**Informe Técnico**

**Detección de Formularios HTML y Campos de Entrada con Python – Lección 5**

**1. Escenario**

Durante las fases de **reconocimiento web** en pruebas de seguridad, resulta fundamental identificar los **formularios HTML** presentes en un sitio, ya que suelen ser puntos de interacción críticos y posibles vectores de ataque. El objetivo de este laboratorio fue desarrollar un script en **Python** que permita conectarse a una página web pública y listar de manera automatizada los formularios detectados, sus métodos de envío y los campos de entrada asociados.

**2. Metodología**

Se aplicaron los siguientes pasos:

1. Uso de la librería **requests** para realizar la conexión HTTP al sitio.
2. Análisis del contenido HTML mediante **BeautifulSoup**.
3. Extracción de etiquetas <form> y recorrido de sus atributos relevantes.
4. Para cada formulario: mostrar el método (GET/POST), acción (action) y campos de entrada (input, textarea, select).
5. Discusión sobre qué formularios podrían ser **críticos si no cuentan con validaciones adecuadas**.

**3. Implementación Técnica**

**Código en Python (detectar\_formularios.py)**

import requests

from bs4 import BeautifulSoup

def detectar\_formularios(url):

try:

res = requests.get(url)

soup = BeautifulSoup(res.text, "html.parser")

forms = soup.find\_all("form")

print(f"[+] Formularios detectados en {url}: {len(forms)}\n")

for i, form in enumerate(forms, start=1):

print(f"=== Formulario {i} ===")

print("Método:", form.get("method", "GET").upper())

print("Acción:", form.get("action", ""))

campos = form.find\_all(["input", "textarea", "select"])

for campo in campos:

nombre = campo.get("name")

tipo = campo.get("type", "textarea" if campo.name == "textarea" else "select")

print(f" - name: {nombre} | type: {tipo}")

print()

except Exception as e:

print("Error al conectar:", e)

if \_\_name\_\_ == "\_\_main\_\_":

detectar\_formularios("https://httpbin.org/forms/post")

**4. Ejemplo de Salida**

[+] Formularios detectados en https://httpbin.org/forms/post: 1

=== Formulario 1 ===

Método: POST

Acción: /post

- name: custname | type: text

- name: custtel | type: tel

- name: custemail | type: email

- name: size | type: select

- name: topping | type: checkbox

- name: comments | type: textarea

**5. Análisis de Seguridad**

Los formularios constituyen la principal interfaz de entrada de datos en aplicaciones web. Si no se implementan **controles de validación y saneamiento**, pueden dar lugar a ataques como:

* **SQL Injection (SQLi)** en campos de texto.
* **Cross-Site Scripting (XSS)** si se permite insertar HTML/JavaScript malicioso.
* **CSRF** si los formularios carecen de tokens únicos de verificación.

**6. Recomendaciones de Ciberseguridad**

1. Implementar **validación en servidor** de todos los campos recibidos.
2. Escapar y sanitizar entradas para prevenir XSS.
3. Usar **consultas preparadas** para prevenir SQLi.
4. Incorporar **tokens CSRF** en formularios críticos.
5. Configurar cabeceras HTTP de seguridad (**CSP, HSTS, X-Frame-Options**).

**7. Reflexión**

El laboratorio mostró cómo con pocas líneas de Python se puede automatizar el reconocimiento de formularios, etapa fundamental para un análisis de seguridad web. Este tipo de herramientas permiten a los equipos de **pentesting** y desarrolladores identificar la superficie de ataque y reforzar la seguridad de aplicaciones en fases tempranas.

**8. Conclusión**

El script desarrollado permitió detectar y listar formularios web de manera efectiva, incluyendo métodos, acciones y campos. Se comprobó que los formularios constituyen puntos críticos de interacción con los usuarios y requieren controles estrictos de seguridad. Este ejercicio refuerza la importancia del **reconocimiento web automatizado** y de aplicar lineamientos **OWASP** para proteger las aplicaciones frente a ataques comunes.