



PAI-4. VULNAWEB. AUDITORÍA DE SEGURIDAD EN SISTEMAS INFORMÁTICOS Y ANÁLISIS DE VULNERABILIDADES EN APLICACIONES WEB PARA EMPRESA DE COMERCIO ELECTRÓNICO.

Introducción

Dentro del **Plan Director para la Seguridad de la Información** de la empresa cliente se han desarrollado ya varios proyectos de seguridad de la información que abordaban los problemas de la seguridad de las transmisiones de la información entre los sistemas informáticos, y que generalmente se realizan a través de redes públicas cableadas o no cableadas.



Nos requiere con este proyecto la empresa de comercio electrónico, tanto para el **aseguramiento de los sistemas informáticos y las aplicaciones Web que están dando soporte a su negocio, como para los sistemas informáticos y aplicaciones cliente (browser) que tienen disponibles los usuarios clientes para conectarse por https a dicha empresa.**

En la empresa cliente se utilizan equipos clientes de sobremesa y dispositivos móviles (Android) tales como los que aparecen en la figura de arriba y servidores HTTP Apache que disponen de aplicaciones Web para ofrecer/comprar los productos/servicios a sus clientes/usuarios. Sin embargo, en la empresa cliente son conocedores que tanto los usuarios de la plataforma de comercio electrónico como la propia organización no son todavía totalmente conscientes de los problemas relacionados con la seguridad que existen en **los endpoints y las aplicaciones instaladas en ellos para intervenir en las transmisiones de la información. Un fallo de seguridad tanto en los sistemas informáticos que usan las aplicaciones clientes/servidoras por no estar correctamente configuradas, desarrolladas y por no validar/tratar adecuadamente las entradas/salidas a las mismas** puede conllevar no sólo la pérdida de datos por la empresa con el consecuente perjuicio de reputación y pérdidas económicas, sino también unas importantes sanciones económico-administrativa para la misma.

Es por tanto fundamental llevar a cabo **un análisis que permitan auditar los sistemas informáticos que se usan (equipos de sobremesa, servidores y dispositivos móviles) y las aplicaciones correspondientes** por si estos/éstas presentan vulnerabilidades que puedan provocar ataques.

Existen herramientas específicas que nos permiten **identificar ciertas vulnerabilidades típicas** en las aplicaciones Web tales como pueden ser la **inyección de código malicioso, cross-site scripting, cross-site request forgery, path traversal**, etc.; entre ellas podríamos citar:

- *WebScarab NG*
- *OWASP ZAP*
- *Burp Suite*
- *Vega*
- *Wa3f*
-

NOTA MUY IMPORTANTE: El mal uso de estas herramientas puede llevar a situaciones de pérdida de la **disponibilidad, confidencialidad e integridad de la información**, bloqueo, lentitud en la velocidad de conexión e incluso denegación de servicio de las máquinas analizadas e incluso problemas a los usuarios del servicio. **Estas herramientas no deben ser utilizadas contra servidores en producción para los que no se encuentre debidamente autorizado. El único fin de este proyecto es didáctico, con el objeto de dar a conocer cómo realizar auditorías de seguridad informática de sistemas informáticos y aplicaciones Web, y se debe evitar el uso de dichas herramientas para otros fines ilegales o no lícitos.**

Objetivos del Proyecto

Teniendo en cuenta todo lo anteriormente comentado por la empresa cliente, los objetivos del proyecto son los siguientes:

- Realizar la auditoría de seguridad para sistemas informáticos de sobremesa, portátiles y dispositivos móviles que utilizan los usuarios de la empresa de comercio electrónico.
- Realizar la configuración de una herramienta de escaneo de vulnerabilidades Web y que permita además llevar a cabo trazabilidad de las peticiones/respuestas HTTP/HTTPS
- Auditir las aplicaciones Web que dan soporte a las actividades de la empresa cliente, identificando las posibles vulnerabilidades de seguridad, tales como Inyecciones, XSS, CSRF, Path Traversal, etc... y mostrar las pruebas que ha realizado (mientras nos llega la autorización de la empresa de comercio electrónico, indicando todos los aspectos a tener en cuenta en la auditoría Web, se realizarán pruebas sobre determinado servidor Web vulnerable de prueba en local).

Tareas para desarrollar

Tarea 1. Auditoría de Seguridad y Bastionado de un determinado sistema informático de sobremesa

Cada **Security Team** de INSEGUS, puede utilizar la herramienta que estime más oportuna para obtener el **índice de hardening** que tiene el equipo informático de sobremesa concreto y realizar las acciones correspondientes para alcanzar el **índice de hardening** requerido por la Política de Seguridad de la empresa cliente. Por si el Security Team no tiene prevista alguna herramienta concreta, una posible herramienta para ello, que le presentamos desde INSEGUS, es la **herramienta Lynis que permite realizar auditorías bastante completas sobre seguridad de servidores y sistemas informáticos con sistemas operativos Linux o Unix**. Esta herramienta funciona con las distribuciones Linux más conocidas y usadas como *Ubuntu, Arch, Debian, Fedora y OpenSUSE*. La herramienta escanea el sistema informático buscando el software instalado y determinando el cumplimiento con los estándares y las buenas prácticas en

seguridad de la información. También detecta errores de configuración. Finalizado el escaneado muestra un informe con la información correspondiente.

Se puede descargar **Lynis** desde su página web, instalarlo y ejecutarlo. En ciertas distribuciones no hay ya necesidad de instalación viene instalado por defecto. Podemos probar un escaneo rápido desde la terminal con el siguiente comando:

- ***sudo lynis -Q*** (la *q* proviene de rápido-quick)

Si se desea un auditoria más completa usar check-all (o -c). **Lynis** analizará el gestor de arranque del sistema auditado, los servicios que arrancamos al iniciar y diferentes aspectos sobre el kernel. Además veremos información sobre:

- Herramientas del Sistema
- Procesos y memoria
- Usuarios, Grupos y métodos de autenticación
- Shells
- Sistema de Archivos
- Almacenamiento
- Sistemas de Archivos en la Red, NFS
- Aplicaciones
- ...

Estos conjuntos de comprobaciones para analizar la seguridad del sistema informático se muestran al final por pantalla. Cada resultado **debe interpretarse por el correspondiente auditor y comprobar que significa**. La salida de los test es [OK] o [WARNING], donde el primero es considerado como un resultado **ESPERADO**, y el segundo es un resultado **NO ESPERADO (de acuerdo a la política, no quiere decir que sea malo)**. También aparecen [NO FOUND], como que no se ha encontrado el objeto de testeo. Además, información adicional sobre las comprobaciones realizadas se guardan en un fichero de logs que por defecto es */var/log/lynis.log*, que es actualizado en cada ejecución. Esta información será muy útil para ver que hace la herramienta en background o dónde las anomalías aparecen (y frecuentemente porqué).

