Nombre: **Fredy Ottoniel Reyes Xitumul**
Carnet: **5990-20-15777**
Materia: **Seguridad y Auditoría de Sistemas**
Sección: **A**

Top 10 Riesgos de Seguridad de Aplicaciones Web

1. **A01:2021-Control de Acceso Roto** sube desde la quinta posición; el 94% de las aplicaciones fueron probadas para algún tipo de control de acceso roto. Las 34 Enumeraciones Comunes de Debilidad (CWE) asignadas al Control de Acceso Roto tuvieron más ocurrencias en las aplicaciones que en cualquier otra categoría.

Ejemplo de escenarios de ataque

La aplicación utiliza datos no verificados en una llamada SQL que accede a información de una cuenta:

pstmt.setString(1, request.getParameter("acct"));

ResultSet results = pstmt.executeQuery( );

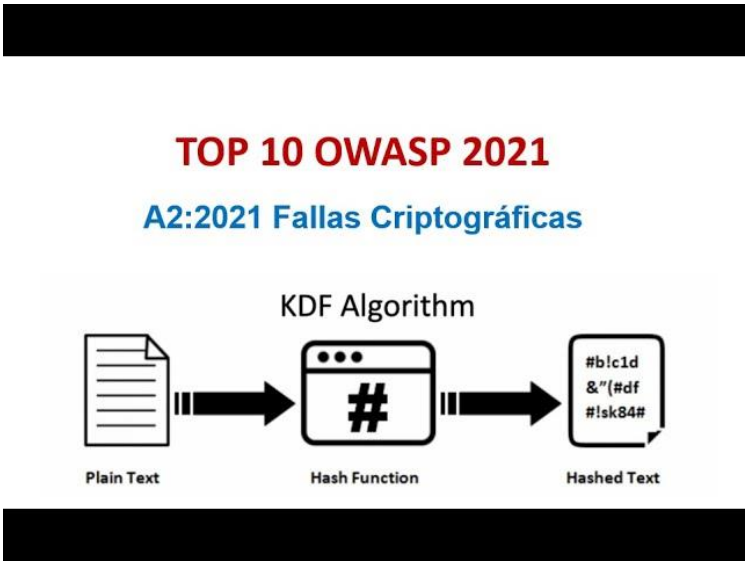
Un atacante simplemente modifica el parámetro 'acct' en el navegador para enviar el número de cuenta que desee. Si no es verificado correctamente, el atacante puede acceder a la cuenta de cualquier usuario.

https://example.com/app/accountInfo?acct=notmyacct

2. **A02:2021-Fallas Criptográficas** cambia una posición a #2, anteriormente conocida como Exposición a Datos Sensibles, que era un síntoma amplio en lugar de una causa raíz. El enfoque renovado aquí está en las fallas relacionadas con la criptografía, que a menudo conduce a la exposición de datos confidenciales o al compromiso del sistema.

Ejemplo de escenarios de ataque

Una aplicación cifra los números de tarjetas de crédito en una base de datos mediante el cifrado automático de la base de datos. Sin embargo, estos datos se descifran automáticamente cuando se recuperan, lo que permite que por una falla de inyección SQL se recuperen números de tarjetas de crédito en texto sin cifrar.



3. **A03:2021-Inyección** se desliza hasta la tercera posición. El 94% de las aplicaciones se probaron para alguna forma de inyección, y las 33 CWE mapeadas en esta categoría tienen la segunda mayor cantidad de ocurrencias en las aplicaciones. Cross-site Scripting es ahora parte de esta categoría en esta edición.

Ejemplo de escenarios de ataque

Una aplicación usa datos no confiables en la construcción de la siguiente sentencia SQL vulnerable:

String query = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "'";

Del mismo modo, la confianza total de una aplicación en frameworks puede resultar en consultas que siguen siendo vulnerables a inyecciones, (por ejemplo: Hibernate Query Language (HQL)):

Query HQLQuery = session.createQuery("FROM accounts WHERE custID='" + request.getParameter("id") + "'");

En ambos casos, el atacante modifica el valor del parámetro "id" en su navegador y enviar por ejemplo: ' UNION SLEEP(10);--.

http://example.com/app/accountView?id=' UNION SELECT SLEEP(10);--

4. **A04:2021-Diseño Seguro** es una nueva categoría para 2021, con un enfoque en los riesgos relacionados con los defectos de diseño. Si realmente queremos “mover a left” como industria, requiere un mayor uso del modelado de amenazas, patrones y principios de diseño seguros y arquitecturas de referencia.

