Zach Wasserman – CTO, Fleet

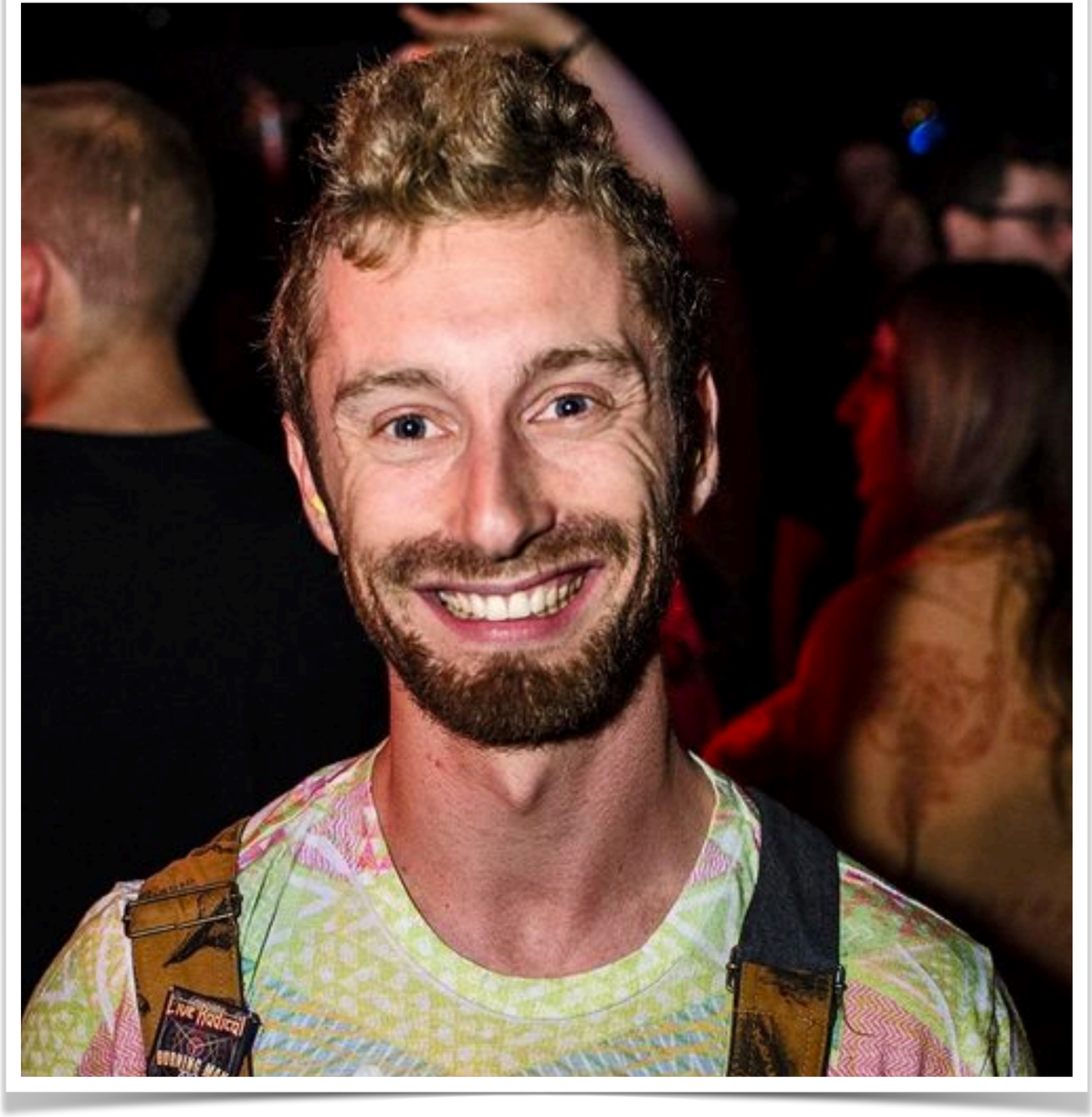
eBPF and the future of osquery on Linux

osquery@scale 2021

fleet

Who am I

- Co-creator of osquery
- Cofounder & CTO of FleetDM
- (Former) Cofounder & Principal Engineer of Kolide



