UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

IMPLEMENTACIÓN DE CONTROLES A UNA APLICACIÓN WEB MEDIANTE LA METODOLOGÍA OWASP PARA EL ASEGURAMIENTO DE SU SEGURIDAD

GALLEGOS CHAMBA MONICA JANINA
INGENIERA DE SISTEMAS

MACHALA
2019

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

IMPLEMENTACIÓN DE CONTROLES A UNA APLICACIÓN WEB MEDIANTE LA METODOLOGÍA OWASP PARA EL ASEGURAMIENTO DE SU SEGURIDAD

GALLEGOS CHAMBA MONICA JANINA
INGENIERA DE SISTEMAS

MACHALA
2019

UTMACH

UNIDAD ACADÉMICA DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

EXAMEN COMPLEXIVO

IMPLEMENTACIÓN DE CONTROLES A UNA APLICACIÓN WEB MEDIANTE LA METODOLOGÍA OWASP PARA EL ASEGURAMIENTO DE SU SEGURIDAD

GALLEGOS CHAMBA MONICA JANINA
INGENIERA DE SISTEMAS

HONORES TAPIA JOOFRE ANTONIO

MACHALA, 04 DE FEBRERO DE 2019

MACHALA
04 de febrero de 2019

Nota de aceptación:

Quienes suscriben, en nuestra condición de evaluadores del trabajo de titulación denominado IMPLEMENTACIÓN DE CONTROLES A UNA APLICACIÓN WEB MEDIANTE LA METODOLOGÍA OWASP PARA EL ASEGURAMIENTO DE SU SEGURIDAD, hacemos constar que luego de haber revisado el manuscrito del precitado trabajo, consideramos que reúne las condiciones académicas para continuar con la fase de evaluación correspondiente.

HONORES TAPIA JOOFRE ANTONIO
0704811751
TUTOR - ESPECIALISTA 1

MOLINA RIOS JIMMY ROLANDO
0703691980
ESPECIALISTA 2

JUMBO CASTILLO FREDDY ANIBAL
0704167949
ESPECIALISTA 3

Fecha de impresión: lunes 04 de febrero de 2019 - 14:29

URKUND

Urkund Analysis Result

Analysed Document: Gallegos-Informe de Examen Complexivo.docx (D46926999)
Submitted: 1/17/2019 7:47:00 PM
Submitted By: mjgallegos_est@utmachala.edu.ec
Significance: 1 %

Sources included in the report:

https://www.redalyc.org/html/5122/512251501005/

Instances where selected sources appear:

1

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

La que suscribe, GALLEGOS CHAMBA MONICA JANINA, en calidad de autora del siguiente trabajo escrito titulado IMPLEMENTACIÓN DE CONTROLES A UNA APLICACIÓN WEB MEDIANTE LA METODOLOGÍA OWASP PARA EL ASEGURAMIENTO DE SU SEGURIDAD, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

La autora declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

La autora como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 04 de febrero de 2019

GALLEGOS CHAMBA MONICA JANINA
0750189664

DEDICATORIA

El presente trabajo va dedicado principalmente a Dios, por haber puesto en mí la sabiduría, perseverancia y dedicación necesaria para llegar hasta esta etapa de mi vida.

A mis padres que siempre me han brindado su apoyo incondicional y me han motivado a seguir adelante en toda circunstancia.

A mis hermanos y amigos cercanos que de alguna u otra manera me han motivado y apoyado a cumplir mis objetivos propuestos.

A mi fiel y viejo amigo Hachi, por haberme acompañado en todo el proceso Universitario.

Srta. Mónica Janina Gallegos Chamba

AGRADECIMIENTO

Agradezco a Dios, por darme su amor incondicional en todo momento y ser mi guía espiritual en cada decisión; a mis padres, por inculcarme sus valores y hacer de mí una buena persona; a mis hermanos que pusieron en mí su confianza apoyándome y motivándome permanentemente; A mis amigos, con quienes he compartido gratos momentos y me han demostrado su constante apoyo; a mi Tutor Joofre Honores por impartirme sus conocimientos y guiarme en cada proceso del presente proyecto.

Srta. Mónica Janina Gallegos Chamba

RESUMEN

IMPLEMENTACIÓN DE CONTROLES A UNA APLICACIÓN WEB MEDIANTE LA METODOLOGÍA OWASP PARA EL ASEGURAMIENTO DE SU SEGURIDAD

Las aplicaciones web han sido acogidas universalmente por un alto índice de la sociedad moderna, de la misma manera han incrementado los riesgos y con ello los ataques informáticos, una forma de prevenir esta situación es detectar las vulnerabilidades que pueden ser aprovechadas por los atacantes cibernéticos. En base a ello surge el presente proyecto que consiste en detectar vulnerabilidades y establecer controles en el sitio web Machala Restaurantes, el cual se basó en los riesgos de inyección, pérdida de autenticación y exposición a datos sensibles, pertenecientes al proyecto OWASP. La simulación del escenario fue realizada en la herramienta GNS3, en el cual se representan dos redes, en la primera se ubica el atacante y, en la otra el servidor, ambas se conectan por medio de un enrutador. Las herramientas utilizadas en la explotación de vulnerabilidades fueron las del sistema Operativo Kali Linux; sqlmap para inyección sql, BurpSuite para la pérdida de autenticación por fuerza bruta, y, Wireshark para la captura de datos sensibles. Una vez ejecutados los ataques, se logró evidenciar las vulnerabilidades del sitio web, por lo que se procedió a establecer los controles para mitigar los riesgos, en el ataque de inyección SQL se cambió la forma de acceso a la base de datos a través de la utilización del ORM del Framework Django, la pérdida de autenticación y exposición de datos sensibles se controlaron a través de la implementación de un certificado SSL. Así, los controles permitieron garantizar la seguridad de la aplicación web.

PALABRAS CLAVES: OWASP, Inyección SQL, Pérdida de autenticación, Exposición de datos sensibles, SSL, SQLmap, Wireshark, Burp Suite.

ABSTRACT

IMPLEMENTATION OF CONTROLS ON A WEB APPLICATION USING THE OWASP METHODOLOGY TO ENSURE ITS SECURITY

Web applications have been universally accepted by a high number of modern society, in the same way has increased the risks and with it computer attacks, one way to prevent this situation is to detect vulnerabilities that can be exploited by cyber attackers. This is the basis for the present project, which consists of detecting vulnerabilities and establishing controls on the Machala Restaurantes website, which was based on the risks of injection, broken authentication and exposure to sensitive data contained in the OWASP project. The scenario simulation was performed in the GNS3 tool, in which two networks are represented, in the first one the attacker is located and, in the other the server, both are connected by a router. The tools used to exploit vulnerabilities were those of the Kali Linux operating system; sqlmap for sql injection, BurpSuite for the broken authentication by brute force, and Wireshark for the capture of sensitive data. Once the attacks were executed, the vulnerabilities of the web site were identified, so controls were established to mitigate risks, in the SQL injection attack the form of access to the database was changed through the use of the ORM of the Django Framework, the broken authentication and exposure of sensitive data were controlled through the implementation of an SSL certificate. Thus, the controls made it possible to provide security for the web application.

KEYWORDS: OWASP, SQL injection, Broken Authentication, Sensitive Data Exposure, SSL, SQLmap, Wireshark, BurpSuite.

CONTENIDO

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

Pág.

1. INTRODUCCIÓN

Con el pasar de los años el internet ha ido tomando un control vertiginoso en diversas actividades cotidianas del ser humano, aspectos como la comunicación y compartición de recursos han sido factores para que este fenómeno se introduzca masivamente a nivel mundial. Dentro de los servicios que se desarrollan en este entorno denominado internet están las aplicaciones web; las cuales facilitan el acceso simultáneo de varios usuarios a las operaciones del sistema [1], esto a su vez abre un camino hacia múltiples posibilidades de acceso a la información desde cualquier lugar.

