

>_cmd

DETECTION IN LINUX CONTAINERS:

MITRE ATT&CK IN CONTAINER ENVIRONMENTS
& UPDATES TO CUSTOMER ADOPTION

Martin Bowyer &
Jake King



Agenda_

- # Deploying **MITRE ATT&CK Detection policies** in Production.
- # Your feedback from the last session - **gaps in Container Observability**
- # Applying **MITRE ATT&CK & OSS tooling** **Kubernetes environments**



Gathering better data for TTP's on Linux systems

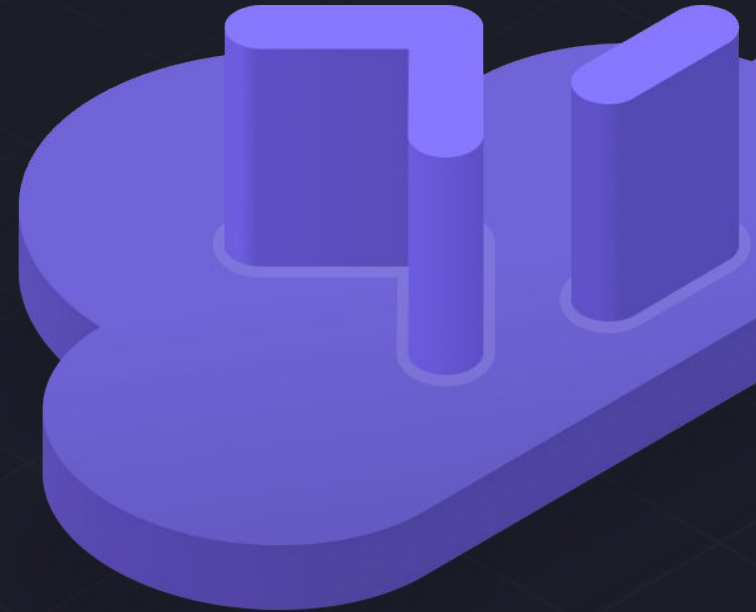
- **We need data** - Execution logs and detailed process / filesystem information
- **We need context** - User, Session, parent / child process information, prevalence, enrichment
- **We need Retention** - Over time, and across environments
- **We must have control** - Whitelists aren't super easy & remediation is post-breach



Gathering better data for TTP's on Linux systems

- **We need data** - Execution logs and detailed process / filesystem information
- **We need context** - User, Session, parent / child process information, prevalence, enrichment
- **We need Retention** - Over time, and across environments
- **We must have control** - Whitelists aren't super easy & remediation is post-breach

Observation



What tools are used for Linux?

- **OS Logs:** AuditD, Syslog, eBPF
 - Augmented alerting via Auditd Policies
- **FOSS Sensors:** ES-Beats, ProcSpy, ExecSnoop, SSH-Bastion
- **Vendors Sensors:** EDR / HIDS / PAM tools
- **Vendor / FOSS SIEM:** Splunk, Sumo, ELK
- **Defense / Remediation Tools:** Ansible, Chef, Puppet, LimaCharlie



