Checkmarx

LIBRO BLANCO

Creación de aplicaciones seguras: Recomendaciones para los servicios financieros

Introducción

- Requisitos normativos para los servicios financieros 3
- El cambiante mundo del desarrollo de software y aplicaciones 4
  - El cambio desde las metodologías de desarrollo de software en cascada 4
  - El origen de las implementaciones DevOps 4
  - El ascenso del código abierto 5
- Las pruebas de seguridad deben adaptarse al nuevo paradigma 6
- Cinco pasos para crear unas aplicaciones más seguras 7
  - 1. Aplicaciones de clasificación del riesgo 7
  - 2. Establecer unos requisitos de seguridad claros 8
  - 3. Identificar las vulnerabilidades durante todo el SDLC 8
  - 4. Capacitar a los desarrolladores para que codifiquen de forma segura desde el principio 10
  - 5. Recuerde que la seguridad de las aplicaciones es un proceso que no tiene fin 10
- Conclusión 12

Requisitos normativos para los servicios financieros

Las organizaciones de servicios financieros operan según determinadas normas reglamentarias. Es algo lógico, ya que los activos y la información que gestionan estas empresas son valiosos, sensibles y pueden ser objeto de sofisticados ciberataques diarios. Estos desafíos se ven agravados por el gran volumen de información de identificación personal (IIP) que las organizaciones financieras gestionan regularmente.

La IIP está sujeta a muchas regulaciones y estándares, en concreto Graham, Leach, Bliley (GLBA), el Estándar de Seguridad de Datos de la Industria de Tarjetas de Pago (PCI-DSS), y la ley Sarbanes Oxley (SOX). Hoy en día, el Reglamento General de Protección de Datos (RGPD) es también una regulación prioritaria, ya que regula no solo el tratamiento de los datos personales que incluyen IIP de los ciudadanos de la Unión Europea, sino también el tratamiento por parte de cualquier organización que procese datos personales de residentes en la UE. Para los consumidores de la banca en Estados Unidos, la Sección 5 (actos o prácticas desleales o fraudulentos) de la Ley de la Comisión Federal de Comercio y numerosas regulaciones estatales refuerzan las medidas básicas de protección al consumidor, que las organizaciones financieras también deben respetar.

El panorama normativo de las organizaciones de servicios financieros continúa expandiéndose. La intención, por supuesto, es proteger las inversiones de los clientes y la información sensible, así como prevenir el fraude. Las organizaciones sujetas a este reglamento y las normas deben sortear una gran variedad de estrictos requisitos.

Algunos son bastante explícitos; el PCI-DSS incluye actividades específicas que las organizaciones deben realizar (por ejemplo, el uso de manuales o procesos automatizados para identificar vulnerabilidades comunes como la inyección SQL, la falsificación cruzada de solicitudes entre páginas, la secuencia cruzada de comandos de páginas y los desbordamientos de búfer, entre otros). Otras regulaciones son bastante generales, simplemente establecen que la IIP debe protegerse contra los ataques.

Si bien las diferentes regulaciones y normas emplean diferentes enfoques, requieren el mismo enfoque en materia de seguridad. En especial, para cumplir con cualquiera de las normas principales, las organizaciones deben tener visibilidad de los riesgos y las vulnerabilidades de su software y sus sistemas (realizando evaluaciones periódicas de la vulnerabilidad) y planificar para abordar las vulnerabilidades (estableciendo y realizando un seguimiento del plan de administración de las vulnerabilidades).

Las evaluaciones de vulnerabilidades pueden realizarse de varias maneras. Por ejemplo, existen soluciones comerciales para el escaneo de entornos para identificar los entornos no parcheados o aplicaciones o sistemas mal configurados. Aunque son útiles, estas soluciones se centran en el software comercial y sistemas operativos como Linux y Windows. Estos tipos de escaneos no se aplican a las miles de aplicaciones internas que crean y mantienen la mayoría de las organizaciones de servicios financieros. Las aplicaciones internas requieren require un conjunto diferente de habilidades y soluciones para identificar y abordar las vulnerabilidades.

