
**UNIVERSIDAD NACIONAL**

**SAN AGUSTIN DE AREQUIPA**

**Facultad de Ingeniería, Producción y Servicios**

**Escuela Profesional de Ciencia de la Computación**

---

# Laboratorio 6

---

**Presentado por:**

Parizaca Mozo, Paul Antony

**CUI:**

20210686

**Github:**

https://github.com/PaulParizacaMozo/IS2/tree/main/Laboratorio06

**Curso:**

Ingeniería de Software II

**Docente:**

Edgar Sarmiento Calisaya

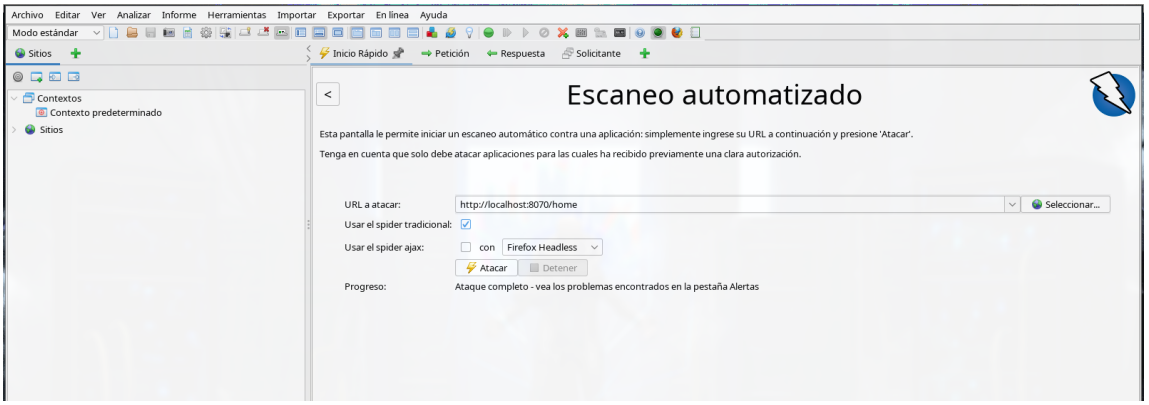
Arequipa, Perú

2023

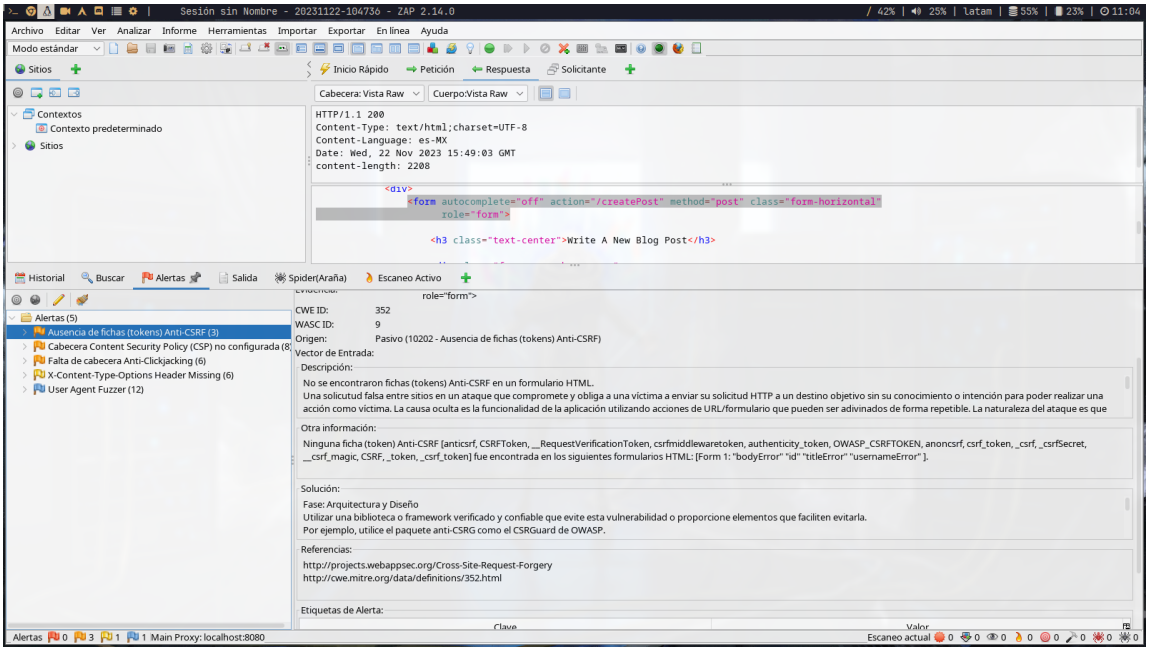
## Lanzamos la aplicación web a probar:

```
 SimpleBlogDemo-0.0.1-SNAPSHOT.jar
❯ java -jar SimpleBlogDemo-0.0.1-SNAPSHOT.jar
10:48:07,779 |-INFO in ch.qos.logback.classic.LoggerContext[default] - Could NOT find resource [logback-test.xml]
10:48:07,780 |-INFO in ch.qos.logback.classic.LoggerContext[default] - Could NOT find resource [logback.groovy]
10:48:07,781 |-INFO in ch.qos.logback.classic.LoggerContext[default] - Found resource [logback.xml] at [jar:file:/home/paul/Temp/IS2/Practica06/SimpleBlogDemo-0.0.1-SNAPSHOT.jar!/BOOT-INF/classes!/logback.xml]
10:48:07,820 |-INFO in ch.qos.logback.core.joran.spi.ConfigurationWatchList@1c72da34 - URL [jar:file:/home/paul/Temp/IS2/Practica06/SimpleBlogDemo-0.0.1-SNAPSHOT.jar!/BOOT-INF/classes!/logback.xml] is not of type file
10:48:07,953 |-INFO in ch.qos.logback.classic.joran.action.ConfigurationAction - debug attribute not set
10:48:07,966 |-INFO in ch.qos.logback.classic.joran.action.LoggerAction - Setting level of logger [com.barry.home.SimpleBlogDemo.service.implementation.CommentServImpl] to DEBUG
10:48:07,971 |-WARN in ch.qos.logback.core.joran.action.IncludeAction - Could not find resource corresponding to [org/springframework/boot/logging/logback/defaults.xml]
10:48:07,971 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - About to instantiate appender of type [ch.qos.logback.core.ConsoleAppender]
10:48:07,982 |-INFO in ch.qos.logback.core.joran.action.AppenderAction - Naming appender as [STDOUT]
