Runtime Security with eBPF

Datadog

Sylvain Afchain
sylvain.afchain@datadoghq.com

Sylvain Baubeau
sylvain.baubeau@datadoghq.com

Guillaume Fournier
guillaume.fournier@datadoghq.com

I.Runtime security

I. Runtime security

What is it and why it is important

- Detection of IOC (Indicator Of Compromise)
- Highly dynamic environments
- Third party dependency scanner surely helps
- Zero days are a thing
- Compliance requirement

I. Runtime security
Constraints

- Event context
- Safety
- Low overhead
- Wide support of kernels

II.Extended Berkeley Packet Filter (eBPF)

II. Extended Berkeley Packet Filter (eBPF)

- Virtual machine in the kernel
- Hook to kernel functions using kprobes
- Lots of limitations: no loop, 4096 instructions, 512 bytes stack, ...
- Highly dependent on kernel version

II. Extended Berkeley Packet Filter (eBPF)

User / Kernel space communication

- Maps
  - In Kernel key/value data stores
  - User space access through file descriptor
  - Hash maps, array, LRU, ...
  - No bulk operation

- Ring Buffer
  - Stream of events

II. Extended Berkeley Packet Filter (eBPF)

Context resolution

- Syscall levels is not enough
  - Insufficient context: relative path, mount point unresolved, symlink
  - Vulnerable to TOCTOU attacks
  - Page faults

- Kprobes on multiple hookpoints of the call flow
  - Syscall entry
  - Path resolution using dentry structures, program capabilities
  - At syscall return, we send (or not) event to userspace