En la actualidad gran parte de las organizaciones emplean sitios web para la realización de sus diferentes operaciones, y es que no solo existe la parte informativa sino la interactiva a través de varias actividades como: compra, venta u otros servicios relacionados [2], es así que su implementación es prácticamente considerada una estrategia de comunicación entre la empresa y su cliente. Los sitios web al ser medios de interacción directa con los usuarios están propensos a tener vulnerabilidades y por ende a caer en ataques o intrusiones que puedan causar un sinnúmero de perjuicios en los involucrados. Entre los principales ataques se encuentran los de inyección en lenguaje SQL, pérdidas de autenticación, control de acceso ininterrumpido, entre otros [3]. Por el lado organizacional, inspirar confianza es un factor clave en sus estrategias comerciales, por esta razón, proteger la información se ha convertido en una situación de alta prioridad en aquellas organizaciones que implementan servicios web.

Por otro lado, existen proyectos creados para prevenir y combatir las posibles amenazas a sitios web, uno de ellos es el Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP por sus siglas en inglés) su finalidad es manejar aplicaciones de manera eficiente y segura, y, garantizar los tres pilares fundamentales de la seguridad: Confidencialidad, Integridad y disponibilidad.

En correspondencia a lo anterior se desarrolla el presente proyecto, que tiene por objetivo implementar controles de seguridad al sitio web Machala Restaurantes para ello se basó en los tres primeros riesgos del proyecto OWASP top 10 2017(Inyección, Pérdida de autenticación y Exposición de datos sensibles), se hizo uso de herramientas sobre un escenario de auditoría que permitieron explorar el impacto de dichos riesgos en el sitio web mencionado con anterioridad. Además, se implementaron soluciones o controles que permitieron mitigar el impacto de las vulnerabilidades.

La estructura del presente proyecto se organiza de la siguiente manera:
Capítulo I: Describe detalladamente los factores que llevaron a la creación de la investigación: Introducción, marco contextual, problema y objetivo.
Capitulo II: Corresponde a la fundamentación teórica, al diseño del escenario de trabajo realizado para la solución y a los resultados obtenidos
Capítulo III: Evidencia los resultados obtenidos a través de las conclusiones.

1.1 Marco Contextual

La implementación de sitios web se ha convertido en un espacio para que la mayoría de empresas ofrezcan sus servicios de todo tipo, en Ecuador todas las empresas conocidas y consolidadas cuentan ya con este servicio, y la mayoría de las pequeñas empresas que van en crecimiento también. Una de las principales preocupaciones en este sector es la seguridad, y es que es normal que una empresa se preocupe por mantener sus datos seguros, no solo por su beneficio sino también por el de los clientes, ya que si estos se ven afectados probablemente no querrán involucrarse nuevamente con dicha organización. La seguridad puede verse afectada por faltas de control en las partes vulnerables, un atacante puede valerse de esto para robar información y cometer actos fraudulentos, varios han sido los casos de ciberataques a sitios web, los cuales han dejado grandes pérdidas y por ende deterioro en la imagen profesional.

1.2 Problema

Las aplicaciones web con faltas de control en seguridad son propensas a caer en ataques maliciosos, garantizar un sitio web altamente confiable puede resultar complejo pero necesario, pues la información al ser considerada uno de los principales activos de las empresas debe ser tratada con sigilosidad. Los cibercriminales disponen de técnicas, herramientas y estrategias que buscan la mínima debilidad en un sitio web para robar su información y causar perjuicios. El problema viene dado desde el momento en el que inicia el desarrollo del sitio web; las malas prácticas de programación, configuraciones por defecto y datos descifrados son algunas de las vulnerabilidades más comunes. Por esta razón, realizar una auditoría de seguridad a una aplicación web es de vital importancia para una organización, basados en esto, surge el presente proyecto; el cual implementa controles a una aplicación web mediante la metodología OWASP.

1.3 Objetivo General

Implementar controles de seguridad a una aplicación web mediante la metodología OWASP top 10 2017 para mitigar los riesgos que afectan la seguridad del sitio.

2. DESARROLLO

2.1. Marco Teórico

2.1.1 Ataques Informáticos

Son aquellas actividades sistemáticas enfocadas a alterar la seguridad de una entidad, básicamente son amenazas ocasionadas por un determinado sujeto que tiene por finalidad valerse de las vulnerabilidades para obtener sus propios beneficios [4].

2.1.2 OWASP top 10

Proyecto creado en el año 2001 que incluye 10 de las vulnerabilidades mayormente empleadas por los atacantes a los sitios web [5]. Su finalidad es identificar y prevenir los riesgos logrando hacer que las organizaciones elaboren y garanticen aplicaciones de confianza [6]. En sí, representa un marco de trabajo orientado a la seguridad en aplicaciones web.

2.1.3 Inyección SQL

Es el ataque más empleado por los ciberdelincuentes a páginas web que contienen bases de datos [7], surge por la falta de validaciones en las entradas de la aplicación, dichas malas prácticas de programación permiten al atacante utilizar sentencias de lenguaje de consulta estructurado lenguaje (SQL) para ejecutar comandos mal intencionados o acceder a información no autorizada [8], [9].

2.1.4 Pérdida de Autenticación

Permite a los atacantes obtener usuarios y contraseñas de una persona a través de la explotación de las vulnerabilidades de la aplicación, este proceso se da con herramientas automatizadas, generalmente a través de listas. Como resultado se obtiene el robo de identidad y con ello la suplantación de usuarios o permisos [10].

2.1.5 Exposición de Datos Sensibles

Un ataque de este tipo afecta a información altamente comprometida con el usuario (números de cuentas bancarias, documentos de identidad, credenciales, etc.) [11] se da cuando la información sensible no es adecuadamente protegida por los sitios web, por lo que debe poseer diferentes mecanismos de control que garanticen al máximo la integridad de los datos.

2.1.6 SQLmap

Es una herramienta de código abierto, creada para la detección y explotación de vulnerabilidades en sitios web, esta actividad la realiza a través del ataque denominado inyección sql, el cual el atacante extrae información de la base de datos y la manipula a su conveniencia [12].

2.1.7 BurtSuite

Es una herramienta de prueba de seguridad de aplicaciones, su funcionamiento está basado en interceptar y mostrar mensajes estructurados provenientes del protocolo HTTP mediante la configuración del servidor proxy, los cuales pueden ser controlados a conveniencia del usuario [13].

El procedimiento para fuerza bruta es el siguiente [14]:

- En Intruder se intercepta una solicitud enviada en la aplicación
- Se selecciona el parámetro a ser forzado en forma bruta
- Se configuran dichos parámetros e inicia el ataque.

2.1.8 Wireshark

Es una de las herramientas gratuitas más completas y populares en el seguimiento de paquetes de red, entre sus funcionalidades está el análisis de protocolos, redes, paquetes, análisis forense y filtros de visualización [15], [16].

2.1.9 Framework Django

Framework es un marco de trabajo que facilita el desarrollo de aplicaciones, está compuesto por bibliotecas, técnicas y herramientas que agilizan procesos y garantizan aplicaciones web concretas y robustas [17]. Django es un framework de código limpio que permite ejercer buenas prácticas de programación, una de sus principales características es el mapeo objeto-relacional (ORM) el cual escribe código Python por SQL, permitiendo realizar de manera más eficiente las consultas de la vista [18].