El cambiante mundo del desarrollo de software y aplicaciones

El desarrollo de software ha cambiado drásticamente durante los últimos años de tres maneras significativas, lo que ha obligado a cambiar también la seguridad de las aplicaciones.

El cambio desde las metodologías de desarrollo de software en cascada

En el pasado, la mayoría de los proyectos de desarrollo de software utilizaban un modelo de desarrollo en cascada. En este modelo, los equipos seguían un proceso estructurado de recopilación de requisitos, diseño de software, implementación/codificación específica de una versión, verificación/prueba de ese lanzamiento, y mantenimiento continuo. La fase de desarrollo puede durar meses, con dos o tres grandes lanzamientos cada año. Las pruebas de seguridad de aplicaciones, si se realizan, se llevan a cabo mediante equipos de seguridad autónomos en una fase posterior de la etapa de desarrollo.

Si bien todavía se utilizan para algunas aplicaciones heredadas más grandes, las organizaciones ahora ven el desarrollo ágil de software como una forma de obtener una ventaja competitiva, y cada vez más, como un requisito de su negocio.

Estas metodologías de desarrollo de software más rápidas, como el desarrollo ágil, se centran en equipos integrados que incluyen a los arquitectos de software, los desarrolladores y los equipos de pruebas funcionales y de seguridad que trabajan juntos para ofrecer funciones prácticas lo más rápido posible. Escuchando atentamente a los clientes e implementando rápidamente los nuevos requisitos descubiertos, las organizaciones pueden ganar cuota de mercado frente a competidores más lentos.

El origen de las implementaciones DevOps

El uso de procesos de desarrollo ágiles puede ayudar a las organizaciones a construir y ofrecer el software más rápidamente. La adición de conceptos de DevOps como la integración continua y la entrega continua (CI/CD, por sus siglas en inglés) en un entorno ágil ayuda a descomponer los silos integrando el desarrollo de software y las operaciones de software, aumentar la calidad y eficiencia y poner a disposición de los usuarios los cambios graduales más rápidamente. La integración continua (CI) se refiere al proceso por el cual el nuevo código se integra en la base de códigos principal con un nivel mínimo de retrasos. La entrega continua (CD) es una práctica de de desarrollo de software en la que cada cambio de código pasa por todo el proyecto y se pone a disposición de los usuarios finales rápidamente.

Las claves para una adecuada migración de DevOps requieren el uso de herramientas apropiadas y automatización. Aunque las herramientas pueden ser diferentes en cada organización, un proyecto de integración continua (CI) como Jenkins o TeamCity es fundamental para automatizar con éxito los procesos. Por ejemplo, un proyecto de CI permite a las organizaciones automatizar las pruebas de seguridad durante el proceso de creación y también aprovechar las pruebas funcionales existentes con la automatización de la seguridad durante el proceso de CD.

El ascenso del código abierto

Hace años, los equipos internos de desarrollo de software crearon software personalizado de forma integral. Los componentes de código abierto se consideraban peligrosos y rara vez se utilizaban debido a su origen desconocido y modelos de licencia inusuales. A medida que aumentó el uso del código abierto, también lo hizo la aceptación de la ya conocida comunidad de proyectos de código abierto como Linux, OpenSSL y los marcos de trabajo como Apache Struts. Hoy en día, el software de código abierto a menudo comprende la mayoría de las bases de datos de códigos, tanto en el software interno como en el comercial.

Los componentes y marcos de trabajo de código abierto ofrecen muchos beneficios, como la eliminación de la necesidad de escribir funciones comunes de cero, reduciendo así los costes de desarrollo y acelerando el tiempo de comercialización. Pero las implementaciones de código abierto también presentan nuevos riesgos para una organización. Cada año, se descubren más de 3000 nuevas vulnerabilidades en el código abierto. A menudo, los puntos de explotación de estas vulnerabilidades están disponibles al público a los pocos días de su divulgación, lo que proporciona a los ciberatacantes un vector de ataque simple y sin mucho esfuerzo. En 2017, este riesgo se demostró dramáticamente en una filtración de alto perfil, cuando se explotó una vulnerabilidad revelada públicamente en Apache Struts. Los hackers robaron IIP de más de 148 millones de consumidores, resultando en una pérdida de 6000 millones de dólares para la capitalización del mercado de la organización, y el despido de el director general, el director de información y director de seguridad de la empresa.

