Enarx: un proyecto líder de código abierto en computación confidencial

# Resumen

Las organizaciones que operan con código o datos confidenciales necesitan asegurar su integridad y confidencialidad al ejecutar sus aplicaciones, especialmente cuando se despliega en la nube pública o en Edge. Enarx es un proyecto de código abierto que utiliza TEE (Trusted Execution Environment), o entornos de ejecución seguros, que permite a las organizaciones ejecutar aplicaciones y procesar datos confidenciales en sistemas no confiables.

# Introduction

Las organizaciones de diferentes sectores lanzan sus cargas de trabajo en múltiples entornos de ejecución, desde sus propias instalaciones hasta la nube pública y Edge. Y requieren la mayor garantía posible de que su código y datos confidenciales estén protegidos. Este asunto es especialmente importante para sectores como banca y finanzas, sector público y administrativo, telecomunicaciones, Internet de las cosas (IoT), sanidad (HIPAA, la norma de confidencialidad de la Ley de Portabilidad y Responsabilidad de Seguros Médicos en EEUU), datos del cliente (RGPD, Reglamento general de protección de datos), funciones empresariales de carácter confidencial y derechos humanos.

Hay tres estados en los que se pueden proteger los datos: en almacenamiento, en tránsito y en uso. Los datos en almacenamiento son archivos y objetos guardados. Los datos en tránsito son aquellos que están viajando de un sitio a otro a través de Internet o por una red privada. Los datos en uso son los que se están procesando en la CPU o RAM. Cifrar los datos en almacenamiento y en tránsito ya se ha convertido en una norma en la nube, mientras que la necesidad de proteger los datos en uso sólo está empezando a surgir.

Cuando una carga de trabajo se ejecuta en la nube pública, bien como una máquina virtual o un contenedor, puede ser manipulada por cualquier persona o software con acceso al host. Incluso si el proveedor de la nube tiene políticas de seguridad estrictas, la carga de trabajo sigue siendo vulnerable en caso de que el propio sistema del host esté comprometido: su sistema operativo, las librerías de firmware, el hipervisor, la pila de aplicaciones, las librerías de terceros, el middleware y los drivers.

Los entornos de ejecución seguros (TEE) proporcionan una solución basada en hardware para esta necesidad de mantener la confidencialidad y la integridad de los datos en uso, independientemente de quién sea el propietario o tenga acceso al sistema host en el que se ejecuta la aplicación.

# Contexto

Los entornos de ejecución seguros (TEE) permiten a las organizaciones ejecutar sus aplicaciones dentro de un conjunto de páginas de memoria encriptadas con una clave secreta por la CPU del host de tal manera que estas páginas no son accesibles para el sistema operativo o cualquier otro software, incluso con el nivel más alto de privilegios.

Actualmente hay dos principales modelos de TEE:

* *Basado en proceso*: en este modelo, un proceso se divide en dos componentes: confiable y no confiable. El componente confiable reside en la memoria cifrada y maneja la computación confidencial, mientras que el componente no confiable interactúa con el sistema operativo y comunica por E/S la memoria cifrada con el resto del sistema. Los datos solo pueden entrar y salir de esta zona cifrada por unos canales determinados, que tienen controles estrictos de tamaño y del tipo de datos que pueden pasar. Uno de los ejemplos de implementación actual de este modelo es SGX (Software Guard eXtensions) de Intel.
* *Basado en máquina virtual*: en este modelo la memoria cifrada se delimita como una VM tradicional, gestionada por el VMM. Las VM tradicionales (y los contenedores) ofrecen cierta medida de aislamiento, pero en este tipo de TEE las VM están protegidas por claves de cifrado basadas en hardware, evitando así la interferencia de un VMM malicioso. Este modelo de TEE está empleado en SEV (Secure Encrypted Virtualization) de AMD.

