PENTESTING AL PROYECTO WEB "QUADODO LOGIN SCRIPT" DESARROLLADO Y SOPORTADO EN LENGUAJE PHP VERSIÓN 5.5.0

HENRY ALFONSO GARZÓN PINZÓN

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA "UNAD"
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
BOGOTÁ D.C
2017

PENTESTING AL PROYECTO WEB "QUADODO LOGIN SCRIPT" DESARROLLADO Y SOPORTADO EN LENGUAJE PHP VERSIÓN 5.5.0

HENRY ALFONSO GARZÓN PINZÓN

Tesis de grado para optar por el título:

Especialista en Seguridad Informática

Director de Monografía:

Ing. Hernando José Peña Hidalgo

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA "UNAD"
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
ESPECIALIZACIÓN EN SEGURIDAD INFORMÁTICA
BOGOTÁ D.C
2017

Nota de Aceptación:

_____
_____
_____
_____
_____
_____
_____

_____

Firma del Presidente del Jurado

_____

Firma del Jurado

_____

Firma del Jurado

Bogotá, 14 de abril de 2017

DEDICATORIA

Dedico esta monografía a mis padres y mi hermano los cuales han sido un soporte para continuar adelante con mis estudios y sacar adelante esta especialización con la cual complemento mi ciclo profesional como Ingeniero de Sistemas.

AGRADECIMIENTOS

A los miembros de mi familia dedico todo mi proceso en esta especialización y sobre todo la realización de esta monografía de grado en el cual consigno los conocimientos adquiridos a través de la misma.

Por supuesto a la Universidad Abierta y a Distancia, mediante la cual y a través de su maravilloso sistema a distancia realizar la especialización en seguridad informática fue posible. También a los docentes que estuvieron acompañándome a mi y a mis compañeros para llevarla a feliz término.

CONTENIDO

pág.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

1. TÍTULO

PENTESTING AL PROYECTO WEB "QUADODO LOGIN SCRIPT" DESARROLLADO Y SOPORTADO EN LENGUAJE PHP VERSIÓN 5.5.0

2. INTRODUCCIÓN

La presente monografía se desarrolla como proyecto final de la especilización de seguridad informática y para optar al título de especialista. Se centra en la práctica de ejecución de pruebas de penetración (pentesting) sobre un sistema o módulo seleccionado, en este caso Quadodo Login Script, un módulo de gestión de usuarios y de accesos que puede ser integrado a cualquier aplicación desarrollada en PHP.

Quadodo es un software open-source y de libre distribución, creado con el objetivo de ser usado “por cualquiera que desee permitir a usuarios ingresar a su sitio web”, según lo indicado en la página oficial (Rennehan, PHP Login Script?, 2015). Este script provee la funcionalidad de autenticación de usuarios, además de otras funcionalidades como administración de usuarios, administración de grupos, permisos de acceso sobre páginas y creación de máscaras de permisos. Todo esto, puede ser integrado a un sistema existente desarrollado en PHP que desee usar estas funcionalidades para la administración de seguridad de su sitio.

Las pruebas de pentesting antes mencionadas van encaminadas a identificar las vulnerabilidades y ventajas de seguridad de la información de un sistema, componente, programa, etc, en este caso Quadodo Login Script y se aplican en el desarrollo de esta monografía sobre los módulos *Security, Administration y Group Control Panel*. Para la ejecución de las mismas se toma en cuenta la cuarta versión de la guía de pruebas de OWASP de pentesting para definir las pruebas a ejecutar y se acude a las normas ISO NORMAS ISO 30001:2009 e ISO/IEC 31010:2009 para una vez identificadas las vulnerabilidades, relacionar los riesgos asociados y plantear opciones de mejora para gestionar los mismos.

3. DESCRIPCIÓN DEL PROBLEMA

3.1 ANTECEDENTES DEL PROBLEMA

Quadodo se integra con aplicaciones web construidas con lenguaje PHP para reusar funcionalidades previamente construidas de seguridad como autenticación y administración de usuarios y permisos. En caso que una de estas funcionalidades a nivel de seguridad informática presente vulnerabilidades de cualquier tipo, heredará con certeza las mismas al aplicativo que lo implemente, comprometiendo de esta manera su sistema y la información que se almacena, generando también riesgos de diferente magnitud al exponer a atacantes puntos débiles a aprovechar.

3.2 FORMULACIÓN DEL PROBLEMA

Actualmente no existe una evaluación formal ejecutada ni documentada de seguridad informática sobre Quadodo Login Script que permita identificar las posibles vulnerabilidades en materia de seguridad de la información que pueda presentar dicho script y que por ende deban corregirse, ni tampoco evidenciar las ventajas que pueda presentar el mismo y que deban mantenerse.

3.3 DESCRIPCIÓN Y PREGUNTA DEL PROBLEMA

Por lo anterior, se requiere que se identifiquen tanto las vulnerabilidades presentes en el script como las fortalezas en materia de seguridad informática para garantizar que las aplicaciones que lo integren no sufran o se vean afectadas, es por esto que se plantea la incógnita: *¿Cómo es posible identificar las vulnerabilidades de seguridad informática que puedan estar presentes en Quadodo Login Script y las funcionalidades que este ofrece de manera segura?*

4. JUSTIFICACIÓN

Ya que Quadodo Login Script es un componente que puede ser integrado en aplicaciones como un módulo de gestor de usuarios y permisos, si el mismo presenta vulnerabilidades de seguridad informática, seguramente el sistema que lo integre heredará dichas vulnerabilidades. Por lo cual es de suma importancia evaluar y validar si Quadodo presenta vulnerabilidades que deban ser corregidas para garantizar la seguridad de la información en los sistemas que lo integren.

Según el sitio web de Quadodo, la versión 2.0.0 de este script tuvo alrededor de 4000 descargas al poco tiempo de ser publicada en el año 2006. Sin embargo, el autor reconoce que dicha versión no estaba bien codificada y no permitía un buen manejo de plantillas. Por esta razón posteriormente se liberaron las versiones 3.0.0 y 3.1.0 incluyendo una mejor codificación, una mejor implementación del panel de administración y el uso de componentes como AJAX (Javascript asíncrono y XML).

Por lo mencionado anteriormente, Quadodo es un script que “se volvió bastante popular” según lo indica su autor por la cantidad de descargas que ha tenido el mismo, sin embargo a pesar de las mejoras a nivel de codificación y del uso de sus funcionalidades, no se tiene conocimiento de que tan seguro sea el mismo para poder ser integrado con toda confianza en una aplicación. Es por esto que se hace necesario identificar las posibles vulnerabilidades de seguridad de la información para saber si es un componente o módulo seguro. (Rennehan, My PHP Login Script, 26)

5. OBJETIVOS DEL PROYECTO

5.1 OBJETIVO GENERAL

Identificar vulnerabilidades de seguridad presentes en los módulos *Security, Administration y Group Control Panel* del proyecto web "Quadodo Login Script", mediante pruebas de pentesting utilizando la metodología OWASP.

5.2 OBJETIVOS ESPECÍFICOS

- Definir las pruebas de pentesting a ejecutar sobre Quadodo Login Script alineadas a la guía OWASP de pruebas de pentesting versión 4 (OWASP Testing Guide v. 4).

- Ejecutar pruebas de pentesting sobre los módulos *Security, Administration y Group Control Panel*.

- Relacionar las vulnerabilidades que se encuentren asignando un identificador basado en la estructura del Common Vulnerabilities and Exposures (CVE).

- Identificar y evaluar los riesgos de seguridad informática asociados a las vulnerabilidades que se encuentren en Quadodo Login Script.

- Generar recomendaciones de acciones de seguridad a ejecutar para mitigar las vulnerabilidades y riesgos que se encuentren en Quadodo Login Script.

6. MARCO REFERENCIAL

6.1 MARCO CONTEXTUAL

6.1.1 El concepto o propósito de Quadodo Script

Quadodo es un script desarrollado por Douglas Rennehan en lenguaje PHP que permite ser ajustado e implementado en cualquier aplicación web desarrollada en el mismo lenguaje, para la gestión y autenticación de usuarios. Soporta bases de datos MySQL y PostgreSQL y entre una de sus principales ventajas es la "personalización para diferentes idiomas, grupos de permisos, diferentes formas de activar el registro, etc.", según lo menciona el artículo de Vinicius (Vinicius, 2013)

6.1.2 Versionamiento y estado actual de la aplicación

Según se indica en la web oficial del script, en materia de seguridad se tienen las siguientes ventajas e implementaciones (Rennehan, Features, 2016):

- Passwords hashed
- Límite en Número de intentos de sesión
- Toda la información de entrada es protegida contra SQL injection
- La imagen de seguridad evita los robots para autenticación
- Acceso limitado para gran cantidad de páginas
- Profundos límites de acceso para usuarios administradores
- Prohibir usuarios a los cuales no se concede acceso al sitio.

La versión actual y oficial del script es la 3.1.11, y fue desarrollada en el año 2013, año desde el cual no se registra actualización a la fecha o un 'release' oficial más reciente (Rennehan, Downloading the Script, 2015). Según el log de cambios publicado por el autor en el sitio web del script, se implementan gran cantidad de ajustes e implementaciones en la versión 3.1.9 (Rennehan, ChangeLog for the Quadodo Login Script 3.1.x, 2013).

6.2 MARCO TEÓRICO

El presente documento expone el proceso que es llevado a cabo para identificar las posibles vulnerabilidades a nivel de seguridad informática que puedan estar presentes en Quadodo Login Script como software desarrollado en lenguaje de programación PHP que provee funcionalidades de administración de usuarios, administración de grupos y administración de permisos para ser integradas en un sitio o sistema web que también esté desarrollado en dicho lenguaje de programación.

El proceso indicado corresponde a la ejecución de pruebas de penetración, también conocidas como pentesting, las cuales según indica la comunidad OWASP (Open Web Application Security Project) consisten “esencialmente en probar una aplicación de forma remota para encontrar vulnerabilidades sin conocer los procesos internos de la aplicación”. El objetivo de las mismas es encontrar fallas en la seguridad de la información y se ejecutan en un contexto de hacking ético (ethical hacking) con el fin de identificar fallas para evitar que atacantes busquen acceder o ejecutar acciones sin autorización en el sistema.

Las fases del pentesting son las siguientes (Rouse, 2014):

1. Recolección de información del sistema u objetivo previo a la ejecución de las pruebas a ejecutar.
2. Identificación de los puntos a evaluar del sistema objetivo (puntos de entrada)
3. Ejecución de pruebas, realizando intentos de entrada y explotación de posibles vulnerabilidades.
4. Consolidación y reporte de resultados.

Por lo anterior, para iniciar se realiza una recolección de información (gathering) de Quadodo Login Script usando como principal fuente la página oficial de dicho script (www.quadodo.net) y realizando todas las configuraciones y pasos de instalación correspondientes.

Continuando con la siguiente fase los puntos de Quadodo que se evalúan en el presente proyecto de grado corresponden a los módulos *Security, Administration y Group Control Panel*. De los cuales se prueban a su vez funcionalidades especificas a través de 20 pruebas de pentesting como parte de la tercera fase de pentesting listada anteriormente.

Las pruebas indicadas corresponden a pruebas planteadas cuarta versión de la guía de pruebas de OWASP. OWASP es una comunidad colaborativa que tiene como objetivo el fomentar el desarrollo, comprar y mantener aplicaciones o sistemas seguros (OWASP, 2014). Es pues esta guía un documento que busca cumplir con dicho objetivo mediante la aplicación de pruebas de seguridad que permitan determinar el nivel de seguridad de las aplicaciones, dando instrucciones de ejecución, herramientas y criterios de evaluación.

Para dichas pruebas se cuenta actualmente con herramientas que automatizan algunos procesos de evaluación de seguridad de la información, existiendo también distribuciones de sistemas operativos que las integran y que se especializan en la ejecución de dichas pruebas. Un ejemplo de estos sistemas operativos y el cual es usado en las pruebas presentadas en este documento es la distribución Kali Linux (anteriormente Backtrack Linux) la cual es basada en el sistema operativo Debian (Kali Linux, 2016). Es una distribución Libre, Open Source y bajo licencia GNU (al

igual que Debian) creada por la compañía Offensive Security. Tiene entre muchas otras ventajas, las siguientes (Offensive Security, 2015):

1. Más de 300 herramientas utilizadas para pruebas de penetración
2. Amplio apoyo a dispositivos inalámbricos
3. Entorno de desarrollo seguro

Correspondiente a la última fase se realiza una consolidación y reporte de resultados haciendo uso del diccionario CVE (Common Vulnerabilities and Exposures).

El diccionario CVE de nombres o identificadores es utilizado para publicar vulnerabilidades conocidas de la seguridad de la información, haciendo más fácil compartir datos y evaluar la cobertura de una organización en materia de seguridad. Las ventajas del CVE son las siguientes (CVE, 2015):

1. Provee un nombre para cada una de las vulnerabilidades
2. Otorga una descripción estandarizada para cada una de las vulnerabilidades
3. Establece un diccionario en vez de una base de datos
4. Establece un solo lenguaje para herramientas y bases de datos.
5. Indica la forma de operar y mejorar la cobertura de seguridad
6. Establece una base de evaluación sobre herramientas y bases de datos
7. Es libre para descarga y uso publico

Finalmente se realiza un análisis de riesgos de seguridad de la información basado en los resultados obtenidos de las pruebas ejecutadas haciendo uso de las normas ISO 30001:2009 e ISO/IEC 31010:2009, proponiendo con base en estas posibles soluciones a los riesgos que se identifican.

La norma ISO 30001 surge con el fin de ‘proponer unas pautas genéricas sobre cómo gestionar los riesgos de forma sistemática y transparente.’ (Avantium, Business Consulting, 2015). Esta norma tiene tres elementos en los cuales se centra para lograr una efectiva gestión de riesgos:

1. Los principios para la gestión de riesgos.
2. La estructura de soporte.
3. El proceso de gestión de riesgos.

Por otro lado, se apoya a su vez en la norma ISO/IEC 31010:2009 la cual provee varios artefactos o herramientas para la evaluación de riesgos como parte del proceso de gestión de riesgos listado anteriormente. Para este caso, se lleva a cabo dicho proceso basando los criterios y estructura de la matriz de riesgos cualitativa y el catálogo de reducción de riesgos, que se expone en el artículo de Nates Parra con base en estas normas (Nates Parra, 2011).

En este proyecto de grado por lo tanto, se lleva a cabo el proceso de hallazgo de vulnerabilidades y de riesgos en Quadodo Login Script, consolidando a través de herramientas de gestión de riesgos los resultados encontrados y el nivel de impacto que estos tienen en materia de seguridad de la información, generando de acuerdo a esto recomendaciones y acciones de mejora para mitigarlos.

6.3 MARCO LEGAL

Ya que Quadodo Login Script puede ser integrado a cualquier sistema desarrollado en PHP, es de código abierto y su distribución y descarga es libre, el mismo puede ser entonces utilizado en diferentes sistemas a nivel mundial. Es por esto que es importante conocer el contexto legal que aplica en Colombia en caso tal que un sistema desarrollado en este país desee integrar un administrador de permisos y de usuarios a sistemas como lo es Quadodo Login Script.

6.3.1 Ley 1273 de 2009

Inicialmente, la ley indica los delitos que pueden ser perpetrados en un sistema informático y que son castigados por la ley, lo cual ayuda a evaluar si es conveniente la integración de Quadodo por temor a que se presenten vulnerabilidades que dejen la puerta abierta a dichos delitos con o sin intención. Dichos delitos están contenidos en la Ley 1273 de 2009, la cual es una modificación del Código Penal que se denomina “de la protección de la información y de los datos” (LEY 1273 DE 2009, 2009). Está dividida en 2 capítulos: el primero hace referencia a los atentados contra la confidencialidad, la integridad y la disponibilidad de los datos y de los sistemas informáticos y el segundo capítulo hace referencia a atentados informáticos y otras infracciones. En cada uno de ellos se detallan las acciones que incurren en el delito y los castigos correspondientes.

6.3.2 Ley estatutaria 1266 de 2008

Por otro lado, es necesario que se cumplan los estatutos establecidos en los artículos legales relacionados con el hábeas data, es decir “el derecho fundamental que tiene toda persona para conocer, actualizar y rectificar toda aquella información que se relacione con ella y que se recopile o almacene en bancos de datos” (Vicepresidencia Jurídica y Administrativa del Fondo Nacional de Garantías, 2008). Es pues, la ley estatutaria 1266 de 2008 la que dictamina las disposiciones relacionadas con el hábeas data y se regula a través de ella el manejo de información almacenada en bases de datos personales; es una ley dirigida especialmente al sector financiero, crediticio, comercial, de servicios y a toda información proveniente de otros países. Enmarca entre otras cosas todos los principios de administración de datos, destacando entre estos el numeral ***f*** del artículo 4, correspondiente al Principio de seguridad donde señala: “*La información que conforma los registros individuales constitutivos de los bancos de datos a que*

se refiere la ley, así como la resultante de las consultas que de ella hagan sus usuarios, se deberá manejar con las medidas técnicas que sean necesarias para garantizar la seguridad de los registros evitando su adulteración, pérdida, consulta o uso no autorizado" (Alcaldía de Bogotá, 2008).

Este último es el que se busca salvaguardar con el proceso llevado a cabo en el presente proyecto el cual va de la mano con otros principios como el principio de confidencialidad y el principio de veracidad o calidad de los registros o datos, entre otros señalados por la ley 1266.

6.4 MARCO CONCEPTUAL

***CVE*:** Son las siglas de Common Vulnerabilities and Exposures (Vulnerabilidades y Exposiciones comunes en español), es un diccionario centralizado de libre uso y de libre descarga donde se registran vulnerabilidades de seguridad conocidas en la red a las cuales se les asigna un identificador (CVE ID) teniendo como objetivo la clasificación de vulnerabilidades por herramientas, repositorios y servicios. (CVE, 2015)

Ethical Hacking: Conocido en español como Hackeo Ético, es la disciplina de seguridad de la información que se encarga de detectar y explotar las vulnerabilidades sobre sistemas de información con el fin de evaluar y verificar la seguridad tanto física como lógica de los mismos, ejecutando pruebas de intrusión y de penetración como lo haría un atacante. Todo esto. Los resultados obtenidos de estas pruebas permiten tener claro el estado del sistema de interés en materia de seguridad de la información y llevar a cabo decisiones estableciendo acciones de mejora para mitigar o erradicar los riesgos y vulnerabilidades presentes en el sistema. (Reyes Plata, 2010).

ISO/IEC: Estas siglas hacen referencia a las directivas, normas y políticas aplicados a procedimientos, definidos como estándares internacionales para asegurar que los servicios y/o productos sean realizados y ofrecidos con calidad. (ISO, 2017).

Kali Linux: Sistema operativo basado en Debian creado por la empresa Offensive Security que contiene herramientas para realizar pruebas y ataques de seguridad de la información desde el punto de vista del hacking ético. (Kali Linux, 2016)

OWASP: Es la sigla de Open Web Application Security Project (Proyecto Abierto de Seguridad en Aplicaciones Web), una fundación sin ánimo de lucro cuyo objetivo es promover e incrementar la seguridad del software y la web en general. (OWASP, 2014)

Pentesting: Textualmente en la página OpenWebinars el autor Esaú A. define pentesting de la siguiente manera: "*Pentesting o Penetration Testing es la práctica*

de atacar diversos entornos con la intención de descubrir fallos, vulnerabilidades u otros fallos de seguridad, para así poder prevenir ataques externos hacia esos equipos o sistemas" (Esaú, 2015).

Seguridad de la Información: Según la norma ISO 27001, este término hace referencia a "la confidencialidad, la integridad y la disponibilidad de la información y los datos importantes para la organización, independientemente del formato que tengan" como formatos electrónicos, en físico, medios audiovisuales, etc. (SGSI, 2015)

SQL Injection: Es un tipo de ataque que tiene como objetivo leer y manipular datos sensibles de la base de datos "inyectando" sentencias SQL de selección, modificación y/o administración a través de campos o parámetros vulnerables en la aplicación o sistema objetivo. (OWASP, 2016)

XSS: Es la sigla de Cross-site Scripting, un tipo de ataque que realiza inyección de scripts o de código malicioso en sitios web a través del navegador web y usando diferentes mecanismos como cookies, sesiones de usuario, comandos javascript o etiquetas HTML. (OWASP, 2016)

7. RECURSOS DEL PROYECTO

7.1 SOFTWARE

Tabla 1. Recursos de software

Recurso	*Costo*	*Descripción*
Windows 10 Pro	$ 0	Sistema Operativo OEM (Original equipment manufacturer), es decir, preinstalado en el equipo que será utilizado para la ejecución de pruebas
WAMP Server 2.5	$ 0	Servidor web para aplicaciones PHP gratuito que soporta aplicaciones desarrolladas en la versión 5.5.0 de este lenguaje
Netbeans IDE	$ 0	Herramienta gratuita de desarrollo para aplicaciones PHP
Kali Linux	$ 0	Plataforma gratuita basada en Linux y Unix que provee las herramientas necesarias para la ejecución de pentesting
VirtualBox	$ 0	Programa para la instalación de máquinas virtuales. Se configurará e instalará en este programa Kali Linux.

Fuente: El autor.

7.2 HARDWARE E IT

Tabla 2. Recursos de Hardware e IT

Recurso	*Costo*	*Descripción*
Dell XPS 14z	$150.000 (mensual)	Equipo que será utilizado para la instalación de programas y herramientas de penetración. El costo indicado corresponde al alquiler mensual del equipo.
Internet Mi-Fi	$70.000 (mensual)	Medio utilizado para las consultas y acceso a la web.

Fuente: El autor.

7.3 MANO DE OBRA

Tabla 3. Mano de Obra

Recurso	***Costo***	***Descripción***
Ingeniero de Sistemas	$40.000 (hora)	Corresponde a la mano de obra del ejecutor de esta propuesta y que corresponde al autor de este documento. Este es un costo autogestionable y autocosteable pero que se refleja como parte de los recursos a integrar a la propuesta.

Fuente: El autor.

8. DISEÑO METODOLÓGICO DEL PROYECTO

8.1 TIPO DE INVESTIGACIÓN

Se realiza una investigación aplicada la cual se encamina a aplicar los conocimientos adquiridos a lo largo de la especialización en seguridad informática, ejecutando pruebas de pentesting definidas e identificadas por OWASP (The Open Web Application Security Project) con el objetivo de identificar posibles vulnerabilidades de seguridad de la información en Quadodo Login Script que puedan poner en riesgo a los sistemas que lo integren como parte de su administración de usuarios y permisos.

8.2 TÉCNICAS DE RECOLECCIÓN DE DATOS.

La presente monografía usa las pruebas aplicadas como principal fuente de datos, es aplicado mediante la ejecución de pruebas de pentesting a Quadodo Login Script, con el fin de validar si existen vulnerabilidades de seguridad informática en el mismo. Usa la observación de resultados como criterio para determinar la presencia de dichas vulnerabilidades usando como punto de referencia la cuarta versión de la guía de pruebas de OWASP.

8.3 TÉCNICAS DE ANÁLISIS DE DATOS.

Una vez obtenidos los resultados y datos pertinentes a las vulnerabilidades presentes en Quadodo Login Script se realiza basada en la guía OWASP mencionada en puntos anteriores la clasificación y tabulación de las mismas, haciendo uso también del diccionario CVE (Common Vulnerabilities and Exposures) mediante el cual se asigna un identificador único a cada una de ellas (codificación de resultados). Se establece finalmente una relación de riesgos aplicado el análisis preliminar de riesgos y matrices de probabilidad y consecuencia como métodos de consulta establecidos en la norma ISO/IEC 31010:2009, proponiendo diferentes soluciones basadas en la experiencia y en la documentación técnica encontrada.

8.4 POBLACIÓN Y MUESTRA.

8.4.1 Población o universo

El universo o población lo constituyen todos los módulos y funcionalidades disponibles en Quadodo Login Script

8.4.2 Muestra

De la población indicada se seleccionan los módulos *Security, Administration y Group Control Panel* de los cuales se evalúan las siguientes funcionalidades sobre las cuales se ejecutan pruebas de pentesting:

- ***Security:*** Autenticación de usuarios en la aplicación, límite en el número de intentos de acceso a la aplicación y protección de información de entrada de inyección SQL (SQL Injection).

- ***Administration:*** Agregar, editar, eliminar y listar usuarios, grupos y máscaras de permisos.

- ***Group Control Panel:*** Usuarios en ciertos grupos no deben tener permisos en dichos grupos.

8.5 METODOLOGÍA DE DESARROLLO

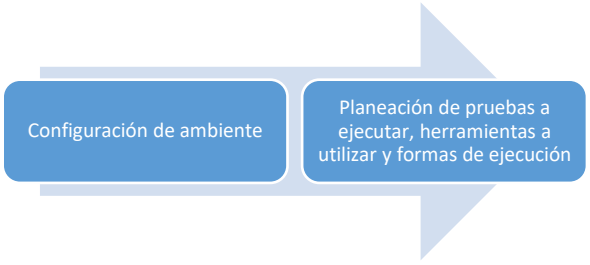
Para el desarrollo de la monografía planteada, se definen las siguientes etapas y procesos para llevarlo a feliz término.

8.5.1 Configuraciones y adecuación de ambiente:

Inicialmente, se realiza la virtualización del Sistema Operativo (S.O.) Kali Linux 2.0, el cual se ejecuta sobre un sistema Windows 8.1 Pro, posterior a esto se descarga y configura el script Quadodo siguiendo los pasos que se estipulan en la guía de usuario suministrada por el autor en el sitio web. (Rennehan, User Guide - Free Login Script, 2013)

Esta etapa consta de las siguientes actividades:

Figura. 1. Proceso de Inicial de Configuración de Ambiente



Fuente: El autor

8.5.2 Planeación, alcance y ejecución de las pruebas de pentesting

Una vez implementado y configurado el ambiente de pruebas, se documentó cada una de las etapas expuestas en este marco metodológico del presente documento para la ejecución de las pruebas de pentesting, las cuales a su vez fueron únicamente aplicadas a las funcionalidades listadas en el siguiente enlace: http://www.quadodo.net/features.php, abarcando las correspondientes a *Security, Administration y Group Control Panel*. Esto con el fin de acotar y dar un alcance especifico al proyecto planteado.

Esta fase consta de las siguientes actividades:

Figura. 2. Proceso de ejecución de pruebas de Pentesting

Fuente: El autor

8.5.3 Consolidación de resultados y generación de reporte

En la última fase del proceso de pentesting se procedió a realizar consolidación y reporte de los resultados obtenidos en cada una de las pruebas ejecutadas, resultados que reflejan tanto los casos exitosos como los casos que evidenciaron vulnerabilidades y que se encontraron en las funcionalidades indicadas anteriormente del script Quadodo.

El reporte de resultados contiene algunas gráficas que muestran de forma cuantitativa el número y porcentaje de casos exitosos y casos no exitosos detectados, así como las funcionalidades donde se concentra la mayor parte de las vulnerabilidades.

8.5.3.1 Análisis de resultados y propuestas de opción de mejora.

Finalmente se dispuso a plasmar en una matriz las vulnerabilidades que hayan sido encontradas en las funcionalidades indicadas del script Quadodo usando CVE como el marco de referencia para la documentación de las mismas. Una vez hecho esto, en el formato del catálogo de reducción de riesgos expuesto por Nates Parra (Nates

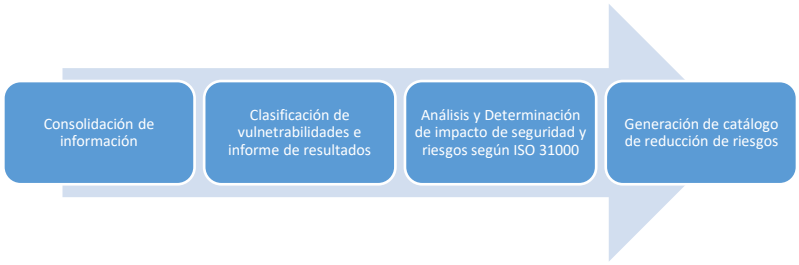
Parra, 2011) se procederán a registrar las recomendaciones o work-arounds que permitan subsanar las vulnerabilidades y por ende los riesgos detectados.