10:48:08,152 |-INFO in ch.qos.logback.classic.joran.action.RootLoggerAction - Setting level of ROOT logger to INFO
10:48:08,152 |-INFO in ch.qos.logback.core.joran.action.AppenderRefAction - Attaching appender named [STDOUT] to Logger[ROOT]
10:48:08,153 |-INFO in ch.qos.logback.classic.joran.action.ConfigurationAction - End of configuration.
10:48:08,154 |-INFO in ch.qos.logback.classic.joran.JoranConfigurator@6b0c2d26 - Registering current configuration as safe fallback point


  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.2.2.RELEASE)

2023-11-22 10:48:09.070 [main] INFO  c.b.h.S.SimpleBlogDemoApplication - Starting SimpleBlogDemoApplication v0.0.1-SNAPSHOT on ArchLinux with PID 26345 (/home/paul/Temp/IS2/Practica06/SimpleBlogDemo-0.0.1-SNAPSHOT.jar started by paul in /home/paul/Temp/IS2/Practica06)
2023-11-22 10:48:09.075 [main] INFO  c.b.h.S.SimpleBlogDemoApplication - No active profile set, falling back to default profiles: default
2023-11-22 10:48:11.505 [main] INFO  o.s.d.r.c.RepositoryConfigurationDelegate - Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2023-11-22 10:48:11.627 [main] INFO  o.s.d.r.c.RepositoryConfigurationDelegate - Finished Spring Data repository scanning in 102ms. Found 2 JPA repository interfaces.
2023-11-22 10:48:12.556 [main] INFO  o.s.c.s.PostProcessorRegistrationDelegate$BeanPostProcessorChecker - Bean 'org.springframework.transaction.annotation.ProxyTransactionManagementConfiguration' of type [org.springframework.transaction.annotation.ProxyTransactionManagementConfiguration] is not eligible for getting processed by all BeanPostProcessors (for example: not eligible for auto-proxying)
2023-11-22 10:48:13.311 [main] INFO  o.s.b.w.e.tomcat.TomcatWebServer - Tomcat initialized with port(s): 8070 (http)
2023-11-22 10:48:13.326 [main] INFO  o.a.coyote.http11.Http11NioProtocol - Initializing ProtocolHandler ["http-nio-8070"]
2023-11-22 10:48:13.328 [main] INFO  o.a.catalina.core.StandardService - Starting service [Tomcat]
```