Las aplicaciones que se van a ejecutar en un TEE deben desarrollarse específicamente para cada plataforma, ya que la arquitectura depende del tipo del TEE: basado en procesos o en VM. Además, deben implementar lo que se llama “atestado”, un proceso de validación cuyo objetivo es confirmar que el TEE que se va a usar por la aplicación sea auténtico y seguro. Reescribir la aplicación o el VMM personalizado que lo ejecuta, hacer el atestado para cada plataforma de hardware es extremadamente complejo y requiere mucho tiempo.

En la siguiente sección presentaremos Enarx, un framework de código abierto para ejecutar aplicaciones en TEE y explicaremos cómo resuelve los problemas planteados. Ofreceremos una descripción general simplificada de la arquitectura de componentes de Enarx (y su adaptabilidad a múltiples plataformas de hardware). Explicaremos cómo se crean las aplicaciones y se implementan en las instancias de TEE usando Enarx.

# Enarx

Enarx es un framework que ejecuta aplicaciones en lo que llamamos “Keeps”, instancias de TEE, sin la necesidad de implementar los atestados de forma separada, ni de reescribir la aplicación, ni de confiar en muchas dependencias. Proporciona un runtime de WebAssembly, basado en wasmtime. Está diseñado para funcionar en diferentes arquitecturas del procesador de forma transparente para el usuario, de modo que la aplicación pueda ejecutarse con facilidad tanto en plataformas Intel (SGX o el recientemente anunciado TDX), como plataformas AMD (SEV) o plataformas futuras como Arms 'Realms y PEF de IBM, sin necesidad de recompilar el código de la aplicación.

El objetivo de Enarx es minimizar las relaciones de confianza al ejecutar aplicaciones, así que los únicos componentes en los que habría que confiar son: la CPU y el firmware asociado, la carga de trabajo en sí y el middleware de Enarx, que es de código totalmente abierto y auditable. Las aplicaciones se ejecutan sin que ninguna de las capas de la pila (hipervisor, kernel o espacio de usuario) pueda acceder al Keep, ni alterar su contenido.

Enarx permite que el atestado, el empaquetado y el aprovisionamiento de la aplicación sean transparentes para el usuario. Cada instancia de la aplicación pasa por tres etapas:

1. *Atestado*: Enarx verifica que el host en el que se va a desplegar la aplicación sea una instancia auténtica de TEE.
2. *Empaquetado*: una vez que se completa el atestado y se verifica la instancia de TEE, el componente de gestión de Enarx encripta la aplicación, junto con el resto de los datos necesarios.
3. *Aprovisionamiento*: Enarx luego envía la aplicación y los datos al host para su ejecución en Enarx Keep. En ningún momento el sistema host puede acceder a la aplicación o modificar los datos.

Enarx gestiona el proceso del atestado y el aprovisionamiento en un “Keep” runtime, basado en WebAssembly, lo que permite a los desarrolladores usar una amplia gama de lenguajes. El runtime de Keep soporta aplicaciones escritas en Rust, C, C ++, C #, Go, Java, Python y Haskell. Enarx no depende de la arquitectura de la CPU, lo que permite implementar el mismo código de aplicación en distintas plataformas, evitando problemas como la compilación cruzada o los diferentes mecanismos de atestado, dependiendo del proveedor de hardware.

# Conclusión

Enarx es un proyecto de código abierto que utiliza TEE (entornos de ejecución de confianza) para ejecutar aplicaciones dentro de "Keeps" en sistemas que no son de confianza. Enarx gestiona la creación de estos Keeps, comprueba que la CPU que se va a usar sea válida, luego encripta y pasa las aplicaciones y los datos al Keep usando claves criptográficas de un solo uso. Las aplicaciones se ejecutan sin que ninguna de las capas de la pila (por ejemplo, hipervisor, kernel, espacio de usuario, middleware) pueda acceder al Keep. Enarx permite ejecutar aplicaciones en TEE de varios fabricantes de CPU sin tener que preocuparse por la portabilidad, gestiona el atestado y el despliegue.