El informe presenta antes de mostrar el **hardening index (métrica para el grado de aseguramiento)** un conjunto de sugerencias tales como:

- [02:33:38] Suggestion: update to the latest stable release.
- [02:33:41] Suggestion: Install a PAM module for password strength testing like pam_cracklib or pam_passwdqc [test:AUTH-9262]
- [02:33:41] Suggestion: When possible set expire dates for all password protected accounts [test:AUTH-9282]
- [02:33:41] Suggestion: Configure password aging limits to enforce password changing on a regular base [test:AUTH-9286]
- [02:33:41] Suggestion: Default umask in /etc/profile could be more strict like 027 [test:AUTH-9328]
- [02:33:41] Suggestion: Default umask in /etc/login.defs could be more strict like 027 [test:AUTH-9328]
- [02:33:41] Suggestion: Default umask in /etc/init.d/rc could be more strict like 027 [test:AUTH-9328]

- [02:33:41] Suggestion: To decrease the impact of a full /tmp file system, place /tmp on a separated partition [test:FILE-6310]
- [02:33:42] Suggestion: Disable drivers like USB storage when not used, to prevent unauthorized storage or data theft [test:STRG-1840]
- [02:33:42] Suggestion: Disable drivers like firewire storage when not used, to prevent unauthorized storage or data theft [test:STRG-1846]
- [02:33:54] Suggestion: Install package apt-show-versions for patch management purposes [test:PKGS-7394]
- [02:33:55] Suggestion: Configure a firewall/packet filter to filter incoming and outgoing traffic [test:FIRE-4590]
- [02:33:56] Suggestion: Add legal banner to /etc/issue, to warn unauthorized users [test:BANN-7126]
- [02:33:56] Suggestion: Add legal banner to /etc/issue.net, to warn unauthorized users [test:BANN-7130]

Posteriormente presenta los nombres de los ficheros con los informes y el **hardening index**:

Files:

- Test and debug information : /var/log/lynis.log
- Report data : /var/log/lynis-report.dat
- Current version : 129 Latest version : 139
=====
- Hardening index : [44] [#####]

Se observa que el **Hardening index** es 44 y la **Política de Seguridad de la empresa dice que los sistemas informáticos que se usen en la misma deben tener un Hardening Index igual o superior a 68 según esta herramienta.** Para ello se plantean las siguientes acciones:

- Realizar las acciones que se indican en las sugerencias de la herramienta para mejorar el **Hardening Index** en el caso que sea **menor de 68**.
- Realizar acciones teniendo en cuenta los WARNING:
 - **Resolver el problema:** Leyendo el fichero de logs acerca de los detalles técnicos.
 - **Deshabilitar el test no de interés de la lista:** Dentro del perfil de escaneado, los tests se pueden deshabilitar porque no estés interesado en ello, con la opción= **test_skip_always**.

Desde INSEGUS, los miembros de cada **Security Team** se suponen que son potenciales clientes de dicha empresa de comercio electrónico. Se trataría entonces en esta tarea de aplicar a cada sistema informático (en este caso que usted disponga o del laboratorio de prácticas) la correspondiente auditoría para alcanzar el **hardening index** especificado en la Política de Seguridad (**no inferior a 68**), mediante las correspondientes actuaciones que **tienen que ser todas documentadas en el informe final de este proyecto. Indique también las razones por las cuales ha seleccionado unas determinadas acciones y no otras.**

Además, en cada uno de los equipos informáticos del **Security Team** debería estar configurado el browser, que se vaya a usar para conectarse con la empresa de comercio electrónico, con la configuración segura para llevar a cabo la comunicación con dicho servidor. **En la sesión de seguimiento del Proyecto se comprobará dicha configuración por INSEGUS y en el informe del Proyecto se indicará las acciones llevadas a cabo para realizar una configuración “hard” de dicho browser.**

Tarea 2. Auditoría de Seguridad de un determinado dispositivo móvil

Cada **Security Team** de INSEGUS, puede utilizar la herramienta que estime más oportuna para obtener **una configuración “hard”** de dicho dispositivo móvil y realizar las acciones correspondientes para alcanzar el **hardening** requerido por la Política de Seguridad de la empresa cliente. Por si el **Security Team** no tiene prevista alguna herramienta concreta, una posible herramienta para ello, que le presentamos desde INSEGUS, es la app denominada **CONAN mobile** que surge en el proyecto europeo "[Advanced Cyber Defense Centre \(ACDC\)](#)" en el que **Instituto Nacional de Ciberseguridad de España (INCIBE)** participa, y que tiene como objetivo la detección temprana y lucha contra las amenazas generadas por las redes botnets para mejorar la ciberseguridad de España y Europa. **CONAN mobile** que permite realizar un análisis de los dispositivos móviles verificando los parámetros de configuración y las aplicaciones instaladas. Para ejecutar este análisis el usuario una vez instalada la aplicación en el dispositivo puede usar el botón "**Analizar mi dispositivo**" que aparece en la pantalla inicial.

Durante la realización de este análisis **CONAN mobile** necesita para ello disponer de una conexión a Internet para realizar consultas a la base de conocimiento de INCIBE. Una vez finalizado el análisis, se muestra la información sobre el estado de seguridad del dispositivo. Esta información está dividida en varias pantallas.



Como se observa en la figura anterior el resultado del análisis está dividido en cuatro secciones:

- **Configuración:** describe los problemas de configuración encontrados
- **Aplicaciones:** muestra incidencias detectadas en las aplicaciones instaladas.
- **Permisos:** Acceso a los permisos declarados por las aplicaciones por orden de peligrosidad.
- **Servicio Proactivo**, eventos de seguridad detectados y eventos sobre las conexiones realizadas por las aplicaciones del dispositivo, así como información extendida de las mismas

Como los miembros del **Security Team** se suponen que son potenciales clientes de dicha empresa de comercio electrónico. Se trataría entonces en esta tarea de aplicar a cada dispositivo móvil del **Security Team** de la correspondiente auditoría para alcanzar el **hardening**, especificado en la Política de Seguridad de la empresa cliente que nos indica lo siguiente:

Para la seguridad de las transmisiones con esta empresa de comercio electrónico a través de dispositivos móviles se requiere:

Respecto a la configuración del dispositivo:

- No tener habilitada la instalación de software de orígenes desconocidos
- No tener el dispositivo móvil *rootead o jailbreak*
- Tener instaladas todas las actualizaciones del fabricante del dispositivo del sistema operativo y de la aplicación browser
- No conectarse a redes Wifi sin seguridad adecuada o con SSID oculta
- No tener activado el dispositivo Bluetooth durante las conexiones Web

Respecto a las aplicaciones instaladas

- No tener instaladas aplicaciones maliciosas o sospechosas de serlo.
- No tener abiertas conexiones de red de aplicaciones diferentes al browser y que se consideren sospechosas
- No tener habilitados permisos de otras aplicaciones para lectura de información sobre lo que el browser se descarga/envía.

