moz://a

Attacking Rust for Fun and Profit

Also an introduction to Rust and gdb

Diane Hosfelt | @avadacatavra

October 18, 2017


MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A
0x6339392C,
0x7363682E.
I HATE YOU.

Setting up rust-gdb

- Requires codesigning on Mac
- ‘rust-gdb‘ comes with cargo
- ‘gdb target/debug/deps/<crate>-xxx‘

rust-gdb without symbols (boo)

```
$ gdb target/debug/examples/buffer_overflow
warning: '/Users/ddh/mozilla/rust-exploits/target/debug/
    examples/buffer_overflow-bfdb27d82546b886.0.o':
    can't open to read symbols: No such file or directory.
(gdb) b buffer_overflow::main
Breakpoint 1 at 0x1000046b4
(gdb) r
Thread 2 hit Breakpoint 1, 0x00000001000046b4
    in buffer_overflow::main ()
(gdb) info locals
No symbol table info available.
(gdb) b buffer_overflow.rs:47
No source file named buffer_overflow.rs.
Make breakpoint pending on future shared library load?
    (y or [n]) n
```

rust-gdb with symbols (yay!)

```
$ gdb target/debug/examples/buffer_overflow-bfdb27d82546b886
(gdb) b buffer_overflow::main
Breakpoint 1 at 0x1000048e0: file
    examples/buffer_overflow.rs, line 80.
(gdb) r
Thread 2 hit Breakpoint 1, buffer_overflow::main () at
    examples/buffer_overflow.rs:80
    80          let mut f = Box::new(Foo::new("corgi".to_string()
(gdb) info locals
f = 0x100620000
(gdb)  b buffer_overflow.rs:47
Breakpoint 2 at 0x100004858: file
    examples/buffer_overflow.rs, line 47.
```


OKAY, HUMAN.
HUH?
BEFORE YOU HIT 'COMPILE,' LISTEN UP.
YOU KNOW WHEN YOU'RE FALLING ASLEEP, AND YOU IMAGINE YOURSELF WALKING OR SOMETHING,
AND SUDDENLY YOU MISSTEP, STUMBLE, AND JOLT AWAKE?
YEAH!
WELL, THAT'S WHAT A SEGFAULT FEELS LIKE.
DOUBLE-CHECK YOUR DAMN POINTERS, OKAY?

Pointers

Example

Printing out a pointer in Rust

```
println!("{:p}", &a)
```

Example

Printing out a pointer in gdb

```
(gdb) p self
$1 = (buffer_overflow::Foo *) 0x101820000
(gdb) p self.bar
$2 = alloc::vec::Vec<u8>
   {buf: alloc::raw_vec::RawVec<u8, alloc::heap::Heap>
   {ptr: core::ptr::Unique<u8> {pointer: core::nonzero::NonZero<
   (0x101816010 "corgi\000"), _marker: core::marker::PhantomData
    cap: 5, a: alloc::heap::Heap}, len: 5}
(gdb) p self.baz
$3 = (void (*)(void)) 0x100004880 <buffer_overflow::bark>
(gdb) p &self.bar
$4 = (alloc::vec::Vec<u8> *) 0x101820000
```

Where are things allocated?

Vec	heap
String	heap
str	heap
static str	static data section
slice	heap
usize	stack
u8 array	can be placed on stack

Buffer overflow in C

```c
int main ( int argc, char **argv ) {
  char buf[10];
  strcpy(buf, "hello, world\0");
  return 0;
}
```

Buffer overflow in Rust

```rust
fn this_is_ugly(input: &str) -> [u8;100] {
 let mut buf = [0u8; 100];
 // let mut buf = [0u8; 10];
 &mut buf[..13].copy_from_slice(input.as_bytes());
 buf
}
```

Buffer overflow in Rust

```
Thread 2 hit Breakpoint 1, buffer_overflow::this_is_ugly (input=...) at examples/buffer_overflow.rs:38
38      fn this_is_ugly(input: &str) {//-> [u8;10] {
(gdb) n
39          let mut buf = [0u8; 100];
(gdb)
41              &mut buf[..13].copy_from_slice(input.as_bytes());
(gdb) p buf
$3 = [0 <repeats 100 times>]
(gdb) p &buf
$4 = (u8 (*)[100]) 0x7fff5fbff79c b'\000' <repeats 100 times>
(gdb) info frame
Stack level 0, frame at 0x7fff5fbff840:
 rip = 0x1000046d0 in buffer_overflow::this_is_ugly (examples/buffer_overflow.rs:41); saved rip = 0x10000480e
 called by frame at 0x7fff5fbff940
 source language rust.
 Arglist at 0x7fff5fbff830, args: input=...
 Locals at 0x7fff5fbff830, Previous frame's sp is 0x7fff5fbff840
 Saved registers:
  rbp at 0x7fff5fbff830, rip at 0x7fff5fbff838
```