2.1.10 Https – Protocolo de transferencia de hipertexto

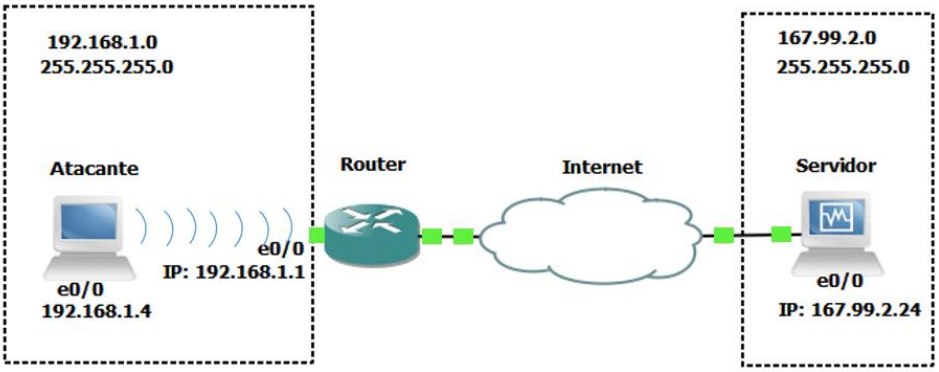
Garantiza un canal de comunicación seguro entre servidores y clientes, está diseñado para evitar las escuchas ilegales, la manipulación o la falsificación de mensajes [19], [20], en pocas palabras está orientado a proteger la integridad de los datos. TSL es quien asegura lo antes mencionado, que involucre cifrado, integridad y autenticación.

2.2 Solución del problema

2.2.1 Diseño del Escenario

Para la elaboración del escenario se utilizó la herramienta de simulación de red GSN3. El escenario de red 1 está compuesto por una red LAN y WAN, en la primera se encuentra el atacante con el sistema operativo KaliLinux ubicado en la maquina VirtualBox, para la conexión a internet (WAN) se hace uso de un router, el cual accede al servidor por medio del dominio www.machalarestaurantes.com, en el otro extremo se encuentra el servidor el cual contiene el sitio web antes mencionado. Este escenario fue utilizado para analizar los riesgos de inyección y pérdida de autenticación ya que no requieren de la presencia del cliente para llevar a cabo sus procesos. Las configuraciones de cada dispositivo se muestran en la figura 1:

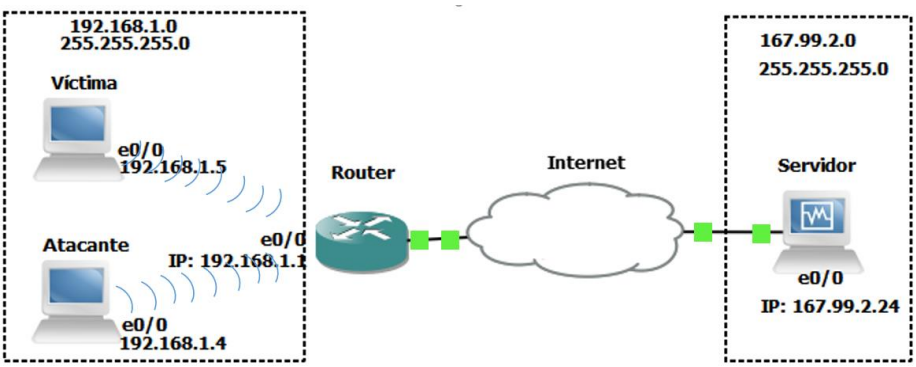
Figura 1: Escenario de trabajo - Topología



Fuente: Elaboración propia

El escenario 2 utiliza una víctima para la captura de datos en el riesgo de exposición de datos sensibles, esto se representa en la figura 2:

Figura 2: Escenario de trabajo con una víctima de por medio.



Fuente: Elaboración propia

2.2.2 Explotación de las vulnerabilidades basadas en Inyección SQL

Este ataque se realizó con la herramienta sqlmap, la cual se encarga de detectar sitios web vulnerables y con ello aprovechar la inyección. De las diferentes operaciones que se pueden ejercer se ha elegido la de listar los usuarios de la base de datos correspondiente a Machala Restaurantes. Los parámetros a utilizar son los siguientes:

-u: Especifica la dirección de la página web o URL a auditar (Ver Anexo A).

-dbs: Solicita la lista de todas las bases de datos correspondientes a la URL ubicada anticipadamente.

Figura 3: Acceso a la base de datos

```
root@kali:~/.sqlmap# cd ..
root@kali:~# pwd
/root
root@kali:~# sqlmap -u http://192.168.1.4:8000/blogBusqueda?id=hola%27 --dbs
```

Fuente: Elaboración propia

El resultado alojó como resultado las bases de datos existentes en el sitio web (Ver Anexo B). Una vez obtenida la base de datos se procedió a listar las tablas:

-D: especifica la base de datos con la que se quiere trabajar.

--tables: Señala que se desea obtener las tablas de la base de datos indicada con anterioridad.

Figura 4:Acceso a las tablas del sitio web

```
root@kali:~# sqlmap -u http://192.168.1.4:8000/blogBusqueda?id=hola%27 -D public --tables
```

Fuente: Elaboración propia

De dicho comando se listan las 28 tablas contenidas en la base de datos del sitio web (Ver Anexo C). Ahora, finalmente se listarán los usuarios contenidos en la tabla auth_user, para ello se utiliza el siguiente comando:

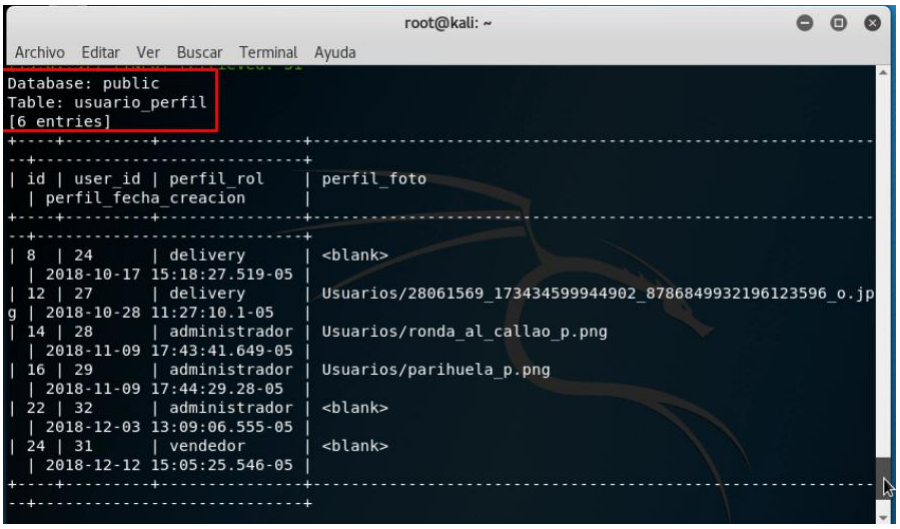
Figura 5: Listar los campos de una tabla

```
root@kali:~# sqlmap -u http://192.168.1.4:8000/blogBusqueda?id=hola%27 -D public -T usuario
perfil --dump
```

Fuente: Elaboración propia

En la figura 6 se muestra el resultado de todo el proceso de inyección, el cual permitió listar los usuarios de la base de datos con su respectiva información, comprobando de esta manera lo vulnerable que es la página.

Figura 6: Lista de los usuarios obtenidos



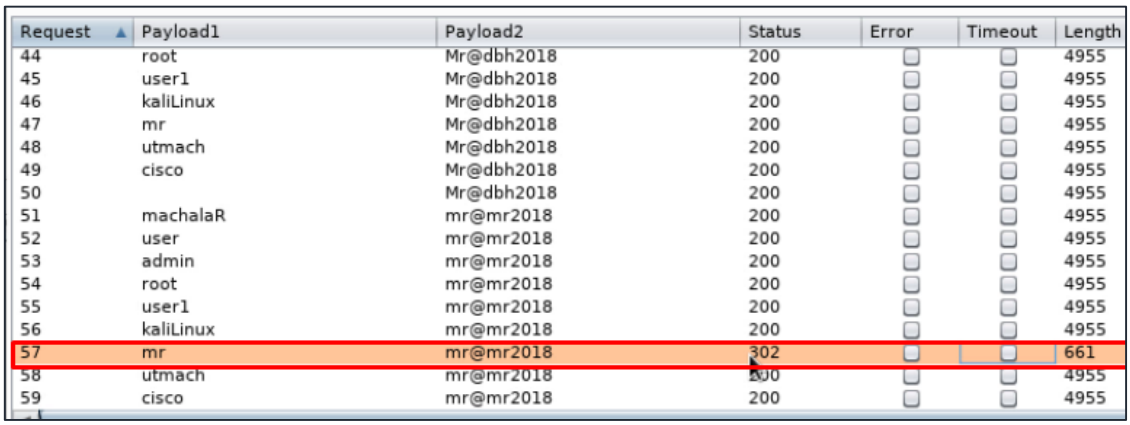
Fuente: Elaboración propia

2.2.3 Explotación de las vulnerabilidades de pérdida de autenticación:

Para la pérdida de autenticación se utilizó el ataque de fuerza bruta, el cual utiliza miles o millones de combinaciones para encontrar tanto usuarios como contraseñas. La herramienta utilizada para dichas combinaciones fue crunch (Ver Anexo D), mientras que la utilizada para ejecutar el ataque fue Burp Suite.

