Cote Valencia Alexis Daniel

8.25

Fundamentos y Técnicas de Seguridad para Aplicaciones

Evaluación de fin de cursado

1. *Gestión de Dependencias como Problema de Seguridad* — 2 puntos — 2

Analiza por qué la gestión de dependencias de software (ej. en NPM, PyPI, apt) se ha convertido en un vector crítico de ataques a la cadena de suministro. Propón dos prácticas, una técnica y una organizacional, para concientizar o reducir este riesgo.

2. *Antipatrones de seguridad* — 1 punto — 0.75

¿Cuáles de los siguientes son *antipatrones de seguridad* reconocidos? Indica cuáles son, e indica qué correspondería hacer si encuentras código con este *antipatrón* en un sistema que estés trabajando.

3/4

a) "Security by Obscurity" (Seguridad por Oscuridad).
b) Almacenar contraseñas en texto plano en la base de datos.
c) Implementar el principio de mínimo privilegio para los servicios.
d) Realizar validación de entrada únicamente del lado del cliente.

3. *Desbordamientos* — 2 puntos — 1.5

3/4

Si bien los desbordamientos más famosos (¡y por buenas razones!) son los desbordamientos de pila (*stack overflows*), la cantidad de amenazas que pueden caracterizarse de esta manera es mucho mayor. Compara la explotación de un *desbordamiento de pila* (*stack overflow*) con un *desbordamiento de enteros* (*integer overflow*).

4. *Enumeración de debilidades, enumeración de vulnerabilidades* — 2 puntos — 2

Dos de los índices mas importantes para la comprensión y el seguimiento de la seguridad en aplicaciones son el **CWE** (*Common Weakness Enumeration*) y el **CVE** (*Common Vulnerabilities Enumeration*). ¿Cuál es la relación entre ambos? ¿Existe algún *camino* que lleve a algún *problema* que descubramos en determinado código a *transitar* de clasificarse como CWE a clasificarse como CVE (o a la inversa)?

5. *Ataques de inyección* — 2 puntos — 1

1/2

Elige un ataque de inyección específico (SQLi, des-serialización, SSRF, etc.) y propón una estrategia de "defensa en profundidad", ubicando controles de seguridad distintos en diferentes capas (ej. aplicación, red, SO) para mitigarlo.

6. *Tipos de inyección* — 1 punto — 1

Relaciona cada concepto (*tipo de inyección*) con su descripción correspondiente.

Tipo de inyección	**Descripción**
a) Inyección de comandos	*a*) La aplicación des-serializa datos no confiables, instanciando objetos o invocando métodos arbitrarios
b) Inyección de objetos	*b*) Datos no validados se envían a un intérprete de comandos del sistema operativo
c) Server-Side Request Forgery (SSRF)	*c*) Entradas de usuario se incorporan directamente en una consulta de base de datos, alterando su lógica
d) Inyección SQL	*d*) La aplicación es engañada para realizar consultas a un destino interno o restringido

Cote Valencia Alexis Daniel

¿Lo peor es que en estos ataques se han comprometido sitios que normalmente consideramos confiables :(

① Esto debido a que hoy en día hay software de orígenes no tanto confiables que, al momento de realizar su instalación, podrían albergar instrucciones/software malicioso sin nosotros darnos cuenta.
Técnica: No realizar la instalación siempre en modo administrador para que, en caso de que exista algo malicioso, no tenga acceso a todo el sistema; o bien hacerlo en entornos contenidos (Dockers, venv, etc.)
Organizacional: Realizar "campañas"/publicidad para informar al usuario de buenas prácticas de búsqueda de software evitando sitios maliciosos como softonic

¿Bastan con campañas y orientación? Hará vez es suficiente"

② b) Esto debido a que si logran acceder a la BD pueden extraer las contraseñas de manera sencilla; para evitar eso en vez de almacenar las contraseñas en plano, almacenaría el HASH de ellas, así sólo se debería corroborar en la BD que el HASH coincide

¿Y qué me dices de @?

En resumen, no debes confiar nunca en que lo que el cliente diregue es correcto realmente lo sea. ¿Y si el cliente es hostil? Probablemente pueda engañarlo

d) Esto debido a que no únicamente el cliente se puede ver comprometido si no también el servidor; propondría una verificación de doble firma para corroborar que, tanto cliente como servidor, son quienes dicen ser una verificación. Los datos siempre deben validarse en el servidor.

③ En el primer caso sería el más peligroso debido a que:

Si hay suficiente nivel, no existe tal cosa como "memoria reservada".

a) Se podría acceder a localidades de memoria reservada? con información importante provocando su modificación o extracción
b) Se podrían ejecutar instrucciones maliciosas en el SO

En el segundo caso provocaría un mal/extraño funcionamiento del programa o de otros

Un integer overflow también puede llevar a la ejecución de código malicioso - pero estoy de acuerdo, hace falta mucho más "talento" para lograr explotarlo, y típicamente son menos peligrosos.
*)0)0-> CVE-2025-7985 por ejemplo.

Cote Valencia Alexis Daniel

④ CVE más que nada es una "BD" donde podemos encontrar diferentes vulnerabilidades ya registradas proporcionando una descripción detallada de cómo suceden, por qué suceden, su impacto pero no como solucionarlo; esto nos permite revisar nuestro sistema/software en busca de estas vulnerabilidades
CWE igual es una "BD" con una clasificación de tipos de vulnerabilidades, de una forma más global
Su relación podría ser/o podríamos verla en cuestión de querer categorizar las vulnerabilidades CVE con el "listado" de CWE! si debido a que CVE es más específico y CWE más general entonces puede darse el caso que la forma de resolver un CVE no sea acorde con CWE y viceversa

típicamente cada CVE incluye una sección de "mitigación"

⑤ SQLi
Aplicación: Realizar funciones de depuración, cómo búsqueda de caracteres especiales para su eliminación/omisión o "parseo" de la información al tipo de dato buscado, para evitar acceso no autorizado a la BD o sustracción de información

La doy por buena porque es una medida en la capa correcta. Pero recuerda que hablamos sobre que no hay caracteres buenos/malos; mucho mejor, prepared statements!

Sesión: Si es una aplicación privada, corroborar las credenciales del usuario ¿Un usuario autenticado no va a hacer un SQLi?
Si es una aplicación pública dar la opción de crear cuenta para que aquellos que tengan una cuenta tengan mayor acceso a la BD que los invitados

Red: Tratar de corroborar si la consulta realizada viene de una IP autorizada o que no está catalogada como insegura

SO: Corroborar, con otro software, que la información solicitada tiene permiso de "difundirse" → ¿Adivinación? :o

En general, me parecen medidas débiles, pero el enfoque de tu respuesta es correcto. ¡Te pido que profundices más respecto a estas medidas!