Buffer overflow in Rust

```
(gdb) x/4w 0x7fff5fbff838
0x7fff5fbff838: 0x0000480e      0x00000001      0x0013b000      0x00000001
(gdb) x/100w 0x7fff5fbff79c
0x7fff5fbff79c: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7ac: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7bc: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7cc: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7dc: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7ec: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7fc: 0x00000000      0x0060e0c8      0x00000001      0x00000002
0x7fff5fbff80c: 0x00000000      0x5fbff878      0x00000000      0x0060e000
0x7fff5fbff81c: 0x00000001      0x0010a000      0x00000001      0x00001000
0x7fff5fbff82c: 0xff800000      0x5fbff930      0x00007fff      0x0000480e
0x7fff5fbff83c: 0x00000001      0x0013b000      0x00000001      0x00000000
0x7fff5fbff84c: 0x00000000      0xc9aab000      0x00007fff      0x00000018
0x7fff5fbff85c: 0x00000000      0x5fbff890      0x00007fff      0xc0d1a282
0x7fff5fbff86c: 0x00007fff      0x00000018      0x00000000      0x002020c0
0x7fff5fbff87c: 0x00000001      0x00000000      0x00000000      0x00007450
0x7fff5fbff88c: 0x00000001      0x5fbff8b0      0x00007fff      0xc0d19200
0x7fff5fbff89c: 0x00007fff      0x00000000      0x00000000      0x002020c0
0x7fff5fbff8ac: 0x00000001      0x5fbff8f0      0x00007fff      0xc0b967ab
0x7fff5fbff8bc: 0x00007fff      0x002020c0      0x00000001      0x002020c0
0x7fff5fbff8cc: 0x00000001      0x0013b000      0x00000001      0x00000000
0x7fff5fbff8dc: 0x00000000      0x00001000      0x00000000      0x5f401000
0x7fff5fbff8ec: 0x00007fff      0x5fbff990      0x00007fff      0x0000aa4b
0x7fff5fbff8fc: 0x00000001      0x002020d0      0x00000001      0x002020c8
0x7fff5fbff90c: 0x00000001      0x00000001      0x00000000      0x5f401000
0x7fff5fbff91c: 0x00007fff      0x0061b000      0x00000001      0x00000001
(gdb) n
```

Buffer overflow in Rust

```
Thread 2 hit Breakpoint 2, buffer_overflow::this_is_ugly (input=...) at examples/buffer_overflow.rs:43
43      }
(gdb) x/100w 0x7fff5fbff79c
0x7fff5fbff79c: 0x6c6c6568      0x77202c6f      0x646c726f      0x00000000
0x7fff5fbff7ac: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7bc: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7cc: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7dc: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7ec: 0x00000000      0x00000000      0x00000000      0x00000000
0x7fff5fbff7fc: 0x00000000      0x0060e0c8      0x00000001      0x0000000d
0x7fff5fbff80c: 0x00000000      0x00039981      0x00000001      0x0000000d
0x7fff5fbff81c: 0x00000000      0x00039981      0x00000001      0x0000000d
0x7fff5fbff82c: 0x00000000      0x5fbff930      0x00007fff      0x0000480e
0x7fff5fbff83c: 0x00000001      0x0013b000      0x00000001      0x00000000
0x7fff5fbff84c: 0x00000000      0xc9aab000      0x00007fff      0x00000018
0x7fff5fbff85c: 0x00000000      0x5fbff890      0x00007fff      0xc0d1a282
0x7fff5fbff86c: 0x00007fff      0x00000018      0x00000000      0x002020c0
0x7fff5fbff87c: 0x00000001      0x00000000      0x00000000      0x00007450
0x7fff5fbff88c: 0x00000001      0x5fbff8b0      0x00007fff      0xc0d19200
0x7fff5fbff89c: 0x00007fff      0x00000000      0x00000000      0x002020c0
0x7fff5fbff8ac: 0x00000001      0x5fbff8f0      0x00007fff      0xc0b967ab
0x7fff5fbff8bc: 0x00007fff      0x002020c0      0x00000001      0x002020c0
0x7fff5fbff8cc: 0x00000001      0x0013b000      0x00000001      0x00000000
0x7fff5fbff8dc: 0x00000000      0x00001000      0x00000000      0x5f401000
0x7fff5fbff8ec: 0x00007fff      0x5fbff990      0x00007fff      0x0000aa4b
0x7fff5fbff8fc: 0x00000001      0x002020d0      0x00000001      0x002020c8
0x7fff5fbff90c: 0x00000001      0x00000001      0x00000000      0x5f401000
0x7fff5fbff91c: 0x00007fff      0x0061b000      0x00000001      0x00000001
```

