Guía de Estudio en Español

CompTIA Security+ (SY0-701) — Capítulo 7

Protecting Against Advanced Attacks — Protección contra Ataques Avanzados

---

Este material es un **resumen conceptual elaborado con palabras propias**, pensado como guía de apoyo para estudiar junto al libro original en inglés. No es una traducción ni una reproducción del texto original; cubre los mismos temas, en el mismo orden, para facilitar el repaso en español.

Incluye: resumen por secciones, tablas comparativas, glosario EN↔ES y preguntas de práctica originales.

1. Ataques de Red

1.1 Denegación de servicio

Un **DoS** es un ataque de un solo origen contra un objetivo; un **DDoS** (distribuido) proviene de múltiples equipos. Ambos buscan el **agotamiento de recursos** del objetivo. Indicadores: tráfico de red sostenido y anormalmente alto, y uso elevado de CPU o memoria.

- **DDoS reflejado (reflected):** el atacante envía solicitudes con la IP de origen falsificada (la de la víctima) a un tercero; la respuesta va hacia la víctima.
- **DDoS amplificado (amplified):** combina reflexión con amplificación — una solicitud pequeña genera una respuesta mucho más grande, multiplicando el volumen de tráfico hacia la víctima.
- **Ataque SYN flood:** el atacante envía muchos paquetes SYN sin completar el saludo de tres vías (never envía el ACK final), dejando conexiones 'medio abiertas' que agotan los recursos del servidor.

1.2 Falsificación (forgery) y ataques on-path

Spoofing es un tipo de falsificación donde alguien se hace pasar por otra entidad, modificando la dirección de email, IP o MAC de origen.

Ataque on-path (antes 'man-in-the-middle'): un equipo intermedio intercepta el tráfico entre dos partes sin que ellas lo sepan, pudiendo espiar o insertar código malicioso. Indicadores: demoras en la comunicación, advertencias de certificados no confiables, o advertencias de SSH sobre una huella digital (fingerprint) de clave que cambió inesperadamente.

SSL stripping: variante on-path que degrada una conexión HTTPS cifrada a HTTP sin cifrar. Indicador: el navegador muestra 'No seguro' o la URL usa HTTP en vez de HTTPS donde se esperaba HTTPS.

1.3 Ataques a DNS

- **Envenenamiento de DNS (DNS poisoning):** corrompe los registros almacenados en un *servidor* DNS. DNSSEC ayuda a prevenirlo.
- **Pharming:** similar, pero corrompe la resolución de nombres en el *equipo del usuario* (ej. modificando el archivo hosts local).
- **Redirección de URL:** el usuario termina en un sitio distinto al que intentó visitar.
- **Secuestro de dominio (domain hijacking):** el atacante cambia el registro de un dominio sin permiso del dueño, generalmente accediendo primero a su correo mediante ingeniería social.
- **Filtrado DNS / sinkhole:** técnica defensiva — un DNS sinkhole intercepta consultas hacia dominios maliciosos conocidos, útil para interrumpir botnets.

Ambos ataques (DNS poisoning y pharming) comparten el mismo indicador principal: el usuario escribe una URL pero termina en un sitio web diferente.

1.4 Ataques de repetición (replay)

Un ataque de repetición captura datos de una sesión legítima (incluidas credenciales) y los reenvía más tarde para hacerse pasar por una de las partes — se llama **credential replay** cuando se repiten

credenciales de autenticación. Se combate con marcas de tiempo (timestamps), números de secuencia y autenticación multifactor.

2. Conceptos de Codificación Segura

2.1 Validación de entrada (input validation)

Es la práctica más importante para prevenir ataques web: verificar que los datos sean válidos antes de usarlos (tipo de carácter correcto, bloqueo de código HTML, caracteres peligrosos, rangos válidos). Previene inyecciones SQL, desbordamientos de búfer, XSS y más.

- **Del lado del cliente (client-side):** más rápida, pero se puede evadir (deshabilitando JavaScript o interceptando el tráfico con un proxy).
- **Del lado del servidor (server-side):** más segura, garantiza que la aplicación nunca reciba datos inválidos. Lo ideal es combinar ambas.

2.2 Condiciones de carrera (race conditions)

Ocurren cuando dos procesos intentan acceder al mismo recurso al mismo tiempo, generando resultados inconsistentes. Un caso especial es **TOCTOU** (time-of-check to time-of-use): el atacante intenta modificar un recurso justo entre el momento en que el sistema verifica el permiso y el momento en que efectivamente lo usa.

2.3 Manejo de errores y otras prácticas

- **Manejo de errores:** los mensajes al usuario deben ser genéricos (para no dar pistas a un atacante); la información detallada debe quedar solo en los logs.
- **Ofuscación de código:** hace el código difícil de leer (renombrar variables, eliminar espacios) — no es una medida de seguridad confiable por sí sola.
- **Diversidad de software:** compiladores que introducen variaciones aleatorias para que un exploit que funciona en un sistema falle en otro.
- **Código de terceros/outsourcing:** riesgos de código vulnerable, código malicioso oculto (backdoors, bombas lógicas) y falta de actualizaciones futuras.
- **Cookies seguras:** el atributo 'Secure' obliga a transmitir la cookie solo por canales cifrados (HTTPS).
- **Firma de código (code signing):** certifica quién es el autor y garantiza, mediante hash, que el código no fue modificado.

