Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'strncpy.3' command

$ man strncpy.3

STRCPY(3) Linux Programmer's Manual STRCPY(3)

NAME

strcpy, strncpy - copy a string

SYNOPSIS

```
#include <string.h>

char *strcpy(char *dest, const char *src);

char *strncpy(char *dest, const char *src, size_t n);
```

DESCRIPTION

The strcpy() function copies the string pointed to by src, including the terminating null byte ('\0'), to the buffer pointed to by dest. The strings may not overlap, and the destination string dest must be large enough to receive the copy. Beware of buffer overruns! (See BUGS.)

The strncpy() function is similar, except that at most n bytes of src are copied. Warning: If there is no null byte among the first n bytes of src, the string placed in dest will not be null-terminated.

If the length of src is less than n, strncpy() writes additional null bytes to dest to ensure that a total of n bytes are written.

A simple implementation of strncpy() might be:

```
char *
strncpy(char *dest, const char *src, size_t n)
{
    size_t i;
```

```
        for (i = 0; i < n && src[i] != '\0'; i++)
            dest[i] = src[i];
        for ( ; i < n; i++)
            dest[i] = '\0';
        return dest;
    }
```

RETURN VALUE

The strcpy() and strncpy() functions return a pointer to the destina? tion string dest.

ATTRIBUTES

For an explanation of the terms used in this section, see at? tributes(7).

Interface	Attribute	Value
strcpy(), strncpy()	Thread safety	MT-Safe

CONFORMING TO

POSIX.1-2001, POSIX.1-2008, C89, C99, SVr4, 4.3BSD.

NOTES

Some programmers consider strncpy() to be inefficient and error prone. If the programmer knows (i.e., includes code to test!) that the size of dest is greater than the length of src, then strcpy() can be used.

One valid (and intended) use of strncpy() is to copy a C string to a fixed-length buffer while ensuring both that the buffer is not over? flowed and that unused bytes in the destination buffer are zeroed out (perhaps to prevent information leaks if the buffer is to be written to media or transmitted to another process via an interprocess communica? tion technique).

If there is no terminating null byte in the first n bytes of src, strncpy() produces an unterminated string in dest. If buf has length buflen, you can force termination using something like the following:

```
    if (buflen > 0) {
```

```
        strncpy(buf, str, buflen - 1);
        buf[buflen - 1]= '\0';
    }
```

(Of course, the above technique ignores the fact that, if src contains more than buflen - 1 bytes, information is lost in the copying to dest.)

strlcpy()

Some systems (the BSDs, Solaris, and others) provide the following function:

```
    size_t strlcpy(char *dest, const char *src, size_t size);
```

This function is similar to strncpy(), but it copies at most size-1 bytes to dest, always adds a terminating null byte, and does not pad the destination with (further) null bytes. This function fixes some of the problems of strcpy() and strncpy(), but the caller must still han? dle the possibility of data loss if size is too small. The return value of the function is the length of src, which allows truncation to be easily detected: if the return value is greater than or equal to size, truncation occurred. If loss of data matters, the caller must either check the arguments before the call, or test the function return value. strlcpy() is not present in glibc and is not standardized by POSIX, but is available on Linux via the libbsd library.

BUGS

If the destination string of a strcpy() is not large enough, then any? thing might happen. Overflowing fixed-length string buffers is a fa? vorite cracker technique for taking complete control of the machine. Any time a program reads or copies data into a buffer, the program first needs to check that there's enough space. This may be unneces? sary if you can show that overflow is impossible, but be careful: pro? grams can get changed over time, in ways that may make the impossible possible.

SEE ALSO

bcopy(3), memccpy(3), memcpy(3), memmove(3), stpcpy(3), stpncpy(3), strdup(3), string(3), wcscpy(3), wcsncpy(3)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.