Security superpowers with eBPF and Tetragon

CISCO Live!

Liz Rice
Chief Open Source Officer, Isovalent at Cisco

BRKSEC-2167

Cisco Webex App

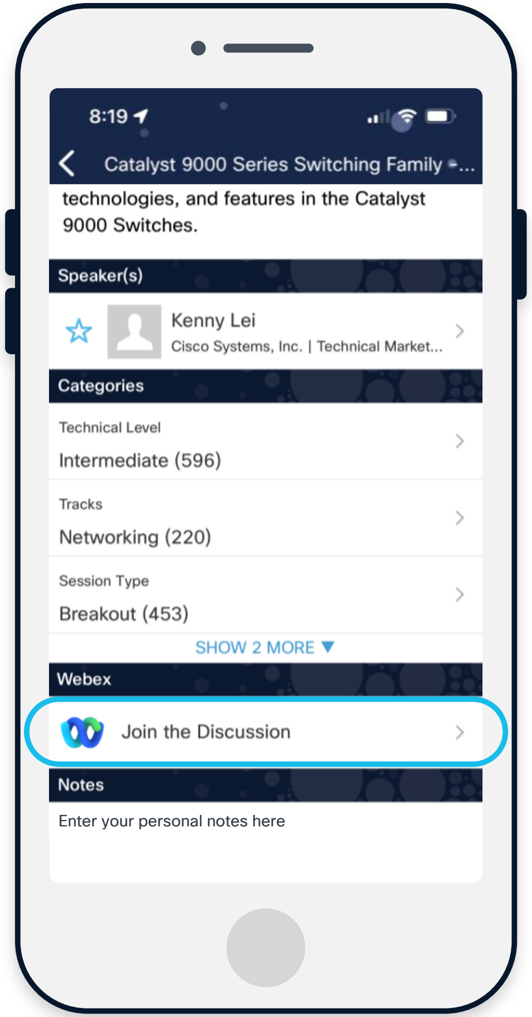
Questions?

Use Cisco Webex App to chat with me after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated until June 13, 2025.



8:19

Catalyst 9000 Series Switching Family ...

technologies, and features in the Catalyst 9000 Switches.

Speaker(s)

Kenny Lei
Cisco Systems, Inc. | Technical Market...

Categories

Technical Level
Intermediate (596)

Tracks
Networking (220)

Session Type
Breakout (453)

SHOW 2 MORE ▼

Webex

Join the Discussion

Notes

Enter your personal notes here

https://ciscolive.ciscoevents.com/
ciscolivebot/#**BRKSEC-2167**

Do not edit
How to change the design

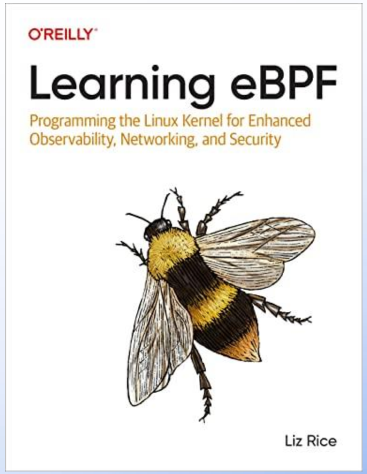
Join at slido.com
#BRKSEC-2167

The Slido app must be installed on every computer you're presenting from

slido

Hello, I’m Liz 👋

- Open source and community at Isovalent, now part of Cisco!
- Author Learning eBPF & Container Security
- Formerly CNCF Governing Board and chair of Technical Oversight Committee
- Early career writing network protocol code

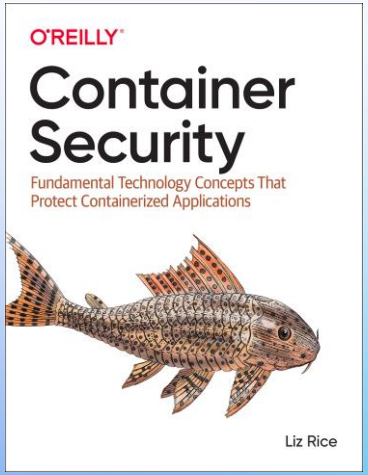


O'REILLY

Learning eBPF

Programming the Linux Kernel for Enhanced Observability, Networking, and Security

Liz Rice

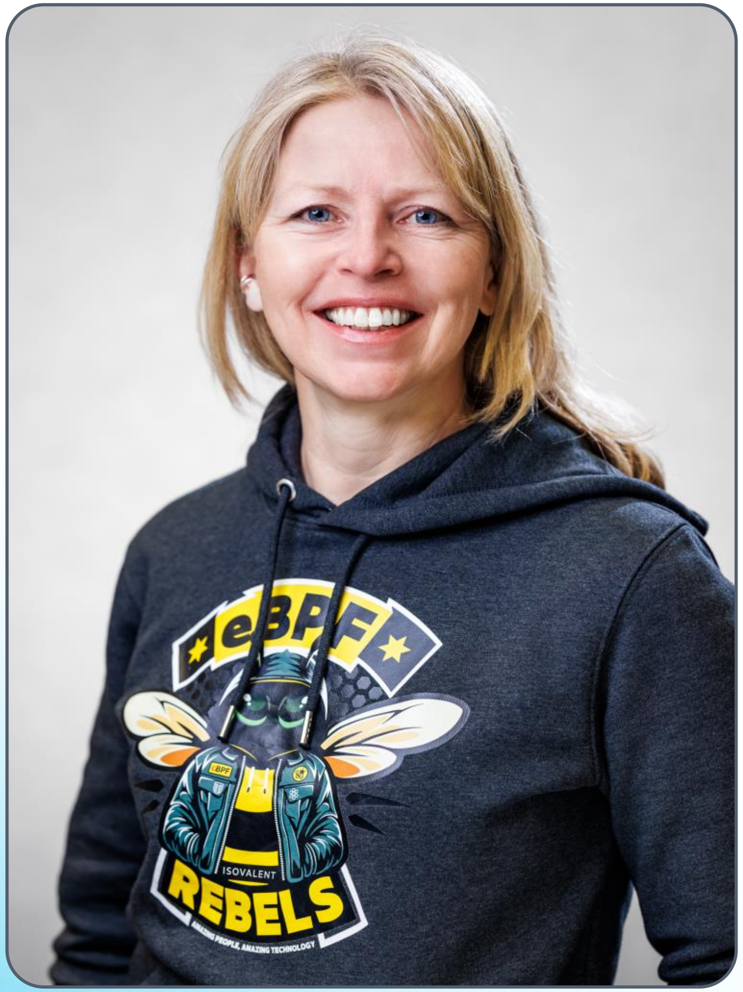


O'REILLY

Container Security

Fundamental Technology Concepts That Protect Containerized Applications

Liz Rice



How familiar are you with eBPF?

Do not edit
How to change the design

Audience Q&A

The Slido app must be installed on every computer you're presenting from

slido

Agenda

01 What is eBPF?

02 eBPF security observability

03 Tetragon

04 Dive into policies

05 Tetragon enforcement

06 Example use cases

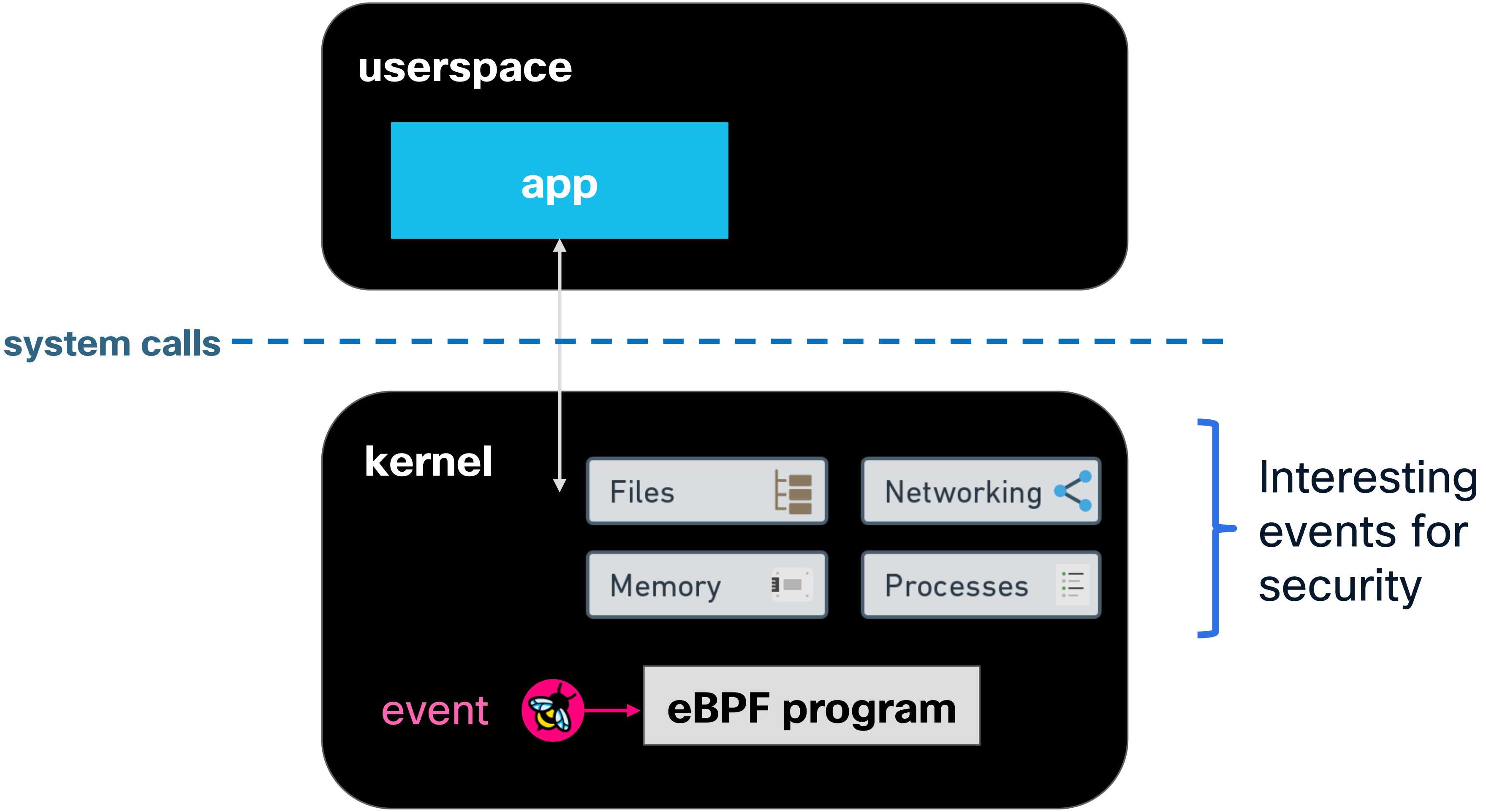
Tetragon provides eBPF abstractions for security so that you don’t need to learn eBPF!

What is eBPF?

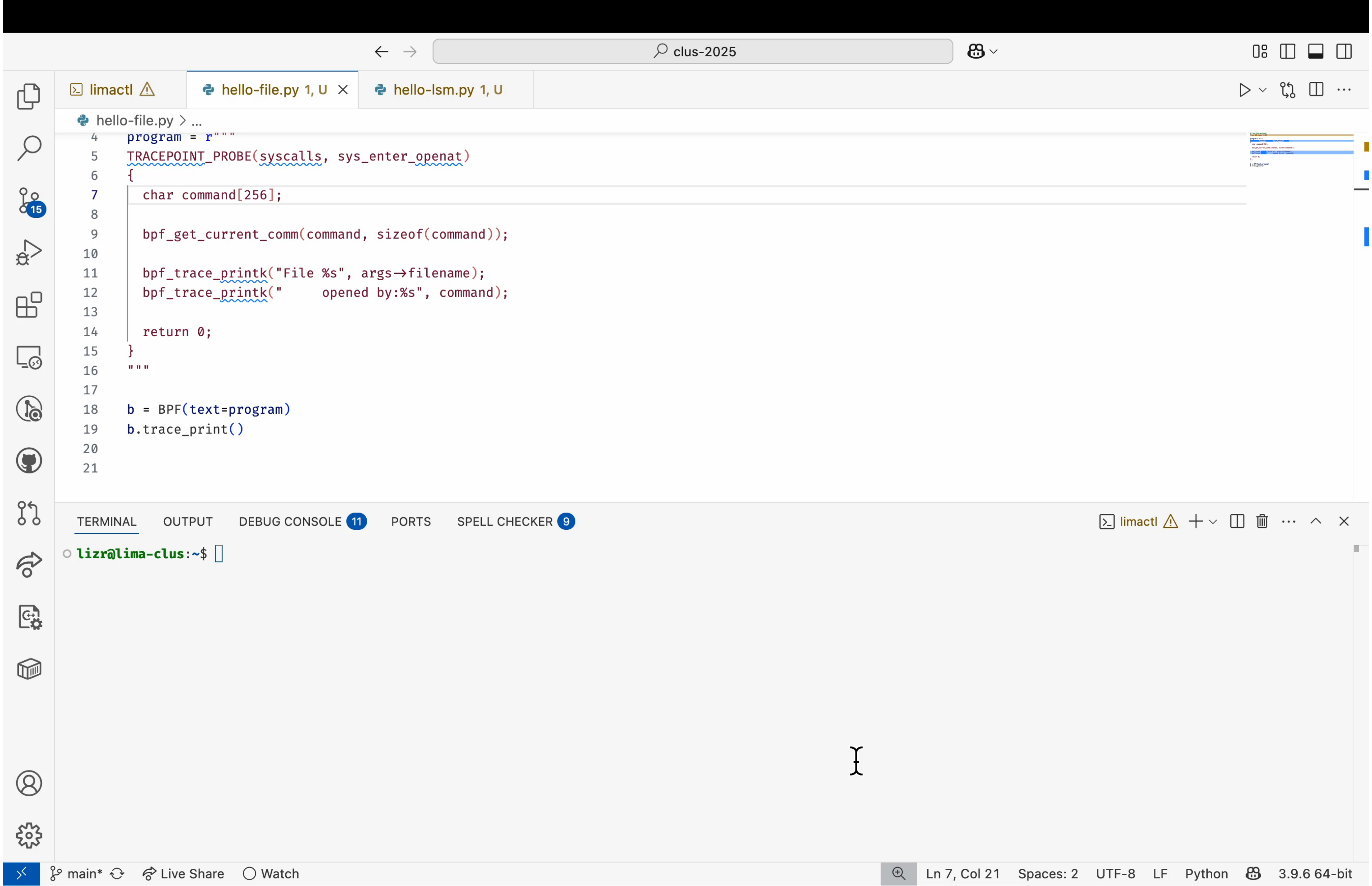
What is eBPF?