En primera instancia se configuró el proxy en el navegador Mozilla FireFox (Ver Anexo E), luego Burp Suite (Ver Anexo F). En la figura **7** se muestra el usuario y contraseña encontrada con esta herramienta.

Figura 7: Usuario y contraseña encontrada con Burp Suite



Request	Payload1	Payload2	Status	Error	Timeout	Length
44	root	Mr@dbh2018	200			4955
45	user1	Mr@dbh2018	200			4955
46	kaliLinux	Mr@dbh2018	200			4955
47	mr	Mr@dbh2018	200			4955
48	utmach	Mr@dbh2018	200			4955
49	cisco	Mr@dbh2018	200			4955
50		Mr@dbh2018	200			4955
51	machalaR	mr@mr2018	200			4955
52	user	mr@mr2018	200			4955
53	admin	mr@mr2018	200			4955
54	root	mr@mr2018	200			4955
55	user1	mr@mr2018	200			4955
56	kaliLinux	mr@mr2018	200			4955
57	mr	mr@mr2018	302			661
58	utmach	mr@mr2018	200			4955
59	cisco	mr@mr2018	200			4955

Fuente: Elaboración propia

2.2.4 Explotación de las vulnerabilidades de exposición a datos sensibles

Para la ejecución de este ataque se utilizó la herramienta WireShark, la cual se basa en captura de paquetes de diversos protocolos, en este caso HTTP. La configuración de

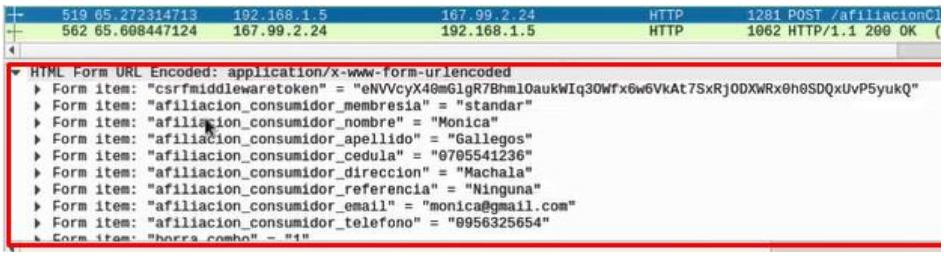
esta herramienta se muestra en el Anexo G. En esta sección se requiere la presencia de la víctima, que será quien ingrese la información sensible, esto se muestra en la figura **8:**

Figura 8: Formulario de datos sensibles llenados por el cliente

Fuente: Elaboración propia

Una vez que el cliente digite su información, el atacante podrá acceder a ella por medio de la herramienta antes mencionada, tal y como se muestra en la figura 9:

Figura 9: Captura de datos con Wireshark



Fuente: Elaboración propia

2.2.5 controles para Inyección

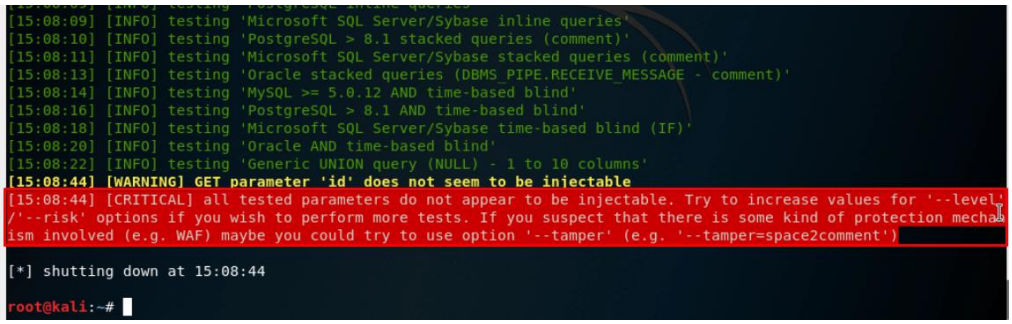
Para prevenir ataques de inyección SQL se hizo la ejecución de las consultas por medio de los parámetros otorgados por el framework Django, ya que este otorga mecanismos para evitar dichos ataques (Ver Anexo H). En la figura 10 se muestra un nuevo intento de inyección:

Figura 10: Control de Inyección SQL

Fuente: Elaboración propia

En la figura 11 se muestra le denegación de esta petición, esto debido al control antes implementado.

Figura 11: Denegación al ataque de Inyección SQL

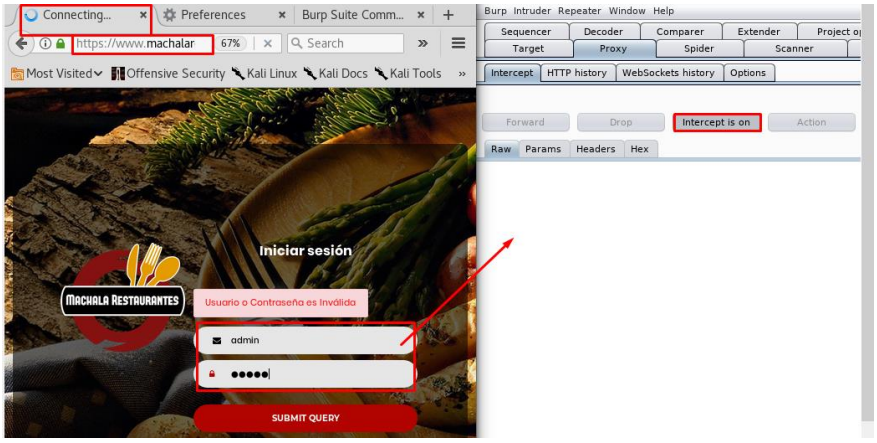


Fuente: Elaboración propia

2.2.6 Controles para pérdida de autenticación y datos sensibles

Para prevenir el ataque de Pérdida de autenticación y exposición a datos sensibles se implementó un certificado de seguridad denominado SSL (Secure Sockets Layer o capa de sockets seguros), con ello la transferencia de información fue enviada directamente al servidor (ver Anexo I**)**. En la figura 12 se muestra el intento de ataque a través de fuerza bruta, en dicha imagen se puede apreciar como las peticiones realizadas en el sitio web no pueden ser capturadas ya que el servidor no entrega ningún tipo de información.