MITRE ATT&CK for Linux_

10B+
EXECUTIONS
PROCESSED

60,000+
IOC'S
DETECTED

580+
POLICIES IN
PRODUCTION

One Caveat_



Media & Streaming

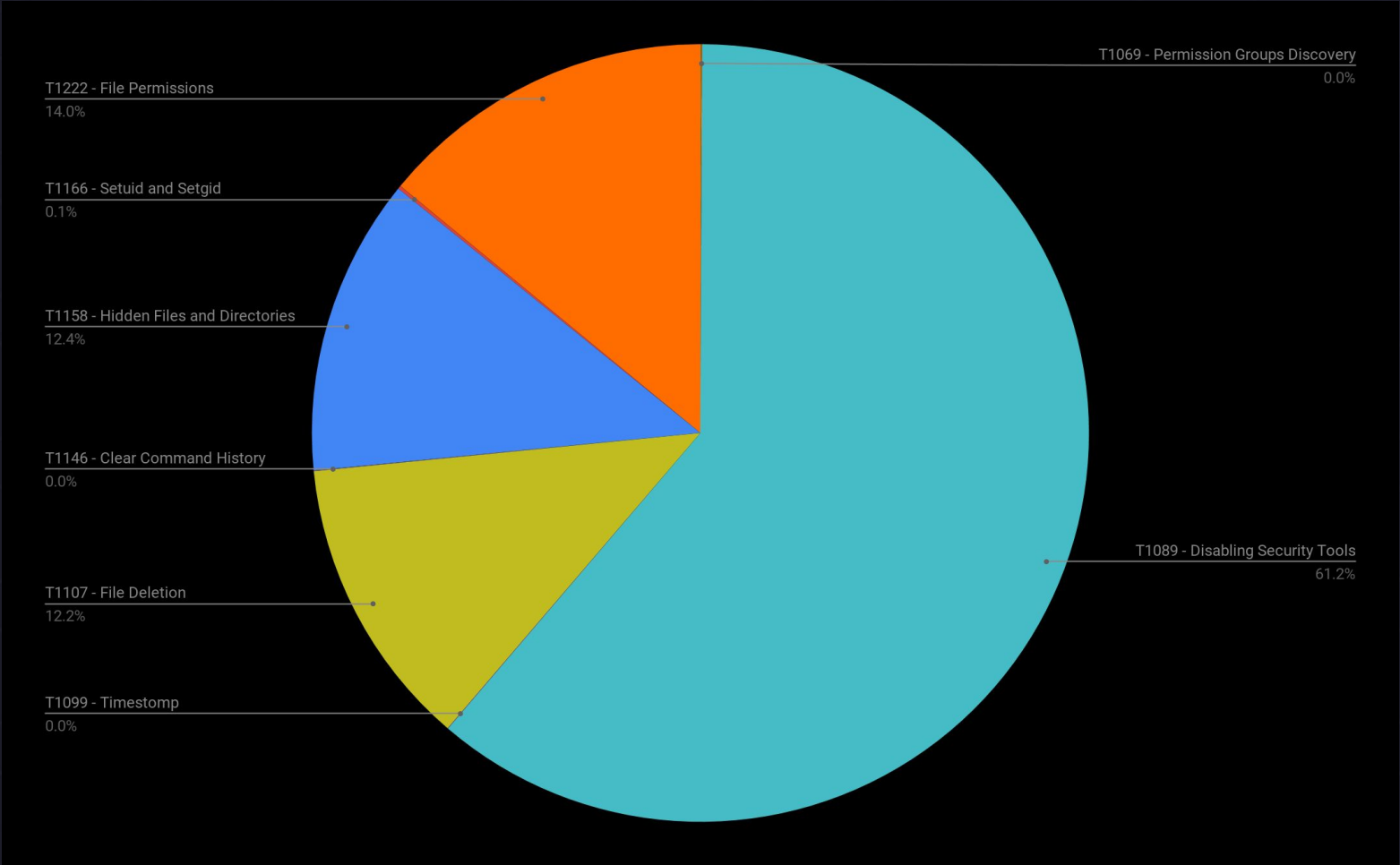


Banking

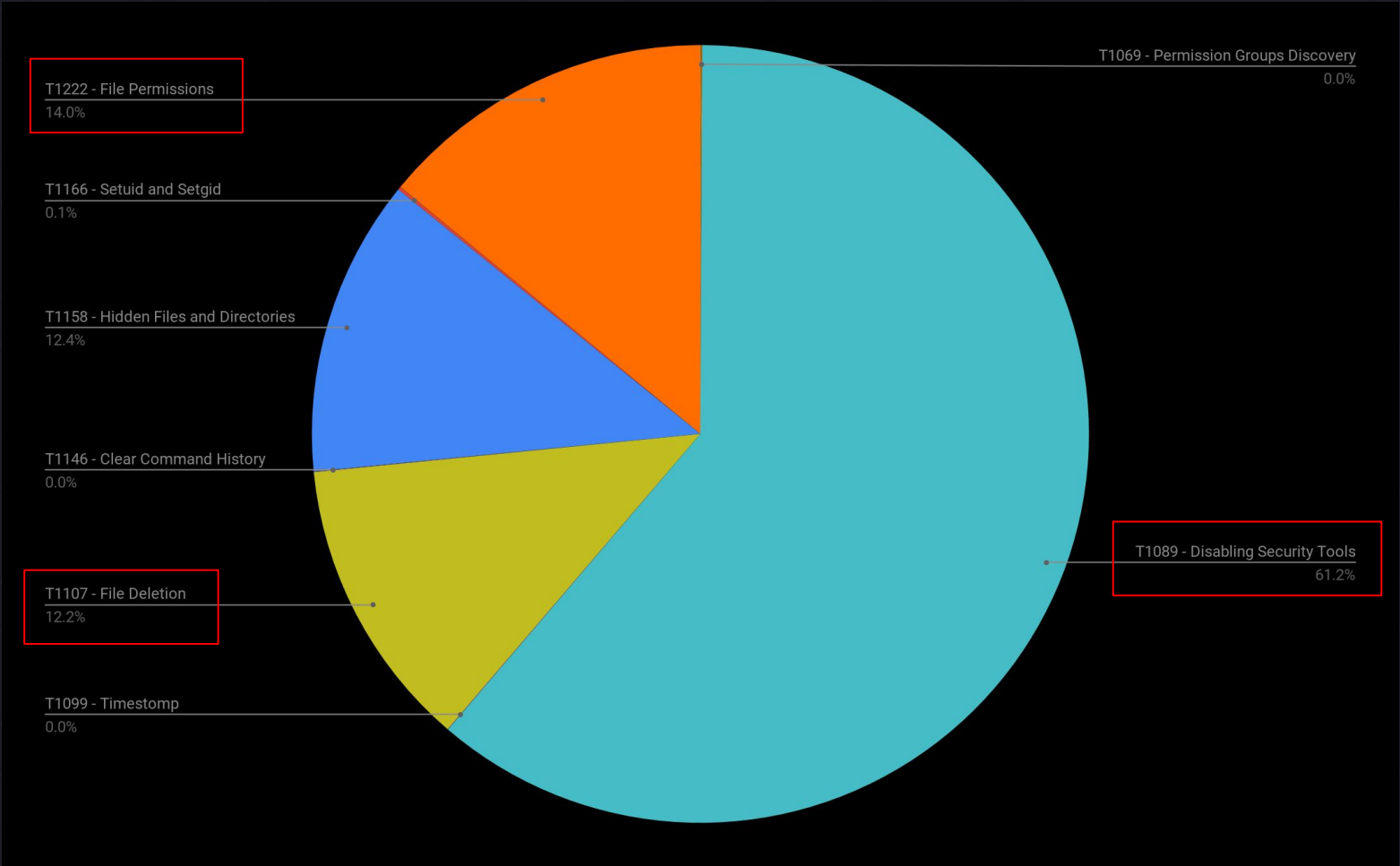


Ecommerce

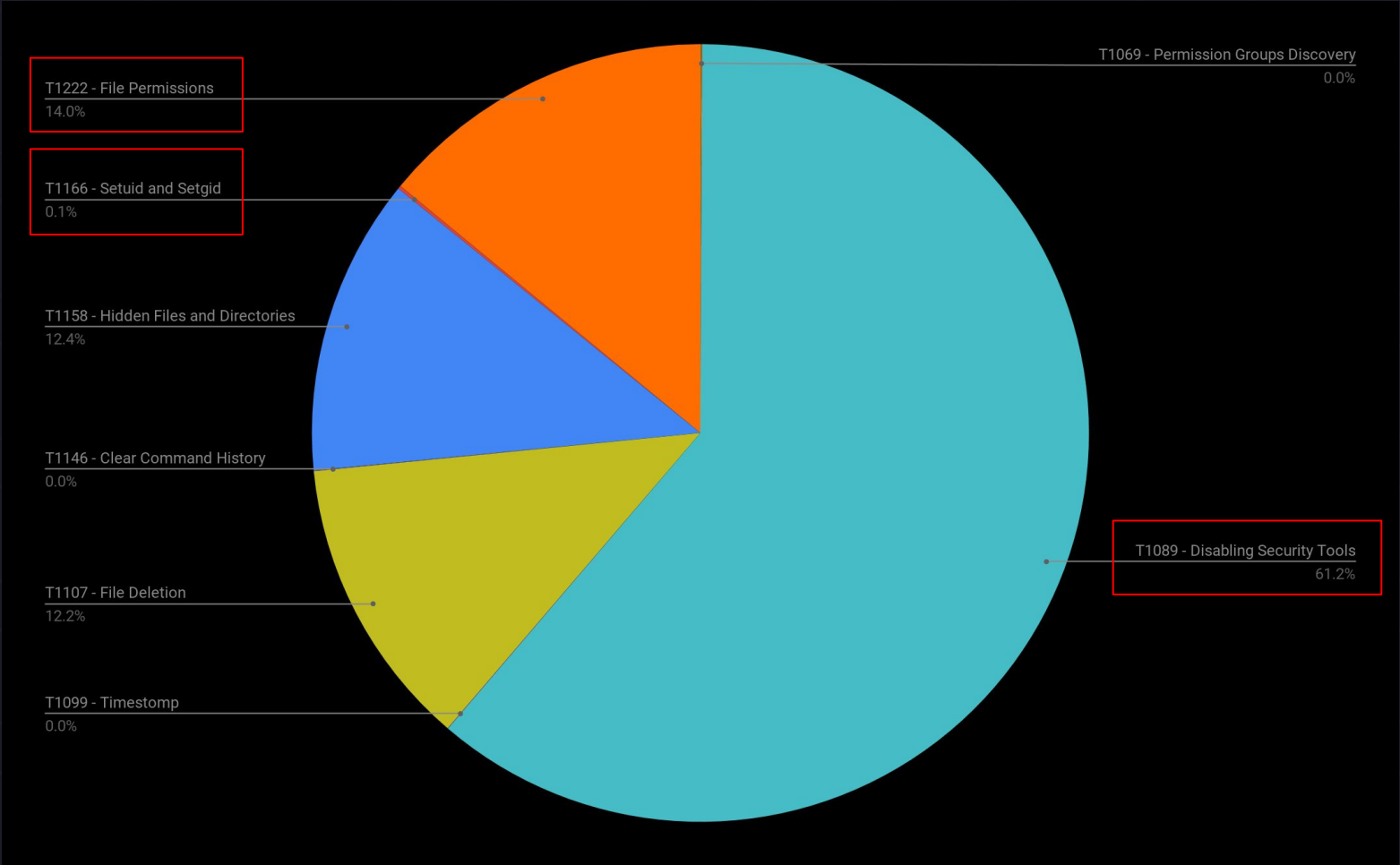
Some Results from Production_



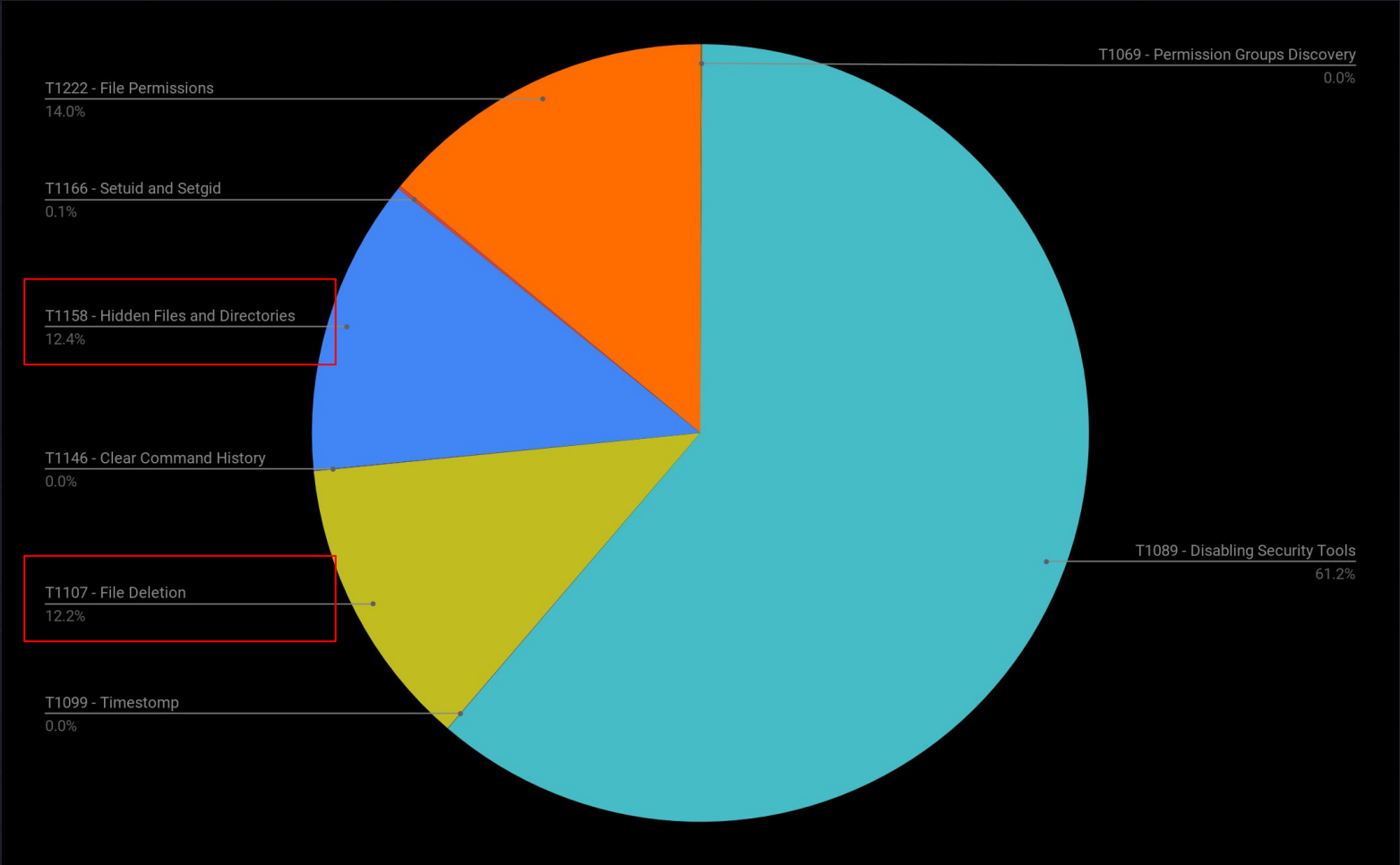
Some Results from Production_



Some Results from Production_