III. Datadog Runtime Security Agent

Architecture

File Integrity Monitoring

Process Execution Monitoring

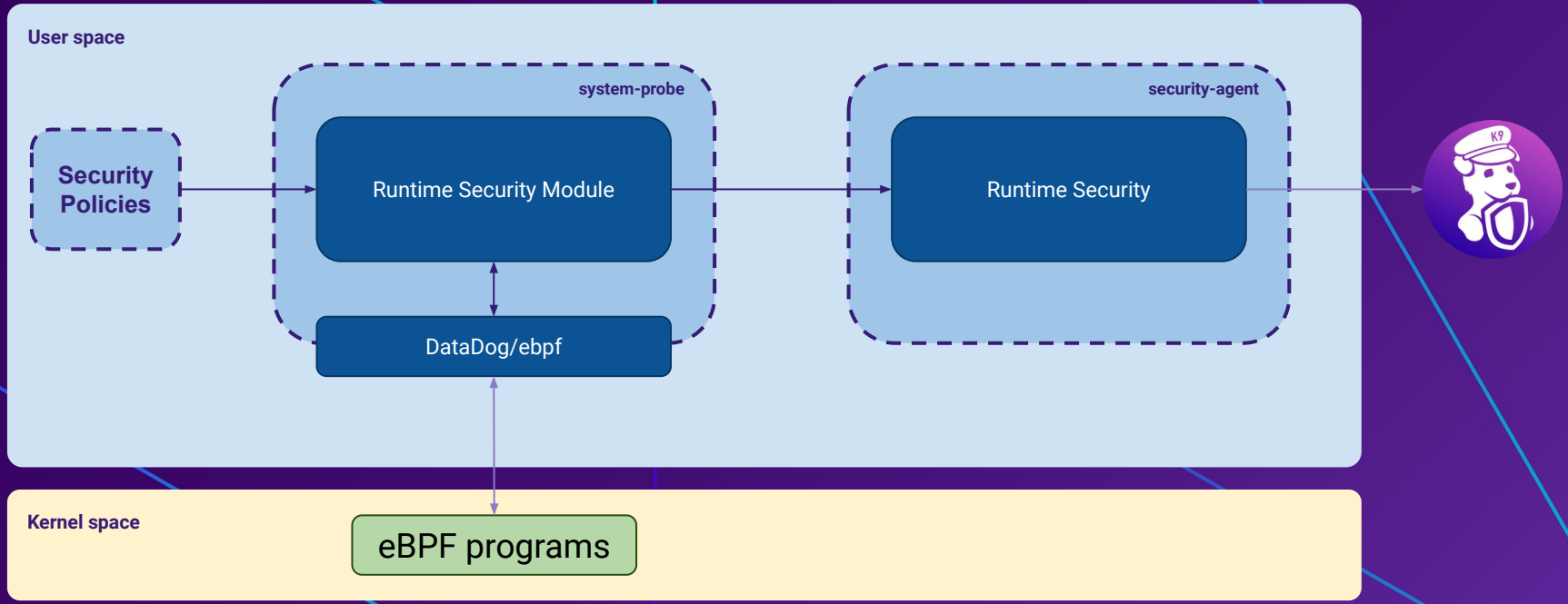
III. Datadog Runtime Security Agent

Architecture

- 2 services
  - System-probe
  - Security-agent
- eBPF programs
  - Depending on event type, kernel versions, etc.
  - kprobe/kretprobe
  - Tracepoint
- Rule engine
  - Evaluation
  - Determine In-kernel filters

III. Datadog Runtime Security Agent

Architecture



III. Datadog Runtime Security Agent

Architecture - User / Kernel space communication

- Maps
  - Used for in-Kernel Filters
  - Used for file path resolution
- Ring Buffer
  - Stream of events

