



UNIVERSIDADE
BEIRA INTERIOR

Preventing Data Exfiltration inside Virtual Machines

eBPF-based approach

Supervisor: André Passos

Orientador: Prof. Doutora Manuela Pereira

Co-Orientador: Prof. Doutor Simão Melo de Sousa

Orientando: Carlos Pinto



Problema

O problema abordado nesta tese é o da prevenção de data exfiltration.



Problema

O problema abordado nesta tese é o da prevenção de data exfiltration.

Data exfiltration pode ser caracterizado como:



Problema

O problema abordado nesta tese é o da prevenção de data exfiltration.

Data exfiltration pode ser caracterizado como:

Transferência não autorizada de dados de um computador ou outro dispositivo.



Problema

O problema abordado nesta tese é o da prevenção de data exfiltration.

Data exfiltration pode ser caracterizado como:

Transferência não autorizada de dados de um computador ou outro dispositivo.

Envolve a cópia ilícita de dados, com agência maliciosa.



Problema

O problema abordado nesta tese é o da prevenção de data exfiltration.

Data exfiltration pode ser caracterizado como:

- Transferência não autorizada de dados de um computador ou outro dispositivo.

- Envolve a cópia ilícita de dados, com agência maliciosa.

- Pode ser conduzido de forma manual ou de forma automatizada usando malware.



Problema II

A possibilidade de dados poderem ser transferidos sem autorização é particularmente gravosa para empresas, visto poderem incorrer não só em danos monetários, e de roubo de propriedade intelectual, mas também poderem ver a sua credibilidade afetada caso tal aconteça.



Solução Proposta

Desenvolver uma aplicação que faça uso de kernel-based security, com base em eBPF, de modo a prevenir data exfiltration em máquinas virtuais.



Conceitos Base



Conceitos Base

Linux Kernel

O Kernel de Linux é o componente base do sistema operativo Linux, servindo como mediador entre software e hardware.



Conceitos Base

Linux Kernel

O Kernel de Linux é o componente base do sistema operativo Linux, servindo como mediador entre software e hardware.

eBPF

eBPF é uma tecnologia que permite correr programas dentro do kernel sem modificar o mesmo.



Tarefas do Kernel



Tarefas do Kernel

- ▶ Gestão de processos.



Tarefas do Kernel

- ▶ Gestão de processos.
- ▶ Gestão de memória.



Tarefas do Kernel

- ▶ Gestão de processos.
- ▶ Gestão de memória.
- ▶ Drivers de dispositivos.

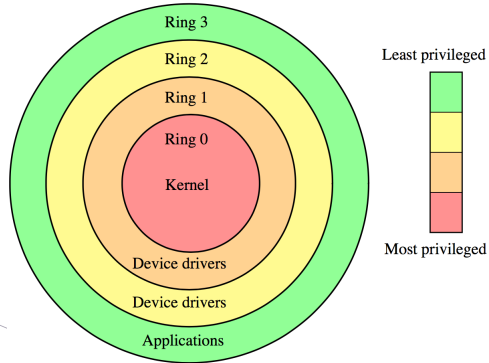


Tarefas do Kernel

- ▶ Gestão de processos.
- ▶ Gestão de memória.
- ▶ Drivers de dispositivos.
- ▶ **System Calls e Segurança.**



Kernel Space



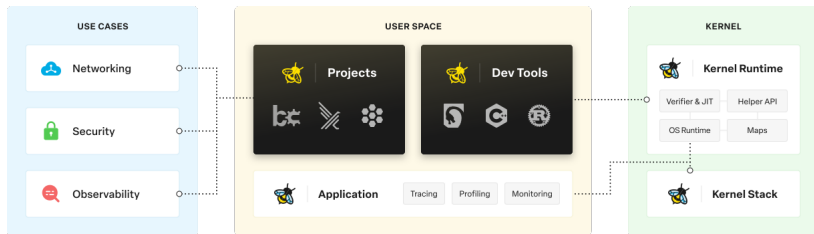


System Calls

% time	seconds	usecs/call	calls	errors	syscall
0.00	0.000000	0	5		read
0.00	0.000000	0	9		close
0.00	0.000000	0	18		mmap
0.00	0.000000	0	7		mprotect
0.00	0.000000	0	1		munmap
0.00	0.000000	0	3		brk
0.00	0.000000	0	2		ioctl
0.00	0.000000	0	4		pread64
0.00	0.000000	0	2	2	access
0.00	0.000000	0	1		execve
0.00	0.000000	0	2	2	statfs
0.00	0.000000	0	2	1	arch_prctl
0.00	0.000000	0	2		getdents64
0.00	0.000000	0	1		set_tid_address
0.00	0.000000	0	7		openat
0.00	0.000000	0	7		newfstatat
0.00	0.000000	0	1		set_robust_list
0.00	0.000000	0	1		prlimit64
0.00	0.000000	0	1		getrandom
0.00	0.000000	0	1		rseq
100.00	0.000000	0	77	5	total

