PUCE Virtual

CLASE
4

Arquitectura de computadoras

Funcionamiento, riesgos de Seguridad, persistencia de malware a nivel firmware y técnicas de protección

QS WORLD UNIVERSITY RANKINGS

Educación de calidad

INTRODUCCIÓN DE LA CLASE

GRADO / POSGRADO / TECNOLOGÍAS

Estudia a tu tiempo
son interrupciones

EDUCACIÓN EN LÍNEA DE CALIDAD

La seguridad en los sistemas informáticos debe abordarse desde sus niveles más fundamentales, siendo el firmware un componente crítico en la inicialización del hardware y en el establecimiento del entorno de ejecución del sistema operativo. En este contenido se analiza su funcionamiento, así como los riesgos asociados, destacando la persistencia de malware a este nivel, capaz de evadir mecanismos tradicionales y mantenerse activo incluso tras la reinstalación del sistema. Asimismo, se estudian diversas técnicas de protección orientadas a garantizar la integridad y autenticidad del firmware.

También se centran en el arranque seguro, abordando mecanismos como Secure Boot y Trusted Boot, los cuales permiten asegurar que el sistema solo ejecute software confiable desde sus primeras etapas. En este contexto, se profundiza en los mecanismos de verificación de integridad y en la cadena de confianza, que establecen un proceso de validación secuencial desde la raíz de confianza hasta el sistema operativo. Estos contenidos permiten comprender la importancia de proteger el sistema desde su base para garantizar su seguridad global.

RDA 2: Comprender la arquitectura de la CPU, la memoria, el almacenamiento y los dispositivos de entrada y salida, así como la interacción entre el hardware y el software de una computadora.

Reto 2.

6.3 Funcionamiento, riesgos de seguridad, persistencia de malware a nivel firmware y técnicas de protección

El firmware constituye la capa más cercana al hardware dentro de la arquitectura de un sistema computacional. Se trata de un conjunto de instrucciones almacenadas en memoria no volátil (como ROM, EEPROM o flash) que controlan el comportamiento básico del hardware, incluyendo la inicialización del sistema durante el arranque. Este nivel es crítico porque actúa como intermediario entre el hardware físico y el software de alto nivel, estableciendo el estado inicial del sistema operativo (Stallings, 2019).

Desde el punto de vista de funcionamiento, el firmware ejecuta rutinas esenciales como el POST (Power-On Self-Test), la configuración de dispositivos y la carga del sistema operativo. En arquitecturas modernas, como las basadas en RISC-V, el firmware también puede incluir capas más sofisticadas como entornos de ejecución seguros y gestores de arranque avanzados (Jaramillo Villegas et al., 2022). Esto evidencia que el firmware ha evolucionado de ser un simple inicializador por convertirse en un componente clave de la seguridad del sistema.

La Figura 1 es una infografía técnica que ilustra cómo funciona el firmware en un sistema de computadora, los riesgos de seguridad asociados y las técnicas de protección.

Figura 1

Riesgos y protección en firmware

Autor: Milton Román

Riesgos de seguridad en el firmware

El firmware representa un objetivo altamente atractivo para atacantes debido a su bajo nivel de visibilidad y control. A diferencia del software tradicional, las infecciones a nivel de firmware son más difíciles de detectar y eliminar, ya que pueden persistir incluso después de formatear el sistema o reinstalar el sistema operativo.

Entre los principales riesgos destacan:

- **Acceso privilegiado**: El firmware opera con altos privilegios, lo que permite a un atacante controlar completamente el sistema.
- **Falta de monitoreo**: Muchas herramientas de seguridad no inspeccionan el firmware, lo que facilita ataques persistentes.
- **Actualizaciones inseguras**: Procesos de actualización sin autenticación pueden ser explotados para inyectar código malicioso (Martínez Amador, 2012).

Además, la complejidad creciente de los sistemas actuales ha incrementado la superficie de ataque, especialmente en entornos donde múltiples componentes de firmware interactúan (Fabra Gallo et al., 2020).

