**Ring -1 Hypervisor.** Este VMM (monitor de máquina virtual) crea y ejecuta máquinas virtuales.

**Anillo 0 Kernel**. Este es el kernel del sistema operativo; Los sistemas operativos de código abierto permiten la visibilidad del código detrás del kernel.

**Firmware de código abierto**

**Niveles de privilegio**

El firmware de código abierto puede ayudar a llevar la informática a un lugar más seguro al hacer que las acciones del firmware sean más visibles y menos propensas a dañar.

Las vulnerabilidades en el firmware pueden causar mucho daño debido a las muchas operaciones privilegiadas de las que el firmware es responsable.

Cada computadora o servidor generalmente viene con firmware producido por el proveedor que lo fabricó. El firmware vive en el SSD / HD.

Cuando se inicia una computadora, la interfaz principal para inicializar la DRAM, el silicio y los dispositivos es el firmware. El firmware inicializa el sistema operativo con un cargador de arranque.

Los sistemas operativos como Windows, Linux y macOS tienen kernels. El kernel controla el acceso a los recursos del sistema. Contiene la lógica para permitir que múltiples procesos compartan mecanismos de hardware como CPU, memoria, E / S de disco y redes.

**Ring 3** Espacio de usuario. Este anillo tiene la menor cantidad de privilegios. Aquí es donde se ejecutan los programas de usuario.

**Ring -2** Modo de gestión del sistema (SMM), kernel de interfaz de firmware extensible unificada (UEFI). Este es un código propietario que controla todos los recursos de la CPU (más sobre esto más adelante).

**Motor de gestión Ring -3.** Este es un código propietario que se ejecuta mientras la placa base esté recibiendo energía, incluso si está apagada (más sobre esto más adelante).

Esto significa que está disminuyendo la superficie de ataque al utilizar menos variaciones de código y está haciendo un esfuerzo por confiar en el código que es de código abierto. Linux mejora la confiabilidad del arranque al reemplazar los controladores de firmware mínimamente probados con controladores de Linux reforzados.

Maneja los controladores de dispositivos, administra la pila de red y proporciona un entorno multiusuario y multitarea. Está construido con Linux, por lo que un solo núcleo puede funcionar para varias placas.

El objetivo de la raíz de confianza debe ser verificar que el software instalado en cada componente del hardware sea el software que se pretendía. De esta manera puede saber sin lugar a dudas y verificar si el hardware ha sido pirateado.

**Raiz de confianza**

Boot Guard verifica las firmas de firmware del procesador. El problema con esto, en el caso de los procesadores Intel, es que solo Intel tiene las claves para firmar paquetes de firmware. Esto hace que sea imposible utilizar coreboot y LinuxBoot o sus equivalentes como firmware en esos procesadores. Si lo intentara, el firmware no estaría firmado con la clave de Intel y el intento fallido de arrancar bloquearía la placa.

**Protector de arranque de Intel**

**Heads.**es una configuración de coreboot que tiene un kernel de Linux configurado de forma segura como carga útil de coreboot. Funciona en servidores y portátiles.

**Uroot**. es un conjunto de herramientas de espacio de usuario y cargador de arranque de Golang. Se utiliza como initramfs para el kernel de Linux desde LinuxBoot.

Los tiempos de ejecución permiten que los sistemas utilicen firmware de código abierto y ejecuten lógica de programación personalizada.

**Tiempos de ejecución**

u-boot y coreboot son firmware de código abierto. Manejan la inicialización de silicio y DRAM. Los Chromebook de Google usan ambos: coreboot en x86 y u-boot para el resto.

**Proyectos de firmware**

**Firmware de código abierto**