NAME

resolver − DNS client

SYNOPSIS

/etc/resolv.conf

DESCRIPTION

The **resolver** is the DNS client used on most Linux and BSD systems. It comes with glibc. Its configuration file /etc/resolv.conf (note the spelling) determines the DNS servers to use, and various other options - see below.

Almost all machines have a DNS server set up in this file - if it doesn't exist, the system will assume there's a DNS server running on the local machine, and work out the search path from the machines domain name.

The config file is read the first time the DNS client is invoked by a process.

The different configuration options are:

nameserver IP address of a DNS server to use. Multiple name servers may be listed, each on their own line. The **resolver** will use them in order listed - if the first server times out answering the query, the next server will be tried, and so on. If the resolver runs out out of name servers, the first server will be queried again, until a maximum number of retries are made.

The maximum number of DNS servers to use is set by MAXNS (see <resolv.h>)

search Domain(s) to use for DNS lookups when no domain is specified. List each domain following the search keyword with spaces or tabs between them. Each possible domain will be checked in order until a match is found. Note that this process may be slow (queries will time out if no server is available for a domain) and will generate a lot of network traffic if the servers for the listed domains aren't local.

The search list is currently limited to six domains with a total of 256 characters. If search isn't specified, the search list will be determined from the local domain name (whatever comes after the first dot). If the host name doesn't contain a domain, the root domain is used.

By default, it search contains only the local domain name.

domain Local domain name. You can use this instead of the search option to specify a single domain to check if a hostname isn't specified. Most people just use search instead (that option lets you use multiple servers, domain doesn't). You can't use domain and search at the same time - they're mutually exclusive.

If domain isn't specified, the domain will be determined from the local domain name (whatever comes after the first dot). If the host name doesn't contain a domain, the root domain is used.

sortlist Sorts addresses returned by the gethostbyname system call. 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

```
options option ...
```

where *option* is one of the following:

`debug` sets `RES_DEBUG` in *`_res.options`*.

`ndots:`*`n`* sets a threshold for the number of dots which must appear in a name given to **`res_query`**() (see `resolver`(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> ).`

`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> ).`

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

`ip6-dotint / no-ip6-dotint`
sets / clears the `RES_NOIP6DOTINT` bit in *`_res.options`*, which when set (ip6-dotint) will enable reverse IPv6 lookups to be made in the (deprecated) ip6.int zone; when clear (no-ip6-dotint), reverse IPv6 lookups are made in the ip6.arpa zone by default.

`ip6-bytestring`
sets `RES_USEBSTRING` in *`_res.options`*. This causes reverse IPv6 lookups to be made using the bit-label format of RFC 2673; if not set, then nibble format is used.

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.

FILES

`/etc/resolv.conf` `<resolv.h>`

SEE ALSO

`gethostbyname`(3), `hostname`(7), `named`(8), `resolver`(3), `resolver`(5). "Name Server Operations Guide for **BIND**"