**Informe Técnico**

**Detección de Vulnerabilidad XSS mediante Fuzzing con Burp Suite**

**1. Escenario**

Durante una auditoría ética sobre el sitio de pruebas [**http://testphp.vulnweb.com**](http://testphp.vulnweb.com?utm_source=chatgpt.com), se analizó el formulario de búsqueda en la ruta /artsearch.php. Se sospechaba que el parámetro artname carecía de validación y podía ser vulnerable a **Cross-Site Scripting (XSS) reflejado**. El objetivo fue simular un ataque de fuzzing automatizado con Burp Suite Intruder para confirmar la existencia de la vulnerabilidad.

**2. Metodología**

La práctica se ejecutó en tres fases principales:

1. **Interceptación de la solicitud**: Configuración del navegador con proxy en 127.0.0.1:8080 y envío de la petición interceptada hacia Intruder.
2. **Configuración del ataque**: En la pestaña *Positions* se seleccionó el valor de artname. Se cargaron payloads XSS comunes:
   * <script>alert(1)</script>
   * "><img src=x onerror=alert(1)>
   * <svg/onload=alert('XSS')>
   * '\"><svg onload=confirm(1)>
3. **Análisis de resultados**: Revisión de las respuestas de la aplicación en Burp Suite (códigos de estado, longitud, renderizado de HTML) para verificar la ejecución de los scripts inyectados.

**3. Resultados**

* Varias respuestas reflejaron los payloads en el HTML sin sanitización.
* En modo *Render* se ejecutaron alertas JavaScript, confirmando la vulnerabilidad.
* La aplicación no implementa controles de salida ni filtros para el campo artname.

**4. Evidencia Técnica**

* **Solicitud interceptada**: parámetro artname modificado con payloads XSS.
* **Respuestas**: el navegador mostró ejecución de alertas en la vista renderizada de Burp Suite.
* **Diagnóstico**: la aplicación es vulnerable a **XSS reflejado**, con riesgo de robo de cookies, secuestro de sesión o redirecciones maliciosas.

**5. Recomendaciones de Mitigación**

1. Implementar **escapado y codificación de salida** en todos los valores mostrados al usuario.
2. Validar y sanitizar entradas en el servidor antes de procesarlas.
3. Configurar políticas de seguridad de contenido (**Content Security Policy – CSP**).
4. Establecer filtros en el WAF para bloquear intentos de inyección de scripts.
5. Deshabilitar mensajes de error detallados que puedan revelar más información del sistema.

**6. Preguntas de Reflexión**

* **¿Por qué son peligrosos los XSS reflejados?**  
  Porque permiten al atacante ejecutar scripts en el navegador de la víctima, comprometiendo su sesión y datos.
* **¿Qué importancia tiene codificar correctamente la salida?**  
  Garantiza que el navegador interprete el input como texto y no como código ejecutable.
* **¿Qué limitaciones tiene el análisis automático?**  
  Puede generar falsos positivos o no detectar vectores complejos; por ello es necesario complementarlo con pruebas manuales.

**7. Conclusión**

El fuzzing realizado con Burp Suite confirmó que el parámetro artname es vulnerable a **XSS reflejado**. Esta falla permite la ejecución de código malicioso en navegadores de usuarios finales, representando un riesgo alto para la confidencialidad y la integridad de los datos. Se recomienda aplicar controles de validación, sanitización y políticas de seguridad para mitigar este tipo de ataques conforme a los lineamientos del **OWASP Top 10**.