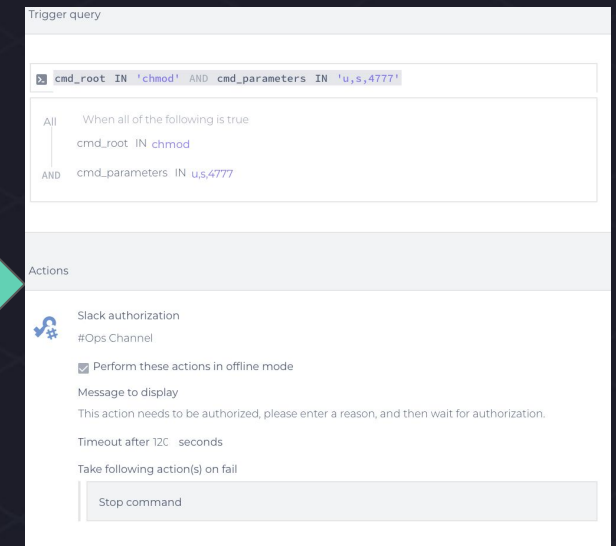
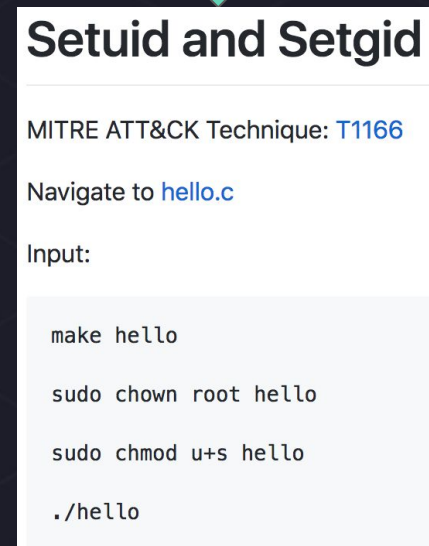
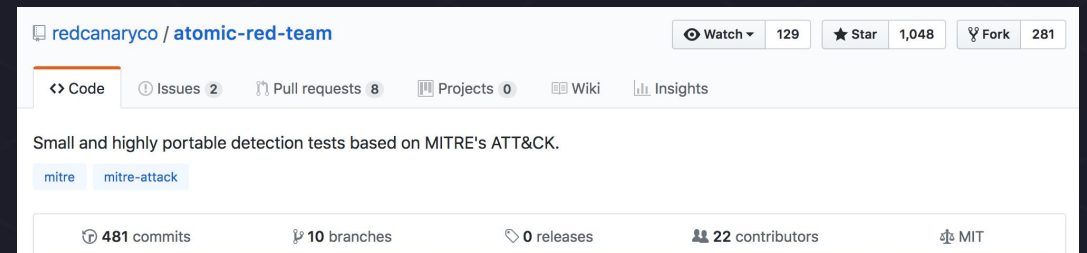


Some Results from Production_



How do we create Policies?

- # Developing **policies via Atomic Red Team**
 - Still working well, but we want to translate / share in SIGMA.
- # Work to be done to polish rules into effectively for Containers / Kube
