2. Seguridad de las aplicaciones

2.1

Una parte clave de la protección de una organización implica la protección de las aplicaciones, los sitios web y los servicios en línea que desarrolla y utiliza.

La seguridad debe ser la máxima prioridad al desarrollar, probar e implementar aplicaciones.

2.2 Desarrollo de aplicaciones

Imagen con áreas seleccionables. Seleccione cada botón para mostrar más información.

Para mantener la seguridad en todas las etapas del desarrollo de aplicaciones es necesario seguir un proceso sólido.

Desarrollo y prueba

El software se desarrolla y actualiza en un entorno de desarrollo, donde se puede desarrollar, probar y depurar antes de implementarlo. Un entorno de desarrollo es menos restrictivo que el entorno real y tiene un nivel de seguridad más bajo. El software de control de versiones ayuda a seguir y gestionar los cambios en el código del software. Los desarrolladores también pueden trabajar en un entorno aislado (sandbox) para que el código no se sobrescriba mientras lo desarrollan.

Durante las pruebas, los desarrolladores observan cómo interactúa el código con el entorno normal. El control de calidad (QA) puede encontrar defectos en el software. Es mucho más fácil corregir cualquier defecto encontrado en esta fase.

Ensayo y producción

Los entornos de ensayo deben coincidir con el entorno de producción de la organización.

Al realizar las pruebas en un entorno de ensayo, los desarrolladores pueden verificar que el software se ejecuta con la configuración de seguridad requerida. Después de que el desarrollador ejecuta y prueba la seguridad, el programa puede implementarse en producción.

Aprovisionamiento y desaprovisionamiento

El aprovisionamiento es la creación o actualización de software. El desaprovisionamiento es su eliminación.

Una organización puede utilizar un portal de autoservicio para automatizar el aprovisionamiento y desaprovisionamiento de software.

2.3 Técnicas de codificación de seguridad

Al codificar aplicaciones, los desarrolladores utilizan varias técnicas para validar que se hayan cumplido todos los requisitos de seguridad.

La **normalización** se utiliza para organizar los datos en una base de datos y ayudar a mantener la integridad de los mismos. La normalización convierte una cadena de entrada a su forma más simple conocida para garantizar que todas las cadenas tengan representaciones binarias únicas y que se identifique cualquier entrada maliciosa.

Un **stored procedure** es un grupo de instrucciones SQL precompiladas almacenadas en una base de datos que ejecuta una tarea. Si utiliza un stored procedure para aceptar parámetros de entrada de clientes que utilizan datos de entrada diferentes, reducirá el tráfico de red y obtendrá resultados más rápidos.

Un desarrollador puede utilizar la **ofuscación y el camuflaje** para evitar que el software sea objeto de ingeniería inversa. La ofuscación oculta los datos originales con caracteres o datos aleatorios. El camuflaje sustituye los datos sensibles por datos ficticios realistas.

La **reutilización del código** significa utilizar el software existente para construir un nuevo software, ahorrando tiempo y costos de desarrollo. Sin embargo, hay que tener cuidado para evitar la introducción de vulnerabilidades.

Las bibliotecas de terceros y los **kits de desarrollo de software** (SDK) proporcionan un repositorio de código útil para que el desarrollo de aplicaciones sea más rápido y barato. El inconveniente es que cualquier vulnerabilidad en los SDK o en las bibliotecas de terceros puede afectar potencialmente a muchas aplicaciones.

2.4

Los ciberdelincuentes suelen apuntar a información confidencial almacenada en bases de datos. La implementación de prácticas de seguridad de aplicaciones ayuda a proteger las bases de datos contra ataques.

2.5 Validación de entrada

El control del proceso de introducción de datos es fundamental para mantener la integridad de la base de datos. Muchos ataques se ejecutan contra una base de datos e insertan datos con formato incorrecto. Estos ataques pueden confundir, bloquear o hacer que la aplicación divulgue demasiada información al atacante. Desplácese hacia abajo para ver un ejemplo - en este caso, un ataque de entrada automatizado.

