Ataque y defensa en entorno de pruebas, a aplicaciones web (Hacking ético)

Negación de credenciales en aplicaciones web

La sencillez en las contraseñas utilizadas dentro de los sistemas de control solo es uno de los muchos problemas que afectan a las credenciales. Entre ellos destacan (Información obtenida de INCIBE):

- **Uso de contraseñas por defecto:** Es habitual que los dispositivos en ubicaciones remotas dispongan de contraseñas por defecto. De esta manera los operadores pueden acceder a ellos sin tener necesidad de conocer la ubicación, ya que todos poseen las mismas credenciales de acceso.
- **Uso de contraseñas compartidas:** Muchos dispositivos no permiten el uso de múltiples usuarios y en otros se hace simplemente por mejorar la eficiencia, del equipo de mantenimiento o soporte por ejemplo, pero es habitual que las credenciales se compartan entre varios usuarios físicos. Esto impide realizar una trazabilidad de las acciones realizadas, aparte de permitir que los usuarios sigan conociendo las credenciales de acceso aún después de dejar la empresa.
- **Incremento de uso de credenciales:** Cada vez es mayor el número de aplicaciones y usuarios que deben disponer de credenciales para acceder a los sistemas de control. Esto implica un mayor traspaso de información de autenticación por la red, que en caso de no estar correctamente asegurada puede suponer el descubrimiento

de las credenciales.

- **Uso de credenciales privilegiadas:** El uso de cuentas, con permisos administrador o similares, para acceder a diferentes componentes de los sistemas de control es cada vez mayor y en muchos casos llega a ser inmanejable, impidiendo la gestión y el seguimiento de las acciones de forma correcta. Personal de mantenimiento, fabricantes, operadores, ingenieros de control, procesos automáticos y aplicaciones corporativas son algunos ejemplos de uso de cuentas privilegiadas.
- **Contraseñas embebidas:** La introducción de equipos de propósito general, y no de diseño específico para la tarea, ha permitido ahorrar costes y mejorar la compatibilidad, pero también ha supuesto un incremento del uso de contraseñas embebidas. Esto incrementa al riesgo de compromiso y de accesos no autorizados a todo el sistema, que pueden suponer la manipulación de los dispositivos, la ejecución de código o provocar una denegación de servicio.

Recolección de información

La recopilación de datos es el proceso de recopilar y medir información sobre variables específicas en un sistema establecido, que luego permite responder preguntas relevantes y evaluar resultados.

Automatización de conexiones a servidores web (ejemplo: Selenium)

Selenium es un entorno de pruebas de software para aplicaciones basadas en la web. Selenium provee una herramienta de grabar/reproducir para crear pruebas sin usar un lenguaje de scripting para pruebas.

Ejercicios

- https://www.jesusninoc.com/02/01/realizar-busquedas-leyendo-desde-un-fichero-y-pulsar-sobre-un-enlace-en-el-navegador-desde-python/
- https://www.jesusninoc.com/05/01/pulsar-sobre-un-enlace-en-el-navegador-desde-python/
- https://www.jesusninoc.com/04/25/realizar-una-busqueda-en-google-con-python/
- https://www.jesusninoc.com/01/24/realizar-una-busqueda-y-pulsar-sobre-un-enlace-en-chrome-desde-python/
- https://www.jesusninoc.com/05/03/realizar-una-busqueda-y-pulsar-sobre-un-enlace-en-el-navegador-desde-python/
- https://www.jesusninoc.com/05/15/pulsar-sobre-un-enlace-de-la-segunda-pagina-de-los-resultados-de-una-busqueda-desde-python/

Análisis de tráfico a través de proxies de intercepción

Muchos sitios web usan tanto protocolos HTTP como HTTPS para enviar información a usuarios. Mientras que el tráfico HTTP puede ser examinado con facilidad, el tráfico HTTPS está cifrado. Para examinar el tráfico HTTPS solicitado por un usuario en su red, se debe configurar su Firebox para que descifre la información y luego la cifre con un certificado firmado por una CA en la que confía cada cliente de la red (Información obtenida de Watchguard).

El funcionamiento de un proxy de interceptación es el siguiente: cuando su elemento proxy intermedio escanea una conexión HTTPS, el Proxy HTTPS intercepta la solicitud HTTPS e inicia su propia conexión al servidor HTTPS de destino en nombre del cliente. Después de que el proxy reciba del servidor HTTPS de destino una respuesta y una copia del certificado del servidor remoto, el proxy presenta al cliente de origen su propio certificado de firma. Los valores CN, SAN,

entre otros, se mantienen para la validación de identidad. El certificado para nueva firma puede ser el Certificado de Autoridad Proxy Predeterminado o un Certificado CA importado.

Más información

- https://www.jesusninoc.com/02/14/burp-suite/
- https://www.jesusninoc.com/02/19/fiddler-free-web-debugging-proxy/

Búsqueda de vulnerabilidades habituales en aplicaciones web

Información obtenida de Akamai y OWASP.

Las 10 principales vulnerabilidades según OWASP engloban los tipos de vulnerabilidades más frecuentes que se observan en las aplicaciones web. Para evitar la percepción errónea que suelen perpetuar los proveedores de seguridad, no constituyen una lista de comprobación de los vectores de ataque que pueden bloquearse simplemente a través de un firewall de aplicaciones web (WAF). En cambio, su objetivo es concienciar sobre las vulnerabilidades de seguridad más habituales que deben tener en cuenta los desarrolladores de aplicaciones, mejorar dicha concienciación en una serie de prácticas de desarrollo y ayudar a inculcar una cultura de desarrollo seguro.

Para abordar las 10 principales vulnerabilidades según OWASP, es necesario que comprenda el papel que desempeñan tanto los proveedores de seguridad como su propia organización a la hora de proteger las aplicaciones web. Algunas áreas de riesgo solo podrán abordarlas los propios desarrolladores de aplicaciones. Aunque muchos proveedores de seguridad puedan ser de ayuda en

determinadas áreas, no suelen ofrecer una total cobertura o la mejor cobertura posible contra una vulnerabilidad. Las mejores soluciones incluyen una combinación de personas, procesos y tecnologías para mitigar los riesgos asociados con las 10 principales vulnerabilidades.

Para sacar el máximo partido de la lista de las 10 principales vulnerabilidades según OWASP, es necesario comprender dónde, cómo y cuánto pueden ayudar los proveedores de seguridad a ampliar las mejoras de sus propias prácticas de desarrollo.

Top 10 Web Application Security Risks (OWASP)

1. **Injection**. Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker’s hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
2. **Broken Authentication**. Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users’ identities temporarily or permanently.
3. **Sensitive Data Exposure**. Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
4. **XML External Entities (XXE)**. Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can

be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

5. **Broken Access Control**. Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.
6. **Security Misconfiguration**. Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
7. **Cross-Site Scripting (XSS)**. XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
8. **Insecure Deserialization**. Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
9. **Using Components with Known Vulnerabilities**. Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components

with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

10. **Insufficient Logging & Monitoring**. Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

Herramientas para la explotación de vulnerabilidades web

Las principales herramientas para la explotación de vulnerabilidades web son:

- Burp Suite
- Acunetix
- Nessus
- SQLmap
- Aquatone
- Nmap
- Metasploit
- TheHarvester
- WPScan