Además, en cada uno de los dispositivos móviles del **Security Team** debería estar configurado el browser, que se vaya a usar para conectarse con la empresa de comercio electrónico, con la configuración segura para llevar a cabo la comunicación con dicho servidor. **En la sesión de seguimiento del Proyecto se comprobará dicha configuración por INSEGUS y en el informe del Proyecto se indicará las acciones llevadas a cabo para realizar una configuración “hard” de dicho browser.**

Tarea 3. Auditoría de vulnerabilidades en aplicaciones Web y Trazabilidad de peticiones/resuestas para los protocolos HTTP/HTTPS

Cada **Security Team** de INSEGUS, puede utilizar la herramienta que estime más oportuna para auditar las vulnerabilidades de las aplicaciones Web de la empresa cliente y realizar las acciones correspondientes para alcanzar el **hardening** requerido por la Política de Seguridad de dicha empresa y estándares de buenas prácticas en seguridad Web. Por si el **Security Team** no tiene prevista alguna herramienta concreta, les proponemos desde INSEGUS utilizar alguna de las herramientas **Dynamic Application Security Testing (DAST)** que usan la **metodología de testing de seguridad de caja-negra** en el que la aplicación Web es auditada desde fuera, como si fuera un posible atacante. Estas herramientas complementan a las de **Static Application Security Testing (SAST)**, que usan una metodología de testing de caja-blanca, donde el auditor examina la aplicación Web desde dentro, buscando en el código fuente los aspectos que expliquen la presencia de ciertas vulnerabilidades de seguridad.

En concreto, una herramienta para llevar a cabo DAST sería la herramienta **OWASP ZAP**, que es un marco de trabajo para la trazabilidad de peticiones/resuestas HTTP/HTTPS y analizar posibles vulnerabilidades de las aplicaciones web que se comunican con los clientes usando estos protocolos. Su uso más común es operar **como un proxy de intercepción, que permite al operador revisar y modificar las peticiones creadas por el navegador antes de que sean enviadas al servidor, y para revisar y modificar respuestas enviadas por el servidor antes de que sean recibidas** por el navegador.

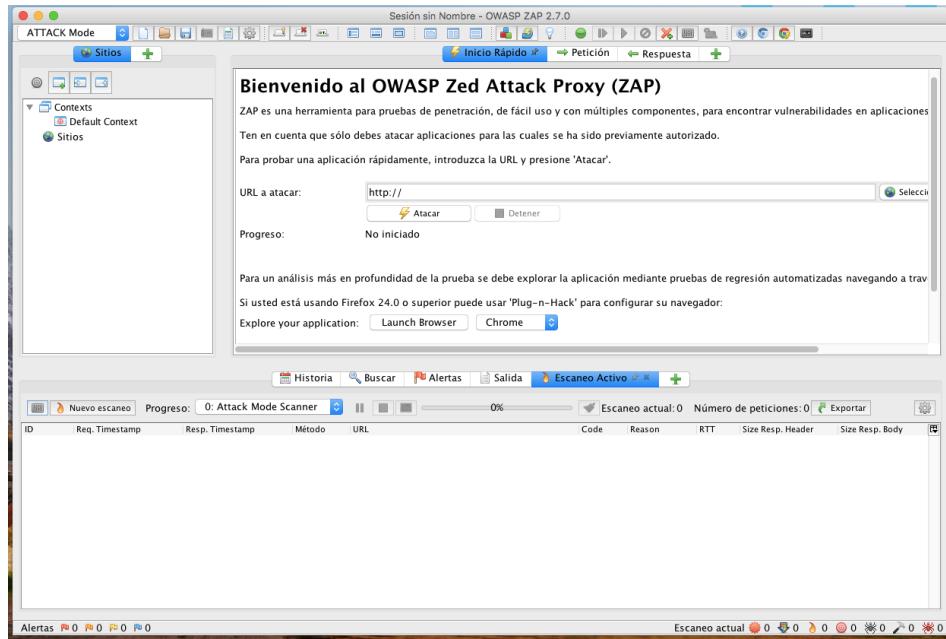


Figura 1: Interfaz de OWASP ZAP

Configuración de la herramienta OWASP ZAP y del cliente Web

En primer lugar, para poder interceptar el tráfico (peticiones/respuestas) hacia una aplicación web vamos a configurar la herramienta para que actúe de proxy. En la herramienta en la pestaña “Proxy” tendremos que configurar los siguientes aspectos:

1. OWASP ZAP: Preferencias

- **Address:** dirección IP de escucha
- **Port:** puerto de escucha.

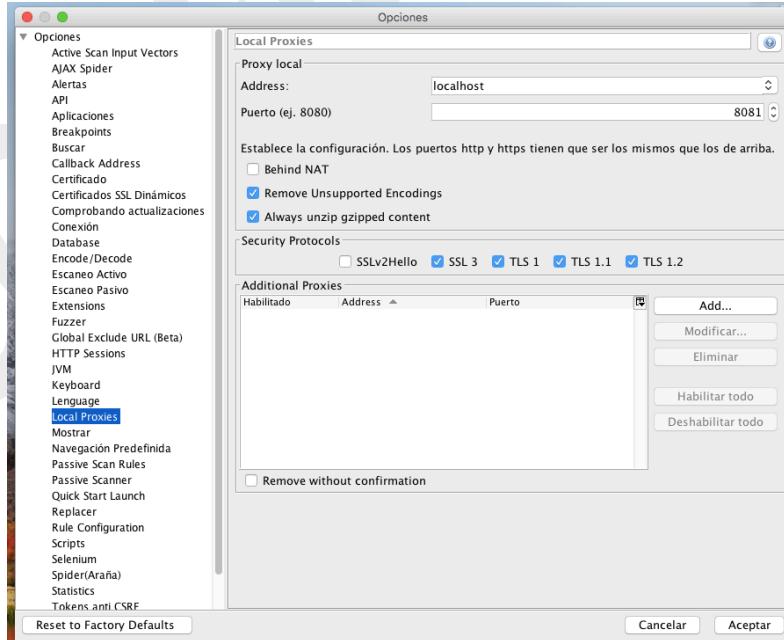


Figura 2: Configuración del proxy en OWASP ZAP

Con estas opciones ya tenemos preparada la herramienta para escuchar todo el tráfico en la dirección/puerto indicadas. Toda petición saliente será interceptada por nuestra herramienta pudiendo así interactuar con las peticiones.

Para comprobar que nuestro *proxy* está configurado correctamente, podemos lanzar una instancia de un cliente web y usar la dirección localhost:8081 donde está escuchando nuestro proxy.