- Makes the kernel programmable
- Allows bespoke, dynamic changes to kernel behavior
- Enables high performance, low overhead infrastructure tools

Run custom code in the kernel



Demo – detect file access



```python
program = r"""
TRACEPOINT_PROBE(syscalls, sys_enter_openat)
{
  char command[256];

  bpf_get_current_comm(command, sizeof(command));

  bpf_trace_printk("File %s", args->filename);
  bpf_trace_printk("     opened by:%s", command);

  return 0;
}
"""

b = BPF(text=program)
b.trace_print()
```

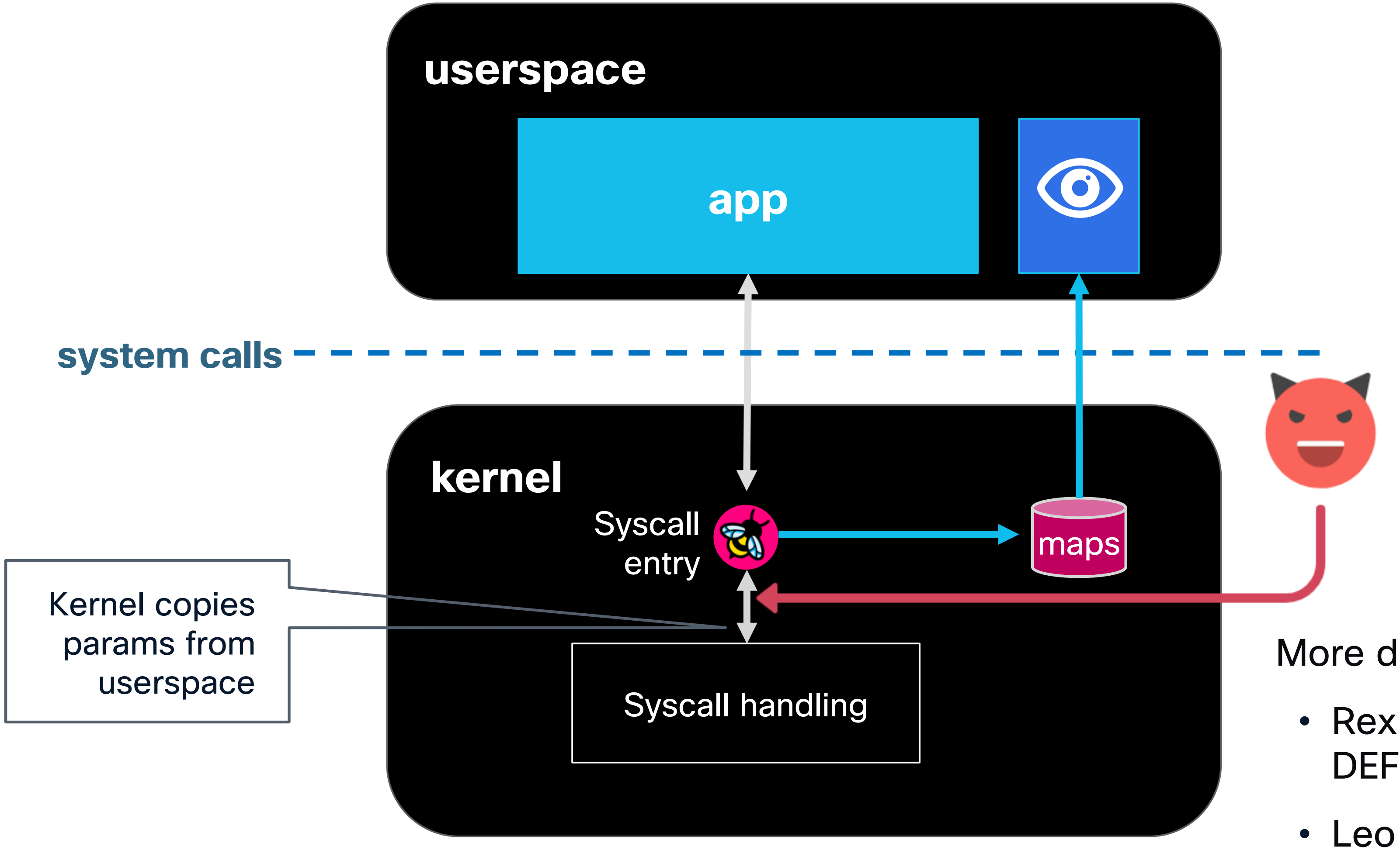
```
lizr@lima-clus:~$ 
```

Demo – detect file access with syscall openat

```
TRACEPOINT_PROBE(syscalls, sys_enter_openat)
{
  char command[256];
  bpf_get_current_comm(command, sizeof(command));

  bpf_trace_printk("File %s", args->filename);
  bpf_trace_printk("     opened by %s", command);

  return 0;
}
```

```
...
cat-509761  [001] ....1 695983.115616: bpf_trace_printk: File out.txt'
cat-509761  [001] ....1 695983.115617: bpf_trace_printk:      opened by cat’
...
```

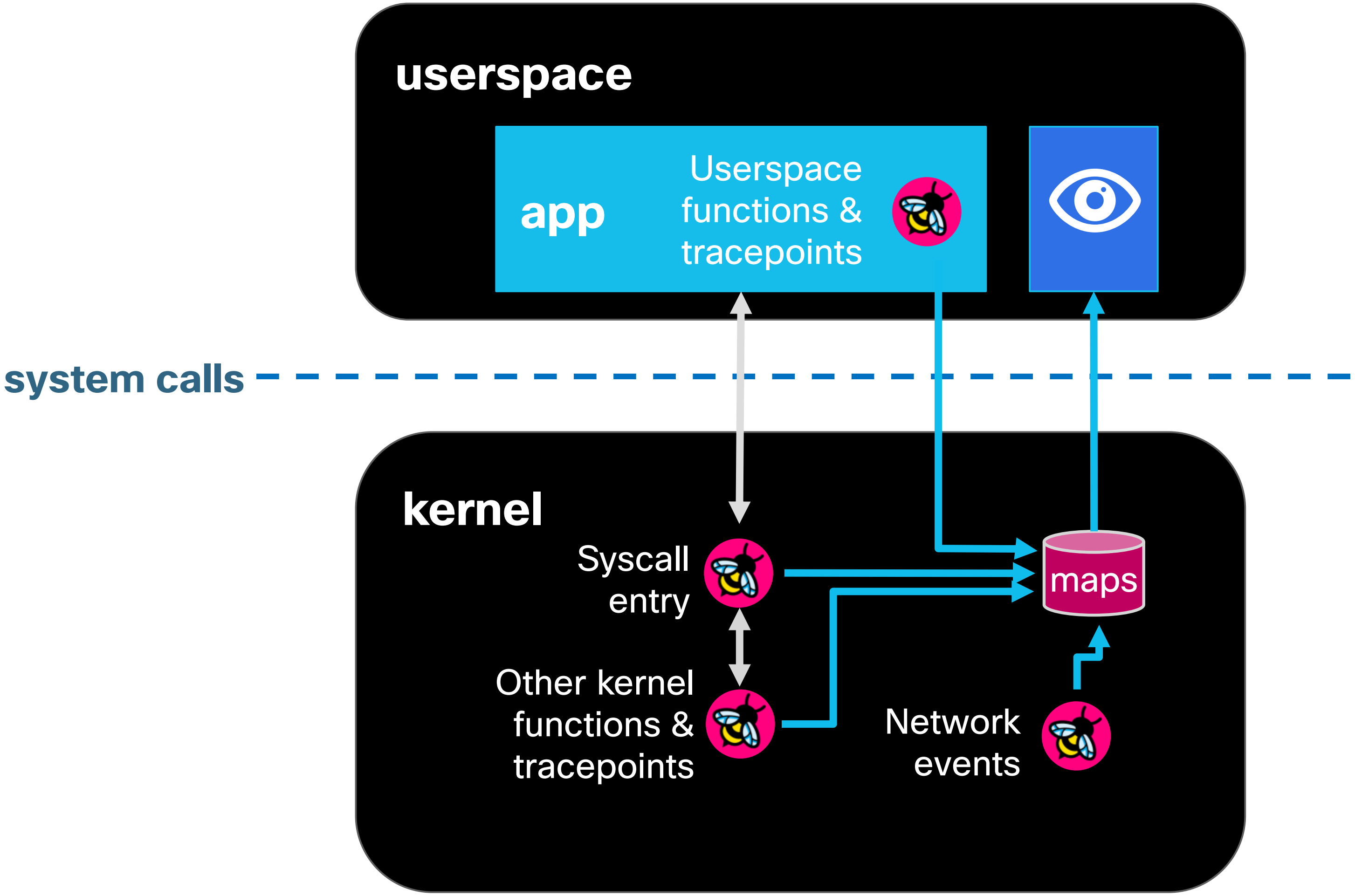
Syscall TOCTOU vulnerabilities



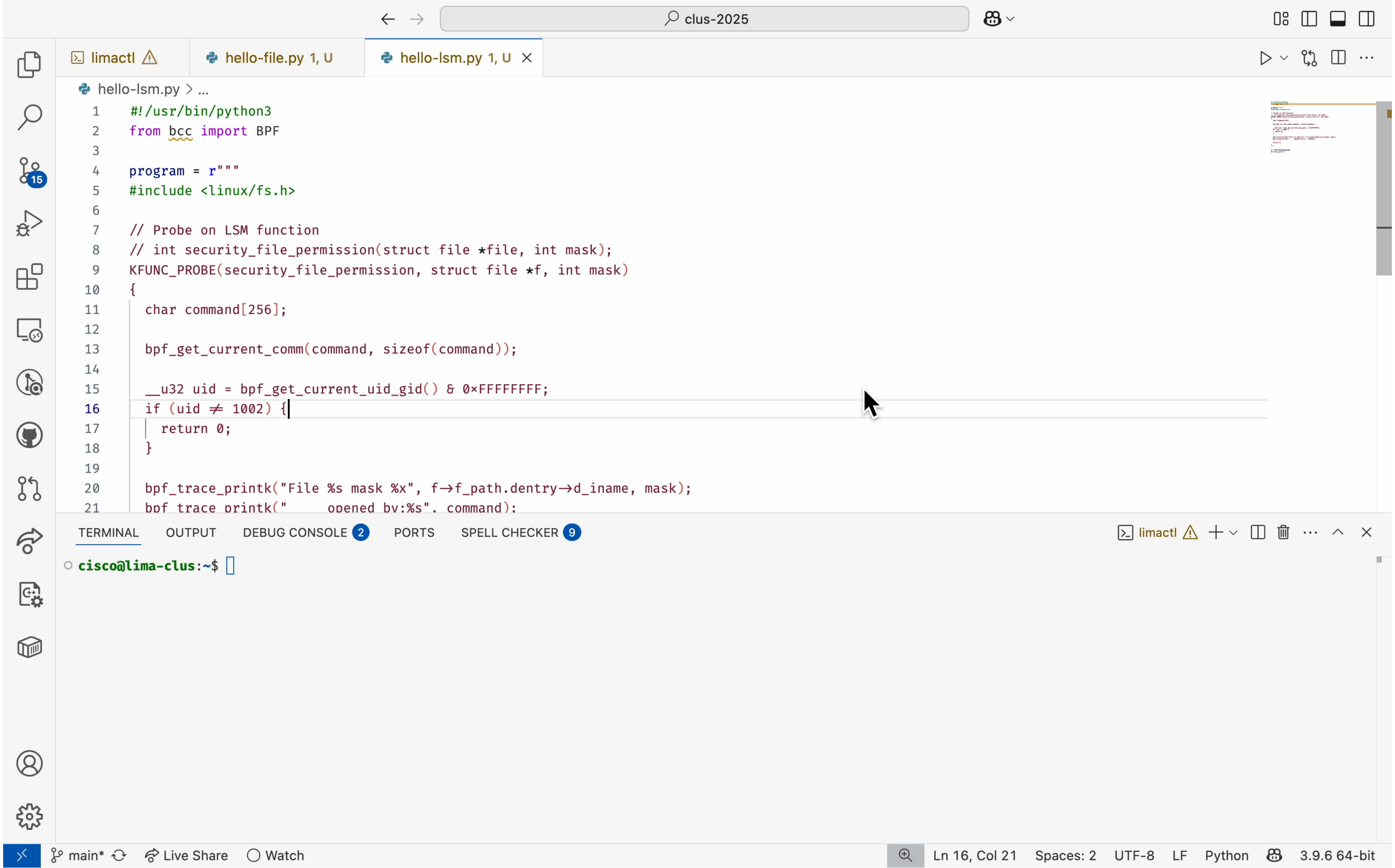
More details:

- Rex Guo & Junyuan Zeng at DEFCON 29 on Phantom attacks
- Leo Di Donato & KP Singh at CN eBPF Day 2021

eBPF attachments aren't just for syscalls


userspace
app
Userspace functions & tracepoints
system calls
kernel
Syscall entry
Other kernel functions & tracepoints
maps
Network events

Demo – detect file access with kernel security function



```
#!/usr/bin/python3
from bcc import BPF

program = r"""
#include <linux/fs.h>

// Probe on LSM function
// int security_file_permission(struct file *file, int mask);
KFUNC_PROBE(security_file_permission, struct file *f, int mask)
{
  char command[256];

  bpf_get_current_comm(command, sizeof(command));

  __u32 uid = bpf_get_current_uid_gid() & 0xFFFFFFFF;
  if (uid != 1002) {
    return 0;
  }

  bpf_trace_printk("File %s mask %x", f->f_path.dentry->d_iname, mask);
  bpf_trace_printk("     opened by:%s", command);
```

```
cisco@lima-clus:~$
```

Demo – detect file access with kernel security function

```
KFUNC_PROBE(security_file_permission, struct file *f, int mask)
{
  char command[256];
  bpf_get_current_comm(command, sizeof(command));

  bpf_trace_printk("File %s mask %x", f->f_path.dentry->d_iname, mask);
  bpf_trace_printk("     opened by %s", command);

  return 0;
}
```

```
...
vi-511361  [000] ....1 701931.021891: bpf_trace_printk: File .out.txt.swp mask 2
vi-511361  [000] ....1 701931.021891: bpf_trace_printk:      opened by: vi
vi-511361  [000] ....1 701931.022173: bpf_trace_printk: File out.txt mask 4
vi-511361  [000] ....1 701931.022173: bpf_trace_printk:      opened by: vi
...
```

High performance eBPF runtime security with Tetragon

> Tetragon provides our security teams with rich data ...
> to answer questions about activity ...
> It was quick to set up and has minimal overhead, which is critical at our scale.

