NAME

resolver, res_init, res_mkquery, res_send, res_search, dn_comp, dn_expand – resolver routines

SYNOPSIS

cc [*flag* ...] *file* ... **–lresolv –lsocket –lnsl** [*library* ...]

#include <sys/types.h>
#include <netinet/in.h>
#include <arpa/nameser.h>
#include <resolv.h>

int res_init(void);

int res_mkquery(int *op*, **char** **dname*, **int** *class*, **int** *type*, **char** **data*, **int** *datalen*,
  struct rrec **newrr*, **char** **buf*, **int** *buflen***);**

int res_send(char **msg*, **int** *msglen*, **char** **answer*, **int** *anslen***);**

int res_search(char **dname*, **int** *class*, **int** *type*, **uchar** **answer*, **int** *anslen***);**

int dn_comp(char **exp_dn*, **char** **comp_dn*, **int** *length*, **char** ***dnptrs*, **char** ***lastdnptr***);**

int dn_expand(char **msg*, **char** **comp_dn*, **char** *exp_dn*, **int** *msglen*, **int** *length***);**

MT-LEVEL

Unsafe

DESCRIPTION

These routines are used for making, sending and interpreting packets to Internet domain name servers. Global information that is used by the resolver routines is kept in the variable **_res**. Most of the values have reasonable defaults and can be ignored. Options are a simple bit mask and are OR-ed in to enable. Options stored in **_res.options** are defined in **<resolv.h>** and are as follows.

RES_INIT True if the initial name server address and default domain name are initialized (that is, **res_init( )** has been called).

RES_DEBUG Print debugging messages.

RES_AAONLY Accept authoritative answers only. **res_send( )** will continue until it finds an authoritative answer or finds an error. Currently this is not implemented.

RES_USEVC Use TCP connections for queries instead of UDP.

RES_PRIMARY Query primary server only.

RES_IGNTC Unused currently (ignore truncation errors, that is, do not retry with TCP).

RES_RECURSE Set the recursion desired bit in queries. This is the default. **res_send( )** does not do iterative queries and expects the name server to handle recursion.

RES_DEFNAMES Append the default domain name to single label queries. This is the default.

RES_STAYOPEN Used with **RES_USEVC** to keep the TCP connection open between queries. This is useful only in programs that regularly do many queries. UDP should be the normal mode used.

RES_DNSRCH Search up local domain tree.

res_init( ) reads the initialization file to get the default domain name and the Internet address of the initial hosts running the name server. If this line does not exist, the host running the resolver is tried.

res_mkquery( ) makes a standard query message and places it in *buf*. **res_mkquery( )** will return the size of the query or −**1** if the query is larger than *buflen*. *op* is usually **QUERY** but can be any of the query types defined in **<arpa/nameser.h>**. *dname* is the domain name. If *dname* consists of a single label and the **RES_DEFNAMES** flag is enabled (the default), *dname* will be appended with the current domain name. The current domain name is defined in a system file and can be overridden by the environment variable **LOCALDOMAIN**. *newrr* is currently unused but is intended for making update messages. *class* and *type* define the class and type of query. **data* is the resource record; and *datalen* is the length of the record.

res_send( ) sends a query to name servers and returns an answer. It will call **res_init( )** if **RES_INIT** is not set, send the query to the local name server, and handle timeouts and retries. *msg* is the query sent; *msglen* is its length. *answer* is the response returned. The length of the response is stored in *anslen*. **res_send( )** returns the length of the response or −**1** if there were errors.

res_search( ) formulates and sends a normal query (**QUERY**) message, and stores the response in a buffer supplied by the caller. **dname** is the domain name. **class** and **type** define the class and type of query (see **<arpa/nameser.h>**). The response is returned in the user-supplied buffer **answer**. **res_search** returns the length of **answer** in **anslen**. **res_search( )** will call **res_init( )** if the **RES_INIT** flag is not enabled. If **dname** consists of a single label and the **RES_DEFNAMES** flag is enabled (the default), **dname** will be appended with the current domain name. If the **RES_DNSRCH** flag is enabled, **res_search( )** will search up the local domain tree until an answer has been retrieved or an unrecoverable error has been encountered. **res_search( )** returns the length of **answer** on success and −**1** on error. Note that **res_search( )** is only useful for queries in the same name hierarchy as the local host.

dn_expand( ) expands the compressed domain name *comp_dn* to a full domain name. Expanded names are converted to upper case. *msg* is a pointer to the beginning of the message, *exp_dn* is a pointer to a buffer of size *length* for the result. The size of compressed name is returned or −**1** if there was an error.

dn_comp( ) compresses the domain name *exp_dn* and stores it in *comp_dn*. The size of the compressed name is returned or −**1** if there were errors. *length* is the size of the array pointed to by *comp_dn*. *dnptrs* is a list of pointers to previously compressed names in the current message. The first pointer points to to the beginning of the message and the list ends with **NULL**. *lastdnptr* is a pointer to the end of the array pointed to *dnptrs*. A side effect is to update the list of pointers for labels inserted into the message by **dn_comp( )** as the name is compressed. If *dnptrs* is **NULL**, do not try to compress names. If *lastdnptr* is **NULL**, do not update the list.

FILES

/etc/resolv.conf

SEE ALSO

in.named(1M), **nstest**(1M), **resolv.conf**(4)

NOTES

These interfaces are unsafe in multithreaded applications. Unsafe interfaces should be called only from the main thread.