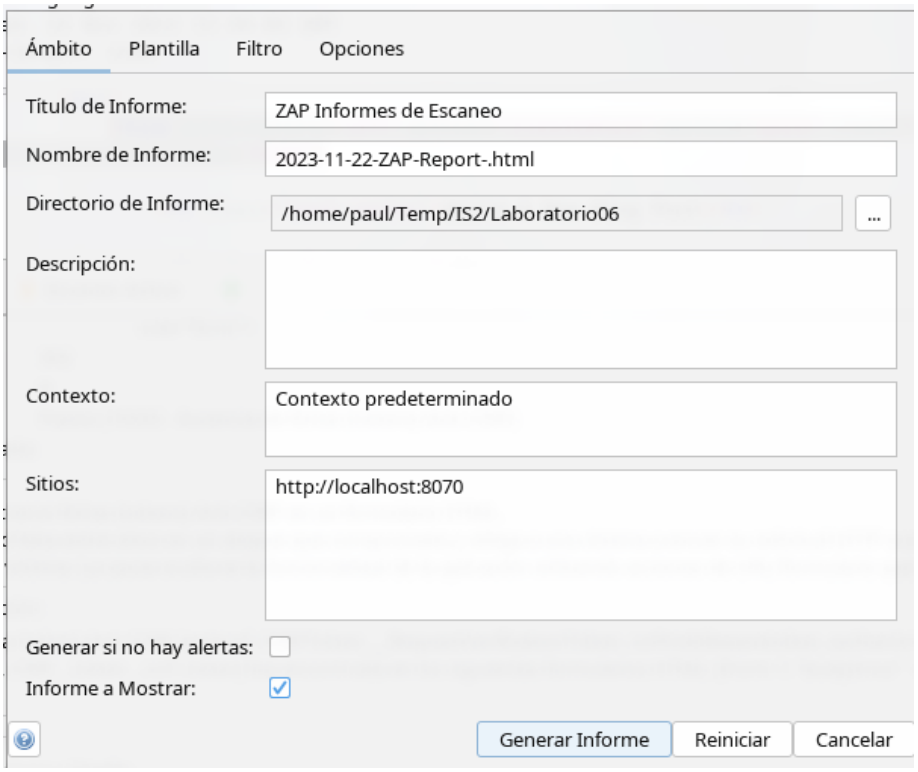
## Ejecutamos nuestra aplicación zaproxy, colocamos las url de nuestra aplicación en este caso será de forma local en el puerto 8070



