C-string and C-time functions banned by git

Adam Graliński

C++ FFFE, March 2021

Banned!

https://github.com/git/git/blame/master/banned.h bans

- `strcpy`
- `strcat`
- `sprintf` and `vsprintf`
- and a bunch of time functions:
  - `gmtime`
  - `localtime`
  - `ctime` and `ctime_r`
  - `asctime` and `asctime_r`

. . . but why?

strcpy (and later **strncpy**)

- `strcpy (char* target, const char* source)`

Solution: use **snprintf** instead.

`strcpy` (and later `strncpy`)

- `strcpy (char* target, const char* source)`
- **may overflow target**

Solution: use **`snprintf`** instead.

`strcpy` (and later `strncpy`)

- `strcpy (char* target, const char* source)`
- **may overflow target**
- `strncpy (char* target, const char* source, size_t num)`

Solution: use **`snprintf`** instead.

`strcpy` (and later `strncpy`)

- `strcpy (char* target, const char* source)`
- **may overflow target**
- `strncpy (char* target, const char* source, size_t num)`
- **does not append NUL automatically**
- **clang** warns against using both of these.

Solution: use **`snprintf`** instead.

strcat (and later **strncat**)

- `strcat (char* target, const char* source)`

Solution: use **snprintf** instead.

strcat (and later **strncat**)

- `strcat (char* target, const char* source)`
- **may overflow target**

Solution: use **snprintf** instead.

strcat (and later **strncat**)

- `strcat (char* target, const char* source)`
- **may overflow target**
- `strncat (char* target, const char* source, size_t num)`

Solution: use **snprintf** instead.

strcat (and later **strncat**)

- `strcat (char* target, const char* source)`
- **may overflow target**
- `strncat (char* target, const char* source, size_t num)`
- **fixed: appends NUL automatically, even for truncated entries**

> strncat() has the same quadratic behavior as strcat() and is difficult-to-read and bug-prone. While it hasn't yet been a problem in git iself, strncat() found it's way into 'master' of cgit and caused segfaults on my system.
>
> Signed-off-by: Eric Wong <e@80x24.org>
> Signed-off-by: Junio C Hamano <gitster@pobox.com>

Solution: use **snprintf** instead.

sprintf (and later **vsprintf**)

- `sprintf (char* target, const char* format, ...)`
- `vsprintf (char* target, const char* format, va_list arg)`

Solution: use **snprintf** instead.

sprintf (and later **vsprintf**)

- `sprintf (char* target, const char* format, ...)`
- `vsprintf (char* target, const char* format, va_list arg)`
- **may overflow target**

Solution: use **snprintf** instead.

sprintf (and later **vsprintf**)

- `sprintf (char* target, const char* format, ...)`
- `vsprintf (char* target, const char* format, va_list arg)`
- **may overflow target**
- **Scott Meyers, ages ago**

Solution: use **snprintf** instead.

C-Time

What's wrong with:

```
gmtime
localtime
ctime
asctime
```

What's wrong with:

`gmtime`		`gmtime_r`
`localtime`	$\rightarrow$	`localtime_r`
`ctime`		`ctime_r`
`asctime`		`asctime_r`

Return pointers to static storage (not thread-safe)

What's wrong with:

`gmtime`		`gmtime_r`
`localtime`	$\rightarrow$	`localtime_r`
`ctime`		`ctime_r`
`asctime`		`asctime_r`

Return pointers to static storage (not thread-safe)

. . . when suddenly! $\rightarrow$

C-Time, again

banned.h: mark ctime_r() and asctime_r() as banned

The ctime_r() and asctime_r() functions are reentrant, but have no check that the buffer we pass in is long enough (the manpage says it "should have room for at least 26 bytes"). Since this is such an easy-to-get-wrong interface, and since we have the much safer strftime() as well as its more convenient strbuf_addftime() wrapper, let's ban both of those.

Signed-off-by: Jeff King <peff@peff.net>
Reviewed-by: Taylor Blau <me@ttaylorr.com>
Signed-off-by: Junio C Hamano <gitster@pobox.com>

...just use **`strftime`**.

Summary

So. . .

Summary

So...

- C-strings are **hard to use** and **error-prone**

So...

- C-strings are **hard to use** and **error-prone**
- C++'s **`std::string`**s are a lot easier to use, concatenate, modify*

Summary

So...

- C-strings are **hard to use** and **error-prone**
- C++'s **`std::string`**s are a lot easier to use, concatenate, modify*
- **`snprintf`**: not a silver bullet, but close to

Summary

So...

- C-strings are **hard to use** and **error-prone**
- C++'s **`std::string`**s are a lot easier to use, concatenate, modify*
- **`snprintf`**: not a silver bullet, but close to
- C-time is **easy to do wrong** too.

Summary

So...

- C-strings are **hard to use** and **error-prone**
- C++'s **`std::string`**s are a lot easier to use, concatenate, modify*
- **`snprintf`**: not a silver bullet, but close to
- C-time is **easy to do wrong** too.

So...

- C-strings are **hard to use** and **error-prone**
- C++'s **`std::string`**s are a lot easier to use, concatenate, modify*
- **`snprintf`**: not a silver bullet, but close to
- C-time is **easy to do wrong** too.

Thank you!