DESARROLLO DE EXPLOITS DIRIGIDOS A DRIVERS VULNERABLES EN ENTORNOS WINDOWS

RAMON GONZÁLEZ GAZTELUPE (*RAJKIT*)

UCAM Universidad Católica de Murcia

Campus Internacional Ciberseguridad

CRONOGRAMA DE EJECUCIÓN

Análisis del driver capcom.sys

Análisis de las estructuras del kernel

Creación de shellcode y exploit para el robo del token a SYSTEM

Explotación

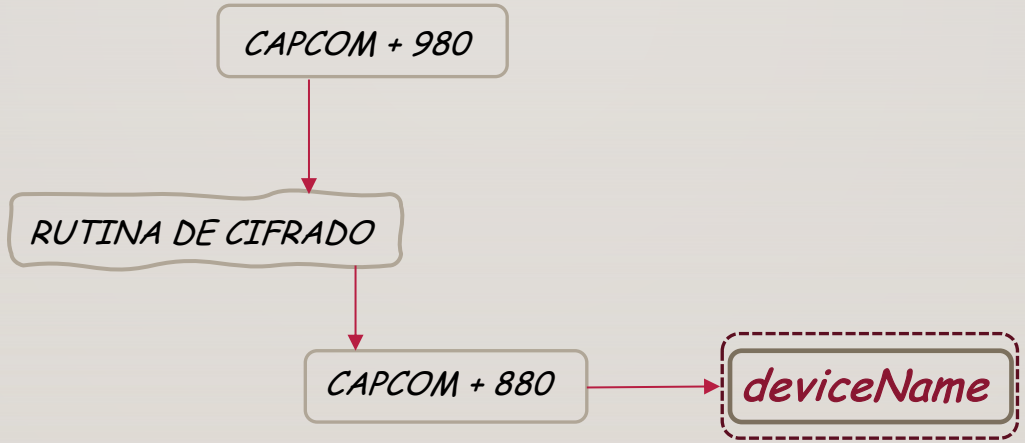
ENTORNO DE EXPLOTACIÓN

- Windows 10 – 1909 64bits
- Ensamblador NASM
- Enlazador
- Visual Studio 2022-2019
- Driver Capcom.sys
- IDA
- WinDBG preview

UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA

ANÁLISIS DEL DRIVER CAPCOM.SYS

- Necesitaremos el manejador para realizar la llamada a través de CreateFile, ese manejador esta codificado, lo obtendremos después de usar la decodificación interna que contiene el driver ya que lo almacena en una variable y se ejecuta en DriverEntry.

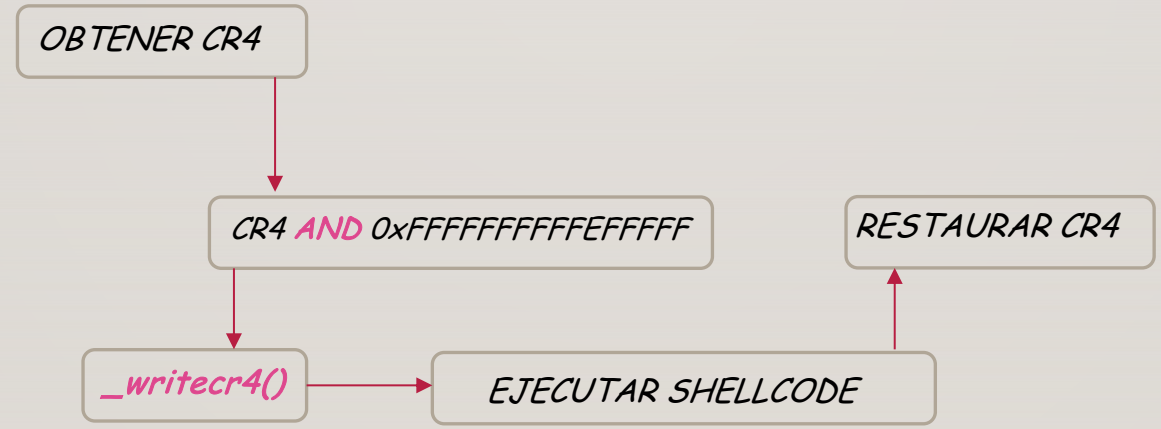


ANÁLISIS DEL DRIVER CAPCOM.SYS

- Para interactuar con el driver tendremos que obtener el IOCTL correspondiente que nos permitirá *"activar"* el envío de la dirección de memoria del buffer donde reside nuestro shellcode.