Jason Cetina - Staff Security Engineer at GitHub

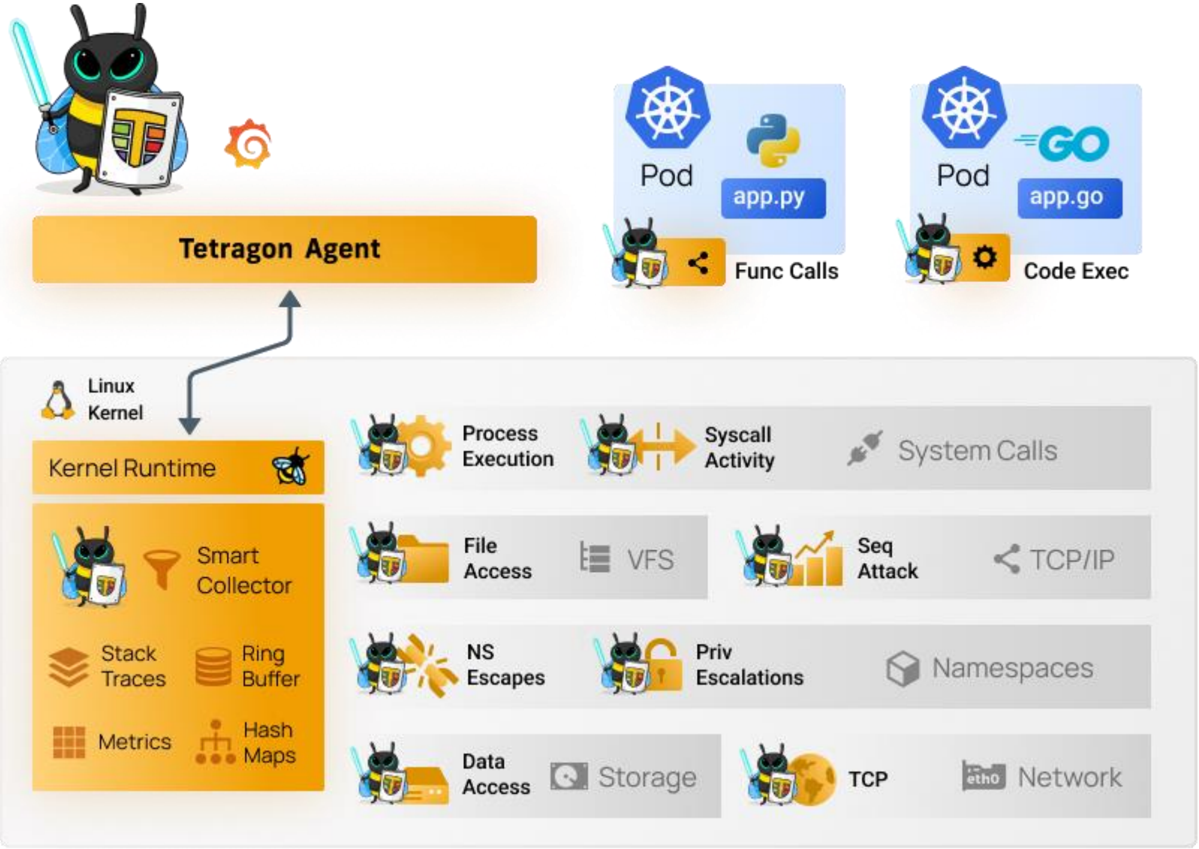
tetragon – high performance eBPF runtime security

Security Observability:
Rich event data: process execution, network communication, file access, etc.

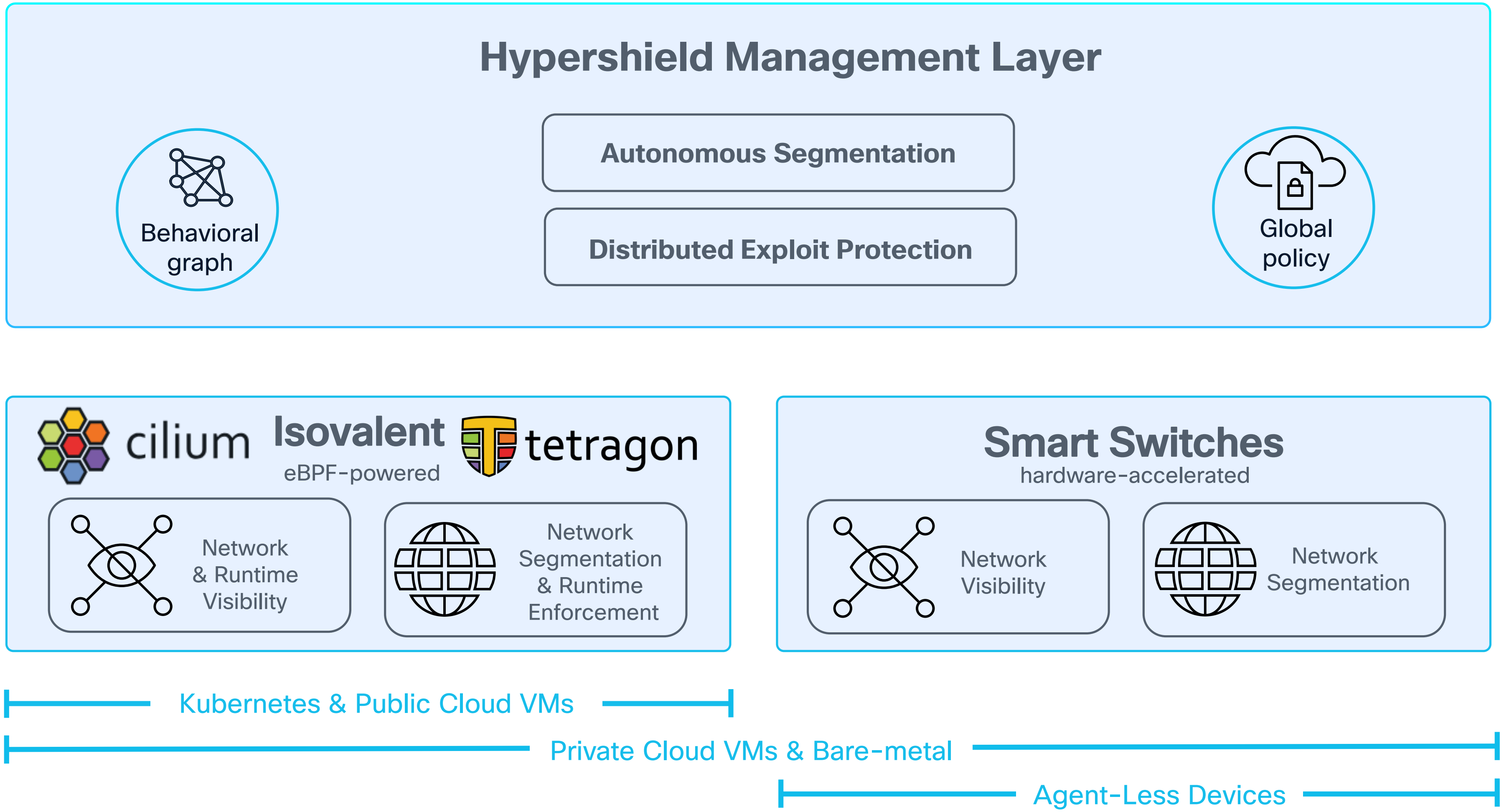
Runtime Enforcement:
Block malicious activities in-kernel

Cloud Native Awareness:
Correlate events to container and Kubernetes identities

Open Source github.com/cilium/tetragon



Core OSS component of Isovalent Enterprise & Hypershield



Hypershield Management Layer

Behavioral graph

Autonomous Segmentation

Distributed Exploit Protection

Global policy

cilium

Isovalent

eBPF-powered

tetragon

Network & Runtime Visibility

Network Segmentation & Runtime Enforcement

Smart Switches

hardware-accelerated

Network Visibility

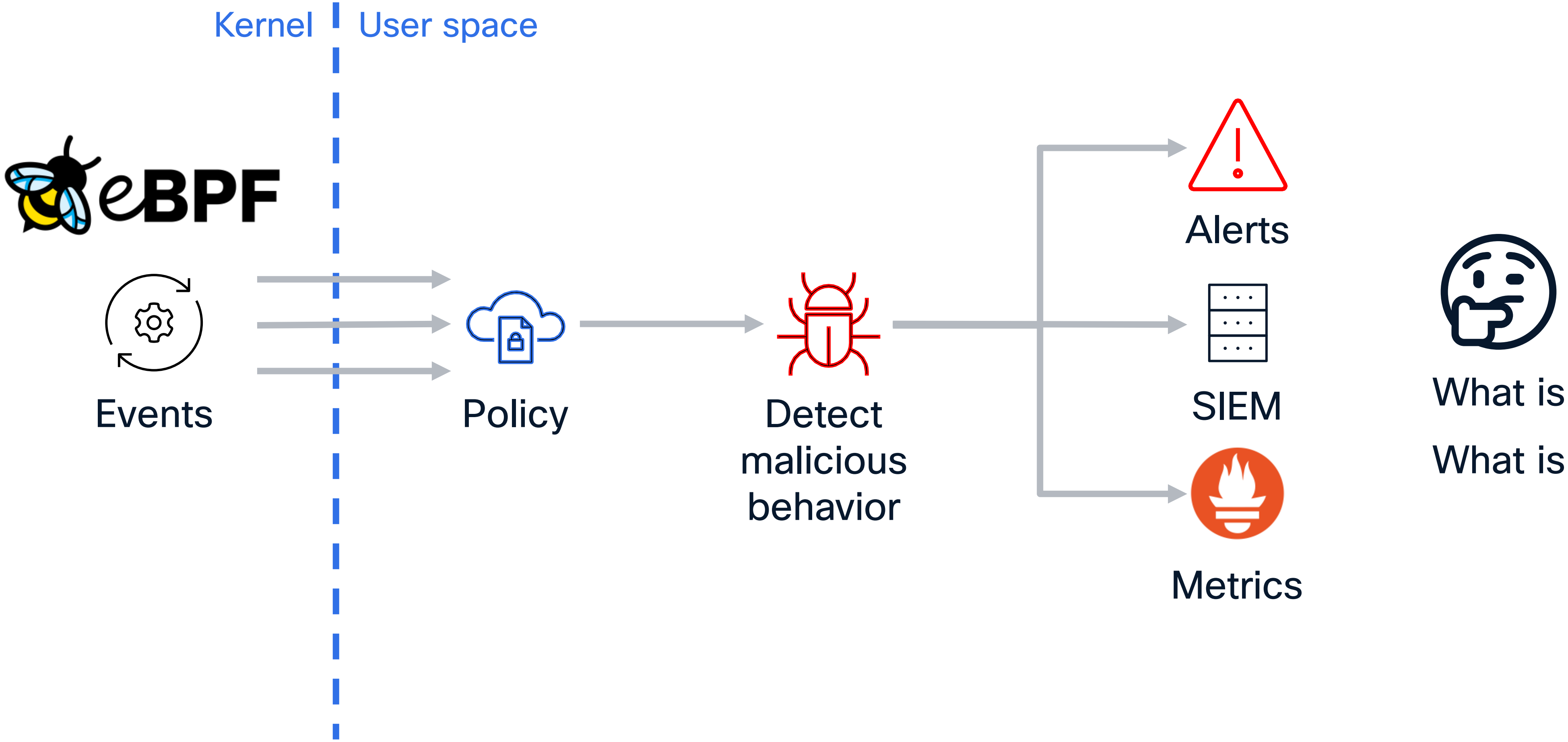
Network Segmentation

Kubernetes & Public Cloud VMs

Private Cloud VMs & Bare-metal

Agent-Less Devices

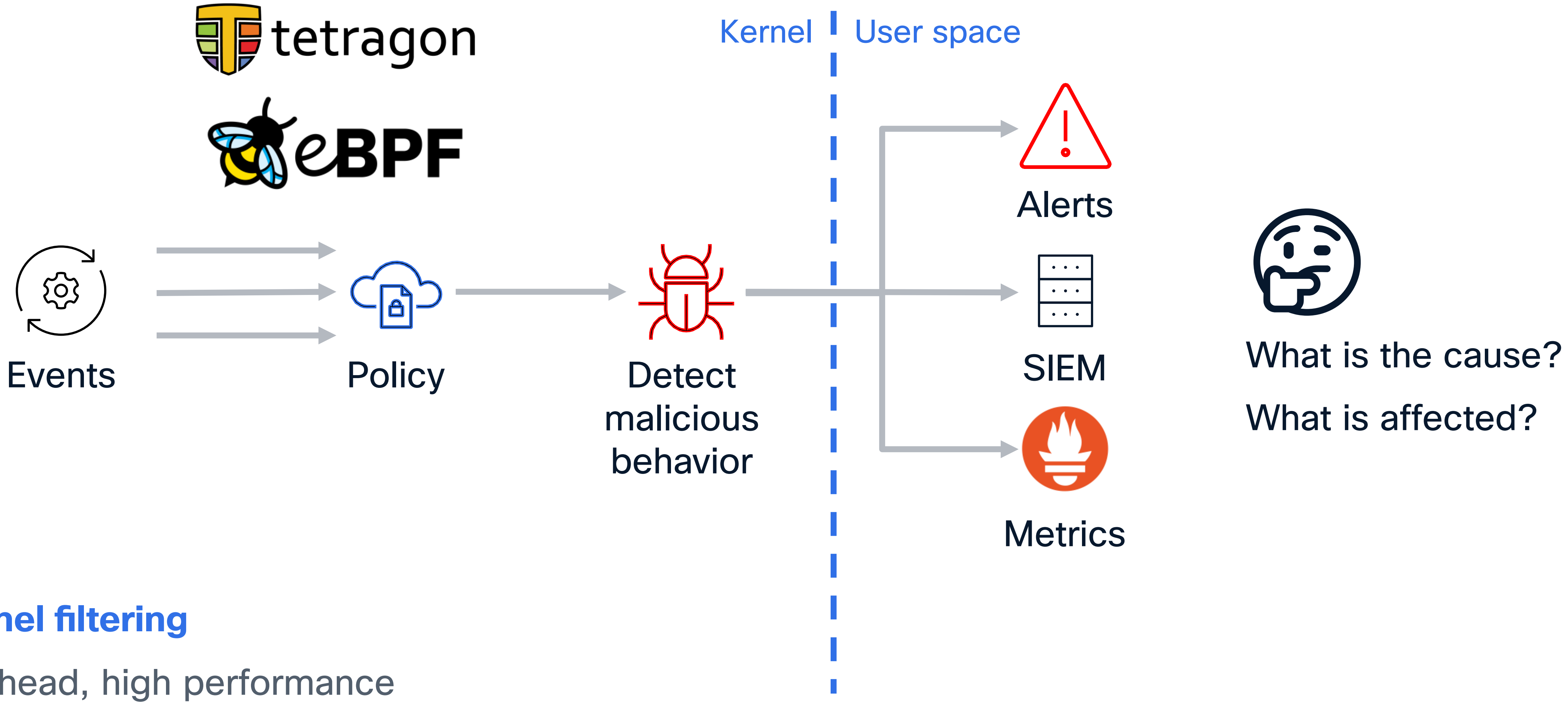
Security observability with eBPF – other apps



What is the cause?