Ejemplo de escenarios de ataque

Un flujo de trabajo de recuperación de credenciales puede incluir "preguntas y respuestas", lo cual está prohibido por NIST 800-63b, OWASP ASVS y OWASP Top 10. No se puede confiar en preguntas y respuestas como evidencia de identidad ya que más de una persona puede conocer las respuestas. Dicho código debe eliminarse y reemplazarse por un diseño más seguro.

5. **A05:2021-Configuración de seguridad** sube del #6 en la edición anterior; el 90% de las aplicaciones fueron probadas para alguna forma de configuración incorrecta. Con más cambios en el software altamente configurable, no es sorprendente ver que esta categoría sube. La categoría anterior para XML External Entities (XXE) ahora es parte de esta categoría.

Ejemplo de escenarios de ataque

El servidor de aplicaciones contiene aplicaciones de ejemplo que no se eliminan del servidor de producción. Estas aplicaciones de ejemplo poseen fallas de seguridad conocidas que los atacantes utilizan para comprometer el servidor. Supongamos que una de estas aplicaciones es la consola de administración y no se modificaron las cuentas predeterminadas. En ese caso, el atacante inicia sesión con las contraseñas predeterminadas y toma el control.

6. **A06:2021-Componentes Vulnerables y Desactualizados** anteriormente se titulaba Using Components with Known Vulnerabilities y está #2 en la encuesta comunitaria Top 10, pero también tenía suficientes datos para hacer el Top 10 a través del análisis de datos. Esta categoría sube del #9 en 2017 y es un tema conocido que luchamos por probar y evaluar el riesgo. Es la única categoría que no tiene ninguna Vulnerabilidad Común y Exposiciones (CVE) asignadas a las CWE incluidas, por lo que se tienen en cuenta los pesos predeterminados de exploit e impacto de 5.0 en sus puntajes.

Ejemplo de escenarios de ataque

Los componentes normalmente se ejecutan con los mismos privilegios que la propia aplicación, por lo que las fallas en cualquier componente pueden tener un impacto grave. Tales fallas pueden ser accidentales (por ejemplo, error de codificación) o intencionales (por ejemplo, una puerta trasera en un componente). Algunos ejemplos de vulnerabilidades de componentes explotables descubiertos son:

- CVE-2017-5638, una vulnerabilidad de ejecución remota de código de Struts 2 que permite la ejecución arbitraria de código en el servidor, ha sido culpada de brechas importantes.
- Si bien el Internet de las Cosas (IoT) es con frecuencia difícil o imposible de parchear, la importancia de parchearlo puede ser grande (por ejemplo, dispositivos biomédicos).

Existen herramientas automatizadas para ayudar a los atacantes a encontrar sistemas sin parches o mal configurados. Por ejemplo, el motor de búsqueda Shodan IoT puede ayudarlo a encontrar dispositivos que aún sufren la vulnerabilidad Heartbleed parchada en abril de 2014.

```
1  from setuptools import setup
2
3  setup(
4      name='learninglib',
5      version='0.2',
6      packages=[],
7      install_requires=['maratlib==0.6'],
8      description='A working ml python API.'
9      )
10
```

7. **A07:2021-Fallas de Identificación y Autenticación** anteriormente se había roto Autenticación y se está deslizando hacia abajo desde la segunda posición, y ahora incluye CWE que están más relacionados con fallas de identificación. Esta categoría sigue siendo una parte integral del Top 10, pero la mayor disponibilidad de marcos estandarizados parece estar ayudando.

Ejemplo de escenarios de ataque

Relleno de credenciales, el uso de listas de contraseñas conocidas, es un ataque común. Supongamos que una aplicación no se implementa protección automatizada de relleno de credenciales. En ese caso, la aplicación puede usarse como oráculo de contraseñas para determinar si las credenciales son válidas.

8. **A08:2021-Fallas de Integridad de Software y Datos** es una nueva categoría para 2021, que se centra en hacer suposiciones relacionadas con actualizaciones de software, datos críticos y tuberías de CI/CD sin verificar la integridad. Uno de los impactos ponderados más altos de los datos de Common Vulnerability and Exposures/Common Vulnerability Scoring System (CVE/CVSS) asignados a las 10 CWE en esta categoría. La deserialización insegura de 2017 ahora es parte de esta categoría más grande.

