Seguridad en Servicios Web en C++

Antonio Espín Herranz

Contenidos

- Autenticación y autorización: OAuth2, JWT y Basic Auth.
- Protección contra ataques comunes (CSRF, XSS, SQL Injection).
- Cifrado y TLS en servicios web.
- Configuración de logs y auditoría de accesos

Autenticación y autorización: OAuth2, JWT y Basic Auth

OAuth2

JWT

JWT

- JSON Web Tokens
  - Formato compacto y seguro
  - Sirve para transmitir información entre partes como un objeto JSON
  - Se utiliza para autenticación y autorización

- Tiene 3 partes codificadas en Base64
  - HEADER.PAYLOAD.SIGNATURE

jwt Header

- Contiene el tipo de token y el algoritmo de firma:

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

jwt - PayLoad

- Contiene los datos (claims) que quieres transmitir, como el usuario, roles, expiración, etc.

```
{
  "sub": "1234567890",
  "name": "Antonio",
  "admin": true,
  "exp": 1699999999
}
```

jwt - Signature

- Es una firma digital generada con el algoritmo especificado (como HMAC o RSA), que garantiza que el contenido no ha sido alterado.

JWT se utiliza

- **Autenticación de usuarios**: El servidor genera un token tras el login y el cliente lo usa en cada petición.
- **Autorización**: El token puede incluir roles o permisos.
- **Intercambio seguro de datos**: Entre servicios o microservicios.

Tipo de Seguridad que ofrece

- JWT **no cifra** el contenido, pero **sí lo firma**. Proporciona:
  - **Integridad**: Nadie puede modificar el contenido sin invalidar la firma.
  - **Confidencialidad**: Cualquiera puede leer el contenido si intercepta el token (por eso se recomienda usar HTTPS).
  - **Autenticidad**: El receptor puede verificar que el token fue emitido por una fuente confiable.

Tipos de JWT

Tipo	Seguridad	Uso común
JWS	Firmado	Autenticación y autorización
JWE	Encriptado	Transmisión de datos sensibles

Tipos de proyectos con JWT y C++

- Servidores REST en C++ (con Pistache, Boost.Beast, Crow, etc.)
- Microservicios en entornos embebidos
- Aplicaciones cliente que consumen APIs protegidas
- Sistemas distribuidos que necesitan autenticación sin sesiones
- Integración con sistemas web donde C++ actúa como backend o middleware

Trabajar en jwt con C++

- La biblioteca más popular en jwt-cpp. Es ligera moderna y compatible con C++11.

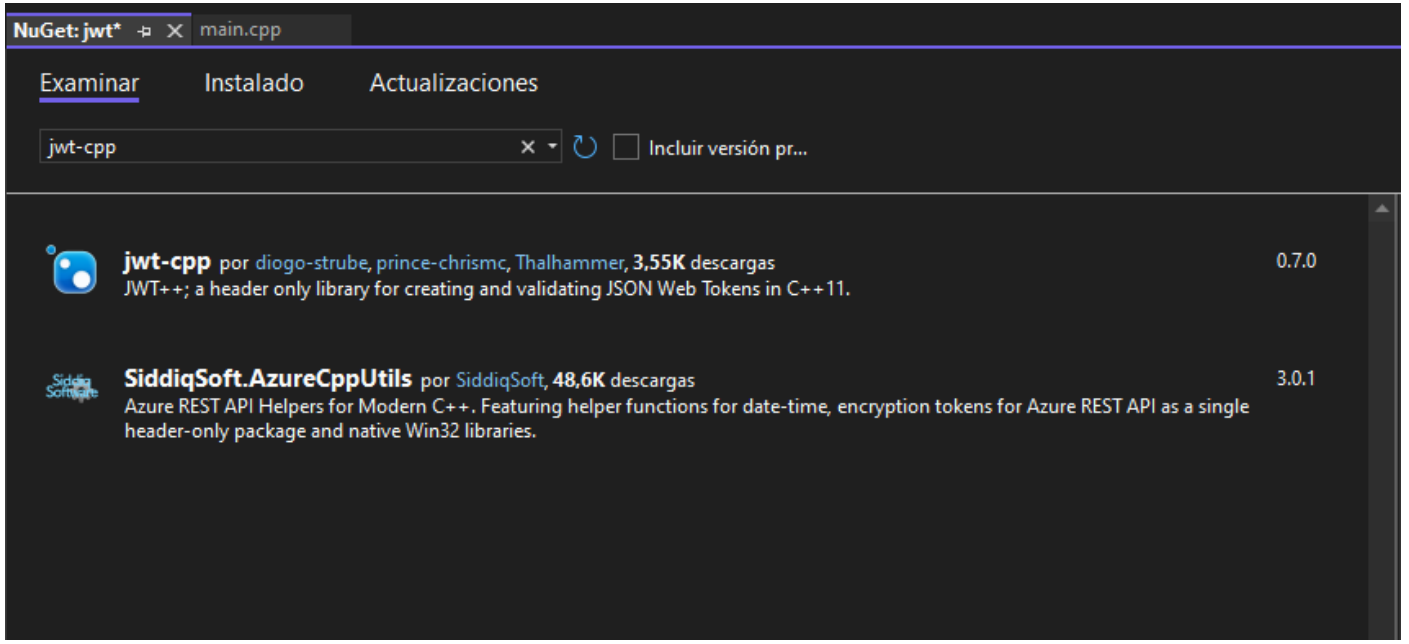
- Se puede crear, firmar y verificar tokens fácilmente y soporta algoritmos como HS256, RS256 y ES256.