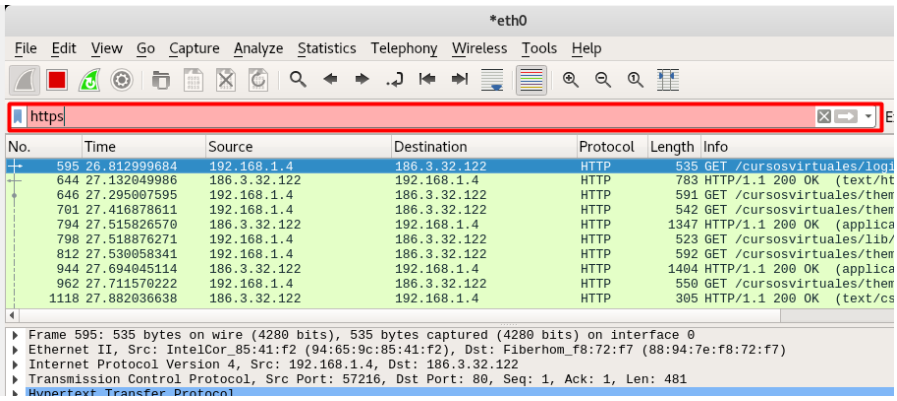
Figura 12: Control del ataque de fuerza bruta en pérdida de autenticación



Fuente: Elaboración propia

En lo que respecta al control de exposición de datos sensibles sucede algo similar, con SSL los datos se trasmiten de forma directa y cifrada entre cliente y servidor, en la figura 13 se muestra el intento de captura de los paquetes contenedores de datos sensibles en donde no se encuentran resultados, lo que quiere decir que solo se muestran peticiones http o sitios web inseguros.

Figura 13: Control del ataque de exposición de datos sensibles



Fuente: Elaboración propia

2.3 Resultados

Después de la ejecución de los tres tipos de ataques; Inyección, Pérdida de autenticación y exposición a datos sensibles se evidenció la vulnerabilidad del sitio web Machala Restaurantes. En lo que respecta a inyección sql, se encontraron malas prácticas de programación referentes a consultas directas a la base de datos, esto se originó por la falta de aplicación del mapeo objeto-relacional (ORM) del framework Django; el cual incluye un método llamado RAW que permite enviar consultas SQL directas, pero encapsuladas. El control se basó en la utilización de dicho método, que hizo que la consulta sea más rigurosa; por tanto, evitó que las sentencias sean vulnerables a ataques de inyección SQL. En el ataque de exposición de datos sensibles se encontró que las peticiones del sitio web corrían por medio del puerto 80 (HTTP), que significa inseguridad en el tráfico de la red, en otras palabras; vulnerable a que cualquier herramienta espía (Wireshark) capture el tráfico y con ello todos los paquetes transferidos por la red, obtenida dicha vulnerabilidad se estableció el respectivo control, que consistió en cifrar los datos a través de la certificación SSL, protocolo que corre a través del puerto seguro 443 (HTTPS), de esta manera se aseguró el tráfico de datos en la red. En el ataque de fuerza bruta se pudo evidenciar la vulnerabilidad de la página web a través de la captura del usuario y contraseña, dicho proceso consistió en configurar tanto en el navegador como en la herramienta Burp Suite, el proxy; con ello se interceptaron todas las peticiones tipo POST realizadas en el login del sitio web, esto permitió identificar los campos a atacar, en este caso “username” y “password”, a cada uno se le asignó una lista de posibles usuarios y contraseñas y, luego de la ejecución del ataque, se identificaron las credenciales del administrador. El control se basó en evitar la intercepción de información por medio del proxy, para lo cual se implementó el certificado SSL el cual logró asegurar el tráfico de información en la red y con ello la prevención hacia este tipo de ataques.

3. CONCLUSIONES

La utilización de las herramientas sqlmap, Burp Suite y Wireshark en los diferentes ataques permitieron comprobar la existencia de vulnerabilidades en la aplicación web; la cual después de una búsqueda exhaustiva de soluciones, fue controlada.

La utilización del framework Django permitió parametrizar las consultas SQL a través del método raw, e hizo la aplicación web más robusta; lo cual permitió controlar el ataque de inyección SQL y asegurar la confidencialidad e integridad de la base de datos.

La implementación del certificado SSL logró controlar dos de los ataques del proyecto OWASP: pérdida de autenticación y exposición a datos sensibles; su función principal está basada en garantizar la seguridad de la información desde la transferencia hasta su recepción.

Los controles implementados permitieron mitigar las vulnerabilidades encontradas, y con ello garantizar un sitio fiable para los usuarios, pero este proceso no queda ahí, es importante realizar auditorías periódicas ya que, así como el desarrollo web está en constante evolución, las técnicas de ataque también.

BIBLIOGRAFÍA

[1] J. R. Molina Ríos, M. P. Zea Ordóñez, J. A. Honores Tapia, and A. S. Gómez Moreno, “Analysis Methodologies Web Application Development,” *Int. J. Appl. Eng. Res.*, vol. 11, no. 16, pp. 9070–9078, 2016.

[2] P. P. Marín Dueñas, C. Lasso Vega González, and J. J. Mier-Terán Franco, “La interactividad de las webs en las pequeñas y medianas empresas,” *Opcion*, vol. 31, no. 3, pp. 735–750, 2015.

[3] H. R. González Brito and R. Montesino Perurena, “Capacidades de las metodologías de pruebas de penetración para detectar vulnerabilidades frecuentes en aplicaciones web,” *Rev. Cuba. Ciencias Informáticas*, vol. 12, no. 4, pp. 52–65, 2018.

[4] M. Soriano, *Seguridad en redes y seguridad de la información*, Primera ed. Praga, 2014.

[5] P. Gaona García, C. Montenegro Marín, and J. Barón Velandia, “Modelo ontológico para la predicción de ataques informáticos a partir de Honeynets virtualizadas,” *Rev. Logos, Cienc. Tecnol.*, vol. 8, no. 1, pp. 101–114, 2016.

[6] OWASP, *OWASP Top 10 - 2017 Los diez riesgos más críticos en Aplicaciones Web*, Creative C. 2017.

[7] B. Soewito, F. E. Gunawan, and F. Hirzi, “Prevention Structured Query Language Injection Using Regular Expression and Escape String,” *Procedia Comput. Sci.*, vol. 135, pp. 678–687, 2018.

[8] F. Román Muñoz, I. I. Sabido Cortes, and L. J. García Villalba, “Enlargement of vulnerable web applications for testing,” *J. Supercomput.*, vol. 74, no. 12, pp. 1–20, 2017.

[9] P. R. McWhirter, K. Kifayat, Q. Shi, and B. Askwith, “SQL Injection Attack classification through the feature extraction of SQL query strings using a Gap-Weighted String Subsequence Kernel,” *J. Inf. Secur. Appl.*, vol. 40, pp. 199–216, 2018.

[10] Y. Ayachi, E. H. Ettifouri, J. Berrich, and B. Toumi, “Modeling the OWASP Most Critical WEB Attacks,” *Smart Innov. Syst. Technol.*, vol. 111, pp. 442–450, 2019.

[11] J. Hernández Saucedo, Ana Laura; Mejía Miranda, “Guía de ataques, vulnerabilidades, técnicas y web,” *Rev. electrónica Comput. Informática*

Biomédica y Electrónica, vol. 4, no. 1, 2015.

[12] R. E. L. De Jimenez, “Pentesting on web applications using ethical - Hacking,” *2016 IEEE 36th Cent. Am. Panama Conv. CONCAPAN 2016*, no. 503, 2017.

[13] C. Mainka, V. Mladenov, T. Guenther, and J. Schwenk, “Automatic Recognition , Processing and Attacking of Single Sign-On Protocols with Burp Suite,” *Lect. Notes Informatics (LNI), Proc. - Ser. Gesellschaft fur Inform.*, vol. 251, pp. 117–131, 2015.

[14] C. Joshi and U. K. Singh, “Performance Evaluation of Web Application Security Scanners for More Effective Defense,” *Int. J. Sci. Res. Publ.*, vol. 6, no. 6, pp. 660–667, 2016.

[15] G. Bagyalakshmi *et al.*, “Network Vulnerability Analysis on Brain Signal/Image Databases Using Nmap and Wireshark Tools,” *IEEE Access*, vol. 6, pp. 57144–57151, 2018.

[16] L. R. Roba Iviricu, J. R. Vento Alvarez, and L. E. García, “Metodología para la Detección de Vulnerabilidades en las Redes de Datos utilizando Kali-Linux,” *Avances*, vol. 23, no. 3, pp. 334–344, 2016.