III. Datadog Runtime Security Agent

Architecture - Rule engine, why a dedicated language

- Determine what hook points are required at rule compilation time
- Determine a first set of in-kernel filters at rule compilation time
- Extract in-kernel filters at runtime
- Optimized lazy evaluation

III. Datadog Runtime Security Agent

Architecture - Approvers

- In-kernel filters at compile time
- Extracted from the whole set of rules
- Values that for sure match a least one rule

`open.file.path == “/etc/shadow” && open.flags & O_RDWR > 0`

`Approvers => Basename: shadow; Flags: O_RDWR`

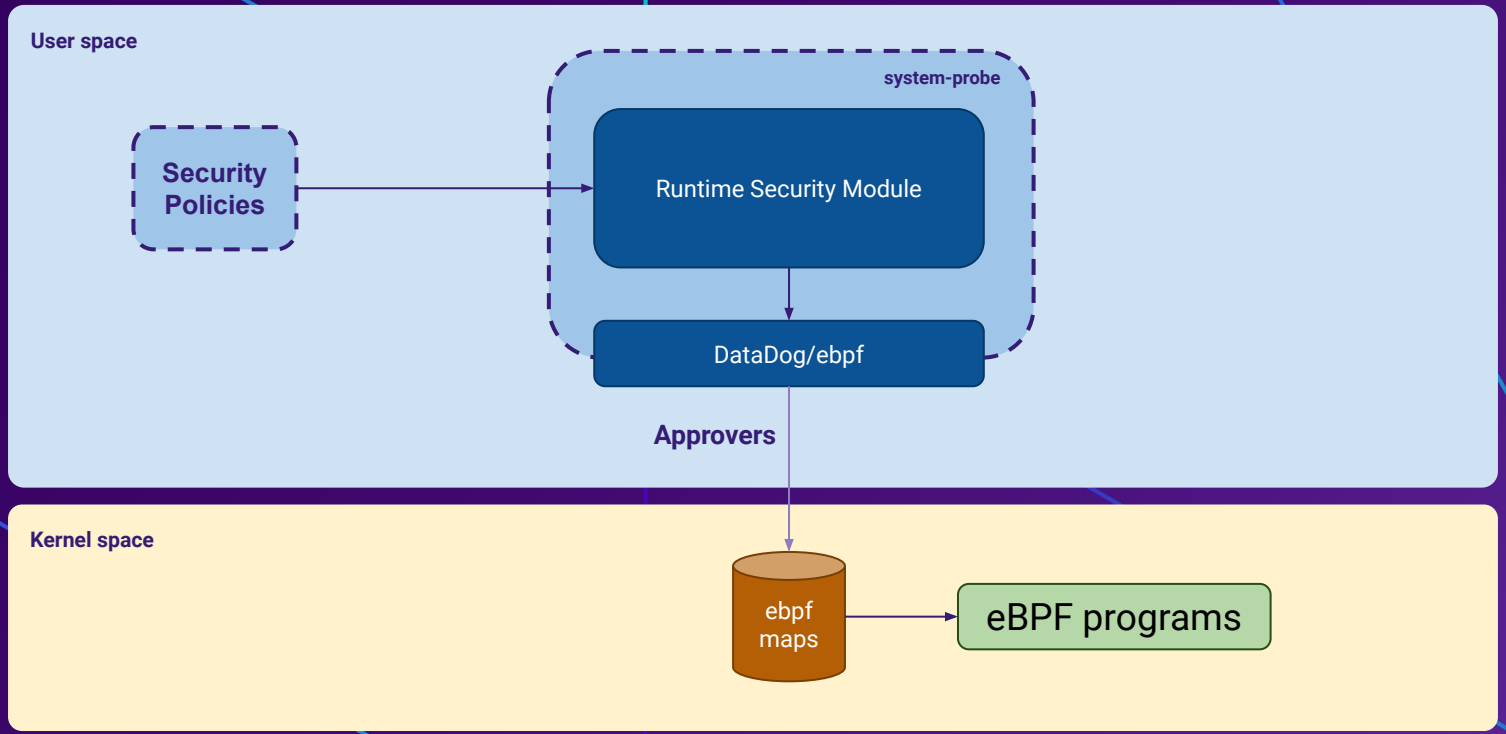
- Some limitations, doesn’t work with wildcards

