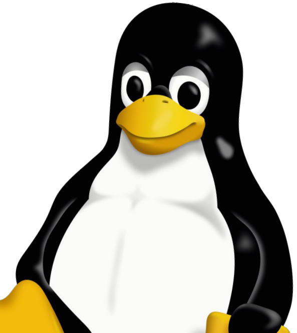
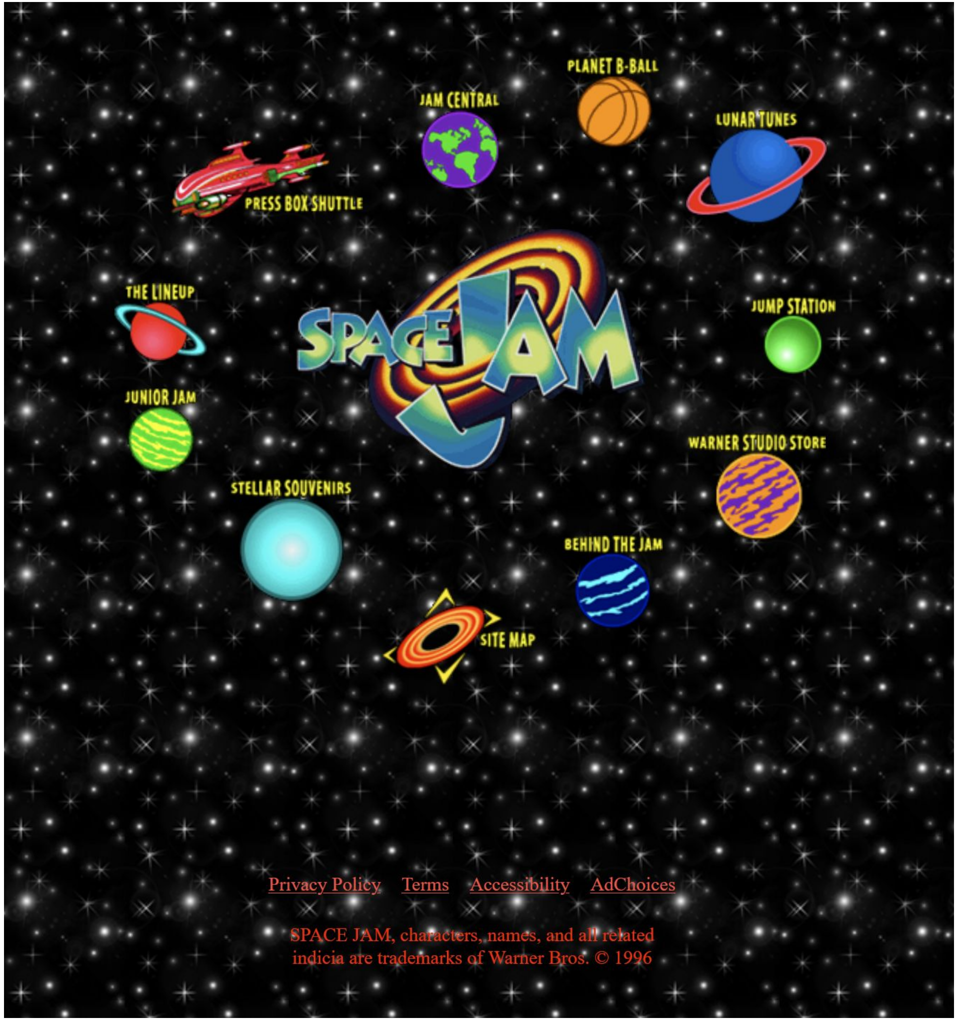
Protecting the Penguin: Linux Security Monitoring with eBPF

by Sam

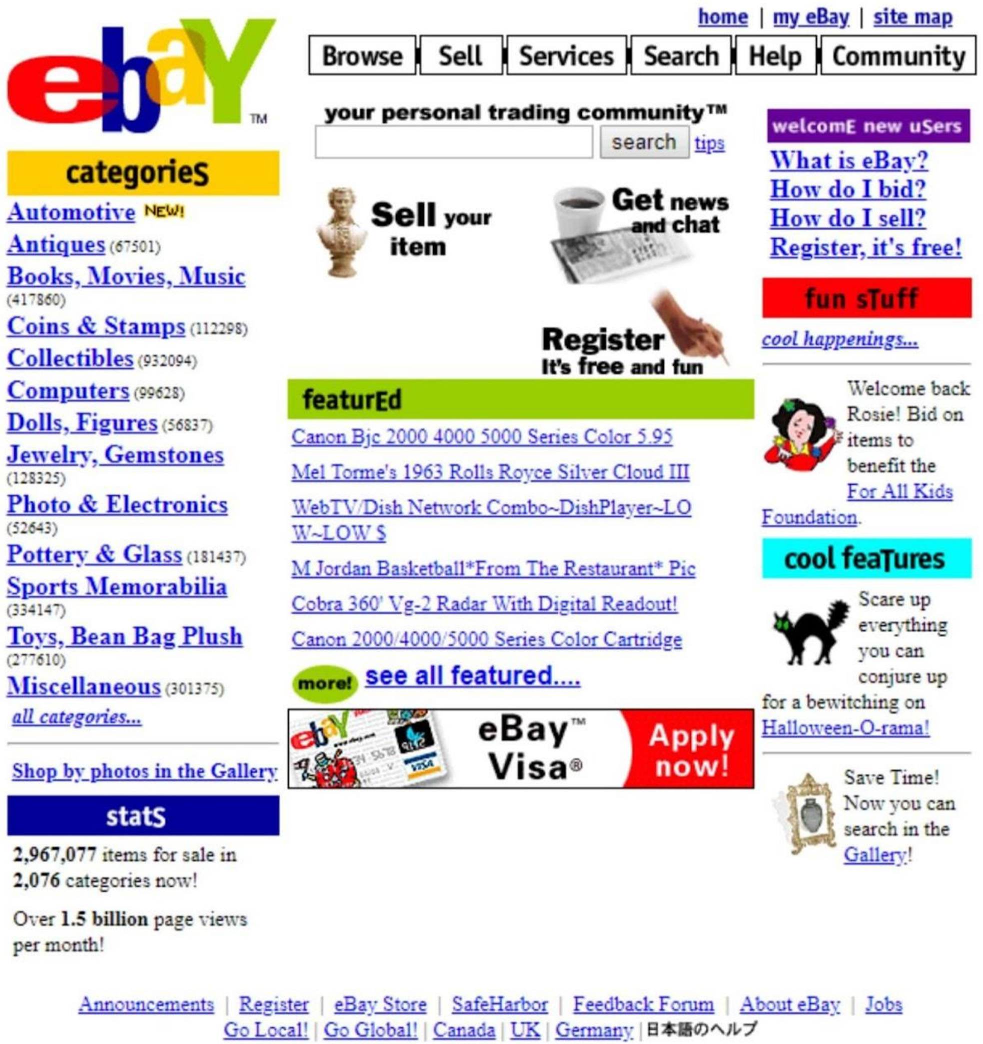


who am i

- Sam
- Ethical Hacking grad (2021)
- Security Analyst at Jane Street
- Linux nerd
- Vegan btw



SPACE JAM, characters, names, and all related indicia are trademarks of Warner Bros. © 1996



your personal trading community™

categories

Automotive NEW!
Antiques (67501)
Books, Movies, Music (417860)
Coins & Stamps (112298)
Collectibles (932094)
Computers (99628)
Dolls, Figures (56837)
Jewelry, Gemstones (128325)
Photo & Electronics (52643)
Pottery & Glass (181437)
Sports Memorabilia (334147)
Toys, Bean Bag Plush (277610)
Miscellaneous (301375)
all categories...

Shop by photos in the Gallery

stats

2,967,077 items for sale in **2,076** categories now!

Over **1.5 billion** page views per month!

Sell your item

Get news and chat

Register It's free and fun

featured

Canon Bjc 2000 4000 5000 Series Color 5.95
Mel Torme's 1963 Rolls Royce Silver Cloud III
WebTV/Dish Network Combo~DishPlayer~LOW~LOW $
M Jordan Basketball*From The Restaurant* Pic
Cobra 360' Vg-2 Radar With Digital Readout!
Canon 2000/4000/5000 Series Color Cartridge

see all featured....

welcome new users

What is eBay?
How do I bid?
How do I sell?
Register, it's free!

fun stuff

cool happenings...