La figura 2 explica los riesgos de seguridad en el firmware de un sistema. En el centro aparece un chip de firmware, del que se desprenden flechas hacia los principales peligros: acceso privilegiado, falta de monitoreo y actualizaciones inseguras.

Figura 2

Riesgos de seguridad en el firmware

Autor: Milton Román

Persistencia de malware a nivel firmware

El malware a nivel de firmware se caracteriza por su capacidad de mantenerse oculto y sobrevivir a acciones tradicionales de limpieza. Este tipo de amenaza puede alojarse en componentes como la BIOS/UEFI (Unified Extensible Firmware Interface - Interfaz de Firmware Extensible Unificada, en español), controladores de dispositivos o incluso firmware de periféricos.

Una vez instalado, este malware puede:

- Reinfectar el sistema operativo en cada arranque.
- Manipular procesos de arranque.
- Evadir mecanismos de detección convencionales.

Este nivel de persistencia se explica porque el firmware forma parte de la base del sistema, lo que le permite actuar antes que cualquier software de seguridad (Hennessy et al., 2019). Casos reales han demostrado que este tipo de ataques puede mantenerse activo durante largos periodos sin ser detectado.

La figura 3 muestra un chip de firmware (BIOS/UEFI) infectado con malware que persiste incluso tras reinicios, afectando el arranque del sistema y evadiendo detección, con conexiones visuales hacia procesos y discos comprometidos, resaltando su invisibilidad y peligro.

Figura 3

Persistencia de Malware en Firmware: Riesgos y Evasión

Autor: Milton Román

Técnicas de protección

Las técnicas de protección del firmware no son opcionales hoy en día, sin embargo, si no se implementan correctamente, el sistema queda expuesto a compromisos profundos y persistentes.

Para mitigar los riesgos asociados al firmware, se han desarrollado diversas estrategias de seguridad:

a) Firmas digitales en firmware

La utilización de firmas digitales en el firmware es un mecanismo esencial para garantizar su autenticidad e integridad, asegurando que provenga de una fuente legítima y no haya sido modificado. Este proceso se basa en criptografía asimétrica, donde el fabricante firma el firmware con una clave privada y el sistema valida dicha firma mediante una clave pública.

Antes de ejecutar o actualizar el firmware, el sistema realiza una verificación de la firma digital; si esta no es válida, el proceso se bloquea automáticamente, evitando la ejecución de código potencialmente malicioso. Este enfoque es ampliamente aplicado en tecnologías como UEFI Secure Boot, donde solo se permite la carga de componentes firmados y autorizados. De forma similar, en equipos empresariales de fabricantes como HP o Dell, las actualizaciones de BIOS son rechazadas si se detecta cualquier alteración en los archivos.

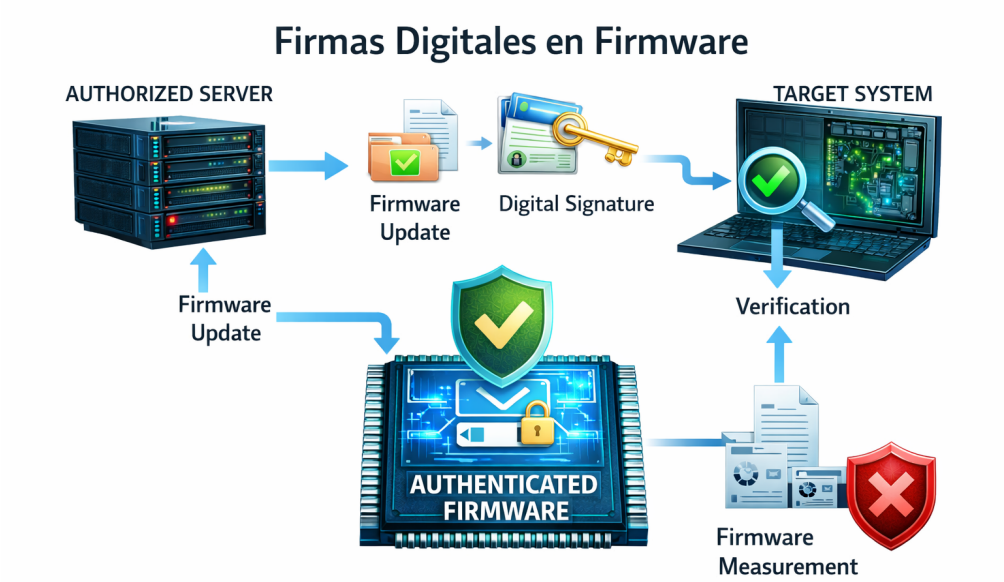
En términos de ciberseguridad, esta técnica permite mitigar riesgos críticos como la instalación de firmware malicioso y los ataques a la cadena de suministro, consolidándose como un