2.4 Análisis y pruebas de código

Técnica	Qué hace
Análisis estático	Examina el código sin ejecutarlo (revisión manual línea por línea, o herramientas automáticas).
Análisis dinámico	Prueba el código mientras se ejecuta. El fuzzing envía datos aleatorios buscando fallas.
Sandboxing	Prueba en un entorno aislado, sin afectar el sistema real (ej. VMs).

Monitoreo de paquetes	Rastrea el uso de librerías compartidas de terceros para poder parchear rápido si aparece una vulnerabilidad.

Entorno de desarrollo seguro: Desarrollo → Pruebas (Test) → Staging (simula producción) → Producción, con control de versiones y aseguramiento de calidad (QA) en todo el proceso.

3. Bases de Datos y Ataques SQLi

SQL es el lenguaje usado para consultar bases de datos. Una **inyección SQL (SQLi)** ocurre cuando un atacante ingresa datos adicionales en un formulario web para alterar la consulta SQL que se ejecuta en el servidor.

Ejemplo de patrón clásico de SQLi:

```
' or 1=1;--
```

Esto convierte la condición WHERE en siempre verdadera, permitiendo al atacante extraer todos los registros de una tabla, o incluso ejecutar una segunda consulta completa separada por punto y coma.

Protecciones principales: validación de entrada y **procedimientos almacenados parametrizados** (stored procedures) — el dato ingresado se trata como parámetro, no como parte de la sentencia SQL, así que aunque contenga código SQL, se interpreta como texto inofensivo.

3.1 Otros ataques de inyección

- **Inyección DLL:** inyecta una biblioteca de vínculos dinámicos maliciosa en la memoria de un proceso en ejecución.
- **Inyección LDAP:** manipula una consulta a un directorio (ej. Active Directory) para obtener acceso más amplio del previsto.
- **Inyección XML:** inserta datos extra en un documento XML para crear entradas no autorizadas (ej. una cuenta de usuario adicional).
- **Directory traversal:** usa secuencias como '../' para navegar fuera del directorio esperado y acceder a archivos sensibles (ej. /etc/passwd en Linux).

4. Vulnerabilidades de Memoria y Cross-Site Scripting

- **Fuga de memoria (memory leak):** una app reserva memoria y nunca la libera; el sistema se vuelve cada vez más lento hasta requerir reinicio.
- **Desbordamiento de búfer (buffer overflow):** una app recibe más datos de los esperados y expone memoria que debería estar protegida; puede permitir **inyección de memoria** (memory injection) con código malicioso ejecutable.
- **Desbordamiento de enteros (integer overflow):** un valor numérico excede el tamaño reservado para almacenarlo, generando resultados incorrectos.

Cross-Site Scripting (XSS): vulnerabilidad web que permite inyectar scripts en páginas.

- **XSS reflejado (no persistente):** el código malicioso viaja en la solicitud del usuario (ej. un enlace de phishing) y el servidor lo devuelve tal cual en la respuesta.
- **XSS almacenado (persistente):** el código malicioso queda guardado en el servidor (ej. en una base de datos) y se ejecuta cada vez que alguien carga esa página.

La principal defensa contra XSS es la validación de entrada rigurosa y el uso de librerías de codificación/sanitización de HTML.

5. Automatización y Orquestación en Operaciones de Seguridad

5.1 Casos de uso comunes

Aprovisionamiento de usuarios y recursos, guardrails (barandas de seguridad automatizadas), gestión de grupos de seguridad, creación de tickets, escalamiento de incidentes, habilitar/deshabilitar servicios y accesos, integración y pruebas continuas (CI/CD), e integraciones vía APIs.

5.2 Beneficios

- Eficiencia y ahorro de tiempo.
- Aplicación consistente de líneas base (baselines) y configuraciones estándar.
- Escalado seguro de la infraestructura.
- Mayor retención de empleados (menos tareas repetitivas y tediosas).
- Mejor tiempo de reacción ante incidentes.
- Multiplicador de fuerza laboral (workforce multiplier) — el mismo equipo gestiona más sistemas.

5.3 Consideraciones y riesgos

Complejidad añadida, costo inicial, riesgo de un punto único de falla (single point of failure) si se depende demasiado de la automatización, deuda técnica (technical debt) por scripts desactualizados, y necesidad de soporte y mantenimiento continuo de las soluciones automatizadas.