- # **40+** Policies created, **but can be noisy without custom profiles / Prioritization**



Feedback from our last Session_

*“What about visibility into **Kubernetes**?”*

*“Visibility into **containers**?”*

A gap in Observability: Containers



- # OSS Offerings are limited, and hard to deploy for Containers without the right knowledge
- # Not as low-touch as you'd assume - Constantly touched / interacted with
- # Common abuses
 - # Insider Threats
 - # API key compromise (Kubectl)
 - # Service/image Compromise*

* Check out the talk from Hack.lu on Containing containers.

Observability with eBPF

What is eBPF?

- Bytecode interpreter in the Linux kernel
- Traditionally used for packet filtering
- Extended to other kernel objects
 - kprobes, uprobes, tracepoints

BPF Compiler Collection

- **execsnoop**, **opensnoop**, many more!
- **github.com/iovisor/bcc**

✕ root@mb-demo: /home/mb (ssh)

```
root@mb-demo:/home/mb# execsnoop-bpfcc -t
```

TIME(s)	PCOMM	PID	PPID	RET	ARGS
3.982	uname	1895	1827	0	/bin/uname -a
13.535	ls	1896	1827	0	/bin/ls --color=auto -

```
█
```

✕ mb@mb-demo: ~ (ssh)

```
mb@mb-demo:~$ uname -a
```

```
Linux mb-demo 4.15.0-66-generic #75-Ubuntu SMP Tue Oct 1 05:24:09  
4 x86_64 x86_64 GNU/Linux
```

```
mb@mb-demo:~$ ls -l
```

```
total 0
```

```
mb@mb-demo:~$ █
```

