed description). The second ‘‘primary’’ line states that the file ‘‘ucbhosts.rev’’ contains authoritative data for the domain ‘‘32.128.IN-ADDR.ARPA,’’ which is used to translate addresses in network 128.32 to hostnames. Each master file should begin with an SOA record for the zone (see below).

The first ‘‘secondary’’ line specifies that all authoritative data under ‘‘CC.Berkeley.EDU’’ is to be transferred from the name server at 128.32.137.8. If the transfer fails it will try 128.32.137.3 and continue trying the addresses, up to 10, listed on this line. The secondary copy is also authoritative for the specified domain. The first non-dotted-quad address on this line will be taken as a filename in which to backup the transfered zone. The name server will load the zone from this backup file if it exists when it boots, providing a complete copy even if the master servers are unreachable. Whenever a new copy of the domain is received by automatic zone transfer from one of the master servers, this file will be updated. The second "secondary" line states that the address-to-hostname mapping for the subnet 128.32.136 should be obtained from the same list of master servers as the previous zone.

The "forwarders" line specifies the addresses of sitewide servers that will accept recursive queries from other servers. If the boot file specifies one or more forwarders, then the server will send all queries for data not in the cache to the forwarders first. Each forwarder will be asked in turn until an answer is returned or the list is exhausted. If no answer is forthcoming from a forwarder, the server will continue as it would have without the forwarders line unless it is in "slave" mode. The forwarding facility is useful to cause a large sitewide cache to be generated on a master, and to reduce traffic over links to outside servers. It can also be used to allow servers to run that do not have access directly to the Internet, but wish to act as though they do.

The "slave" line (shown commented out) is used to put the server in slave mode. In this mode, the server will only make queries to forwarders. This option is normally used on machine that wish to run a server but for physical or administrative reasons cannot be given access to the Internet, but have access to a host that does have access.

The "sortlist" line can be used to indicate networks that are to be preferred over other, unlisted networks. Queries for host addresses from hosts on the same network as the server will receive responses with local network addresses listed first, then addresses on the sort list, then other addresses. This line is only acted on at initial startup. When reloading the nameserver with a SIGHUP, this line will be ignored.

The master file consists of control information and a list of resource records for objects in the zone of the forms:

```
$INCLUDE <filename> <opt_domain>
$ORIGIN <domain>
<domain> <opt_ttl> <opt_class> <type> <resource_record_data>
```

where *domain* is "." for root, "@" for the current origin, or a standard domain name. If *domain* is a standard domain name that does not end with ".", the current origin is appended to the domain. Domain names ending with "." are unmodified. The *opt_domain* field is used to define an origin for the data in an included file. It is equivalent to placing a $ORIGIN statement before the first line of the included file. The field is optional. Neither the *opt_domain* field nor $ORIGIN statements in the included file modify the current origin for this file. The *opt_ttl* field is an optional integer number for the time-to-live field. It defaults to zero, meaning the minimum value specified in the SOA record for the zone. The *opt_class* field is the object address type; currently only one type is supported, **IN**, for objects connected to the DARPA Internet. The *type* field contains one of the following tokens; the data expected in the *resource_record_data* field is in parentheses.

A a host address (dotted quad)

NS an authoritative name server (domain)

MX a mail exchanger (domain)

CNAME the canonical name for an alias (domain)

SOA marks the start of a zone of authority (domain of originating host, domain address of maintainer, a serial number and the following parameters in seconds: refresh, retry, expire and minimum TTL (see RFC883))

MB a mailbox domain name (domain)

MG a mail group member (domain)

MR a mail rename domain name (domain)

NULL a null resource record (no format or data)

WKS a well know service description (not implemented yet)

PTR a domain name pointer (domain)

HINFO host information (cpu_type OS_type)

MINFO mailbox or mail list information (request_domain error_domain)

Resource records normally end at the end of a line, but may be continued across lines between opening and closing parentheses. Comments are introduced by semicolons and continue to the end of the line.

Each master zone file should begin with an SOA record for the zone. An example SOA record is as follows:

```
@       IN      SOA     ucbvax.Berkeley.EDU. rwh.ucbvax.Berkeley.EDU. (
                                2.89       ; serial
                                10800      ; refresh
                                3600       ; retry
                                3600000    ; expire
                                86400 )    ; minimum
```

The SOA lists a serial number, which should be changed each time the master file is changed. Secondary servers check the serial number at intervals specified by the refresh time in seconds; if the serial number changes, a zone transfer will be done to load the new data. If a master server cannot be contacted when a refresh is due, the retry time specifies the interval at which refreshes should be attempted until successful. If a master server cannot be contacted within the interval given by the expire time, all data from the zone is discarded by secondary servers. The minimum value is the time-to-live used by records in the file with no explicit time-to-live value.

## NOTES

The boot file directives "domain" and "suffixes" have been obsoleted by a more useful resolver based implementation of suffixing for partially qualified domain names. The prior mechanisms could fail under a number of situations, especially when then local nameserver did not have complete information.

The following signals have the specified effect when sent to the server process using the *kill*(1) command.

SIGHUP
: Causes server to read named.boot and reload database.

SIGINT
: Dumps current data base and cache to /usr/tmp/named_dump.db

SIGIOT
: Dumps statistics data into /usr/tmp/named.stats if the server is compiled -DSTATS. Statistics data is appended to the file.

SIGSYS
: Dumps the profiling data in /usr/tmp if the server is compiled with profiling (server forks, chdirs and exits).

SIGTERM
: Dumps the primary and secondary database files. Used to save modified data on shutdown if the server is compiled with dynamic updating enabled.

SIGUSR1
: Turns on debugging; each SIGUSR1 increments debug level. (SIGEMT on older systems without SIGUSR1)

SIGUSR2
: Turns off debugging completely. (SIGFPE on older systems without SIGUSR2)

**FILES**

| | |
|---|---|
| /etc/named.boot | name server configuration boot file |
| /etc/named.pid | the process id |
| /usr/tmp/named.run | debug output |
| /usr/tmp/named_dump.db | dump of the name server database |
| /usr/tmp/named.stats | nameserver statistics data |

**SEE ALSO**

kill(1), gethostbyname(3N), signal(3c), resolver(3), resolver(5), hostname(7), RFC882, RFC883, RFC973, RFC974, *Name Server Operations Guide for BIND*