Los clientes rellenan un formulario web para suscribirse a un boletín de noticias. Una aplicación de base de datos genera y envía automáticamente confirmaciones por correo electrónico a los clientes. Cuando los clientes reciben el correo electrónico con un enlace URL para confirmar su suscripción, los atacantes han modificado el enlace URL.

Estas modificaciones pueden cambiar el nombre de usuario, la dirección de correo electrónico o el estado de suscripción de los clientes cuando hacen clic para confirmar su suscripción. De este modo, cuando el correo electrónico es devuelto al host, éste recibe información falsa de la que podría no ser consciente si no comprueba cada dirección de correo electrónico con la información de suscripción.

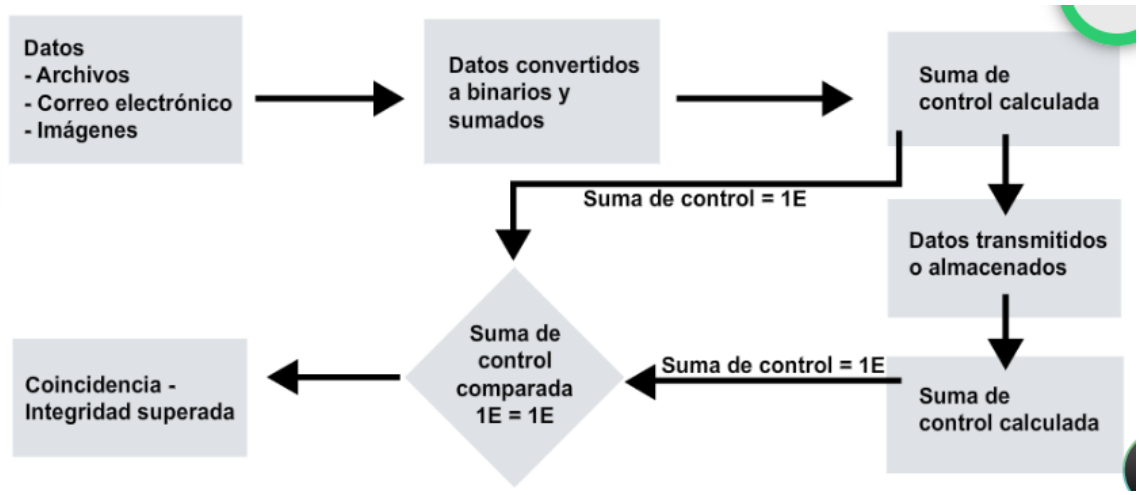
Los hackers pueden automatizar este ataque para inundar la aplicación web con miles de suscriptores no válidos a la base de datos del boletín.

2.6 Reglas de validación

Una regla de validación verifica que los datos se incluyan en los parámetros definidos por el diseñador de la base de datos. Una regla de validación ayuda a garantizar la integridad, la precisión y la coherencia de los datos. Los criterios utilizados en una regla de validación incluyen los siguientes:

- Tamaño: Controla la cantidad de caracteres en un elemento de datos
- Formato: Controla que los datos se ajusten a un formato específico
- Coherencia: Controla la coherencia de los códigos en los elementos de datos relacionados
- Rango: Controla que los datos se encuentran dentro de un valor mínimo y un valor máximo
- Dígito de control: Proporciona un cálculo adicional para generar un dígito de control para la detección de errores.

2.7 Controles de integridad



Los datos comprometidos pueden poner en riesgo la seguridad de sus dispositivos y sistemas.

Un **control de integridad** puede medir la consistencia de los datos en un archivo, imagen o registro para garantizar que no se hayan dañado. El control de integridad realiza una **función hash** para tomar una instantánea de los datos y luego utiliza esta instantánea para garantizar que los datos permanezcan sin cambios. Un **checksum** es un ejemplo de una función de hash.