What is affected?

Security observability with eBPF and in-kernel filtering



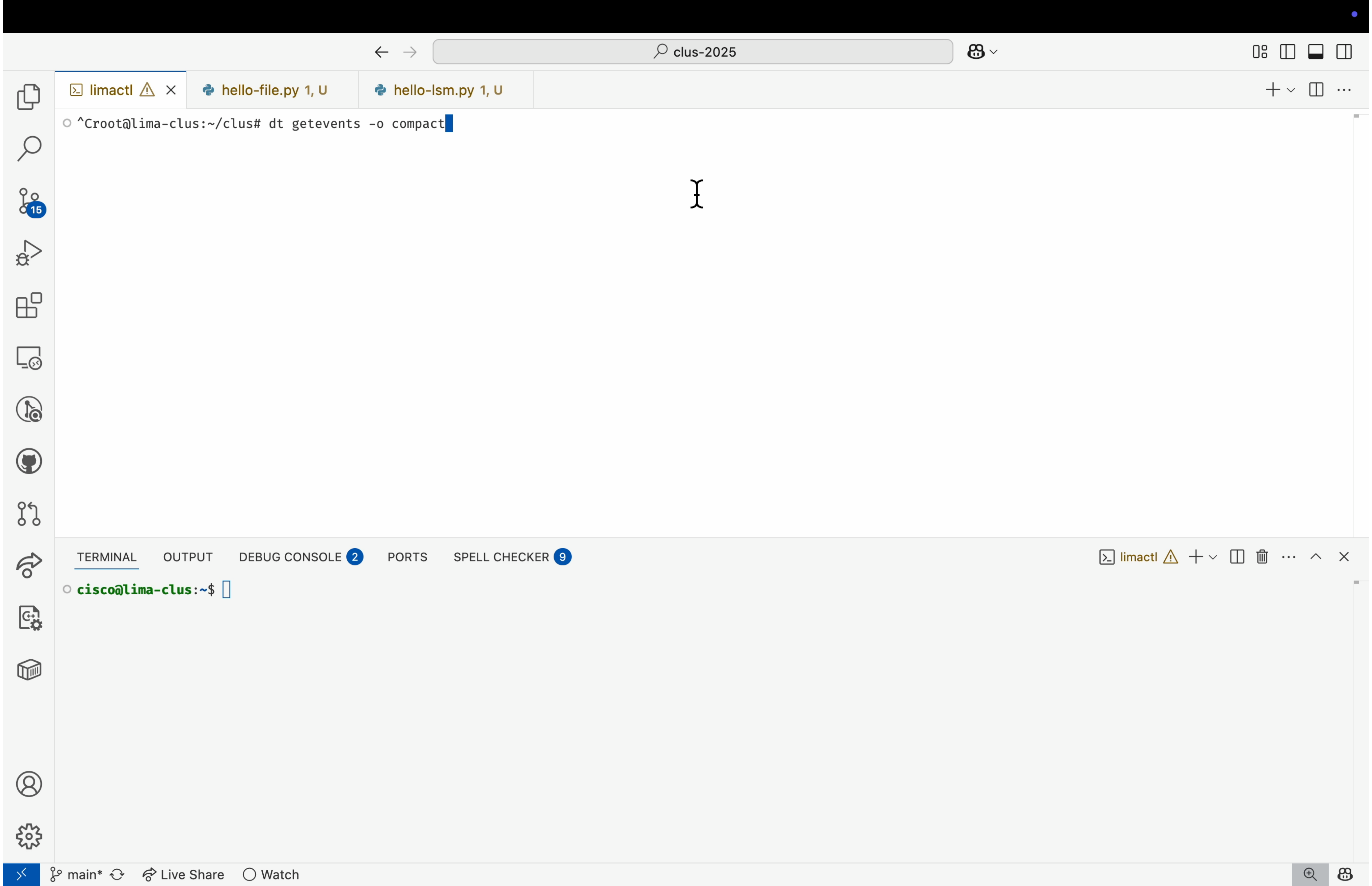
What is the cause?

What is affected?

In-kernel filtering

Low overhead, high performance observability

Demo – Tetragon



```
^Croot@lima-clus:~/clus# dt getevents -o compact
```

```
cisco@lima-clus:~$
```

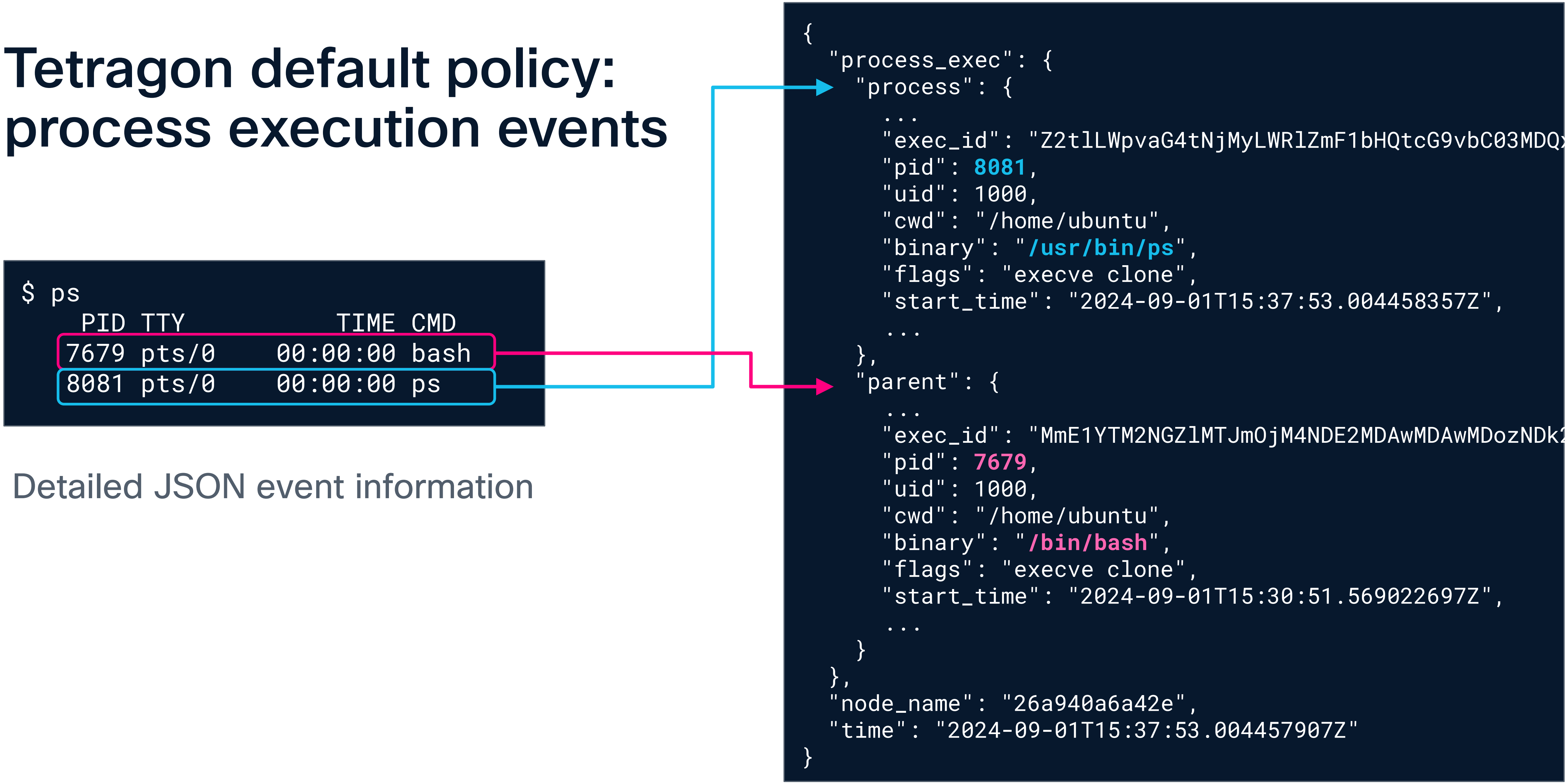
Tetragon default policy: process execution events

```
$ ps
    PID TTY          TIME CMD
   7679 pts/0    00:00:00 bash
   8081 pts/0    00:00:00 ps
```

Compact (human-readable) output

```
🚀 process 719a648a9d54 /usr/bin/ps
💥 exit    719a648a9d54 /usr/bin/ps  0
```

Tetragon default policy: process execution events

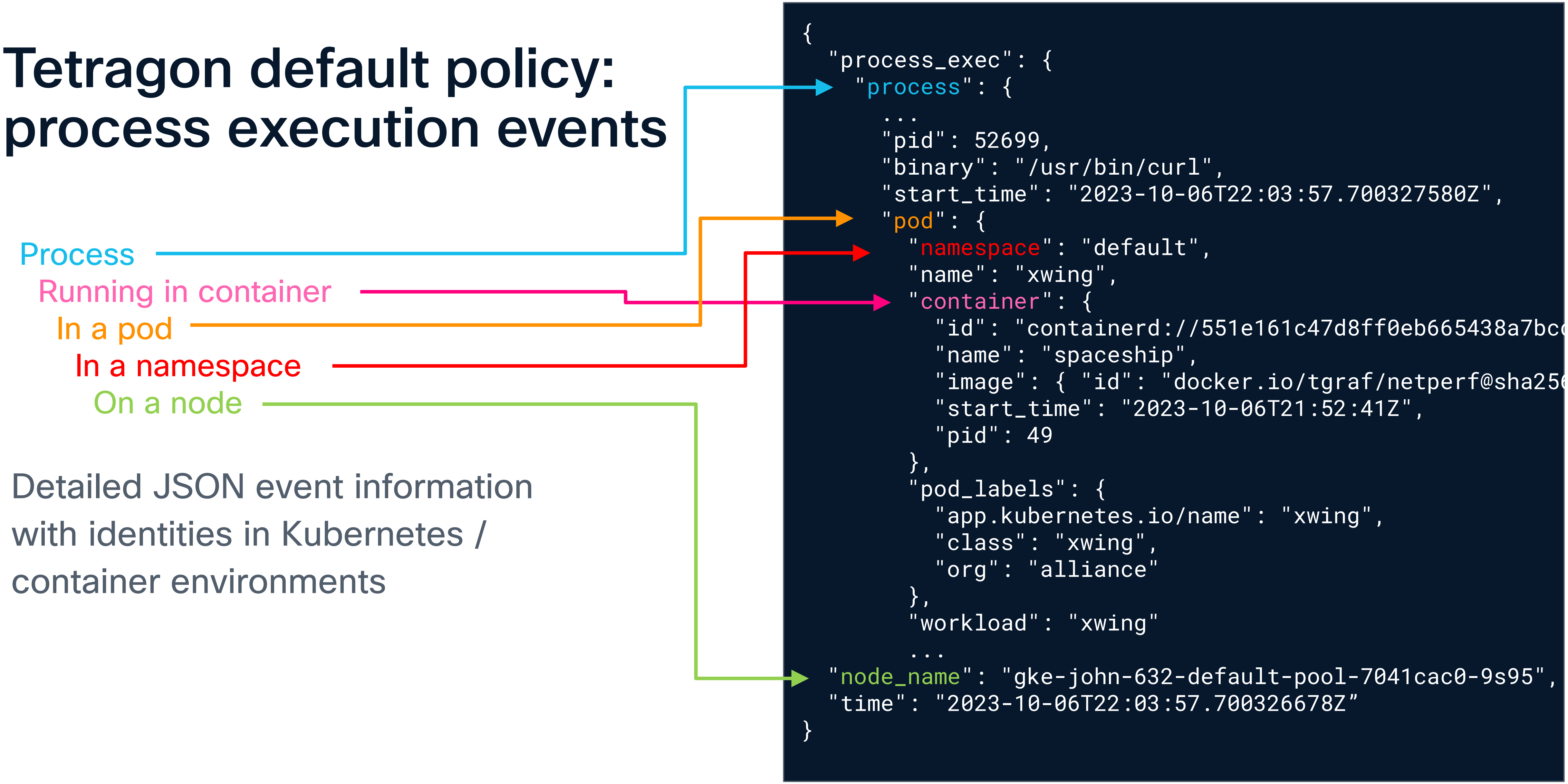


```
$ ps
    PID TTY          TIME CMD
   7679 pts/0    00:00:00 bash
   8081 pts/0    00:00:00 ps
```

```
{
  "process_exec": {
    "process": {
      ...
      "exec_id": "Z2tlLWpvaG4tNjMyLWRlZmF1bHQtcG9vbC03MDQ
      "pid": 8081,
      "uid": 1000,
      "cwd": "/home/ubuntu",
      "binary": "/usr/bin/ps",
      "flags": "execve clone",
      "start_time": "2024-09-01T15:37:53.004458357Z",
      ...
    },
    "parent": {
      ...
      "exec_id": "MmE1YTM2NGZlMTJmOjM4NDE2MDAwMDAwMDozNDk
      "pid": 7679,
      "uid": 1000,
      "cwd": "/home/ubuntu",
      "binary": "/bin/bash",
      "flags": "execve clone",
      "start_time": "2024-09-01T15:30:51.569022697Z",
      ...
    }
  },
  "node_name": "26a940a6a42e",
  "time": "2024-09-01T15:37:53.004457907Z"
}
```

