how to spy on your programs with strace

in which we learn about...

★ how one standard Linux utility can make you a 'WIZARD';

★ why you should ♥ your operating system

★ that system calls are THE BEST (and what my favourites are !!)

$5.00 or trades ♥

CC-BY-NC-SA
Julia Evans, strace wizard now fun yay industries 2015
Hi! I'm Julia! I look kind of like this:

I found out one day that understanding your operating system's internals makes you a little more awesome. WAY BETTER PROGRAMMER.

I've written like 7 posts about strace because I have an unhealthy obsession:

http://jvns.ca/categories/strace

(In) Frequently asked questions:

Q: Is there strace on OS X?
A: No, but you can use dtrace/dtruss and it's actually much more powerful!

Q: Can I strace strace?
A: Yup! It uses the ptrace system call.

Q: Can I strace PID 1 (init)?
A: APPARENTLY YES! (use extreme caution :]

Q: Should I strace my production database?
A: NONONONO. It will run MUCH more slowly never do this.

I write more like this at

blog: jvns.ca
twitter: @b0rk
email: julia@jvns.ca
That's it! Now you're a **WIZARD**

more seriously obviously there's a TON more to learn about operating systems and many further levels of wizardry, but I find strace by itself to be an incredibly useful tool.

And so fun! Once on a 12-hour train ride from New York to Montreal I had no book and no internet so I just started stracing programs on my computer and I could totally see how killall worked without reading the source code or ANYTHING.

also it helps me debug all the time ❤️

❤️ a tiny manifesto ❤️

operating systems are

❤️ AWESOME ❤️

the strace zine thinks:

- your computer is yours
- your OS is yours
- open licenses mean you can read and change the code!!
- Linux is REALLY COOL
- just because some Linux kernel devs (cough Linus cough) act like jerks doesn’t mean we can’t still learn AWESOME STUFF❤️

LET'S GO LEARN❤️it’s really fun❤️
What is this strace thing???

Strace is a program on Linux that lets you inspect what a program is doing without

- a debugger
- or the source
- or even knowing the programming language at all (?!? how can it be!)

Sometimes I'm looking at the output of a recvfrom and it's like
recvfrom(6, "And then the monster... ")
and OH NO THE SUSPENSE

strace -s 800 will show you the first 800 characters of each string. I use it all the time ★

Let's get real. No matter what, strace prints too much damn output. Use
strace -o too_much_stuff.txt
and sort through it later.

Putting it all together:

Let's say you wanted to spy on a ssh session!

strace -f -o ssh.txt ssh juliabox

Or see what files a Dropbox sync process is opening (made up PID: 230)

strace -f -p 230 -e open

basically strace makes you a WIZARD 😎

To understand how this works, let's talk a little about operating systems
strace command line flags I love

-e
overwhelmed by all the system calls you don't understand? Try

    strace -e open

and it'll just show you the opens... much simpler.

-f
Does your program start? Subprocesses? Lots use! -f to see what those are doing too.
Or just always use -f! That's what I do.

-p
p is for pid

"OH NO I STARTED THE PROGRAM 6 HOURS AGO AND NOW I WANT TO STRACE IT"

do not worry! Just find your process's PID (like 747) and

    strace -p 747

Why you should ♥ your operating system♥

Some things it does for you:

- understand how your hard drive works and how the filesystem on it organizes the bytes into files so you can just read your damn file 😊
- run code every time you press a key so that you can type
- implement networking protocols like TCP/IP so that you can get webpages, pictures of cats from the internet
- keep track of all the memory every process is using!
- basically know everything about how all your hardware works so you can just write programs! 😊

So great 😊
but wait, Julia, how do my programs use all this great stuff the operating system does?

you

yes!

wow!

amazing!

you

System calls are the API for your operating system.

interface

System calls are the API for your operating system.

System calls are the API for your operating system.

System calls are the API for your operating system.

System calls are the API for your operating system.

System calls are the API for your operating system.

Send to + recv from

What's fun? Spying on network activity is fun. If you have a HTTP service or and you're debugging and totally at your wits' end, maybe it's time to look at what's really exactly being sent over the network...

these are your pals

note: network activity can show up in read and write syscalls too. We saw that in the SSH example!

want to open a file? use open and then read and write to it

Send data over a network? Use socket to open a connection and sendto and recvfrom pictures of cats

execute

My first day at work, a Ruby script that ran some ssh commands wasn't working. Oh no!

But who wants to read code to find out why? ugh.

strace -f -e execve ./script.rb!

told us what the problem ssh command was, and we fixed it!

Every program on your computer is using system calls all the time to manage memory, write files, do networking, and lots of other stuff.
my favorite system calls

Have you ever not been sure what configuration files a program is using? THAT NEVER NEEDS TO HAPPEN TO YOU AGAIN 😎. Skip the docs and head straight for

```
strace -f -e open mplayer Rick.Astley.mp3
```

psst: I'm going to explain -e and -f in a couple of pages 😊

open

a first cup of strace

You might think with all this talk of operating systems and system calls that using strace is hard.

It's easy! If you have a Linux machine I want you to try it RIGHT NOW

```
strace ls
```

wizard time!

write

Programs write logs.

```
write(fd, "OH NOEZ")
```

If you're sure your program is writing Very Important Information but don't know what or where, `strace -e write` may be for you.

write

There's a LOT of output and it's pretty confusing at first. I've annotated some for you on the next page 😊

write

try stracing more programs! Google the system calls! Don't worry if you don't understand everything! I sure don't! 😊
Let's explain just a couple more things!

```
still the name of the syscall
file to open
open the file with read/write permissions
```

The 3 here is a file descriptor number, which
Internally Linux tracks files with numbers! You can see
all the file descriptors for process id 42, and what
they point to by doing

```
ls-l /proc/42/fd
```

(file descriptor) (what got read) (number of bytes)

If you don’t understand something in your strace output:

* me too! It’s normal!
* try reading the man page for the system call!
* remember that just understanding read/write/open/execve
  can take you a long way!