Monitoring and protecting SSH sessions with eBPF

SSTIC 2021

Guillaume Fournier
github.com/gui774ume
guillaume.fournier@datadoghq.com
I. SSH: state of the art

- Secured access to an infrastructure over an untrusted network
- Least privilege principle should be applied
- 4 security recommendations:
  1. Certificate based authentication
  2. Bastion host
  3. MFA
  4. Audit logs
I. SSH: state of the art  

II. Limitations

• But ...
  ○ MFA isn’t perfect
  ○ Access is rarely granular
  ○ SSH audit logs are limited

We’re still missing a solution to mitigate the impact of stolen credentials.

October 2019
I. SSH: state of the art  

II. Limitations  

III. ssh-probe  

IV. Demo  

- Goals:  
  - Provide real time visibility into active sessions  
  - Time, session and scope based access control  

- Security profiles per SSH user (regardless of UNIX user / sudoer):  
  - Allow (audit)  
  - MFA  
  - Block  
  - Kill session
I. SSH: state of the art  
II. Limitations  
III. ssh-probe  
IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

setlogin( ... )
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

setlogin(...)

User space

Kernel space

Uprobe

PID - Sessions
eBPF hashmaps

Uprobe hook
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

setlogin( ... )

User space

Kernel space

Uprobe

PID - Sessions eBPF hashmaps

Bash

python

curl

exec

exec

exec
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

setlogin( ... )

Uprobe hook

User space

Kernel space

Open( ... )

PID - Sessions eBPF hashmaps

Uprobe

exec

Bash

python

curl

syscall
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

setlogin( ...)
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

callsetlogin(...)

User space

Kernel space

Uprobe

PID - Sessions

eBPF hashmaps

Kprobe

Open(...)

syscall

OK

KO

exec

python

curl

LSM
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 1: eBPF hook points

OpenSSH daemon

setlogin( ... )

User space

Kernel space

Open( ... )

LSM

127 instrumented syscalls!
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 2: MFA with eBPF

Session 1

ssh-probe-auth

User space

Kernel space
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 2: MFA with eBPF

Session 1

ssh-probe-auth

syscall

User space

Kernel space

stat("otp://fim:10000@234123")

Kprobe

PID - Sessions eBPF hashmaps
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 2: MFA with eBPF
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 2: MFA with eBPF

User space

Kernel space

Session 1

ssh-probe-auth

syscall

stat("otp://fim:10000@234123")

SIGUSR2

ssh-probe daemon

MFA request

Kprobe

PID - Sessions eBPF hashmaps
I. SSH: state of the art

II. Limitations

III. ssh-probe

IV. Demo

- Technical bits 2: MFA with eBPF

```
stat("otp://fim:10000@234123")
sigusr1
```

User space

Kernel space

- Session 1
  - ssh-probe-auth
  - syscall
  - SIGUSR1
  - ssh-probe daemon
  - Temporary access granted

- MFA request
- Kprobe
- PID - Sessions eBPF hashmaps
I. SSH: state of the art  II. Limitations  III. ssh-probe  IV. Demo

Demo
Thanks!

Source code:
github.com/Gui774ume/ssh-probe