Introducción

Miguel Ángel Solinas 12 may 2022
Javier Alejandro Jorge 29 may 2023
Javier Alejandro Jorge 19 may 2025

Bueno, ya conoce algo de C, ha escrito algunos programas normales para que se ejecuten como procesos, y quiere ir donde está la acción real, donde un solo puntero salvaje puede borrar su sistema de archivos y un volcado del núcleo significa un reinicio.

¿Qué es exactamente un módulo del núcleo? Los módulos son fragmentos de código que se pueden cargar y descargar en el kernel según se requiera. Extienden la funcionalidad del kernel sin necesidad de reiniciar el sistema. Por ejemplo, un tipo de módulo es el controlador de dispositivo, que permite que el núcleo acceda al hardware conectado al sistema. Sin módulos, tendríamos que construir kernels monolíticos y agregar nuevas funciones directamente en la imagen del kernel. Además de tener kernels más grandes, esto tiene la desventaja de requerir que reconstruyamos y reiniciemos el kernel cada vez que queramos una nueva funcionalidad.

Preparación

Vamos a necesitar un SO Linux instalado con sus fuentes o al menos los headers. La descarga puede demorar algunos minutos, dependiendo del BW de descarga de su conexión a internet.
Por otro lado, en esta primera parte vamos a trabajar con los siguientes programas fuentes y make files.

Unset

```
fork
https://gitlab.com/sistemas-de-computacion-unc/kenel-modules.g
it

git clone (su propia url… empieza con SU nombre de usuario)

sudo apt-get install build-essential checkinstall
kernel-package linux-source
```

Condiciones de aprobación

En el transcurso de la clase se le plantearán dos desafíos que serán evaluados en coloquios grupales con la entrega de la segunda parte del TP#4. Concretamente serán:

Desafío #1
¿Qué es checkinstall y para qué sirve?
¿Se animan a usarlo para empaquetar un hello world ?
Revisar la bibliografía para impulsar acciones que permitan mejorar la seguridad del kernel, concretamente: evitando cargar módulos que no estén firmados. rootkits ?

Desafío #2
Debe tener respuestas precisas a las siguientes preguntas y sentencias:

- ¿ Qué funciones tiene disponible un programa y un módulo ?
- Espacio de usuario o espacio del kernel.
- Espacio de datos.
- Drivers. Investigar contenido de /dev.

Bibliografía
https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/managing_monitoring_and_updating_the_kernel/signing-kernel-modules-for-secure-boot_managing-kernel-modules

https://sysprog21.github.io/lkmpg/#what-is-a-kernel-module

https://opensource.com/article/19/10/strace

https://docs.google.com/presentation/d/1BYES6Zkfx5K85REWyXsFeW-VngBLOzlDzaYCsTVoc0Y/edit#slide=id.g724a4c87a0_0_5

Pasos

cd part1
make
sudo insmod mimodulo.ko
sudo dmesg
lsmod | grep mod

```
Unset
[67375.506122] mimodulo: loading out-of-tree module taints
kernel.
[67375.506166] mimodulo: module verification failed: signature
and/or required key missing - tainting kernel
[67375.506348] Modulo cargado en el kernel.
```

sudo rmmod mimodulo
sudo dmsg
lsmod | grep mod

cat /proc/modules | grep mod

```
Unset
mimodulo 16384 0 - Live 0xffffffffc097e000 (OE)
```

modinfo mimodulo.ko
modinfo /lib/modules/$(uname -r)/kernel/crypto/des_generic.ko

1. ¿Qué diferencias se pueden observar entre los dos modinfo ?

2. ¿Qué divers/modulos estan cargados en sus propias pc? **comparar las salidas con las computadoras de cada integrante del grupo. Expliquen las diferencias. Carguen un txt con la salida de cada integrante en el repo y pongan un diff en el informe.**
3. ¿cuales no están cargados pero están disponibles? que pasa cuando el driver de un dispositivo no está disponible.
4. Correr hwinfo en una pc real con hw real y agregar la url de la información de hw en el reporte.
5. ¿Qué diferencia existe entre un módulo y un programa ?
6. ¿Cómo puede ver una lista de las llamadas al sistema que realiza un simple helloworld en c?
7. ¿Qué es un segmentation fault? ¿Cómo lo maneja el kernel y como lo hace un programa?
8. ¿Se animan a intentar firmar un módulo de kernel ? y documentar el proceso ? https://askubuntu.com/questions/770205/how-to-sign-kernel-modules-with-sign-file
9. Agregar evidencia de la compilación, carga y descarga de su propio módulo imprimiendo el nombre del equipo en los registros del kernel.
10. ¿Que pasa si mi compañero con secure boot habilitado intenta cargar un módulo firmado por mi?
11. Dada la siguiente nota https://arstechnica.com/security/2024/08/a-patch-microsoft-spent-2-years-preparing-is-making-a-mess-for-some-linux-users/
    a. ¿Cuál fue la consecuencia principal del parche de Microsoft sobre GRUB en sistemas con arranque dual (Linux y Windows)?
    b. ¿Qué implicancia tiene desactivar Secure Boot como solución al problema descrito en el artículo?
    c. ¿Cuál es el propósito principal del Secure Boot en el proceso de arranque de un sistema?