Buffer overflow in Rust

```rust
fn this_is_ugly(input: &str) -> [u8;10] {
 // let mut buf = [0u8; 100];
 let mut buf = [0u8; 10];
 &mut buf[..13].copy_from_slice(input.as_bytes());
 buf
}
```

thread 'main' panicked at 'index 13 out of range for slice of length 10', src/libcore/slice/mod.rs:748:4

Buffer overflow in Rust

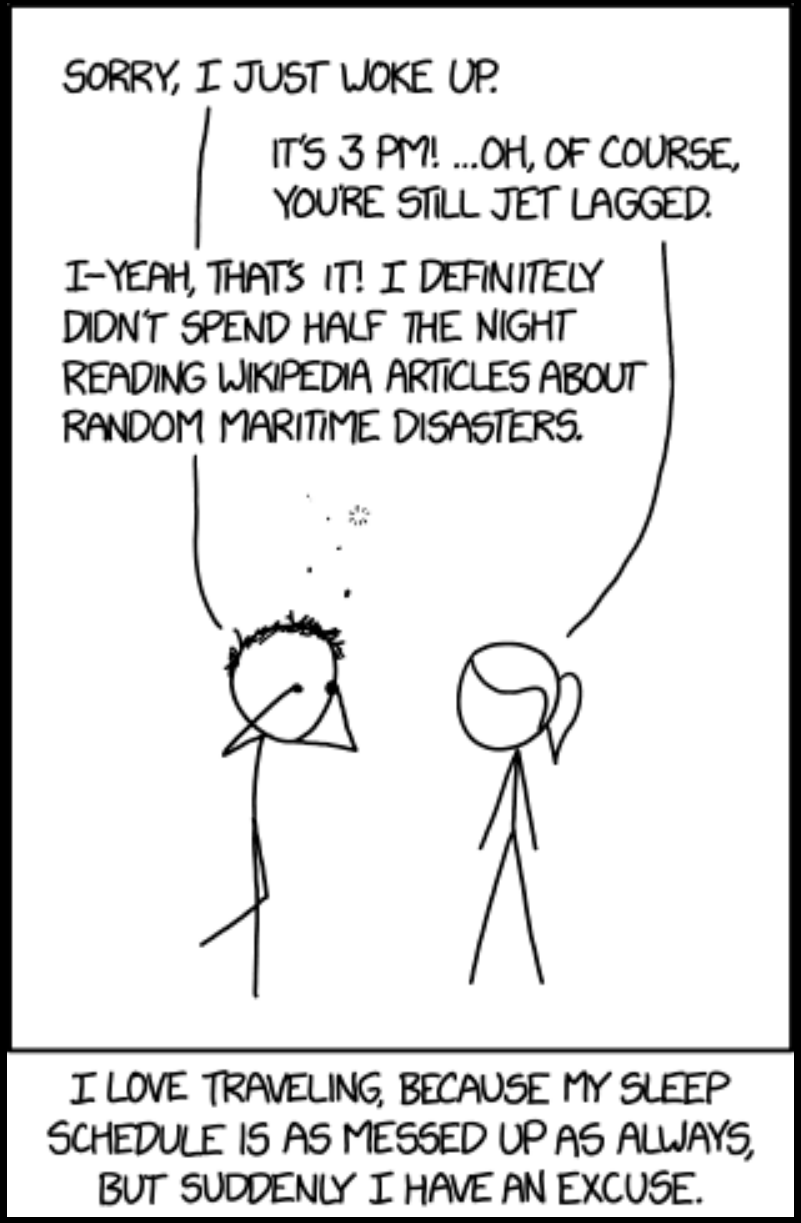
```rust
fn slightly_less_contrived(input: &str){
        let mut vec = vec!(input);

        unsafe {
                vec.set_len(10);
        }
        ...
}
```

Buffer overflow in Rust

```rust
fn actual_rust_code_i_might_write(input: &mut str)
  -> Vec<u8> {
  let mut vec = "this is a secret".as_bytes().to_vec();
  vec.append(&mut input.as_bytes().to_vec());
  vec
}
```