Detailed JSON event information

Tetragon default policy: process execution events



Process

Running in container

In a pod

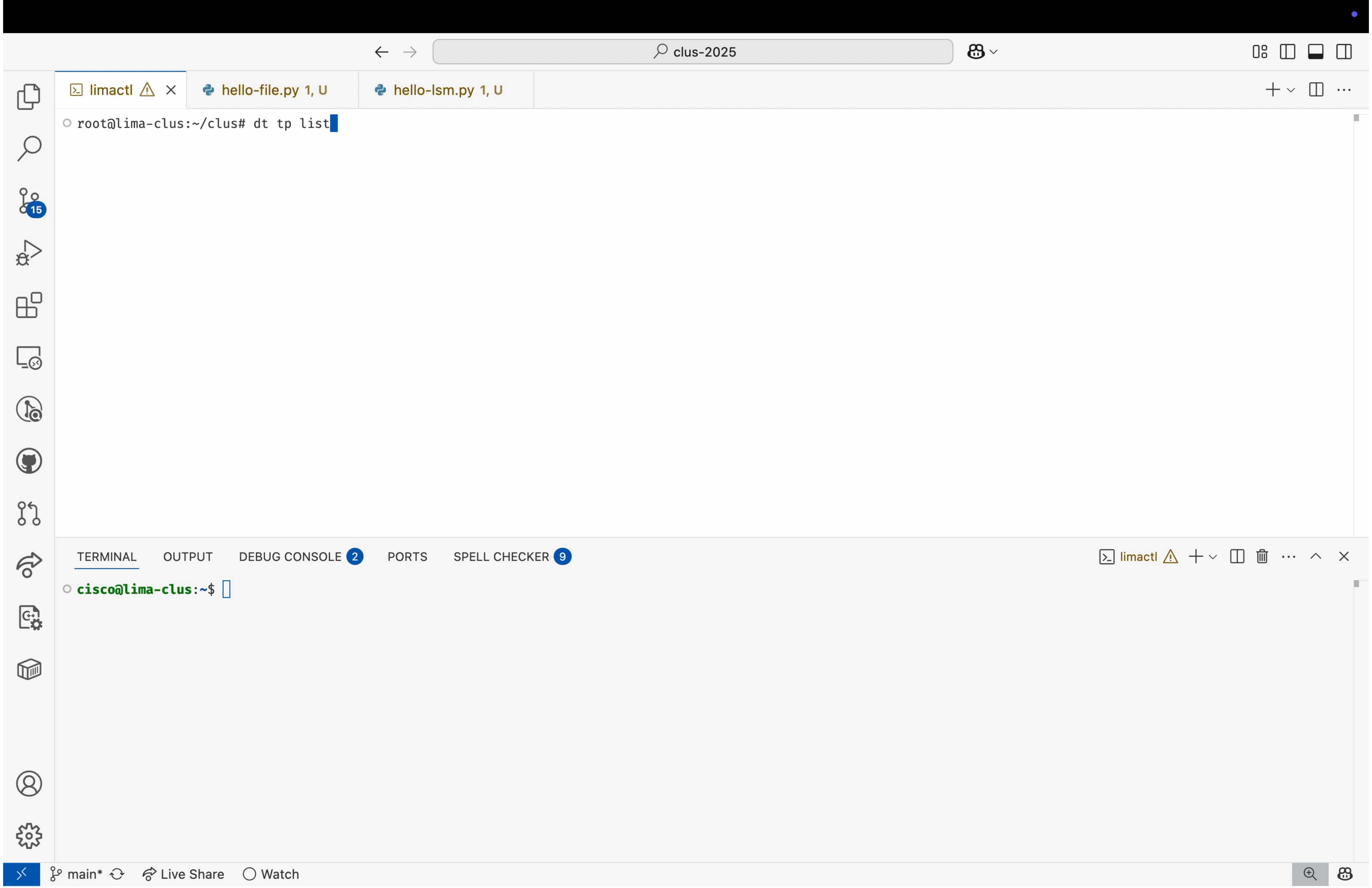
In a namespace

On a node

```
{
  "process_exec": {
    "process": {
      ...
      "pid": 52699,
      "binary": "/usr/bin/curl",
      "start_time": "2023-10-06T22:03:57.700327580Z",
      "pod": {
        "namespace": "default",
        "name": "xwing",
        "container": {
          "id": "containerd://551e161c47d8ff0eb665438a7bc
          "name": "spaceship",
          "image": { "id": "docker.io/tgraf/netperf@sha25
          "start_time": "2023-10-06T21:52:41Z",
          "pid": 49
        },
        "pod_labels": {
          "app.kubernetes.io/name": "xwing",
          "class": "xwing",
          "org": "alliance"
        },
        "workload": "xwing"
        ...
  "node_name": "gke-john-632-default-pool-7041cac0-9s95",
  "time": "2023-10-06T22:03:57.700326678Z"
}
```

Detailed JSON event information with identities in Kubernetes / container environments

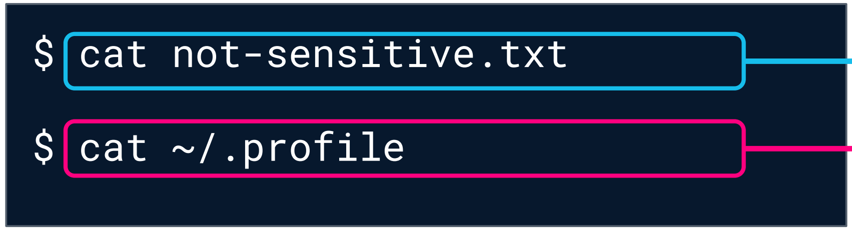
Demo – Tetragon monitoring sensitive files



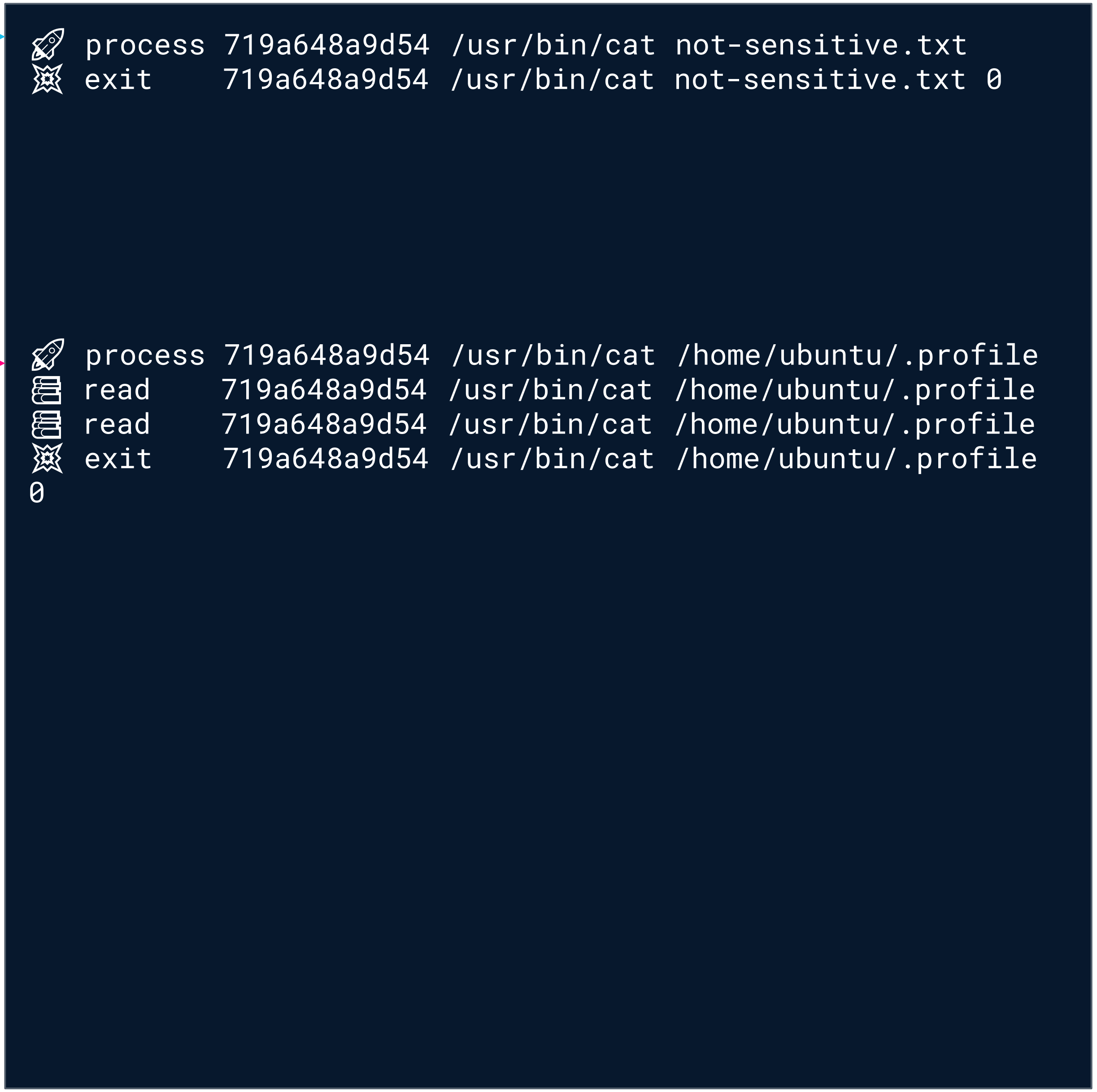
```
root@lima-clus:~/clus# dt tp list
```

```
cisco@lima-clus:~$
```

Tetragon policy: monitor sensitive files



```
$ cat not-sensitive.txt

$ cat ~/.profile
```



```
🚀 process 719a648a9d54 /usr/bin/cat not-sensitive.txt
💥 exit    719a648a9d54 /usr/bin/cat not-sensitive.txt 0
```

```
🚀 process 719a648a9d54 /usr/bin/cat /home/ubuntu/.profile
📚 read    719a648a9d54 /usr/bin/cat /home/ubuntu/.profile
📚 read    719a648a9d54 /usr/bin/cat /home/ubuntu/.profile
💥 exit    719a648a9d54 /usr/bin/cat /home/ubuntu/.profile
0
```

⚡ **In-kernel filtering**:

Events only generated for files specified by policy

Dive into Tetragon policies

Tetragon `TracingPolicy`

Kubernetes custom resource

- You don’t have to be running Kubernetes

Abstraction defining eBPF programs and attachments

- **Hook point** – where to attach eBPF program
- **Selectors** - in-kernel filtering and actions

```yaml
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

Tetragon `TracingPolicy` example

Attach to kprobe for security_file_permission() kernel function

```yaml
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

Tetragon `TracingPolicy` example

Attach to kprobe for security_file_permission() kernel function

```
security_file_permission() - Check file
permissions

@file: file
@mask: requested permissions

Check file permissions before accessing an
open file. This hook is called by various
operations that read or write files.
```

```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

Tetragon TracingPolicy example

Attach to kprobe for security_file_permission() kernel function

```
security_file_permission() - Check file
permissions

@file: file
@mask: requested permissions

Check file permissions before accessing an
open file. This hook is called by various
operations that read or write files.
```

```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

Tetragon `TracingPolicy` example

Attach to kprobe for security_file_permission() kernel function

```
security_file_permission() - Check file
permissions

@file: file
@mask: requested permissions

Check file permissions before accessing an
open file. This hook is called by various
operations that read or write files.
```

```yaml
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

Tetragon TracingPolicy example

```
$ cat /tmp/liz
```

Generates Tetragon event (simplified for readability)

```
“process_kprobe” {
   “process” {
      "cwd": "/home/liz",
      "binary": "/usr/bin/cat",
      "arguments": "/tmp/liz",
   }
   "function_name": "security_file_permission",
    "args": [
      { "file_arg": { "path": "/tmp/liz” }
      { "int_arg": 4 }
    ],
    "action": "KPROBE_ACTION_POST",
    "policy_name": "example",
    "return_action": "KPROBE_ACTION_POST"
  },
...
```

```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

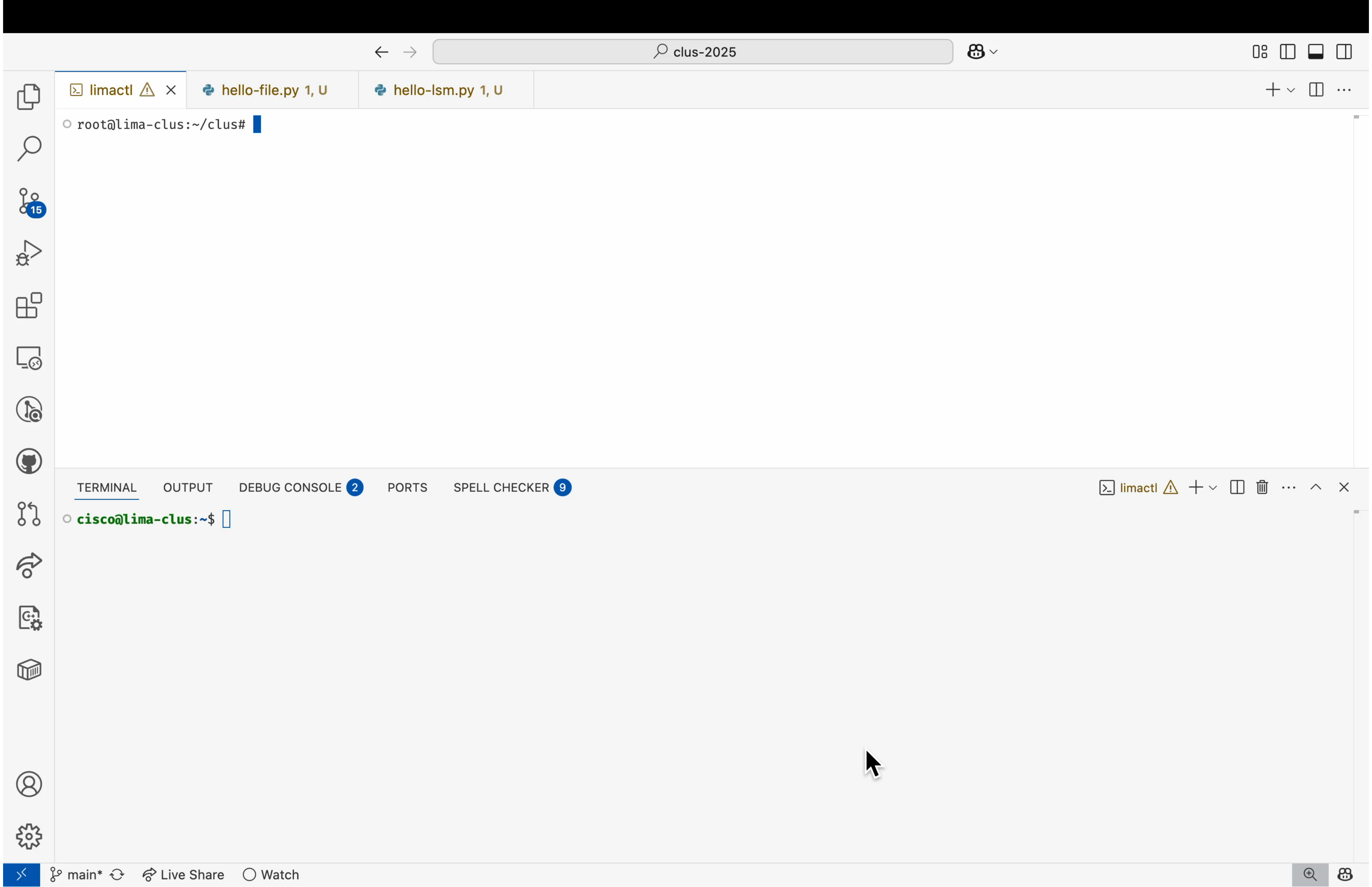
Tetragon TracingPolicy example

```
$ cat /tmp/liz
```