`open.file.path =~ “/etc/*” && open.flags & O_RDWR > 0`

`Approver => Flags: O_RDWR`

III. Datadog Runtime Security Agent

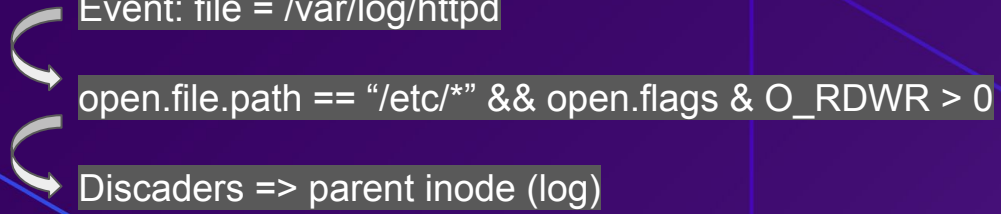
Architecture



III. Datadog Runtime Security Agent

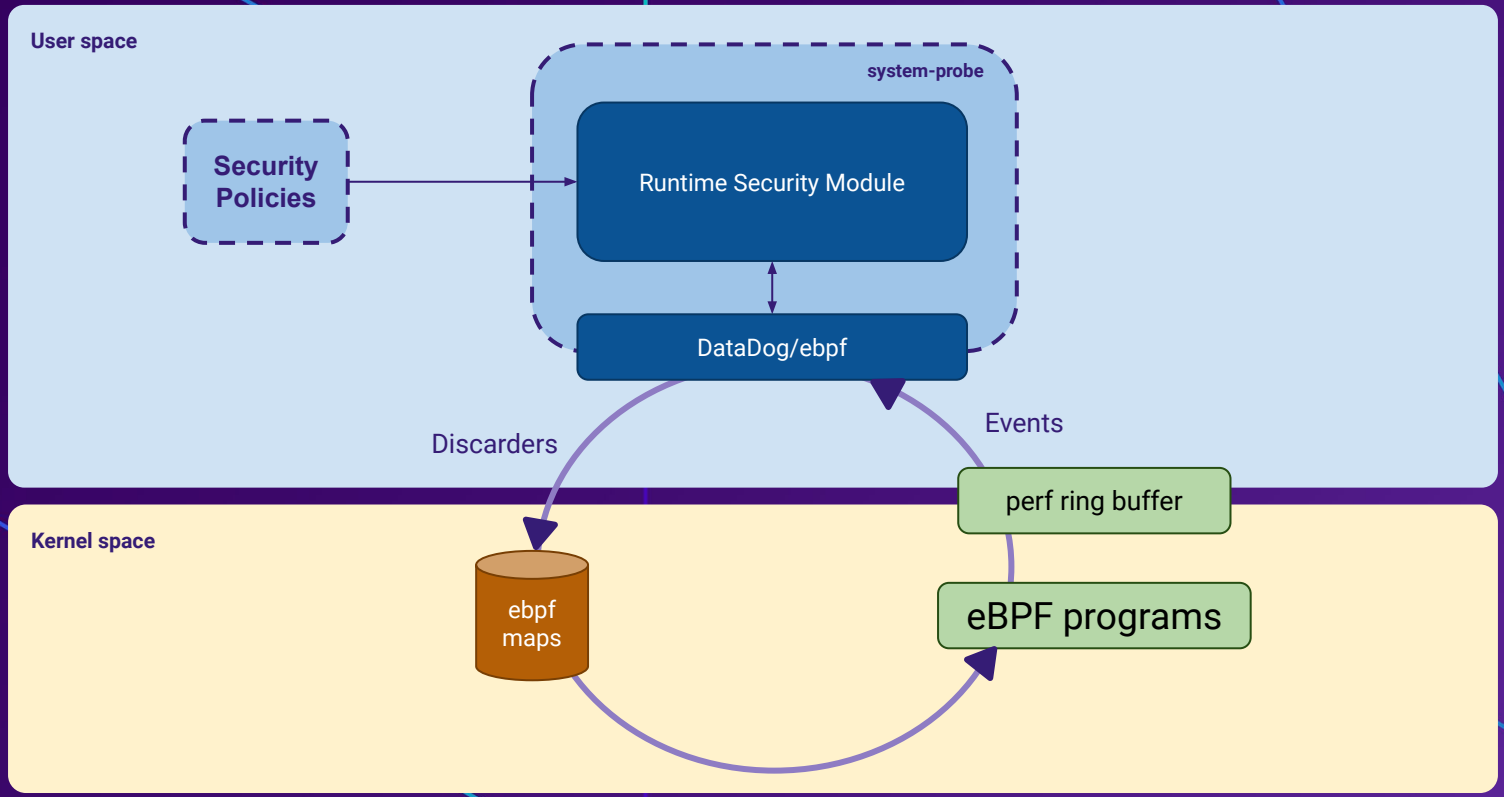
Architecture - Discarders

- In-kernel filters at runtime from an event
- Extracted from pre-compiled dedicated rules
- Values that for sure won't match any rules



