Detección y corrección de vulnerabilidades de aplicaciones web (Puesta en producción segura)

Desarrollo seguro de aplicaciones web

Cualquier aplicación desarrollada para el entorno web tiene que ser verificada y probada para evitar un mal uso de la misma. Hay que pensar que el usuario puede realizar cualquier operación que no tengamos controlada, los usuarios no va a utilizar las aplicaciones como nos gustaría…

Listas públicas de vulnerabilidades de aplicaciones web. OWASP Top Ten

OWASP es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que el software sea inseguro. La Fundación OWASP es un organismo sin ánimo de lucro que apoya y gestiona los proyectos e infraestructura de OWASP.

Top 10 Web Application Security Risks (https://owasp.org/www-project-top-ten/):

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.

Más información

- https://www.jesusninoc.com/google-hacking-database/

Entrada basada en formularios. Inyección. Validación de la entrada

Cuando un usuario rellena datos en un formulario puede intentar realizar acciones no esperadas por los programadores, diseñadores, etc.

Rellenar datos de forma anómala puede convertirse en una inyección, por ejemplo una inyección SQL es un método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar operaciones sobre una base de datos.

Es fundamental validar los datos que cumplimenta el usuario antes de mandar las peticiones a las bases de datos de los servidores.

Ejercicios

Fuerza bruta

- https://www.jesusninoc.com/04/01/simular-una-fuerza-bruta-mediante-peticiones-http-desde-powershell-utilizando-el-metodo-get/
- https://www.jesusninoc.com/02/23/ataque-de-fuerza-bruta-con-burp-suite-intruder/
- https://www.jesusninoc.com/12/23/script-para-generar-diccionarios-para-realizar-fuerza-bruta-para-sha1-con-powershell/

Estándares de autenticación y autorización

Con la proliferación de servicios en Internet, especialmente los relacionados con las redes sociales, se presenta el problema de la multitud de sistemas de autenticación de usuarios y de acceso a los recursos y datos personales.

Algunos estándares y protocolos de autenticación y autorización utilizados:

- OAuth
- OpenID OAuth Hybrid Protocol
- Facebook Connect
- OpenID Connect

Robo de sesión

Secuestro de sesión, a veces también conocido como secuestro de cookie es la explotación de una sesión de ordenador válida—a veces también llamado una llave de sesión—para obtener acceso no autorizado a información o servicios en un sistema computacional. En particular, suele referirse al robo de una cookie mágica utilizada para autenticar un usuario a un servidor remoto. Es particularmente relevante para desarrolladores web, ya que las cookies de HTTP utilizadas para mantener una sesión en muchos sitios web pueden ser fácilmente robadas por un atacante utilizando un ordenador de intermediario o al acceder a cookies guardadas en el ordenador de la víctima.

Ejemplo

Utilizar sesiones de las aplicaciones web en PowerShell

- https://www.jesusninoc.com/02/26/utilizar-sesiones-de-la

s-aplicaciones-web-en-powershell/

Vulnerabilidades web

Un agujero de seguridad o vulnerabilidad es un fallo en un sistema de información que se puede explotar para violar la seguridad del sistema web.

Más información

- https://www.jesusninoc.com/03/24/detectar-la-version-de-wordpress-leyendo-desde-un-fichero-cada-dominio-en-powershell/

Almacenamiento seguro de contraseñas

El almacenamiento de contraseñas y otros tipos de datos confidenciales se tienen que almacenar de forma segura, lo primero que tenemos que pensar es qué datos debemos almacenar sí o sí, lo mejor es evitar el almacenamiento de datos confidenciales que no nos sirvan así evitamos disgustos.

Un procedimiento que se ha utilizado es aplicar criptografía a los datos confidenciales, tanto el uso de hash como criptografía simétrica o asimétrica.

Ejercicios

- https://www.jesusninoc.com/04/01/ejercicios-de-powershell-generar-sha1-con-las-palabras-de-un-diccionario/

Más información

- https://www.jesusninoc.com/02/01/criptografia-en-powershell/
- https://www.jesusninoc.com/12/18/utilizacion-de-tecnicas-de-programacion-segura-programacion-de-servicios-y-procesos/#Criptografia_de_clave_publica_y_clave_privada

---

Contramedidas. HSTS, CSP, CAPTCHAs, entre otros

Sistemas y protocolos que ofrecen seguridad en algunos punto de las arquitecturas web:

- HTTP Strict Transport Security o HTTP con Seguridad de Transporte Estricta, es una política de seguridad web establecida para evitar ataques que puedan interceptar comunicaciones, cookies, etc.
- Content Security Policy es un estándar de seguridad informática introducido para evitar secuencias de comandos en sitios cruzados, ataques de clic y otros ataques de inyección de código resultantes de la ejecución de contenido malicioso en el contexto de la página web confiable.
- CAPTCHA son las siglas de Completely Automated Public Turing test to tell Computers and Humans Apart. Son pruebas desafío-respuesta controladas por máquinas que son utilizadas para determinar cuándo el usuario es un

humano o un programa automático.

Más información

- https://www.jesusninoc.com/01/05/descargar-la-imagen-del-captcha-de-amazon/
- https://www.jesusninoc.com/01/06/resolver-el-captcha-de-amazon/

Seguridad de portales y aplicativos web. Soluciones WAF (Web Application Firewall)

Un firewall de aplicaciones web es un tipo de firewall que supervisa, filtra o bloquea el tráfico HTTP hacia y desde una aplicación web. Se diferencia de un firewall normal en que puede filtrar el contenido de aplicaciones web específicas, mientras que un firewall de red protege el tráfico entre los servidores.