[17] J. R. Molina Ríos, N. M. Loja Mora, M. P. Zea Ordóñez, and E. L. Loaiza Sojos, “Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python,” *Rev. Latinoam. Ing. Softw.*, vol. 4, no. 4, pp. 201–207, 2016.

[18] L. J. Ayala Condori, “Python – Django Framework de desarrollo web para perfeccionistas Basado en el Modelo MTV,” *Rev. Inf. Tecnol. y Soc.*, no. 7, pp. 36–37, 2012.

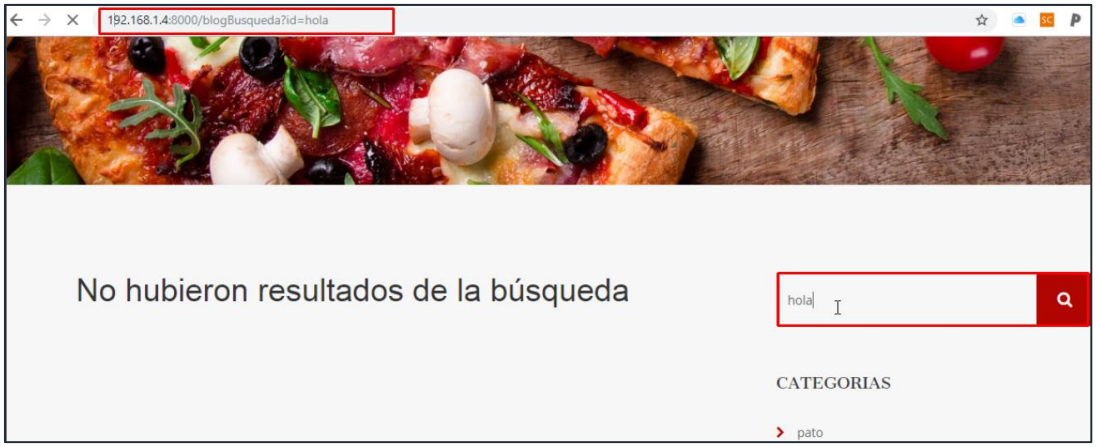
[19] J. Liang, J. Jiang, H. Duan, K. Li, T. Wan, and J. Wu, “When HTTPS meets CDN: A case of authentication in delegated service,” *IEEE Symp. Secur. Priv.*, pp. 67–82, 2014.

[20] S. Stricot-Tarboton, S. Chaisiri, and R. K. Ko, “Taxonomy of man-in-the-middle attacks on HTTPS,” *IEEE Comput. Soc.*, pp. 527–534, 2016.

ANEXOS

Anexo A: Obtención de la URL a realizar el ataque

En el Anexo A se muestra la generacion de la url a traves de un campo de búsqueda, esta url será utilizada para llevar a cabo el ataque de inyeccion SQL.

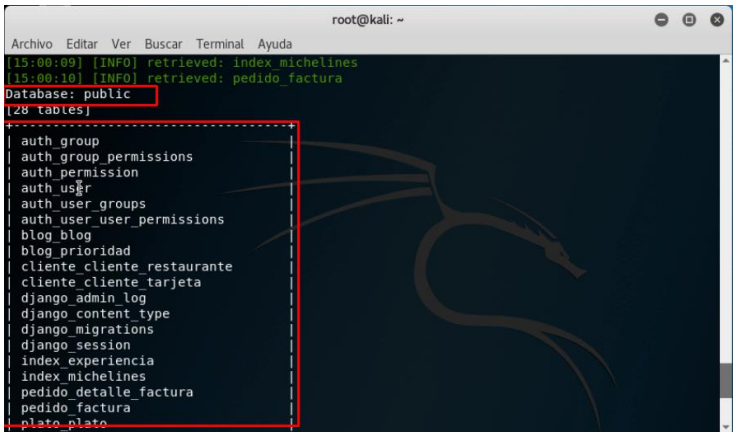


Fuente: Elaboración propia del autor

Anexo B: Lista de la base de datos obtenida con inyección SQL.

Fuente: Elaboración propia

Anexo C: Lista de las tablas obtenidas con Inyección SQL



Fuente: Elaboración propia

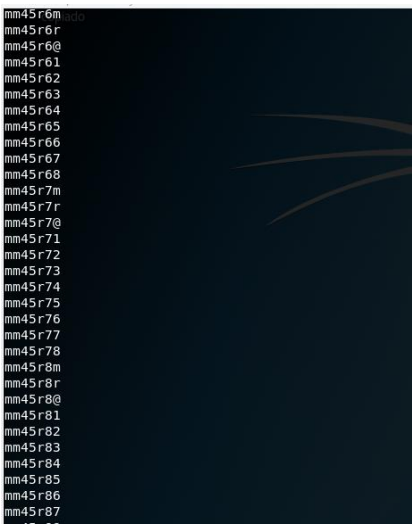
Anexo D: Lista de combinaciones generadas por crunch

La herramienta crunch permite generar una cantidad masiva de combinaciones, posee varios parametros, entre los principales, el minimo y maximo de longitud y los caracteres a combinar:

```
root@kali:~# crunch 7 7 mr@12345678
```

Fuente: Elaboración propia

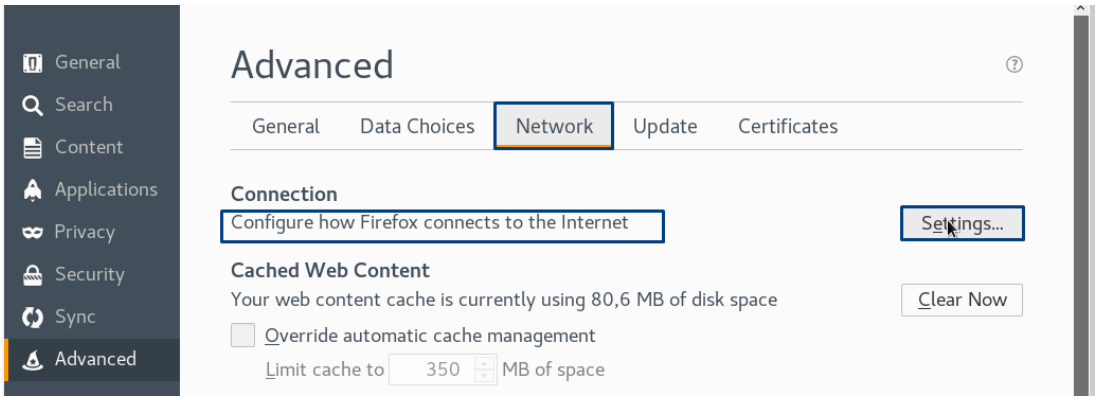
Lista de combinaciones obtenidas del comando anterior:



Fuente: Elaboración propia

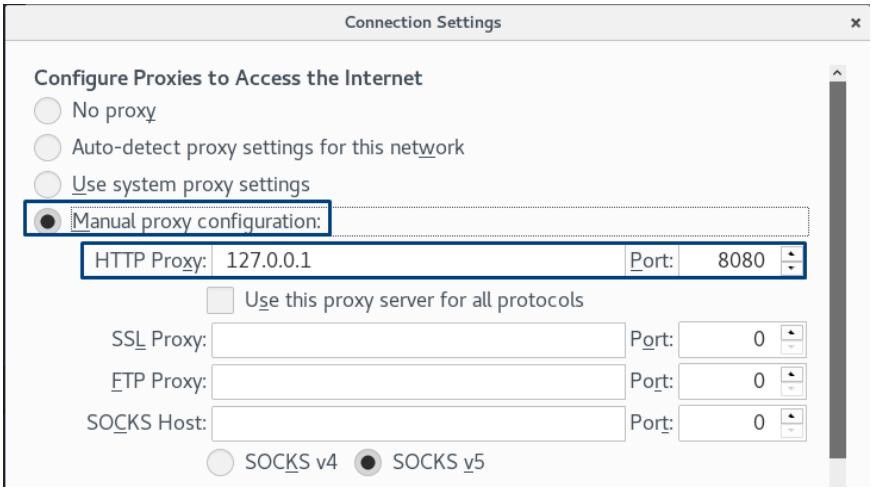
Anexo E: Configuración del proxy

Se configura Proxy para indicarle al navegador que las peticiones realizadas en el sitio web deben ser enviadas a Burp Suite.