Generates Tetragon event (simplified for readability)

```
“process_kprobe” {
   “process” {
      "cwd": "/home/liz",
      "binary": "/usr/bin/cat",
      "arguments": "/tmp/liz",
   }
   "function_name": "security_file_permission",
    "args": [
      { "file_arg": { "path": "/tmp/liz” }
      { "int_arg": 4 }
    ],
    "action": "KPROBE_ACTION_POST",
    "policy_name": "example",
    "return_action": "KPROBE_ACTION_POST"
  },
...
```

```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```



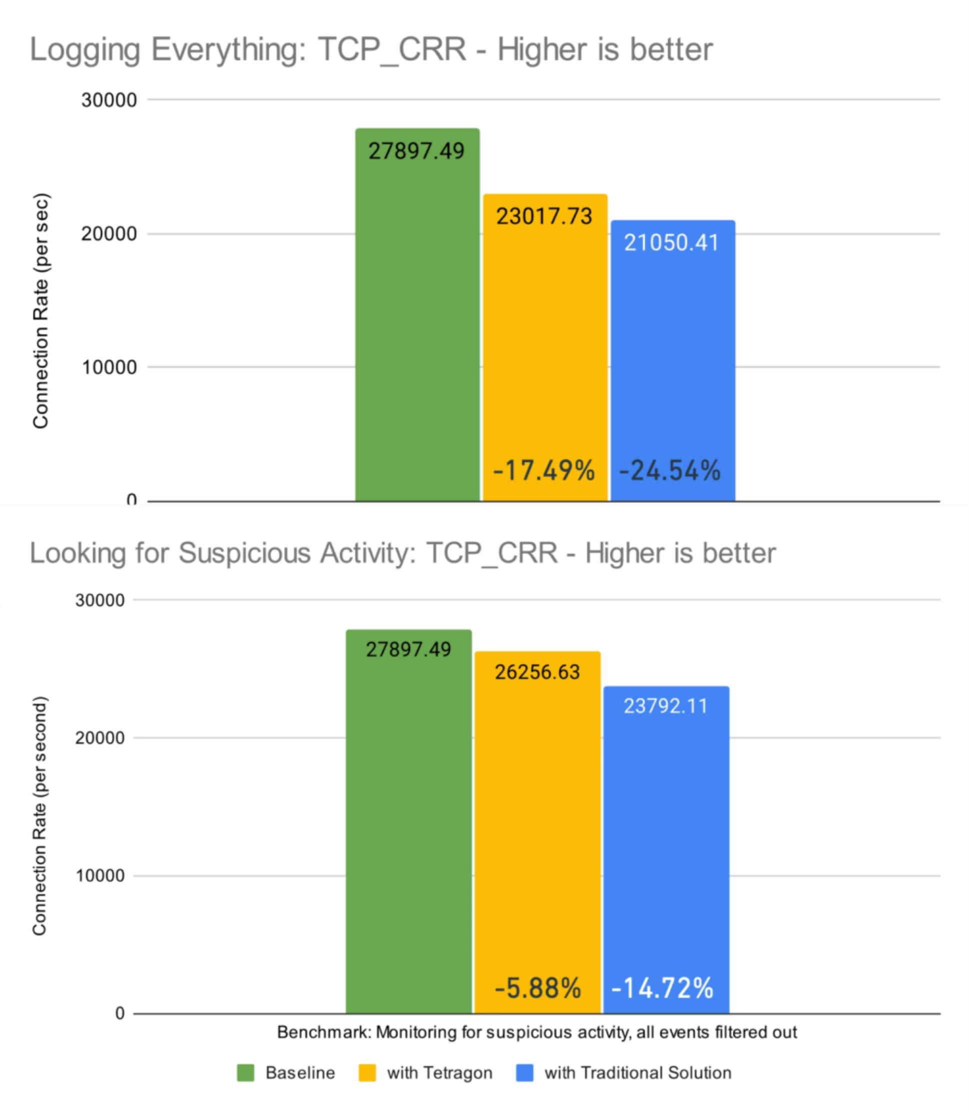
```
root@lima-clus:~/clus#
```

```
cisco@lima-clus:~$
```

Filtering -> better performance

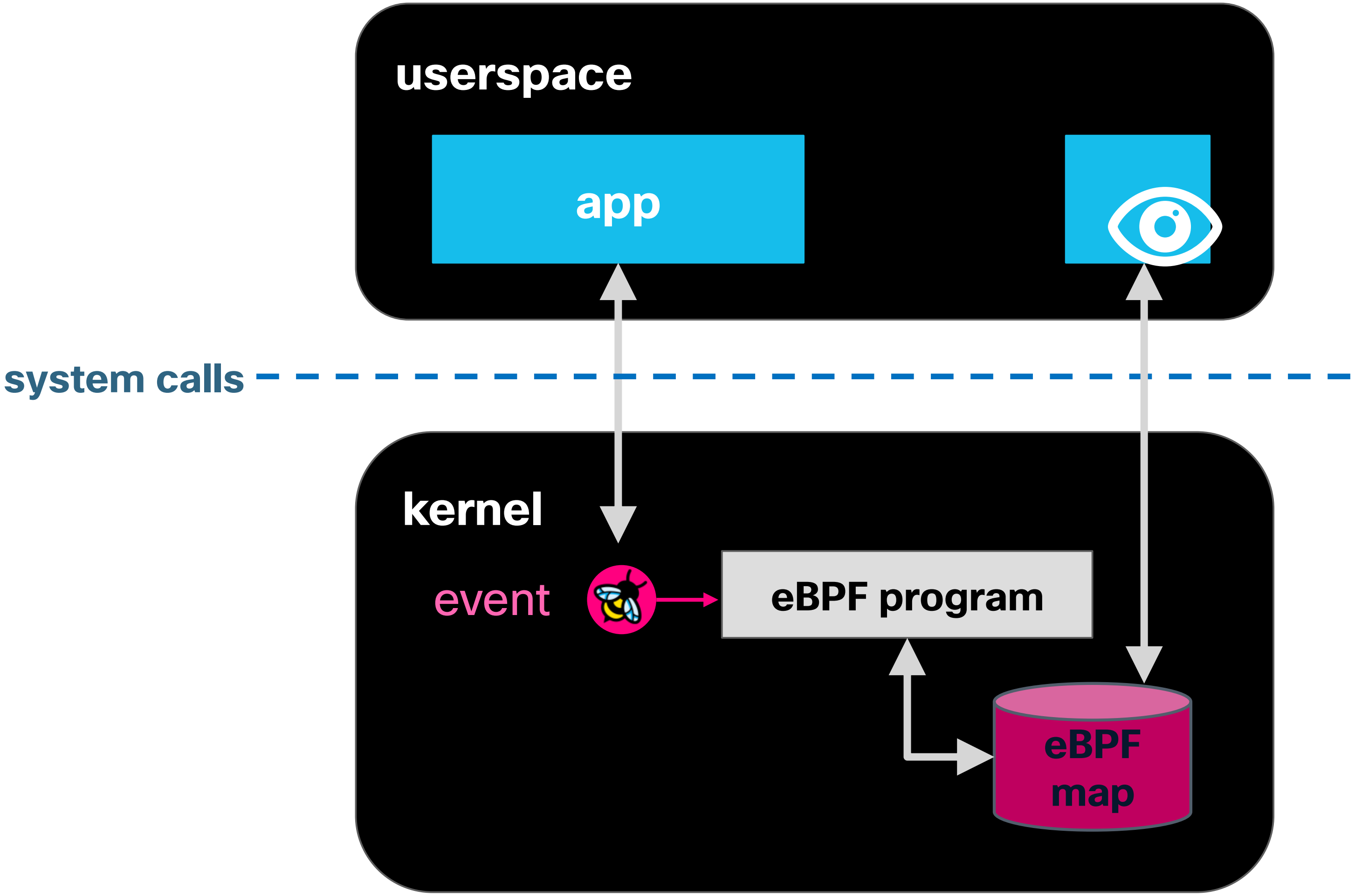
👌 Low overhead:

Near baseline overhead (<2%) in core tasks like process execution tracking



The eBPF behind Tetragon policies

eBPF programs and maps



Adding Tetragon policies creates programs and maps

```
$ bpftool prog list | grep tag | wc -l
18

$ bpftool map list | grep flags | wc -l
29

# Enable one Tetragon policy
$ dt tp enable example

$ bpftool prog list | grep tag | wc -l
25

$ bpftool map list | grep flags | wc -l
72
```

Policy disabled:
- 18 progs
- 29 maps

Policy enabled:
- 25 progs
- 72 maps

```
$ bpftool prog list
110: cgroup_device  tag 3918c82a5f4c0360
...
171: cgroup_skb  name sd_fw_ingress  tag 6deef7357e7b4530  gpl
195: kprobe  name generic_kprobe_setup_event  tag 2dd70e32b285
196: kprobe  name generic_kprobe_process_event  tag 412f816e5e
197: kprobe  name generic_kprobe_filter_arg  tag 4e03413e11408
198: kprobe  name generic_kprobe_actions  tag 1eda0a448d53bc5c
199: kprobe  name generic_kprobe_event  tag 319f1085c07b2002
200: kprobe  name generic_kprobe_process_filter  tag 8dc48498a
201: kprobe  name generic_kprobe_output  tag 75357b43eac559eb

$ bpftool map list
5: hash  name tg_conf_map  flags 0x0
…
139: percpu_array  name execve_heap  flags 0x0
531: lru_hash  name fdinstall_map  flags 0x0
532: array  name config_map  flags 0x0
533: prog_array  name kprobe_calls  flags 0x0
534: array  name filter_map  flags 0x0
536: array_of_maps  name argfilter_maps  flags 0x0
538: array_of_maps  name addr4lpm_maps  flags 0x0
540: array_of_maps  name addr6lpm_maps  flags 0x0
542: array_of_maps  name string_maps_0  flags 0x0
544: array_of_maps  name string_maps_1  flags 0x0
546: array_of_maps  name string_maps_2  flags 0x0
548: array_of_maps  name string_maps_3  flags 0x0
550: array_of_maps  name string_maps_4  flags 0x0
552: array_of_maps  name string_maps_5  flags 0x0
554: array_of_maps  name string_maps_6  flags 0x0
556: array_of_maps  name string_maps_7  flags 0x0
```

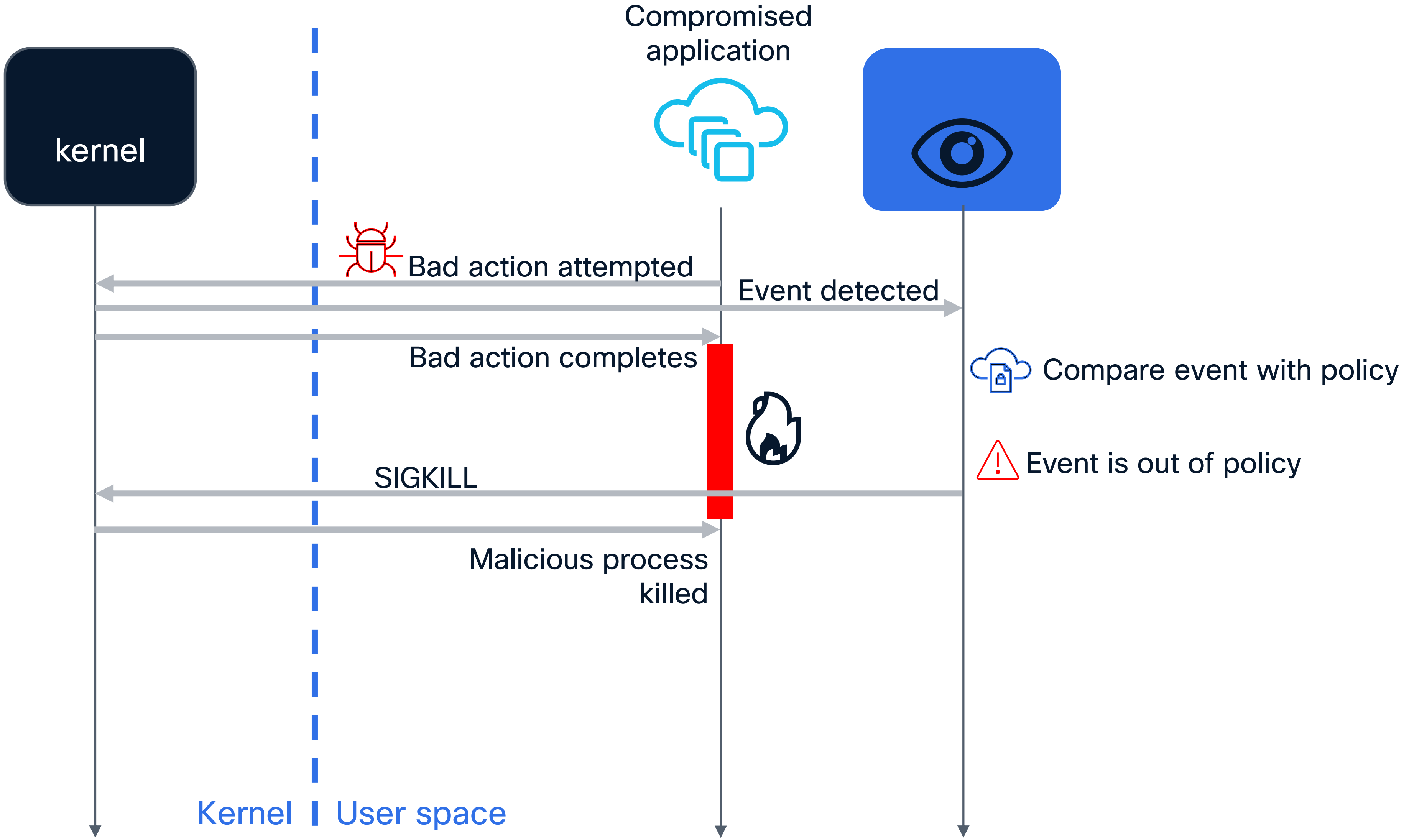
Tetragon TracingPolicy example

