6.5

Jorge Macaulay Rosales Torralva.

Fundamentos y Técnicas de Seguridad para Aplicaciones

Evaluación de fin de cursado

técnico: Considera cómo evitar las descargas arbitrarias, como el Great Firewall of China? Piensa por qué no es una solución habitual.

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

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

¿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.

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

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

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

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

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

Jorge Macaulay Rosales Torralva

1.- Para poder evitar ataques a la cadena de suministro es importante que la organización implemente medidas de seguridad cuando tanto empleados como operadores descarguen software o actualizaciones.

Para la parte técnica propondría tener un repositorio interno donde se reguarde este tipo de software y que ya haya sido analizado por expertos en seguridad, antivirus, se haya checado el hash y se haya hecho el escaneo correspondiente, así evitaremos que usuarios descarguen software malicioso de lugares que podrían ser peligrosos (todo software y actualizaciones unicamente se podría descargar en este).

Para la parte organizacional buscaría promover cursos para la concientización sobre la seguridad a todos los niveles, desde un usuario común, hasta administradores y operadores, todos en la organización tendrían tips que les ayudaría a identificar riesgos y buenas prácticas de seguridad.

2.- Almacenar contraseñas en texto plano en la base de datos

b) Para aumentar la seguridad se tendrían que cifrar las contraseñas, en caso de que sea necesario almacenarlas. pero lo ideal sería unicamente guardar el hash tanto para seguridad de la BD como para el usuario.

d) Realizar validación de entrada únicamente del lado de cliente.

Esto es una mala práctica de seguridad ya que muchas veces el acceso a una aplicación no solo viene delado del cliente, puede haber acceso interno por lo que es necesario validar la entrada tanto para cliente) como todo tipo de usuario. Es necesario implementa- AAA (Authentication, Autenication y Accounting)? así como controles de acceso distintos.

doné me dices de "Seguridad por Obscuridad?

↳ Si hablamos de validación yo asumiría que ya lo sabías este punto. Hay que hacer validación en el servidor.

esas son las explotaciones, los desbordamientos pueden "simplemente ocurrir" como consecuencia del modelo de manejo de datos de C.

3.- Los desbordamientos stack overflows & integer overflow buscan causar un daño al sistema ya sea para causar una denegación de servicio, buscar comportamiento no deseado o poder hacer escalación de privilegios, entre muchos otros tipos de ataques. → podría ser, aunque en general, sólo se considera la escalación de privilegios.
El stack overflow lo que busca es desbordar la pila para así tomar control del sistema y/o escalar privilegios en cambio el integer overflow buscar desbordar la memoria asignada para ese tipo de dato y así lograr comportamiento no deseado del sistema. Ambos buscan atacar pero ambos tienen diferentes vectores de ataque y objetivos. ¡Sí! ¡Justo que describieras esa diferencia!

4.- Tanto el CWE y CVE son repositorios gigantes que nos ayudan a los expertos en seguridad a saber que "debilidades" o "vulnerabilidades" existe en un determinado sistema, así como las remediaciones, especificaciones y características de estos. Sin estos repositorios la comunidad de seguridad no tendría manera de saber todas vulnerabilidad y debilidades de los sistemas. Un código puede tener tanto vulnerabilidades como debilidades por lo que no está limitado a clasificarse como CVE o CWE ya que puede tener ambos. Para hay una clara distinción entre ambos: CWE habla de clases de debilidad CVE habla de vulnerabilidades específicas. Un CVE pertenece a solo una categoría de CWE.

5.- Para tener una defensa en profundidad protegerla todas las capas del OSI, física, enlace de datos, red, transporte, sesión y aplicación

Jorge Macaulay Rosales Torralva

Para la capa física: Implementaría controles físicos como resguardo de servidores, tener la temperatura adecuada, etc. ¿Cómo contribuye esto contra una inyección? Sí, es buena práctica general.
Para la capa 2: Utilizaría controles de acceso por la MAC address, mismo comentario que capa 1
Capa de Red: Implementaría firewalls, políticas control de acceso, etc. mismo comentario que capa 1
capa de Enlace: implementaría IDS y IPS, así como políticas de seguridad para dicha capa para permitir únicamente ciertos puertos.
para las últimas capas implementaría controles de seguridad para la parte aplicativa como, WAF, control de acceso, cifrado, escaneo de código, etc.

6.- Inyección de comandos Datos no validados se envían a un intérprete de comandos del S.O.

Inyección de objetos La aplicación des-serializa datos no confiables, instanciando objetos, o invocando métodos abritrarios

SSFR La aplicación es engañada para realizar consultas a un destino interno o restringido

Inyección SQL Entradas de usuario se incorporan directamente a una consulta a la base de datos, alterando su lógica.

La respuesta a la pregunta se ubicaría a estos niveles, y de hecho un IPS, seguridad en el aplicativo son parte de la respuesta. Sin embargo, ninguna de las medidas propuestas hace referencia específica al papel del tipo de vulnerabilidad que indiqué.