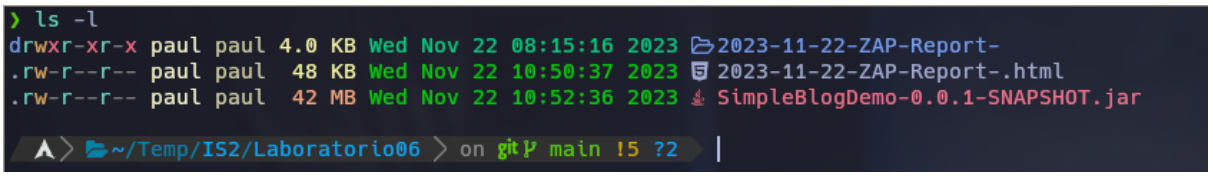

## Nos muestra las alertas encontradas en la aplicación web, en este caso son 5 alertas

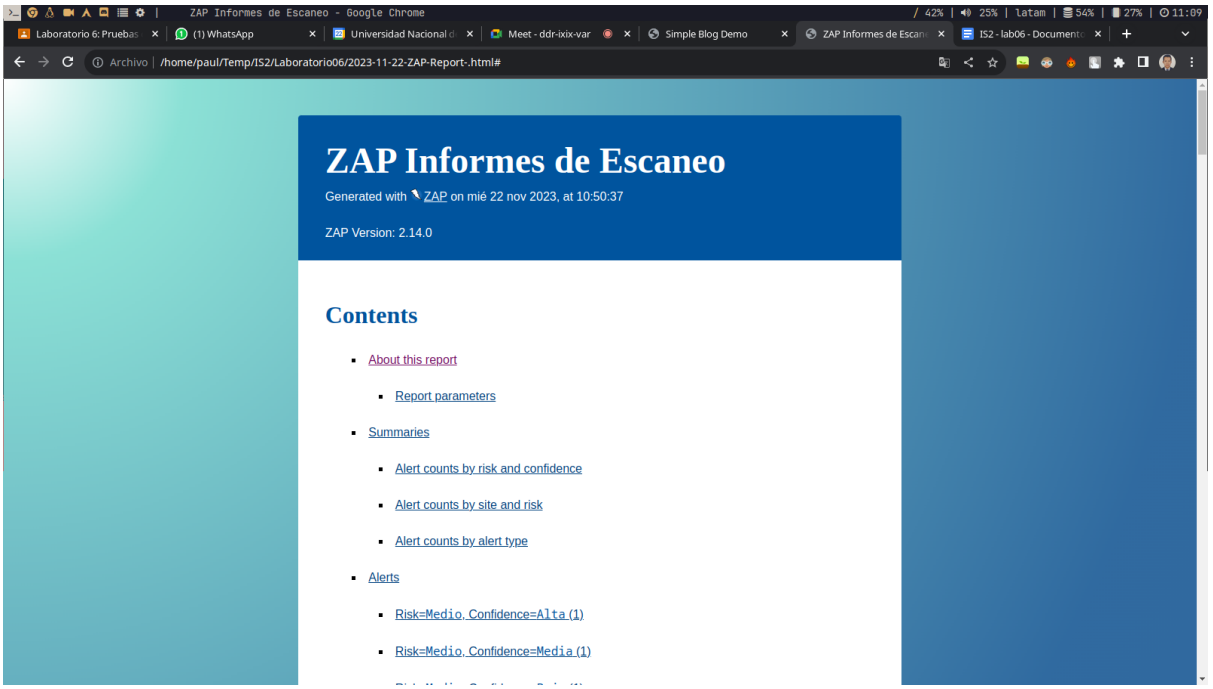


**Generamos un informe:**




**Si listamos nuestro directorio, vemos que se encuentra un directorio y un fichero html el cual si lo cargamos nos mostrará el reporte**



**En este caso tenemos 5 alertas**

**1.**

**Ausencia de fichas (tokens) Anti-CSRF**

| | |
|---|---|
| Source | raised by a passive scanner (Ausencia de fichas (tokens) Anti-CSRF) |
| CWE ID | 352 |
| WASC ID | 9 |
| Reference | ▪ http://projects.webappsec.org/Cross-Site-Request-Forgery<br>▪ http://cwe.mitre.org/data/definitions/352.html |

Anti-CSRF Tokens: Estos tokens son un mecanismo de defensa contra ataques CSRF (Cross-Site Request Forgery). En un ataque CSRF, un atacante puede realizar acciones en nombre de un usuario autenticado sin su consentimiento. Los tokens Anti-CSRF se utilizan para asegurar que las solicitudes provengan de usuarios legítimos y no de atacantes.