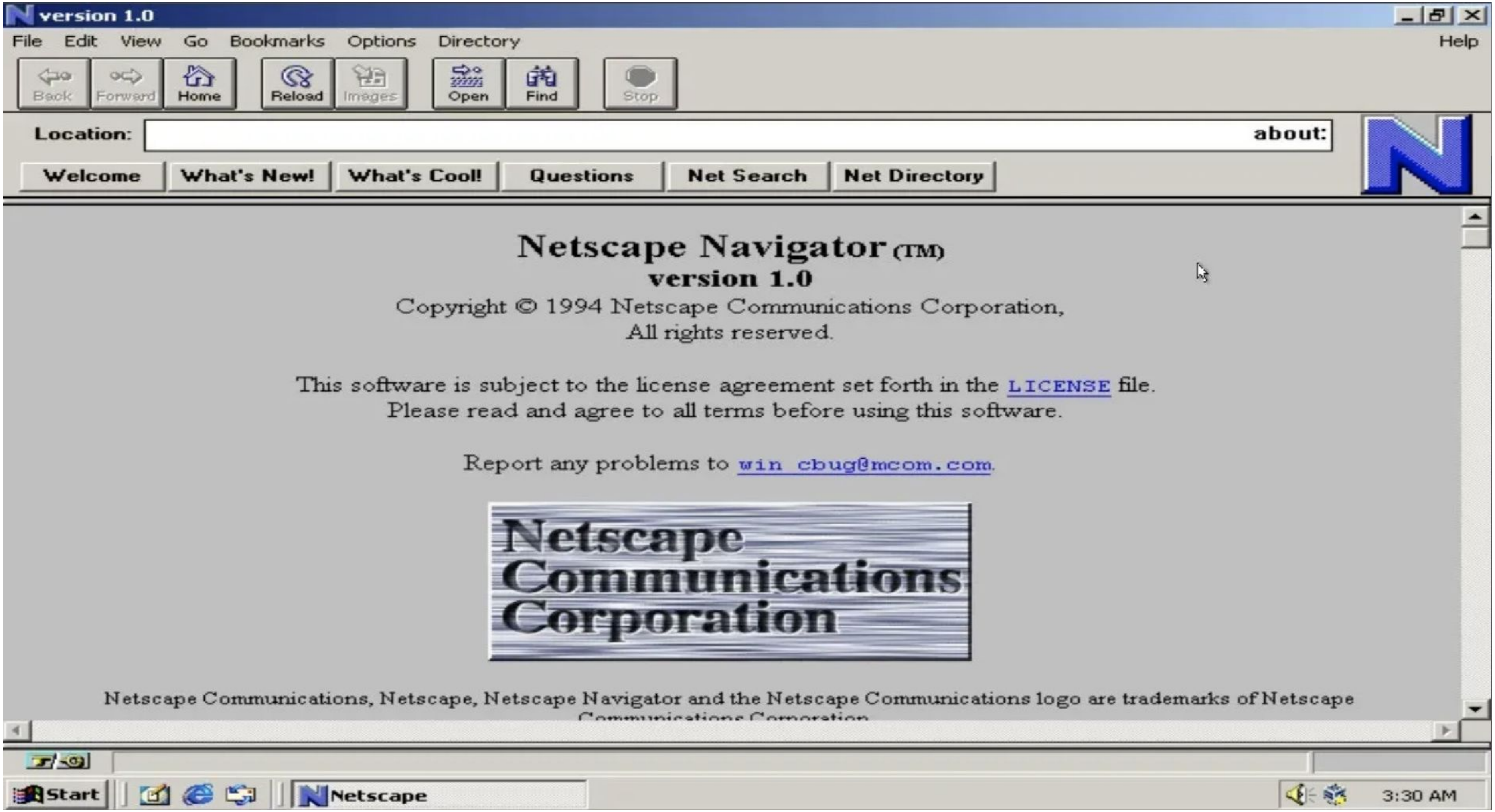
Welcome back Rosie! Bid on items to benefit the For All Kids Foundation.

cool features

Scare up everything you can conjure up for a bewitching on Halloween-O-rama!

Save Time! Now you can search in the Gallery!

Announcements | Register | eBay Store | SafeHarbor | Feedback Forum | About eBay | Jobs
Go Local! | Go Global! | Canada | UK | Germany | 日本語のヘルプ



Netscape Navigator (TM)

version 1.0

Copyright © 1994 Netscape Communications Corporation,
All rights reserved.

This software is subject to the license agreement set forth in the LICENSE file.
Please read and agree to all terms before using this software.

Report any problems to win_cbug@mcom.com.

Netscape
Communications
Corporation

Netscape Communications, Netscape, Netscape Navigator and the Netscape Communications logo are trademarks of Netscape Communications Corporation

How did we get here??

Get cash for the festive season

Sell what you no longer need to fund your holiday shopping.

Deals for you

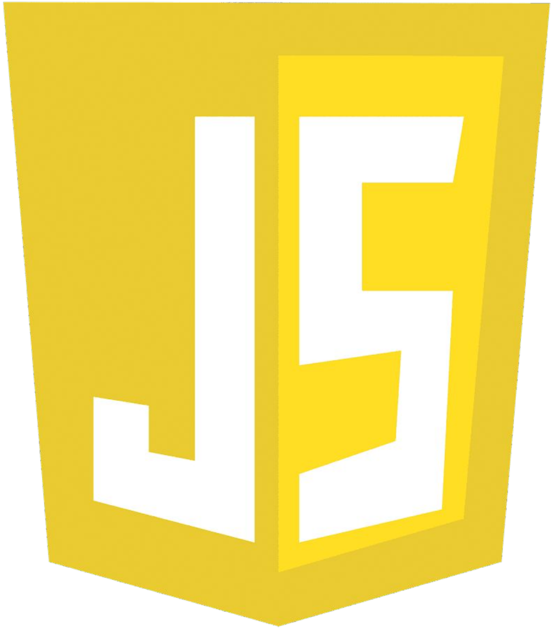
Grab a must-have saving.

Trending on eBay

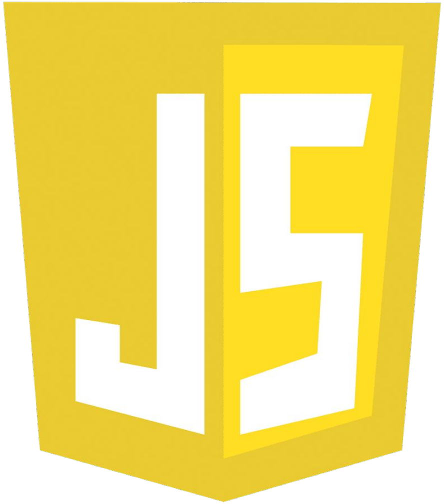
We use cookies and other technologies that are essential to provide you our services and site functionality, as described in our Cookie Notice. If you agree and click 'Accept all', we will also allow 23 third-party companies, who we partner with, to store cookies on your device and use similar technologies to collect and use your personal data (e.g., IP address) for advertising personalisation, measurement and analytics, on and off our sites. You may decline consent by clicking 'Decline all' or make granular choices by clicking 'Customise my choices'. You may withdraw your consent at any time by visiting https://www.ebay.co.uk/gdpr

Data processing purposes: Store and/or access information on a device. Personalised advertising, advertising measurement, audience research and services development.

JavaScript



JavaScript



Makes the **browser programmable**

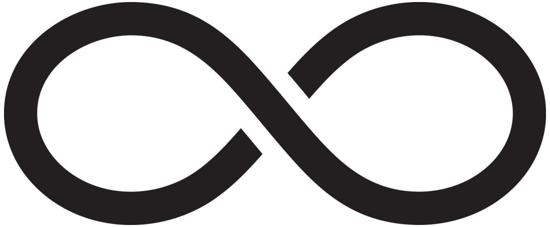
Arbitrary code? In my browser?🤯

Safety

Untrusted code can’t run wild, we need

sandboxing

Continuous Delivery



New functionality without shipping new browser, we need

easy integration

Performance

There should be minimal overhead, so we need

native execution

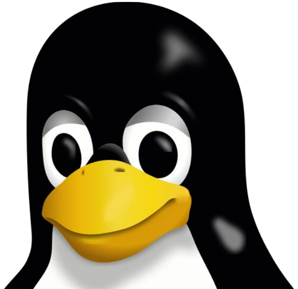
Arbitrary code? In my browser?🤯

- **Sandboxing**: each website is isolated
- **Easy integration**: it just runs
- **Native execution**: JIT compiler

JavaScript

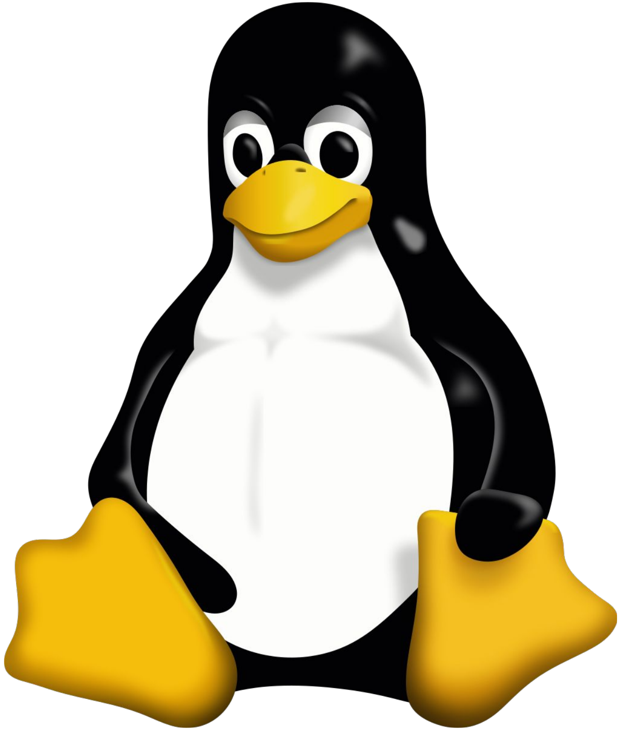
The **browser** is now **programmable**

What is Linux?

