

Re: Why is it dangerous?

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2008-08/msg01234.html

- *From:* Richard Heathfield <rjh@xxxxxxxxxxxxxxxx>
 - *Date:* Sun, 10 Aug 2008 00:59:49 +0000
-

Julian said:

'evening.

I'm not new to C and have been programming in it since I was 8 but here's a strange problem I've never seen before.

When I compile a program from our C course with a windows compiler there is no problem but when I try to compile it with a linux compiler it complains that

a_03.c:(.text+0x4d): warning: the `gets' function is dangerous and should not be used.

Is linux more dangerous than windows?

No. Your Linux compiler warned you about a dangerous function that should never be used. Your Windows compiler clearly forgot to do this. So it could be argued that Windows is more dangerous than Linux.

Where can I download a non dangerous gets function?

Nowhere. The functionality of gets() is defined by ISO; it takes a pointer to the first character in a buffer, and stores an entire line from stdin into that buffer, *regardless of the buffer's size*!! There is no safe way to use such a function.

Instead, you can use fgets(), another standard ISO C function, which lets you specify the size of the buffer, and which will not attempt to store more in the buffer than you say will fit. So if you get the size right, fgets() is not dangerous.

I have never used gets before is

Re: Why is it dangerous?

there undefined behavior somewhere?

It depends on how well-behaved your user is (will they restrain themselves and only type as many characters as you've provided for in your buffer?), but it's safest to assume that you should never, ever, ever use gets().

Here is a trimmed down example program from my assignment that demonstrates the problem

```
#include <stdio.h>
#include <malloc.h>
```

C has no header by that name (although some implementations do). For the prototypes for malloc and free, #include <stdlib.h> instead.

```
void main()
```

```
int main(void)
```

```
{
    char *string;
    printf("enter string (max 2000 chars): ");
    fflush(stdin);
```

The behaviour of fflush is defined