Libbpf incontro 2

Windows è sottoset per linux e copre solo alcune aree come la rete.

* Passaggio dati con mappe

Recuperare informazioni del programma e dei dati che si vanno a salvare.

* Recuperare informazioni sul programma e sui dati (process id)

Componentizzare programmi BPF per farli interagire tra loro.

Es: caricare mappa user space

* Inserire input
* Inserire dati
* Stato dei programmi BPF e del sistema

Tracciare più programmi e mappa tiene i programmi tracciati, mentre altra mappa carica le roce in modo da ottimizzare l’user space

Mappe diverse per mantenere informazioni diverse su vari programmi.

Kernel windows diverso da quello di Linux.

In BPF per windows ci sono chiamate e mappe diverse.

Quando si entra in un programma il valore delle variabili riparte da 0: non rimane in memoria la modifica ad una variabile.

Mappe per informazioni salvate tra diverse informazioni.

Mappe su filesystem per tutta la vita del programma bpf.

Mappe globali nel programma generale. diverse funzioni nella sezione SEC (tipologia di funzione).

Ogni funzione è un programma a se stante e verrà eseguita dal kernel in fasi diverse in base alla sezione SEC.

* Exec
* Fileopen

Stato globale nel programma: servono condizioni per tracciare solo il programma specifico.

Altrimenti programma BPF caricato da tutti i programmi che invocano quelle syscall.

Ogni applicazione sulla macchina esegue le chiamate di funzione.

Dentro la funzione si hanno gli attributi della chiamata della funzione.

* Mappe: stato globale modificabile mantenuto tra un’esecuzione e l’altra.

Parametri modificabili, ma per avere uno stato persistente si usano le mappe.

Programma BPF salva dato nella mappa. prima di terminare chiama un altro programma per avere una richiesta di security.

Programmi che si attaccano in punti diversi a livello logico (ls open, ma prima exec).

* Durante ls si crea mappa dei processi da tracciare
* Open-> stampa file aperto

Diversi programmi comunicano.

Strace ls -> avere syscall fatte per capire ordine delle chiamate di sistema.

<https://github.com/microsoft/ebpf-for-windows/blob/main/docs/tutorial.md>

<https://nakryiko.com/>

ogni funzione con sezione SEC è un programma bpf a parte.

Carico bpf.c, ogni pezzo di una funzioen è un programma BPF tracciato un punti diversi.

Mappe sono variabili globali.

Programmi BPF nello stesso file hanno visibilità su quei programmi lì.

File BPF definisce tutti i programmi, ma il codice è scritto in diversi programmi c.

Macro SEC dice che c’è un programma che si attacca ad a…

Funzioni senza SEC si possono richiamare.

Linux ocn filesystem dedicato a BPF

* sudo ls /sys/fs/bpf/

file system in cui vengono salvate le risorse BPF.

Mappe e programmi caricati come file in quel commando.

Caricando un solo fle .bpf.c con più programmi, se caricati tutti si vedono n entry, ognuno con il nome della funzione.

* Tenere traccia programmi BPF a sistema.

Se programma cancella programmi BPF essi scompaiono dalla directory.

Guardare quando i programmi sono attivi nel sistema (dopo unload non funziona).

Oppure bpfdum che dà un dump dei programmi bpf caricati in un momento particolare specificando nomi, punti di attacco,…