10

Beneficios de DevOps para las organizaciones de servicios financieros

1. Reduce el plazo de salida al mercado
2. Automatiza los procesos manuales y más laboriosos
3. Fomenta la colaboración entre los equipos
4. Aumenta la eficiencia operativa
5. Mejora el rendimiento general
6. Agiliza el cumplimiento normativo y simplifica las auditorías
7. Reduce los costes de desarrollo y de infraestructura informática
8. Aumenta la calidad
9. Mejora la experiencia del cliente
10. Reduce los periodos de inactividad, los fallos y los rollbacks

Las pruebas de seguridad deben adaptarse al nuevo paradigma

Este nuevo paradigma de desarrollo de software requiere que los equipos de seguridad se adapten, ya que los retrasos causados por los procesos «fuera de banda», o nuevos/diferentes,romperán de manera efectiva con el modelo de DevOps. Esto comienza con la definición clara y la identificación de los errores de creación y la mitigación de las amenazas de manera directa en los procesos diarios, no al final del proceso. Las pruebas funcionales y de seguridad también deben ejecutarse automáticamente como parte del proceso. La velocidad de DevOps no puede tolerar ciclos de prueba independientes. Y es importante recordar que cuando se trata de DevOps, la seguridad basada en la experiencia y las herramientas son tan importantes como la orquestación de CI o las pruebas funcionales.

También se requiere una visibilidad completa de los componentes de código abierto en el software desarrollado internamente para asegurar eficazmente el ciclo de vida del desarrollo de software moderno (SDLC). Las soluciones de Prueba de Seguridad de Aplicaciones (AST, por sus siglas en inglés), que incluyen análisis estáticos e interactivos, son excepcionales en la identificación de errores de codificación que pueden resultar en vulnerabilidades de seguridad en el código personalizado. Lamentablemente, las soluciones de análisis estático e interactivo son ineficaces en la identificación de las vulnerabilidades previamente reveladas en el software de código abierto. Para abordar la parte de código abierto de la base de código, se requieren soluciones de composición de software y análisis de código abierto. Estas soluciones analizan el software para generar una lista de todos los componentes de código abierto en uso, y luego asignan esos componentes a bases de datos de vulnerabilidades para identificar los componentes con vulnerabilidades conocidas.

Cinco pasos para crear unas aplicaciones más seguras

Aunque las organizaciones de servicios financieros se encuentran bajo ataque constante de sus adversarios, existe una serie de medidas específicas que pueden tomarse para abordar la seguridad en el software que crean.

1. Aplicaciones de clasificación del riesgo

Muchas organizaciones de servicios financieros tienen equipos de desarrollo de software que superan a los de las grandes empresas de software comercial. De hecho, algunas organizaciones bancarias globales tienen más de 20000 ingenieros de software creando y gestionando miles de aplicaciones individuales. Asegurar el ciclo de vida del desarrollo de software moderno es un aspecto extremadamente desafiante para las organizaciones financieras actuales. Con una superficie de ciberataque en constante expansión, los equipos de seguridad a menudo sienten que necesitan «evaporar el océano» por así decirlo. Pero desde la perspectiva del riesgo empresarial, no todas las aplicaciones son iguales.

El primer paso para reducir el riesgo es cuantificar el riesgo inherente asociado a cada aplicación. Esto se puede lograr mediante la utilización de una metodología de riesgo prioritario para clasificar las aplicaciones la utilización de una metodología de riesgo prioritario para clasificar las aplicaciones según el riesgo potencial de daño a los objetivos comerciales de la empresa como resultado de un ataque. Por ejemplo, la seguridad de una aplicación de banca en línea que permite a los clientes transferir fondos, realizar grandes transacciones y cambiar los privilegios es crucial para los objetivos comerciales de un banco. Una filtración en esa aplicación podría causar daños financieros, regulatorios y de reputación al banco. Asimismo, las aplicaciones que gestionan la información sujeta a los estándares de seguridad como el PCI-DSS se consideran críticas y deben ser seguras.