8.5.3.2 Documentación de evidencias y consolidación de monografía

Este es un proceso transversal y constante en cada una de las actividades a desarrollar, puesto que este es el insumo para la monografía a entregar como producto de este proyecto, esta documentación involucra screenshots, lista de vulnerabilidades encontradas, pasos ejecutados y cualquier factor documental que sirva de evidencia.

Esta fase consta de las actividades indicadas en la figura a continuación:

Figura. 3. Proceso de consolidación de resultados



Fuente: El autor

9. CONFIGURACIÓN Y ADECUACIÓN DEL AMBIENTE

9.1 INSTALACIÓN DE COMPONENTES NECESARIOS

Se realizó la instalación y actualización de los componentes necesarios para desplegar el script Quadodo. Para ello se requiere instalar php5, la extensión PEARL de php, php5-xcache y el módulo de php para apache 2.

Para realizar lo indicado anteriormente se utilizó el comando **`apt-get install`** en la terminal de Kali Linux 2.0 (de ahora en adelante y para efectos de abreviación en este documento **KL2**) el cual ejecuta la herramienta avanzada de empaquetamiento de dicho sistema operativo que permite la instalación y actualización de paquetes del sistema (Ubuntu, 2016)

Junto con mencionado comando se instalaron los módulos indicados de la siguiente manera:

- `apt-get install libapache2-mod-php5 php5 php-pear php5-xcache`
- `apt-get install php5-mysql`

Figura. 4. Instalación de componentes necesarios

```
root@6garzon:~# apt-get install libapache2-mod-php5 php5 php-pear php5-
xcache
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
libapache2-mod-php5 ya está en su versión más reciente.
fijado libapache2-mod-php5 como instalado manualmente.
php5 ya está en su versión más reciente.
Los paquetes indicados a continuación se instalaron de forma automática
y ya no son necesarios.
  apache2.2-bin apache2.2-common bkhive docutils-doc geoclue geoclue-
hostip geoclue-localnet geoclue-manual geoclue-yahoo gir1.2-clutter-
gst-1.0 gir1.2-folks-0.6 gir1.2-gee-1.0 gir1.2-gst-plugins-base-0.10
  gir1.2-gstreamer-0.10 gir1.2-javascriptcoregtk-1.0 gir1.2-webkit-1.0
gnome-js-common gnuradio-dev gstreamer0.10-ffmpeg html2text latex-
beamer latex-xcolor libafpclient0 libcamel-1.2-33 libclutter-gst-1.0-0
  libclutter-imcontext-0.1-0 libclutter-imcontext-0.1-bin
libcluttergesture-0.0.2-0 libcolord1 libcrypt-passwdmd5-perl libdconf0
libdrm-nouveau1a libebackend-1.2-2 libecal-1.2-11 libedata-cal-1.2-15
  libedataserver-1.2-16 libfilter-perl libgcr-3-1 libgdata13 libgee2
libgeoclue0 libgnome-bluetooth10 libgnome-media-profiles-3.0-0
libgnuradio-analog3.6.5.1 libgnuradio-atsc3.6.5.1 libgnuradio-
audio3.6.5.1
  libgnuradio-blocks3.6.5.1 libgnuradio-comedi3.6.5.1 libgnuradio-
digital3.6.5.1 libgnuradio-fcd3.6.5.1 libgnuradio-fcdproplus
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Figura. 4. (Continuación)

```
libgnuradio-fft3.6.5.1 libgnuradio-filter3.6.5.1 libgnuradio-noaa3.6.5.1
  libgnuradio-pager3.6.5.1 libgnuradio-qtgui3.6.5.1 libgnuradio-trellis3.6.5.1
libgnuradio-uhd3.6.5.1 libgnuradio-video-sdl3.6.5.1 libgnuradio-vocoder3.6.5.1
libgnuradio-wavelet3.6.5.1 libgphoto2-2
  libgphoto2-port0 libgraphite3 libical0 libimobiledevice2 libjson0
libmusicbrainz5-0 libmx-1.0-2 libmx-bin libmx-common libncp libntfs10
libosmocodec0 libosmocore libosmocore4 libosmogb2 libosmogsm4
  libosmovty0 libplist1 libplrpc-perl libpthread-stubs0 libqwtplot3d-qt4-0
librest-extras-0.7-0 libseed-gtk3-0 libsocialweb-client2 libsocialweb-common
libsocialweb-service libsocialweb0 libssl-dev libssl-doc
  libsystemd-daemon0 libsystemd-login0 libt1-5 libtelepathy-farstream2
libtelepathy-logger2 libts-0.0-0 libusbmuxd1 libvte-2.90-9 libvte-2.90-common
libxatracker1 libyaml-syck-perl
  linux-headers-3.12-kali1-common linux-kbuild-3.12 luatex memtest86+ nautilus-
sendto openssh-blacklist openssh-blacklist-extra pgf python-utidylib syslinux-
themes-debian syslinux-themes-debian-wheezy
  texlive-luatex tsconf wwwconfig-common
Use 'apt-get autoremove' to remove them.
Paquetes sugeridos:
  php5-dev
Se instalarán los siguientes paquetes NUEVOS:
  php-pear php5-xcache
0 actualizados, 2 se instalarán, 0 para eliminar y 341 no actualizados.
Necesito descargar 392 kB de archivos.
Se utilizarán 2.434 kB de espacio de disco adicional después de esta operación.
¿Desea continuar [S/n]? S
Des:1 http://security.kali.org/kali-security/ sana/updates/main php-pear all
5.6.13+dfsg-0+deb8u1 [268 kB]
Des:2 http://http.kali.org/kali/ sana/main php5-xcache i386 3.2.0-1 [124 kB]
Descargados 392 kB en 2seg. (177 kB/s)
Seleccionando el paquete php5-xcache previamente no seleccionado.
(Leyendo la base de datos ... 353325 ficheros o directorios instalados
actualmente.)
Preparando para desempaquetar .../php5-xcache_3.2.0-1_i386.deb ...
Desempaquetando php5-xcache (3.2.0-1) ...
Seleccionando el paquete php-pear previamente no seleccionado.
Preparando para desempaquetar .../php-pear_5.6.13+dfsg-0+deb8u1_all.deb ...
Desempaquetando php-pear (5.6.13+dfsg-0+deb8u1) ...
Configurando php5-xcache (3.2.0-1) ...
php5_invoke: Enable module xcache for cli SAPI
php5_invoke: Enable module xcache for apache2 SAPI

Configurando php-pear (5.6.13+dfsg-0+deb8u1) ...
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Seguidamente, se instaló mysql

Figura. 5. Instalación mysql y PHP5

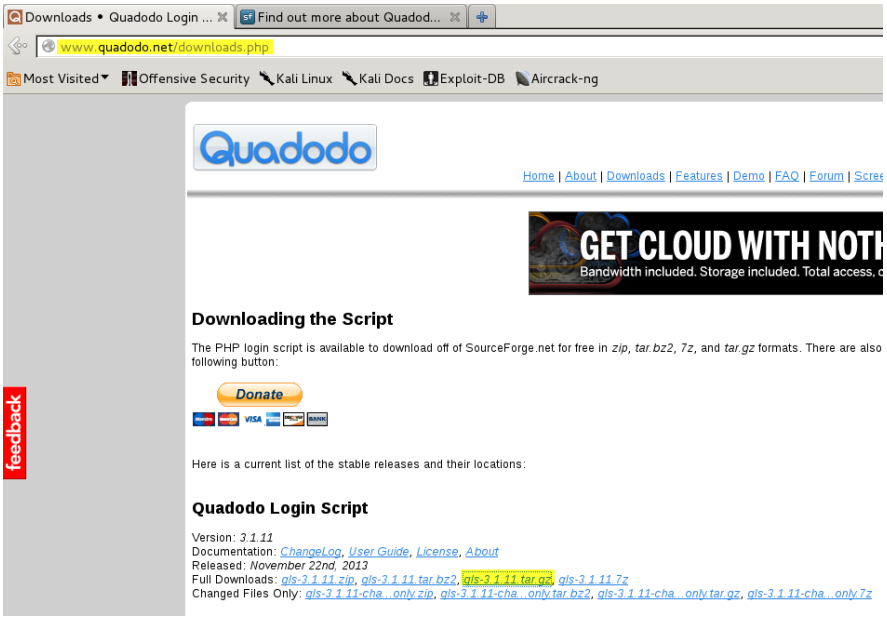
```
root@6garzon:~# apt-get install php5-mysql
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
php5-mysql ya está en su versión más reciente.
Los paquetes indicados a continuación se instalaron de forma automática
y ya no son necesarios.
  apache2.2-bin apache2.2-common bkhive docutils-doc geoclue geoclue-
hostip geoclue-localnet geoclue-manual geoclue-yahoo gir1.2-clutter-
gst-1.0 gir1.2-folks-0.6 gir1.2-gee-1.0 gir1.2-gst-plugins-base-0.10
  gir1.2-gstreamer-0.10 gir1.2-javascriptcoregtk-1.0 gir1.2-webkit-1.0
gnome-js-common gnuradio-dev gstreamer0.10-ffmpeg html2text latex-
beamer latex-xcolor libafpclient0 libcamel-1.2-33 libclutter-gst-1.0-0
  libclutter-imcontext-0.1-0 libclutter-imcontext-0.1-bin
libcluttergesture-0.0.2-0 libcolord1 libcrypt-passwdmd5-perl libdconf0
libdrm-nouveau1a libebackend-1.2-2 libecal-1.2-11 libedata-cal-1.2-15
[...]
  libsystemd-daemon0 libsystemd-login0 libt1-5 libtelepathy-farstream2
libtelepathy-logger2 libts-0.0-0 libusbmuxd1 libvte-2.90-9 libvte-2.90-
common libxatracker1 libyaml-syck-perl
  linux-headers-3.12-kali1-common linux-kbuild-3.12 luatex memtest86+
nautilus-sendto openssh-blacklist openssh-blacklist-extra pgf python-
utidylib syslinux-themes-debian syslinux-themes-debian-wheezy
  texlive-luatex tsconf wwwconfig-common
Use 'apt-get autoremove' to remove them.
0 actualizados, 0 se instalarán, 0 para eliminar y 341 no actualizados.
root@6garzon:~#
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

9.2 DESPLIEGUE Y CONFIGURACIÓN DE QUADODO

Para el proceso de instalación se procedió a realizar la instalación del software descargando el paquete de instalación disponible en la página web de descargas Quadodo:

Figura. 6. Página de descarga de instalador de Quadodo



Fuente: Página oficial de Quadodo Login Script

Se descomprimió el archivo descargado

Figura. 7. Descompresión de archivo instalador de Quadodo

```
root@6garzon:/home/henrygarzon# cd
/home/henrygarzon/ProyectoGrado/Quadodo/
root@6garzon:/home/henrygarzon/ProyectoGrado/Quadodo# ls
root@6garzon:/home/henrygarzon/ProyectoGrado/Quadodo# tar -zxvf qls-
3.1.11.tar.gz
qls-3.1.11/
qls-3.1.11/activate.php
qls-3.1.11/admincp.php
qls-3.1.11/change_password.php
qls-3.1.11/docs/
qls-3.1.11/docs/ABOUT
qls-3.1.11/docs/CHANGELOG
qls-3.1.11/docs/FONTS
qls-3.1.11/docs/GPL
qls-3.1.11/docs/USERGUIDE.css
qls-3.1.11/docs/USERGUIDE.html
qls-3.1.11/docs/USERGUIDE.jpeg
qls-3.1.11/groupcp.php
qls-3.1.11/html/
qls-3.1.11/html/admin_add_group_form.php
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Figura. 7. (Continuación)

```
qls-3.1.11/html/admin_add_mask_form.php
qls-3.1.11/html/admin_add_page_form.php
qls-3.1.11/html/admin_add_user_form.php
qls-3.1.11/html/admin_configuration_form.php
qls-3.1.11/html/admin_edit_group_form.php
qls-3.1.11/html/admin_edit_group_real_form.php
qls-3.1.11/html/admin_edit_mask_form.php
qls-3.1.11/html/admin_edit_mask_real_form.php
[...]
qls-3.1.11/install/updates/
qls-3.1.11/login.php
qls-3.1.11/login_process.php
qls-3.1.11/logout.php
qls-3.1.11/members.php
qls-3.1.11/register.php
qls-3.1.11/security_image.php
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Se procedió a crear directorio de publicación y directorio de archivos de log para la aplicación en la carpeta de despliegue de Apache

Figura. 8. Creación de directorios de publicación y de log

```
root@6garzon:/home/henrygarzon/ProyectoGrado/Quadodo# mkdir -p
/var/www/Quadodo/public_html
root@6garzon:/home/henrygarzon/ProyectoGrado/Quadodo# mkdir
/var/www/Quadodo/logs
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Se otorgaron permisos a la carpeta creada para la aplicación:

Figura. 9. Otorgar permisos a carpeta Quadodo

```
root@6garzon:/var/www/Quadodo/logs# chown -R www-data:www-data
/var/www/Quadodo
root@6garzon:/var/www/Quadodo/logs# chmod -R 755 /var/www
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Se copiaron los archivos de publicación que se descomprimieron en el directorio de publicación creado

Figura. 10. Copia de archivos Quadodo a directorio de publicación

```
root@6garzon:~# mv /home/henrygarzon/ProyectoGrado/Quadodo/qls-3.1.11/*
/var/www/Quadodo/public_html/
root@6garzon:~# cd /var/www/Quadodo/public_html/
root@6garzon:/var/www/Quadodo/public_html# ls
activate.php  admincp.php  change_password.php  docs  groupcp.php  html
includes  install  login.php  login_process.php  logout.php
members.php  register.php  security_image.php
root@6garzon:/var/www/Quadodo/public_html#
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

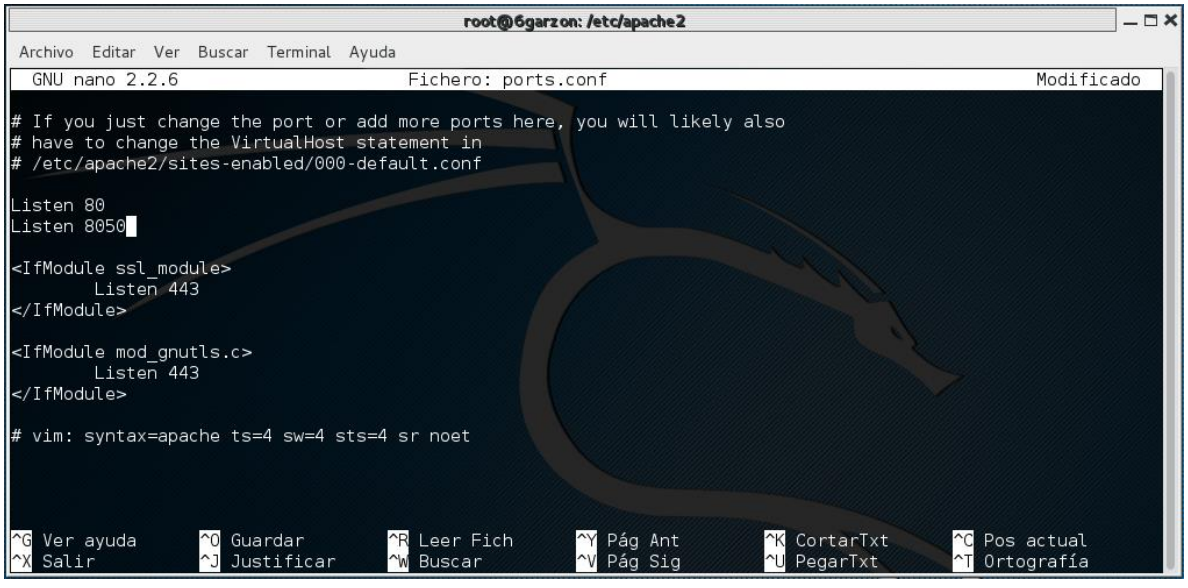
Se configuró el puerto de escucha de apache, el sitio fue desplegado en el puerto 8050. Una vez hecho esto se reinició el servidor.

Figura. 11. Modificación puerto de escucha apache

```
root@6garzon:/etc/apache2# ls
apache2.conf  conf-available  conf-enabled  envvars  magic  mods-
available  mods-enabled  ports.conf  sites-available  sites-enabled
root@6garzon:/etc/apache2# pico ports.conf
root@6garzon:/etc/apache2# service apache2 restart
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Figura. 12. Modificación archivo puertos escucha



Fuente: El autor. Archivo de configuración de puertos apache.

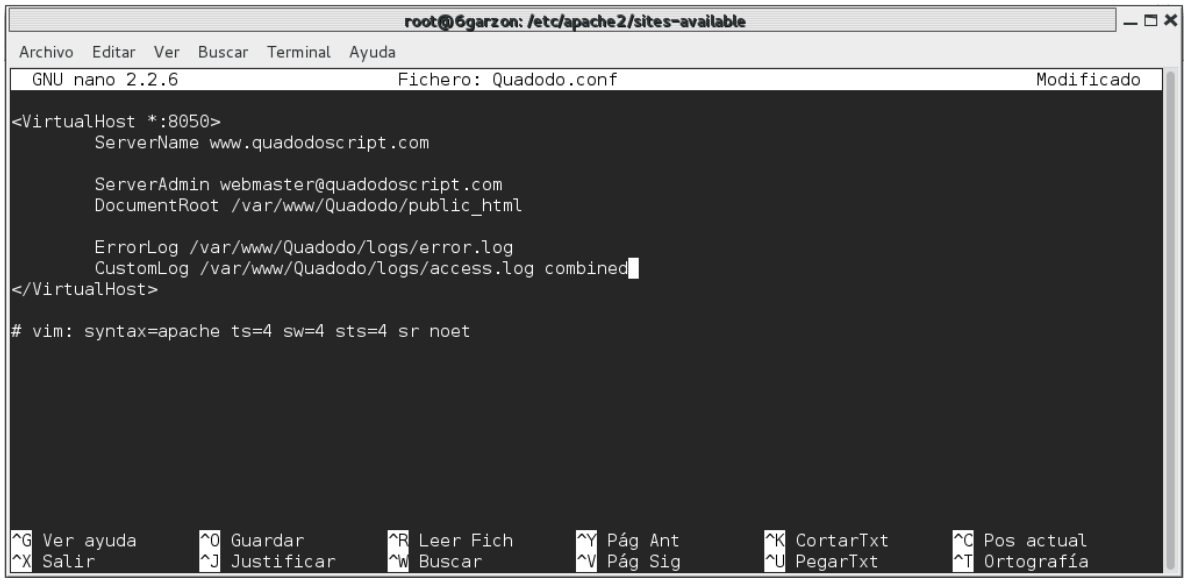
Se creó el archivo de configuración del sitio y se ajustan los parámetros del mismo indicando que el puerto de escucha es el 8050

Figura. 13. Creación archivo configuración

```
root@6garzon:~# cd /etc/apache2/sites-available/
root@6garzon:/etc/apache2/sites-available# pico Quadodo.conf
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

Figura. 14. Archivo de configuración del sitio



Fuente: El autor. Archivo de configuración sitio Quadodo en apache

Finalmente se habilitó el sitio y se reiniciaron los servicios de apache.

Figura. 15.Habilitación sitio Quadodo

```
root@6garzon:~# a2ensite Quadodo.conf
Enabling site Quadodo.
To activate the new configuration, you need to run:
  service apache2 reload
root@6garzon:~# service apache2 start
root@6garzon:~# service apache2 reload
root@6garzon:~# service apache2 restart
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

9.3 CREACIÓN DE BASE DE DATOS

Se creó la base de datos a utilizar la cual se nombró QuadodoBD.

Figura. 16. Creación de Base de Datos Quadodo

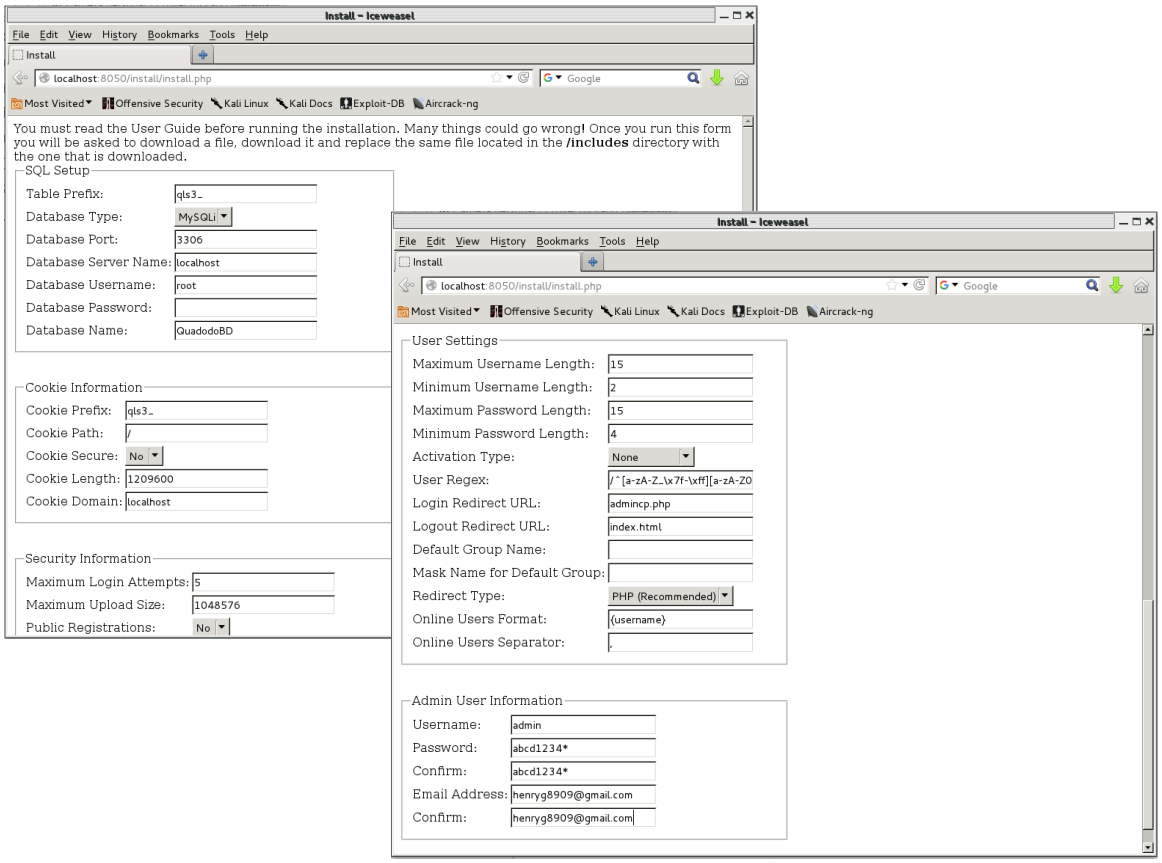
```
root@6garzon:~# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 43
Server version: 5.5.44-0+deb8u1 (Debian)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

mysql> create database QuadodoBD
    -> ;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| QuadodoBD          |
| dvwa               |
| mysql              |
| performance_schema |
+--------------------+
5 rows in set (0.01 sec)
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux

9.4 INSTALACIÓN DE QUADODO

De acuerdo al manual de configuración que se provee en la página guía de usuario de Quadodo http://dev.quadodo.net/qls-3.1.11/USERGUIDE.html, se indicaron los siguientes datos para la instalación:

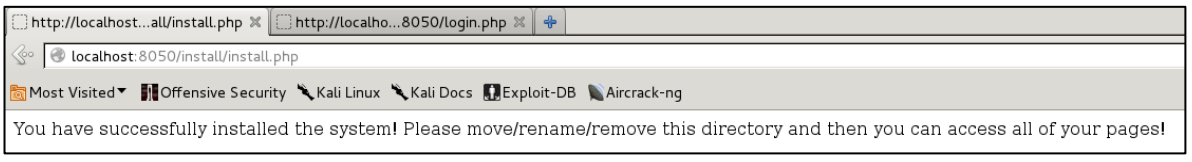
Figura. 17. Parámetros de instalación



Fuente. El autor. Screenshot de pantalla de Quadodo.

Una vez al hacer clic en el botón ***Install*** de la página, el sistema confirmó la instalación correcta del sistema, indicando renombrar por último la carpeta de instalación.

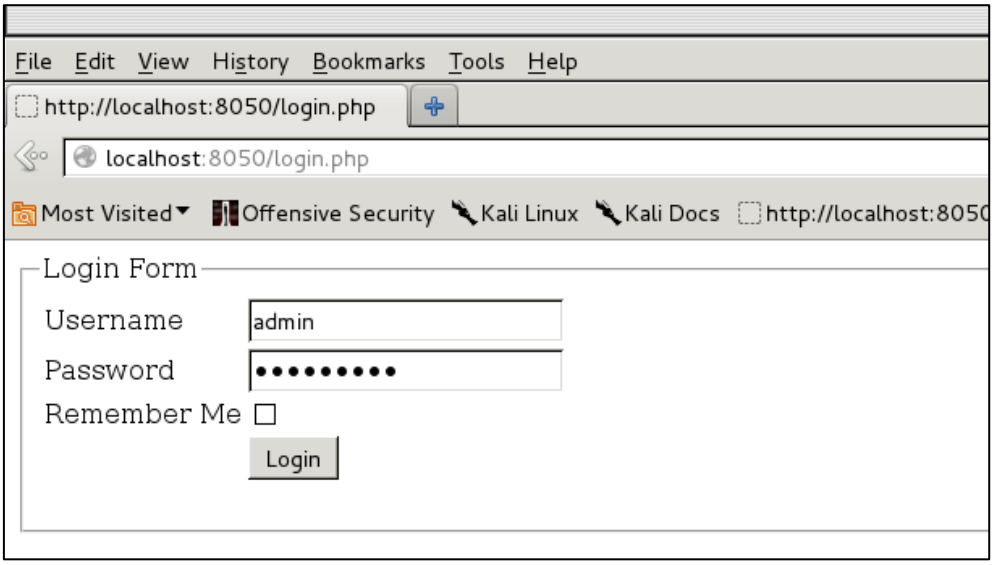
Figura. 18. Confirmación de instalación de Quadodo



Fuente. El autor. Screenshot de pantalla de Quadodo.