Silent Accomplice

Alessandro Gario
Senior Software Engineer
Trail of Bits

Audit & Osquery

fleet

Audit & Osquery

Tables

- Audit powers many of the event-based tables for osquery on Linux
  - `process_events`
  - `process_file_events`
  - `socket_events`
  - `user_events`
  - `apparmor_events`
  - `selinux_events`

Audit & Osquery

Configuration

- Base configuration to enable audit
  - `--disable_audit=false`
  - `--audit_allow_config=true`
  - `--audit_persist=true`

Audit & Osquery

Configuration

- Enable each feature separately
  - `--audit_allow_`**`apparmor_events`**
  - `--audit_allow_`**`fim_events`**
  - `--audit_allow_`**`fork_process_events`**
  - `--audit_allow_`**`kill_process_events`**
  - `--audit_allow_`**`process_events`**
  - `--audit_allow_`**`selinux_events`**
  - `--audit_allow_`**`sockets`**
  - `--audit_allow_`**`user_events`**

Audit Drawbacks

fleet


There can be only one

Audit Drawbacks

There can be only one

- Audit’s design allows only a single consumer of the generated events
- Receiving audit events in osquery means disabling auditd
- Disable auditd -> No audit events written to file
  - Some tools expect to be able to retrieve audit events from file!

Highlander Quote - *Linux Audit: Moving Beyond Kernel Namespaces to Audit Container IDs* - Richard Guy Briggs (Linux Security Summit 2018)

Audit Drawbacks

There can be only one

- Many SELinux tools rely on audit logs in `/var/log/audit/audit.log`
  - `sealert`
  - `audit2allow`

