API’s - Tipos de Ataques

DigitalGeko

🛡️ Tipos de ataques comunes a APIs SOAP

1. XXE (XML External Entity)

🧨 **Qué es:**

El atacante incluye entidades externas dentro del XML enviado al servidor SOAP.

🧪 **Ejemplo de payload malicioso:**

xml

<!DOCTYPE foo [<!ENTITY xxe SYSTEM "file:///etc/passwd">]> <soap:Envelope>
<soap:Body>&xxe;</soap:Body> </soap:Envelope>

🎯 **Objetivo del atacante:**

Leer archivos sensibles del servidor o hacer peticiones no autorizadas.

🔒 **Solución:**

Desactivar el uso de DTD (Document Type Definition) o entidades externas en XmlReaderSettings.

2. XML Bomb (Billion Laughs Attack)

🧨 Qué es:

Se crea un XML con entidades anidadas que se expanden exponencialmente al ser procesadas.

🧪 Ejemplo:

xml

```
<!DOCTYPE lolz [ <!ENTITY lol "lol"> <!ENTITY lol1 "&lol;&lol;"> <!ENTITY lol2 "&lol1;&lol1;"> <!ENTITY lol3 "&lol2;&lol2;"> ... ]> <soap:Envelope> <soap:Body>&lol9;</soap:Body> </soap:Envelope>
```

🎯 Objetivo del atacante:

Consumir toda la memoria (RAM) del servidor → **Denegación de servicio (DoS)**.

🔒 Solución:

- Limitar el número de caracteres desde entidades (MaxCharactersFromEntities)
- Limitar el tamaño del documento (MaxCharactersInDocument)

3. Replay Attack

🧨 **Qué es:**
El atacante intercepta un mensaje SOAP válido y lo reenvía más tarde para repetir acciones (como una compra o acceso).

🎯 **Objetivo del atacante:**
Repetir transacciones o acciones sin autorización.

🔒 **Solución:**

Usar tokens de una sola vez (nonce)

Incluir timestamps y validarlos

Firmar mensajes digitalmente (WS-Security)

4. Man-in-the-Middle (MITM)

🧨 Qué es:

El atacante intercepta la comunicación entre cliente y servidor SOAP.

🎯 Objetivo:

Leer, modificar o robar datos transmitidos (contraseñas, tokens, etc.).

🔓 Solución:

Usar HTTPS obligatoriamente

Validar certificados

Implementar WS-Security con firmas y cifrado

5. Schema Poisoning

🧨 Qué es:

El atacante manipula el esquema XML (XSD) para engañar al validador y pasar datos maliciosos.

🎯 Objetivo:

Evadir validaciones, inyectar comandos, o romper lógica del servidor.

🔒 Solución:

Validar contra esquemas predefinidos que controlás

No permitir esquemas cargados dinámicamente

6. Injection (XML/SQL/XPath Injection)

🧨 Qué es:

El atacante inyecta comandos maliciosos en campos XML esperando que el servidor los interprete como código o consulta.

🧪 Ejemplo:

xml

```
<username>' or '1'='1</username>
```

🎯 Objetivo:

Acceder a datos sin permisos, alterar lógica, extraer información.

🔒 Solución:

- Sanitizar entradas
- Usar consultas parametrizadas
- Validar bien el contenido del XML

Resumen

XXE	Leer archivos desde XML	DtdProcessing = Prohibit
XML Bomb	Agotar memoria	Limitar tamaño XML y entidades
Replay Attack	Repetir acciones con mensajes válidos	Timestamps, tokens únicos, firmas
Man-in-the-Middle	Interceptar tráfico	HTTPS + WS-Security
Schema Poisoning	Manipular validación XML	Validar contra esquemas confiables
Injection	Inyectar código o consultas	Validar y sanitizar campos XML

Conceptos

XML	Es el idioma que usan los mensajes SOAP
Ataques XXE	Como si alguien tratara de espiar archivos secretos desde el XML
XML Bomb	Un mensaje que parece chiquito pero explota la memoria del servidor
XmlReader	El lector que lee los mensajes
XmlReaderSettings	Las reglas para que el lector no se trague cualquier cosa
SoapCore	El motor que lee y responde mensajes SOAP

Soluciòn

Configuraciòn	¿Qué hace?	¿Por qué es útil?
ConfigureKestrel(...)	Limita tamaño del cuerpo (10KB)	Protege de archivos grandes o ataques SOAP
AddMemoryCache()	Habilita cache en RAM	Mejora rendimiento
Configure<IpRateLimitOptions>	Lee reglas de rate limit	Controla el abuso por IP
AddInMemoryRateLimiting()	Activa el conteo por IP en memoria	Limita spam o bots
AddSingleton<...>	Configura la lógica del limitador	Hace que todo funcione