CSRF (Cross-Site Request Forgery): Es un tipo de ataque en el que un atacante engaña al navegador de un usuario para que realice acciones no deseadas en una aplicación web en la que el usuario está autenticado. Esto puede incluir acciones como cambiar la contraseña de un usuario, realizar compras no autorizadas, etc.

Alerta de Ausencia de Tokens Anti-CSRF: Esta alerta indica que la aplicación web que estás evaluando no está utilizando tokens Anti-CSRF como medida de seguridad. Sin estos tokens, la aplicación podría ser vulnerable a ataques CSRF, ya que no hay un mecanismo eficaz para verificar la autenticidad de las solicitudes.

Implementar tokens Anti-CSRF es crucial para prevenir este tipo de ataques y garantizar que las solicitudes provengan de usuarios legítimos. La falta de estos tokens podría poner en riesgo la seguridad de la aplicación y permitir que los atacantes realicen acciones en nombre de los usuarios sin su consentimiento.

**2.**

**Cabecera Content Security Policy (CSP) no configurada**

| | |
|---|---|
| **Source** | raised by a passive scanner (Cabecera Content Security Policy (CSP) no configurada) |
| **CWE ID** | 693 |
| **WASC ID** | 15 |
| **Reference** | ▪ https://developer.mozilla.org/en-US/docs/Web/Security/CSP/Introducing_Content_Security_Policy<br>▪ https://cheatsheetseries.owasp.org/cheatsheets/Content_Security_Policy_Cheat_Sheet.html<br>▪ http://www.w3.org/TR/CSP/<br>▪ http://w3c.github.io/webappsec/specs/content-security-policy/csp-specification.dev.html<br>▪ http://www.html5rocks.com/en/tutorials/security/content-security-policy/<br>▪ http://caniuse.com/#feat=contentsecuritypolicy<br>▪ http://content-security-policy.com/ |

Content Security Policy (CSP): Es un mecanismo de seguridad que ayuda a prevenir ataques como inyecciones de código malicioso (por ejemplo, ataques de script entre sitios) al permitir a los desarrolladores especificar los dominios desde los cuales se pueden cargar recursos en una página web. CSP ayuda a mitigar el riesgo de ataques como XSS (Cross-Site Scripting) al restringir el origen de los scripts y otros recursos.

Cabecera HTTP: La CSP se implementa enviando una cabecera HTTP llamada Content-Security-Policy desde el servidor web al navegador del usuario. Esta cabecera contiene las directivas que especifican las reglas de seguridad para la carga de recursos.

Alerta de Cabecera CSP no Configurada: La alerta indica que la aplicación web no está enviando la cabecera HTTP Content-Security-Policy, lo que significa que no se ha implementado una política de seguridad de contenido.

La falta de una política de seguridad de contenido podría dejar la aplicación vulnerable a ataques como XSS, donde un atacante podría inyectar scripts maliciosos en la página web y ejecutarlos en el navegador de los usuarios.

Para abordar esta alerta, se recomienda configurar una política de seguridad de contenido adecuada para la aplicación web. Esto implica definir las directivas adecuadas en la cabecera Content-Security-Policy para restringir la carga de recursos solo desde fuentes confiables y reducir el riesgo de ataques. Implementar CSP es una práctica sólida de seguridad web.

**3.**


La alerta sobre la "Falta de cabecera Anti-Clickjacking" indica que la aplicación web carece de una medida de seguridad específica para prevenir ataques de clickjacking. El clickjacking es un tipo de ataque en el que un atacante engaña a un usuario para que haga clic en algo diferente de lo que el usuario percibe, posiblemente llevándolo a realizar acciones no deseadas sin su conocimiento.

La medida de seguridad específica mencionada aquí es la cabecera X-Frame-Options en las respuestas HTTP. Esta cabecera se utiliza para controlar si un navegador debe permitir que una página web se cargue en un marco (<iframe>). Su propósito es prevenir ataques de clickjacking al evitar que la página sea incrustada en un marco y, por lo tanto, haciéndola menos susceptible a manipulaciones no deseadas.

