NAME

resolv.conf − resolver configuration file

SYNOPSIS

/etc/resolv.conf

DESCRIPTION

The *resolver* is a set of routines in the C library that provide access to the Internet Domain Name System (DNS). 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 hostname and the domain search path is constructed from the domain name.

The different configuration options are:

nameserver Name server IP address

Internet address (in dot notation) of a name server that the resolver should query. Up to **MAXNS** (currently 3, see <*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 hostname returned by **gethostname**(2); the domain part is taken to be everything after the first '.'. Finally, if the hostname 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. Resolver queries having fewer than *ndots* dots (default is 1) in them will be attempted using each component of the search path in turn until a match is found. For environments with multiple subdomains please read **options ndots:***n* below to avoid man-in-the-middle attacks and unnecessary traffic for the root-dns-servers. 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 This option allows addresses returned by **gethostbyname**(3) 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. Here is an example:

```
sortlist 130.155.160.0/255.255.240.0 130.155.0.0
```

options

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**(3) (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. The maximum value for this option is silently capped to 15.

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** (currently 5, see <*resolv.h*>). The maximum value for this option is silently capped to 30.

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** (currently 2, see <*resolv.h*>). The maximum value for this option is silently capped to 5.

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 hostnames 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**(3) function, and of mapping IPv4 responses in IPv6 "tunneled form" if no AAAA records are found but an A record set exists.

ip6-bytestring (since glibc 2.3.4)
sets **RES_USE_BSTRING** in *_res.options*. This causes reverse IPv6 lookups to be made using the bit-label format described in RFC 2673; if this option is not set, then nibble format is used.

ip6-dotint/**no-ip6-dotint** (since glibc 2.3.4)
Clear/set **RES_NOIP6DOTINT** in *_res.options*. When this option is clear (**ip6-dotint**), reverse IPv6 lookups are made in the (deprecated) *ip6.int* zone; when this option is set (**no-ip6-dotint**), reverse IPv6 lookups are made in the *ip6.arpa* zone by default. This option is set by default.

edns0 (since glibc 2.6)
sets **RES_USE_EDNSO** in *_res.options*. This enables support for the DNS extensions described in RFC 2671.

single-request (since glibc 2.10)
sets **RES_SNGLKUP** in *_res.options*. By default, glibc performs IPv4 and IPv6 lookups in parallel since version 2.9. Some appliance DNS servers cannot handle these queries properly and make the requests time out. This option disables the behavior and makes glibc perform the IPv6 and IPv4 requests sequentially (at the cost of some slowdown of the resolving process).

single-request-reopen (since glibc 2.9)
The resolver uses the same socket for the A and AAAA requests. Some hardware mistakenly only sends back one reply. When that happens the client sytem will sit and wait for the second reply. Turning this option on changes this behavior so that if two requests from the same port are not handled correctly it will close the socket and open a new one before sending the second request.

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), **resolver**(3), **hostname**(7), **named**(8)
Name Server Operations Guide for BIND

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.