elemento clave para la protección de los sistemas desde sus niveles más fundamentales (Stallings, 2019; Hennessy et al., 2019).

La figura 4 muestra el proceso de protección del firmware mediante firmas digitales: un servidor autorizado genera la actualización, la firma digital garantiza su autenticidad y el sistema destino verifica el firmware antes de cargarlo, asegurando que solo software legítimo se ejecute.

Figura 4

Firmas digitales en firmware



Autor: Milton Román

b) Actualizaciones seguras de firmware

Las actualizaciones seguras de firmware constituyen un componente esencial en la protección de los sistemas, ya que procesos no controlados pueden convertirse en vectores de ataque. Para mitigarlo, se emplean mecanismos de cifrado y autenticación que garantizan la integridad y legitimidad del firmware antes de su instalación.

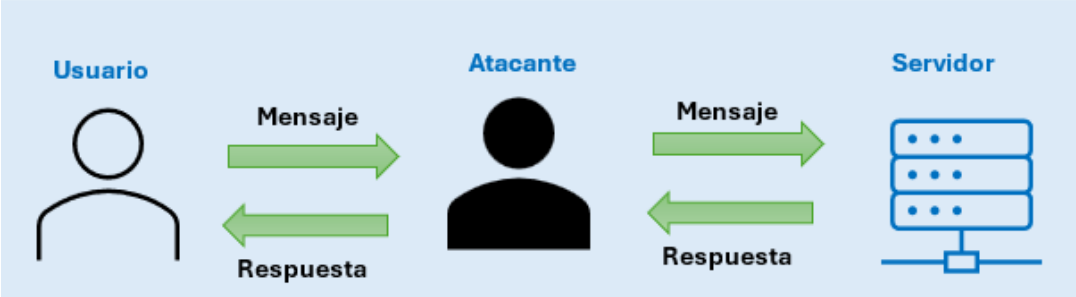
Operativamente, el firmware se obtiene mediante canales cifrados, como HTTPS, y su integridad se verifica mediante funciones hash o firmas digitales, además de validar su origen. En la práctica, dispositivos sin estos controles, como algunos routers domésticos, resultan vulnerables, mientras que equipos empresariales restringen la instalación a firmware firmado y proveniente de fuentes oficiales.

Estas medidas permiten mitigar amenazas como ataques man-in-the-middle y la distribución de firmware malicioso (Stallings, 2019).

La figura 5 muestra cómo un atacante se interpone entre un usuario y un servidor, interceptando y potencialmente modificando los datos que se envían y reciben sin que las partes lo detecten.

Figura 5

Flujo de ataque Man-in-the-Middle (MitM)



Autor: Milton Román

c) Medición de integridad (Integrity Measurement)

La medición de integridad permite detectar alteraciones en el firmware mediante la comparación de valores hash con referencias confiables. Este proceso implica generar un hash del firmware (por ejemplo, SHA-256) y contrastarlo con un valor previamente almacenado.

En sistemas que incorporan módulos TPM (Trusted Platform Module), estas mediciones se registran durante el arranque, facilitando la detección de modificaciones, como aquellas provocadas por rootkits en la BIOS. Tecnologías como Intel Boot Guard implementan este enfoque como parte de sus mecanismos de protección.

De este modo, se mitigan riesgos asociados a modificaciones no detectadas y a la persistencia de malware en niveles profundos del sistema (Hennessy et al., 2019).

d) Aislamiento de ejecución (Trusted Execution Environments - TEE)

El aislamiento de ejecución mediante entornos confiables permite proteger procesos críticos al ejecutarlos en espacios separados del sistema principal. Este enfoque limita la interacción de software comprometido con componentes sensibles.