Audit Drawbacks

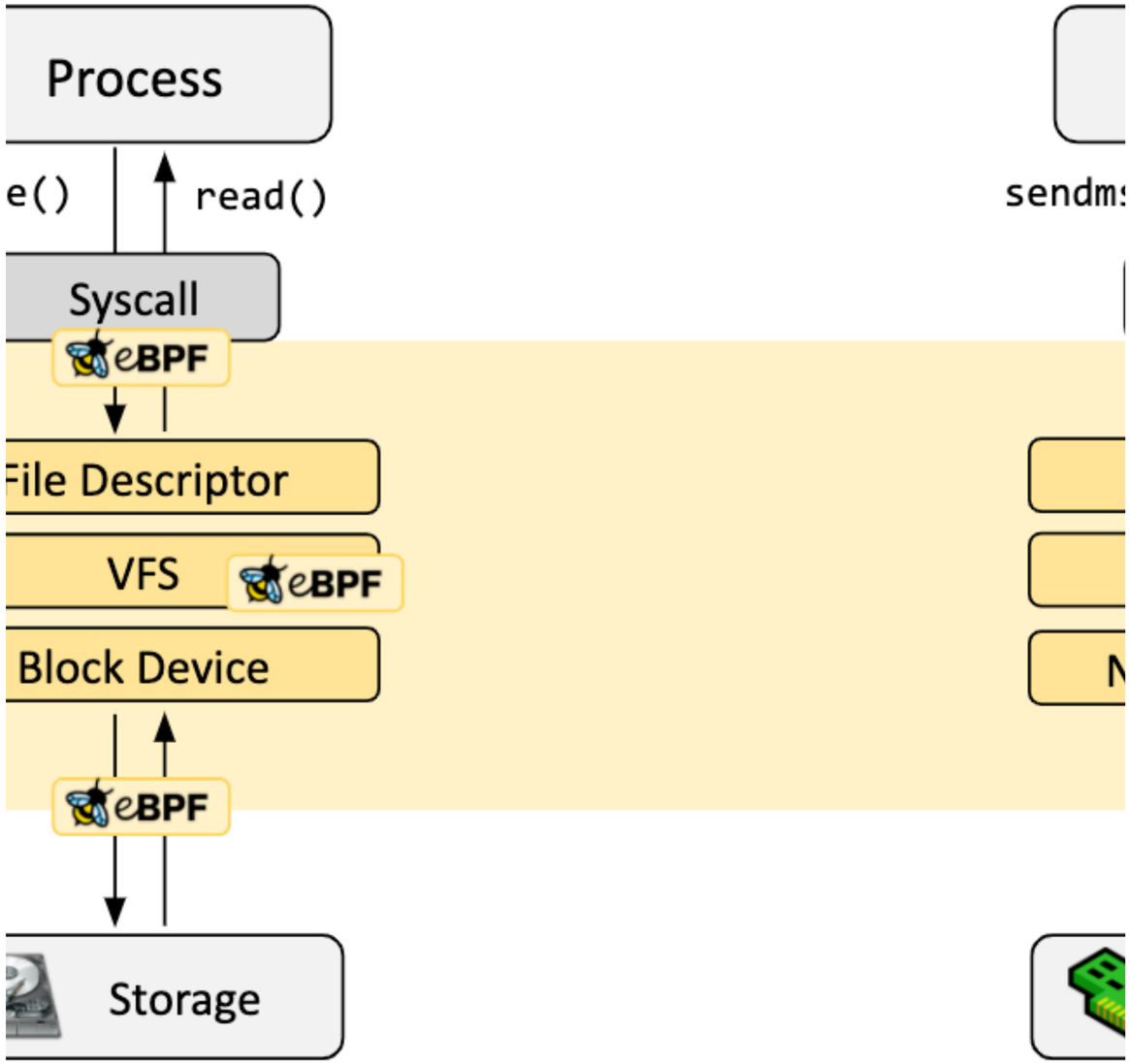
Containers

- Audit does actually work with containers
  - Fixed in Red Hat Bug 893751
- Audit is not “aware” of containers
  - Lack of namespace information hinders usability in container workloads

eBPF

eBPF

Programmable Hooks



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

eBPF

Safety

- Required privileges
- Program verification
  - Limited size
  - Limited complexity
  - Bounded loops
- Controlled memory access

eBPF & Osquery

eBPF & Osquery

State

- eBPF functionality released in osquery 4.6.0
  - Built primarily by Alessandro Gario
- Implemented on top of github.com/trailofbits/ebpfpub

eBPF & Osquery

Tables

- `bpf_process_events`
- `bpf_socket_events`
- ...

eBPF & Osquery

Configuration

- `--enable_bpf_events`
- That’s it!

eBPF & Osquery

Tuning

- `--bpf_buffer_storage_size` (default 512)
- `--bpf_perf_event_array_exp` (default 10)

eBPF & Osquery

Compatibility

- Targeting Kernels 4.18+ (2018)
  - Possible to extend compatibility back to 4.10+ (2017)
- eBPF Probes are generated at runtime
  - One binary can work on most Kernels

eBPF & Osquery

Coming soon...

- Support for correlating BPF events with containers
  - Mapping cgroup_ids to Docker containers
- `process_dns_events`

eBPF & Osquery

Future

- We now have a pattern for instrumenting nearly anything on Linux
  - System calls
  - Kernel tracepoints
  - User-space function calls
- These can be dynamically configured at osquery runtime

eBPF & Osquery

Future - Security

- Instrument any and all syscalls of interest
- Track signals sent to processes
- Kernel module loads/unloads
- Track LD_PRELOAD

eBPF & Osquery

Future - Devops/SRE

- Instrument any and all syscalls of interest(!)
- Measure latency and resource consumption of OS processes
  - Network stack
  - Filesystem
  - Other I/O
- Count and measure functions within user-space

eBPF & Osquery

Future - Imagine

- Let’s look at the tools of today
  - bpftrace
  - BCC
- Which of these use cases map well to osquery’s SQL model?
- How can osquery be useful for shipping the aggregated information from hosts?

Conclusion

Audit & eBPF are both viable approaches

eBPF has potential to dramatically increase scope of observability with osquery

Thank you

zach@fleetdm.com

@thezachw

@zwass

fleet