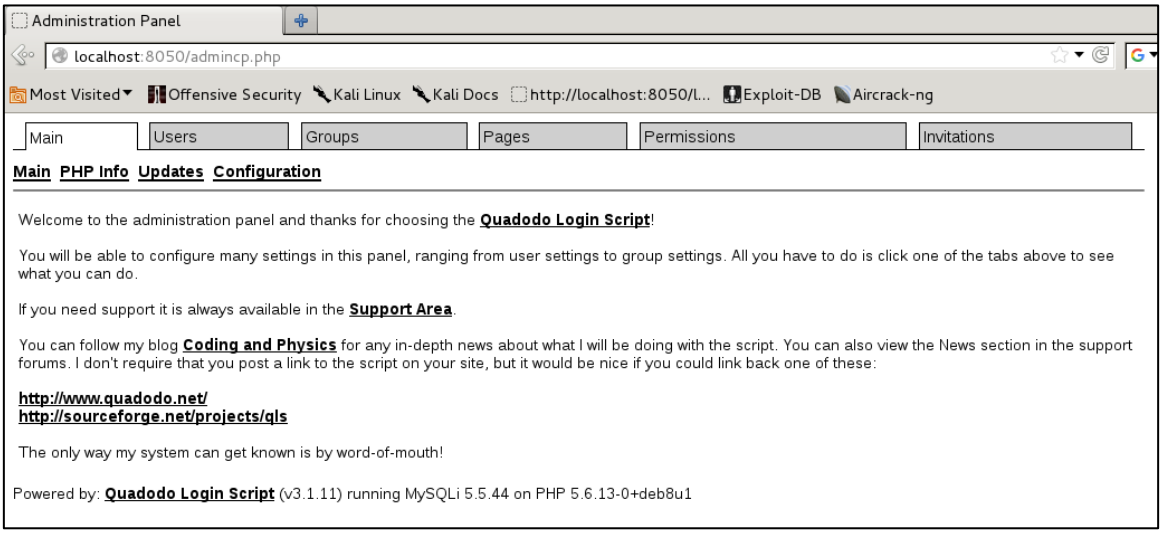
Se evidencia a continuación el acceso al sistema con las credenciales de administración suministradas:

Figura. 19. Página de Login de Quadodo



Fuente: El autor. Screenshot de pantalla de Quadodo.

Figura. 20. Página del panel de administración de Quadodo



Fuente: El autor. Screenshot de pantalla de Quadodo.

Se evidencia igualmente las tablas creadas en base de datos:

Figura. 21. Tablas de Quadodo creadas en base de datos

```
root@6garzon:/var/www/Quadodo/public_html# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.5.44-0+deb8u1 (Debian)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| QuadodoBD          |
| dvwa               |
| mysql              |
| performance_schema |
+--------------------+
5 rows in set (0.02 sec)

mysql> use QuadodoBD;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> show tables;
+------------------------+
| Tables_in_QuadodoBD    |
+------------------------+
| qls3_config            |
| qls3_groups            |
| qls3_invitations       |
| qls3_masks             |
| qls3_pages             |
| qls3_password_requests |
| qls3_security_image    |
| qls3_sessions          |
| qls3_users             |
+------------------------+

9 rows in set (0.00 sec)
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

10. RECONOCIMIENTO PREVIOS A LAS PRUEBAS DE PENTESTING

10.1 RECOLECCIÓN DE INFORMACIÓN (INFORMATION GATHERING)

Dentro de la metodología OWASP, se plantea una fase no etapa de recolección de información (Information Gathering) en la cual se identifica el objetivo a atacar a través de varias pruebas y actividades. Para este caso, se cuenta con la información de los módulos correspondientes y sus funcionalidades.

Por lo anterior, se indican a continuación las características de los módulos de la aplicación que fueron evaluados, el servidor donde se desplegó la aplicación, las rutas de despliegue y la base de datos que se utilizó para su funcionamiento.

10.1.1 Información de los módulos seleccionados

Según el alcance indicado para este proyecto los módulos seleccionados que se exploraron y verificaron tienen las siguientes funcionalidades:

10.1.1.1 Módulo de Seguridad (Security)

a. Passwords hashed

b. Límite en el número de intentos de ingreso

c. Toda la información de entrada es protegida contra SQL Injection.

d. Imagen de seguridad para evitar que robots se registren.

e. Limitar el acceso a ciertas páginas.

f. Limitar los privilegios de acceso de los administradores

g. Prohibir usuarios a los cuales no se desea otorgar acceso al sitio.

10.1.1.2 Módulo de Administración (Administration)

a. Ver la información de PHP information

b. Actualizaciones disponibles para el sistema.

c. Editar la configuración

d. Agregar / Editar / Eliminar / Listar usuarios

e. Activar usuarios inactivados

f. Configurar una máscara de permisos para cada usuario.

g. Agregar / Editar / Eliminar / Listar grupos personalizados

h. Configurar una máscara de permisos para cada grupo

i. Agregar / Editar / Eliminar / Listar páginas personalizadas

j. Subir páginas ya creadas desde el computador

k. Crear una página desde el panel de administración.

l. Editar una página desde el panel

m. Agregar / Editar / Eliminar / Listar máscaras de permisos

n. 21 diferentes permisos para configurar (más páginas personalizadas)

o. Enviar incitaciones a usuarios (en caso tal que se desee desactivar registro de usuarios públicos)

10.1.1.3 Panel de Control de Grupos

a. Permitir usuarios registrarse para grupos públicos

b. Los usuarios pueden abandonar grupos en los que están actualmente incluidos

c. Los líderes de los grupos pueden agregar, editar o remover usuarios de su grupo.

d. Los líderes de los grupos pueden ver estadísticas de su grupo

e. Los usuarios pueden llegar a ser líderes de más de un grupo

f. Los usuarios que estén incluidos en cierto grupo no tienen que tener todos los permisos de ese grupo.

El módulo de seguridad es transversal a la aplicación, uno de los puntos indica la protección que cuenta contra inyección de SQL, además de la encripción del password.

10.1.2 Información de infraestructura

10.1.2.1 Servidor y sistema operativo

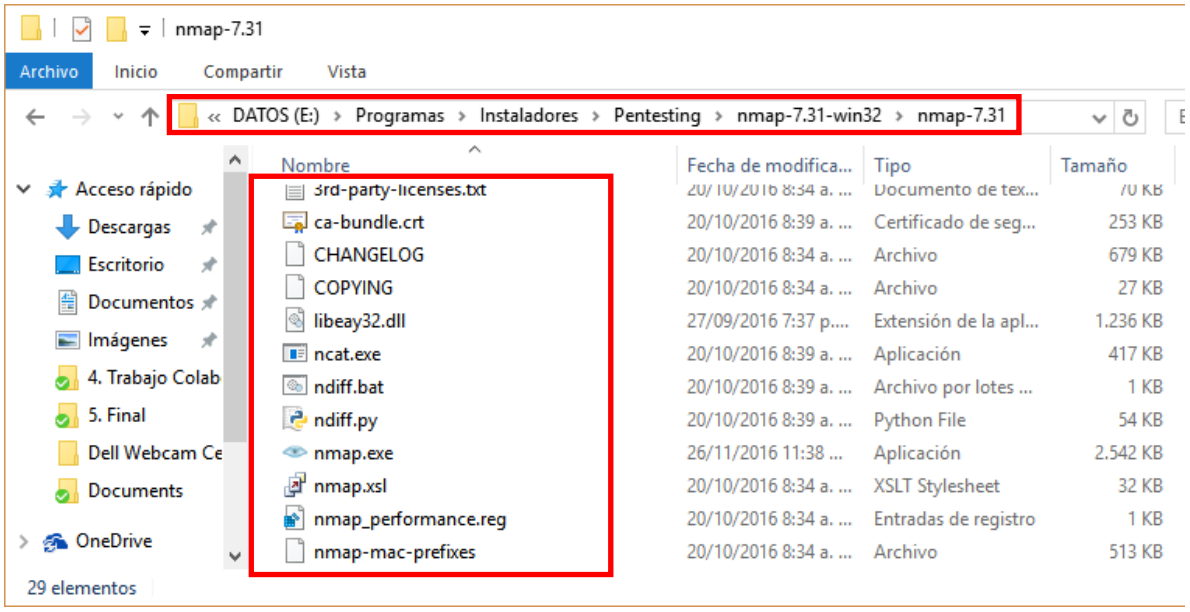
La aplicación Quadodo fue desplegada en servidor Apache, utilizando el puerto de escucha 8050 como se pudo observar en el proceso de instalación de este documento. Dicho servidor es ejecutado bajo el Sistema Operativo Kali Linux 2.0.

Sin embargo, estos datos se conocen debido al proceso de configuración y despliegue que se ha indicado en el presente proyecto. Un atacante debe realizar un proceso de pruebas de recolección de información (gathering) para obtener estos detalles y con base en estos proceder a ejecutar las pruebas de penetración. Para ello se usa la herramienta NMAP, la cual provee diferentes funcionalidades de exploración de red y de descubrimiento de información sobre servidores o terminales como sistema operativo, puertos y servicios expuestos con su correspondiente versión.

Para este caso el atacante inicia la recolección de información desde una terminal o equipo externo al servidor donde se encuentra desplegada la aplicación. Se usa entonces un equipo con sistema operativo Windows 10, en donde se ejecuta la versión 7.31 de NMAP para dicho sistema operativo disponible en la siguiente URL: https://nmap.org/dist/nmap-7.31-win32.zip. Es necesario para su funcionamiento y como prerrequisito instalar la última versión de la librería WinPcap que se encuentra en la siguiente URL: https://www.winpcap.org/install/default.htm

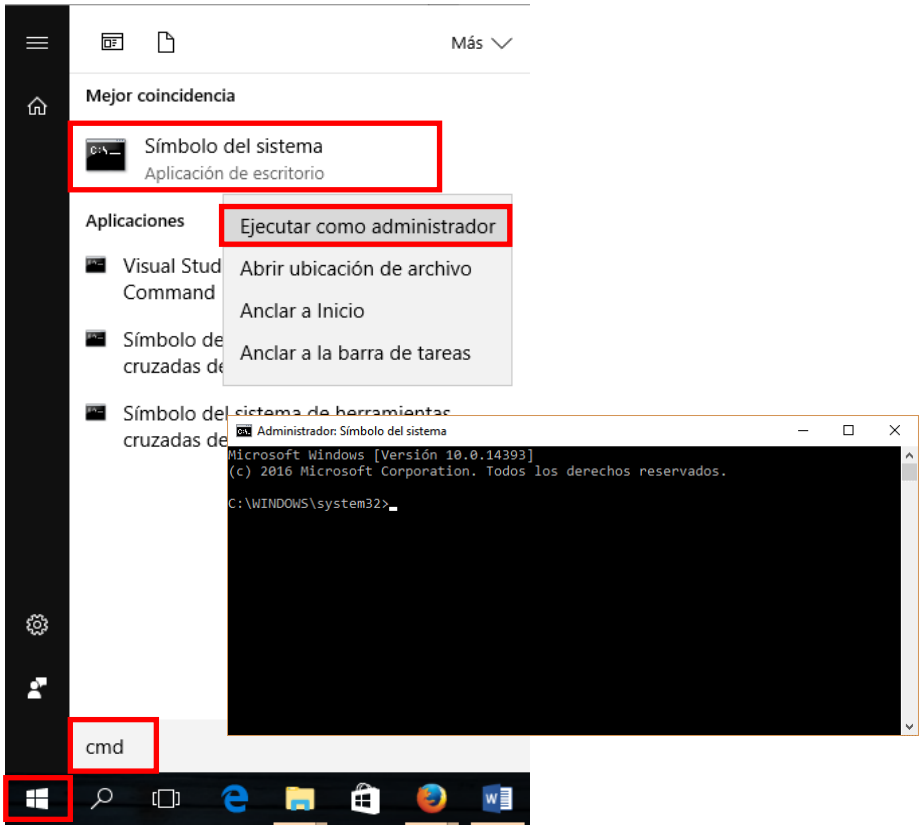
La herramienta NMAP se descomprime en el equipo y se ubica en una ruta del mismo y se abre la consola de comandos como administrador haciendo clic sobre el botón inicio (el cual tiene el logo de Windows), digitando ***cmd***, haciendo clic derecho sobre la opción de ***Símbolo del sistema*** y eligiendo la opción ***Ejecutar como administrador*** como figura en las siguientes imágenes.

Figura. 22. Ruta en el equipo Windows donde se guardó el ejecutable de la herramienta NMAP



Fuente: El autor. Screenshot de explorador de Windows.

Figura. 23. Apertura de consola en Windows 10



Fuente: El autor. Screenshot de menú de Windows.

En este caso se ha dejado la herramienta en la siguiente ruta en el equipo: ***E:\Programas\Instaladores\Pentesting\nmap-7.31-win32\nmap-7.31***. Se navega hasta dicha ruta a través de la consola abierta, digitando primero ***E:*** para ir a la unidad y después el comando ***cd*** seguido de la ruta indicada como se indica en la imagen a continuación.

Figura. 24. Acceso a la ruta del ejecutable de NMAP desde consola de comandos de Windows

Fuente: El autor. Screenshot de consola de comandos de Windows.

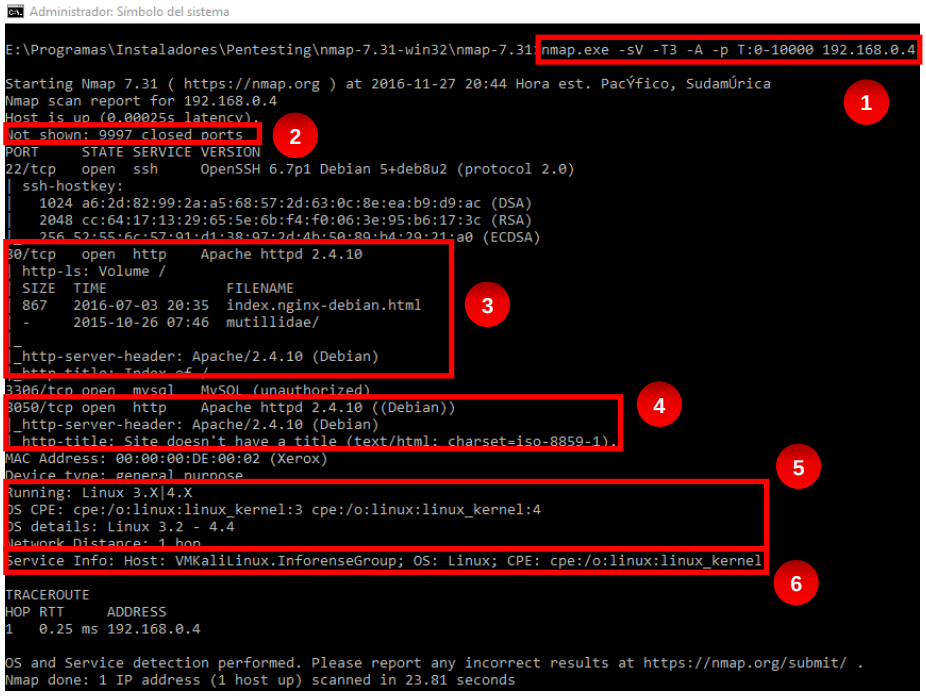
Ahora bien, se usan en la prueba las siguientes opciones de NMAP por medio de las cuales se obtiene información del sistema operativo, puertos disponibles y servicios expuestos en los mismos (NMAP.org, 2016).

- ***-sV***: Ejecuta pruebas sobre los puertos abiertos para determinar la información del servicio expuesto y la versión del mismo.
- ***-T***: Establece un tiempo de ejecución, acepta los valores del 1 al 5. Entre más grande el valor es más rápida la ejecución. Para este caso se usa el valor 3 para una velocidad media de ejecución.
- ***-A***: Esta opción habilita el descubrimiento de sistema operativo, la detección de la versión del mismo, el script de escaneo y trazas de red (traceroute).
- ***-p***: Permite indicar un rango de puertos a escanear y se pueden agrupar por protocolo. Para este caso se especifican para la primera prueba los puertos del 0 a 10000 para ser escaneados por protocolo TCP/IP de la siguiente manera: ***T:0-10000***.

Seguido de estas opciones se indica la IP o el nombre del HOST hacia el cual se ejecuta el comando. En este caso al momento de hacer la prueba la maquina destino (donde se desplegó Quadodo) tiene la IP 192.168.0.4, por lo cual el comando queda de la siguiente manera: ***nmap.exe -sV -T3 -A -p T:0-10000 192.168.0.4***

Al presionar la tecla Enter se ejecuta el comando arrojando el siguiente resultado:

Figura. 25. Resultado de ejecución NMAP sobre puertos y host especificados



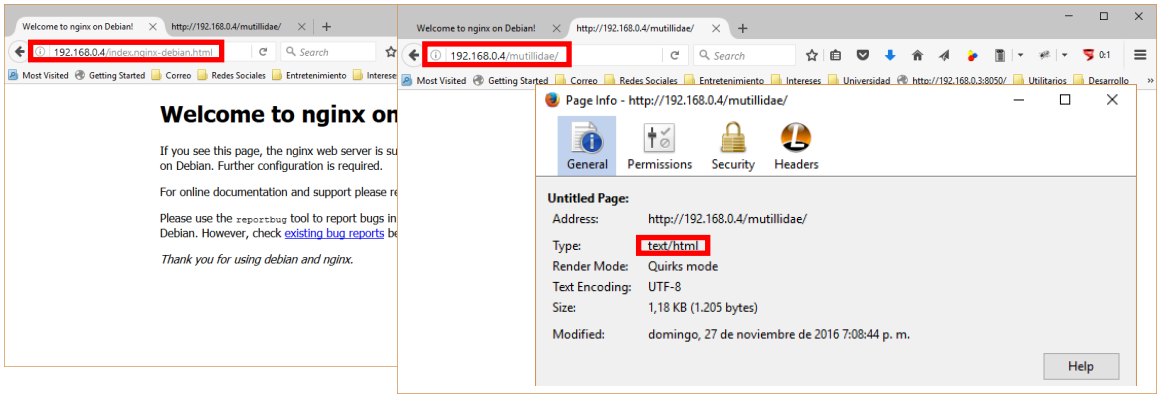
Fuente: El autor. Screenshot de consola de comandos de Windows.

En la figura se muestra en el recuadro número 1 el comando ejecutado, a continuación en el recuadro 2 se indica que hay 9998 puertos cerrados del rango de puertos proporcionados (0 al 10000).

En el recuadro número 3 de la figura se indica que en el puerto 80 está el servicio Apache versión 2.4.10 que se ejecuta sobre Debian como plataforma operativa, y que hay dos elementos expuestos en el mismo: ***index.nginx-debian.html*** y la carpeta ***mutillidae***. Al acceder en un navegador web (en este caso se usa Firefox) desde la terminal Windows, se verifica que efectivamente estos elementos se encuentran disponibles en dicho puerto.

La ruta de la carpeta ***mutillidae*** se muestra en blanco, sin embargo al revisar la información de la página se muestra que corresponde a un archivo HTML.

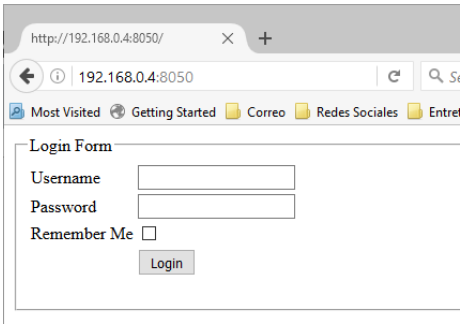
Figura. 26. Elementos expuestos en el puerto 80



Fuente: El autor. Screenshot de navegador Firefox ejecutándose en terminal Windows.

Continuando con el resultado, en el recuadro número 4 se indica que en el puerto 8050 hay un sitio expuesto que corre también sobre el servicio Apache versión 2.4.10. Efectivamente, aquí se encuentra instalado Quadodo, por lo cual el atacante logra identificar la ubicación de la aplicación en el servidor al acceder desde un navegador a la IP y puerto indicados.

Figura. 27. URL de Quadodo encontrada



Fuente: El autor. Screenshot de navegador Firefox ejecutándose en terminal Windows.

Respecto al sistema operativo en el recuadro número 5, NMAP indica que el sistema operativo es basado en Linux y que la versión del mismo se encuentra entre la 3.2 y la 4.4. Cuando NMAP no determina la versión o distribución exacta genera un resultado aproximado y lo presenta como se indica. Ahora bien según la documentación de Kali Linux 2.0, esta distribución está basada en Debian Jessie (Debian 8), el cual a su vez en su detalle técnico indica como versión de Kernel Linux versión 3.16, lo cual corresponde al resultado arrojado por NMAP (Kali, 2015) (Debian.org, 2016).

Por otro lado, como atacante se concluye que el sistema operativo es basado o es una versión de Debian debido a que en el servicio de Apache se indica que se

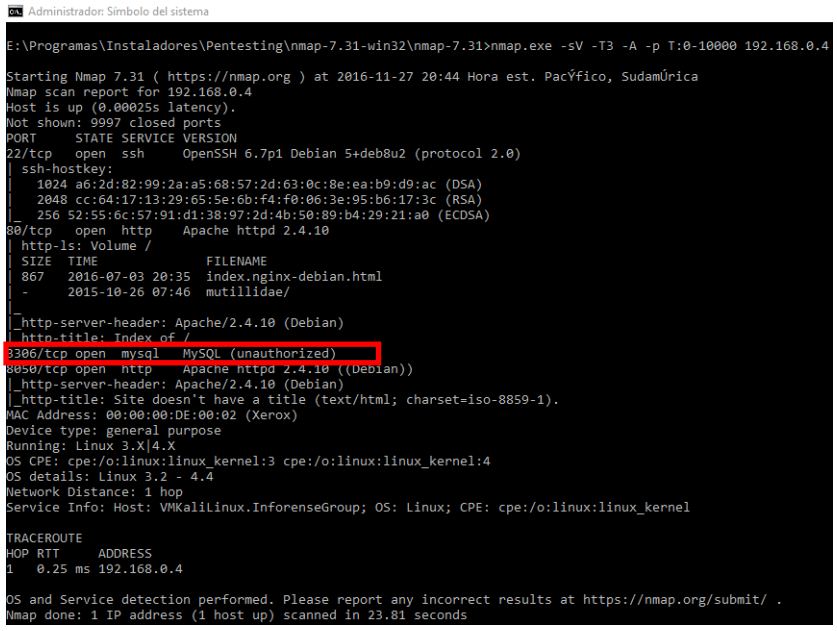
ejecuta sobre el mismo, además que el host especifica el nombre del sistema operativo como se muestra en el recuadro numero 6: ***VMKaliLinux.InforenseGroup***.

10.1.2.2 Base de datos

De acuerdo al proceso de instalación y configuración realizado el servidor de base de datos es MySQL y se ejecuta en el mismo servidor. La base de datos se llama QuadodoBD y el puerto de escucha es el 3306.

Volviendo al resultado obtenido a través de la herramienta NMAP, se indica que en el puerto 3306 se expone el servicio MySQL, detectando de esta manera el motor de base de datos utilizado y el puerto donde se ejecuta.

Figura. 28. Resultado de ejecución NMAP mostrando puerto donde se expone MySQL



Fuente: El autor. Screenshot de consola de comandos de Windows.

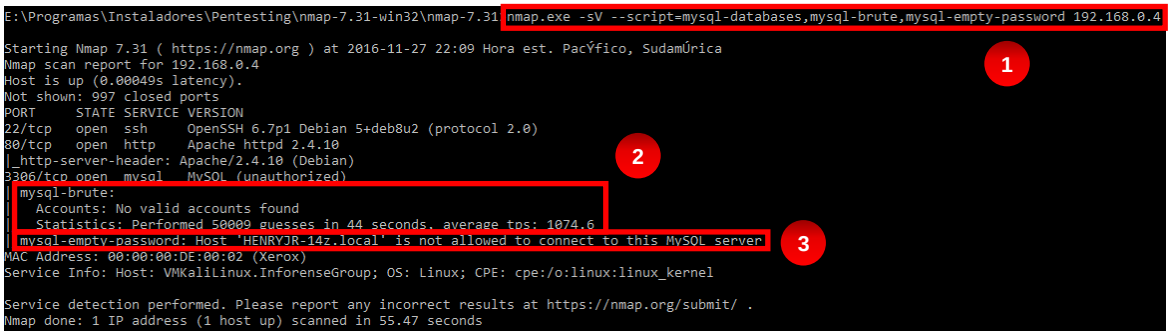
Ahora bien, se intenta a continuación listar las bases de datos MySQL disponibles usando la opción ***--script*** la cual permite ejecutar una rutina o script específicado sobre el host o IP enviado como argumento en el comando ***NMAP***. A dicha opción se le indica usar los siguientes scripts separados por comas:

- ***mysql-databases:*** Intenta listar las bases de datos disponibles en un servidor MySQL (NMAP.org, 2016).

- ***mysql-brute:*** Realiza intentos de acceso a través de un ataque de fuerza bruta, intentando diferentes credenciales y password (NMAP.org, 2016).
- ***mysql-empty-password:*** Intenta acceder al servidor indicado con una clave o password en blanco con las cuentas ***root*** o ***anonymous*** (NMAP.org, 2016).

El comando ***nmap*** se usa esta vez solamente con la opción ***–sV*** y la opción ***--script*** indicada. Al ejecutarlo el resultado es el siguiente:

Figura. 29. Resultado de NMAP al ejecutar scripts para servidor MySQL



Fuente: El autor. Screenshot de consola de comandos de Windows.

En el recuadro número 1 se muestra el comando ejecutado, a continuación en el cuadro 2 se muestra el resultado del script mysql-brute donde se indica que no se detectaron cuentas validas, ejecutando 50009 intentos en 44 segundos.

Sin embargo, se destaca el resultado del recuadro número 3 donde se indica que el host o equipo desde el cual se intenta el acceso (HENRYJR-14z.local) no está autorizado para conectarse al servidor MySQL. Este mensaje es arrojado cuando el equipo no está asociado a las credenciales suministradas (MySQL, 2016), es decir, hubo conexión pero el equipo desde el cual se realiza no está autorizado, por lo cual para las cuentas ***root*** o ***anonymous*** el password está en blanco. Esto es verdad, debido a que el despliegue se realizó sobre el servidor MySQL como estaba configurado por defecto en Kali Linux el cual no tiene password para el usuario root. En caso que el equipo estuviera autorizado podría entrar al mismo.

Respecto a la base de datos, en la prueba de pentesting de Inyección de SQL (OTG-INPVAL-005) que se documenta en el presente documento se intenta descubrir el nombre de la misma.

11. DEFINICIÓN Y EJECUCIÓN DE PRUEBAS DE PENTESTING UTILIZANDO LA METODOLOGÍA OWASP

11.1 DEFINICIÓN DE PRUEBAS DE PENTESTING A EJECUTAR

Antes de dar inicio a las pruebas de pentesting cabe aclarar que fueron revisados todos aquellos aspectos que permitían "atacar" el script de Quadodo. Es decir, que no se evaluó el mismo funcionalmente ya que las pruebas que se realizaron no son de calidad sino de seguridad. Por otro lado, se usan para todas las pruebas el usuario administrador, puesto que este tiene acceso a todas las páginas.

Una vez aclarado lo anterior, la metodología OWASP lista diferentes test o pruebas que pueden ser ejecutadas. Para el caso de Quadodo se ejecutan y delimitan las siguientes pruebas sobre los módulos indicados las cuales se explican en cada subtema de este documento:

Tabla 4. Listado de pruebas de pentesting ejecutadas sobre Quadodo

Campo de evaluación	Código	Test Name
Identity Management Testing	OTG-IDENT-002	Test user registration process
Identity Management Testing	OTG-IDENT-004	Testing for Account Enumeration and Guessable User Account
Authentication Testing	OTG-AUTHN-001	Testing for Credentials Transported over an Encrypted Channel
Authentication Testing	OTG-AUTHN-003	Testing for Weak lock out mechanism
Authentication Testing	OTG-AUTHN-004 – (1)	Testing for Bypassing Authentication Schema *Direct page request (forced browsing)*
Authentication Testing	OTG-AUTHN-004 – (2)	Testing for Bypassing Authentication Schema *Session ID Prediction*
Authorization Testing	OTG-AUTHZ-001	Testing Directory traversal/file include
Authorization Testing	OTG-AUTHZ-002	Testing for Bypassing Authorization Schema
Authorization Testing	OTG-AUTHZ-003	Testing for Privilege escalation
Authorization Testing	OTG-AUTHZ-004	Testing for Insecure Direct Object References
Client-Side Testing	OTG-CLIENT-001	Testing for DOM-based Cross site scripting
Client-Side Testing	OTG-CLIENT-002	Testing for JavaScript Execution
Client-Side Testing	OTG-CLIENT-003	Testing for HTML Injection
Client-Side Testing	OTG-CLIENT-004	Testing for Client Side URL Redirect
Client-Side Testing	OTG-CLIENT-005	Testing for CSS Injection
Client-Side Testing	OTG-CLIENT-012	Test Local Storage
Input Validation Testing	OTG-INPVAL-001	Testing for Reflected Cross site scripting
Input Validation Testing	OTG-INPVAL-005	Testing for SQL Injection
Session Management Testing	OTG-SESS-001	Testing for Session Management Schema
Session Management Testing	OTG-SESS-002	Testing for cookies attributes

Fuente: El autor. Consolidado de pruebas seleccionadas de la guía de pruebas OWASP.

11.2 IDENTITY MANAGEMENT TESTING

Por medio de estas pruebas se busca evaluar el manejo de identidad en la aplicación, estas pruebas constan de varias fases, sin embargo, se dio prelación a las siguientes:

- ***Test User Registration Process (OTG-IDENT-002):*** Tiene como objetivo validar proceso de registro de usuarios con el fin de verificar que los requerimientos de identidad cumplan con los requerimientos de identidad. (OWASP, 2014)
- ***Testing for Account Enumeration and Guessable User Account (OTG-IDENT-004***): El objetivo de esta prueba es (de ser posible) obtener un listado de usuarios válidos mediante la interacción con el mecanismo de autenticación de la aplicación. Logrando esto llega ser de gran utilidad para un ataque de fuerza bruta en el cual el atacante identifica si un nombre de usuario suministrado es o no válido en la aplicación. (OWASP, 2015)

11.2.1 Test user registration process (OTG-IDENT-002)

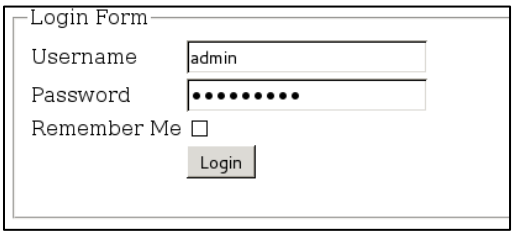
Para esta prueba se empleó el complemento ***LiveHTTPHeaders*** en el navegador ***Iceweasel*** de Kali Linux 2.0. Como parte de la verificación del proceso de registro de usuarios se da respuesta a las siguientes preguntas en los resultados descritos en este documento:

1. ¿Puede la información de identidad ser fácilmente forzada o suplantada?
2. ¿Puede el intercambio de información de identidad ser manipulado durante el registro?

11.2.1.1 Ejecución sobre el módulo de Seguridad

Se procede a ingresar a la aplicación con las credenciales del administrador.

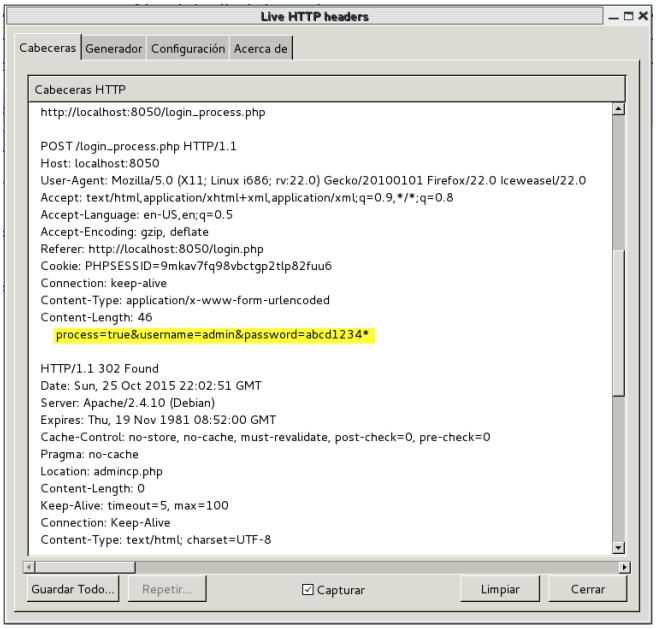
Figura. 30. Formulario de ingreso



Fuente: El autor. Screenshot de pantalla de Quadodo.

Al examinar las cabeceras resultantes se tienen los datos de acceso del usuario administrador.

Figura. 31. Credenciales enviadas en cabecera HTTP



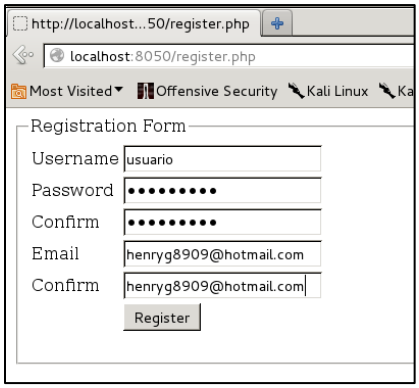
Fuente: El autor. Screenshot de pantalla de Quadodo.

Como se puede observar la página destino es login_process.php a la cual se le envían las variables process, username y password de forma clara y sin encriptación.

11.2.1.2 Ejecución sobre el módulo de Registro

Se ingresa a la página de registro de usuarios disponible en https://localhost:8050/register.php

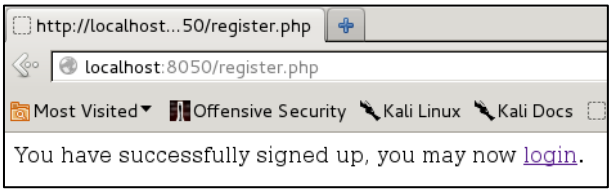
Figura. 32. Página de registro



Fuente: El autor. Screenshot de pantalla de Quadodo.

El sistema confirma registro exitoso.

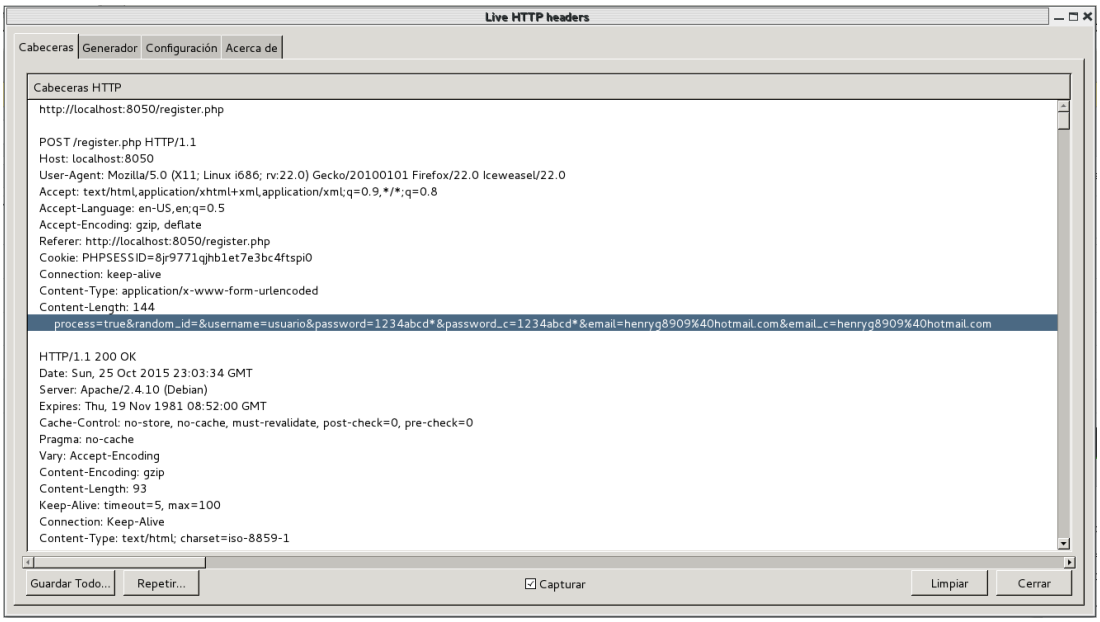
Figura. 33. Confirmación de usuario registrado



Fuente: El autor. Screenshot de pantalla de Quadodo.

Se revisa a continuación el contenido de la cabecera HTTP enviada, como se puede observar la página destino es la misma register.php y se le envían todos los parámetros del formulario a través del método POST.

Figura. 34. Cabecera HTTP enviada en el registro

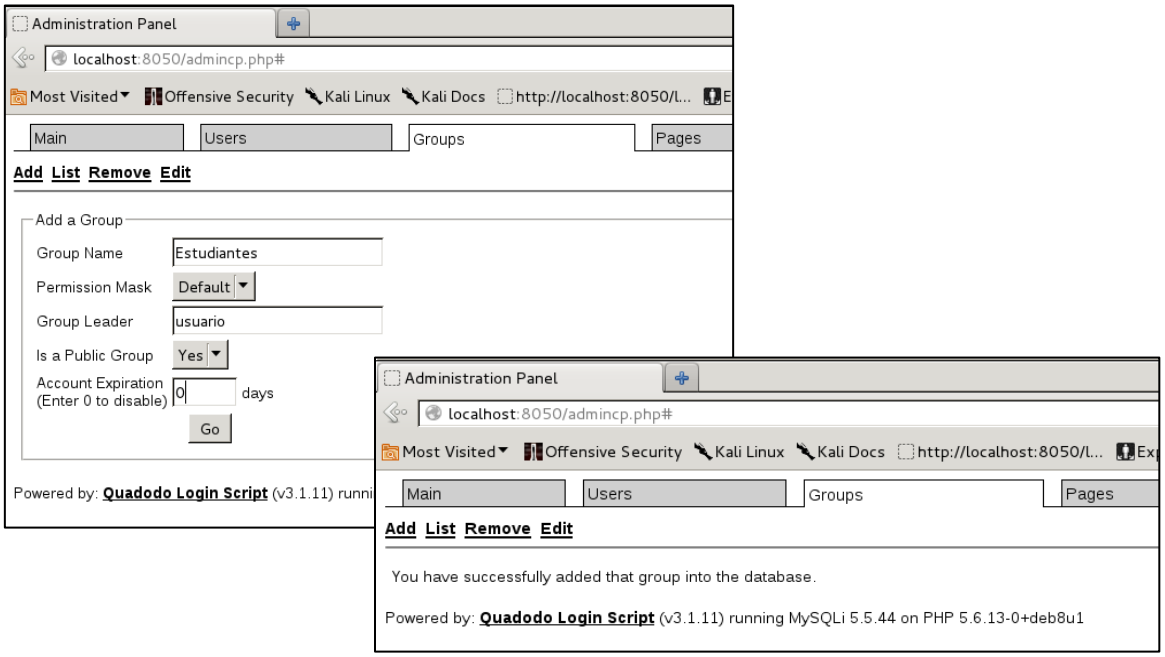


Fuente: El autor. Screenshot de pantalla de Quadodo.

11.2.1.3 Ejecución sobre el Panel de Control de Grupos

Para la prueba de este módulo, se utilizó la función de creación de Grupos a la cual se accede en la pestaña ***Groups*** →opción **Add**. A continuación se muestra la creación del grupo ingresado:

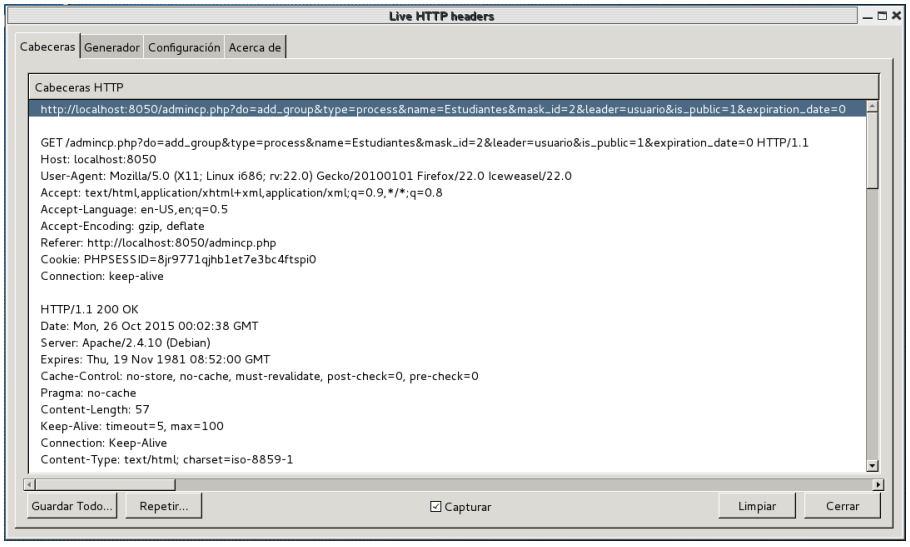
Figura. 35. Formulario de creación de grupos



Fuente: El autor. Screenshot de pantalla de Quadodo.

A continuación se examina la cabecera HTTP enviada, en donde se puede evidenciar que esta vez a través del método GET se envían los parámetros.

Figura. 36. Parámetros de cabecera HTTP en creación de Grupos



Fuente: El autor. Screenshot de pantalla de Quadodo.

11.2.1.4 Resultados obtenidos

Como resultado de la prueba se identifica que la información de los formularios de registro y de ingreso es enviada de forma clara y básica, permitiendo de esta manera ser monitoreada y capturada fácilmente para su manipulación y suplantación.

Al ser la información claramente transmitida sin ningún tipo de encriptación la misma puede ser suplantada o forzada al igual que modificada para enviar datos no deseados. Por otro lado, al identificar claramente las variables utilizadas para el registro de información y los valores enviados, los mismos pueden ser fácilmente modificables o manipulados.

11.2.2 Testing for Account Enumeration and Guessable User Account (OTG-IDENT-004)

Según la guía de pruebas de OWASP, para validar si la aplicación retorna algún tipo de mensaje o código de error revelando de forma directa o indirecta información que pueda ser utilizada para enlistar usuarios de la aplicación, se ejecutan las siguientes pruebas sobre la pantalla de ingreso (Login) del aplicativo.

11.2.2.1 Usuario incorrecto o no existente

Al ingresar a la pantalla de autenticación de Quadodo se escribe el nombre de usuario **usuarionoexiste**, el cual como se observa en la figura a continuación no existe en la tabla tabla ***qls3_users*** de la base de datos:

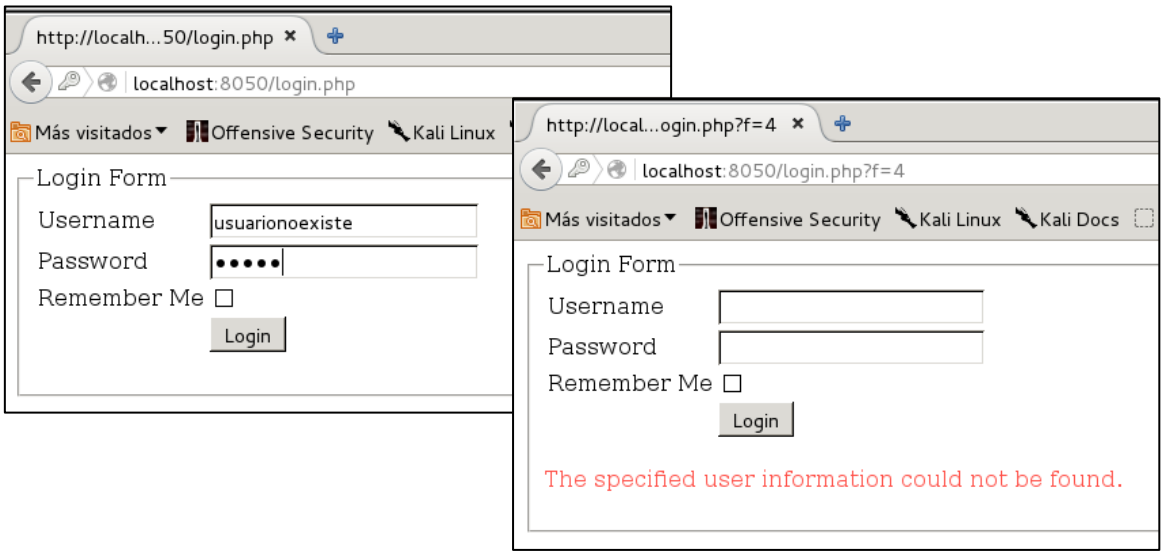
Figura. 37. Lista de usuarios existentes en QuadodoBD