Una **suma de comprobación** verifica la integridad de los archivos, o cadenas de caracteres, antes y después de que se transfieran entre dispositivos a través de una red local o de Internet. Las sumas de comprobación convierten cada pieza de información en un valor y suman el total. Para comprobar la integridad de los datos, un sistema receptor repite el proceso. Si las dos sumas son iguales, los datos son válidos. De lo contrario, se produjo un cambio en algún punto de la línea.

Las **funciones de hash** comunes incluyen MD5, SHA-1, SHA-256 y SHA-512. Utilizan complejos algoritmos matemáticos para comparar los datos con un valor hash. Por ejemplo, después de descargar un archivo, el usuario puede verificar la integridad del mismo comparando los valores hash de la fuente con los generados por cualquier calculadora de hash.

Las organizaciones utilizan el **control de versiones** para evitar que los usuarios autorizados realicen cambios accidentales. El control de versiones significa que dos usuarios no pueden actualizar el mismo objeto, como un archivo, registro de base de datos o transacción, exactamente al mismo tiempo. Por ejemplo, el primer usuario que abre un documento tiene permiso para modificarlo; la segunda persona que intente abrirlo mientras el primer usuario sigue trabajando en él sólo podrá acceder a una versión de solo lectura.

Las **copias de respaldo** precisas permiten mantener la integridad de datos si los datos se dañan. Una organización necesita verificar su proceso de copia de seguridad para garantizar la integridad de la misma.

La **autorización** determina quién tiene acceso a los recursos de una organización según lo que necesita saber. Por ejemplo, los permisos de archivos y los controles de acceso del usuario garantizan que solo ciertos usuarios pueden modificar los datos. Un administrador puede configurar permisos de solo lectura para un archivo. Como resultado, un usuario con acceso a ese archivo no puede realizar ningún cambio.

2.8 Otras prácticas de seguridad de las aplicaciones

¿Cómo puede estar seguro de que un software que está instalando es auténtico o de que la información está segura al navegar por Internet?

Firma de código

La firma de código ayuda a demostrar que un software es auténtico.

Los archivos ejecutables diseñados para instalarse y ejecutarse en un dispositivo están firmados digitalmente para validar la identidad del autor y garantizar que el código de software no haya cambiado desde su firma.

Cookies seguras

El uso de cookies seguras protege la información almacenada en las mismas de los piratas informáticos.

Cuando el sistema cliente interactúa con un servidor, el servidor envía una respuesta HTTP que indica a su navegador que cree al menos una cookie. La cookie luego almacena datos para futuras solicitudes mientras navega por ese sitio web.

Los desarrolladores web deberían usar cookies con HTTPS para protegerlas y evitar que se transmitan a través de HTTP sin cifrar.

2.9 Gestión de amenazas a las aplicaciones

Las organizaciones pueden implementar diversas medidas para manejar las amenazas al **dominio de aplicación**.

Acceso no autorizado

- Aplicar políticas, normas y procedimientos para el personal y los visitantes a fin de garantizar la seguridad de las instalaciones.

Inactividad

- Desarrollar un plan de continuidad empresarial para que las aplicaciones críticas mantengan la disponibilidad de las operaciones.
- Desarrollar un plan de recuperación tras un desastre para las aplicaciones y los datos críticos.

Sistema operativo

- Desarrollar una política para abordar las actualizaciones del sistema operativo y el software de aplicaciones.
- Instale parches y actualizaciones periódicamente.

Acceso

- Use autenticación de varios factores.
- Supervisión de archivos de registro

Datos

- Implementar estándares de clasificación de datos.
- Implementar procedimientos de copia de respaldo.

Vulnerabilidades

- Realizar pruebas de software antes del lanzamiento.

Bibliografía

Cisco Networking Academy. (2024). Seguridad de las aplicaciones. En S. f. all, *Defensa de la red.*