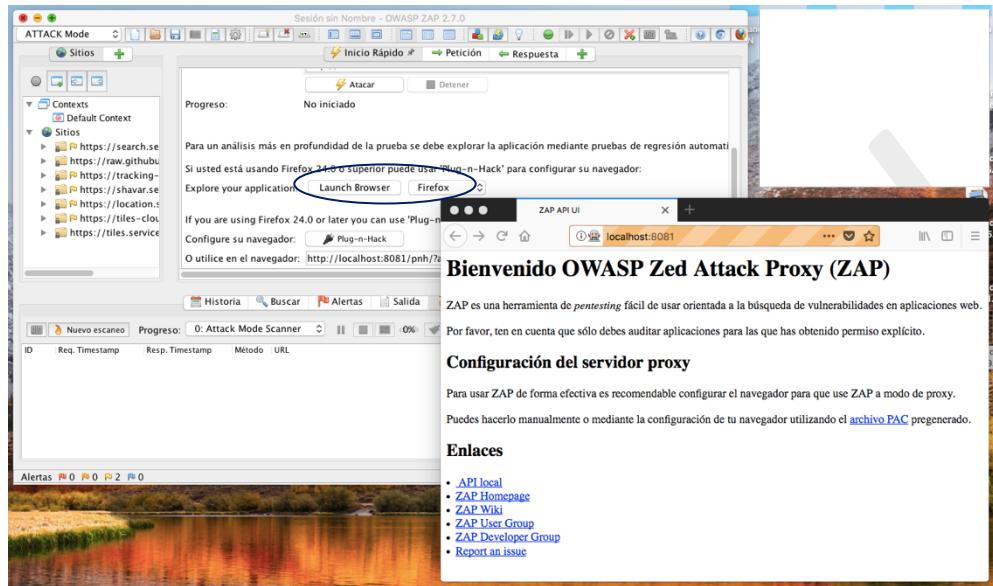


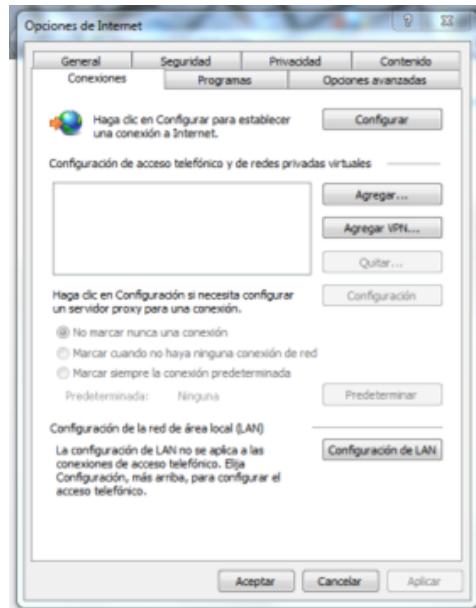
Figura 3: Proxy ZAP funcionando

Podemos usar esta instancia de navegador para realizar las pruebas o configurar nuestro navegador habitual para realizar las pruebas, para ello será necesario configurarlo adecuadamente.

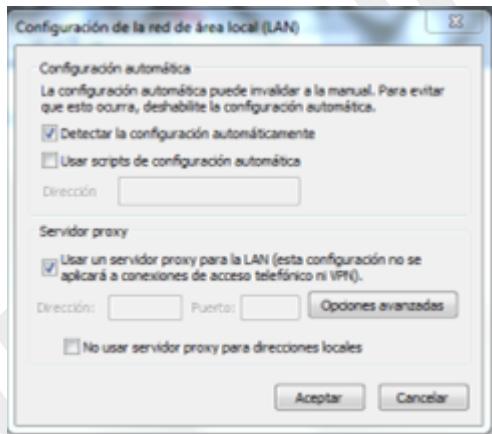
Configuración externa del cliente web

Una vez configurada la herramienta tendremos la posibilidad también que configurar nuestro cliente web para que todas las peticiones desde nuestro navegador se redirijan al proxy de la herramienta. Mostraremos el ejemplo a desarrollar para un navegador *Internet Explorer* sin embargo para otros tipos de navegadores la configuración es bastante parecida.

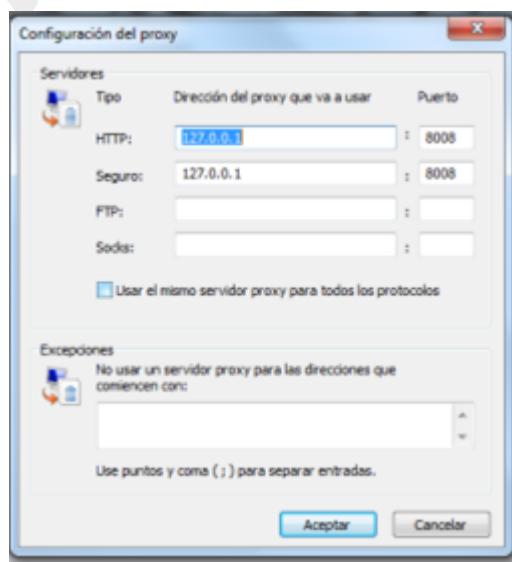
Paso 1: Opciones de Internet → Configuración LAN



Paso 2: Marcamos la opción de usar Proxy y saltamos a “Opciones avanzadas”



Paso 3: Configuración del proxy, donde introduciremos los datos de la dirección IP/puerto que hayamos configurado en ZAP



Una vez realizada esta configuración en nuestro cliente web (“navegador”), las peticiones/resuestas HTTP/HTTPS pasarán a través del proxy de la herramienta OWASP ZAP y se podrán visualizar como se muestra a continuación en la figura 4. En esta ventana, también se puede además de visualizar la petición, modificar los datos de dicha petición y aceptar los cambios, cancelarlos, abortar la petición, y cancelar todas las interceptaciones.

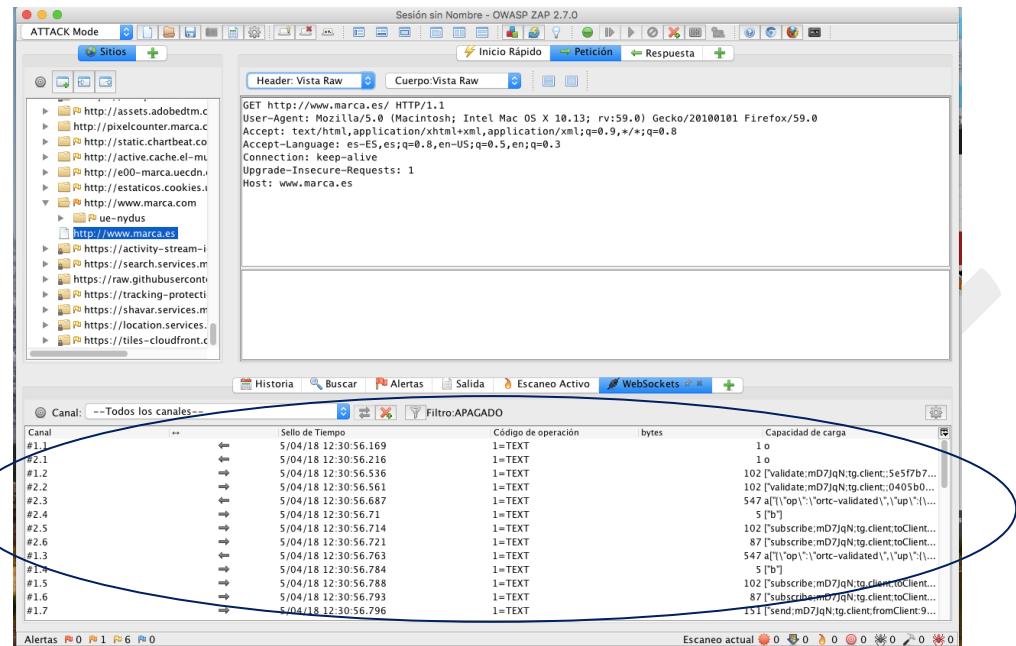


Figura 4: Ventana de interceptación de petición HTTP

Trazabilidad del tráfico HTTP/HTTPS con OWASP ZAP

Si realizamos cualquier petición desde nuestro cliente web, ya desde OWASP ZAP podemos inspeccionar estas peticiones usando la pestaña **History**.