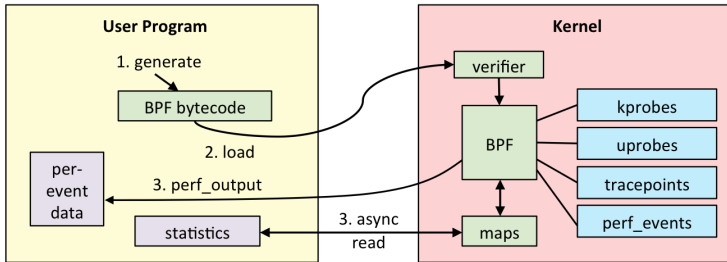


eBPF



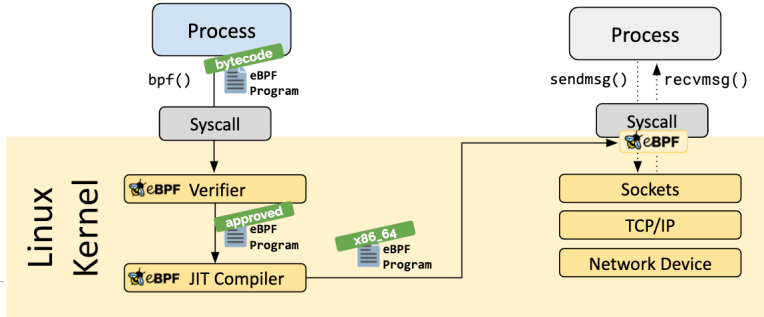


Kernel VS User Space



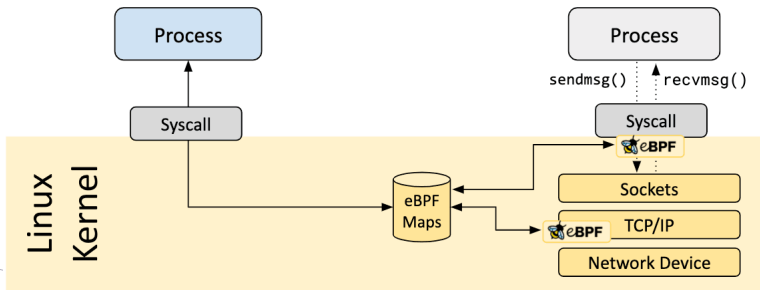


Verificação





Mapas eBPF





Estado da Arte

Ferramentas



Ferramentas

▶ Seccomp-bpf



Ferramentas

- ▶ Seccomp-bpf
- ▶ Tetragon

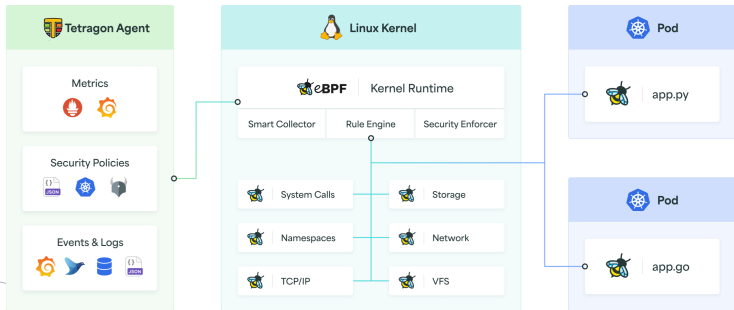


Seccomp-bpf

A ferramenta *seccomp-bpf* é uma extensão ao *seccomp* que permite filtrar as *syscalls* a partir de um programa BPF. É mais flexível que o *seccomp* original, que permitia apenas quatro *syscalls*.



Tetragon





Syscall tooling

Várias ferramentas fazem uso de *syscalls* para monitorizar e responder a eventos relevantes de um ponto de vista de segurança.



Syscall tooling

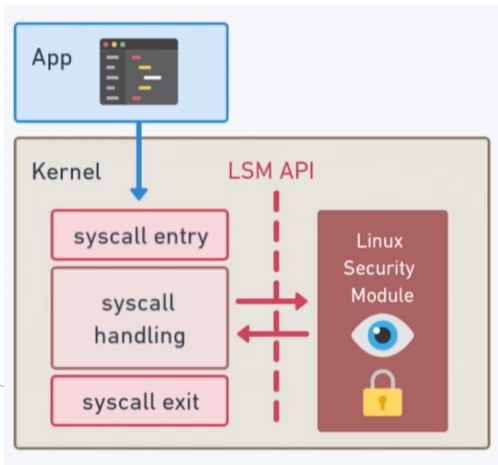
Várias ferramentas fazem uso de *syscalls* para monitorizar e responder a eventos relevantes de um ponto de vista de segurança.

TOCTOU

Estas ferramentas são vulneráveis a ataques de **TOCTOU**, visto estarem normalmente na entrada da função chamada pelo *kernel*.



BPF LSM





Demo



Trabalho futuro



Questões