A nivel funcional, se establece una región segura dentro del procesador donde se ejecutan funciones críticas sin interferencias externas. Tecnologías como Intel SGX y ARM TrustZone permiten proteger información sensible, como claves criptográficas, incluso en entornos comprometidos. En dispositivos móviles, este principio se aplica al manejo seguro de datos biométricos.

Esta técnica contribuye a prevenir el acceso no autorizado y el robo de información sensible (Jaramillo Villegas et al., 2022).

e) Protección contra escritura del firmware

La protección contra escritura del firmware impide modificaciones no autorizadas en componentes críticos del sistema. Este mecanismo bloquea la memoria donde reside el firmware, permitiendo su modificación únicamente bajo condiciones controladas y autenticadas.

En la práctica, algunas placas base incorporan mecanismos físicos o configuraciones que restringen la escritura en la BIOS, mientras que en entornos empresariales estas operaciones se realizan mediante interfaces seguras del fabricante.

Su implementación reduce significativamente el riesgo de infecciones directas y de escalada persistente de privilegios (Martínez Amador, 2012).

f) Monitoreo y auditoría del firmware

El monitoreo y la auditoría del firmware permiten complementar las medidas de protección mediante la supervisión continua del sistema. Este enfoque se basa en el registro y análisis de eventos relacionados con el firmware para detectar comportamientos anómalos.

En entornos corporativos, herramientas como los sistemas SIEM facilitan la identificación de cambios no autorizados, como modificaciones inesperadas en la BIOS, generando alertas para su análisis.

Estas prácticas permiten mitigar riesgos asociados a ataques persistentes y actividades maliciosas de bajo nivel (Fabra Gallo et al., 2020).

g) Uso de hardware Root of Trust

El Root of Trust se refiere a la implementación de un componente de hardware confiable que establece la base de seguridad del sistema. Dispositivos como el TPM almacenan claves criptográficas y realizan verificaciones de integridad desde las primeras etapas del arranque.

Este mecanismo permite validar el firmware antes de iniciar el sistema operativo, bloqueando la ejecución o generando alertas en caso de detectar alteraciones. De esta forma, se garantiza que el sistema opere en un estado confiable desde su inicio.

Su uso permite mitigar riesgos críticos, como el compromiso total del sistema y la ejecución de ataques invisibles al sistema operativo (Ibarra Mayorga, 2009).

En conjunto, estas técnicas establecen una raíz de confianza en el sistema, asegurando la protección de sus componentes fundamentales. En el contexto actual, la seguridad del firmware no es opcional: constituye la base sobre la cual se sostiene la integridad de todo el sistema (Hennessy et al., 2019; Stallings, 2019).

Con la intención de reforzar los conocimientos hacia los temas revisados en este documento, te invito a revisar la siguiente documentación

- **Título**: Los ataques al firmware, una amenaza creciente en tiempos de trabajo híbrido
- **Descripción**: Artículo que explica cómo los ataques de firmware constituyen riesgos de seguridad importantes y por qué son difíciles de detectar y mitigar, con ejemplos y cifras actuales.
- **Enlace**: https://www.silicon.es/ataques-firmware-amenaza-trabajo-hibrido-238019?utm_source=chatgpt.com

7. Arranque seguro (Secure Boot y Trusted Boot)

El arranque seguro es un mecanismo fundamental dentro de la arquitectura de seguridad moderna, diseñado para garantizar que el sistema se inicie únicamente con software confiable. Se basa en una cadena de confianza que comienza desde el firmware y se extiende hasta el sistema operativo.

Existen dos enfoques principales:

- **Secure Boot**: Impide la ejecución de software no firmado o no autorizado durante el arranque.
- **Trusted Boot**: Permite la ejecución, pero registra mediciones de integridad para su verificación posterior.

Ambos enfoques buscan proteger el sistema desde su fase más vulnerable: el inicio (Stallings, 2019).