6. Glosario de Términos Clave (Inglés ↔ Español)

Inglés	Español	Definición breve
DoS / DDoS	DoS / DDoS	Denegación de servicio desde uno / múltiples orígenes.
Resource exhaustion	Agotamiento de recursos	Objetivo de un ataque DoS/DDoS.
Reflected / amplified DDoS	DDoS reflejado / amplificado	Variantes que usan terceros para redirigir o multiplicar el tráfico.
On-path attack	Ataque on-path (intermediario)	Intercepción activa del tráfico entre dos partes.
SSL stripping	Degradación de SSL/TLS	Convierte una conexión HTTPS en HTTP sin cifrar.
DNS poisoning / pharming	Envenenamiento DNS / pharming	Corrompen la resolución de nombres en el servidor / en el equipo del usuario.
Domain hijacking	Secuestro de dominio	Cambiar el registro de un dominio sin permiso del dueño.
Replay attack / credential replay	Ataque de repetición	Reenvío de datos capturados (credenciales) para suplantar a una parte.
Input validation	Validación de entrada	Verificar datos antes de usarlos; previene múltiples ataques.
Race condition / TOCTOU	Condición de carrera	Conflicto por acceso simultáneo a un recurso.
Code signing	Firma de código	Certificación digital de autoría e integridad del código.
Static / dynamic analysis	Análisis estático / dinámico	Revisión de código sin ejecutar / durante la ejecución.
Fuzzing	Fuzzing	Envío de datos aleatorios para encontrar fallas.
Sandboxing	Entorno aislado (sandbox)	Área de pruebas aislada del sistema real.
SQL injection (SQLi)	Inyección SQL	Manipular consultas SQL mediante entradas maliciosas.

Inglés	Español	Definición breve
Stored procedure	Procedimiento almacenado	Grupo de sentencias SQL que ayuda a prevenir SQLi.
Directory traversal	Recorrido de directorios	Acceder a archivos fuera del directorio esperado.
Buffer overflow	Desbordamiento de búfer	Recibir más datos de los esperados, exponiendo memoria.
Cross-Site Scripting (XSS)	Secuencias de comandos entre sitios	Inyección de scripts maliciosos en páginas web.
Guardrails	Barandas de seguridad	Controles automatizados que aplican políticas de seguridad.
Technical debt	Deuda técnica	Problemas acumulados por scripts o código desactualizado.

7. Preguntas de Práctica (Autoevaluación)

Preguntas de elaboración propia, en español, para repasar los conceptos del capítulo 7. Las respuestas están al final.

1. Un servidor recibe tráfico masivo desde cientos de direcciones IP distintas al mismo tiempo, agotando sus recursos. ¿Qué tipo de ataque es?

2. Un atacante envía una pequeña solicitud a un servidor DNS de terceros, falsificando la IP de origen como la de la víctima, y logra que el servidor responda con un volumen de datos mucho mayor directamente a la víctima. ¿Cómo se llama esta técnica?

3. Un administrador se conecta por SSH a un servidor y recibe una advertencia de que la huella digital de la clave del host cambió inesperadamente. ¿Qué tipo de ataque podría estar en marcha?

4. Un usuario escribe la URL correcta de un banco, pero termina en un sitio falso porque el atacante modificó el archivo hosts de su computadora. ¿Es esto un envenenamiento de DNS o un ataque de pharming?

5. Un desarrollador nota que su aplicación web de venta de entradas permite que dos personas compren el mismo asiento al mismo tiempo porque no bloquea la selección hasta confirmar el pago. ¿Qué problema de programación es este?

6. Un formulario web permite que un atacante ingrese ' or 1=1;-- en el campo de búsqueda, logrando extraer todos los registros de una tabla. ¿Qué tipo de ataque es y cuál es la mejor defensa?

7. Un atacante logra ejecutar código malicioso enviando más datos de los que un campo de entrada puede manejar, exponiendo memoria protegida del sistema. ¿Cómo se llama este ataque?

8. Un atacante coloca un comentario malicioso en un foro público; cada vez que otro usuario visita esa página, el script se ejecuta automáticamente en su navegador. ¿Qué variante de XSS es esta?

9. Un atacante modifica la URL de una aplicación web de '/app/info.html' a '/app/../../etc/passwd' intentando acceder a un archivo fuera del directorio permitido. ¿Cómo se llama este ataque?

10. Una empresa automatiza la creación de tickets y el escalamiento de incidentes de seguridad para reducir el tiempo de respuesta. ¿Qué beneficio clave de la automatización está aprovechando principalmente?

Respuestas

1. Un ataque DDoS (denegación de servicio distribuida) — proviene de múltiples orígenes simultáneos.

2. Un ataque DDoS amplificado (amplified) — combina reflexión con amplificación del volumen de tráfico.

3. Un ataque on-path (intermediario) — la clave cambiada indica que podría estar conectándose a un sistema distinto al esperado.

4. Es un ataque de pharming — corrompe la resolución de nombres en el equipo del usuario, no en un servidor DNS.

5. Una condición de carrera (race condition), específicamente un problema de TOCTOU si no revalida la disponibilidad antes de confirmar.

6. Una inyección SQL (SQLi); la mejor defensa es la validación de entrada combinada con procedimientos almacenados parametrizados.

7. Un ataque de desbordamiento de búfer (buffer overflow).

8. XSS almacenado (persistente) — el script queda guardado en el servidor y se ejecuta para cada visitante.

9. Un ataque de recorrido de directorios (directory traversal).

10. Mejor tiempo de reacción (reaction time) ante incidentes de seguridad.
