NAME

gethostbyname, gethostbyaddr, sethostent, endhostent, herror, hstrerror − get network host entry

SYNOPSIS

```
#include <netdb.h>
extern int h_errno;

struct hostent *gethostbyname(const char *name);

#include <sys/socket.h>       /* for AF_INET */
struct hostent *gethostbyaddr(const char *addr,
  int len, int type);

void sethostent(int stayopen);

void endhostent(void);

void herror(const char *s);

const char *hstrerror(int err);


/* GNU extensions */
struct hostent *gethostbyname2(const char *name, int af);

int gethostbyname_r (const char *name,
  struct hostent *ret, char *buf, size_t buflen,
  struct hostent **result, int *h_errnop);

int gethostbyname2_r (const char *name, int af,
  struct hostent *ret, char *buf, size_t buflen,
  struct hostent **result, int *h_errnop);
```

DESCRIPTION

The **gethostbyname()** function returns a structure of type *hostent* for the given host *name*. Here *name* is either a host name, or an IPv4 address in standard dot notation, or an IPv6 address in colon (and possibly dot) notation. (See RFC 1884 for the description of IPv6 addresses.) If *name* is an IPv4 or IPv6 address, no lookup is performed and **gethostbyname**() simply copies *name* into the *h_name* field and its *struct in_addr* equivalent into the *h_addr_list[0]* field of the returned *hostent* structure. If *name* doesn't end in a dot and the environment variable **HOSTALIASES** is set, the alias file pointed to by **HOSTALIASES** will first be searched for *name* (see **hostname**(7) for the file format). The current domain and its parents are searched unless *name* ends in a dot.

The **gethostbyaddr()** function returns a structure of type *hostent* for the given host address *addr* of length *len* and address type *type*. The only valid address type is currently **AF_INET**.

The **sethostent()** function specifies, if *stayopen* is true (1), that a connected TCP socket should be used for the name server queries and that the connection should remain open during successive queries. Otherwise, name server queries will use UDP datagrams.

The **endhostent()** function ends the use of a TCP connection for name server queries.

The (obsolete) **herror()** function prints the error message associated with the current value of *h_errno* on stderr.

The (obsolete) **hstrerror()** function takes an error number (typically *h_errno*) and returns the corresponding message string.

The domain name queries carried out by **gethostbyname()** and **gethostbyaddr()** use a combination of any

or all of the name server **named**(8), a broken out line from */etc/hosts*, and the Network Information Service (NIS or YP), depending upon the contents of the *order* line in */etc/host.conf*. (See **resolv+**(8)). The default action is to query **named**(8), followed by */etc/hosts*.

The *hostent* structure is defined in <*netdb.h*> as follows:

```
struct hostent {
        char    *h_name;        /* official name of host */
        char    **h_aliases;    /* alias list */
        int     h_addrtype;     /* host address type */
        int     h_length;       /* length of address */
        char    **h_addr_list;  /* list of addresses */
}
#define h_addr  h_addr_list[0]  /* for backward compatibility */
```

The members of the *hostent* structure are:

h_name
The official name of the host.

h_aliases
A zero-terminated array of alternative names for the host.

h_addrtype
The type of address; always **AF_INET** at present.

h_length
The length of the address in bytes.

h_addr_list
A zero-terminated array of network addresses for the host in network byte order.

h_addr The first address in *h_addr_list* for backward compatibility.

RETURN VALUE

The **gethostbyname()** and **gethostbyaddr()** functions return the *hostent* structure or a NULL pointer if an error occurs. On error, the *h_errno* variable holds an error number.

ERRORS

The variable *h_errno* can have the following values:

HOST_NOT_FOUND
The specified host is unknown.

NO_ADDRESS or **NO_DATA**
The requested name is valid but does not have an IP address.

NO_RECOVERY
A non-recoverable name server error occurred.

TRY_AGAIN
A temporary error occurred on an authoritative name server. Try again later.

FILES

/etc/host.conf
resolver configuration file

/etc/hosts
host database file

CONFORMING TO

BSD 4.3.

NOTES

The SUS-v2 standard is buggy and declares the *len* parameter of **gethostbyaddr()** to be of type *size_t*. (That is wrong, because it has to be *int*, and *size_t* is not. POSIX 1003.1-2001 makes it *socklen_t*, which is OK.)

The functions **gethostbyname**() and **gethostbyaddr**() may return pointers to static data, which may be overwritten by later calls. Copying the *struct hostent* does not suffice, since it contains pointers - a deep copy is required.

Glibc2 also has a **gethostbyname2()** that works like **gethostbyname()**, but permits to specify the address family to which the address must belong.

Glibc2 also has reentrant versions **gethostbyname_r()** and **gethostbyname2_r()**. These return 0 on success and nonzero on error. The result of the call is now stored in the struct with address *ret*. After the call, **result* will be NULL on error or point to the result on success. Auxiliary data is stored in the buffer *buf* of length *buflen*. (If the buffer is too small, these functions will return **ERANGE**.) No global variable *h_errno* is modified, but the address of a variable in which to store error numbers is passed in *h_errnop*.

POSIX 1003.1-2001 marks **gethostbyaddr()** and **gethostbyname()** legacy, and introduces

struct hostent *getipnodebyaddr (const void *restrict *addr***,**
socklen_t *len***, int** *type***, int *restrict** *error_num***);**

struct hostent *getipnodebyname (const char ***_name_,**
int *type***, int** *flags***, int ***_error_num_**);**

SEE ALSO

resolver(3), **hosts**(5), **hostname**(7), **resolv+**(8), **named**(8)