III. Datadog Runtime Security Agent

Architecture



III. Datadog Runtime Security Agent

Architecture

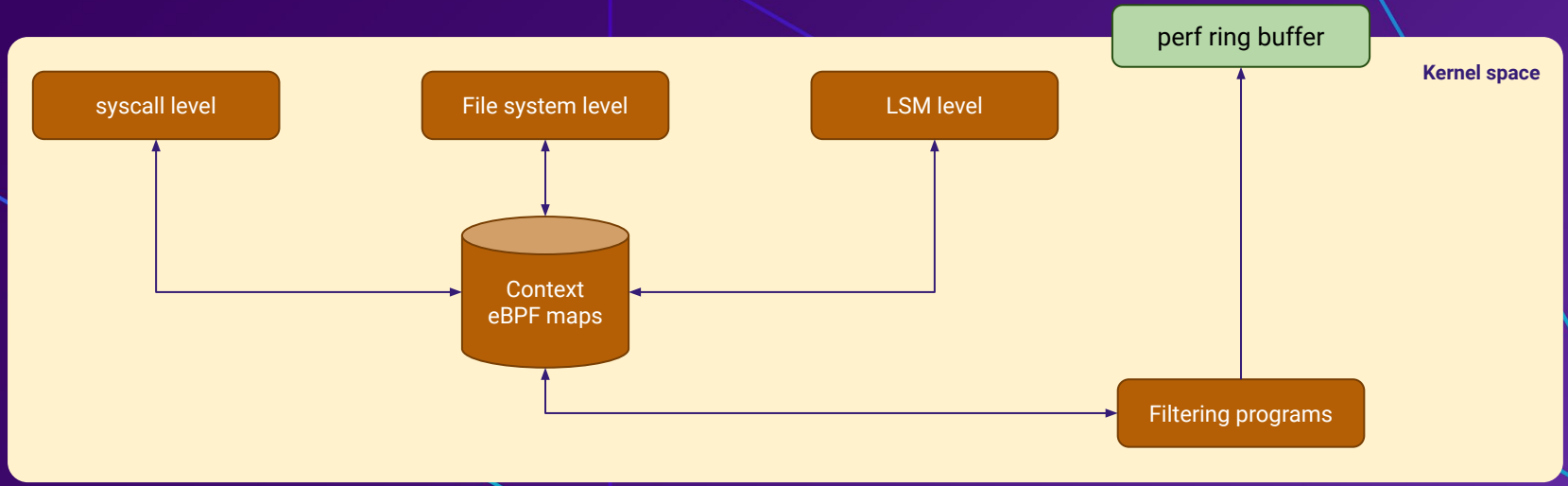
File Integrity Monitoring

Process Execution Monitoring

III. Datadog Runtime Security Agent

File Integrity Monitoring

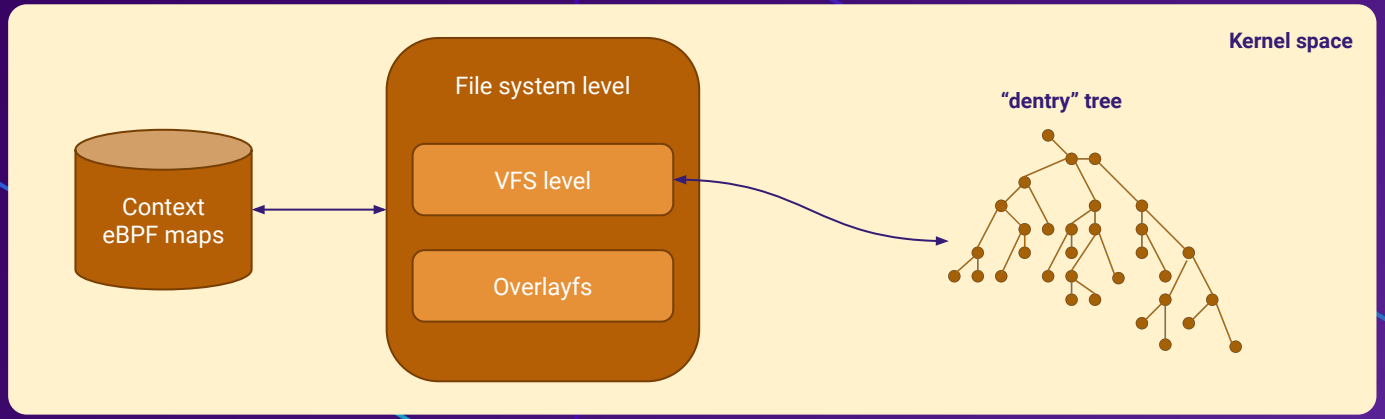
- Detect content & attributes changes
- 12 event types: open, chmod, mkdir, link, mount, …
- Multi stage context gathering



III. Datadog Runtime Security Agent

File Integrity Monitoring

- We choose the granularity of the collected data:
  - Dentry resolution with metadata
  - Layer on overlayfs