Flujo jwt

- **Inicio de sesión:**
  - El cliente envía las credenciales al servidor
- **Creación del JWT:**
  - El servidor valida las credenciales y genera un token firmado
- **Validación del JWT:**
  - El cliente guarda el token y lo envía al servidor en futuras peticiones, el servidor lo verifica
- **Solicitud y respuesta:**
  - Si el token es válido, el servidor procesa la solicitud y responde.

Instalar jwt-cpp en Visual Studio

- Con el botón derecho sobre el proyecto:
- Administrar paquetes **NuGet**
- Examinar: jwt-cpp



Crear un token

Basic Auth

Basic Auth

Protección contra ataques comunes

CSRF

- **Cross-Site Request Forgery**
  - **CSRF** es un tipo de ataque en el que un usuario autenticado en un sitio web es engañado para ejecutar una acción no deseada en ese mismo sitio, sin saberlo.

  - Nos podemos proteger de estos ataques utilizando las librerías: Crow, Boost.Beast o Pistache.

XSS

- **XSS (Cross-Site Scripting)** es una de las vulnerabilidades más comunes y peligrosas en aplicaciones web.

- Afecta principalmente a lenguajes como JavaScript, **los servicios web escritos en C++ también pueden ser vulnerables** si no se validan correctamente los datos que se envían al navegador.

- **XSS** es un tipo de ataque que permite a un atacante **inyectar código malicioso (generalmente JavaScript)** en páginas web que otros usuarios visitan.
  - El código se ejecuta en el navegador de la víctima, no en el servidor, lo que lo hace difícil de detectar.

¿Qué puede hacer un ataque XSS?

- Robar cookies o tokens de sesión
- Suplantar identidad del usuario
- Redirigir a sitios maliciosos
- Mostrar contenido falso o engañoso
- Ejecutar acciones en nombre del usuario

Tipos de XSS

Tipo de XSS	Descripción
Reflejado	El script se envía en la URL y se refleja directamente en la respuesta
Almacenado	El script se guarda en la base de datos y se muestra a otros usuarios
Basado en DOM	El script se ejecuta por el navegador al manipular el DOM con datos maliciosos

SQL Injection

- **SQL Injection (SQLi)** es un ataque que consiste en **insertar código SQL malicioso** en campos de entrada (como formularios o URLs) para manipular las consultas que tu aplicación envía a la base de datos.

- ¿Qué se puede hacer con SQLi?
  - Ver o robar datos confidenciales (usuarios, contraseñas, tarjetas)
  - Borrar o modificar registros
  - Eludir autenticaciones
  - Tomar control del servidor de base de datos
  - Es como si el atacante escribiera comandos directamente en tu consola SQL, aprovechando que tu aplicación confía ciegamente en lo que el usuario escribe

Ejemplo

- std::string query = "SELECT * FROM usuarios WHERE nombre = '" + nombre + “’”;

- Se añade OR ‘1’ = ‘1’

Cifrado y TLS en servicios web

Configuración de logs y auditoría de accesos

Herramientas

- **Fluentd** es un colector de logs de código abierto que:
  - Recolecta logs desde múltiples fuentes (archivos, syslog, stdout, etc.)
  - Los transforma y filtra (formato, etiquetas, niveles)
  - Los reenvía a destinos como Elasticsearch, Kafka, S3, etc.
  - Es parte de la **Cloud Native Computing Foundation (CNCF)** y se usa en producción por empresas como Amazon, Microsoft y Google.

- **Elasticsearch**
  - Es un motor de búsqueda y análisis de texto distribuido que:
    - Indexa logs en tiempo real
    - Permite búsquedas complejas y agregaciones
    - Se integra con **Kibana** para visualización

Integración Fluentd y Elasticsearch

- Fluentd recolecta los logs del servicio C++ (por ejemplo, desde archivos generados por spdlog, Boost.Log, etc.)
- Los transforma (añade etiquetas, convierte a JSON, etc.)
- Los envía a Elasticsearch, donde se almacenan e indexan
- Kibana (opcional) los visualiza en dashboards interactivos

Instalación en Docker

- **Fluentd**

```
docker run -d -p 24224:24224 -p 24224:24224/udp \
  -v /path/to/fluent.conf:/fluentd/etc/fluent.conf \
  fluent/fluentd
```

- **Elasticsearch**

```
docker run -d -p 9200:9200 -p 9300:9300 \
  -e "discovery.type=single-node" \
  elasticsearch:8.12.0
```

Se puede auditar

- Accesos a endpoints
- Errores HTTP
- IPs de origen
- Usuarios autenticados
- Cambios en datos sensibles