## NAME

netinet/in.h - Internet address family

## SYNOPSIS

```
#include <netinet/in.h>
```

## DESCRIPTION

The *<netinet/in.h>* header shall define the following types:

**in_port_t**
: Equivalent to the type **uint16_t** as described in *<inttypes.h>*.

**in_addr_t**
: Equivalent to the type **uint32_t** as described in *<inttypes.h>*.

The *<netinet/in.h>* header shall define the **sa_family_t** type as described in *<sys/socket.h>*.

The *<netinet/in.h>* header shall define the **uint8_t** and **uint32_t** types as described in *<inttypes.h>*. Inclusion of the *<netinet/in.h>* header may also make visible all symbols from *<inttypes.h>* and *<sys/socket.h>*.

The *<netinet/in.h>* header shall define the **in_addr** structure, which shall include at least the following member:

```
in_addr_t  s_addr
```

The *<netinet/in.h>* header shall define the **sockaddr_in** structure, which shall include at least the following members:

```
sa_family_t     sin_family   AF_INET.
in_port_t       sin_port     Port number.
struct in_addr  sin_addr     IP address.
```

The *sin_port* and *sin_addr* members shall be in network byte order.

The **sockaddr_in** structure is used to store addresses for the Internet address family. Pointers to this type shall be cast by applications to **struct sockaddr *** for use with socket functions.

[IP6] ⌦ The *<netinet/in.h>* header shall define the **in6_addr** structure, which shall include at least the following member:

```
uint8_t s6_addr[16]
```

This array is used to contain a 128-bit IPv6 address, stored in network byte order.

The *<netinet/in.h>* header shall define the **sockaddr_in6** structure, which shall include at least the following members:

| | | |
|---|---|---|
| sa_family_t | sin6_family | AF_INET6. |
| in_port_t | sin6_port | Port number. |
| uint32_t | sin6_flowinfo | IPv6 traffic class and flow information. |
| struct in6_addr | sin6_addr | IPv6 address. |
| uint32_t | sin6_scope_id | Set of interfaces for a scope. |

The *sin6_port* and *sin6_addr* members shall be in network byte order.

Prior to calling a function in this standard which reads values from a **sockaddr_in6** structure (for example, *bind()* or *connect()*), the application shall ensure that all members of the structure, including any additional non-standard members, if any, are initialized. If the **sockaddr_in6** structure has a non-standard member, and that member has a value other than the value that would result from default initialization, the behavior of any function in this standard that reads values from the **sockaddr_in6** structure is implementation-defined. All functions in this standard that return data in a **sockaddr_in6** structure (for example, *getaddrinfo()* or *accept()*) shall initialize the structure in a way that meets the above requirements, and shall ensure that each non-standard member, if any, has a value that produces the same behavior as default initialization would in all functions in this standard which read values from a **sockaddr_in6** structure.

The *sin6_scope_id* field is a 32-bit integer that identifies a set of interfaces as appropriate for the scope of the address carried in the *sin6_addr* field. For a link scope *sin6_addr*, the application shall ensure that *sin6_scope_id* is a link index. For a site scope *sin6_addr*, the application shall ensure that *sin6_scope_id* is a site index. The mapping of *sin6_scope_id* to an interface or set of interfaces is implementation-defined.

The *<netinet/in.h>* header shall declare the following external variable:

```
const struct in6_addr in6addr_any
```

This variable is initialized by the system to contain the wildcard IPv6 address. The *<netinet/in.h>* header also defines the IN6ADDR_ANY_INIT macro. This macro must be constant at compile time and can be used to initialize a variable of type **struct in6_addr** to the IPv6 wildcard address.

The *<netinet/in.h>* header shall declare the following external variable:

```
const struct in6_addr in6addr_loopback
```

This variable is initialized by the system to contain the loopback IPv6 address. The *<netinet/in.h>* header also defines the IN6ADDR_LOOPBACK_INIT macro. This macro must be constant at compile time and can be used to initialize a variable of type **struct in6_addr** to the IPv6 loopback address.

The *<netinet/in.h>* header shall define the **ipv6_mreq** structure, which shall include at least the following members:

| | | |
|---|---|---|
| struct in6_addr | ipv6mr_multiaddr | IPv6 multicast address. |
| unsigned | ipv6mr_interface | Interface index. |

⌫

The *<netinet/in.h>* header shall define the following symbolic constants for use as values of the *level* argument of *getsockopt()* and *setsockopt()*:

IPPROTO_IP
    Internet protocol.
IPPROTO_IPV6
    [IP6] ⌦ Internet Protocol Version 6. ⌫
IPPROTO_ICMP

Control message protocol.

IPPROTO_RAW
[RS] ⌦ Raw IP Packets Protocol. ⌫

IPPROTO_TCP
Transmission control protocol.

IPPROTO_UDP
User datagram protocol.

The *<netinet/in.h>* header shall define the following symbolic constant for use as a local address in the structure passed to *bind()*:

INADDR_ANY
IPv4 wildcard address.

The *<netinet/in.h>* header shall define the following symbolic constant for use as a destination address in the structures passed to *connect()*, *sendmsg()*, and *sendto()*:

INADDR_BROADCAST
IPv4 broadcast address.

The *<netinet/in.h>* header shall define the following symbolic constant, with the value specified, to help applications declare buffers of the proper size to store IPv4 addresses in string form:

INET_ADDRSTRLEN
16. Length of the string form for IP.