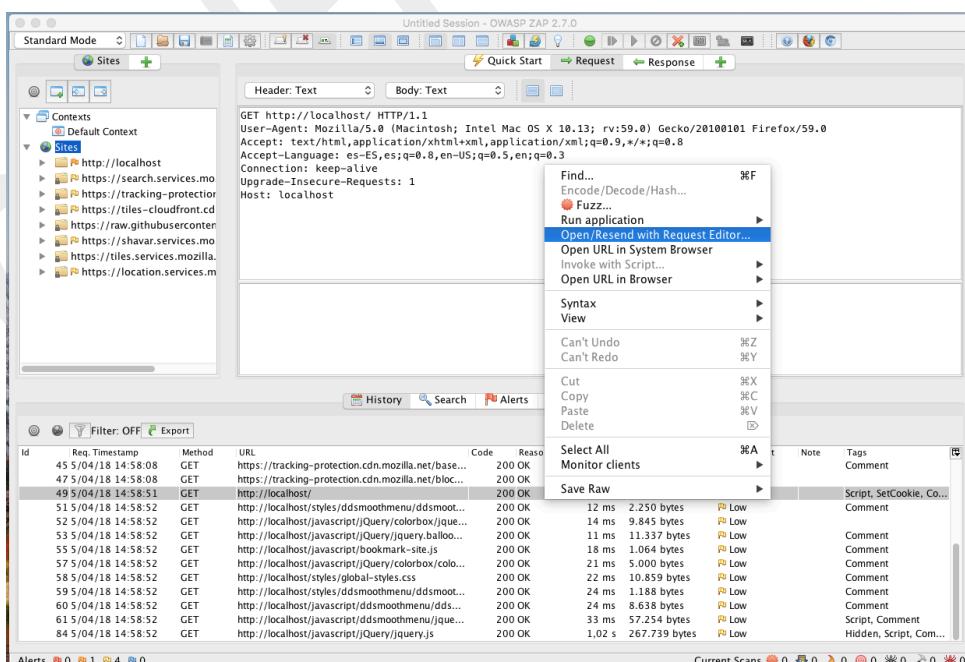


Figura 5: Visionado de peticiones

Haciendo doble-click en algunas de las peticiones realizadas se nos mostrará en la parte superior donde tenemos “Request”. Ya desde aquí podemos ver la información de una petición http y además podemos entrar en más detalles de esta incluso editarla. Desde la sección de “Request” con el botón derecho de ratón, en el menú contextual de “Open/Resend with Request Editor ...” nos permitirá tener una vista más detallada de la petición incluso editarla.

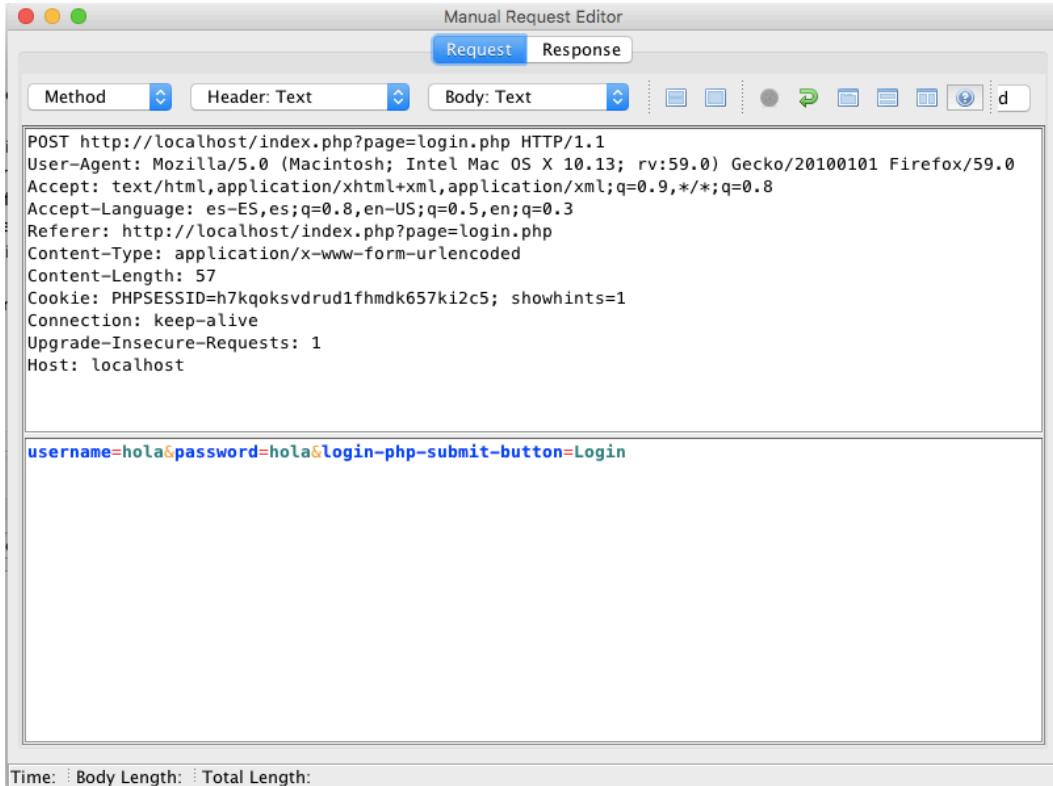


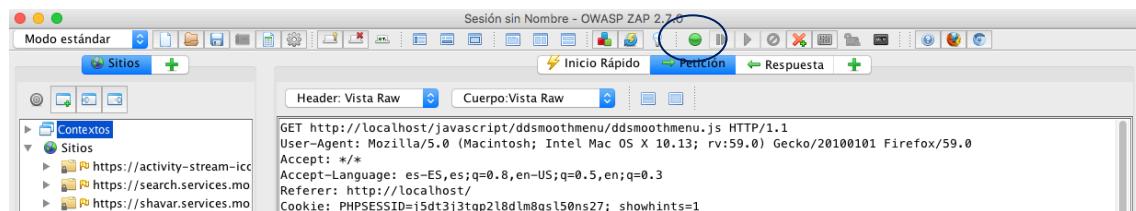
Figura 6: Ventana de inspección / edición de peticiones http.

En la figura anterior podemos ver dos partes:

- Primera parte: Información sobre la cabecera, en la figura podemos ver la cabecera de una petición POST para un login.
- Segunda parte: Información sobre los parámetros de la petición que se están enviando con la petición, en este caso login y password.