Fuente: Elaboración propia

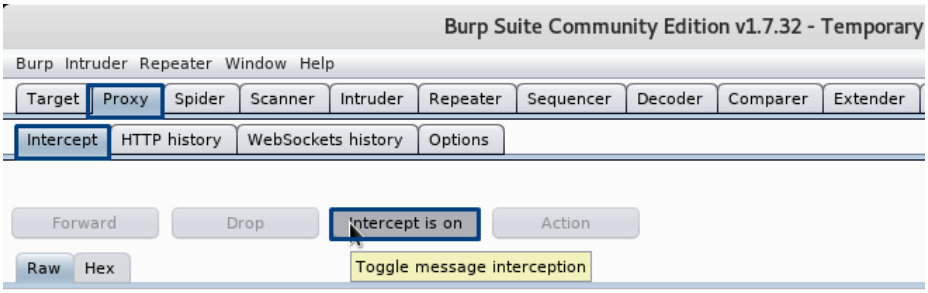
Para ello se trabaja con la dirección de Loopback y con el puerto 8080



Fuente: Elaboración propia

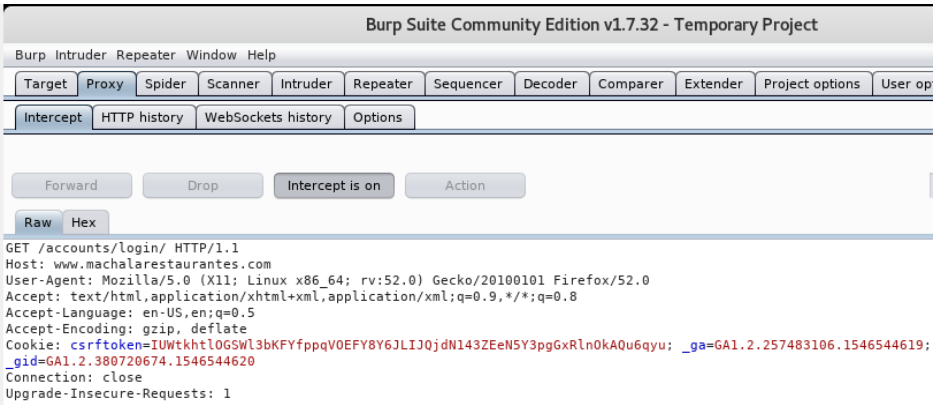
Anexo F: Configuración de BurtSuite

Es necesario interceptar las peticiones que se estan realizando en el navegador, es por ello que se activa la opcion “Intercepre is on”



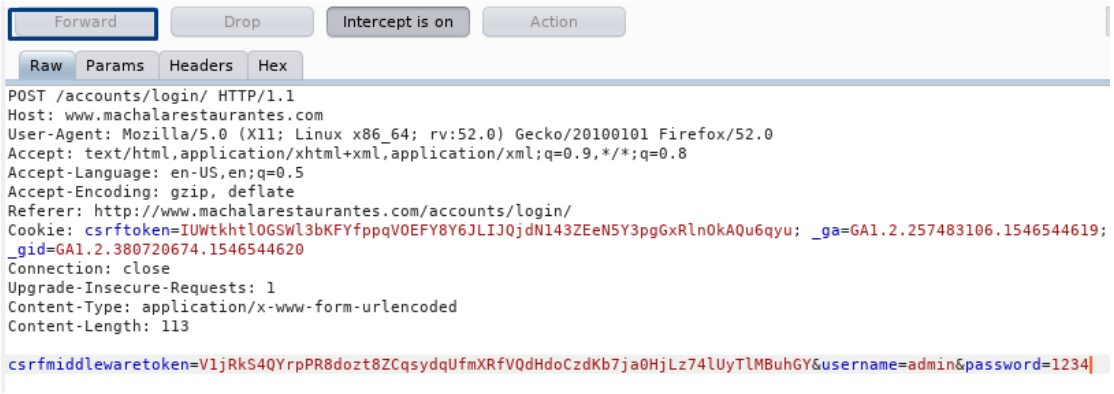
Fuente: Elaboración propia

Cuando se hagan peticiones en el navegador, estas se mostrarán en la consola



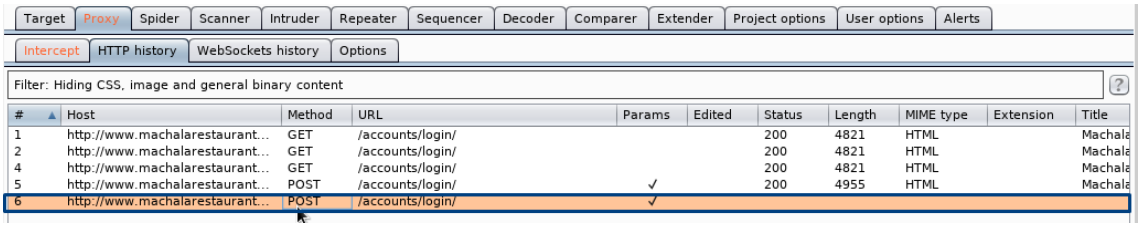
Fuente: Elaboración propia

Se ingresa al formulario de autenticación y se inserta el usuario y contraseña, es indiferente la validación de estos campos, ya que solo se requieren los parámetros, los cuales se presentarán en la consola de Burp Suite:



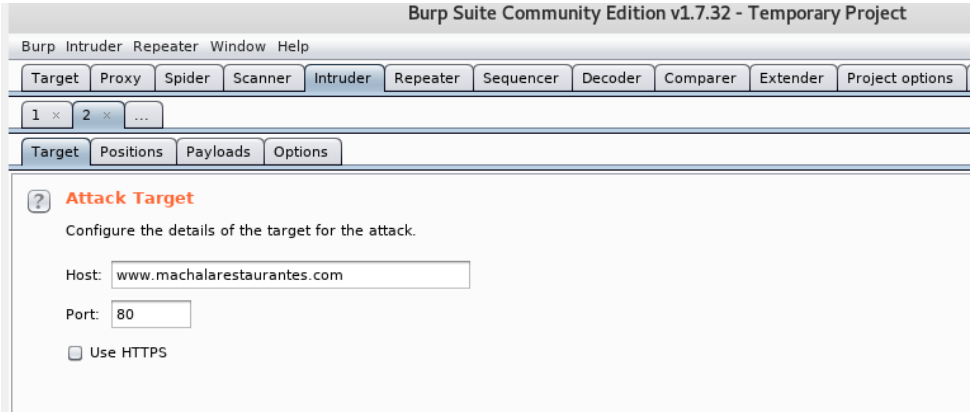
Fuente: Elaboración propia

Como se hizo una inserción, se muestra el método POST, que es con quien se trabajará el ataque