The *htonl()*, *htons()*, *ntohl()*, and *ntohs()* functions shall be available as described in *<arpa/inet.h>*. Inclusion of the *<netinet/in.h>* header may also make visible all symbols from *<arpa/inet.h>*.

[IP6] ⌦ The *<netinet/in.h>* header shall define the following symbolic constant, with the value specified, to help applications declare buffers of the proper size to store IPv6 addresses in string form:

INET6_ADDRSTRLEN
46. Length of the string form for IPv6.

⌫

[IP6] ⌦ The *<netinet/in.h>* header shall define the following symbolic constants, with distinct integer values, for use in the *option_name* argument in the *getsockopt()* or *setsockopt()* functions at protocol level IPPROTO_IPV6:

IPV6_JOIN_GROUP
Join a multicast group.

IPV6_LEAVE_GROUP
Quit a multicast group.

IPV6_MULTICAST_HOPS
Multicast hop limit.

IPV6_MULTICAST_IF
Interface to use for outgoing multicast packets.

IPV6_MULTICAST_LOOP
Multicast packets are delivered back to the local application.

IPV6_UNICAST_HOPS
Unicast hop limit.

IPV6_V6ONLY
Restrict AF_INET6 socket to IPv6 communications only.

The *<netinet/in.h>* header shall define the following macros that test for special IPv6 addresses. Each macro is of type **int** and takes a single argument of type **const struct in6_addr ***:

IN6_IS_ADDR_UNSPECIFIED
Unspecified address.

IN6_IS_ADDR_LOOPBACK
Loopback address.

IN6_IS_ADDR_MULTICAST
 Multicast address.
IN6_IS_ADDR_LINKLOCAL
 Unicast link-local address.
IN6_IS_ADDR_SITELOCAL
 Unicast site-local address.
IN6_IS_ADDR_V4MAPPED
 IPv4 mapped address.
IN6_IS_ADDR_V4COMPAT
 IPv4-compatible address.
IN6_IS_ADDR_MC_NODELOCAL
 Multicast node-local address.
IN6_IS_ADDR_MC_LINKLOCAL
 Multicast link-local address.
IN6_IS_ADDR_MC_SITELOCAL
 Multicast site-local address.
IN6_IS_ADDR_MC_ORGLOCAL
 Multicast organization-local address.
IN6_IS_ADDR_MC_GLOBAL
 Multicast global address.

⌫

---

*The following sections are informative.*

## APPLICATION USAGE

Although applications are required to initialize all members (including any non-standard ones) of a **sockaddr_in6** structure, the same is not required for the **sockaddr_in** structure, since historically many applications only initialized the standard members. Despite this, applications are encouraged to initialize **sockaddr_in** structures in a manner similar to the required initialization of **sockaddr_in6** structures.

Although it is common practice to initialize a **sockaddr_in6** structure using:

```
struct sockaddr_in6 sa;
memset(&sa, 0, sizeof sa);
```

this method is not portable according to this standard, because the structure can contain pointer or floating-point members that are not required to have an all-bits-zero representation after default initialization. Portable methods make use of default initialization; for example:

```
struct sockaddr_in6 sa = { 0 };
```

or:

```
static struct sockaddr_in6 sa_init;
struct sockaddr_in6 sa = sa_init;
```

A future version of this standard may require that a pointer object with an all-bits-zero representation is a null pointer, and that **sockaddr_in6** does not have any floating-point members if a floating-point object with an all-bits-zero representation does not have the value 0.0.

## RATIONALE

The INADDR_ANY and INADDR_BROADCAST values are byte-order-neutral and thus their byte order is not specified. Many implementations have additional constants as extensions, such as

INADDR_LOOPBACK, that are not byte-order-neutral. Traditionally, these constants are in host byte order, requiring the use of *htonl()* when using them in a **sockaddr_in** structure.

## FUTURE DIRECTIONS

None.

## SEE ALSO

*Host and Network Byte Orders*, *<arpa/inet.h>*, *<inttypes.h>* , *<sys/socket.h>*

XSH *connect*, *getsockopt*, *htonl*, *sendmsg*, *sendto*, *setsockopt*

## CHANGE HISTORY

First released in Issue 6. Derived from the XNS, Issue 5.2 specification.

The *sin_zero* member was removed from the **sockaddr_in** structure as per The Open Group Base Resolution bwg2001-004.

IEEE Std 1003.1-2001/Cor 1-2002, item XBD/TC1/D6/12 is applied, adding **const** qualifiers to the *in6addr_any* and *in6addr_loopback* external variables.

IEEE Std 1003.1-2001/Cor 2-2004, item XBD/TC2/D6/22 is applied, making it clear which structure members are in network byte order.

## Issue 7

This reference page is clarified with respect to macros and symbolic constants.

POSIX.1-2008, Technical Corrigendum 1, XBD/TC1-2008/0061 [355] is applied.

POSIX.1-2008, Technical Corrigendum 2, XBD/TC2-2008/0065 [934], XBD/TC2-2008/0066 [952], XBD/TC2-2008/0067 [934], and XBD/TC2-2008/0068 [952] are applied.

*End of informative text.*

---

return to top of page

[ Main Index | XBD | XSH | XCU | XRAT ]

<<< Previous Home Next >>>