```
$ bpftool map list
...
429: hash  name string_maps_0_0  flags 0x0
        key 25B  value 1B  max_entries 1


$ bpftool map dump id 429
key:
08 2f 74 6d 70 2f 6c 69  7a 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00  00
value:
01
Found 1 element

# Hex to ASCII: /tmp/liz
```

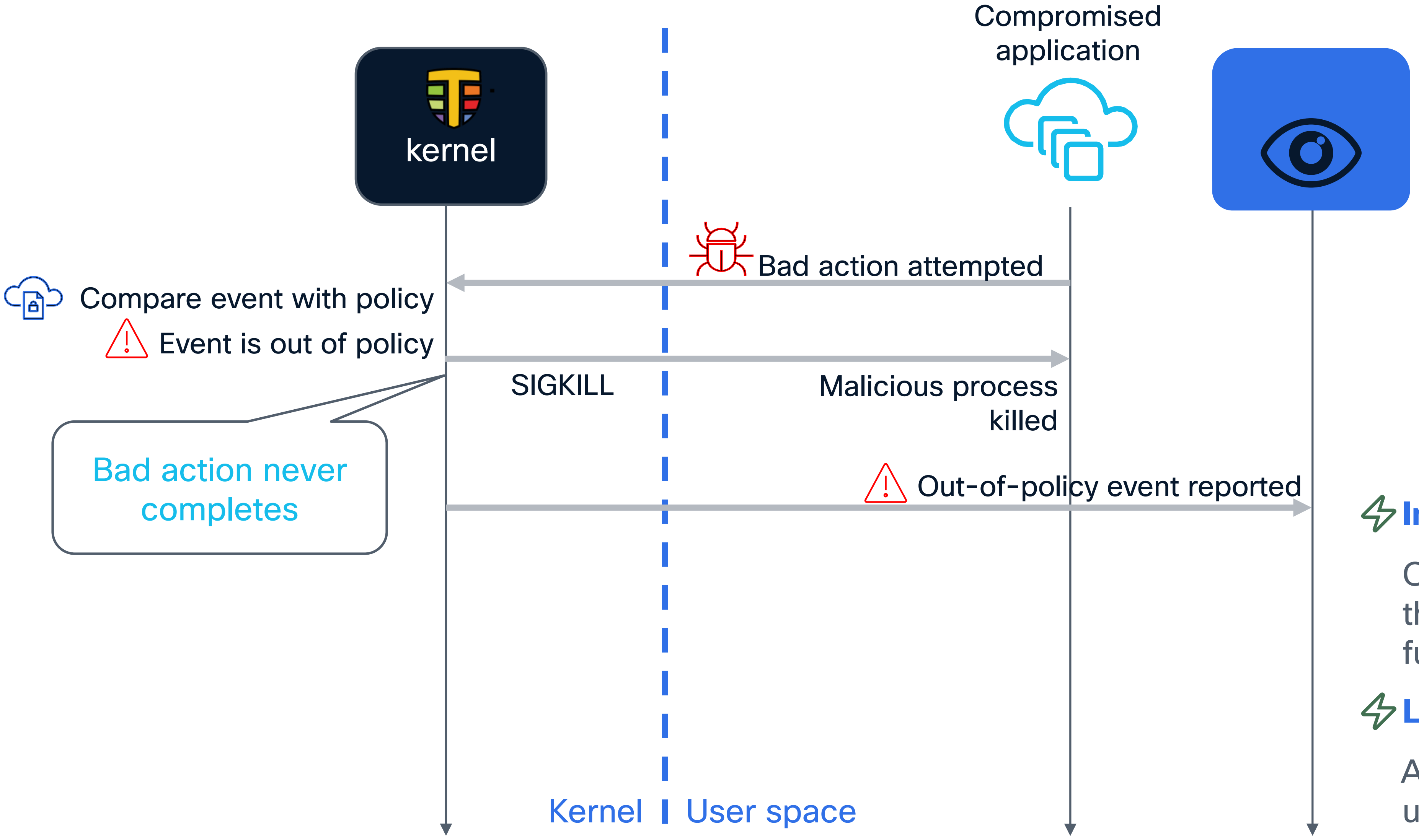
```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: ”example"
spec:
  kprobes:
  - call: ”security_file_permission"
    syscall: false
    args:
    - index: 0
      type: "file"
    - index: 1
      type: "int" # 0x04 is MAY_READ, 0x02 is MAY_WRITE
    selectors:
    - matchArgs:
      - index: 0
        operator: "Equal"
        values:
        - "/tmp/liz"
      matchActions:
      - action: Post
```

Tetragon enforcement

Runtime enforcement – traditional approach



Tetragon runtime enforcement



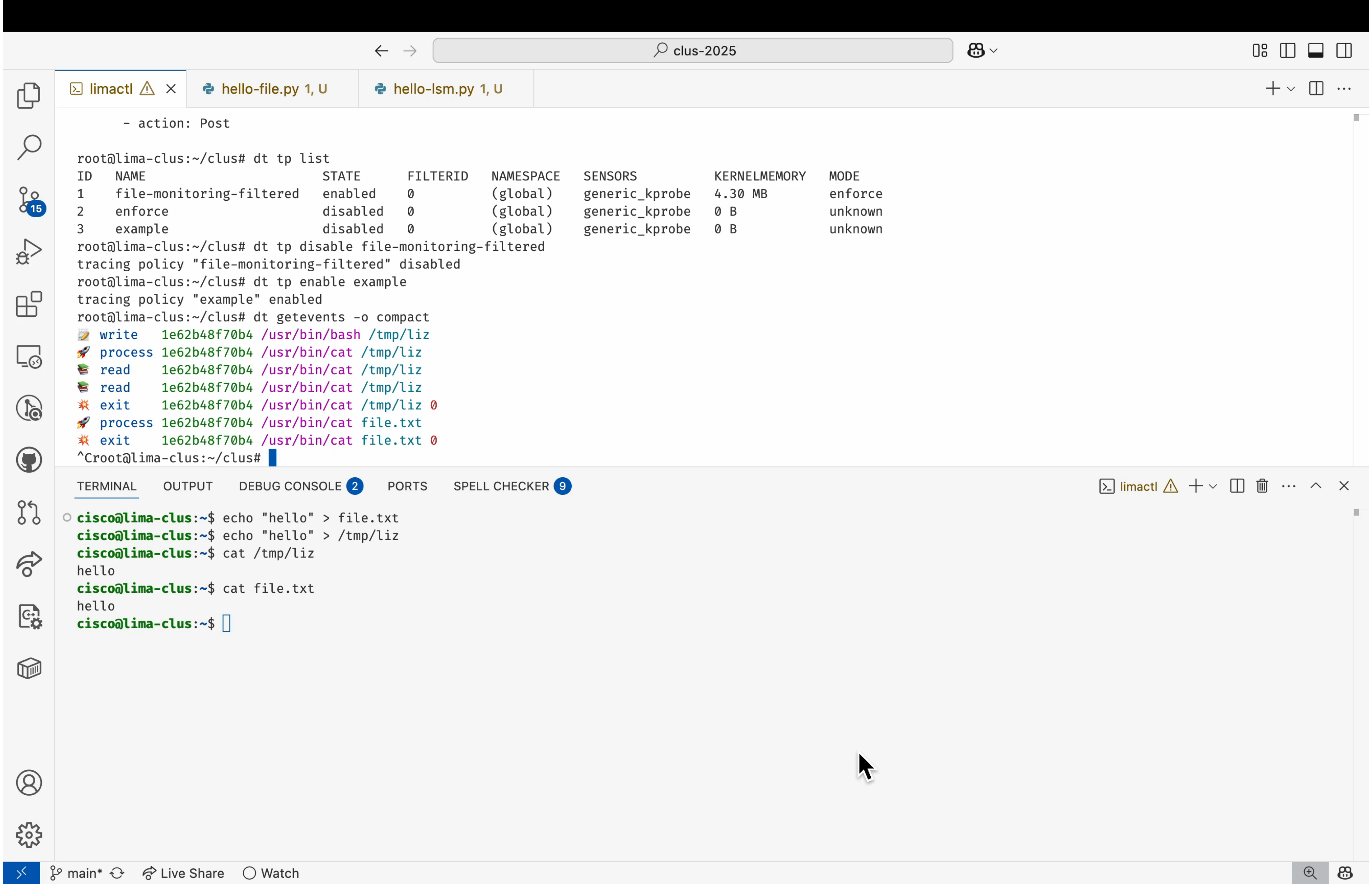
In-kernel enforcement

Options: synchronously SIGKILL the process, or override function return value

Low latency

Avoids overhead of transition to user space

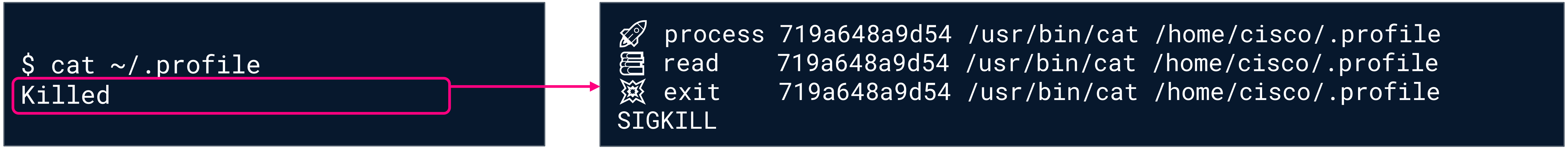
Demo – Tetragon preventing sensitive file access



```
      - action: Post

root@lima-clus:~/clus# dt tp list
ID   NAME                       STATE      FILTERID   NAMESPACE   SENSORS          KERNELMEMORY   MODE
1    file-monitoring-filtered   enabled    0          (global)    generic_kprobe   4.30 MB        enforce
2    enforce                    disabled   0          (global)    generic_kprobe   0 B            unknown
3    example                    disabled   0          (global)    generic_kprobe   0 B            unknown
root@lima-clus:~/clus# dt tp disable file-monitoring-filtered
tracing policy "file-monitoring-filtered" disabled
root@lima-clus:~/clus# dt tp enable example
tracing policy "example" enabled
root@lima-clus:~/clus# dt getevents -o compact
📝 write   1e62b48f70b4 /usr/bin/bash /tmp/liz
🚀 process 1e62b48f70b4 /usr/bin/cat /tmp/liz
📚 read    1e62b48f70b4 /usr/bin/cat /tmp/liz
📚 read    1e62b48f70b4 /usr/bin/cat /tmp/liz
💥 exit    1e62b48f70b4 /usr/bin/cat /tmp/liz 0
🚀 process 1e62b48f70b4 /usr/bin/cat file.txt
💥 exit    1e62b48f70b4 /usr/bin/cat file.txt 0
^Croot@lima-clus:~/clus#
```

```
cisco@lima-clus:~$ echo "hello" > file.txt
cisco@lima-clus:~$ echo "hello" > /tmp/liz
cisco@lima-clus:~$ cat /tmp/liz
hello
cisco@lima-clus:~$ cat file.txt
hello
cisco@lima-clus:~$
```

Tetragon policy:
enforce file access



```
$ cat ~/.profile
Killed
```

```
🚀 process 719a648a9d54 /usr/bin/cat /home/cisco/.profile
📚 read    719a648a9d54 /usr/bin/cat /home/cisco/.profile
💥 exit    719a648a9d54 /usr/bin/cat /home/cisco/.profile
SIGKILL
```

Synchronous termination:

No opportunity for data access

More Tetragon use cases

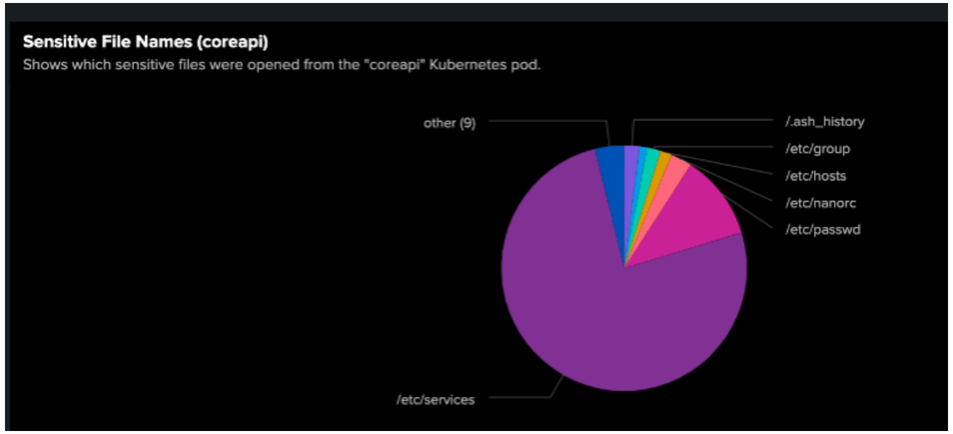
🗁 File integrity monitoring

Detect sensitive file access

Which **binary** performed the operation?

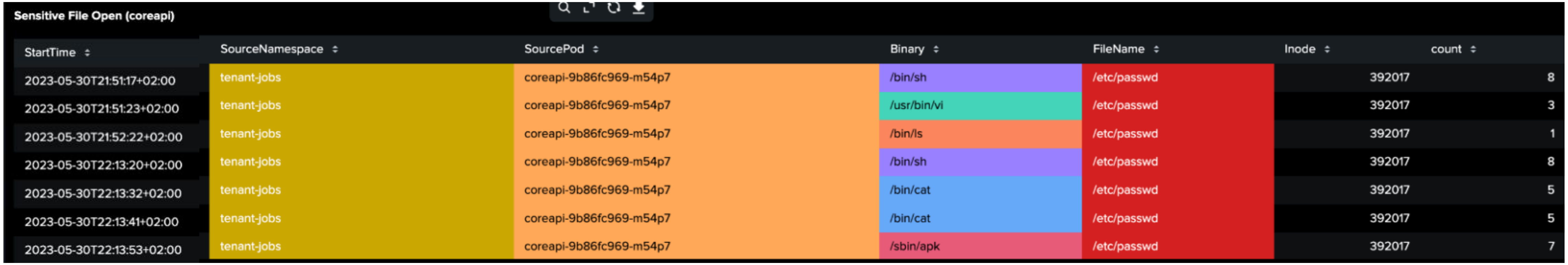
Which **Kubernetes workload**? Which **namespace**?

Did it have **root access** (uid=0)?



Sensitive File Names (coreapi)

Shows which sensitive files were opened from the "coreapi" Kubernetes pod.



Sensitive File Open (coreapi)

StartTime	SourceNamespace	SourcePod	Binary	FileName	Inode	count
2023-05-30T21:51:17+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/bin/sh	/etc/passwd	392017	8
2023-05-30T21:51:23+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/usr/bin/vi	/etc/passwd	392017	3
2023-05-30T21:52:22+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/bin/ls	/etc/passwd	392017	1
2023-05-30T22:13:20+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/bin/sh	/etc/passwd	392017	8
2023-05-30T22:13:32+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/bin/cat	/etc/passwd	392017	5
2023-05-30T22:13:41+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/bin/cat	/etc/passwd	392017	5
2023-05-30T22:13:53+02:00	tenant-jobs	coreapi-9b86fc969-m54p7	/sbin/apk	/etc/passwd	392017	7

Network policies

Monitor TCP connections

Get forensics about unexpected network traffic

Block

Kill processes attempting malicious connections