Modo intercepción de peticiones con OWASP ZAP

Es posible activar un modo especial de ejecución donde capturaremos las peticiones y antes de ser respondidas por el servidor podremos visionar la petición incluso modificarla y reenviarla al servidor modificadas. Este es un caso de ejemplo de Man-in-the-Middle (MitM). Para activar el modo de intercepción sólo tenemos que activar un “Breakpoint”, desde el botón verde situado en la barra de menús de ZAP.



Una vez activada dicha opción ZAP se pondrá en modo recepción de peticiones y podremos capturar peticiones, vamos a mostrar un ejemplo de una petición de login hecha a través del proxy de ZAP puesto en modo interceptación:

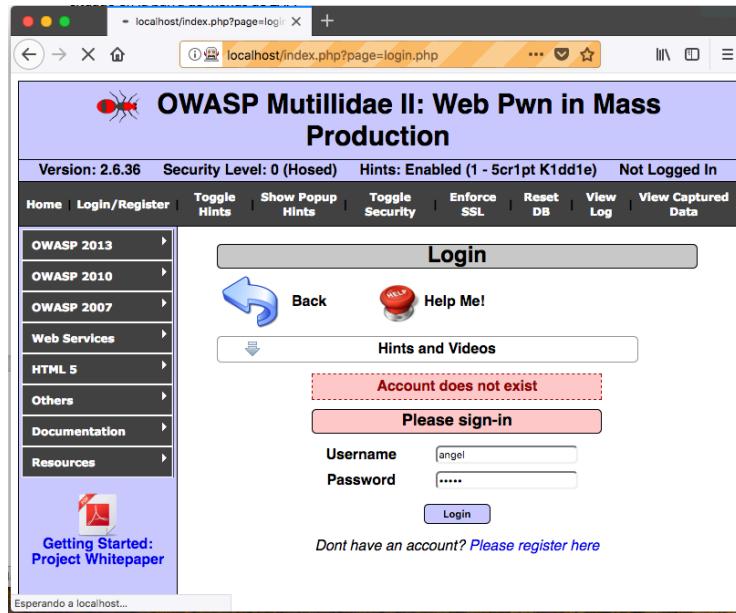


Figura 7: Login de la aplicación web Mutillidae de OWASP pasando por proxy de ZAP

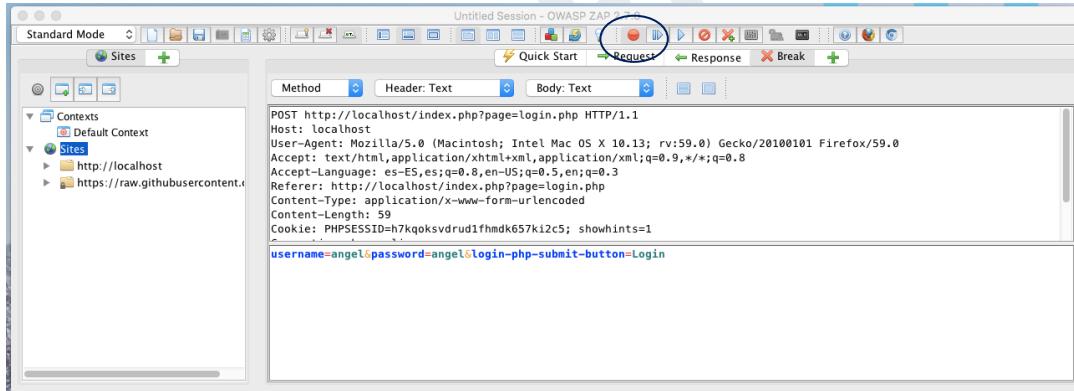


Figura 8: Captura de la petición http con los parámetros

En la parte de los parámetros podemos no sólo observar el nombre de las variables usadas en la aplicación, sino que también podemos ver el valor que toman dichas variables. Como se ha comentado anteriormente, esta ventana nos permite editar la petición y podríamos cambiar el valor de cualquiera de los parámetros de la petición insertando otros valores.

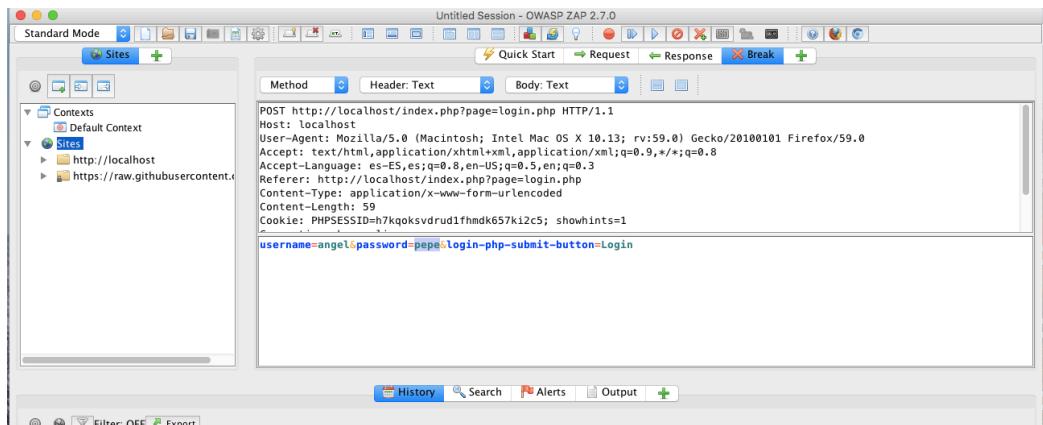


Figura 9: Petición modificada en el parámetro de password.

Una vez modificada la petición simplemente tendremos que darle salida a la petición pulsando en botón en la barra de menús, y tendremos la petición de respuesta “Response”

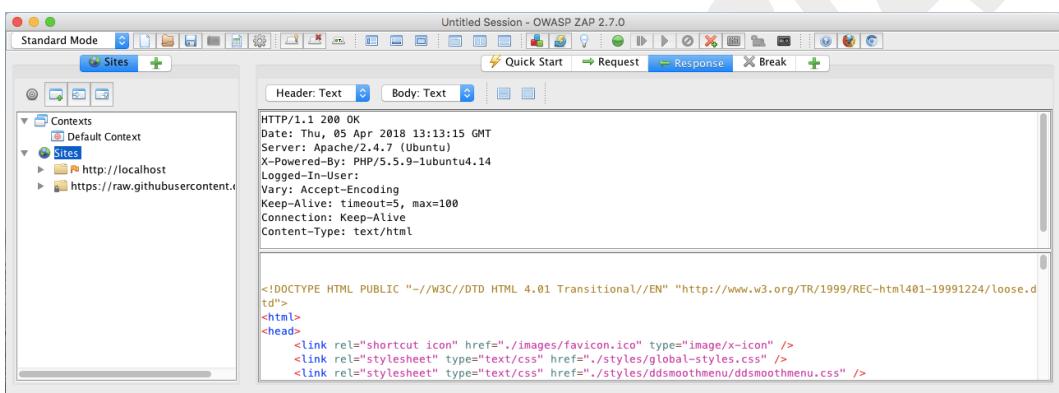


Figura 10: Petición de respuesta una vez cambiado los parámetros de la petición

A la vista de lo anteriormente expuesto, presente a INSEGUS un conjunto de pruebas que muestre **las posibilidades de una de las herramientas seleccionadas de entre las propuestas para comprobar la correcta trazabilidad de las peticiones HTTP/HTTPS sin modificación alguna de los parámetros de las peticiones y las respuestas obtenidas desde el servidor de prueba sin realizar modificación alguna. Muestre también las pruebas relacionadas con la robustez de las cookies de sesión, si ello fuera posible.**