Rust + C buffer overflow



The Rust Stack

Rust won't let you put anything dynamically sized on the stack

Unsafe code

Unsafe code is not the wild, wild west

That doesn't mean that it's 'safe'

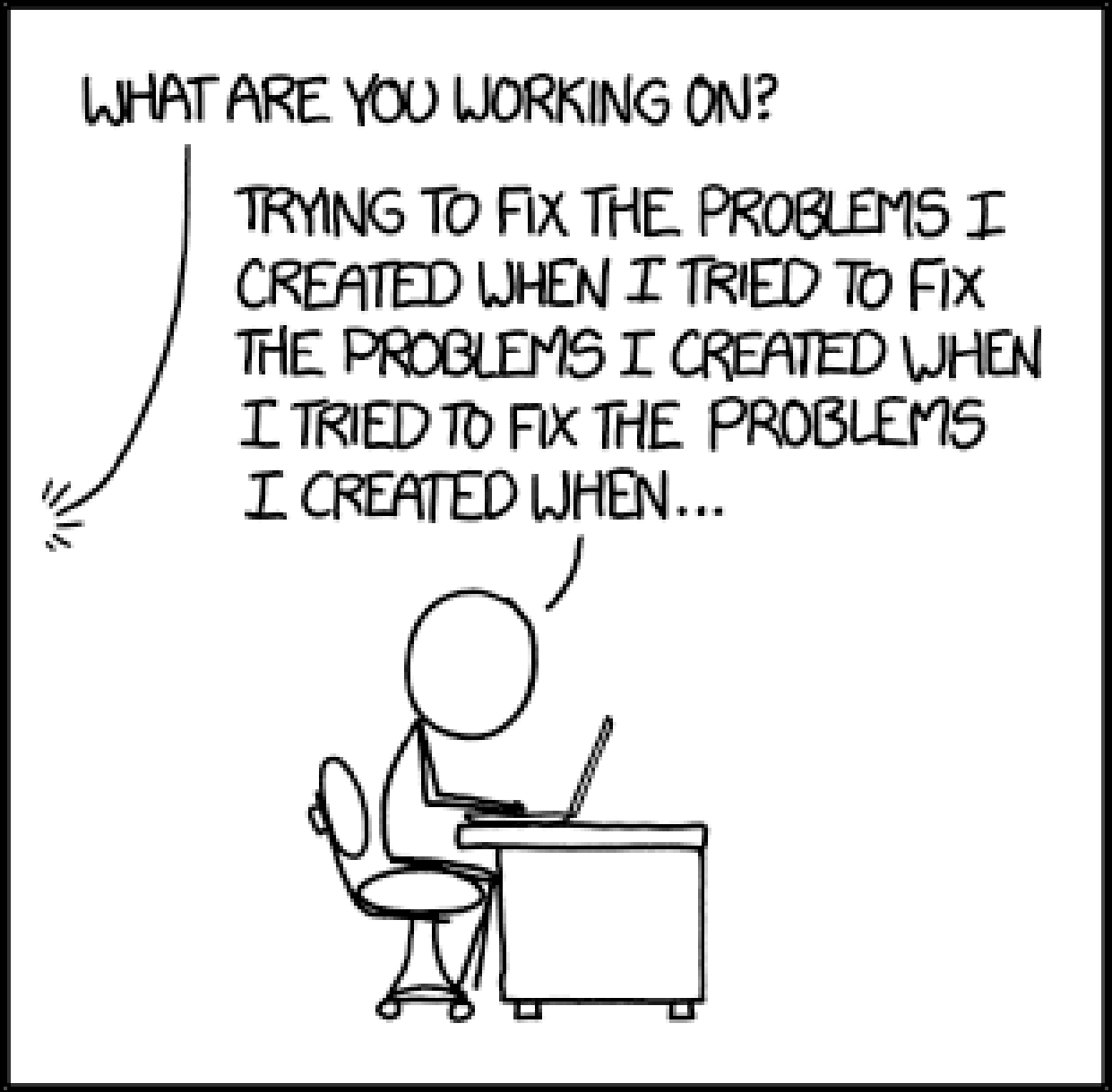
Rust just limits how terrible of a person you can be

Rust has made me a better programmer

Other "traditional" attacks

- format string attack?
- return oriented programming
- arithmetic overflow (prevented in debug builds)
- heap exploits

Demo


WHAT ARE YOU WORKING ON?
TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...

Questions?

- avadacatavra
- avadacatavra
- avadacatavra@mozilla.com
- avadacatavra.github.io/