```yaml
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: "connect"
spec:
  kprobes:
  - call: "tcp_connect"
    syscall: false
    args:
    - index: 0
      type: "sock"
  - call: "tcp_close"
    syscall: false
    args:
    - index: 0
      type: "sock"
  - call: "tcp_sendmsg"
    syscall: false
    args:
    - index: 0
      type: "sock"
    - index: 2
      type: int
```

Tetragon use cases

Network policies

Monitor TCP connections

Get forensics about unexpected network traffic

Block

Kill processes attempting malicious connections

```
apiVersion: cilium.io/v1alpha1
kind: TracingPolicy
metadata:
  name: "connect"
spec:
  kprobes:
  - call: "tcp_connect"
    syscall: false
    args:
    - index: 0
      type: "sock”
    selectors:
    - matchArgs:
      - index: 0
        operator: "DAddr"
        values:
        - "127.0.0.1/8"
        - "192.168.0.0/16"
  - call: "tcp_close"
    syscall: false
    args:
    - index: 0
      type: "sock"
  - call: "tcp_sendmsg"
    syscall: false
    args:
    - index: 0
```

Tetragon use cases

Process lifecycle and privileges

Detect unexpected/unnecessary privileges

Which Kubernetes pods are running with *CAP_SYS_ADMIN* in my cluster?

Which Kubernetes pods have *host network* or *pid namespace* access in my cluster?

Detect privilege escalation

Detect process *capabilities* changes and *kernel namespaces* access

Limit executions

Only permit specified executables

Tetragon use cases

🔐 Host system security

🤔 **Detect / block kernel changes**

Which process or container is changing the kernel?

Which process or container is loading or unloading kernel modules?

Are the loaded kernel modules signed?

Tetragon use cases

Kubernetes Data Exfiltration

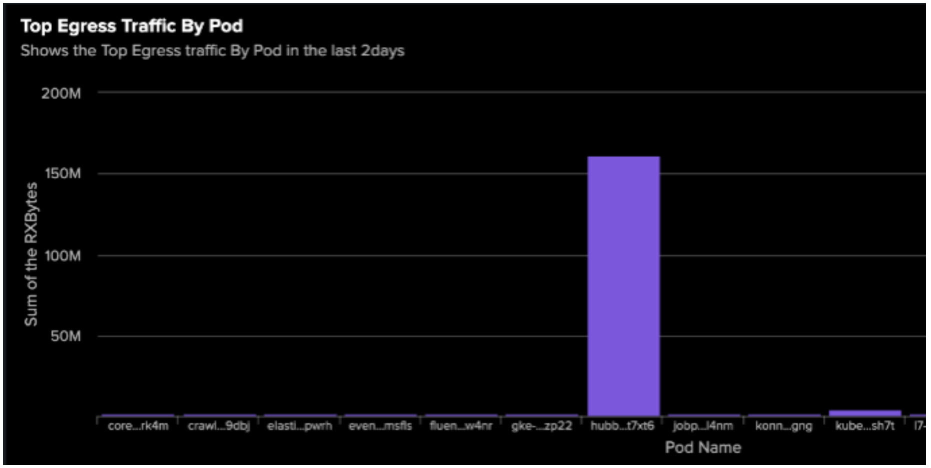
➡ **Detect suspicious levels of egress traffic**

Which workloads sent traffic levels above a suspicion threshold?

Which process initiated it?

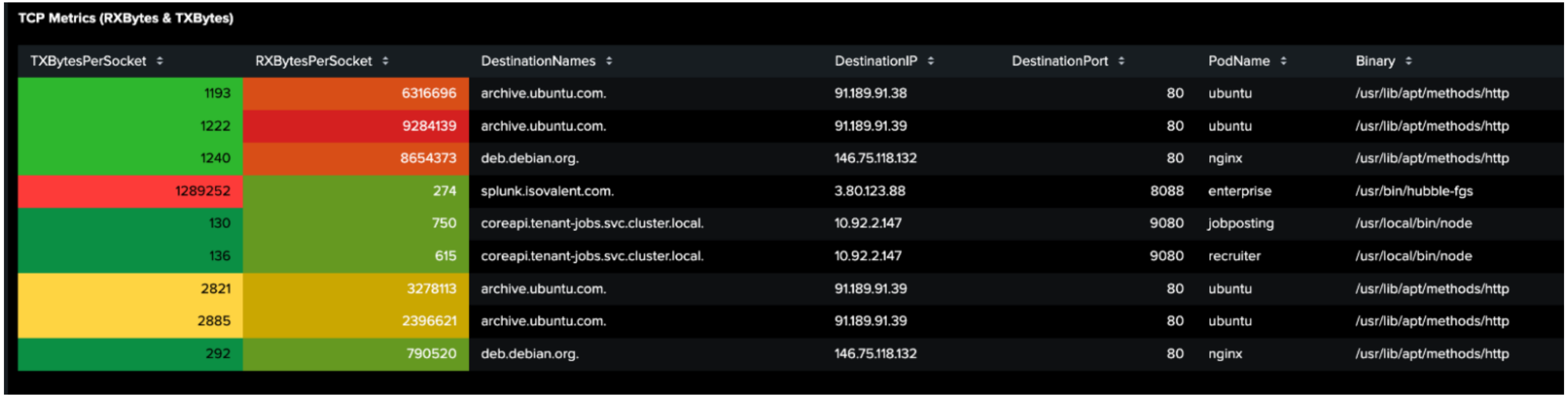
Which team does this workload belong to?

What was the destination?



Top Egress Traffic By Pod

Shows the Top Egress traffic By Pod in the last 2days



TCP Metrics (RXBytes & TXBytes)

TXBytesPerSocket ⇕	RXBytesPerSocket ⇕	DestinationNames ⇕	DestinationIP ⇕	DestinationPort ⇕	PodName ⇕	Binary ⇕
1193	6316696	archive.ubuntu.com.	91.189.91.38	80	ubuntu	/usr/lib/apt/methods/http
1222	9284139	archive.ubuntu.com.	91.189.91.39	80	ubuntu	/usr/lib/apt/methods/http
1240	8654373	deb.debian.org.	146.75.118.132	80	nginx	/usr/lib/apt/methods/http
1289252	274	splunk.isovalent.com.	3.80.123.88	8088	enterprise	/usr/bin/hubble-fgs
130	750	coreapi.tenant-jobs.svc.cluster.local.	10.92.2.147	9080	jobposting	/usr/local/bin/node
136	615	coreapi.tenant-jobs.svc.cluster.local.	10.92.2.147	9080	recruiter	/usr/local/bin/node
2821	3278113	archive.ubuntu.com.	91.189.91.39	80	ubuntu	/usr/lib/apt/methods/http
2885	2396621	archive.ubuntu.com.	91.189.91.39	80	ubuntu	/usr/lib/apt/methods/http
292	790520	deb.debian.org.	146.75.118.132	80	nginx	/usr/lib/apt/methods/http

Tetragon use cases

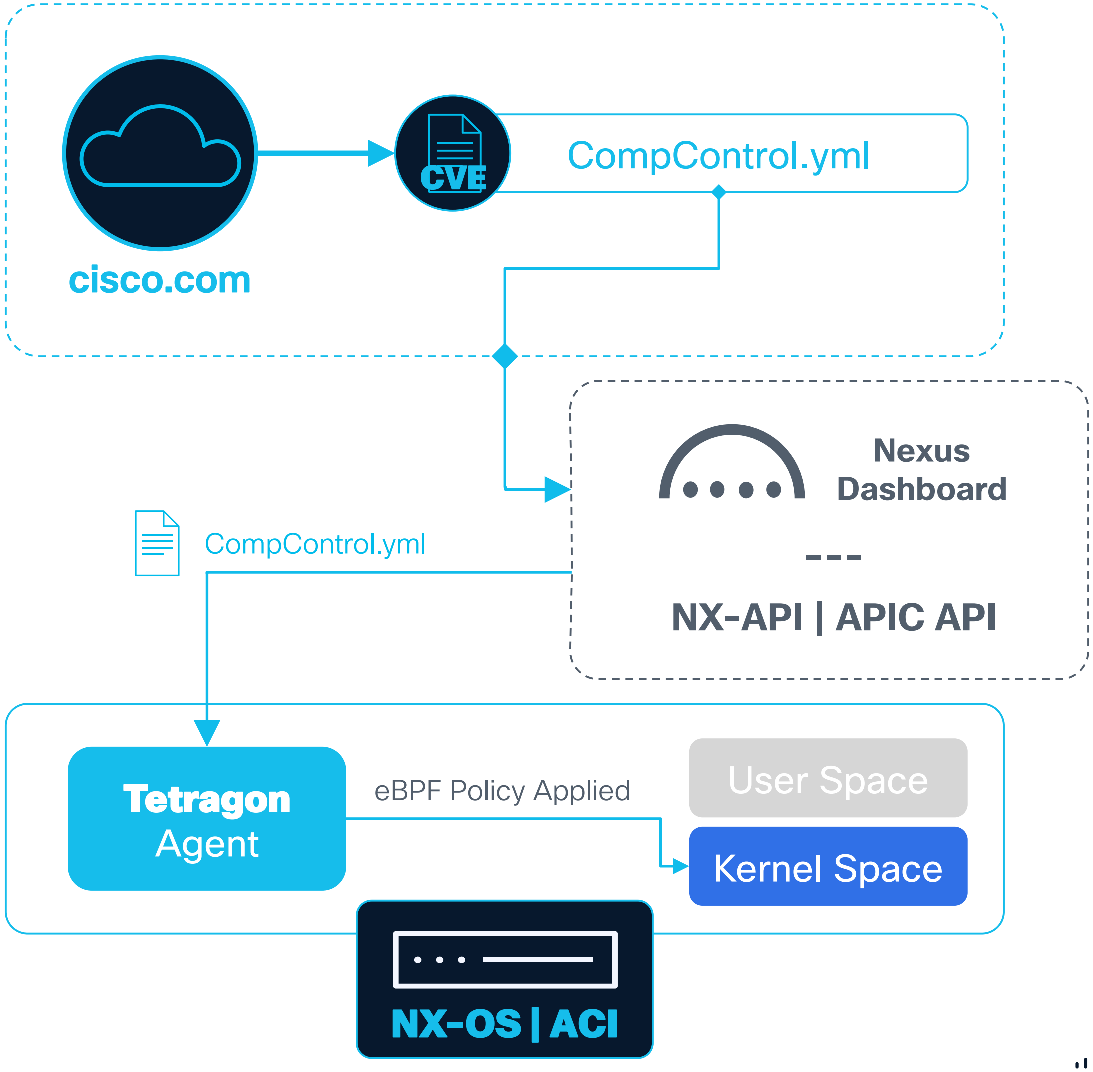
Tetragon Agent on Switch

ACI 6.2(x)F

Planning

Compensating controls against vulnerabilities

Mitigate the risk posed by CVEs without fabric upgrade



eBPF enables tools with security superpowers

Take advantage through Isovalent and Hypershield

Learn more about eBPF security superpowers

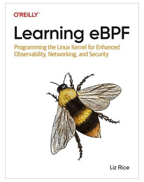
ISOVALENT
now part of CISCO

Visit the AI Ready Data Center area for Isovalent and Hypershield demos

Meet with me and Isovalent specialists through Meet The Engineer

tetragon

Try Tetragon with interactive labs at **isovalent.com/labs**


O'REILLY
Learning eBPF
Programming the Linux Kernel for Enhanced Observability, Networking, and Security
Liz Rice

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

Download eBPF books from **isovalent.com/books**

Contact me at: lizr@cisco.com | LinkedIn | Bluesky: lizrice.com

Complete Your Session Evaluations

Complete a minimum of 4 session surveys and the Overall Event Survey to be entered in a drawing to win 1 of 5 full conference passes to Cisco Live 2026.

Earn 100 points per survey completed and compete on the Cisco Live Challenge leaderboard.

Level up and earn exclusive prizes!

Complete your surveys in the Cisco Live mobile app.

Continue your education

Visit the Cisco Showcase for related demos

Book your one-on-one Meet the Engineer meeting

Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs

Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Contact me at: lizr@cisco.com | LinkedIn | Bluesky: lizrice.com

Do not edit

How to change the design

Audience Q&A

ⓘ The Slido app must be installed on every computer you're presenting from

slido

Thank you

Continue learning at **isovalent.com/labs**

Join BRKSEC-2167 in Cisco Webex App

CISCO Live!