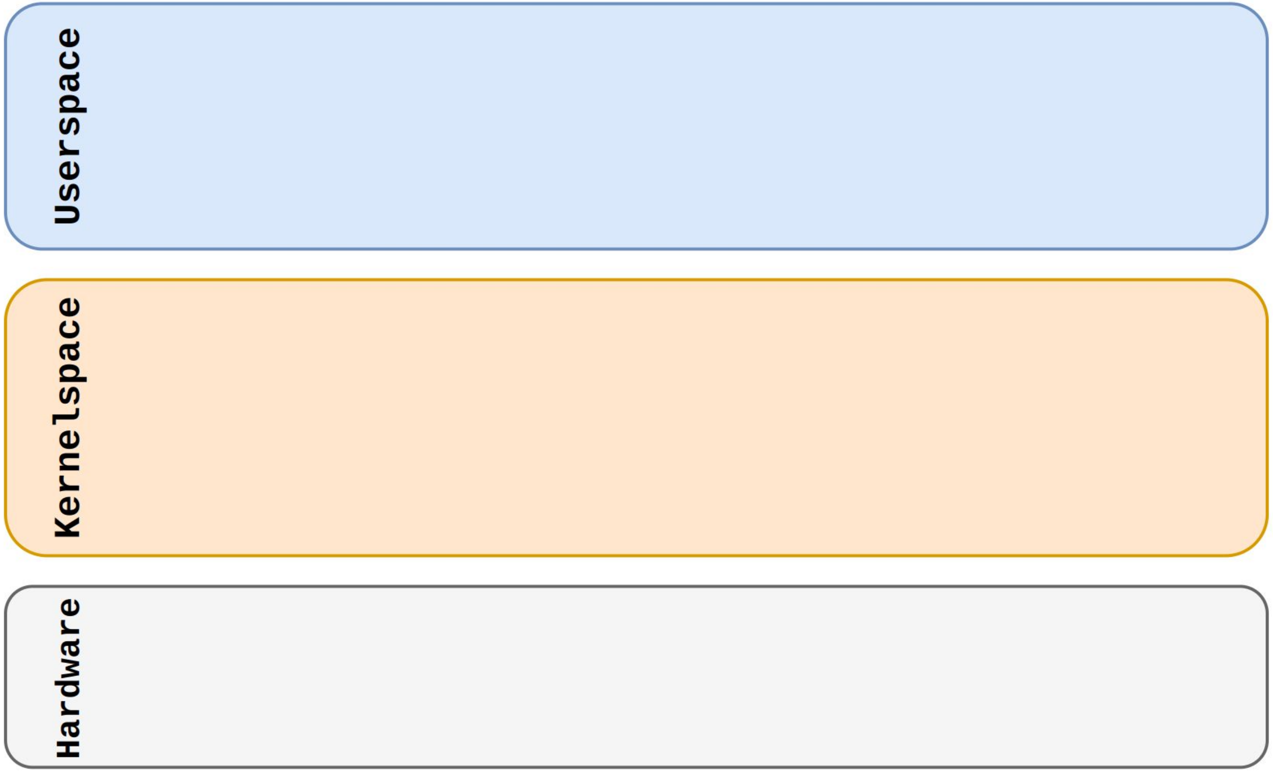


What is Linux?

- It’s an Operating System
- Runs on billions of devices globally
- Free and open source
- Uses the **Linux kernel**



What is Linux?