Tarea 4. Análisis de vulnerabilidades Web para Servidor de Pruebas

Primeramente, reiterar a los ingenieros/as que **el uso de esta herramienta para el análisis de vulnerabilidades se debe realizar siempre sobre aplicaciones Web que no se encuentren en producción, por ejemplo, en aplicaciones en desarrollo propias, en entornos simulados (no reales) o de prueba de concepto.** Cualquier otro tipo de actividad relacionada con el análisis de vulnerabilidades Web que no sean sobre los anteriores entornos **podría conllevar infracciones legales** por el uso de esta herramienta, además de posibles repercusiones sociales y económicas difíciles de determinar. Por ello existen algunos servidores Web de pruebas con vulnerabilidades y entre ellos están **Gruyere, WebGoat, OWASP Mutillidae II.**

Entre las vulnerabilidades más típicas en aplicaciones Web se encuentran **la inyección SQL, Path Traversal, XSS reflejado y almacenado (posiciones A1, A5 y A7 respectivamente en OWASP Top Ten 2017).** Estas vulnerabilidades habrá que **intentar evitarlas en los entornos Web antes de ponerlos en producción** ya que podrían permitir extraer información confidencial de las aplicaciones web y/o permitir ataques a los clientes, pues por ejemplo los campos en los que

son introducidos determinados valores en los formularios no son validados correctamente por el servidor o enviados al cliente.

Modo testeо de inyecciones con OWASP ZAP

Para realizar las pruebas de inyección no es necesario probar uno a uno las diferentes cadenas/payloads, ya que OWASP ZAP está provisto de un “**Fuzzer**” que nos permite una vez interceptada una petición, repetir la misma con diferentes parámetros de entrada. Para ello sobre cualquier petición “Request” que disponga de parámetros le damos botón derecho y tenemos la opción “Fuzz”:

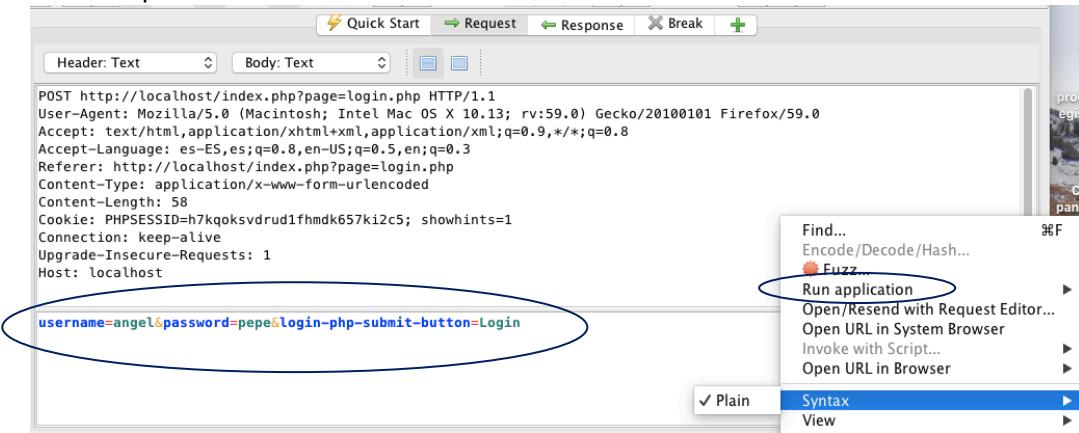
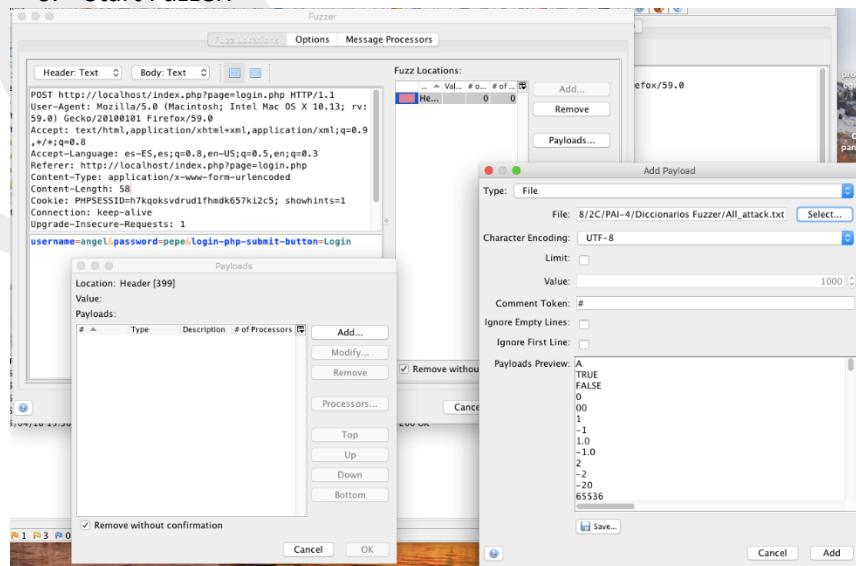


Figura 11: Acceso al Fuzzer desde una petición

Una vez pulsado en la opción de “Fuzz” ya tendremos acceso al Fuzzer que nos permitirá configurar que parámetros usaremos para introducir las diferentes cadenas de búsqueda para probar en las peticiones, y seguimos los siguientes pasos:

1. Seleccionar el parámetro donde introduciremos los payloads.
2. Pulsamos en Payloads ...
3. Seleccionamos el tipo, Type, File, y seleccionamos el archivo donde esté las cadenas.
4. Pulsamos Add, en la ventana de Add Payload
5. Pulsamos OK, en la ventanas de Payloads.
6. Start Fuzzer.



También ZAP viene provisto con un conjunto de diccionarios de payloads precargados para hacer pruebas de inyecciones. En este caso en lugar de marcar en el paso 3 que vamos a usar un fichero, marcamos la opción de “**File Fuzzer**”. Como podremos observar dentro de la opción de “jbrofuzz” tenemos una sección de “SQL Injection” y dentro tenemos varios diccionarios de payloads predefinidos, podemos seleccionar alguno de estos para hacer las correspondientes pruebas.