Sin embargo, hay aplicaciones internas que no procesan información sensible o tienen una superficie de ataque limitada. En términos de valor empresarial, estas aplicaciones son menos críticas y no garantizan el mismo escrutinio desde el punto de vista de la seguridad.

Las aplicaciones de clasificación del riesgo permiten-a los equipos de seguridad con limitaciones de tiempo y recursos dentro de las organizaciones de servicios financieros aplicar los recursos adecuados a las aplicaciones con más riesgo, a la vez que maximizan la eficiencia operativa.

2. Establecer unos requisitos de seguridad claros

Para lograr verdaderos «DevSecOps», los desarrolladores, los equipos de seguridad y los equipos de operaciones deben acordar de antemano las métricas para obtener un nivel adecuado de seguridad. Esto requiere un sistema abierto y continuo de comunicación y colaboración entre equipos. Las métricas serán diferentes para los distintos tipos de aplicación, en función de la clasificación de riesgos y el interés exclusivo de la organización por el riesgo.

Para los componentes de código abierto, estos requisitos deben incluir una comprensión de cada proyecto, incluyendo:

- Hasta qué punto un proyecto dispone de respaldo por parte de la comunidad. Un proyecto de código abierto sin respaldo puede resultar en una gestión y mantenimiento continuos para el equipo interno de operaciones de TI.
- El historial de seguridad del componente. Es importante repasar cuántas vulnerabilidades se han revelado hasta la fecha y con qué rapidez las soluciona la comunidad.
- Los requisitos de las licencias de código abierto. También es importante identificar las licencias de software asociadas para mantener el cumplimiento y adherirse a las restricciones.

