Computer Networks

Assignment 9

2020-21
Even Semester

Submitted By
Abhishek Kumar
19ucs241
Lab Batch: C4

Objective: Report the significance of the following header files used in the `getaddrinfo()` function:

a) arpa/inet.h

arpa/inet.h has definitions for internet operations used in getaddrinfo(). It makes available the type in_port_t and the type in_addr_t and n_addr structure as defined in the description of <netinet/in.h>. Therefore we can conclude that Inclusion of the <arpa/inet.h> header may also make visible all symbols from <netinet/in.h> and <inttypes.h>.

b) netinet/in.h

The netinet/in.h header file contains definitions for the internet protocol family. This in included in getaddrinfo() to define in_port_t and in_addr_t through typedef. The <netinet/in.h> header defines the in_addr structure that includes at least the in_addr_t, s_addr. Also <netinet/in.h> header defines the sockaddr_in structure that includes at least the following member: sa_family_t, sin_family, in_port_t, sin_port, struct in_addr, sin_addr,unsigned char and sin_zero[8].
The <netinet/in.h> header defines the macros for use as values of the level argument of getsockopt() and setsockopt(). It also defines the macros for use as destination addresses for connect(), sendmsg() and sendto(). Also ntohl(), ntohs(), htonl() and htons() as defined in the description of <arpa/inet.h> are available. Inclusion of the <netinet/in.h> header may also make visible all symbols from <arpa/inet.h>.

c) sys/socket.h

sys/socket.h is used to define socklen_t which is intiger of atleast 32 bytes. It is used to define unsigned intiger type sa_family_t It is used to define sockaddr socket which is used in bind(), connect(), recvfrom() and send(). It defines the following macros: SOCK_DGRAM , SOCK_RAW , SOCK_STREAM , SOCK_SEQPACKET etc.

d) net/if.h

It is used to define if_nameindex structure that includes if_index and if_name members. It defines the macro IF_NAMESIZE.

e) errno.h

errno.h defines number of last error. The <errno.h> header file defines the integer variable errno, which is set by system calls and some library functions in the event of an error to indicate what went wrong. The value in errno is significant only when the return value of the call indicated an error (i.e., -1 from most system calls; -1 or NULL from most library functions); a function that succeeds *is* allowed to change errno. The value of errno is never set to zero by any system call or library function.

f) netdb.h

netdb.h defines network database operations. It makes available in_port_t and in_addr_t as defined in <netinet/in.h>.
It defines hostnet structure which includes members: h_name (name of host), h_aliases (pointer to alternative host names), h_addrtype (address type), etc.
It defines netent structure which includes members: n_name, _aliases, n_addrtype, etc.
It defines protent structure which includes members: p_name , p_aliases, p_proto. It defines macro IPPORT_RESERVED, HOST_NOT_FOUND , NO_DATA, etc.

g) ctype.h

The <ctype.h> header file declares a set of functions to classify (and transform) individual characters. For example, isupper() checks whether a character is uppercase or not. Other funtions defined in this header file are int isalnum(int), int isalpha(int), int isblank(int), int iscntrl(int), int isdigit(int), int isgraph(int), int islower(int), int isprint(int), int ispunct(int), int isspace(int), int isupper(int), int isxdigit(int), etc.

h) stdbool.h

The <stdbool.h> header define the following macros: bool (Expands to _Bool), true (Expands to the integer constant 1), false(Expands to the integer constant 0), __bool_true_false_are_defined (Expands to the integer constant 1)

i) resolv/resolv-internal.h

resolv/resolv-internal.h includes following functions: int __res_context_mkquery, __res_context_search, __res_context_query, __res_context_send.

j) resolv/resolv_context.h

resolv/resolv_context.h is used to define in getaddrinfo() the struct resolv_context objects are allocated on the heap, initialized by __resolv_context_get (and its variants), and destroyed by __resolv_context_put. A nested call to __resolv_context_get (after another call to __resolv_context_get without a matching __resolv_context_put call, on the same thread) returns the original pointer, instead of allocating a new context. This prevents unexpected reloading of the resolver configuration.

k) resolv/res_use_inet6.h

resolv/res_use_inet6.h Includes following functions in getaddrinfo(): Static inline void __resolv_context_enable_inet6(struct resolv_context *ctx, bool enable) and __resolv_context_disable_inet6(struct resolv_context *ctx).