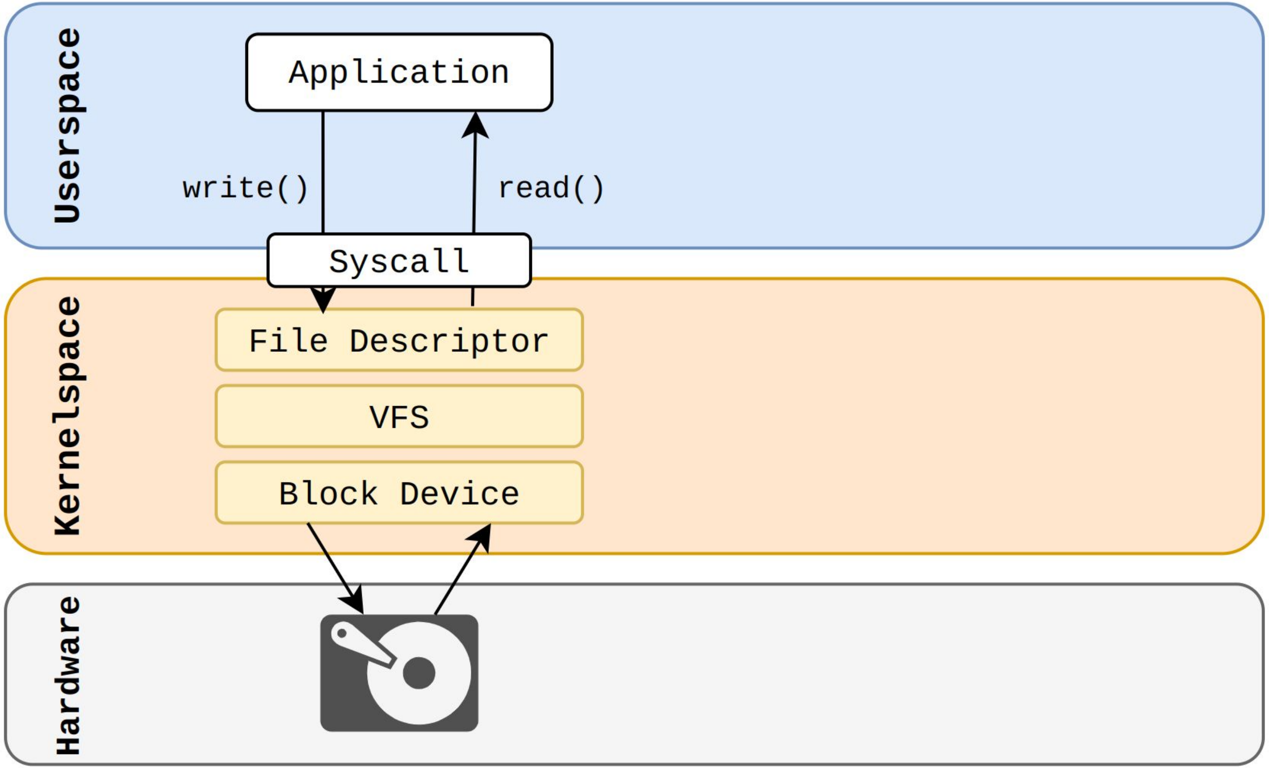


Userspace

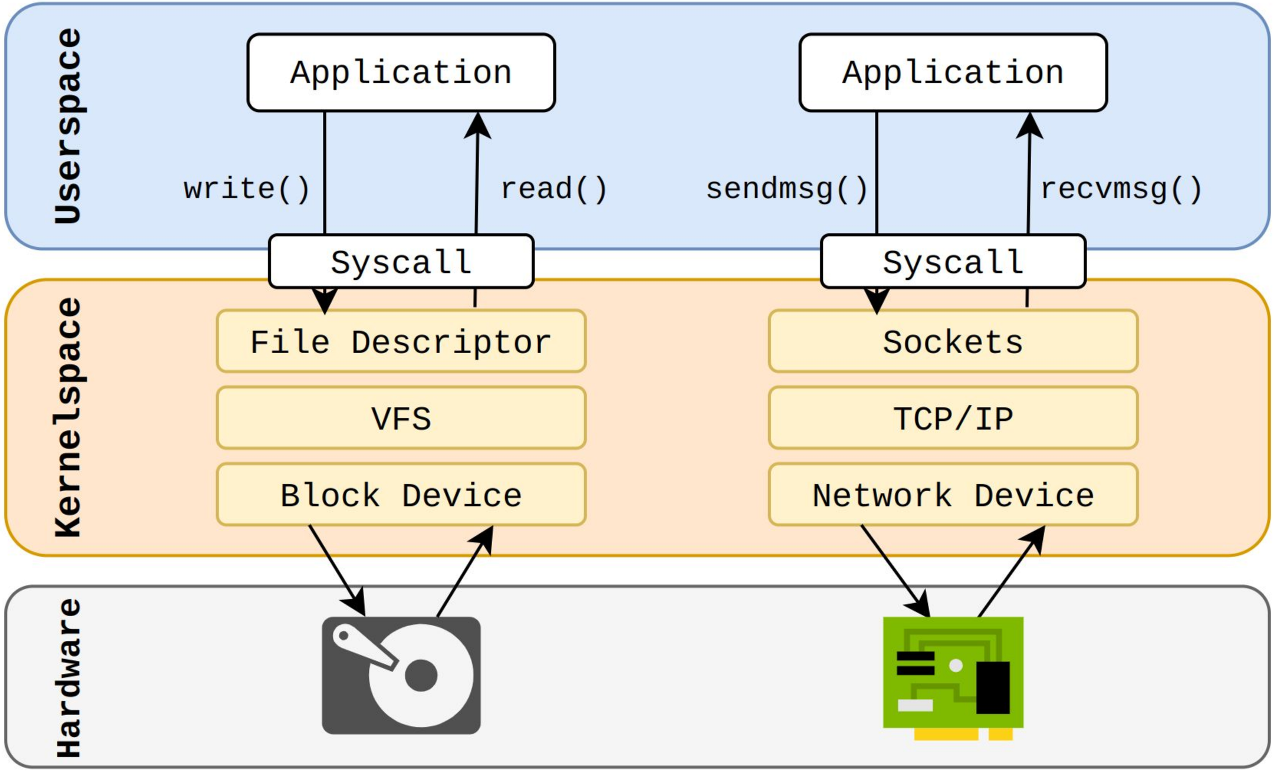
Kernelspace

Hardware

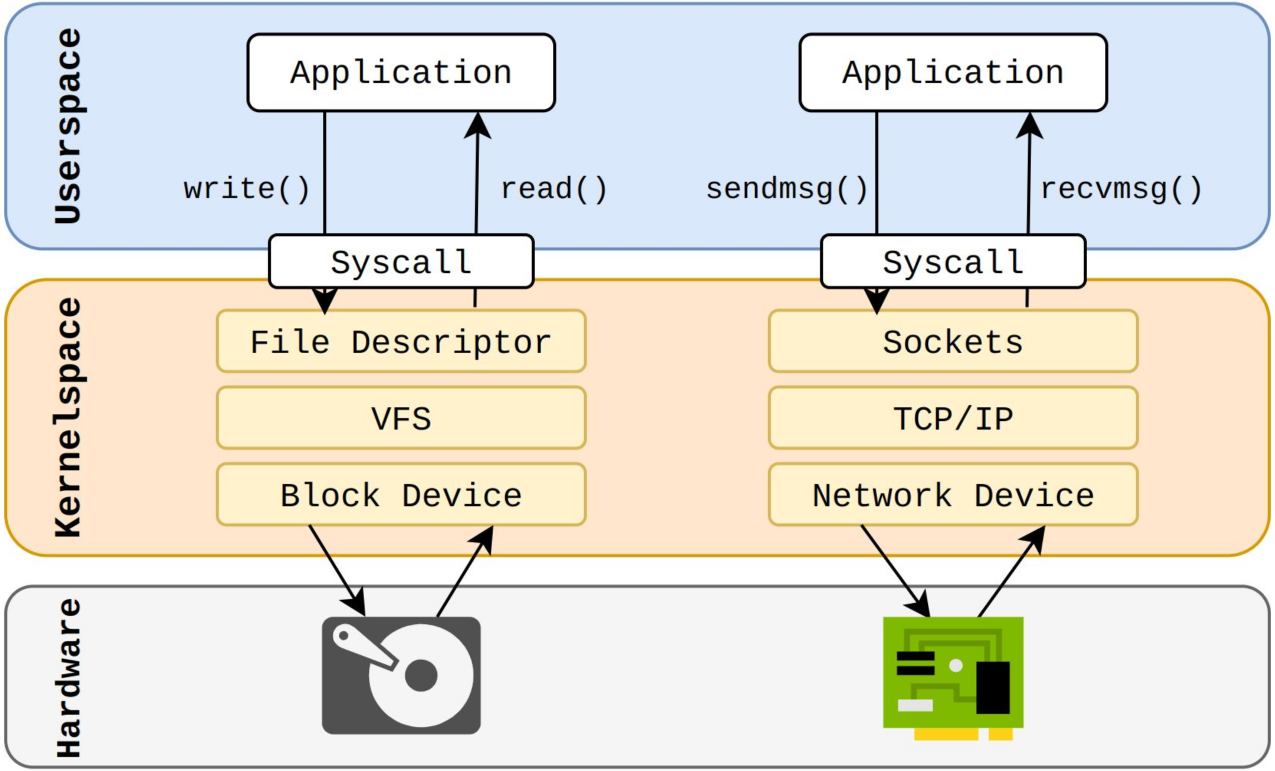
What is Linux?



What is Linux?



What is Linux?



what if we want more?

Extending Kernel

Option 1: **Native Support**

- Change kernel source code
- Email Linus Torvalds
- Wait a few years for your changes to land
- Wait 5 years for users to upgrade kernel

Extending Kernel

Option 1: **Native Support**

- Change kernel source code
- Email Linus Torvalds
- Wait a few years for your changes to land
- Wait 5 years for users to upgrade kernel
- Climate change destroy planet

Extending Kernel

Option 1: **Native Support**

- Change kernel source code
- Email Linus Torvalds
- Wait a few years for your changes to land
- Wait 5 years for users to upgrade kernel
- Climate change destroy planet

Option 2: **Kernel Module**

- Write kernel module
- Compile a few dozen versions
- Create a new package for every distro

Extending Kernel

Option 1: **Native Support**

- Change kernel source code
- Email Linus Torvalds
- Wait a few years for your changes to land
- Wait 5 years for users to upgrade kernel
- Climate change destroy planet

Option 2: **Kernel Module**

- Write kernel module
- Compile a few dozen versions
- Create a new package for every distro
- Every new kernel release might break it
- If you get it wrong your kernel will crash

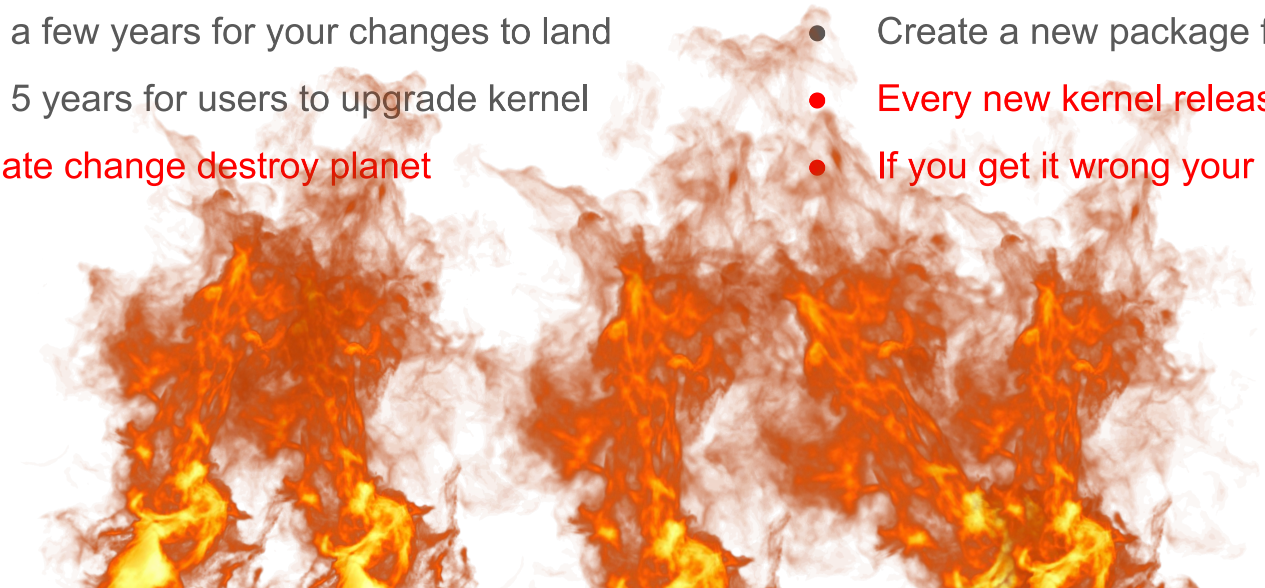
Extending Kernel

Option 1: **Native Support**

- Change kernel source code
- Email Linus Torvalds
- Wait a few years for your changes to land
- Wait 5 years for users to upgrade kernel
- Climate change destroy planet

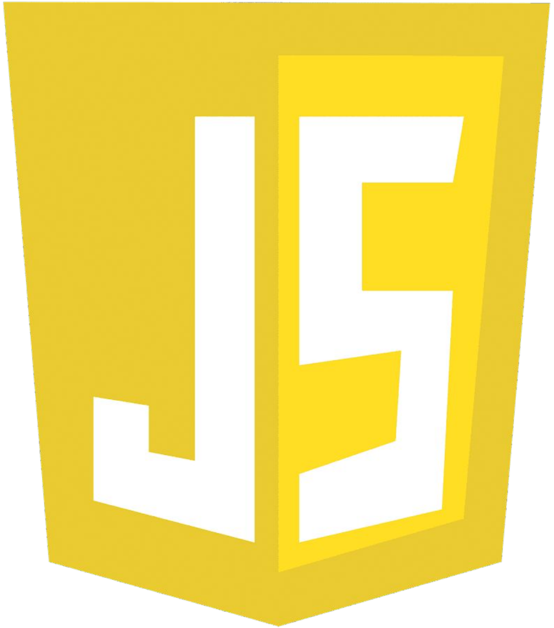
Option 2: **Kernel Module**

- Write kernel module
- Compile a few dozen versions
- Create a new package for every distro
- Every new kernel release might break it
- If you get it wrong your kernel will crash