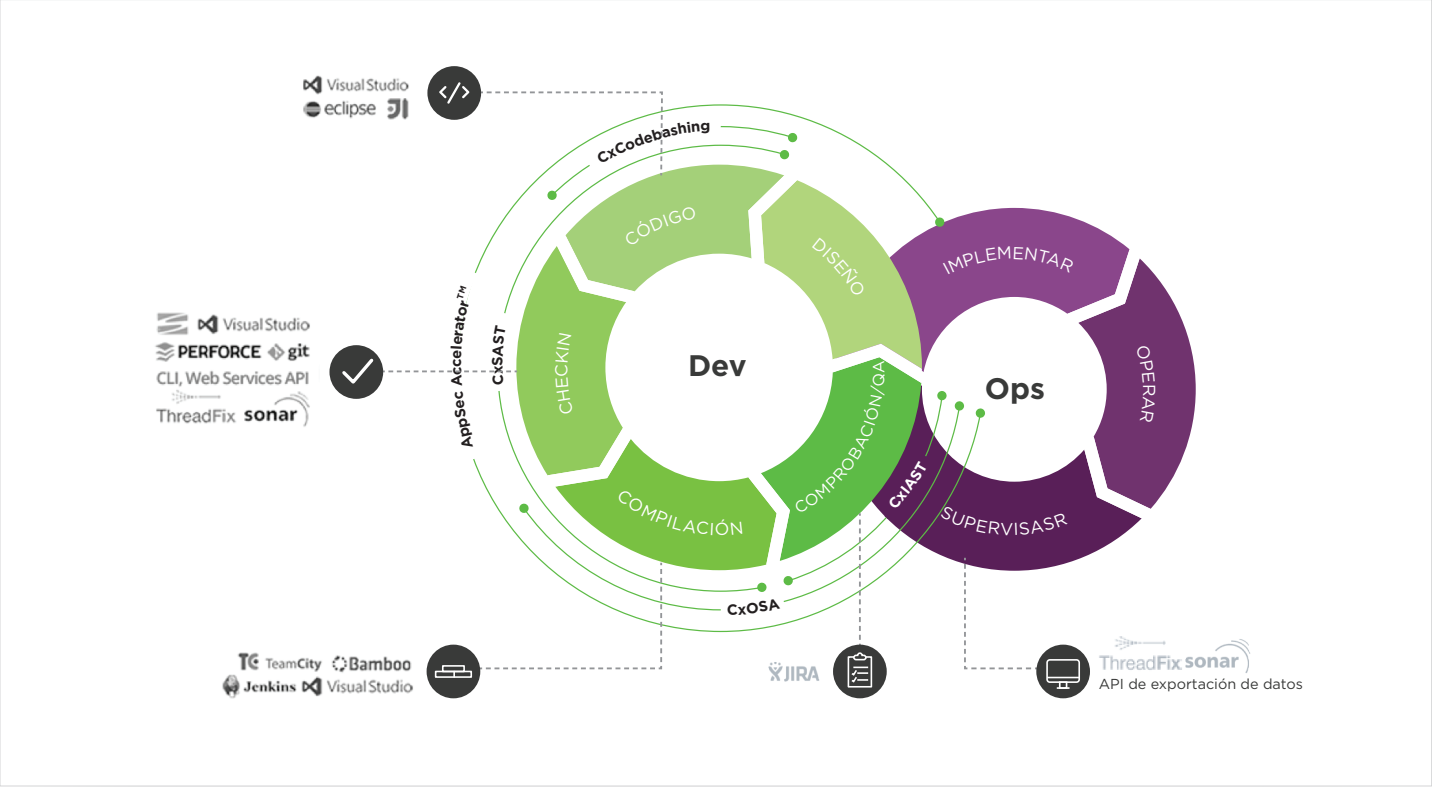
En el caso del código personalizado y de la aplicación completa, es esencial tener un acuerdo que establezca explícitamente cuándo se realizarán las pruebas de seguridad y qué condiciones requerirán la suspensión de una versión. Por ejemplo, una organización puede determinar que las aplicaciones no se pueden implementar si se identifica una vulnerabilidad «grave». Algunas organizaciones pueden optar por detener una compilación cuando se encuentra ese problema, mientras que otras pueden optar por permitir que una compilación continúe incluso con vulnerabilidades graves, si no está programada para su liberación a un entorno de producción.

3. Identificar las vulnerabilidades En todo el SDLC

La seguridad debe integrarse en todas las fases del desarrollo. Este enfoque no solo mejorará la seguridad en los entornos de DevOps, sino que también acelerará el tiempo de comercialización y reducirá el coste de desarrollo, ya que las vulnerabilidades encontradas anteriormente en el SDLC suelen ser menos complicadas y menos costosas de solucionar.

Las soluciones de pruebas de seguridad de aplicaciones estáticas (SAST) se integran en el SDLC desde el principio de la fase de creación de código,-a través de la comprobación y la creación. El análisis de código abierto (OSA) puede utilizarse en las primeras versiones para identificar dependencias de código abierto y asignar esos componentes a vulnerabilidades divulgadas públicamente, continuando en la fase de pruebas o de control de calidad. Las soluciones de pruebas de seguridad de aplicaciones integradas (IAST) utilizadas durante la prueba funcional en la etapa de pruebas o control de calidad también son necesarias. Es fundamental contar con soluciones SAST, OSA e IAST que se integren en la orquestación de CI para que los equipos puedan automatizar los procesos y escanear de manera gradual solamente el código que ha cambiado. Las soluciones que requieren horas para escanear una compilación completa no encajan bien en un entorno de DevOps.

En el caso del software interno, los equipos de seguridad deben recordar que el código abierto y el código personalizado requieren diferentes metodologías de prueba para identificar el riesgo y obtener una verdadera visibilidad de la seguridad de su software.



El diagrama anterior destaca dónde encajan las soluciones de Checkmarx dentro de DevOps.

