NAME

ares_search − Initiate a DNS query with domain search

SYNOPSIS

```
#include <ares.h>

typedef void (*ares_callback_dnsrec)(void *arg,
                                     ares_status_t status,
                                     size_t timeouts,
                                     const ares_dns_record_t *dnsrec);

void ares_search_dnsrec(ares_channel_t *channel,
                        const ares_dns_record_t *dnsrec,
                        ares_callback_dnsrec callback, void *arg);

typedef void (*ares_callback)(void *arg, int status,
                              int timeouts, unsigned char *abuf,
                              int alen);

void ares_search(ares_channel_t *channel, const char *name,
                 int dnsclass, int type,
                 ares_callback callback, void *arg);
```

DESCRIPTION

The **ares_search** function initiates a series of single-question DNS queries on the name service channel identified by *channel*, using the channel's search domains as well as a host alias file given by the HOSTALIAS environment variable. The parameter*name* gi ves the alias name or the base of the query name as a NUL-terminated C string of period-separated labels; if it ends with a period, the channel's search domains will not be used. Periods and backslashes within a label must be escaped with a backslash. The parameters *dnsclass* and *type* give the class and type of the query using the values defined in **<arpa/nameser.h>**. When the query sequence is complete or has failed, the ares library will invoke *callback*. Completion or failure of the query sequence may happen immediately, or may happen during a later call to **ares_process**(3) or **ares_destroy**(3).

If this is called from a thread other than which the main program event loop is running, care needs to be taken to ensure any file descriptor lists are updated immediately within the eventloop. When the associated callback is called, it is called with a channel lock so care must be taken to ensure any processing is minimal to prevent DNS channel stalls.

The callback argument *arg* is copied from the **ares_search** argument *arg*. The callback argument *status* indicates whether the query sequence ended with a successful query and, if not, how the query sequence failed. It may have any of the following values:

ARES_SUCCESS A query completed successfully.

ARES_ENODATA No query completed successfully; when the query was tried without a search domain appended, a response was returned with no answers.

ARES_EFORMERR A query completed but the server claimed that the query was malformatted.

ARES_ESERVFAIL No query completed successfully; when the query was tried without a search domain appended, the server claimed to have experienced a failure. (This code can only occur if the **ARES_FLAG_NOCHECKRESP** flag was specified at channel initialization time; otherwise, such responses are ignored at the **ares_send**(3) level.)

ARES_ENOTFOUND
No query completed successfully; when the query was tried without a search domain appended, the server reported that the queried-for domain name was not found.

ARES_ENOTIMP A query completed but the server does not implement the operation requested by the query. (This code can only occur if the**ARES_FLA G_NOCHECKRESP** flag was specified at channel initialization time; otherwise, such responses are ignored at the **ares_send**(3) level.)

ARES_EREFUSED A query completed but the server refused the query. (This code can only occur returned if the **ARES_FLAG_NOCHECKRESP** flag was specified at channel initialization time; otherwise, such responses are ignored at the **ares_send**(3) level.)

ARES_TIMEOUT No name servers responded to a query within the timeout period.

ARES_ECONNREFUSED
No name servers could be contacted.

ARES_ENOMEM Memory was exhausted.

ARES_ECANCELLED
The query was cancelled.

ARES_EDESTRUCTION
The name service channel *channel* is being destroyed; the query will not be completed.

ARES_ENOSERVER
No query completed successfully; no DNS servers were configured on the channel.

The callback argument *timeouts* reports how many times a query timed out during the execution of the given request.

If a query completed successfully, the callback argument *abuf* points to a result buffer of length *alen*. If the query did not complete successfully, *abuf* will usually be NULL and *alen* will usually be 0, but in some cases an unsuccessful query result may be placed in *abuf*.

The *ares_search_dnsrec(3)* function behaves identically to *ares_search(3)*, but takes an initialized and filled DNS record object to use for queries as the second argument *dnsrec* instead of a name, class and type. This object is used as the base for the queries and must itself represent a valid query for a single name. Note that the search domains will only be appended to the name in the question section; RRs on the DNS record object will not be affected. Moreover, the *callback* argument is of type *ares_callback_dnsrec*. This callback behaves identically to *ares_callback*, but is invoked with a parsed DNS record object *dnsrec* rather than a raw buffer with length. Note that this object is read-only.

The *ares_search_dnsrec(3)* function returns an *ares_status_t* response code. This may be useful to know that the query was enqueued properly. The response code does not reflect the result of the query, just the result of the enqueuing of the query.

AVAILABILITY

ares_search_dnsrec(3) was introduced in c-ares 1.28.0.

SEE ALSO

ares_process(3), **ares_dns_record**(3)