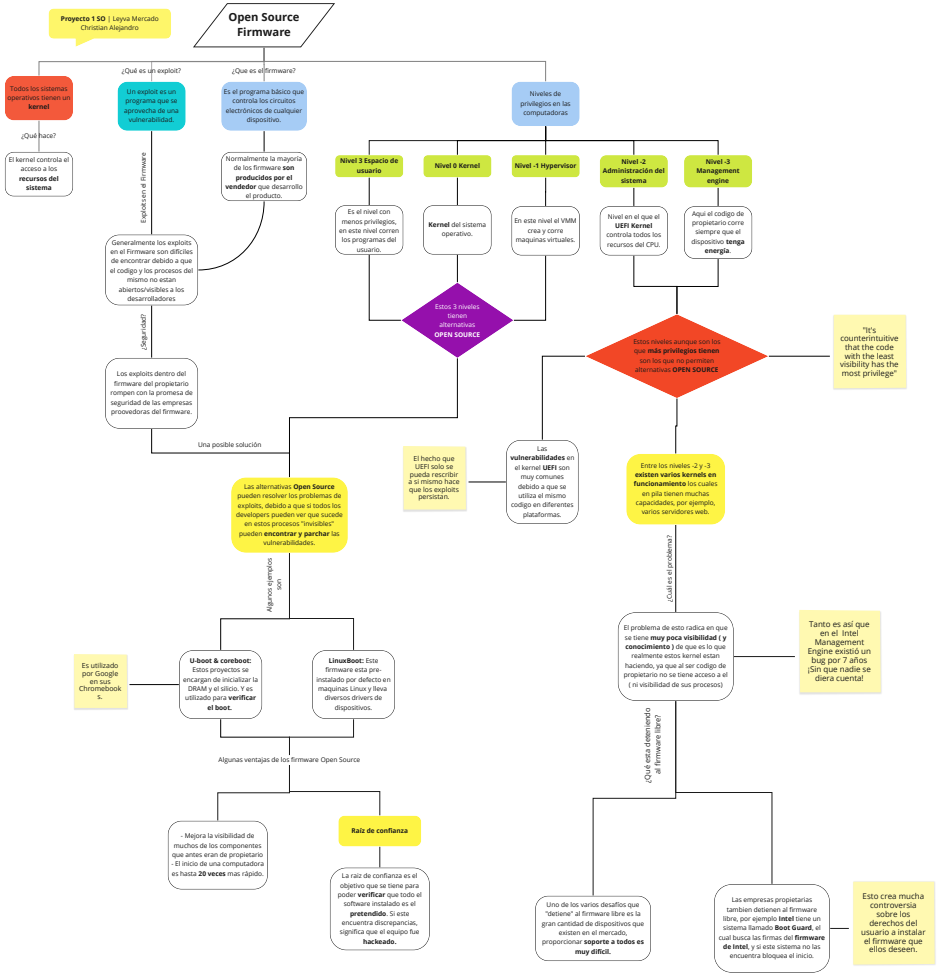
Proyecto 1 SO | Leyva Mercado Christian Alejandro
Open Source Firmware
Todos los sistemas operativos tienen un **kernel**
¿Qué hace?
El kernel controla el acceso a los **recursos del sistema**
¿Qué es un exploit?
Un exploit es un programa que se aprovecha de una vulnerabilidad.
Exploit en el Firmware
Generalmente los exploits en el Firmware son difíciles de encontrar debido a que el codigo y los procesos del mismo no estan abiertos/visibles a los desarrolladores
¿Seguridad?
Los exploits dentro del firmware del propietario rompen con la promesa de seguridad de las empresas prooveedoras del firmware.
¿Que es el firmware?
Es el programa básico que controla los circuitos electrónicos de cualquier dispositivo.
Normalmente la mayoría de los firmware **son producidos por el vendedor** que desarrollo el producto.
Niveles de privilegios en las computadoras
Nivel 3 Espacio de usuario
Es el nivel con menos privilegios, en este nivel corren los programas del usuario.
Nivel 0 Kernel
Kernel del sistema operativo.
Nivel -1 Hypervisor
En este nivel el VMM crea y corre maquinas virtuales.
Nivel -2 Administración del sistema
Nivel en el que el **UEFI Kernel** controla todos los recursos del CPU.
Nivel -3 Management engine
Aquí el codigo de propietario corre siempre que el dispositivo **tenga energía**.
Estos 3 niveles tienen alternativas **OPEN SOURCE**
Estos niveles aunque son los que **más privilegios tienen** son los que no permiten alternativas **OPEN SOURCE**
"It's counterintuitive that the code with the least visibility has the most privilege"
Las **vulnerabilidades** en el kernel **UEFI** son muy comunes debido a que se utiliza el mismo codigo en diferentes plataformas.
El hecho que UEFI solo se pueda recribir a si mismo hace que los exploits persistan.
Entre los niveles -2 y -3 **existen varios kernels en funcionamiento** los cuales en pila tienen muchas capacidades, por ejemplo, varios servidores web.
¿Cuál es el problema?
El problema de esto radica en que se tiene **muy poca visibilidad (y conocimiento)** de que es lo que realmente estos kernel estan haciendo, ya que al ser codigo de propietario no se tiene acceso a el (ni visibilidad de sus procesos)
Tanto es así que en el Intel Management Engine existió un bug por 7 años ¡Sin que nadie se diera cuenta!
¿Qué esta deteniendo al firmware libre?
Uno de los varios desafíos que "detiene" al firmware libre es la gran cantidad de dispositivos que existen en el mercado, proporcionar **soporte a todos es muy difícil**.
Las empresas propietarias tambien detienen al firmware libre, por ejemplo **Intel** tiene un sistema llamado **Boot Guard**, el cual busca las firmas del **firmware de Intel**, y si este sistema no las encuentra bloquea el inicio.
Esto crea mucha controversia sobre los derechos del usuario a instalar el firmware que ellos deseen.
Una posible solución
Las alternativas **Open Source** pueden resolver los problemas de exploits, debido a que si todos los developers pueden ver que sucede en estos procesos "invisibles" pueden **encontrar y parchar** las vulnerabilidades.
Algunos ejemplos son
U-boot & coreboot: Estos proyectos se encargan de inicializar la DRAM y el silicio. Y es utilizado para **verificar el boot**.
Es utilizado por Google en sus Chromebooks.
LinuxBoot: Este firmware esta pre-instalado por defecto en maquinas Linux y lleva diversos drivers de dispositivos.
Algunas ventajas de los firmware Open Source
- Mejora la visibilidad de muchos de los componentes que antes eran de propietario
- El inicio de una computadora es hasta **20 veces** mas rápido.
Raiz de confianza
La raiz de confianza es el objetivo que se tiene para poder **verificar** que todo el software instalado es el **pretendido**. Si este encuentra discrepancias, significa que el equipo fue **hackeado.**