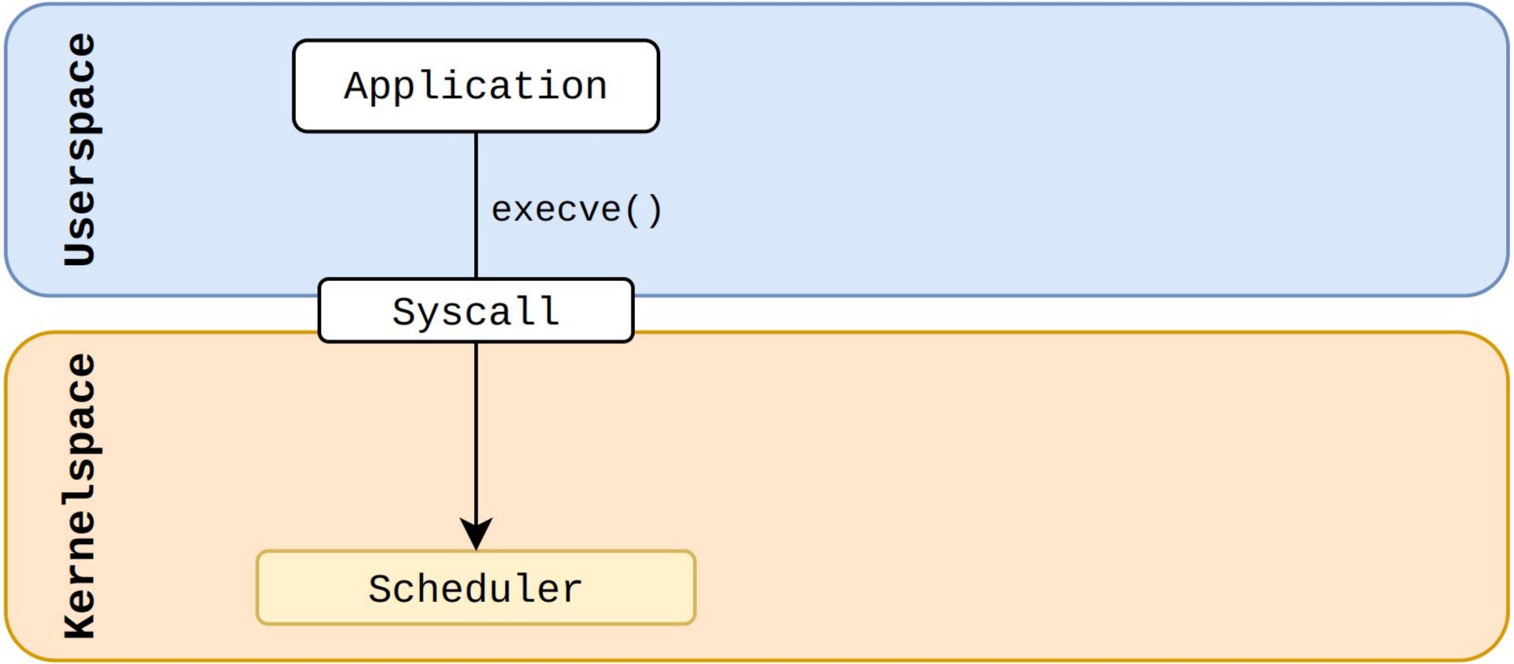
+

JavaScript

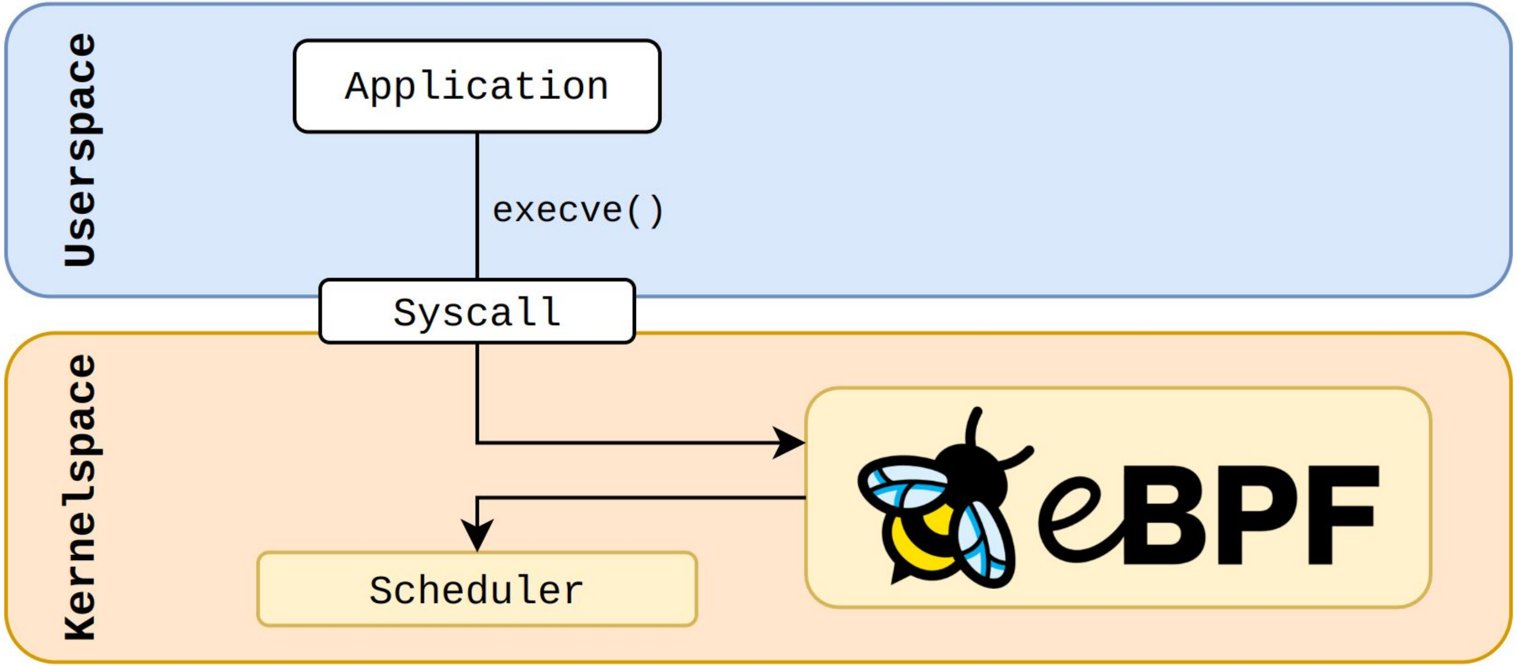


eBPF

What is eBPF?


Userspace
Application
execve()
Syscall
Kernelspace
Scheduler

What is eBPF?


Userspace
Application
execve()
Syscall
Kernelspace
eBPF
Scheduler

Arbitrary code? In my kernel?🤯

Safety

eBPF Verifier: rejects any unsafe program and provides sandboxing

Arbitrary code? In my kernel?🤯

Safety

eBPF Verifier: rejects any unsafe program and provides sandboxing

Performance

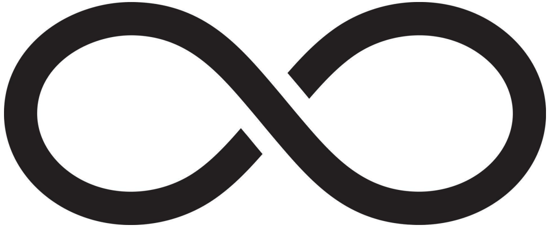
JIT Compiler: generic bytecode compiled to native CPU architecture

Arbitrary code? In my kernel?🤯

Safety

eBPF Verifier: rejects any unsafe program and provides sandboxing

Continuous Delivery



eBPF Hooks: programs can be attached, detached and replaced atomically

Performance

JIT Compiler: generic bytecode compiled to native CPU architecture

What is eBPF?