```
mysql> select username from qls3_users;
+----------+
| username |
+----------+
| admin    |
| usertest |
| usuario  |
+----------+
3 rows in set (0.00 sec)
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

Se procede a continuación a ingresar este usuario con el password ***12345*** y el resultado arrojado es el siguiente:

Figura. 38. Mensaje de error indicando usuario no existente



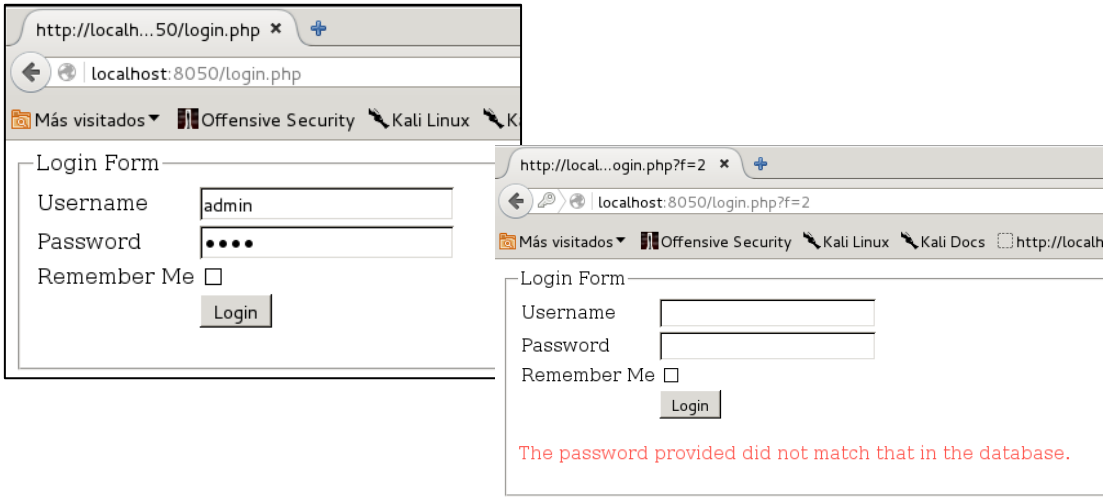
Fuente: El autor. Screenshot de pantalla de Quadodo.

El mensaje indica que efectivamente dicho usuario no se encuentra en la aplicación.

11.2.2.2 Usuario existente y password incorrecto

A continuación se procede a ingresar el nombre de usuario admin, pero con un password erróneo para validar el mensaje de error que genera la aplicación para este escenario.

Figura. 39. Mensaje de password incorrecto



Fuente: El autor. Screenshot de pantalla de Quadodo.

El mensaje retornado indica que el password es incorrecto para el usuario ingresado, por lo cual nos da a entender que el usuario sí existe en la aplicación.

11.2.2.3 Resultado obtenido

Como resultado de las pruebas realizadas se puede evidenciar que según los mensajes obtenidos para cada escenario, los mismos evidencian que la aplicación revela si el usuario ingresado existe o no en el sistema permitiendo al atacante identificar los usuarios a utilizar para realizar ataques que le permitan acceder a la aplicación.

11.3 AUTHENTICATION TESTING

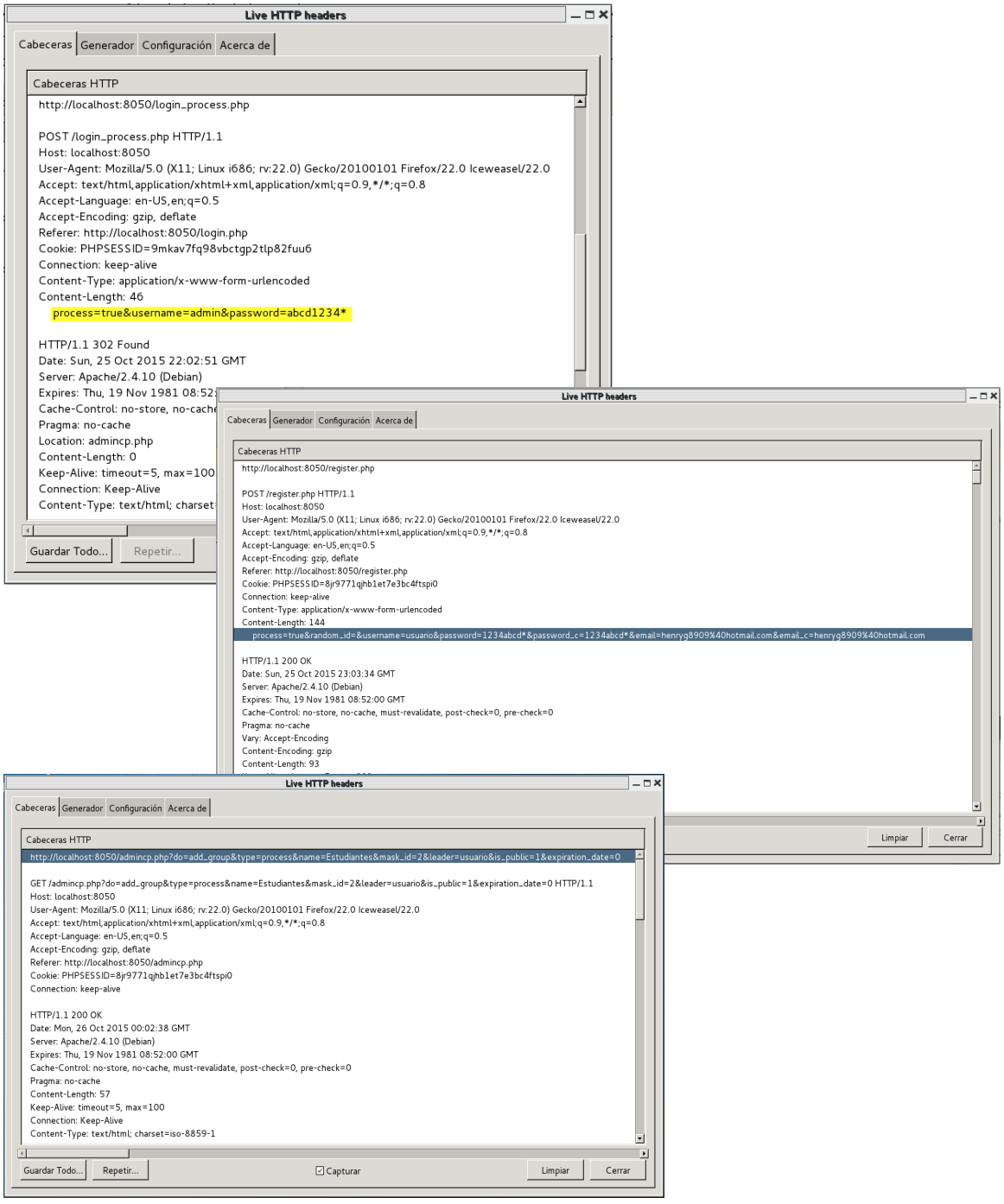
En seguridad computacional, la autenticación es el proceso de intentar verificar la identidad digital del que envía una comunicación, como por ejemplo, el ingreso a una aplicación. Al realizar pruebas sobre este proceso, se busca entender cómo funciona el esquema implementado y de esta manera burlar el mecanismo de autenticación. Para ello, se hará el enfoque en las siguientes pruebas:

- ***Testing for Credentials Transported over an Encrypted Channel (OTG-AUTHN-001)***: Busca verificar que los datos del usuario son transferidos a través de un canal encriptado para de esta manera descartar que puedan ser interceptados por un usuario malicioso. Esta prueba se centra en entender si la aplicación web toma las medidas de seguridad necesarias usando un protocolo como HTTPS.
- ***Testing for Weak lock out mechanism (OTG-AUTHN-003):*** Evalúa que los mecanismos de bloqueo tengan la habilidad de mitigar ataques de fuerza bruta para encontrar la clave de acceso, además de la resistencia a desbloqueo de cuentas no autorizados.
- ***Testing for Bypassing Authentication Schema (OTG-AUTHN-004):*** El objetivo de estas pruebas es identificar esquemas de autenticación que pueden ser pasados por alto invocando directamente una página interna que se supone requiere autenticación, esta prueba puede realizarse a través de diferentes métodos, para este caso se aplican los siguientes:
    - ***Direct page request (forced browsing):*** Esta prueba de acceso se realiza invocando directamente una página, que se supone para su ingreso primero la autenticación y permisos del usuario.
    - ***Session ID Prediction:*** El objetivo es identificar si la generación del identificador de la sesión es predictiva por medio del cual un usuario puede acceder sin necesidad de realizar una autenticación.

11.3.1 Testing for Credentials Transported over an Encrypted Channel (OTG-AUTHN-001):

Esta prueba tiene relación con el resultado obtenido en las pruebas que se realizaron en la ejecución del *Test user registration process (OTG-IDENT-002)* y que se evidencian en el presente documento. Tal y como se puede observar al autenticarse en la aplicación, al crear usuarios y al crear un grupo.

Figura. 40. Resultados obtenidos al autenticarse en la aplicación, al crear usuarios y al crear un grupo.



Fuente: El autor. Screenshots de pantallas de Quadodo.

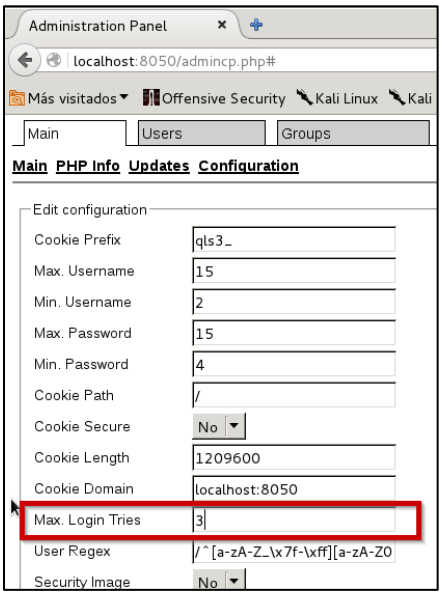
11.3.1.1 Resultado Obtenido

Como se puede evidenciar en los resultados obtenidos al interceptar las cabeceras enviadas en cada petición, las credenciales viajan en texto plano sin ningún tipo de encripción previa y por protocolo HTTP. Claro que esto se debe también a que a la configuración del sitio trabaja sobre este protocolo y no por HTTPS, sin embargo los datos pueden visualizarse a través del método GET y ser manipulados fácilmente para alterar cualquier información.