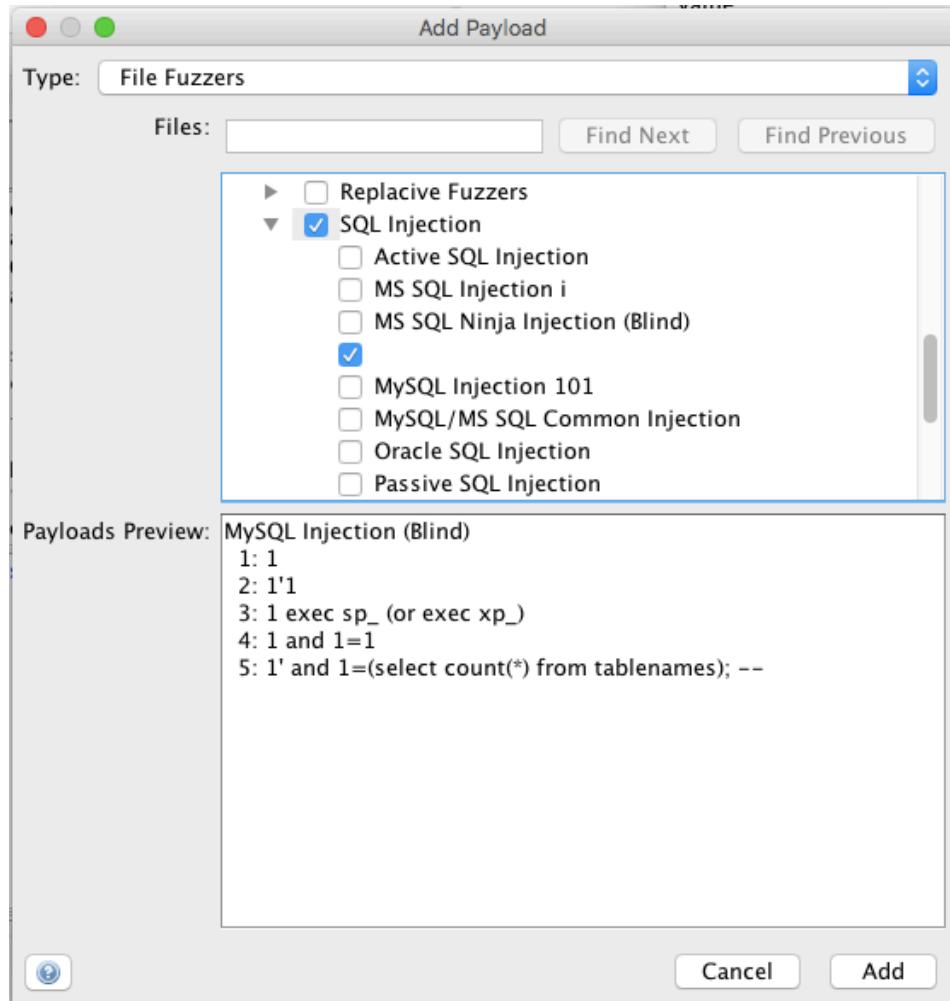


Figura 12: Lista de ficheros de payloads predefinidos en ZAP.

El **Fuzzer** se encargará de usar todas las cadenas del diccionario que hayamos introducido y tomar las respuestas, mostrándonos una pantalla de resultados. Será trabajo nuestro identificar qué peticiones han dado resultados satisfactorios y ver qué cadenas son las más interesantes. Todas aquellas peticiones marcadas como “Reflected” en su “Status” o como “Found” en su “Reason” son peticiones interesantes por sus resultados. Entre ellas podremos encontrar cosas interesantes.

Task ID	Message Type	Code	Reason	RTT	Size Resp. Header	Size Resp. Body	Highest Alert	State	Payloads
27 Fuzzed		200 OK		182 ms	245 bytes	53.129 bytes	Reflected	1	
28 Fuzzed		200 OK		196 ms	245 bytes	54.960 bytes	Reflected	1'1	
29 Fuzzed		200 OK		193 ms	245 bytes	53.129 bytes	Reflected	1 exec sp_(or e...)	
30 Fuzzed		200 OK		188 ms	245 bytes	53.129 bytes	Reflected	1 and 1=1	
31 Fuzzed		200 OK		236 ms	250 bytes	55.095 bytes	Reflected	1' and 1=(select...)	
32 Fuzzed		200 OK		259 ms	250 bytes	53.170 bytes	Reflected	a	
33 Fuzzed		200 OK		183 ms	250 bytes	53.170 bytes	Reflected	1 or 1=1	
34 Fuzzed		302 Found		76 ms	324 bytes	53.170 bytes	Reflected	1' or 1='1	
35 Fuzzed		200 OK		76 ms	250 bytes	55.061 bytes	Reflected	1 and user_nam...	
36 Fuzzed		200 OK		150 ms	250 bytes	53.170 bytes	Reflected	1	

Figura 13: Resultados del Fuzzer

De acuerdo con todo esto, en el informe final del proyecto **presentará un entorno simulado o de desarrollo propio que ha utilizado para hacer las correspondientes pruebas**, y los resultados obtenidos en las mismas. Realizará una detección y análisis de las vulnerabilidades y **presentará todos los resultados obtenidos y las posibles recomendaciones (plan de mitigación) para evitar todo este tipo de vulnerabilidades Web**, al menos para la **inyección SQL, Path/Directory Traversal, XSS reflejado y almacenado (posiciones A1, A5 y A7 respectivamente en OWASP Top Ten 2017)**.

Normas del entregable

- Cada **Security Team** debe entregar a través de la Plataforma de Enseñanza Virtual y en la **actividad** preparada para ello un archivo zip, nombrado **PAI4-SecurityTeamX.zip**, que deberá contener al menos los ficheros siguientes:
 - ✓ **Documento en formato pdf que contenga un informe/resumen del proyecto** con los detalles más importantes de las decisiones, soluciones adoptadas y/o implementaciones desarrolladas, así como el resultado y análisis de las pruebas realizadas (máximo 10 páginas).
- **Código fuente de las posibles implementaciones y/o scripts desarrollados y/o configuraciones y/o logs del analizador de código.**
- El plazo de entrega de dicho proyecto finaliza el **día 27 de abril a las 21:30 horas**.
- Los proyectos entregados fuera del plazo establecidos serán considerados inadecuados por el cliente y por tanto entrarán en penalización por cada día de retraso entrega de 5% del total, hasta agotarse los puntos.
- **El cliente no se aceptará envíos realizados por email, ni mensajes internos de la enseñanza virtual, ni correo interno de la enseñanza virtual. Toda entrega realizada por estos medios conllevará una penalización en la entrega del 5%.**

Métricas de valoración

Para facilitar el desarrollo de los equipos de trabajo el cliente ha decidido listar las métricas que se tendrán en cuenta para valorar los entregables de cada grupo de trabajo:

- **Resumen (30%)**
 - Tamaño del informe
 - Calidad del resumen aportado
 - Calidad de pruebas presentadas y resultados
- **Solución aportada (70%)**
 - Cumplimiento de requisitos establecidos por la empresa cliente
 - Calidad de las auditorías realizadas y del hardening alcanzado
 - Respuesta al conjunto de consultas planteadas
 - Pruebas realizadas