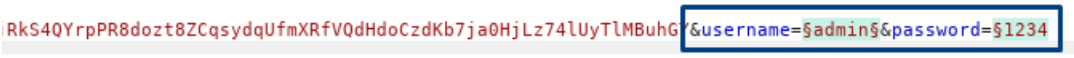
Fuente: Elaboración propia

Sitio web con su respectivo puerto a quien se le aplicará el ataque



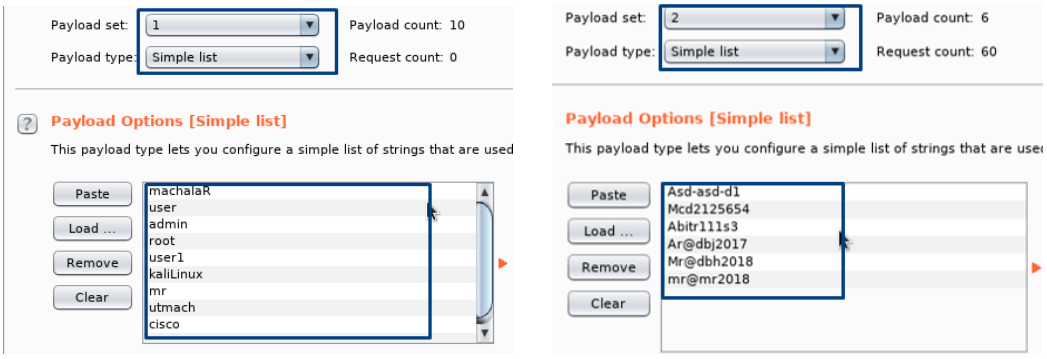
Fuente: Elaboración propia

Es necesario especificar los parámetros, a cada uno de ellos se le asignará una lista de posibles combinaciones de usuario y contraseña generados con anterioridad



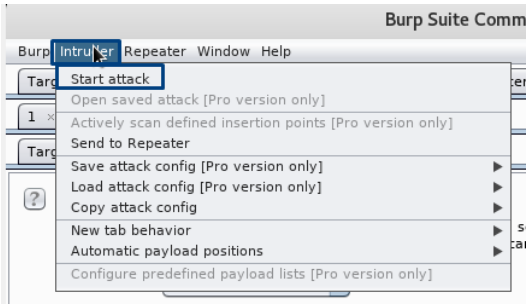
Fuente: Elaboración propia

Asignación de las listas a los parámetros de username y password respectivamente



Fuente: Elaboración propia

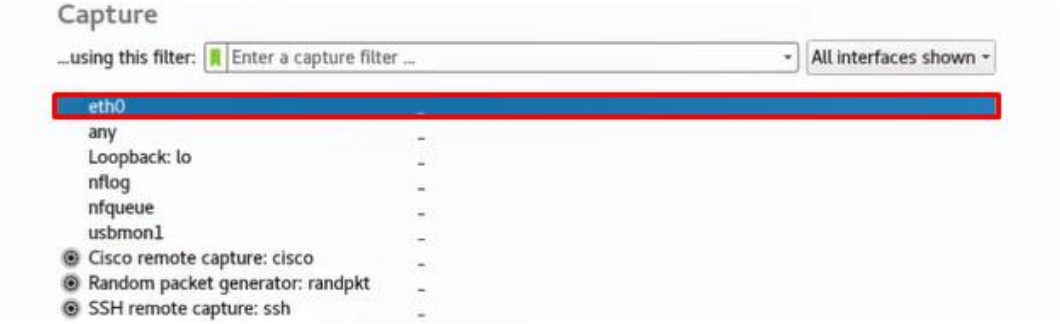
Finalmente se inicia el ataque de fuerza bruta



Fuente: Elaboración propia

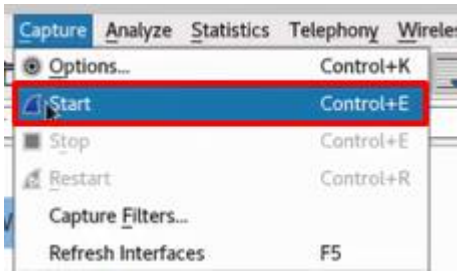
Anexo G: Configuración de le herramienta Wireshark.

Se selecciona la interfaz con la que se va a trabajar, en este caso eth0



Fuente: Elaboración propia

Al seleccionar Start, comenzará a seleccionar los paquetes transferidos en la red.



Fuente: Elaboración propia

Anexo H: Control para el ataque Inyección SQL

Sentencias SQL sin uso de los métodos apropiados (raw) del Framework Django, en donde las consultas se realizan directamente a la base de datos.

```
    cur = conn.cursor()
    s=""
    if not request.GET.get('id')== None:
        s=request.GET.get('id')
        cur.execute("SELECT * FROM restaurante_restaurante where restaurante_id='% s')" % s)
        rows=cur.fetchall()
    return render(request,'publico/inyeccion.html', {'con':s})
```

Fuente: Elaboración propia

Utilización del framework Django, en donde la sentencia se ejecuta a través de los comandos que ofrece el framework.

```
    s=""
    if not request.GET.get('id')== None:
        s=request.GET.get('id')
        h=s+'%'
        rest = Restaurantes.objects.raw("SELECT * FROM restaurante_restaurante where restaurante_id='% s')" % [h])
    return render(request,'publico/inyeccion.html', {'con':rest})
```

Fuente: Elaboración propia

Anexo I: Configuración de SSL

Archivos ssl a configurar, dichos archivos son entregados en la adquisición del certificado.

```
[root@machalarestaurantes ssl]# ls
intermediate.crt  machalarestaurantes.com.crt  machalarestaurantes.com.key
```

Fuente: Elaboración propia

Se debe buscar dentro del servidor web la carpeta donde se creó el virtual host del proyecto, esto con la finalidad de adjuntar restricciones, además se agrega el puerto seguro 443:

```
<VirtualHost *:443>
        SSLEngine on
        SSLCertificateFile /etc/httpd/ssl/machalarestaurantes.com.crt
        SSLCertificateKeyFile /etc/httpd/ssl/machalarestaurantes.com.key
        SSLCertificateChainFile /etc/httpd/ssl/intermediate.crt
```

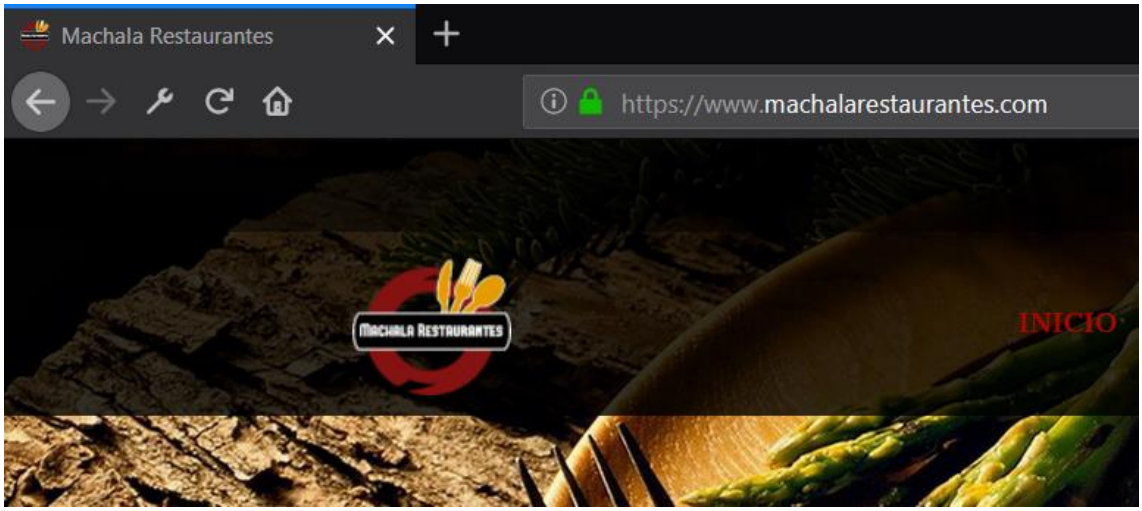
Fuente: Elaboración propia

Por defecto, al realizar una petición los navegadores buscan el puerto 80, por lo tanto, se le debe agregar un virtualhost que contenga de forma permanente una regla que invoque al puerto seguro 443 donde se aloja la página.

```
<VirtualHost *:80>
   Redirect permanent / https://www.machalarestaurantes.com/
</VirtualHost>
```

Fuente: Elaboración propia

Para finalizar, se reinicia el servicio. En el navegador se muestra como la aplicación web está corriendo por un puerto seguro.



Fuente: Elaboración propia