11.3.2 Testing for Weak lock out mechanism (OTG-AUTHN-003):

Quadodo permite a través del módulo de configuración establecer la cantidad de intentos de acceso que puede realizar un usuario previo al bloqueo del mismo. Se procede entonces a establecer el número de intentos a tres y a probar que este parámetro se cumpla ingresando más de 3 veces:

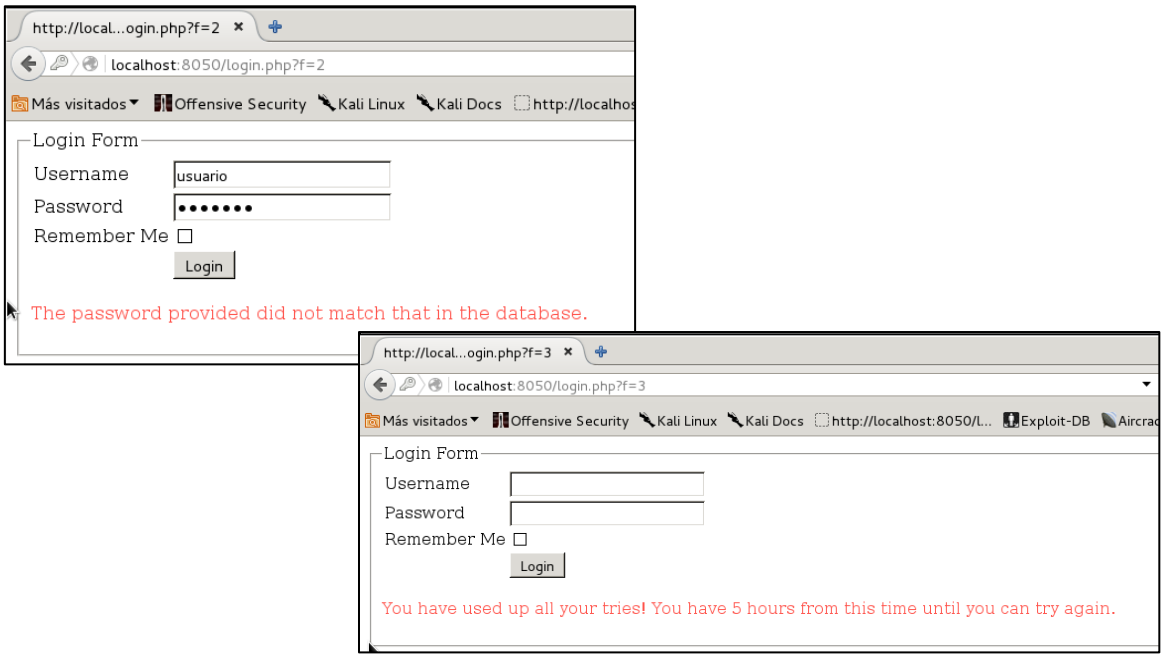
Figura. 41. Configuración de número máximo de intentos



Fuente: El autor. Screenshot de pantalla de Quadodo.

La prueba se realiza con el usuario "**usuario**" y el password incorrecto "**zzzzzzz**". En el primer intento indica que el password es incorrecto y en el tercer intento el sistema indica que el usuario ha excedido el número de intentos para acceder y que tiene que esperar 5 horas para intentar nuevamente

Figura. 42. Mensaje de intentos de acceso excedidos por el usuario



Fuente: El autor. Screenshot de pantalla de Quadodo.

El sistema indica entonces que el usuario ha excedido el número de intentos para acceder y que tiene que esperar 5 horas para intentar nuevamente.

11.3.2.1 Resultado

El número de intentos depende de la configuración establecida en el aplicativo por parte del administrador del sistema. Sin embargo, al tener esta funcionalidad, el script puede adaptar la limitación de intentos de accesos al sistema según la política que crea conveniente cada organización y administrador, dependerá que la primera esté fuertemente establecida estipulando el parámetro más conveniente.

11.3.3 Testing for Bypassing Authentication Schema (OTG-AUTHN-004):

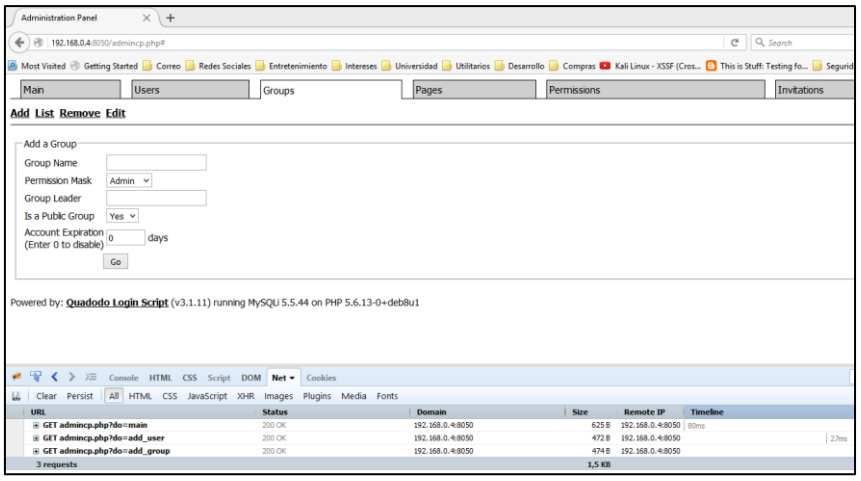
11.3.3.1 Direct page request (forced browsing)

Para esto se identifican tres páginas que corresponden a la página principal de la administración del sitio, a la página de registro y a la página del panel de control de grupos. Para este caso se realiza desde un browser y cliente fuera del servidor donde está desplegada la aplicación con Sistema Operativo Windows 10 y Versión de Firefox 50.0:

- http://192.168.0.4:8050/admincp.php?do=main
- http://192.168.0.4:8050/admincp.php?do=add_user
- http://192.168.0.4:8050/admincp.php?do=add_group

Estas dos últimas fueron identificadas mediante el complemento Firebug de Firefox el cual es una herramienta de desarrollo que permite entre otras cosas monitorear las transacciones sobre la red mientras se navega por el sitio web.

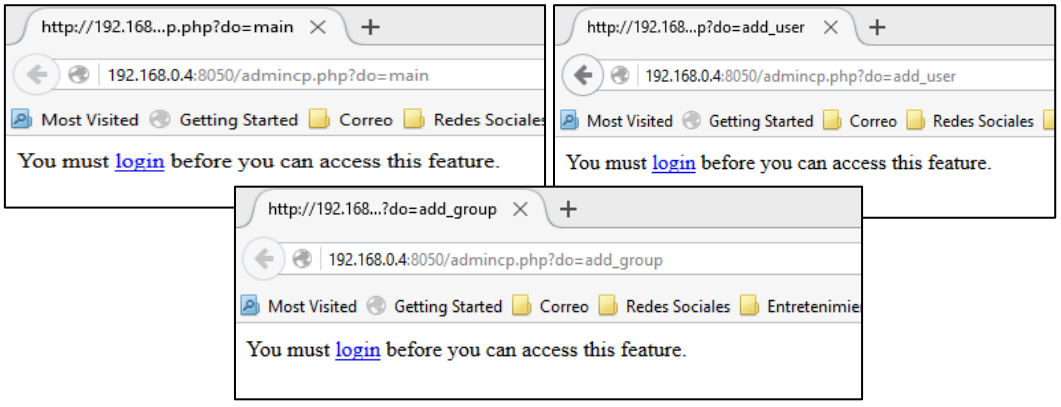
Figura. 43. Listado de URL's a través de Firebug



Fuente: El autor. Screenshot de pantalla de Quadodo y de Firebug en navegador Firefox.

Se procede a acceder a las páginas indicadas directamente colocándolas directamente en el campo de URL del navegador sin haberse autenticado previamente.

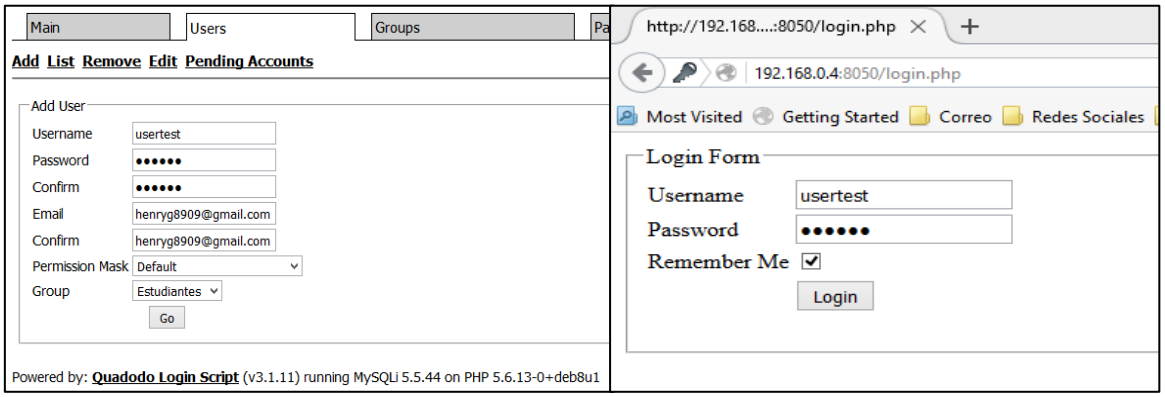
Figura. 44. Mensaje de Intento de acceso a las páginas de administración, creación de usuarios y de grupos



Fuente: El autor. Screenshot de pantalla de Quadodo.

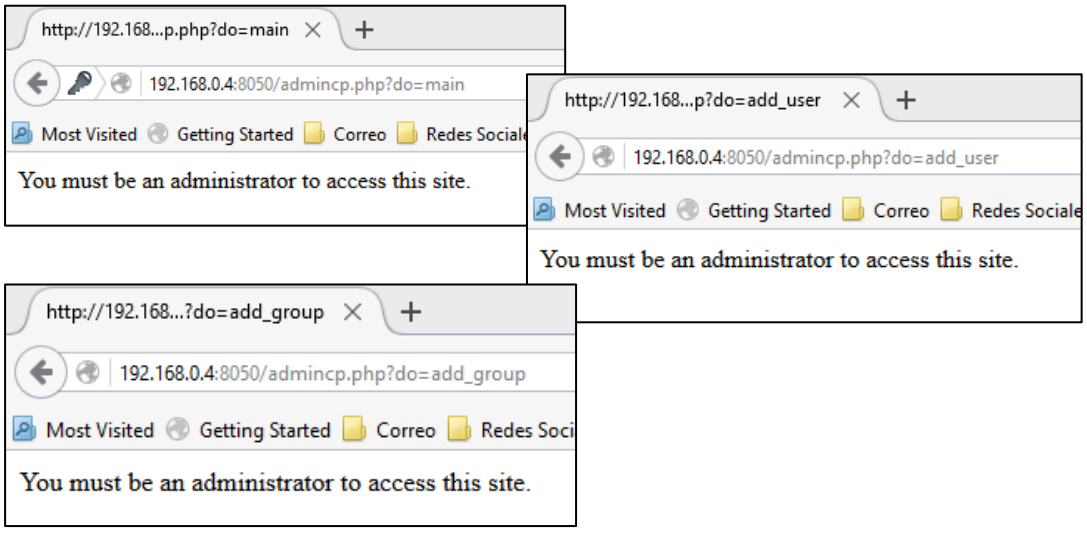
Por otro lado el acceso a esta páginas de administración solamente pueden ser accedidas por el usuario administrador o aquel que tenga dicho rol o permiso. Es por esto que se ingresa con un usuario sin privilegios y se intenta acceder a las páginas indicadas.

Figura. 45. Creación e ingreso de usuario sin privilegios



Fuente: El autor. Screenshots de pantallas de Quadodo.

Figura. 46. Intento de acceso a administración, creación de usuarios y de grupos con usuario sin privilegios



Fuente: El autor. Screenshots de pantallas de Quadodo.

- **Resultado**

Mediante los resultados se ha podido comprobar que al intentar acceder sin autenticación o con un usuario sin privilegios, el sistema evita el ingreso validando cada escenario, lo cual asegura que el mismo no acceda a opciones no autorizadas.

11.3.3.2 Session ID Prediction

Para esta prueba se usa la herramienta Burp Suite, la cual se encuentra disponible en Kali Linux y a modo de proxy, permite interceptar las peticiones realizadas desde un navegador web a través de la red. A través de la opción Sequencer que se encuentra integrada en esta herramienta es posible analizar la asignación de identificadores de sesión aleatorios (INFOSEC Institute, 2014).

A continuación la ejecución de esta prueba:

- **Configuración**

Se abre una consola en Kali Linux y se digita la instrucción **burpsuite.jar**, no sin antes autenticarse como ***root*** en el aplicativo a través de la instrucción ***su***:

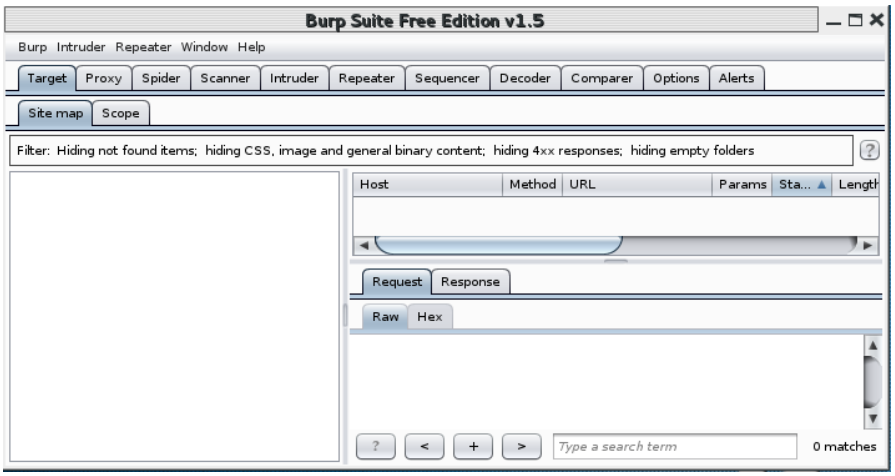
Figura. 47. Instrucción para ejecutar la herramienta Burp Suite

```
henrygarzon@6garzon:~$ su
Contraseña:
root@6garzon:/home/henrygarzon# burpsuite.jar
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

Una vez se ejecuta se abre una interfaz gráfica con algunas opciones que se configuran de la siguiente manera:

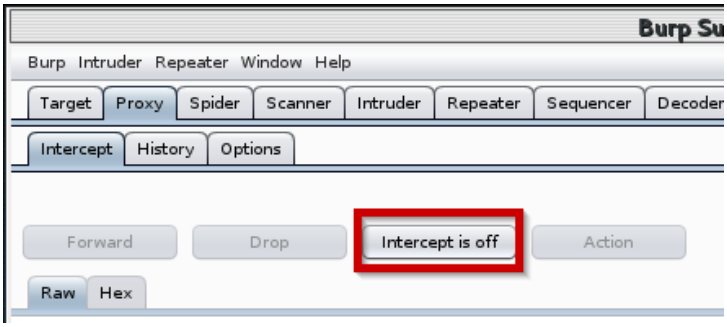
Figura. 48. Interfaz de usuario de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

- En la opción proxy, se le indica que NO se interceptará cada petición que se haga puesto que el objetivo de la pruebas es ser observador del comportamiento del identificador de sesión asignado, para ello se desactiva la opción **Intercept** dejándola en estado **off**:

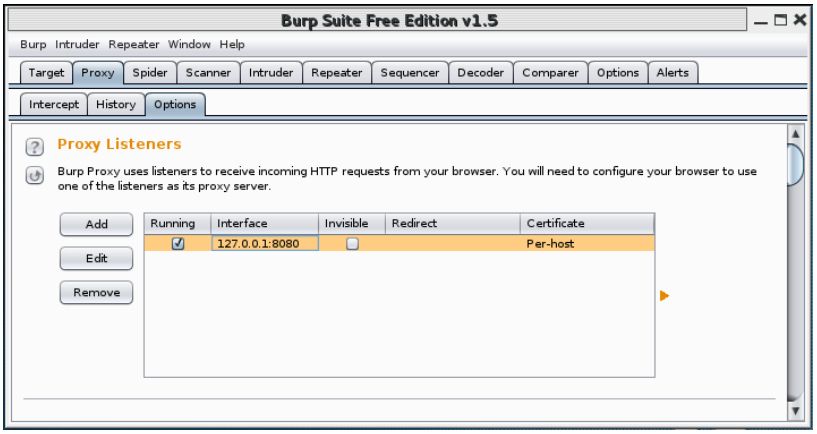
Figura. 49. Opción Intercept de BurpSuite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

- A continuación bajo la pestaña Options que se encuentra a su vez bajo la pestaña Proxy, se identifica la configuración de proxy que se deben establecer en el navegador para que las peticiones puedan ser detectadas.

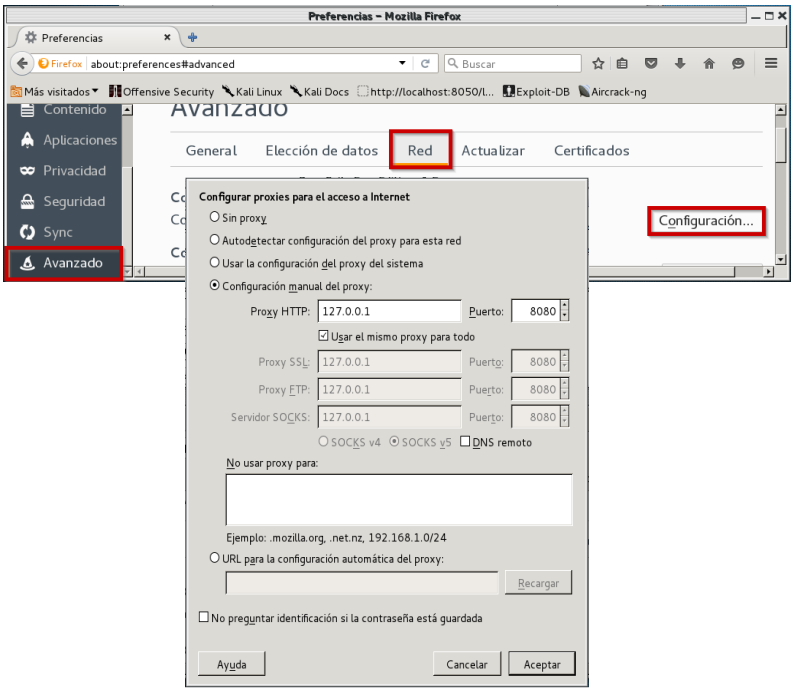
Figura. 50. Configuración de Proxy para Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

- Una vez hecho esto en el navegador se configura esta dirección y puerto. Para este caso el navegador usado es Firefox, y a través de la opción de ***Preferencias → Avanzado → Red*** se le suministran estos parámetros:

Figura. 51. Configuración de Proxy en Firefox

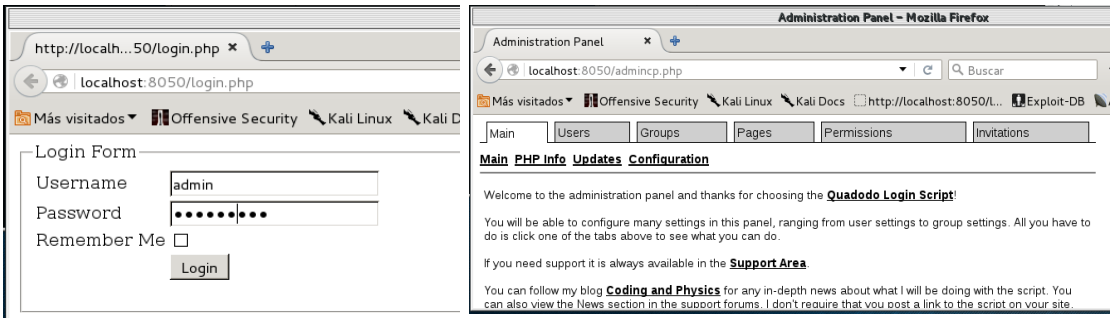


Fuente: El autor. Screenshot de pantalla de Firefox.

- **Ejecución de la prueba**

Ya configurados los parámetros se procede a acceder a Quadodo a través del navegador. Se ingresan las credenciales de administrador para ingresar.

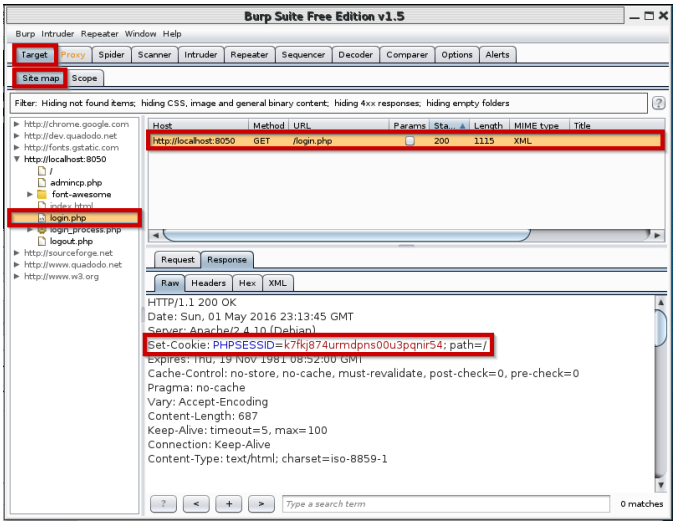
Figura. 52. Acceso a Quadodo como administrador usando Burp Suite



Fuente: El autor. Screenshot de pantalla de Quadodo.

Revisando la interfaz de ***Burp Suite***, bajo las pestañas ***Target → Site map → Request → Raw*** se puede evidenciar que para la página ***login.php*** se generó una cookie con un identificador de sesión:

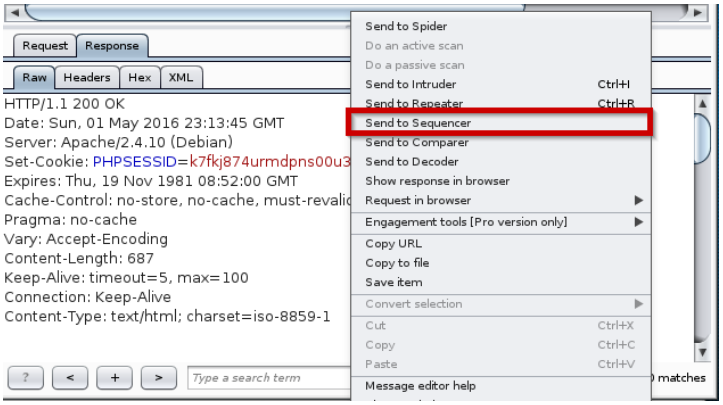
Figura. 53. Identificador de sesión indentificado en Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite

Ahora para generar varias peticiones y revisar el comportamiento se hace clic derecho sobre el panel de la pestaña **Raw** donde se muestra la información de la petición enviada y se selecciona la opción **Send to Sequencer**.

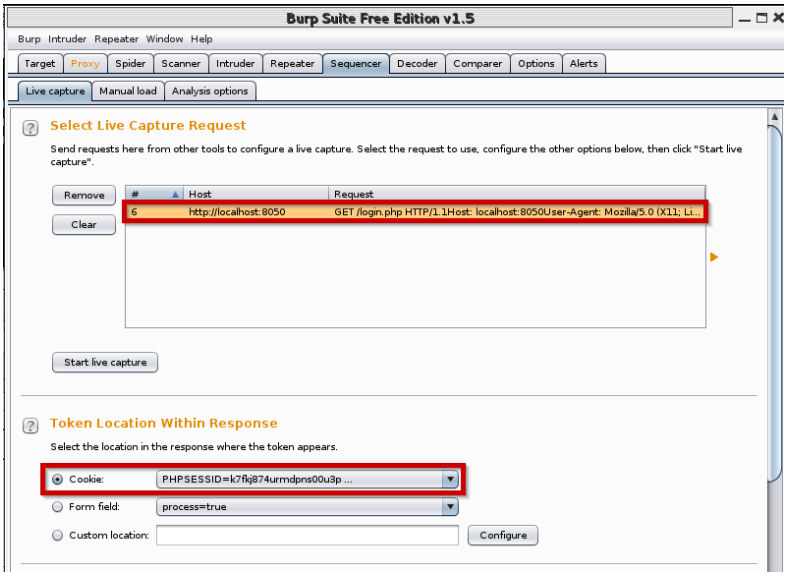
Figura. 54. Opción Send to Sequencer de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

Una vez hecho esto bajo la pestaña ***Sequencer → Live Capture*** se muestra la petición seleccionada agregada al listado de capturas. Ya que la prueba se concentra en revisar los valores asignados al identificador de sesión se selecciona bajo la opción "***Token Location Within Response***" la cookie mostrada en la petición.

Figura. 55. Configuración de Sequencer para petición en Burp Suite

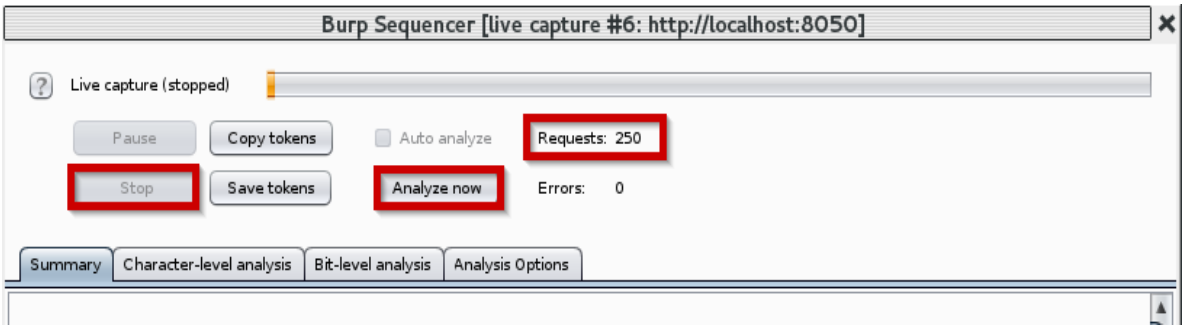


Fuente: El autor. Screenshot de pantalla de Burpsuite.

Se ejecutan las peticiones a través del botón ***Start live capture*** que se encuentra en la parte inferior del listado de peticiones agregadas. Se mostrará a continuación

la ventana de captura, en la cual una vez sean ejecutadas la cantidad de peticiones deseadas, se hace clic en ***Stop*** y se procede a realizar el resultado a través del botón ***Analyze now***.

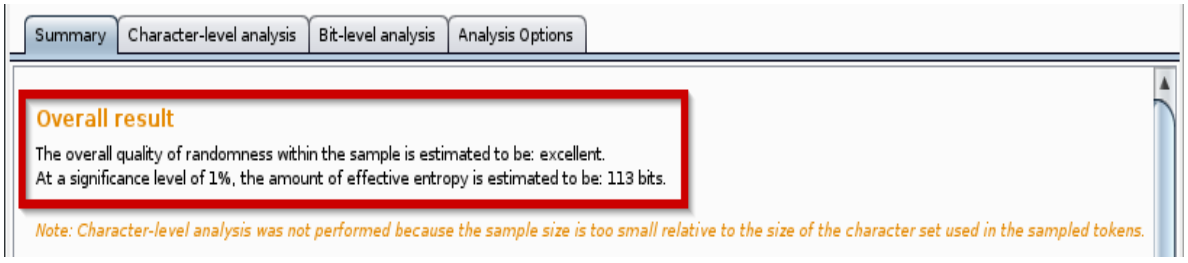
Figura. 56. Opciones de captura de Sequencer de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

Para esta prueba, se enviaron un total de 250 peticiones (como se muestra en la imagen anterior) y el resultado arrojado por la herramienta indica que la calidad de la asignación aleatoria para el identificador de sesión es ***excelente***.

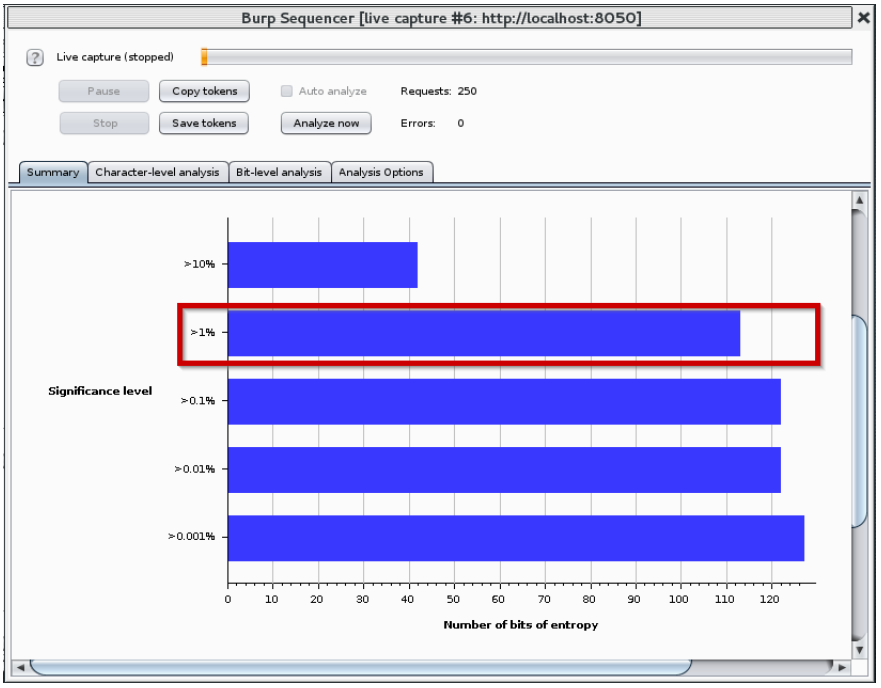
Figura. 57. Resultado de Sequencer de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

La herramienta también arroja un gráfico de análisis de entropía (es decir, el nivel de probabilidad que un mismo valor sea generado), según el resultado señalado en la imagen anterior, la cantidad de entropía efectiva se estima que sea aplicada a cada 113 bits correspondiente a un nivel de significancia del 1%.

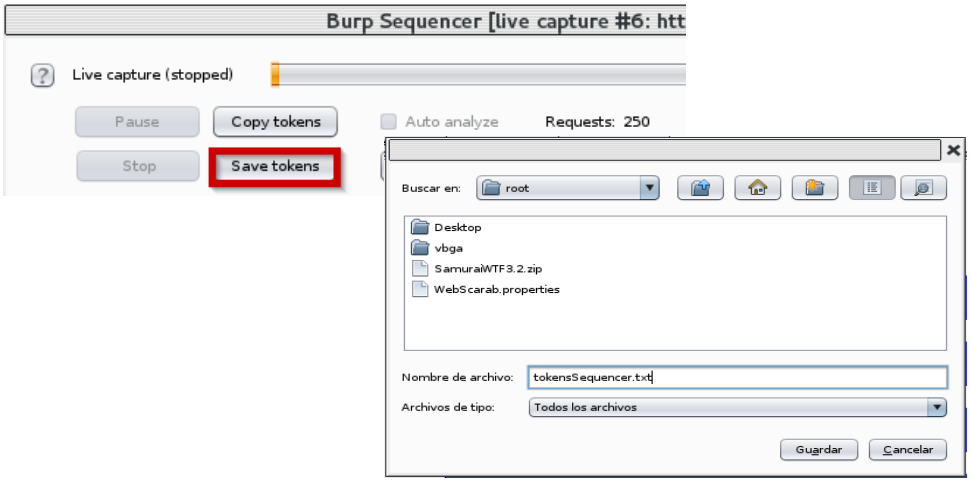
Figura. 58. Gráfico de análisis de entropía de Sequencer de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

Finalmente se revisan los identificadores de sesión generados en la prueba haciendo clic en el botón **Save Tokens**, y listando los mismos a través de la instrucción ***cat*** en la terminal de Kali Linux haciendo uso de la ruta seleccionada para guardar los mismos.

Figura. 59. Opción Save Tokens de la opción Sequencer de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

Figura. 60. Identificadores de Sesión generados