Para abordar esta alerta, se recomienda configurar la cabecera X-Frame-Options en las respuestas HTTP de la aplicación web. Hay tres posibles valores para esta cabecera:

DENY: Indica que la página no debe cargarse en un marco, independientemente del origen.

SAMEORIGIN: Permite que la página se cargue en un marco solo si el documento que contiene la etiqueta <iframe> tiene el mismo origen que la página solicitada.

ALLOW-FROM uri: Permite que la página se cargue en un marco solo si el documento que contiene la etiqueta <iframe> tiene el URI especificado.

**4.**

### X-Content-Type-Options Header Missing

| | |
|---|---|
| **Source** | raised by a passive scanner (X-Content-Type-Options Header Missing) |
| **CWE ID** | 693 |
| **WASC ID** | 15 |
| **Reference** | ▪ http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx<br>▪ https://owasp.org/www-community/Security_Headers |

La alerta sobre "X-Content-Type-Options Header Missing" indica que la aplicación web no está enviando la cabecera HTTP X-Content-Type-Options en sus respuestas HTTP. Esta cabecera es una medida de seguridad que ayuda a mitigar ciertos tipos de ataques, especialmente aquellos relacionados con la interpretación del tipo de contenido por parte del navegador.

La función principal de la cabecera X-Content-Type-Options es prevenir ataques de tipo MIME sniffing. MIME sniffing es cuando un navegador intenta adivinar el tipo de contenido de un recurso basándose en su contenido real en lugar de confiar en la cabecera Content-Type proporcionada por el servidor. Esto puede ser explotado por atacantes para ejecutar ataques de cross-site scripting (XSS) u otros ataques.

Para abordar esta alerta, debes configurar la cabecera X-Content-Type-Options en tus respuestas HTTP. El valor típicamente recomendado para esta cabecera es nosniff, que instruye al navegador a no realizar el MIME sniffing y confiar en la cabecera Content-Type proporcionada por el servidor.

**5.**

### User Agent Fuzzer

| | |
|---|---|
| **Source** | raised by an active scanner (User Agent Fuzzer) |
| **Reference** | ▪ https://owasp.org/wstg |

Un "User Agent Fuzzer" es una herramienta o script diseñado para probar la seguridad y robustez de una aplicación web al manipular o modificar el User Agent que se envía en las solicitudes HTTP. El User Agent es una cadena de texto que identifica el navegador web y su versión, así como a veces el sistema operativo y otros detalles del cliente. Los servidores web utilizan esta información para adaptar la respuesta según el navegador y el dispositivo del usuario.

El objetivo principal de un User Agent Fuzzer es verificar cómo la aplicación web responde a diferentes User Agents y si hay alguna vulnerabilidad de seguridad relacionada con la manipulación del User Agent. Aquí hay algunas maneras en las que se puede utilizar un User Agent Fuzzer:

Detección de Vulnerabilidades: Al enviar solicitudes con User Agents maliciosos o manipulados, el User Agent Fuzzer puede ayudar a detectar posibles vulnerabilidades de seguridad, como la ejecución de código malicioso, inyección de código, o problemas de manejo de entradas no válidas.

Evaluación de la Seguridad del Cliente: La aplicación web puede tomar decisiones basadas en el User Agent, como cargar diferentes scripts o recursos según el navegador. Al fuzzear el User Agent, se puede evaluar cómo la aplicación responde a diferentes clientes y si hay riesgos asociados con estas decisiones.

Prevención de Ataques de Fingerprinting: Al enviar User Agents variados, un Fuzzer puede intentar prevenir técnicas de fingerprinting donde los atacantes intentan identificar tecnologías específicas utilizadas por la aplicación web.

Es importante señalar que, al utilizar un User Agent Fuzzer, debes hacerlo de manera ética y con permisos adecuados para probar la seguridad de la aplicación. Además, ten en cuenta que algunas formas de manipulación del User Agent pueden violar los términos de servicio de ciertos sitios web, por lo que siempre es esencial seguir las políticas y reglas aplicables.