ANÁLISIS DEL DRIVER CAPCOM.SYS

- El controlador nos va a brindar la ejecución en ring 0 a través de una función que toma un puntero hacia un buffer declarado en ring 3.
- Antes de realizar la ejecución del código deshabilitara SMEP para después volver a activarlo.



QUE ES SMEP¿?

- Es una mitigación que impide que el código que se ejecuta en modo supervisor ejecute código que reside en páginas marcadas como de usuario.
- Está habilitada por el registro CR4 en su bit 20, sin embargo, no es el encargado de implementarlo.

1 -> SMEP ON

0 -> SMEP OFF

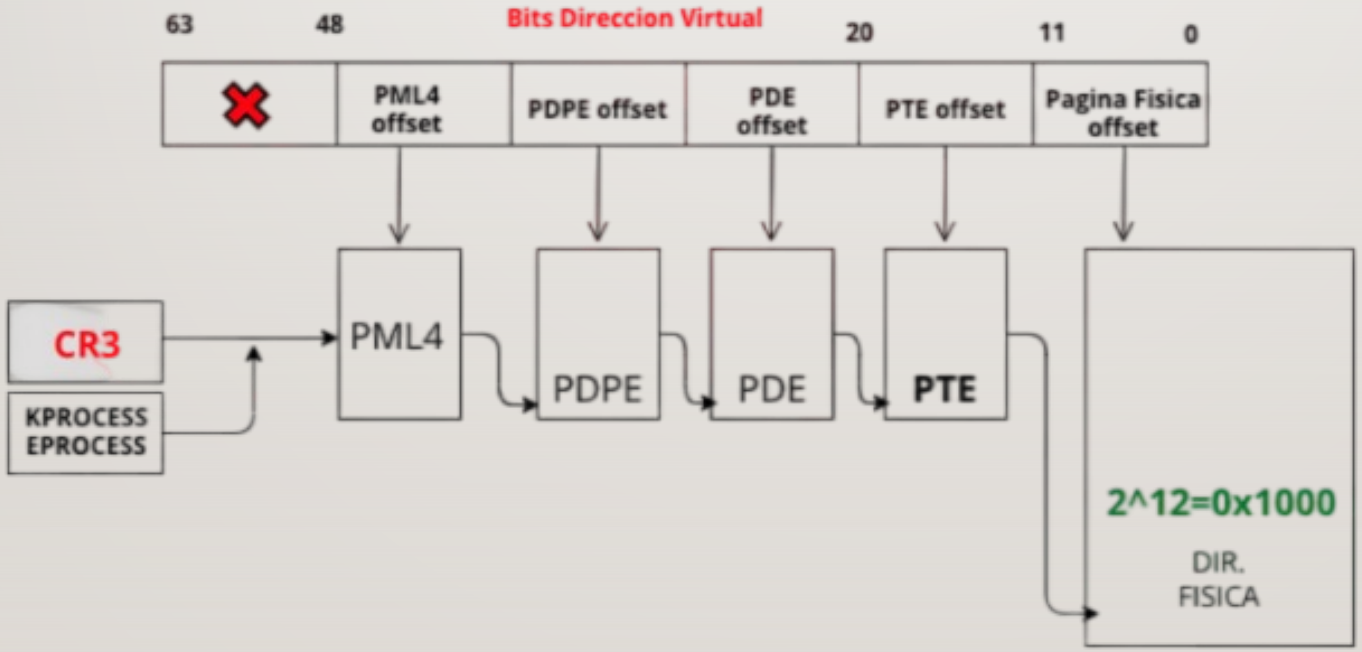
CR4 -> *0x3506F8*

00000000 00000000 00000000 00000000 00000000 001 1 0101 00000110 11111000

21

UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA

QUE ES SMEP¿?



UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA

QUE ES SMEP¿?

- SMEP se *CUMPLE* a través de la entrada de la tabla de páginas (PTE) de una página de memoria en forma de *"flags"*.
- *Recordar que una tabla de páginas es lo que contiene información sobre qué parte de la memoria física se asigna a la memoria virtual.*

!pte 0x180000 *contains* *0x0A00000012C50867*

.formats

00001010 00000000 00000000 00000000 00010010 11000101 00001000 01100 1 11

3

1 -> USER

0-> SUPERUSER

UCAM Universidad Católica de Murcia

SHELLCODE PARA EL ROBO DE TOKEN A SYSTEM

- El proceso SYSTEM, PID de 4, alberga la mayoría de los subprocesos del sistema en modo kernel.
- Los subprocesos almacenados en el proceso del SYSTEM solo se ejecutan en el contexto del modo kernel.
- Recordar que un proceso es una especie de "*contenedor*" para subprocesos.

UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA

ESTRCUTURA EPROCESS

- Un objeto es una estructura creada dinámicamente es decir en tiempo de ejecución.
- Cada objeto de proceso se conoce como EPROCESS y este contiene entre otras cosas un token de acceso, el cual va a determinar el contexto de seguridad de un hilo o un proceso.
- De tal forma que, si el proceso SYSTEM alberga la ejecución del código en modo kernel, su contexto de seguridad requerirá privilegios administrativos o de SYSTEM.

OBTENER EL VALOR DE LA LISTA ENLAZADA TOKEN DE EPROCESS SYSTEM

`!process 0 0 SYSTEM`

_EPROCESS `0xffff870b07662040`

`dt nt!_EPROCESS ffff870b07662040`

`dt nt!_EX_FAST_REF 0xffff870b07662040 + 0x4b8`

`0xffff9e0232254713`

```
+0x490 PeakVirtualSize
+0x498 VirtualSize
+0x4b0 ExceptionPortData
+0x4b0 ExceptionPortValue
+0x4b0 ExceptionPortState

+0x4b8 Token

+0x4c0 MmReserved
+0x4c8 AddressCreationLock
+0x4d8 RotateInProgress
+0x4e0 ForkInProgress
```

UCAM Universidad Católica de Murcia

CREACIÓN DE SHELLCODE PARA EL ROBO DEL TOKEN A SYSTEM

- Guardar los registros del procesador que usaremos en el stack
- Obtener EPROCESS del proceso SYSTEM y de CMD (que es donde ejecutaremos el exploit).
- Extraer el valor de token en su desplazamiento sobre EPROCESS.
- Copiarlo a nuestro proceso.
- Devolver el valor original a los registros del procesador previamente almacenados en el stack.

UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA

MITIGACIONES DE WINDOWS A LA HORA DE EXPLOTAR

- kASLR *(kernel address space layout randomization)*
- PML4 Self-Reference Entry Rasndomization
- kCFG *(kernel control Flow guard)*
- Kernel Virtual Address Shadow
- VBS/HVCI *(por defecto en Windows 11)*

UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA

CONTINUAR EN TEMAS DE EXPLOTACIÓN DEL KERNEL

- https://fuzzysecurity.com/tutorials/expDev/14.html
- *https://networkintelligence.ai/windows-kernel-exploitation/*
- *https://klue.github.io/blog/2017/09/hevd_stack_gs/*
- *https://mdanilor.github.io/posts/hevd-0/*
- *Extreme Vulnerable Driver HEVD*

UCAM
UNIVERSIDAD
CATÓLICA DE MURCIA