7.1 Mecanismos de verificación de integridad durante el arranque y su rol en la cadena de confianza

Los mecanismos de verificación de integridad durante el arranque constituyen un pilar fundamental en la seguridad de los sistemas modernos, ya que permiten garantizar que cada componente ejecutado no ha sido alterado. Estos mecanismos operan dentro de un modelo estructurado conocido como cadena de confianza, en el cual cada elemento valida al siguiente antes de cederle el control.

Mecanismos de verificación de integridad

La verificación de integridad se basa en la utilización de técnicas criptográficas que permiten comprobar la autenticidad y el estado inalterado de los componentes del sistema. Entre los principales mecanismos se destacan:

- **Firmas digitales:** Cada componente crítico del proceso de arranque, como el firmware, el gestor de arranque (bootloader) y el núcleo del sistema operativo, debe estar firmado digitalmente. La verificación de estas firmas garantiza que el código proviene de una fuente confiable y no ha sido modificado.
- **Funciones hash criptográficas:** Se generan valores hash únicos de los componentes del sistema, los cuales son comparados con valores previamente almacenados. Cualquier discrepancia indica una posible alteración.
- **Medición de integridad (Measured Boot):** En este enfoque, cada etapa del arranque calcula y registra el hash del siguiente componente antes de ejecutarlo. Estas mediciones son almacenadas en módulos seguros, como el TPM, permitiendo su posterior auditoría.
- **Módulos de plataforma confiable (TPM):** Actúan como elementos de hardware seguros que almacenan claves criptográficas y registros de integridad. El TPM permite verificar que el sistema ha arrancado en un estado confiable y facilita la detección de manipulaciones.

En conjunto, estos mecanismos permiten establecer un control riguroso desde las primeras etapas de ejecución, reduciendo significativamente la probabilidad de que código malicioso sea cargado en el sistema (Stallings, 2019).

Cadena de confianza

La cadena de confianza es un modelo de seguridad que organiza el proceso de arranque en una secuencia jerárquica de validaciones. Su principio fundamental radica en que ningún componente puede ejecutarse sin haber sido previamente verificado por un elemento confiable anterior.

Este proceso se estructura en varias etapas:

a) **Raíz de confianza (Root of Trust)**: Constituye el punto inicial del sistema y generalmente está implementada en hardware o firmware inmutable. Este componente es inherentemente confiable y actúa como base para todas las verificaciones posteriores.

b) **Verificación del firmware**: La raíz de confianza valida la integridad del firmware antes de su ejecución.

c) **Verificación del gestor de arranque**: El firmware, una vez validado, verifica el bootloader mediante firmas digitales o hashes.

d) **Verificación del sistema operativo**: El bootloader valida el kernel del sistema operativo, asegurando su legitimidad.

e) **Extensión de confianza a aplicaciones**: En algunos modelos avanzados, el sistema operativo continúa verificando la integridad de controladores y aplicaciones críticas.

Este encadenamiento de validaciones asegura que la confianza se propague de manera controlada desde el nivel más bajo hasta los niveles superiores del sistema (Hennessy et al., 2019).

La figura 6 muestra cómo la confianza se transmite desde la raíz de confianza en hardware hasta el firmware, bootloader, sistema operativo y aplicaciones, asegurando que cada componente se verifique antes de ejecutarse.

Figura 6

Cadena de Confianza (Secure Boot / Trusted Chain)

Autor: Milton Román

Importancia en la ciberseguridad

La correcta implementación de mecanismos de verificación y de la cadena de confianza permite prevenir ataques sofisticados que buscan comprometer el sistema desde sus etapas iniciales. Entre las amenazas mitigadas se incluyen:

