Making gets() and its friends more like SIGPIPE and SIGILL

Dr. Paul Vixie, CEO
Farsight Security, Inc.
September 2017 – vBSDCon

Abstract

In the decades since the Morris worm, BSD has substantially modernized, which has included tracking ANSI C and POSIX LibC. Portability was seen as necessary for relevance and success, and for the most part, it has been. There are some exceptions, and it's long past the time when we should have made some hard but sensible choices about what to include or emulate, and what to leave out, and what to poison outright.

NEW YORK POST

ger
or

Credit card fraud spikes after Equifax cyber-attack

BUSINESS EXCLUSIVE

Equifax blames giant breach on vendor software flaw

By Kevin Dugan

September 8, 2017 | 2:31pm | Updated

MORE ON:
EQUIFAX

- Credit card fraud spikes after Equifax cyber-attack
- New Yorkers file suit against Equifax over huge data breach
- Sen. Warren slams Equifax for tricky move
- Senator calls for probe of Equifax hack

Equifax on Friday blamed a flaw in the software running its online databases for allowing hackers to steal the personal information of as many as 143 million Americans, The Post has learned.

Hackers were able to access the info — including Social Security numbers — because there was a flaw in the open-source software created by the Apache Foundation, the company told Jeffrey Meuler, an analyst at RW Baird & Co.

"My understanding is the breach was perpetuated via the Apache STRUTS flaw," Meuler told The Post.

STRUTS is a widely available software system that's used by about 65 percent of Fortune 100 companies, including Lockheed Martin, Citigroup, Vodafone, Virgin Atlantic, Reader's Digest, Office Depot, and Showtime — plus the IRS, according to Igtm, a software development group.

November 2, 1988

- Starting contitions:
  - Inetd on by default, everywhere
  - Fingerd enabled by default, everywhere
  - Firewalls hadn’t been invented yet
  - Fingerd used `gets()`
- New element:
  - Robert T. Morris creates a demo worm
  - The code has a non-demo bug in it
  - All heck breaks loose
  - Note: last known use of DNS HINFO RR

The Birth of Return Oriented Programming

```
char *
gets(char *s) {
   char *t = s;
   int ch;

   while ((ch = getchar()) != ‘\n’ && ch != EOF)
      *t++ = ch;
   *t = ‘\0’;
   return (s);
}
```

FreeBSD's Response to the `gets()` Problem

```
[fb10.local:amd64] cc tgets.c
/tmp/tgets-30878f.o: In function `main':
tgets.c:(.text+0xd): warning: warning: this program uses
gets(), which is unsafe.

[fb10.local:amd64] ./a.out
warning: this program uses gets(), which is unsafe.
foo bar baz
foo bar baz
Hello, world. ovf is 'ar baz'
```

Linux's Response to the `gets()` Problem

```
BUGS
       Never use gets().  Because it is impossible to tell
       without knowing the data in advance how many
       characters gets() will read, and because gets() will
       continue to store characters past the end of the
       buffer, it is extremely dangerous to use.  It has
       been used to break computer security.

       Use fgets() instead.
```

First, They Came for the Printers

"As long as people write parsers and connect them to the Internet, I'll have work." --anon

It's Hardly Just `gets`()

- Consider `strcpy`(), `sprintf`(), `strcat`(), etc.
  - Noting, these can often be hand-verified.
  - But not, sadly, machine-verified.
- And then `strncpy`(), `snprintf`(), `strncat`(), etc.
  - Note especially that truncation may occur.
  - And that \0 termination may not occur.
- What about `strlcpy`() and `strlcat`()?
  - Adds bounds checking, vs. `strcpy`() and `strcat`()
  - Ensures \0 termination, vs. `strncpy`() and `strncat`()
  - But… truncation may still be an undefined result

Time Out for Finger Pointing

- A design should be as simple as possible, but not simpler.
  - C, without bounds checking, is too simple.
  - “Better than PDP-11 assembly language” is a low bar.
- LibC was crafted in light of the Software Tools work.
  - However, fortran (ratfor) and Pascal *had* bounds checking.
- ANSI C had to preserve as much existing code as possible.
  - So, incompatibilities were limited, and nothing was removed until C11.
  - But, `gets`() was in C89, and is still in C++11.
  - And, POSIX C is a superset of ANSI C

Designing By Contract

- A function caller ought to have “reasonable expectations” of the function’s results and side effects.
  - Such as, the program state is still defined.
  - After `gets`(), that expectation is *not* reasonable.
- `s{n,}printf(), str{l,n,}{cat,cpy}`, etc, *can be* hand-verified.
  - E.g., `assert(snprintf(buf, len, …) < len);`
  - We probably need #pragmas to assist in compiler verification.
- But `gets`(), by referring to a future input stream, *cannot be*.
  - Once you call `gets`(), your program state is no longer defined.
  - Why – oh why? – would you want to continue executing?

Correct implementation of `gets`() after 1988

```
char *
gets(char *s) {
   abort();
}
```

SIGPIPE and SIGILL

- SIGPIPE happens when you `write`() on an orphaned pipe
  - It's catchable, but very few programs do this
  - It's nec'y, since few programs check the `write`() return value
  - It's a design response to what would be undefined program state
- SIGILL happens when the Program Counter points at non-code
  - Also catchable, but hardly ever done outside debuggers or interpreters
  - It's a clear sign that the program state has become undefined

BTW, WTF?

- Implementing pipe() using socketpair()?
  - So if I read and write in the wrong direction, it just works?
- Mapping a page full of \0 at address 0?
  - So if I indirect through a NULL pointer, I get an empty C string?
- Demonstrated ignorance must result in a software exception
  - We must make more forms of ignorance self-demonstrating
- The Robustness Principle is precisely, exactly wrong
  - Especially for software and networks
    - (as it was for 10-megabit thickwire ethernet)

Conclusion

- IoT will bring millions of new companies into the Internet ecosystem
  - And they will bring millions of fresh, undamaged programmers with them
- We have to break more things earlier
  - “Confidence level: boots in lab” was the kind of joke that’s only funny once
- We have to stop using a glorified macro assembler for applications
  - Moore’s law has been kind to us – can we allocate resources to safety?
  - In Golang, strings aren’t writable, and everything has boundaries
  - See also Java, Javascript, Perl, PHP, Rust, Haskell, and your favorite
- This is the world we made – these problems are on our account