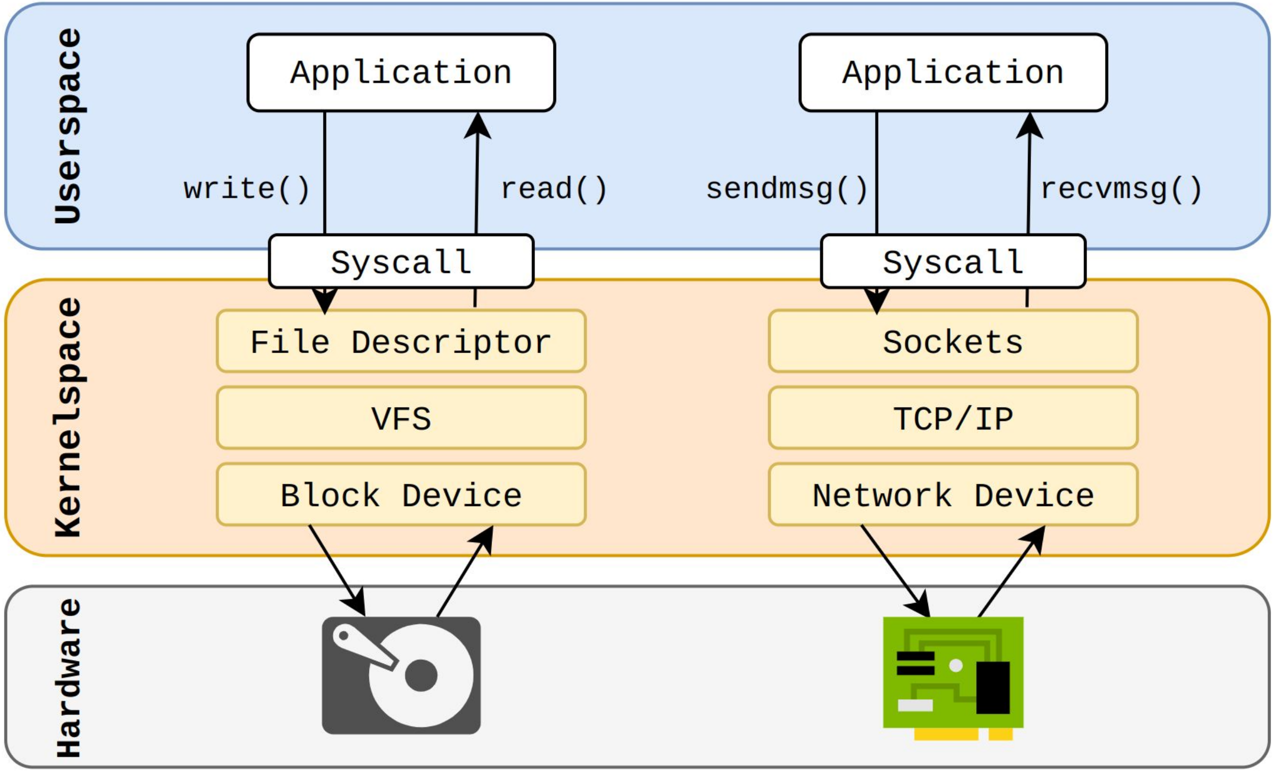
eBPF makes the **kernel programmable**

extended **B**erkeley **P**acket **F**ilter

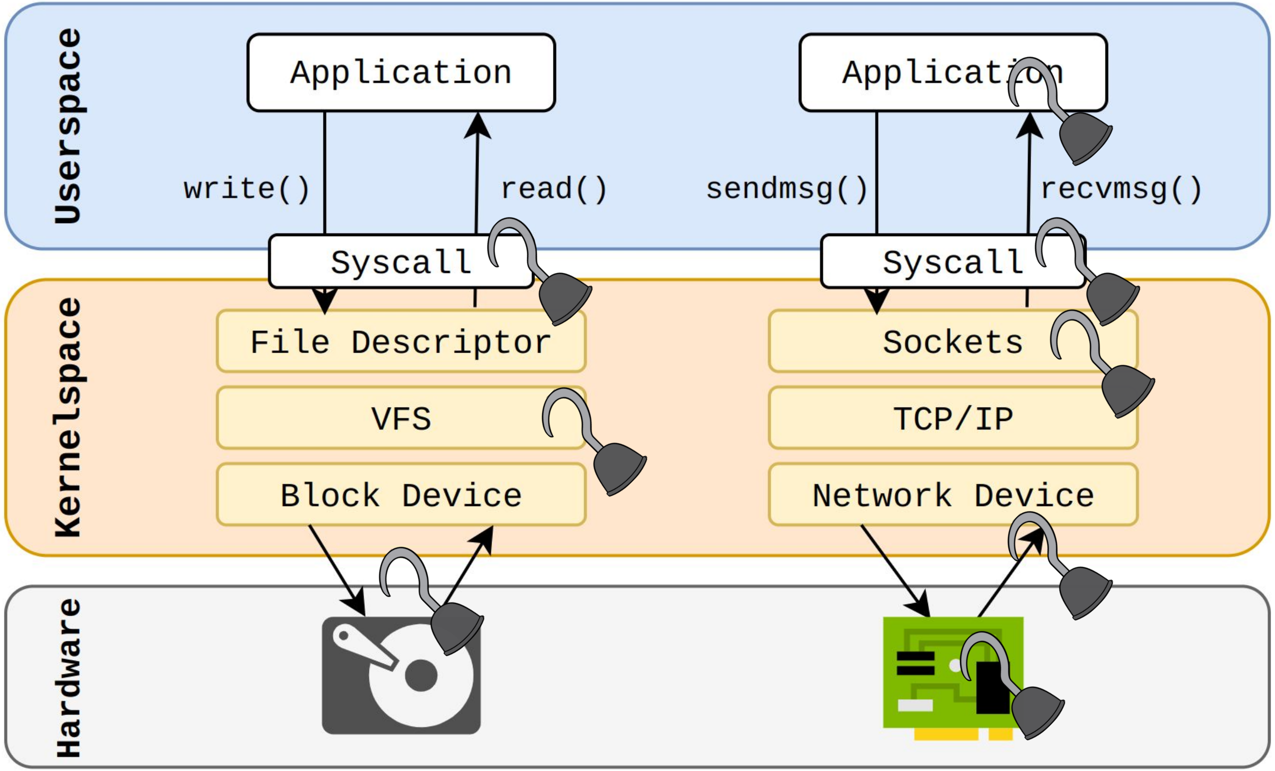
eBPF hooks

- kprobes
- uprobes
- Tracepoints
- Network packets
- Linux security modules
- Perf events
- etc…

eBPF hooks


Userspace
Application
write()
read()
Application
sendmsg()
recvmsg()
Syscall
Syscall
Kernelspace
File Descriptor
VFS
Block Device
Sockets
TCP/IP
Network Device
Hardware

eBPF hooks


Userspace
Kernelspace
Hardware
Application
write()
read()
Syscall
File Descriptor
VFS
Block Device
Application
sendmsg()
recvmsg()
Syscall
Sockets
TCP/IP
Network Device

Demo

eBPF Hello World

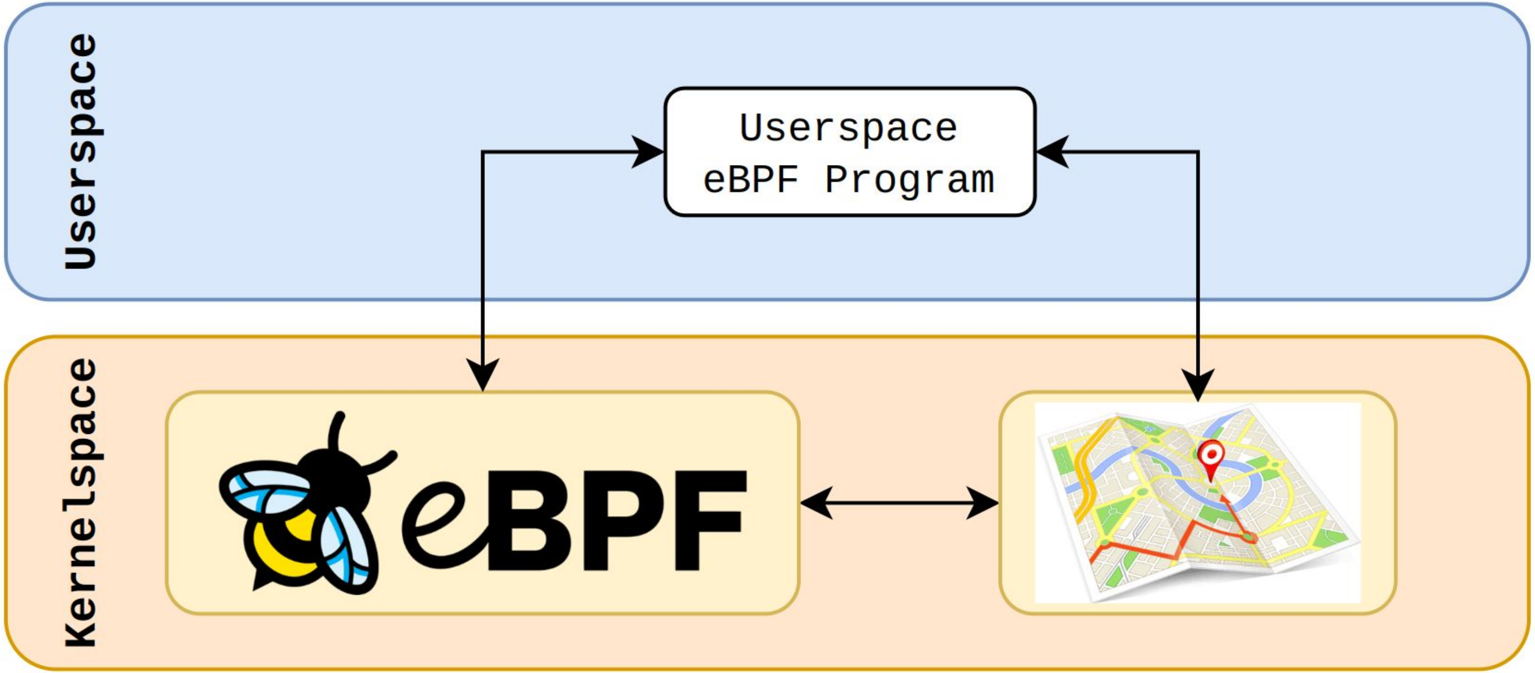
```
SEC("kprobe/__x64_sys_fchmodat")
int demo(void *ctx)
{
    bpf_printk("chmod happened!!\n");
    return 0;
}
```

```
output:
  <...>-123021  [005] d..31 452659.744965: bpf_trace_printk: chmod happened!!
  <...>-123040  [000] d..31 452660.525742: bpf_trace_printk: chmod happened!!
  <...>-123060  [000] d..31 452661.354995: bpf_trace_printk: chmod happened!!
```

eBPF Hello World

- Userspace program makes that syscall
- eBPF application executes
- bpf_trace_printk(), a helper function, is called
- Writes to `/sys/kernel/debug/tracing/trace_pipe`
- Not that useful! We need some more tools

eBPF Maps



eBPF Maps

```
enum bpf_map_type {
        BPF_MAP_TYPE_UNSPEC,
        BPF_MAP_TYPE_HASH,
        BPF_MAP_TYPE_ARRAY,
        BPF_MAP_TYPE_PROG_ARRAY,
        BPF_MAP_TYPE_PERF_EVENT_ARRAY,
        BPF_MAP_TYPE_RINGBUF,
        BPF_MAP_TYPE_PERCPU_HASH,
        BPF_MAP_TYPE_PERCPU_ARRAY,
        BPF_MAP_TYPE_STACK_TRACE,
        BPF_MAP_TYPE_CGROUP_ARRAY,
        BPF_MAP_TYPE_LRU_HASH,
        BPF_MAP_TYPE_LRU_PERCPU_HASH,
};
```

eBPF Maps

```
enum bpf_map_type {
        BPF_MAP_TYPE_UNSPEC,
        BPF_MAP_TYPE_HASH,
        BPF_MAP_TYPE_ARRAY,
        BPF_MAP_TYPE_PROG_ARRAY,
        BPF_MAP_TYPE_PERF_EVENT_ARRAY,
        BPF_MAP_TYPE_RINGBUF,
        BPF_MAP_TYPE_PERCPU_HASH,
        BPF_MAP_TYPE_PERCPU_ARRAY,
        BPF_MAP_TYPE_STACK_TRACE,
        BPF_MAP_TYPE_CGROUP_ARRAY,
        BPF_MAP_TYPE_LRU_HASH,
        BPF_MAP_TYPE_LRU_PERCPU_HASH,
};
```