```
root@6garzon:~# cat tokenSequencer.txt
0rjn3ov9dlh89m400ptt6q0gl4
4rsphauga8dcu48tlpeuvtobl0
dq7kjr793ss55ssr4d9cccftn4
t1sav7g3u9k1lnu27afom0ubq4
eav3ul6hhgo197ap0p1lnnp626
i4c0g2rtt1vk055ed44qfljh44
jokcsq7rffuu6tefr02p6jgho4
4hjgm0hld9m7r6sjr06npfhc60
r634smu3pj3f2nb2b7vk4asf14
tfp4chnph4b3dl974tq4bs7ji0
nk7fgj2m3qsatfigev4ah46541
ek6739k5t5rqc5ldu21dtqsg67
9sfuclmm1hslm78quk5a5jr8m1
17lvqce3hitbiodf82f114sr21
jb91b15cgqptie1k49v781jlv5
e0kd3jbfgc38rhopnkbijamjt6
m275rglf76i6s92kb969n93rk1
aq8bo7ag8bti4q1mpftkvqfkv5
v10ee8d6s1jfu7behtctt9j1j7
3h555gl2fls4puhq4lb5gr8943 [...]
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

- **Resultado**

Como resultado de la prueba realiza se puede concluir que el listado anterior no se identifica un patrón de generación o por lo menos no una secuencia específica de generación de identificador de sesión, por lo cual se puede asegurar, también según el resultado arrojado en la herramienta ***Burp Suite*** que Quadodo no presenta un riesgo de seguridad informática en la generación del identificador de sesión que usa en la aplicación.

11.4 CLIENT-SIDE TESTING

Hace referencia a la ejecución de código del lado del cliente, normalmente a través de un navegador web o un plugin de navegador. A través de la ejecución de las siguientes pruebas se hace referencia al DOM (Modelo de Objetos del Documento por sus siglas en inglés), es decir el formato de estructura usado para representar los documentos en un navegador.

Las pruebas que se realizan sobre Quadodo son las indicadas a continuación:

- ***Testing for DOM-based Cross site scripting (OTG-CLIENT-001):*** el cross-site scripting basado en el DOM es el nombre que se le da a los bugs de XSS (Cross-Site Scripting) que son el resultado de contenido activo del lado del navegador en una página web determinada, normalmente Javascript, por medio de los

cuales es posible obtener información introducida por el usuario y posteriormente ejecutar contenido inseguro a través de código inyectado.

- ***Testing for JavaScript Execution (OTG-CLIENT-002):*** Una vulnerabilidad de inyección de Javascript es un subtipo de Cross-Site Scripting (XSS) que involucre la habilidad de inyectar de forma arbitraria código Javascript que a su vez es ejecutado por la aplicación en el navegador web de la víctima. Las consecuencias pueden ser diversas, como el acceso a cookies de la sesión del usuario que puedan modificarse, realizando de esta manera una personalización de la víctima, accediendo a información privilegiada o restringida.
- ***Testing for HTML Injection (OTG-CLIENT-003):*** La inyección de HTML, es un tipo de inyección que sucede sobre los usuarios de aplicación que tienen permisos para ingresar información, y dicha información contienen código HTML. Si la inyección se realiza con éxito es posible permitir al atacante modificar el contenido que visualiza la víctima en la página.
- ***Testing for Client Side URL Redirect (OTG-CLIENT-004):*** La redirección abierta, es un tipo de vulnerabilidad que es usada para ejecutar un ataque de phising o de su plantación de identidad o un ataque de redirección a páginas maliciosas.
- ***Testing for CSS Injection (OTG-CLIENT-005):*** Consiste en la inyección de código CSS (Hojas de estilo en cascada por sus siglas en inglés) realizando la extracción de datos sensibles desde la interfaz web mostrada al usuario a través del navegador web.
- ***Test Local Storage (OTG-CLIENT-012):*** También conocido como Web Storage (Almacenamiento Web) u Offline Storage (Almacenamiento desconectado), se refiere a un mecanismo para almacenar datos en una estructura de llave-valor asociada un dominio y forzada por la misma política de origen. El objetivo es identificar si datos sensibles están siendo almacenados y que luego puedan ser accesados, por ejemplo, a través de Javascript.

11.4.1 Testing for DOM-based Cross site scripting (OTG-CLIENT-001)

Para esta prueba se usa la plataforma VEGA incluida en Kali Linux, la cual es libre y de código abierto (open source) que se usa como escáner y herramienta de pruebas para validar la seguridad de aplicaciones web. Es muy útil, ya que puede encontrar y validar vulnerabilidades de tipo SQL Injection, Cross-Site Scripting (XSS).

Vega está desarrollada en Java y puede ser utilizada en plataformas Linux, OS X y Windows, además puede ser extendida usando una poderosa API hecha en Javascript (Kali Tools, 2014).

11.4.1.1 Configuración

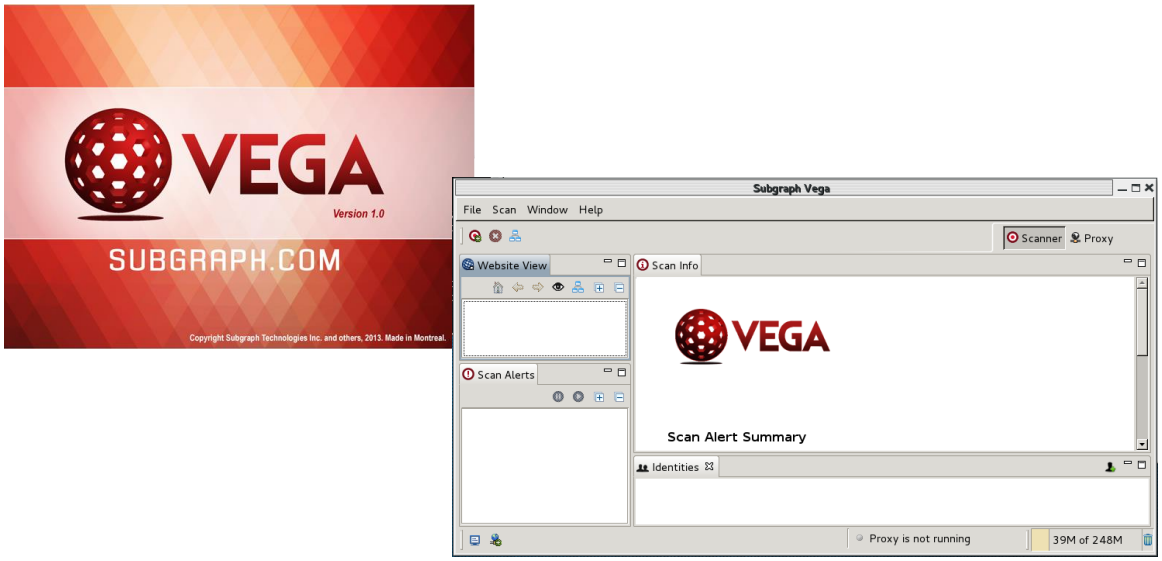
Para iniciar la prueba se procede a iniciar Vega con la siguiente instrucción en una terminal Kali Linux, con permisos de root:

Figura. 61. Ejecución de la plataforma Vega.

```
henrygarzon@6garzon:~$ su
Contraseña:
root@6garzon:/home/henrygarzon# vega
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

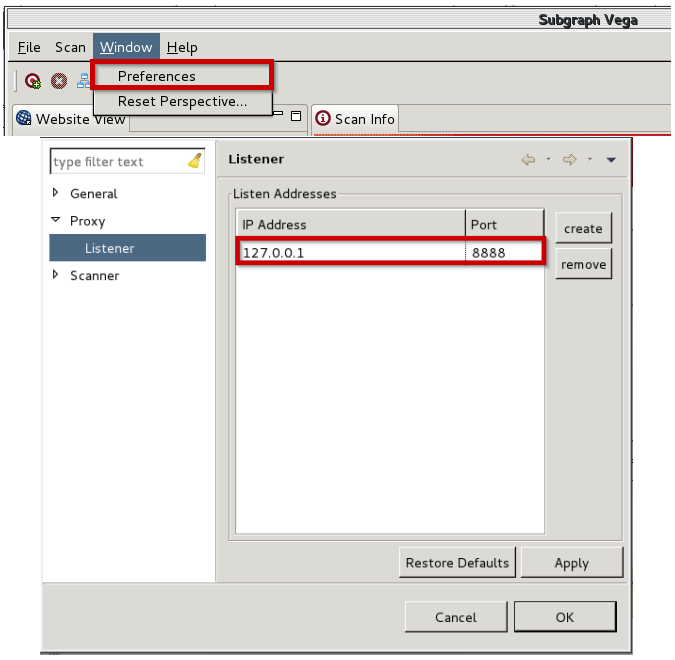
Figura. 62. Imagen y Pantalla de inicio de la plataforma Vega



Fuente: El autor. Screenshot de pantalla de Vega.

Para ejecutar el escaneo de vulnerabilidades, se configura Vega como un proxy accediendo al ***Menú Window → Preferences*** y bajo la opción ***Proxy → Listener*** se indica la IP y puerto. La configuración por defecto para estos parámetros son la IP local ***127.0.0.1*** y el puerto ***8888***, se dejan estos valores.

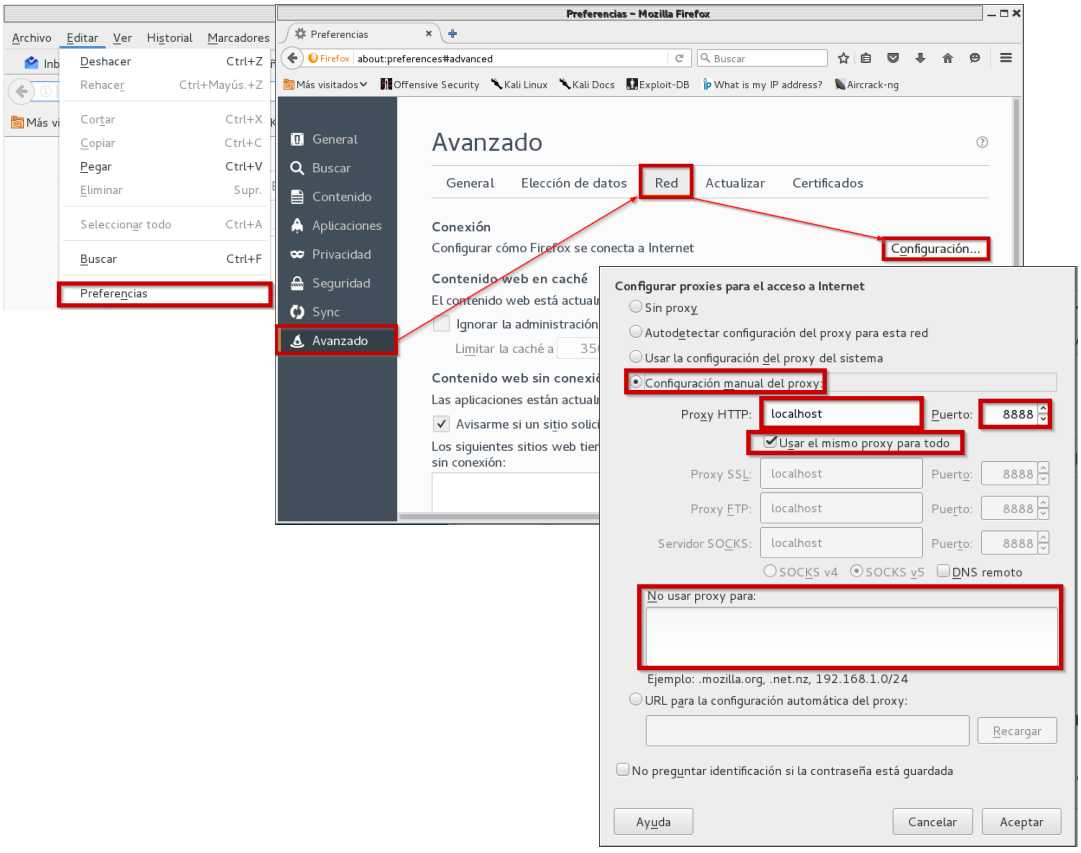
Figura. 63. Opción Preferences y configuración de Proxy para la plataforma Vega



Fuente: El autor. Screenshot de pantalla de Vega.

Una vez realizado lo anterior se configura el navegador para que use la IP y puerto indicados como proxy de conexiones con esto es posible escanear las peticiones web que son realizadas. Para este caso ya que se usa Firefox como navegador, se accede al menú ***Editar → Preferencias*** y bajo las opciones ***Avanzado → Red → Cofiguración…*** se ingresa la IP y puerto, teniendo en cuenta no dejar en el campo "No usar proxy para" como excepción el equipo donde se probará se encuentra desplegado el sitio, es decir, el equipo local (localhost ó 127.0.0.1).

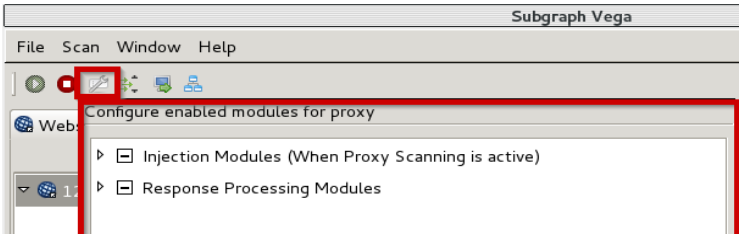
Figura. 64. Acceso a configuración de Proxy en Firefox



Fuente: El autor. Screenshot de pantalla de Firefox.

Antes de iniciar el proxy, se verifican los módulos activados en Vega y que ejecutará mientras se realizan las peticiones, para ello se hace clic en el botón indicado en la imagen siguiente, donde también se muestra el panel que se despliega listando los tipos de módulos disponibles.

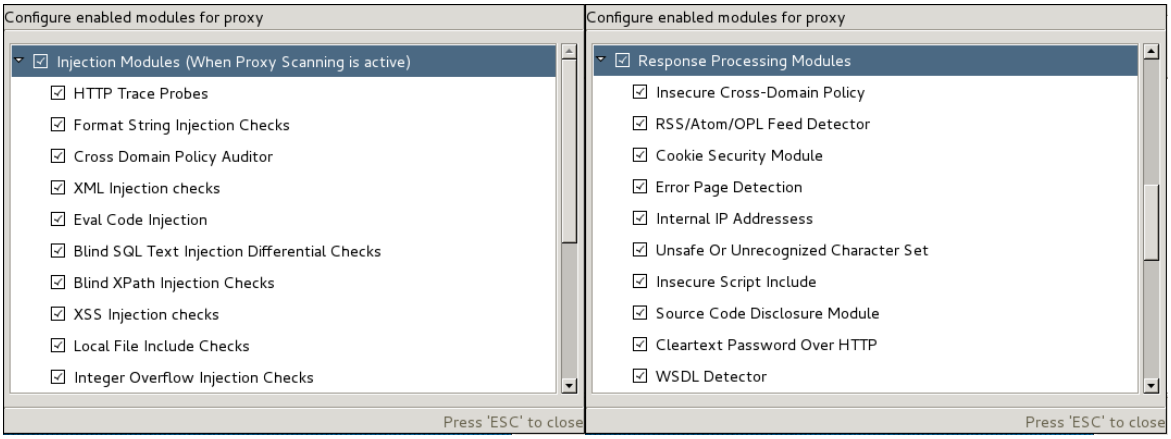
Figura. 65. Listado de módulos para el proxy de la plataforma Vega



Fuente: El autor. Screenshot de pantalla de Vega.

Al desplegar cada una de las listas de tipos de módulos se puede observar que algunos están activos por defecto, los cuales incluyen por ejemplo verificación de inyección de XSS, inclusión de archivos locales, auditoría de política de Cruce de Dominios (Cross Domain), evaluación de inyección de código, entre otros. Se procede entonces a dejar todas las opciones activas.

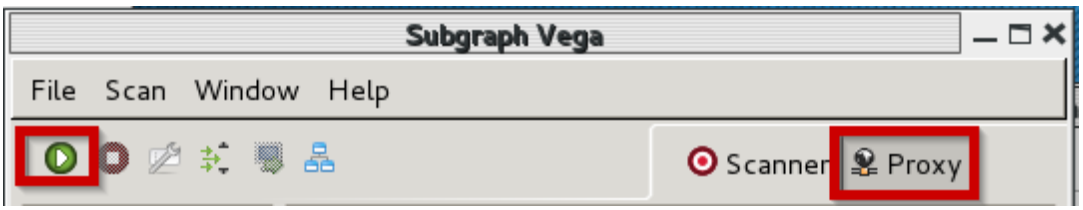
Figura. 66. Módulos activados para proxy de la plataforma Vega



Fuente: El autor. Screenshot de pantalla de Vega.

Se ejecuta a continuación el proxy para que empiece a escanear las peticiones realizadas al sitio. Para ello en la ventana de Vega se hace clic en el siguiente botón y se visualizan las peticiones en la opción Proxy señalada también en la imagen siguiente:

Figura. 67. Opción de inicio de proxy y opción Proxy de visualización de peticiones

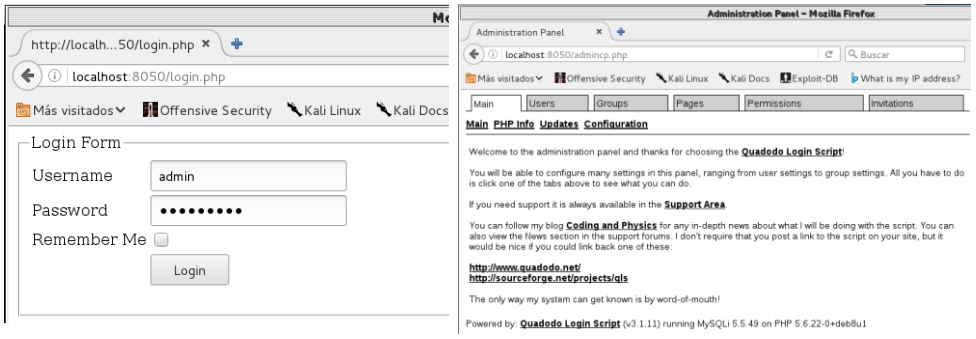


Fuente: El autor. Screenshot de pantalla de Vega.

Se realizan a continuación las siguientes acciones como se evidencia en las imágenes siguientes:

a. Acceso del usuario administrador (admin)

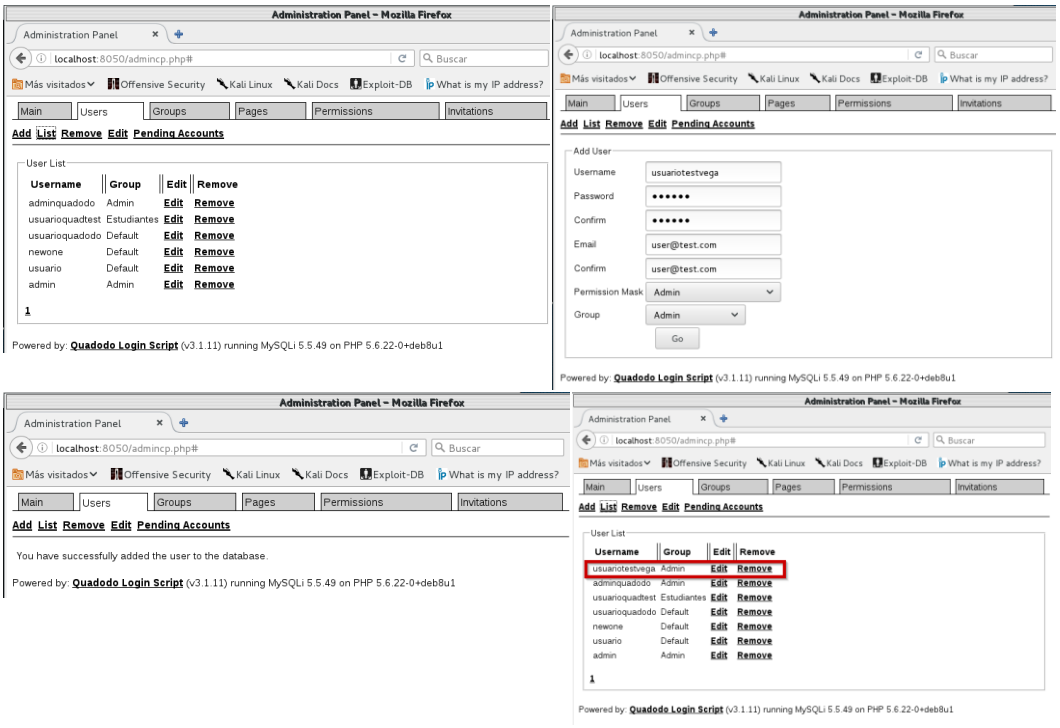
Figura. 68. Acceso de usuario Admin - en ejecución plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

b. Listado de usuarios y creación de usuario administrador

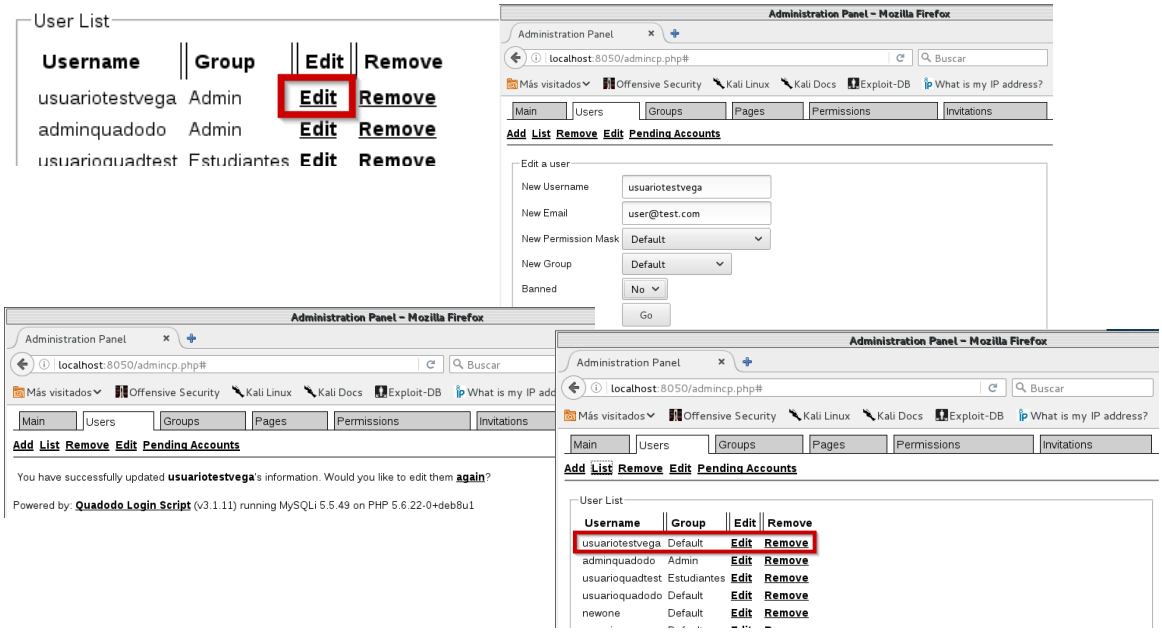
Figura. 69. Lista y creación de usuario administrador - ejecutando plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

c. Edición de usuario creado

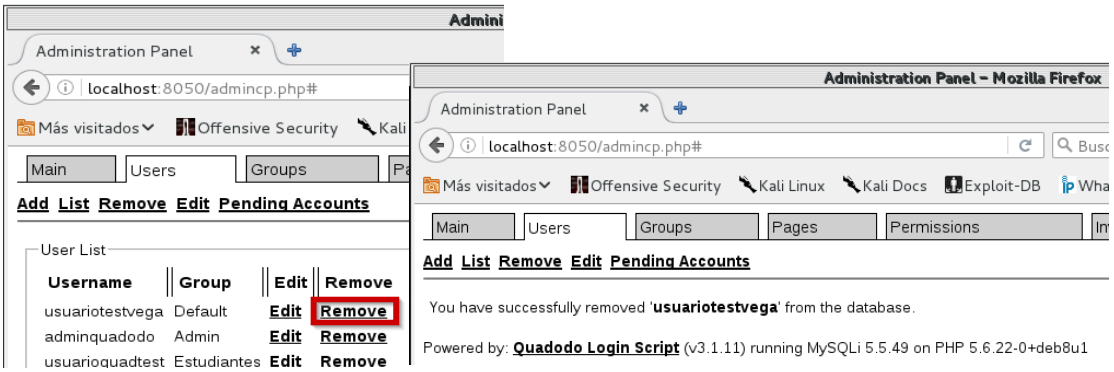
Figura. 70. Edición de usuario creado - ejecutando plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

d. Eliminación de usuario creado

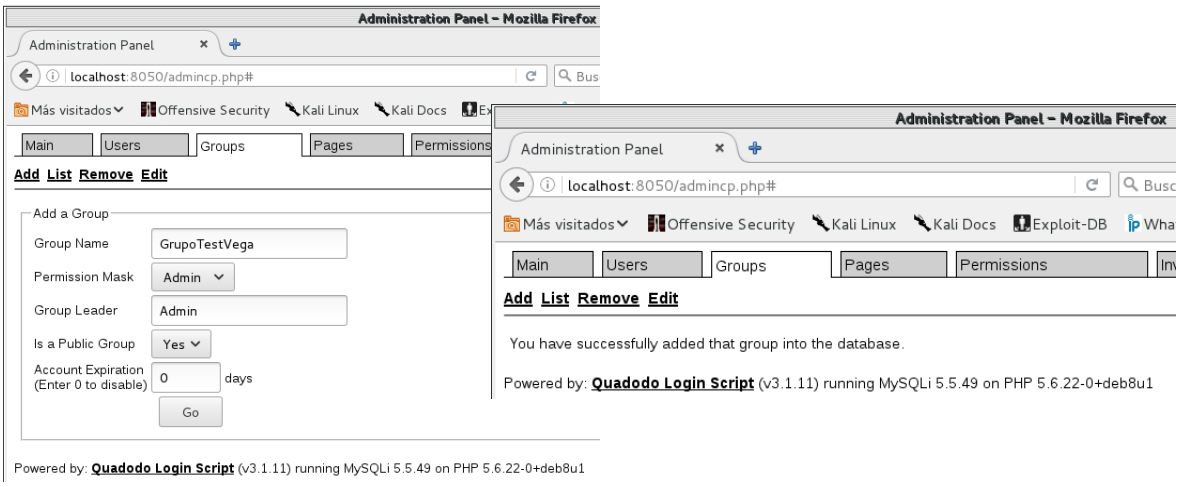
Figura. 71. Eliminación de usuario creado - en ejecución plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

e. Creación de grupo administrador

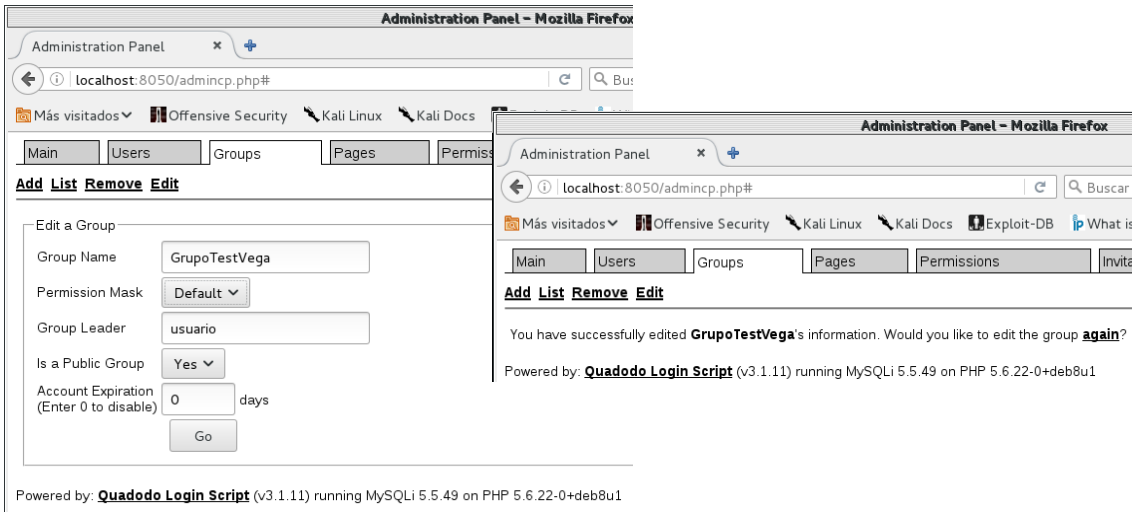
Figura. 72. Creación de grupo Administrador - en ejecución plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

f. Edición de grupo creado

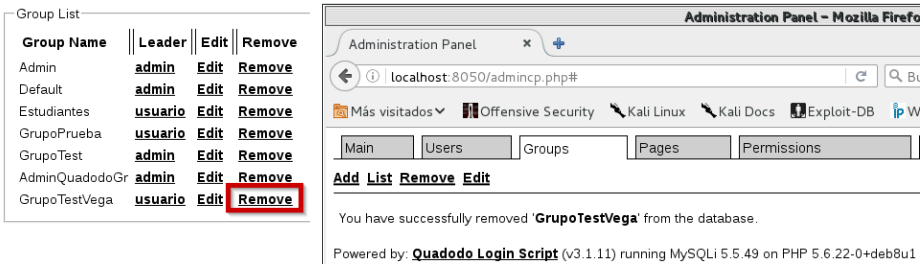
Figura. 73. Editar grupo creado - en ejecución plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

g. Eliminar grupo creado

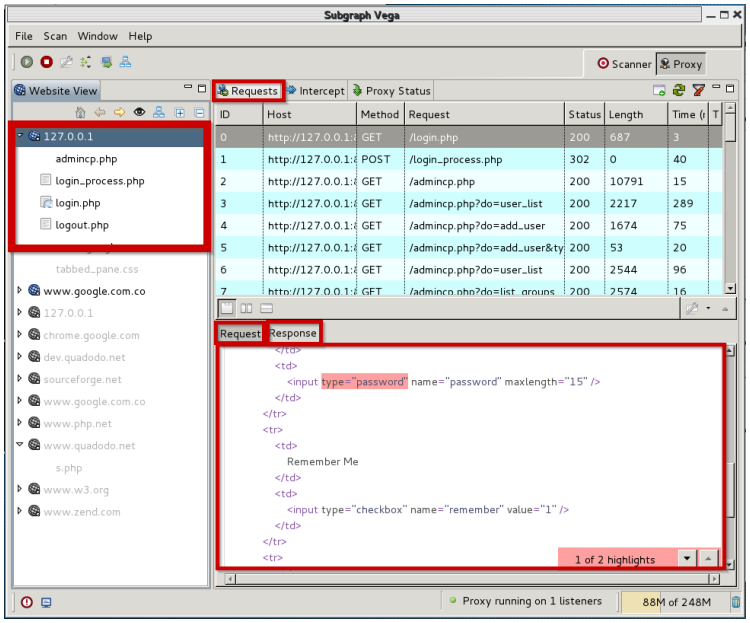
Figura. 74. Eliminación de grupo creado - en ejecución plataforma Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

Una vez realizadas las acciones anteriores, se puede ver en el panel ***Website View*** los sitios accedidos y en la pestaña ***Requests*** las peticiones web realizadas, a su vez en el panel inferior se pueden identificar las pestañas ***Request*** y ***Response*** en donde es posible revisar la petición y respuesta para cada una de las peticiones.

Figura. 75. Resultado de escaneo de peticiones web a Quadodo en la plataforma Vega

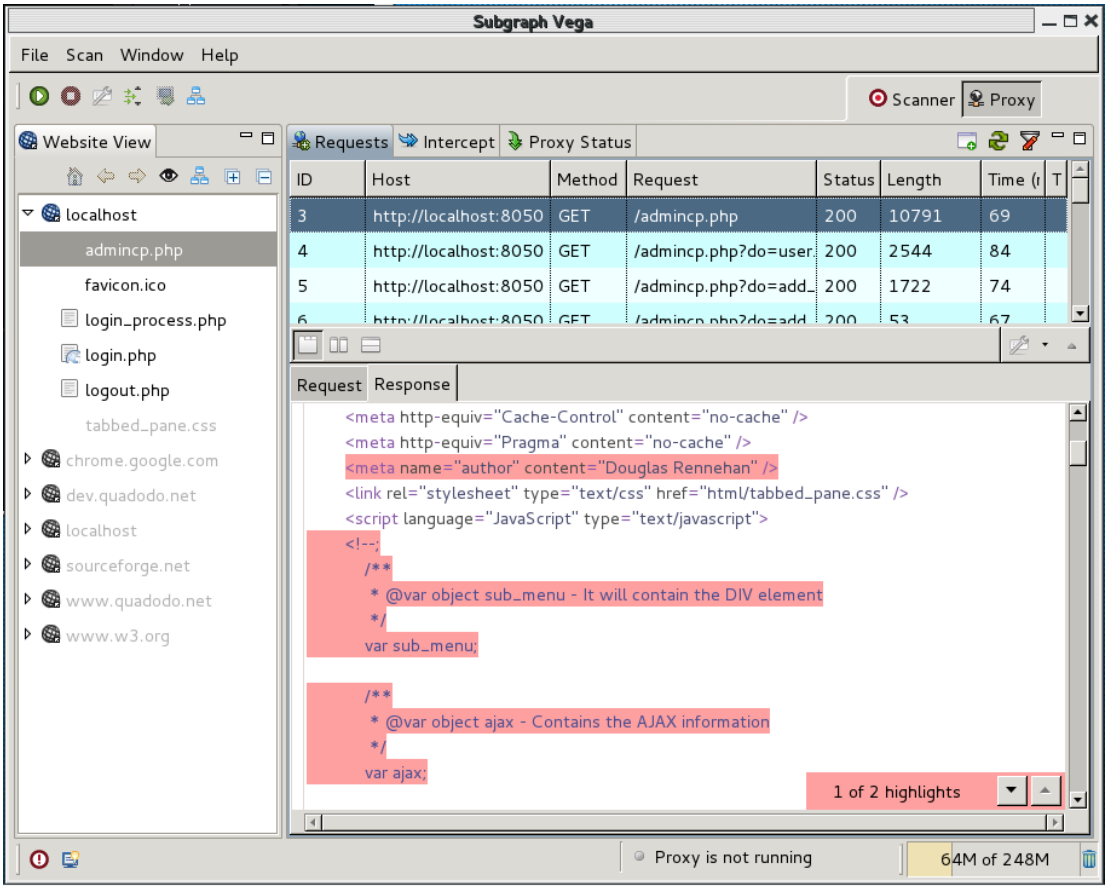


Fuente: El autor. Screenshot de pantalla de Vega.

A continuación se revisan cada una de las peticiones listadas en la pestaña Requests, para cada una de ellas, como se indicó se va mostrando en el panel inferior la petición (Request) y respuesta (Response) de la petición.

Para la petición que se indica a continuación, correspondiente a la página principal de administración (admincp.php), Vega subraya en un color rojo posibles vulnerabilidades que deben ser revisadas.

Figura. 76. Vulnerabilidades detectadas por Vega en página de Administración de Quadodo



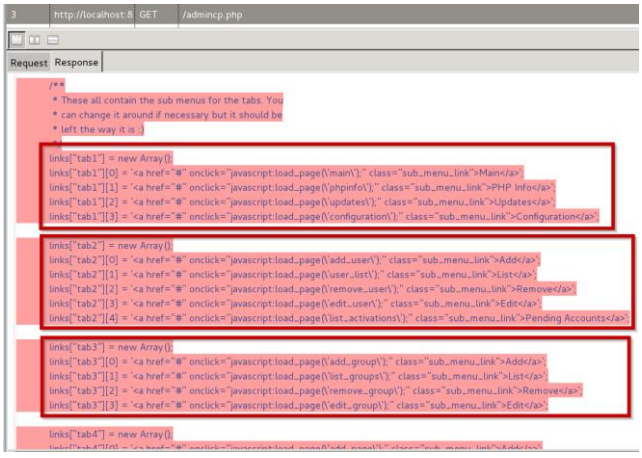
Fuente: El autor. Screenshot de pantalla de Vega.

11.4.1.2 Análisis de la vulnerabilidad

Para este tipo de vulnerabilidad, según lo indica OWASP: al ser ejecutada “se controla el flujo del código a través del uso de elementos del DOM que son manipulados por el atacante para cambiar el mismo.”

El código que VEGA resalta muestra que los enlaces que son utilizados en la página principal de administración, son construidos o armados en código Javascript almacenando las páginas que carga cada uno en un arreglo al cual se le asigna el código HTML del enlace.

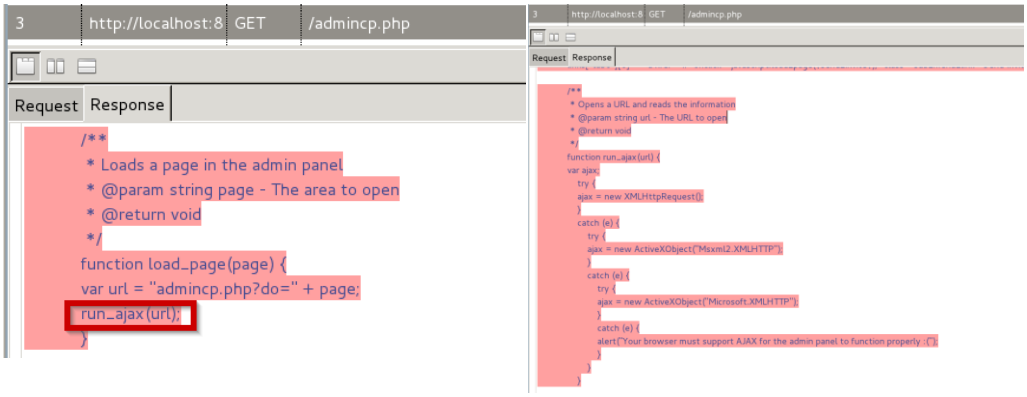
Figura. 77. Enlaces de Quadodo construidos en Javascript



Fuente: El autor. Screenshot de pantalla de Vega.

Adicional a lo anterior, cada enlace invoca una función en javascript llamada ***load_page*** que a su vez usa un método llamado ***run_ajax***, que usa la tecnología AJAX como mecanismo de cargue de las páginas.

Figura. 78. Función AJAX para el cargue de las páginas en Quadodo

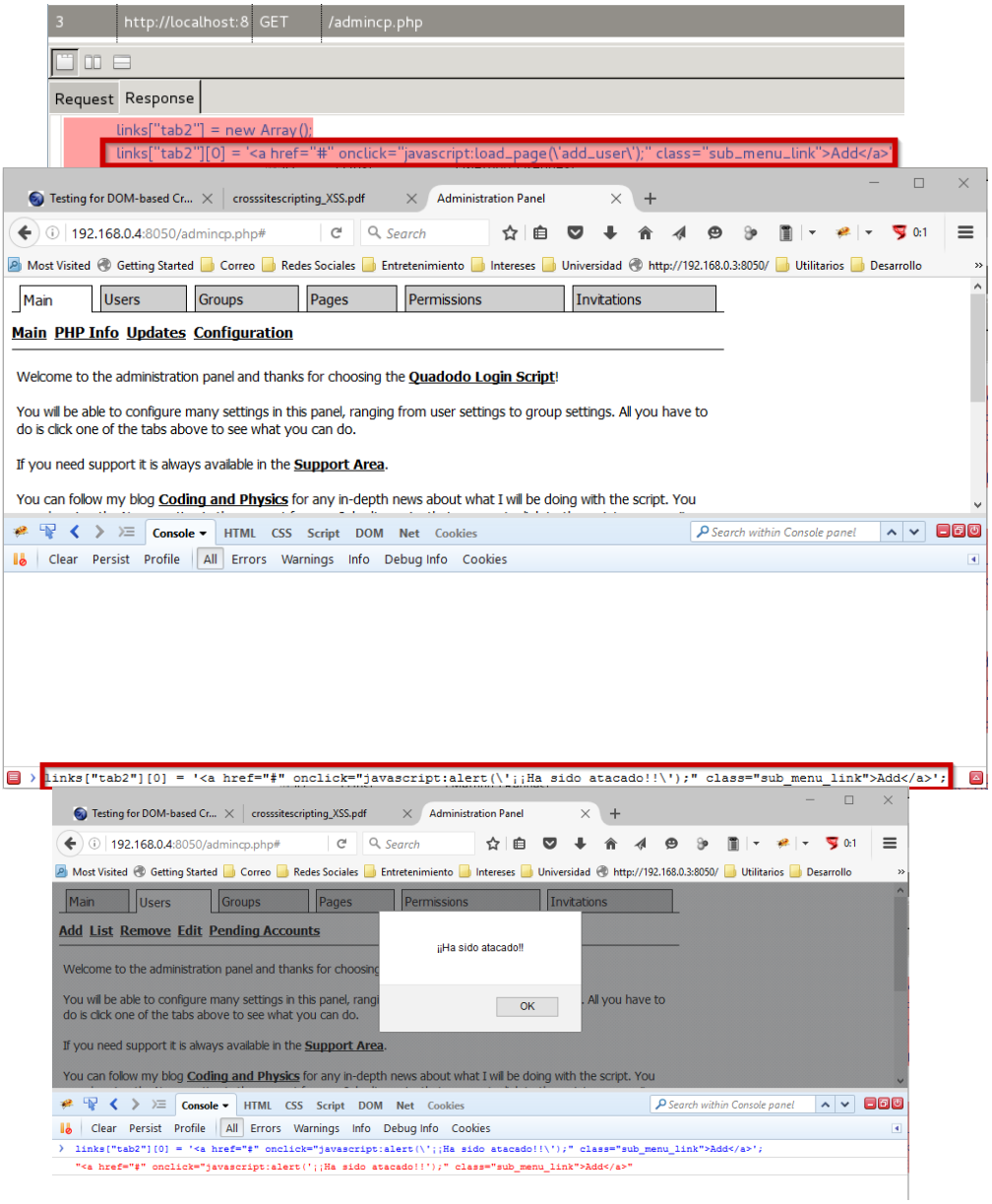


Fuente: El autor. Screenshot de pantalla de Vega.

Ahora bien, haciendo uso de este comportamiento, un atacante puede manipular el código Javascript de los enlaces de la siguiente manera:

Utilizando la herramienta Firebug para el navegador Firefox y accediendo a la Consola de Javascript, es posible modificar uno de los enlaces para que cuando el usuario haga clic le muestre un mensaje en pantalla, para ello se selecciona el enlace de crear usuarios (Add User) y se ejecuta esta acción.

Figura. 79. Modificación de enlace de Creación de Usuarios y mensaje de ataque generado



Fuente: El autor. Screenshot de pantalla de Vega, Quadodo y Firefox.

11.4.1.3 Resultado

Con la ejecución de esta prueba se detecta que es posible la manipulación del DOM en las opciones de menú permitiendo cambiar el flujo o el comportamiento de la página, esto genera a su vez la presencia de otras vulnerabilidades como se detalla a continuación.

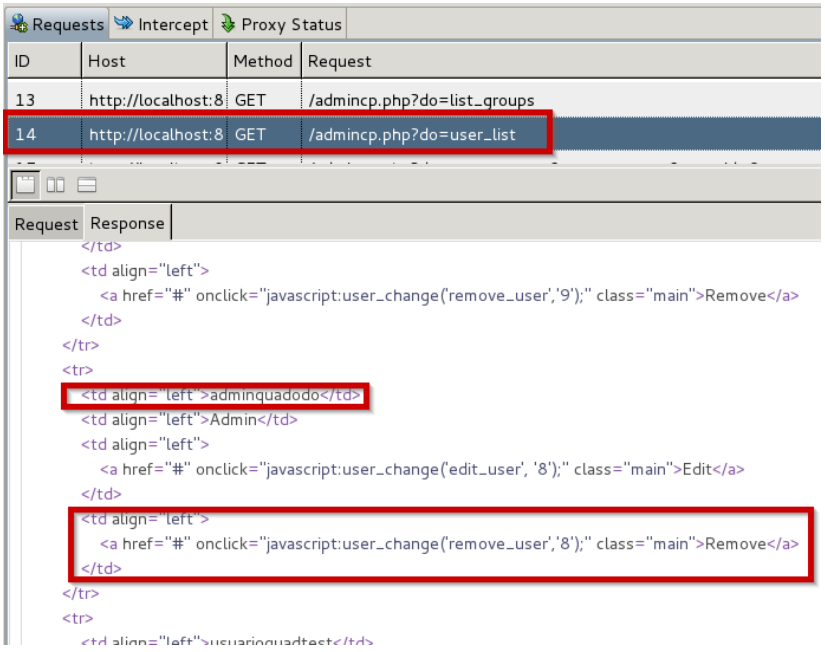
11.4.2 Testing for JavaScript Execution (OTG-CLIENT-002):

Como resultado y continuación de la prueba anterior, fue posible evidenciar que está presente una vulnerabilidad de XSS que puede ser aprovechada por un atacante.

Lo anterior no generó un daño en la aplicación, pero si en vez de mostrar una simple alerta, se toma el enlace que elimina un usuario y se inyecta en uno de estos enlaces del menú, el usuario puede realizar el borrado de un usuario administrador, como se ve a continuación.

De la lista de acciones capturadas en la herramienta VEGA, se toma aquella que lista los usuarios creados en Quadodo. El resultado HTML de respuesta lista los registros con un enlace de eliminación (con el nombre "***Remove***") que contiene un enlace con una función javascript que es ejecutada en el evento ***onclick***.

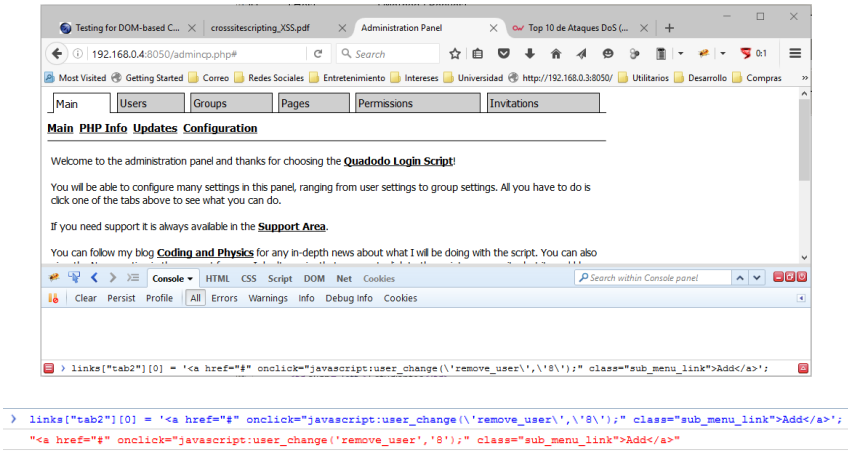
Figura. 80. Enlace de eliminación de usuario adminquadodo



Fuente: El autor. Screenshot de pantalla de Vega.

Se toma por ejemplo la función que aparece para el usuario ***adminquadodo*** y se inyecta a continuación en el enlace de creación de usuarios (***Add***) que se usó en la prueba anterior.

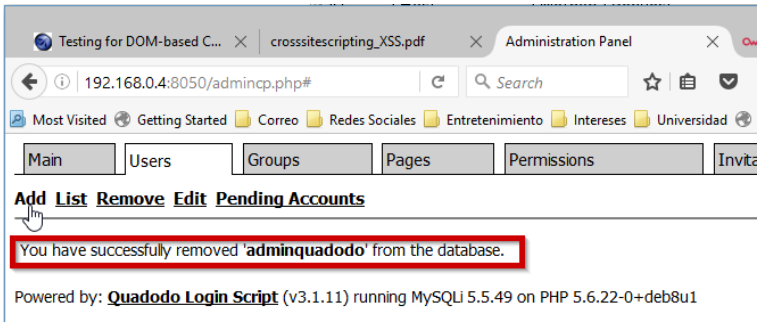
Figura. 81. Inyección de función de eliminación de usuarios



Fuente: El autor. Screenshot de pantalla de Firebug ejecutándose en Firefox.

A continuación el usuario hace clic en dicho enlace y borrar dicho usuario de la base de datos:

Figura. 82. Usuario adminquadodo eliminado como resultado del ataque



Fuente: El autor. Screenshot de pantalla de Quadodo.

11.4.2.1 Resultado

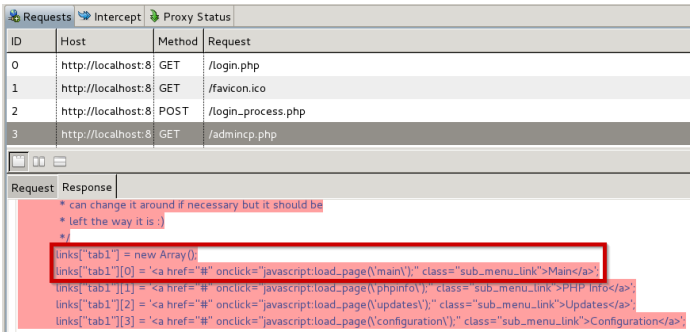
El manejo de construcción y redirección de los enlaces del menú no es seguro, permitiendo la inyección de código Javascript no seguro el cual como se pudo observar, puede afectar y manipular la información de la aplicación.

11.4.3 Testing for HTML Injection (OTG-CLIENT-003):

Es evidente que esta vulnerabilidad está presente de acuerdo a las pruebas dos pruebas realizadas anteriormente. La forma como se despliegan en el navegador las opciones de cada menú, asigna dinámicamente el enlace y la función Javascript a invocar, un ejemplo es el que se muestra a continuación en el que en vez de asignar un enlace se asigna una caja de texto con la cookie de la página.

Se toma uno de los enlaces del arreglo de Javascript esta vez el de la opción ***Main*** de la pestaña ***Main*** que se muestra en la pantalla principal:

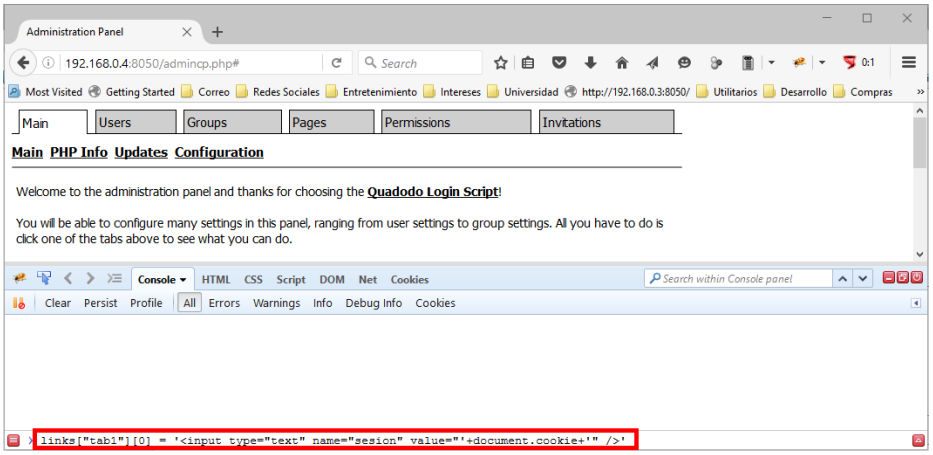
Figura. 83. Enlace asignado al enlace Main detectado por la herramienta Vega



Fuente: El autor. Screenshot de pantalla de Quadodo.

Se asigna el código HTML indicado para mostrar la caja de texto y la cookie del documento

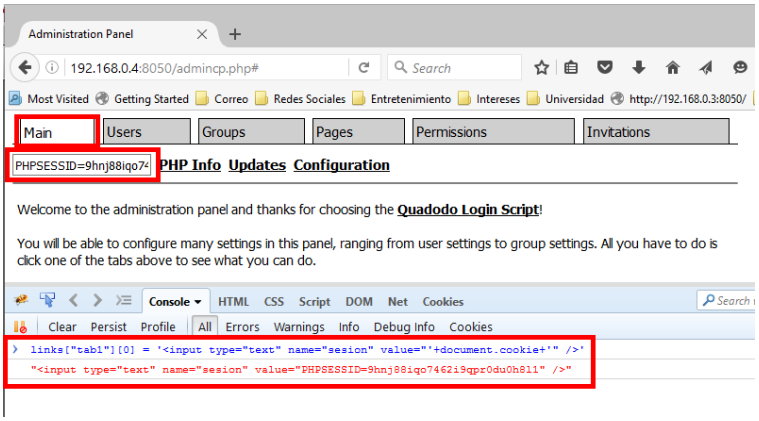
Figura. 84. Inyección de código HTML mediante Firebug



Fuente: El autor. Screenshot de pantalla de Firebug ejecutándose en Firefox.

Al seleccionar nuevamente la pestaña Main se despliega la caja de texto con el valor de la cookie la cual muestra el identificador de sesión de la página.

Figura. 85. Inyección de código HTML realizada

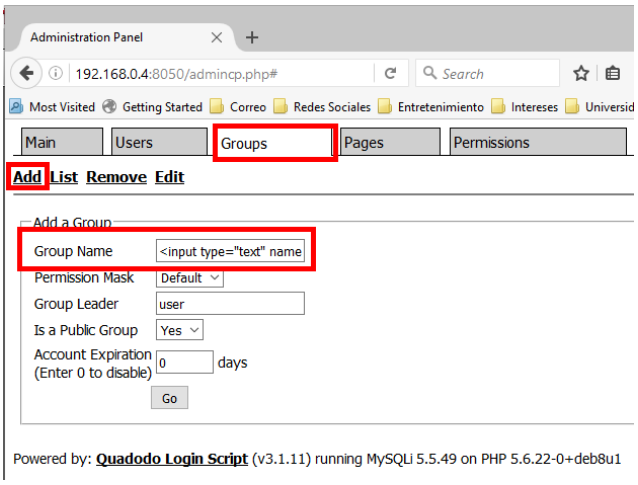


Fuente: El autor. Screenshot de pantalla de Firebug ejecutándose en Firefox.

Es clara la vulnerabilidad que presentan los enlaces del menú principal, ahora se comprueba si al ingresar información, Quadodo permite insertar código HTML que pueda cambiar el contenido de la página.

Se prueba inicialmente insertando un nuevo grupo en la aplicación accediendo a la pestaña ***Groups*** y la opción ***Add***, en el nombre del grupo se asigna código HTML.

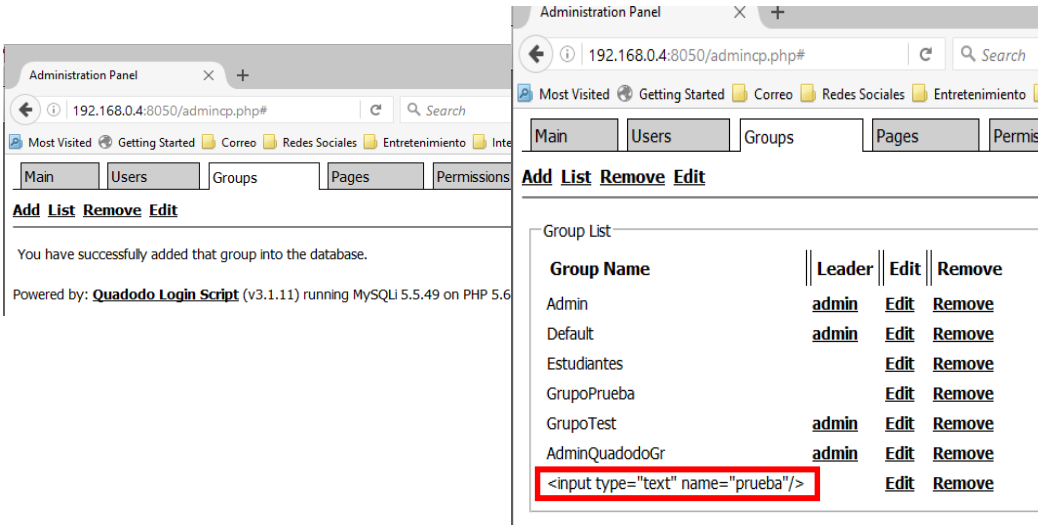
Figura. 86. Ingreso de HTML en nombre de grupo a crear en Quadodo



Fuente: El autor. Screenshot de pantalla de Quadodo.

Al hacer clic en el botón ***Go***, se muestra mensaje de confirmación de creación del grupo en base de datos. Para verificar de qué forma fue guardado y como se despliega se accede a la opción de listado de grupos (***List***) en la pestaña ***Groups***.

Figura. 87. Grupo creado con nombre HTML creado en base de datos



Fuente: El autor. Screenshot de pantalla de Quadodo.

Al verificar en Vega la respuesta para la página anterior se identifica que el nombre del Grupo ha sido guardado con codificación HTML para caracteres especiales. Al revisar en la base de datos se puede ratificar esta afirmación.

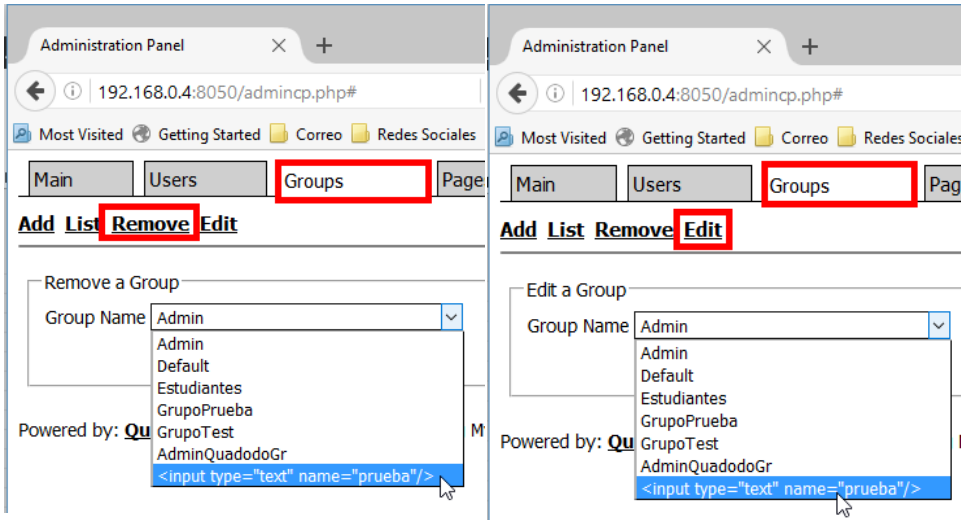
Figura. 88. Selección de grupos en la base de datos de Quadodo