Demo

```
---
version: 1.0.0
rules:

  - id: SSTIC_fim_credentials_leak
    description: Credentials file accessed by an unknown process
    expression: open.file.path == "/etc/my_secrets" && process.file.path != "/usr/local/bin/webapp"

  - id: SSTIC_fim_credentials_delayed_access
    description: Credentials file accessed unexpectedly late
    expression: open.file.path == "/etc/my_secrets" && process.file.path == "/usr/local/bin/webapp" &&
                process.created_at > 5s
```

III. Datadog Runtime Security Agent

Architecture

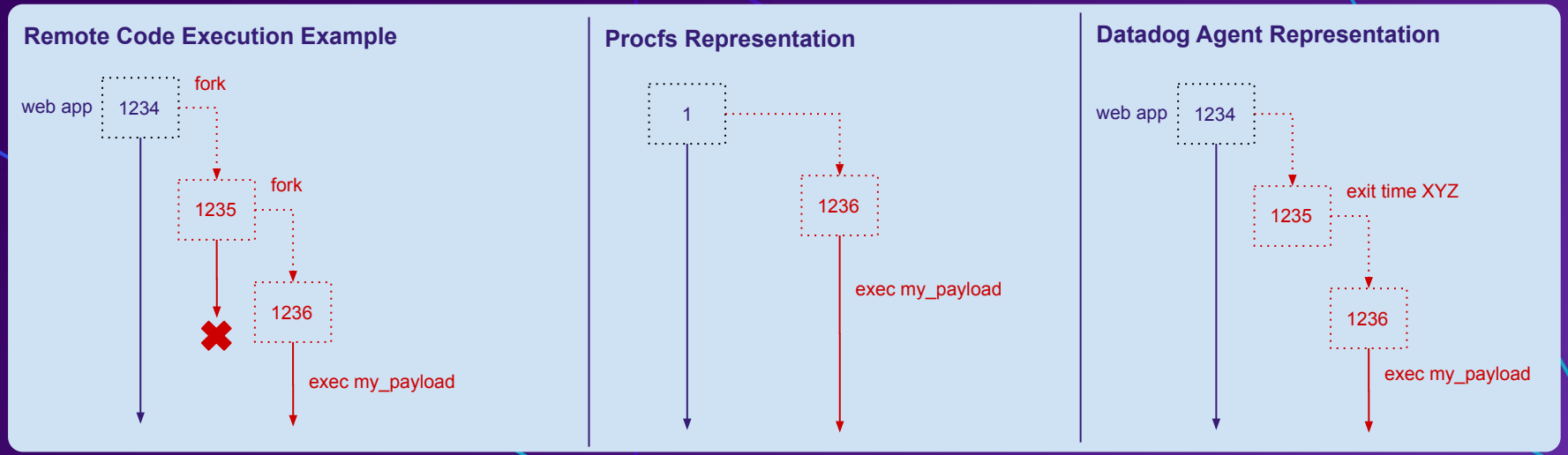
File Integrity Monitoring

Process Execution Monitoring

III. Datadog Runtime Security Agent

Process Execution Monitoring

- Detect abnormal process execution patterns
- Multi stage context gathering
- Historical process tree with short lived processes



Demo

```
---
version: 1.0.0
rules:

  - id: SSTIC_exec_payload
    description: Execution of a payload dropped in a container or of a binary modified from the base image
    expression: process.ancestors.file.path == "/usr/local/bin/webapp" && exec.file.in_upper_layer == true

  - id: SSTIC_exec_shell
    description: Execution of a remote shell
    expression: process.ancestors.file.path == "/usr/local/bin/webapp" && exec.file.name in ["bash", "sh", ...]

  - id: SSTIC_exec_unknown_binary
    description: Execution of unknown binary
    expression: process.ancestors.file.path == "/usr/local/bin/webapp" && exec.file.name not in ["bash", "sh", ...]
```

Thanks !

github.com/DataDog/datadog-agent