Monitoring a Kubernetes Cluster

- Package eBPF programs in a container image
- Deploy container image to cluster as a Daemonset
 - Single instance per cluster node
 - Automatically scales with the cluster
- Map data collected to ATT&CK leveraging either open signatures and/or custom signatures

```
1 apiVersion: apps/v1
2 kind: DaemonSet
3 metadata:
4   labels:
5     k8s-app: cmd-bcc
6     kubernetes.io/cluster-service: "true"
7   name: cmd-bcc
8 spec:
9   selector:
10    matchLabels:
11      k8s-app: cmd-bcc
12      kubernetes.io/cluster-service: "true"
13   template:
14     metadata:
15       annotations:
16         scheduler.alpha.kubernetes.io/critical-pod: ""
17         scheduler.alpha.kubernetes.io/tolerations: '[{"key": "dedicated"}]'
18       labels:
19         k8s-app: cmd-bcc
20         kubernetes.io/cluster-service: "true"
21     spec:
22       containers:
23       - name: cmd-bcc
24         image: cmdinc/bcc:latest
25         command: ["bash"]
26         args: ["/scripts/bcc_setup.sh"]
```

Demo

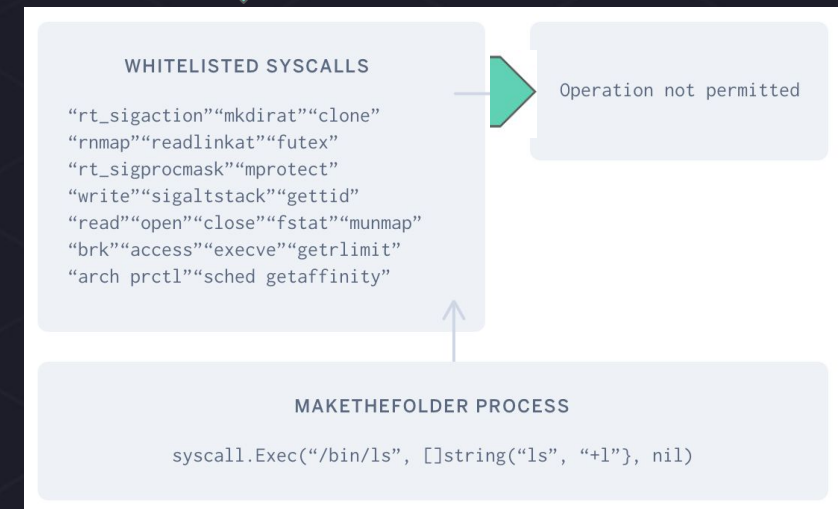
Cmd

17

Applying MITRE ATT&CK to eBPF Data_

- Standard hardening / minimization will defend against numerous attacks, but not all
- eBPF data can be used to capture syscalls being made with **Atomic Red Team** to build effective preventative rules
- Policies can then be created to whitelist syscalls at the point of a binary execution with **SecComp / SELinux / AuditD**
- Kernel 5.0 brings some of these features to user space for awesome future projects!

```
mprotect(0x7fb8fb8de000, 4096, PROT_NONE) = 0
clone(child_stack=0x7fb8fc0ddff0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THRE
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
futex(0x72f7c8, FUTEX_WAIT, 0, NULL) = 0
rt_sigprocmask(SIG_SETMASK, ~[RTMIN RT_1], [], 8) = 0
mmap(NULL, 8392704, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x7fb8
mprotect(0x7fb8fb0dd000, 4096, PROT_NONE) = 0
clone(child_stack=0x7fb8fb8dcff0, flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THRE
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
readlinkat(AT_FDCWD, "/proc/self/exe", "/home/brompwnie/go/src/github.co...", 128) = 68
mmap(NULL, 262144, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fb8fda88000
mkdirat(AT_FDCWD, "/tmp/moo", 0755) = 0
write(1, "I just created a file\n", 22) = 22
exit_group(0) = ?
+++ exited with 0 +++
```



>_cmd

THANKYOU!

Martin Bowyer &
Jake King
cmd.com