Ejemplo de escenarios de ataque

Actualizaciones no firmadas: Muchos routers domésticos, decodificadores de televisión, firmware de dispositivos, entre otros, no verifican las firmas de sus actualizaciones de firmware. El firmware sin firmar es un objetivo creciente para los atacantes y se espera que empeore. Esto es una gran preocupación, ya que muchas veces no existe otro mecanismo para remediarlo que corregirlo en una versión futura y esperar a que las versiones anteriores caduquen.

9. **A09:2021-Fallas de Registro y Monitoreo de Seguridad** anteriormente era Insuficiente Logging & Monitoring y se agrega de la encuesta de la industria (#3), subiendo desde #10 anteriormente. Esta categoría se amplía para incluir más tipos de fallas, es difícil de probar y no está bien representada en los datos de CVE/CVSS. Sin embargo, las fallas en esta categoría pueden afectar directamente la visibilidad, las alertas de incidentes y la medicina forense.

Ejemplo de escenarios de ataque

El sitio web de un prestador de salud que provee un plan para niños no pudo detectar una brecha debido a la falta de monitoreo y registro. Alguien externo informó al prestador que un atacante había accedido y modificados registros médicos sensibles de más de 3,5 millones de niños. Una revisión post incidente detectó que los desarrolladores del sitio web no habían encontrado vulnerabilidades significativas. Como no hubo ni registro ni monitores del sistema, la brecha de datos pudo haber estado en proceso desde el 2013, por un período de más de 7 años.

10. **A10:2021-Sirver-Side Solicitud de falsificación** se agrega de la encuesta comunitaria Top 10 (#1). Los datos muestran una tasa de incidencia relativamente baja con una cobertura de pruebas superior al promedio, junto con calificaciones superiores al promedio para Exploit y Impact Potential. Esta categoría representa el escenario en el que los miembros de la comunidad de seguridad nos dicen que esto es importante, a pesar de que no se ilustra en los datos en este momento.

Escaneo de puertos de servidores internos – Si la arquitectura de red no se encuentra segmentada, los atacantes pueden trazar un mapa de las redes internas y determinar si los puertos están abiertos o cerrados en los servidores internos a partir de los resultados de la conexión o del tiempo transcurrido para conectar o rechazar las conexiones de payload SSRF.

Resumen de Soluciones Propuestas:

SSRF: Validar y limitar las solicitudes del lado del servidor.

Control de Acceso Roto: Implementar controles de acceso robustos y revisar permisos.

Componentes Vulnerables: Actualizar componentes y monitorear vulnerabilidades conocidas.

Monitoreo Deficiente: Implementar registros y monitoreo adecuados de seguridad.

Fallos Criptográficos: Usar algoritmos criptográficos modernos y adecuados.

Inyección: Validar entradas y usar consultas parametrizadas.

Diseño Inseguro: Integrar prácticas de seguridad desde la fase de diseño.

Mala Configuración de Seguridad: Configurar entornos de producción de manera segura y eliminar configuraciones por defecto.

Fallos de Autenticación: Usar autenticación fuerte como MFA (Autenticación Multifactor).

Fallos de Integridad: Asegurar integridad de datos con firmas digitales y verificar actualizaciones.

Principales Acciones para Lograr Aplicaciones Seguras:

Pruebas de Seguridad: Realizar auditorías y pruebas de seguridad periódicas.

Seguridad desde el Diseño: Adoptar un enfoque de seguridad desde las primeras fases del desarrollo.

Autenticación y Autorización Fuertes: Implementar autenticación robusta y controlar el acceso.

Validación de Entrada: Filtrar y validar todas las entradas del usuario.

Actualizaciones Constantes: Mantener actualizados los componentes y bibliotecas.

¿Contar con este tipo de sitios es algo contraproducente, en base al hecho que hackers puedes también documentarse al respecto y tratar de alterar las vulnerabilidades identificadas?

NO es contraproducente, ya que su objetivo principal es educar a desarrolladores y organizaciones sobre cómo mejorar la seguridad. Aunque los hackers pueden acceder a esta información, los desarrolladores también aprenden a cerrar brechas antes de que sean explotadas. La transparencia en seguridad fomenta mejores prácticas y estándares globales, lo que en última instancia ayuda a mitigar ataques. Los beneficios de compartir conocimientos sobre seguridad superan los riesgos, ya que permiten a los equipos de desarrollo estar preparados frente a posibles amenazas.