

Re: question about change in inet_ntoa.c

Source: <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/net/2008-02/msg00303.html>

- *From:* Bruce Evans <brde@xxxxxxxxxxxxxxxx>
 - *Date:* Wed, 27 Feb 2008 03:05:41 +1100 (EST)
-

On Tue, 26 Feb 2008, ithilgore wrote:

Giorgos Keramidas wrote:

On 2008-02-23 02:37, ithilgore <ithilgore.fbsd@xxxxxxxx> wrote:

ithilgore wrote:

I was looking at the differences between some old FreeBSD code and the one of 7.0-RC1 and was wondering about a change in inet_ntoa.c

```
/* 7.0-RC1 */
```

```
sprintf(buf, "%d.%d.%d.%d",  
ucp[0] & 0xff,  
ucp[1] & 0xff,  
ucp[2] & 0xff,  
ucp[3] & 0xff);
```

This version in libkern is best (except `buf' is static, so it is not reentrant and thus quite broken). ucp[N] is unsigned char, but masking with 0xff is needed to support the (unsupported) machines with more than 8 bits in their chars. On normal machines with 8-bit chars, the compiler will optimize away the masks so their only cost is increased portability of the source code. POSIX now requires 8-bit chars, but didn't when the above was written. %d is good enough after masking, since the result is nonnegative and <= INT_MAX. %u would be a style bug except on exotic machines with sizeof(char) == sizeof(int), since the default promotions normally turn all the numeric printf args into ints (not u_ints) thanks to C90's broken "value-preserving" promotion rules.

Re: question about change in inet_ntoa.c

```
/****** 4.11-RELEASE *****/
```

```
static const char fmt[] = "%u.%u.%u%u";  
if ((size_t)snprintf(dst, size, fmt, src[0],  
src[1], src[2], src[3])  
>= size) {
```

This is actually in somewhere/inet_ntop.c:inet_ntop4().

Bugs in this version include:

- benign type mismatch between %u and the promoted args except on exotic machines
- broken on exotic machines. UCHAR_MAX can be arbitrarily large. Then large values will be put in the string. sprintf() could then overrun the buffer, but using snprintf() avoids that.
- bogus cast of snprintf's return value.

-current has a different version in libc, without the bogus cast. Many files in libc/net including inet_ntop.c moved to libc/inet, and some nearby FreeBSD cleanups were lost (ISC cleaned things up differently and not so well in one case that I looked at).

....
....

Was there a specific purpose of changing the more easy and simple way of %u instead of the combination of %d and and-ing with 0xff ??
It essentially gives the same result but increases overhead (i think) in the more recent version.

I think the libkern version was written later, and it is better because its author knows what is portable. It's also much simpler. After and-ing with 0xff, we know the range of the values and don't have to understand UCHAR_MAX to know that our code is only broken on unsupported/exotic machines.

On the other hand, in version 4.11 RELEASE in /usr/src/lib/libc/net/inet_ntoa.c & inet_ntop.c (actually it is inet_ntop.c code but with the same functionality)

But hard to find if you are looking for inet_ntoa.c.

There is also libstand/inet_ntoa.c. It has different bugs and style bugs.

Re: question about change in inet_ntoa.c

Re: question about change in inet_ntoa.c

At least in old versions, it combines the worst features of libkern and libc (static buffer; uses sprintf() and thus can overrun on exotic machines; otherwise mainly style bugs like libc).

which is
called by inet_ntoa there is :

```
static const char *
inet_ntop4(src, dst, size)
const u_char *src;
char *dst;
size_t size;
{
static const char fmt[] = "%u.%u.%u.%u";

if ((size_t)snprintf(dst, size, fmt, src[0], src[1], src[2], src[3])
    >= size) {
```

Another dubious cleanup in the -current version is to first snprintf() to a static buffer of length 16 and copy to the caller's buffer from there. snprintf() makes that unnecessary unless the API requires not touching the caller's buffer on error.

```
errno = ENOSPC;
return (NULL);
}
return (dst);
}
```

Bruce

freebsd-net@xxxxxxxxxxxxx mailing list
<http://lists.freebsd.org/mailman/listinfo/freebsd-net>
To unsubscribe, send any mail to "freebsd-net-unsubscribe@xxxxxxxxxxxxx"