- **CxSAST** - Pruebas de seguridad de aplicaciones estáticas
- **CxOSA** - Análisis de Código Abierto (también conocido como análisis de composición de software)
- **CxIAST** - Pruebas de seguridad de aplicaciones interactivas
- **CxCodebashing** - Educación sobre codificación segura para gamificación
- **AppSec Accelerator** - Pruebas de seguridad de software administrado

4. Capacitar a los desarrolladores para que codifiquen de forma segura desde el principio

Para abordar la seguridad al principio de la fase de desarrollo, es importante que y la colaboración con sus homólogos de DevOps. La educación constituye una parte muy importante del proceso. Los equipos de seguridad deben formar a los equipos de DevOps en materia de métodos específicos de ataque y técnicas populares de hackeo, proporcionar las herramientas educativas que necesitan para identificar las vulnerabilidades a medida que escriben el código, y actuar como una caja de resonancia durante todo el proceso. Al proporcionar una retroalimentación continua y estar disponibles para responder a las preguntas de codificación segura bajo demanda, los equipos de seguridad pueden reducir en gran medida el tiempo necesario para corregir las vulnerabilidades, lo que resulta en una mayor seguridad y una entrega de software más predecible. Al establecer las mejores prácticas y hacer de la Educación en Codificación Segura (SCE) un proceso continuo, los equipos de seguridad pueden facilitar a los desarrolladores la codificación segura desde el principio. Además, los desarrolladores serán más receptivos a la formación cuando sea relevante, retendrán las lecciones aprendidas y, en última instancia, se convertirán en adalides de la seguridad para las organizaciones.

5. Recuerde que la seguridad de las aplicaciones es un proceso que no tiene fin

Los componentes y marcos de código abierto ofrecen claras ventajas, entre las que se incluyen la-comercialización. Para mantener la solidez de la seguridad, los componentes de código abierto deben analizarse durante las fases de codificación y compilación. Pero no puede terminar ahí. Es de vital importancia seguir supervisando el software de código abierto para detectar las vulnerabilidades que se vayan descubriendo durante todo el SDLC. Algunas vulnerabilidades como ShellShock (CVE-2014-6271) se descubrieron décadas después de que se creara la vulnerabilidad original. Sin una visibilidad dentro de la versión del componente de código abierto y su ubicación en la base de código, es imposible encontrar y corregir esas vulnerabilidades. Para que la seguridad de la aplicación sea efectiva debe tener carácter continuo.

Conclusión

Hoy en día, los internautas malintencionados desvelan IIP que se utiliza para el robo de identidad de forma masiva, si bien el impacto de una filtración de datos va mucho más allá de avergonzar al usuario. Hoy en día, las infracciones causan pérdidas en materia de reputación, pérdidas significativas de valor para los accionistas e incluso el despido de miembros de la alta dirección corporativa. Estas filtraciones también pueden suponer multas considerables debido al constante aumento de las regulaciones, junto a un mayor número de Estas filtraciones también pueden suponer multas considerables investigaciones gubernamentales y la desconfianza del público.

La forma en que las organizaciones de servicios financieros crean software hoy en día es completamente diferente a hace tan solo 10 años. Los nuevos modelos de desarrollo ofrecen software más rápido que nunca antes para satisfacer las cambiantes demandas de los consumidores, maximizar la eficiencia operativa e impulsar la transformación digital. En el competitivo mercado de los servicios financieros, ya no es una opción ofrecer software que no haya sido probado para evitar problemas de seguridad a lo largo del proceso de desarrollo. Los riesgos son demasiado altos. El software está en todas partes y los usuarios confían tanto en el software en sí como en su seguridad para realizar miles de millones de transacciones al día. Es hora de incorporar la seguridad desde el inicio del SDLC para gestionar, medir y abordar mejor el riesgo, capacitar a los equipos de desarrollo y garantizar la entrega segura de software a la velocidad de DevOps.

Checkmarx

Acerca de Checkmarx

Seguridad de software para DevOps y mucho más.

Checkmarx hace que la seguridad del software sea una infraestructura esencial: unificada con DevOps, y perfectamente integrada en todo su proyecto de CI/CD, desde el código no compilado hasta las pruebas en tiempo de ejecución. Nuestra plataforma holística establece el nuevo estándar para inculcar la seguridad en el desarrollo moderno. Obtenga más información aquí.