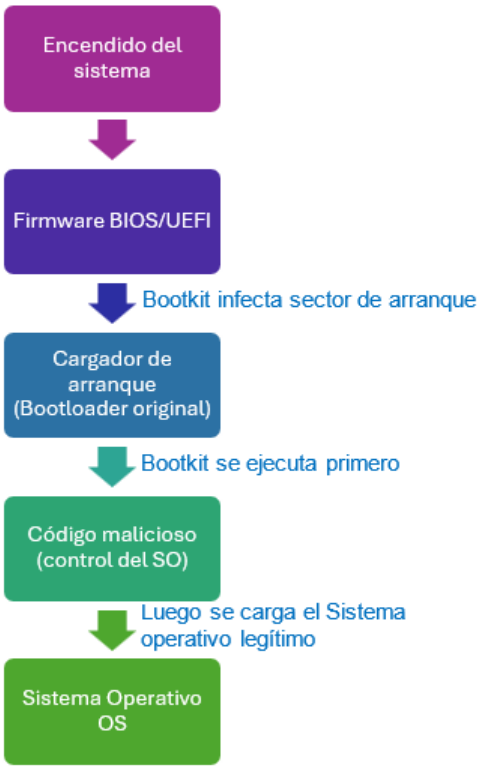
- Rootkits de arranque (bootkits).
- Malware persistente en firmware.
- Modificaciones no autorizadas del kernel.
- Ataques dirigidos a la cadena de suministro.

La forma de cómo un bootkit se infiltra en el proceso de arranque, ejecutando código malicioso antes que el sistema operativo y manteniendo control invisible sobre el equipo se muestra en la figura 7.

Figura 7

Funcionamiento de un Bootkit: Rootkit de Arranque

Autor: Milton Román



Además, estos mecanismos permiten garantizar que el sistema se inicie en un estado conocido y confiable, lo cual es especialmente relevante en entornos críticos, como infraestructuras empresariales, sistemas financieros y dispositivos de uso gubernamental (Fabra Gallo et al., 2020).

En síntesis, tanto la seguridad del firmware como los mecanismos de arranque seguro constituyen pilares fundamentales en la protección de sistemas modernos. Ignorar estas capas equivale a dejar abierta la puerta principal de todo el sistema.

Te propongo ingreses al siguiente enlace para observar una práctica de como habilitar el secure boot en UEFI/BIOS.

- **Título:** Cómo habilitar/deshabilitar Secure Boot en UEFI/BIOS 2025. Configurar arranque seguro.

- **Descripción**: Este video muestra en pasos concretos cómo configurar el arranque seguro desde la BIOS/UEFI, lo cual te ayuda a entender qué hace Secure Boot y cómo actúa verificando firmas de componentes antes de iniciar el sistema operativo, esta es una parte clave de la cadena de confianza.
- **Enlace**: https://youtu.be/hUwN9wG8f1A

Educación de calidad

REFERENCIAS

GRADO / POSGRADO / TECNOLOGÍAS

- **Fabra Gallo, J. M., Martínez Márquez, Y., & Valcárcel Izquierdo, N. (2020).** Estudio comparado de los contenidos de la asignatura Arquitectura de Computadoras: oportunidades de mejoras. Dialnet. Obtenido de https://dialnet.unirioja.es/servlet/articulo?codigo=8590371
- **Hennessy, Hennessy, J. L., & Patterson, D. A. (2019).** ARQUITECTURA DE COMPUTADORES: Un enfoque cuantitativo. McGrawHill. https://doi.org/1-55860-069-8
- **Ibarra Mayorga, M. F. (2009).** Evolución de las arquitecturas de las computadoras. eLibro. Obtenido de https://elibro.puce.elogim.com/es/lc/puce/titulos/28671
- **Jaramillo Villegas, J. A., Zuluaga Bucheli, H. M., & Sepúlveda Caviedes, C. (2022).** Arquitectura de Computadoras con RISC-V. Universidad Tecnológica de Pereira. https://doi.org/9789587227956
- **Martínez Amador, H. (2012).** Arquitectura de computadoras: basado en competencias para nivel superior. Obtenido de https://elibro.puce.elogim.com/es/lc/puce/titulos/130397
- **Stallings, W. (2019)**. Computer organization and architecture : designing for performance (11ava ed.). Pearson Education. Obtenido de https://os.ecci.ucr.ac.cr/ci0114/material/Stallings/Computer-Organization-Architecture-11th.pdf

PUCE Virtual

síguenos en:

www.pucevirtual.puce.edu.ec