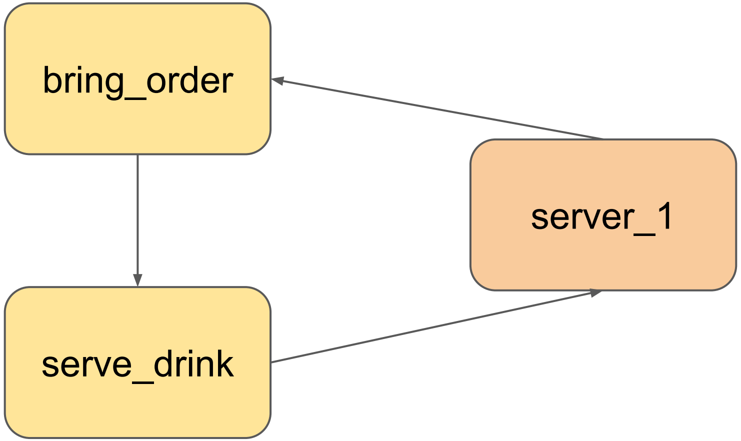
eBPF Maps

```
struct bpf_map_def SEC("maps") my_map = {
      .type = BPF_MAP_TYPE_ARRAY,
      .key_size = sizeof(u32),
      .value_size = sizeof(long),
      .max_entries = 256,
};

u32 index = 42;
long *value;
value = bpf_map_lookup_elem(&my_map, &index);
      if (value)
              __sync_fetch_and_add(value, 1);
```

eBPF Tail and Function Calls

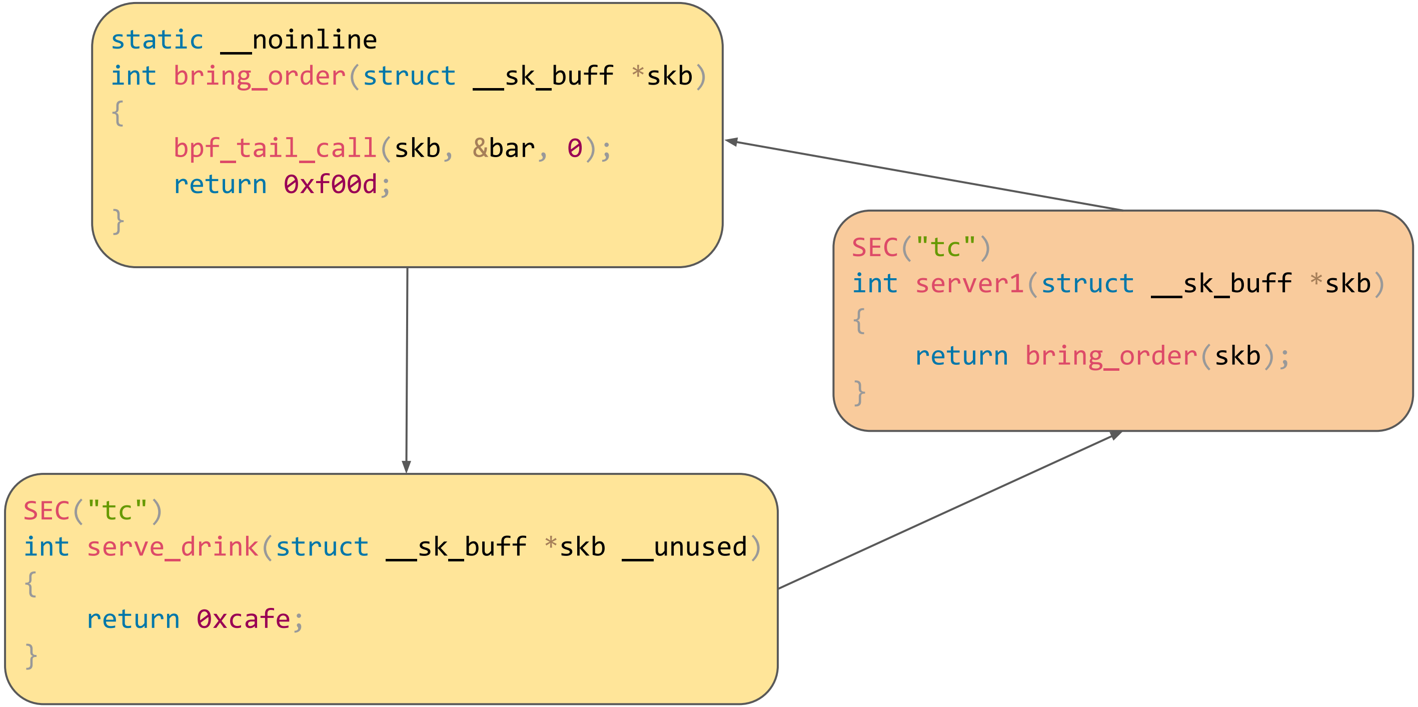
Programs can call other programs!



eBPF Tail and Function Calls

```
struct {
    __uint(type, BPF_MAP_TYPE_PROG_ARRAY);
    __uint(max_entries, 1);
    __uint(key_size, sizeof(__u32));
    __uint(value_size, sizeof(__u32));
} bar SEC(".maps");
```

eBPF Tail and Function Calls



```
static __noinline
int bring_order(struct __sk_buff *skb)
{
    bpf_tail_call(skb, &bar, 0);
    return 0xf00d;
}
```

```
SEC("tc")
int server1(struct __sk_buff *skb)
{
    return bring_order(skb);
}
```

```
SEC("tc")
int serve_drink(struct __sk_buff *skb __unused)
{
    return 0xcafe;
}
```

What is eBPF?

- Makes the **kernel programmable**
- Hooks let us manipulate and interact with kernel data
- eBPF maps let us maintain and share state
- Tail calls & Function calls let us compose larger programs

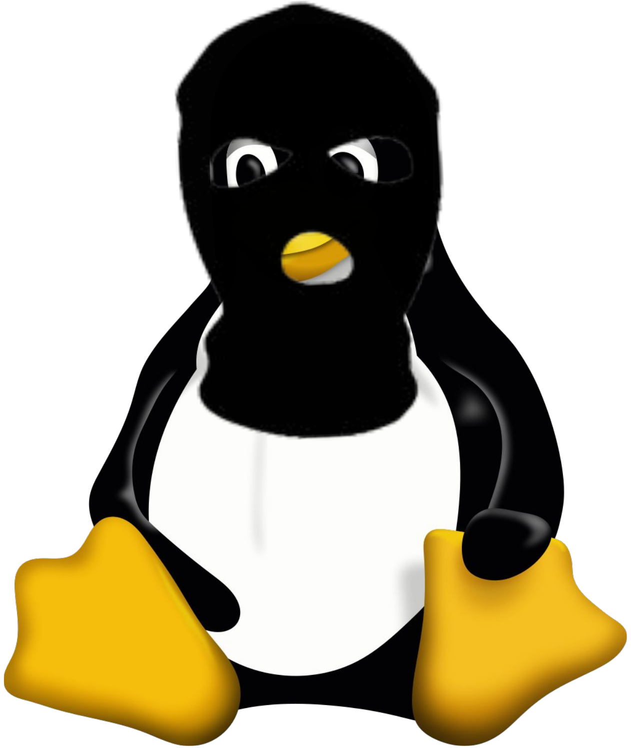
How does this make us secure?

What is Linux Security?

Linux Security

What do we care about?

- **Detecting** malicious activity
- **Reporting** malicious activity
- **Preventing** malicious activity



Linux Security

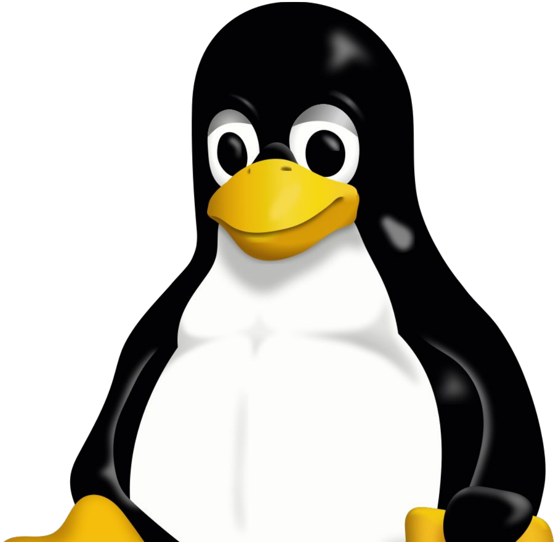
What is activity?

