NAME

inet_aton, **inet_addr**, **inet_network**, **inet_ntoa**, **inet_ntoa_r**, **inet_ntop**, **inet_pton**, **inet_makeaddr**, **inet_lnaof**, **inet_netof** - Internet address manipulation routines

LIBRARY

Standard C Library (libc, -lc)

SYNOPSIS

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int
inet_aton(const char *cp, struct in_addr *pin);

in_addr_t
inet_addr(const char *cp);

in_addr_t
inet_network(const char *cp);

char *
inet_ntoa(struct in_addr in);

char *
inet_ntoa_r(struct in_addr in, char *buf, socklen_t size);

const char *
inet_ntop(int af, const void * restrict src, char * restrict dst, socklen_t size);

int
inet_pton(int af, const char * restrict src, void * restrict dst);

struct in_addr
inet_makeaddr(in_addr_t net, in_addr_t lna);

in_addr_t
inet_lnaof(struct in_addr in);
```

in_addr_t
inet_netof(*struct in_addr in*);

DESCRIPTION

The routines **inet_aton**(), **inet_addr**() and **inet_network**() interpret character strings representing numbers expressed in the Internet standard '.' notation.

The **inet_pton**() function converts a presentation format address (that is, printable form as held in a character string) to network format (usually a *struct in_addr* or some other internal binary representation, in network byte order). It returns 1 if the address was valid for the specified address family, or 0 if the address was not parseable in the specified address family, or -1 if some system error occurred (in which case *errno* will have been set). This function is presently valid for AF_INET and AF_INET6.

The **inet_aton**() routine interprets the specified character string as an Internet address, placing the address into the structure provided. It returns 1 if the string was successfully interpreted, or 0 if the string is invalid. The **inet_addr**() and **inet_network**() functions return numbers suitable for use as Internet addresses and Internet network numbers, respectively.

The function **inet_ntop**() converts an address **src* from network format (usually a *struct in_addr* or some other binary form, in network byte order) to presentation format (suitable for external display purposes). The *size* argument specifies the size, in bytes, of the buffer **dst*. INET_ADDRSTRLEN and INET6_ADDRSTRLEN define the maximum size required to convert an address of the respective type. It returns NULL if a system error occurs (in which case, *errno* will have been set), or it returns a pointer to the destination string. This function is presently valid for AF_INET and AF_INET6.

The routine **inet_ntoa**() takes an Internet address and returns an ASCII string representing the address in '.' notation. The routine **inet_ntoa_r**() is the reentrant version of **inet_ntoa**(). The deprecated routine **inet_makeaddr**() takes an Internet network number and a local host address on that network, and constructs an Internet address from it. It should only be assumed to work for historical class A/B/C networks. The deprecated routines **inet_netof**() and **inet_lnaof**() break apart Internet host addresses, returning the network number and local host address part, respectively, assuming the historical class A/B/C network masks.

All Internet addresses are returned in network order (bytes ordered from left to right). All network numbers and local address parts are returned as machine byte order integer values.

INTERNET ADDRESSES (IP VERSION 4)

The **inet_aton**() and **inet_addr**() functions accept IPv4 values specified using the '.' notation in one of the following forms:

a.b.c.d
a.b.c
a.b
a

When four parts are specified, each is interpreted as a byte of data and assigned, from left to right, to the four bytes of an Internet address.

When a three part address is specified, the last part is interpreted as a 16-bit quantity and placed in the least significant two bytes of the network address.

When a two part address is supplied, the last part is interpreted as a 24-bit quantity and placed in the least significant three bytes of the network address.

When only one part is given, the value is stored directly in the network address without any byte rearrangement.

All numbers supplied as "parts" in a ‘.’ notation may be decimal, octal, or hexadecimal, as specified in the C language (i.e., a leading 0x or 0X implies hexadecimal; otherwise, a leading 0 implies octal; otherwise, the number is interpreted as decimal).

Note that **inet_pton**() does not accept 1-, 2-, or 3-part dotted addresses; all four parts must be specified and are interpreted only as decimal values. This is a narrower input set than that accepted by **inet_aton**().

DIAGNOSTICS

The constant INADDR_NONE is returned by **inet_addr**() and **inet_network**() for malformed requests.

ERRORS

The **inet_ntop**() call fails if:

[ENOSPC] *size* was not large enough to store the presentation form of the address.

[EAFNOSUPPORT] **src* was not an AF_INET or AF_INET6 family address.

SEE ALSO

byteorder(3), getaddrinfo(3), gethostbyname(3), getnameinfo(3), getnetent(3), inet_net(3), hosts(5), networks(5)

IP Version 6 Addressing Architecture, RFC, 2373, July 1998.

STANDARDS

The **inet_ntop**() and **inet_pton**() functions conform to X/Open Networking Services Issue 5.2 ("XNS5.2").

HISTORY

These functions appeared in 4.2BSD.

BUGS

The value INADDR_NONE (0xffffffff) is a valid broadcast address, but **inet_addr**() cannot return that value without indicating failure. The newer **inet_aton**() function does not share this problem. The problem of host byte ordering versus network byte ordering is confusing. The string returned by **inet_ntoa**() resides in a static memory area.

The **inet_addr**() function should return a *struct in_addr*.