- Network traffic
- File interactions
- Running executables
- Changing privileges

All of this activity makes use of the **kernel**

Linux Security

- LD_PRELOAD
- seccomp
- ptrace
- kprobe tracing



LD_PRELOAD

- C library dynamically linked
- Built into Linux
- Enables hooking of any userspace function

LD_PRELOAD

- C library dynamically linked
- Built into Linux
- Enables hooking of any userspace function
- Bypassed by static linking!

Kernel syscall checks

- ptrace
- seccomp
- kprobe tracing (even with eBPF)

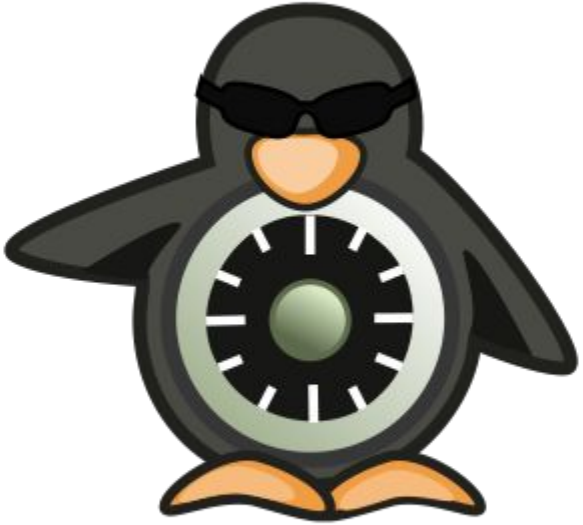
TOCTOU

- Time of Check Time of Use
- Entry point data can be spoofed

Look up DEFCON 29 Phantom Attacks

Linux Security Modules

- Stable, secure interface
- Safe way to introspect syscall data
- No TOCTOU!
- eBPF lets us access these dynamically

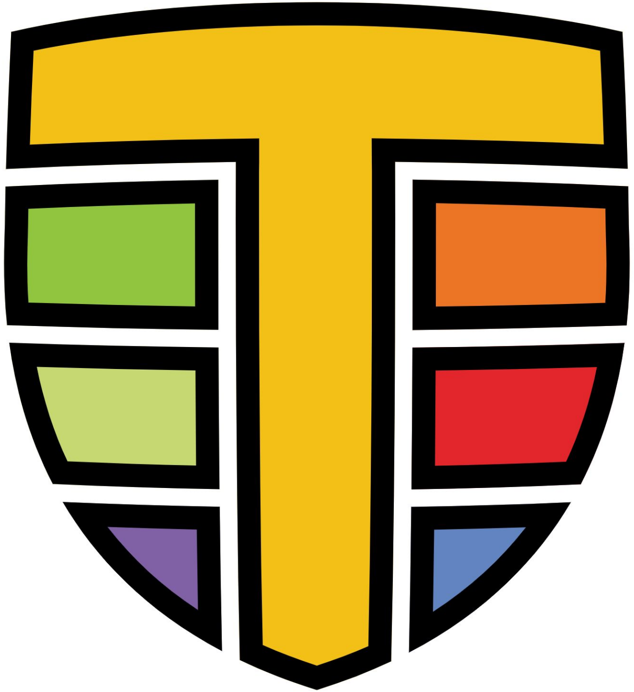


LSM Hook Demo

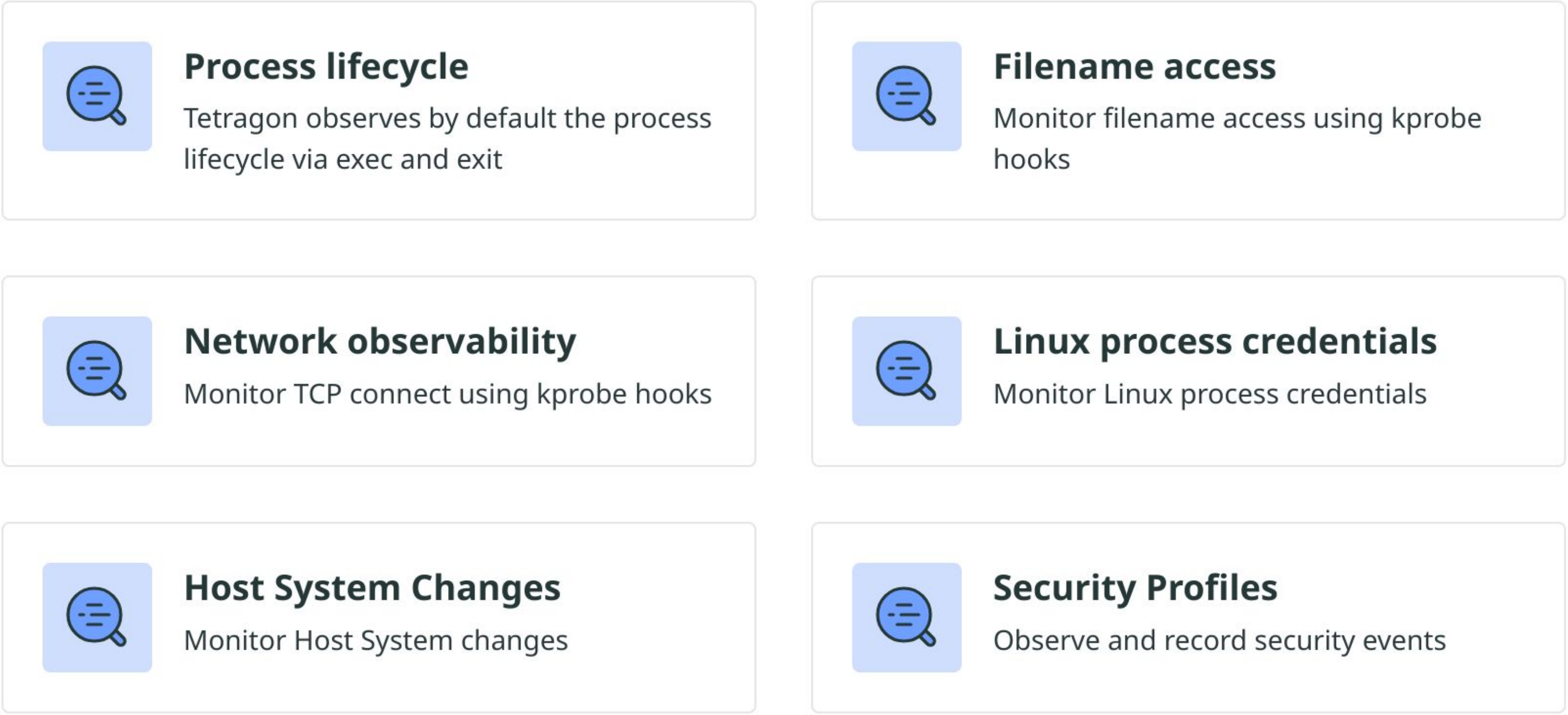
Demo #2

```
SEC("lsm/path_chmod")
int BPF_PROG(path_chmod, const struct path *path, umode_t mode)
{
    bpf_printk("Change mode of file name %s\n", path->dentry->d_iname);
    return 0;
}
```

```
output:
  <...>-128614  [004] d..21 453882.241571: bpf_trace_printk: Change mode of file name boop
  <...>-128634  [010] d..21 453883.248693: bpf_trace_printk: Change mode of file name boop
  <...>-128670  [010] d..21 453884.044865: bpf_trace_printk: Change mode of file name boop
```



Tetragon



Process lifecycle

Tetragon observes by default the process lifecycle via exec and exit

Filename access

Monitor filename access using kprobe hooks

Network observability

Monitor TCP connect using kprobe hooks

Linux process credentials

Monitor Linux process credentials

Host System Changes

Monitor Host System changes

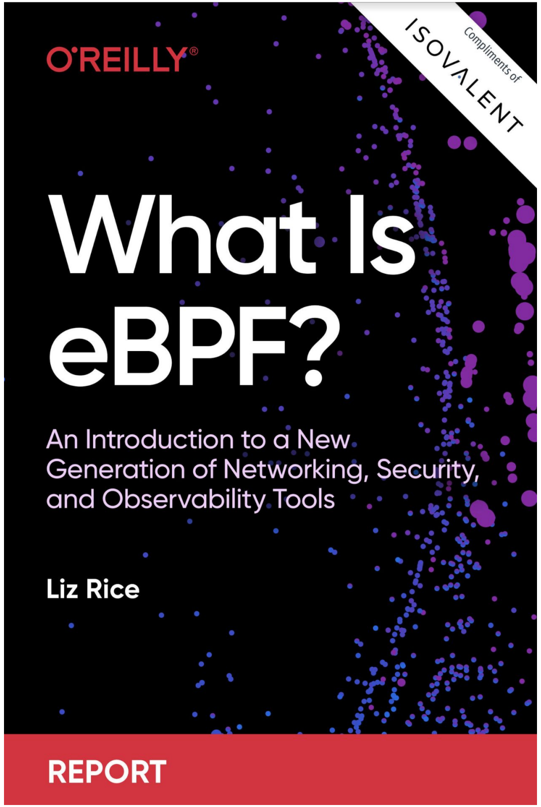
Security Profiles

Observe and record security events

Demo

More stuff!

- ebpf.io
- What is eBPF? - Liz Rice
- libbpf-bootstrap
- libbpf-rs


O'REILLY®
Compliments of
ISOVALENT
What Is eBPF?
An Introduction to a New Generation of Networking, Security, and Observability Tools
Liz Rice
REPORT

Thanks!