```
mysql> show tables
    -> ;
+------------------------+
| Tables_in_QuadodoBD    |
+------------------------+
| qls3_config            |
| qls3_groups            |
| qls3_invitations       |
| qls3_masks             |
| qls3_pages             |
| qls3_password_requests |
| qls3_security_image    |
| qls3_sessions          |
| qls3_users             |
+------------------------+
9 rows in set (0.00 sec)

mysql> select * from qls3_groups
    -> ;
+----+-----------------------------------------------------------------+---------+-----------+--------+-----------------+
| id | name                                                            | mask_id | is_public | leader | expiration_date |
+----+-----------------------------------------------------------------+---------+-----------+--------+-----------------+
|  1 | Admin                                                           |       1 |         0 |      1 |               0 |
|  2 | Default                                                         |       2 |         1 |      1 |               0 |
|  3 | Estudiantes                                                     |       2 |         1 |      2 |               0 |
|  4 | GrupoPrueba                                                     |       2 |         1 |      2 |               0 |
|  5 | GrupoTest                                                       |       2 |         0 |      1 |               0 |
|  6 | AdminQuadodoGr                                                  |       1 |         1 |      1 |               0 |
|  8 | &lt;input type=&quot;text&quot; name=&quot;prueba&quot;/&gt; |       2 |         1 |      6 |               0 |
+----+-----------------------------------------------------------------+---------+-----------+--------+-----------------+
7 rows in set (0.00 sec)
```

Fuente: El autor. Screenshot de resultado de consola de MySql en Kali Linux.

Igualmente se verifica de qué forma se muestra en las opciones ***Remove*** y ***Edit***.

Figura. 89. Verificación de Grupo con nombre HTML en opciones Remove y Edit

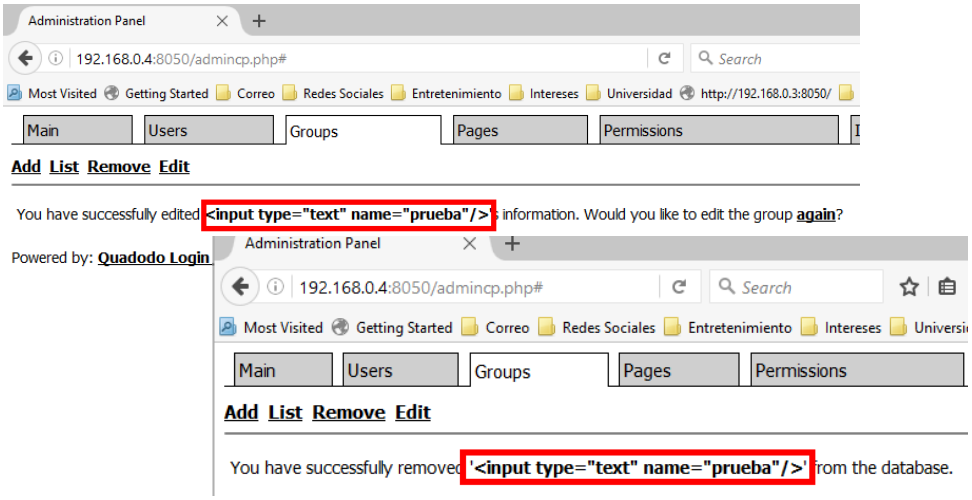


Fuente: El autor. Screenshot de pantalla de Quadodo.

Ya que estas dos opciones manejan listas desplegables el grupo se muestra con el nombre ingresado tal cual sin interpretar HTML, y ya que al editar o eliminar se

muestra un mensaje de confirmación en pantalla se verifica la forma como se muestra dicho mensaje en el navegador.

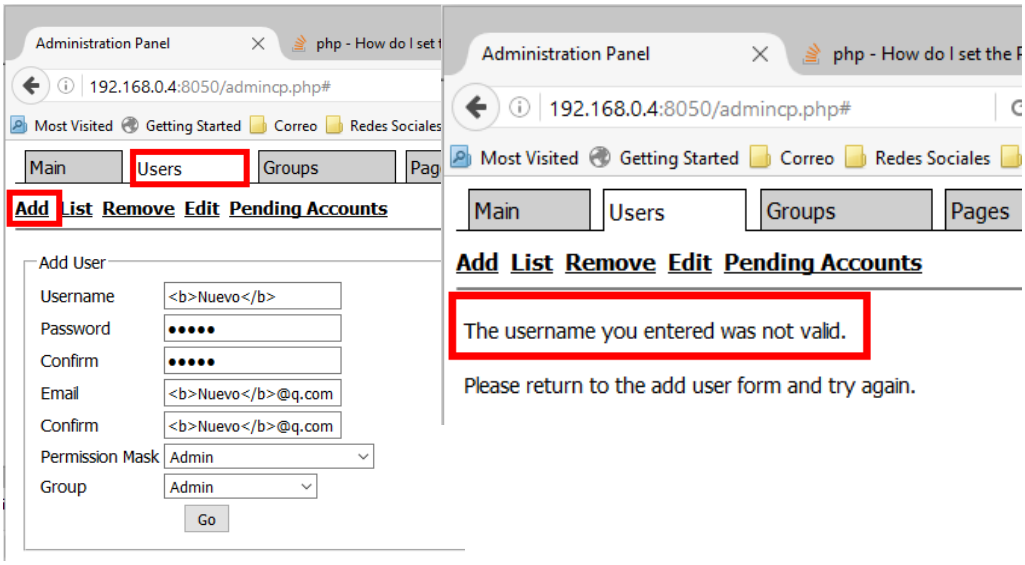
Figura. 90. Mensajes de confirmación edición y eliminación de Grupo con nombre HTML



Fuente: El autor. Screenshot de pantalla de Quadodo.

Se prueba igualmente al intentar ingresar un usuario nuevo ingresando en los campos código HTML.

Figura. 91. Intento de creación con nombre HTML



Fuente: El autor. Screenshot de pantalla de Quadodo.

El mensaje de respuesta indica que el nombre de usuario no es válido, lo cual indica que está correctamente verificado. Sin embargo, hay que tener en cuenta que el nombre de usuario depende de la expresión regular configurada en la opción ***Configuration***, así que el administrador deberá decidir cuál expresión es la más adecuada. Por otro lado, esta misma opción es configuración podría implementarse para validar los nombres de los Grupos que se crean en la aplicación.

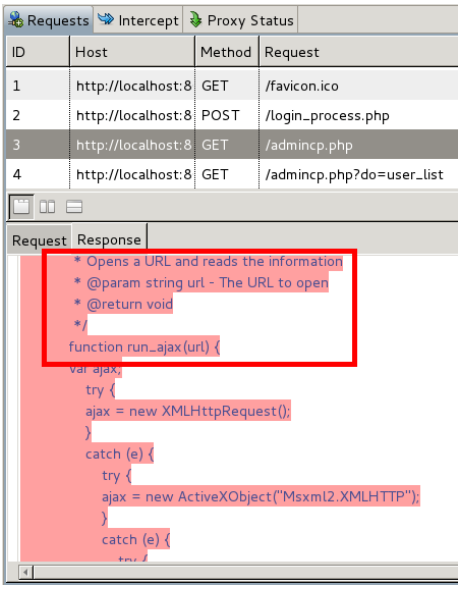
11.4.3.1 Resultado

Como resultado de esta prueba se detecta que La vulnerabilidad de inyección de código HTML está presente en la aplicación mediante la construcción de los menús que ya se ha indicado, por otro lado en la creación de grupos por ejemplo, a pesar de que se codifican los caracteres especiales que se ingresan en el nombre es permitido guardarlos con dichos caracteres, por lo cual se debería implementar una restricción para ello con el fin de disminuir la inyección de HTML. Se detecta sin embargo que se puede establecer el formato de nombre de usuarios bajo la opción de configuración, así que podría contemplarse la misma configuración para la creación de grupos.

11.4.4 Testing for Client Side URL Redirect (OTG-CLIENT-004)

Dentro de las funciones javascript detectadas en la página de administración a través de la herramienta VEGA, se encuentra la función run_ajax, la cual abre una URL específica y lee la información retornada.

Figura. 92. Función de javascript run_ajax detectada por la herramienta VEGA

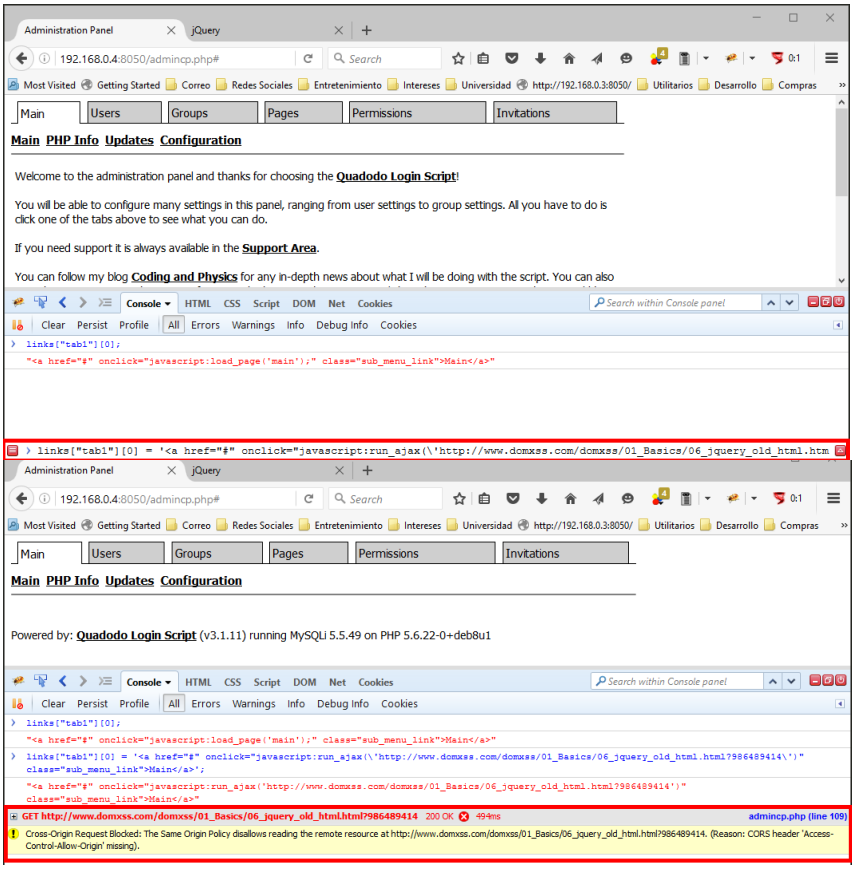


Fuente: El autor. Screenshot de pantalla de VEGA.

Ahora bien, ya que la vulnerabilidad de inyección y ejecución de Javascript está presente (según lo validado en las pruebas para los test ***OTG-CLIENT-001*** y ***OTG-CLIENT-002***), se aprovecha esta vulnerabilidad para asignar a uno de los enlaces del menú el llamado directo a esta función pero invocando una URL externa al sitio. Para ello se usa una de las URL que suministra OWASP en los ejemplos de vulnerabilidades y se asigna a uno de los enlaces. La URL a usar es: http://www.domxss.com/domxss/01_Basics/06_jquery_old_html.html?986489414.

Al hacer clic sobre el enlace modificado se puede evidenciar en la consola de la herramienta Firebug de Firefox que una petición de origen cruzado ha sido bloqueada:

Figura. 93. Inyección código javascript y petición de origen cruzado bloqueada por el navegador



Fuente: El autor. Screenshot de pantalla de Firebug ejecutándose en Firefox.

El código retornado al realizar la petición es 200 OK, es decir que la página fue accedida correctamente, sin embargo fue el navegador el que detectó la petición y la bloqueó. Esto quiere decir que es posible realizar la redirección a páginas externas a través de la función javascript indicada.

11.4.4.1 Resultado

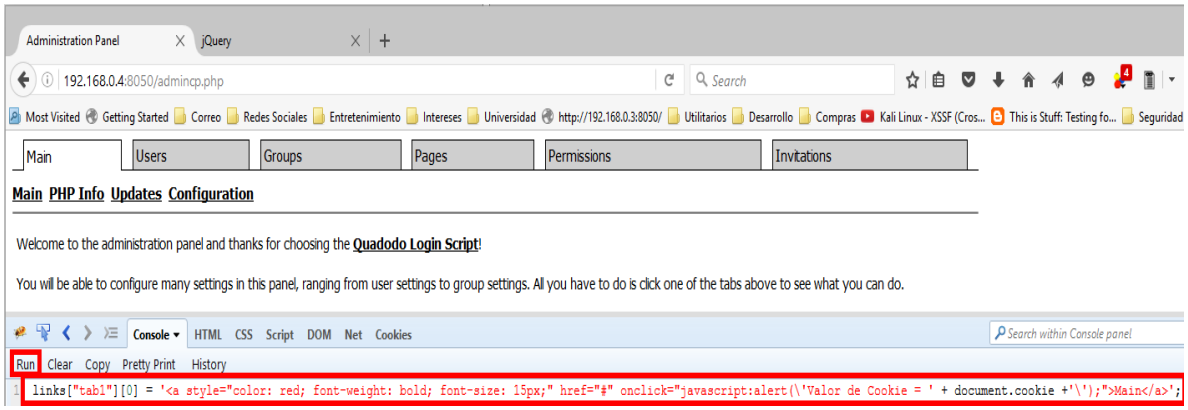
La vulnerabilidad de redirección de URL del lado del cliente está presente, así que es necesario que se implemente un mejor control de acceso en los enlaces para que la redirección o retorno de contenido de páginas externas sea validado con el fin de evitar cualquier riesgo de seguridad o robo de datos.

11.4.5 Testing for CSS Injection (OTG-CLIENT-005)

Finalmente esta vulnerabilidad se presenta como origen a la ya encontrada en el test **OTG-CLIENT-001**, la inyección de CSS es posible realizarla también y puede ser usado en las opciones de menú para que destacar una opción entre todas las demás para que el usuario haga clic y ejecute las funciones que también se modifiquen, como se demuestra en las pruebas realizadas de Javascript y HTML de los test ***OTG-CLIENT-002*** y ***OTG-CLIENT-003*** respectivamente.

A continuación se muestra la inyección de código en uno de los enlaces usando la herramienta Firebug, cambiando el CSS aplicado al mismo y la función que invoca.

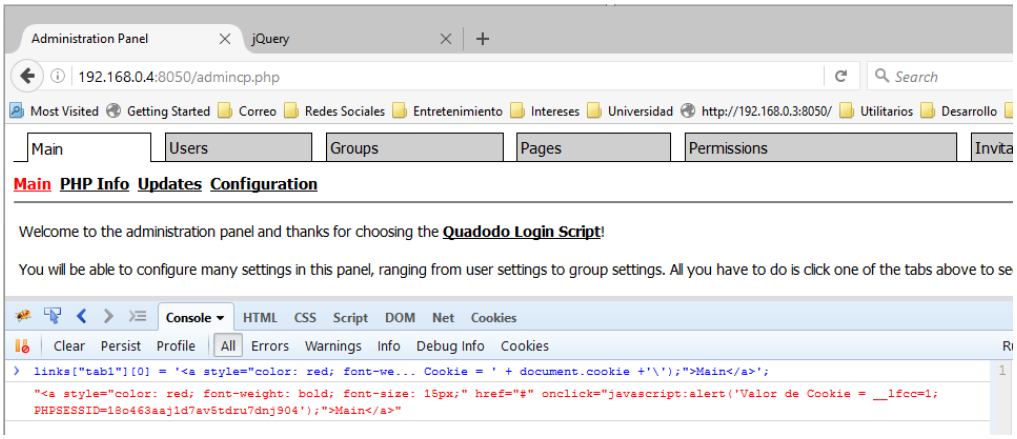
Figura. 94. Inyección de CSS y de javascript



Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

El resultado es el que se muestra en la imagen siguiente, como se observa se ha cambiado el estilo aplicado al enlace ***Main*** de la página principal.

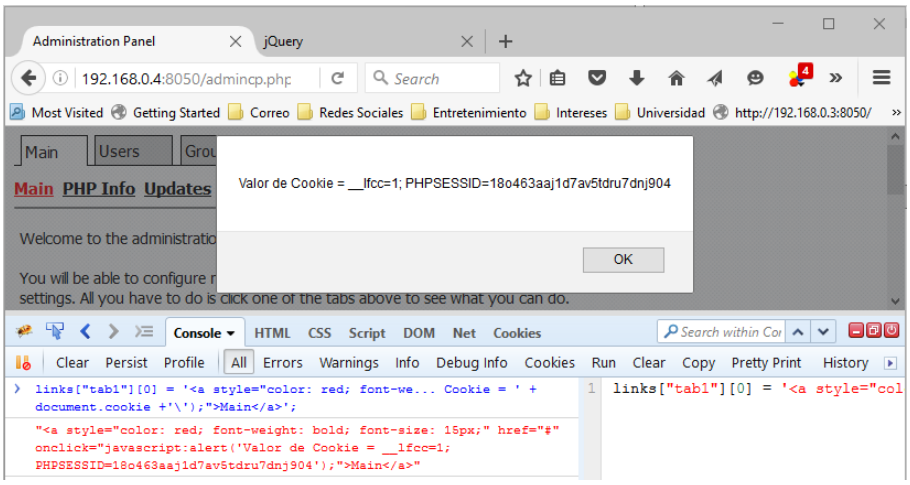
Figura. 95. CSS Inyectado exitosamente.



Fuente: El autor. Screenshot de Quadodo.

Finalmente al hacer clic, el usuario visualiza el siguiente mensaje inyectado.

Figura. 96. Ejecución de mensaje inyectado Javascript y CSS



Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

11.4.5.1 Resultado

Fue posible realizar la inyección de CSS, por lo cual hace posible modificar la interfaz del usuario haciendo que este ejecute acciones sobre opciones que sean resaltadas para a su vez realizar acciones inesperadas.

11.4.6 Test Local Storage (OTG-CLIENT-012)

De acuerdo a lo descrito para este test, el objetivo es identificar si datos sensibles están siendo almacenados y que luego puedan ser accesados, por ejemplo, a través de Javascript.

Para ello y según las indicaciones del documento *OWASP Testing Guide versión 4*, se puede ejecutar el siguiente código javascript para iterar sobre el almacenamiento local (localStorage) y de esta manera identificar si el sitio web o aplicación web almacena datos:

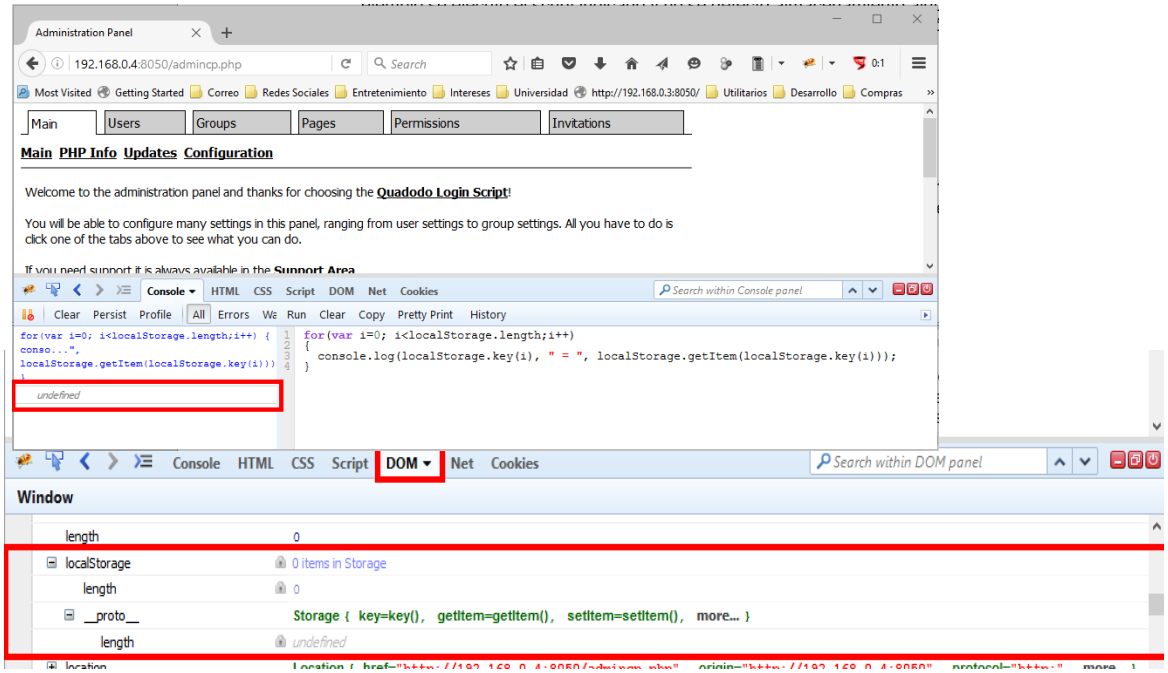
Figura. 97. Javascript de iteración de almacenamiento local

```
for(var i=0; i<localStorage.length;i++)
{
console.log(localStorage.key(i),"=",localStorage.getItem(localStorage.key(i)));
}
```

Fuente: El autor. Código Javascript.

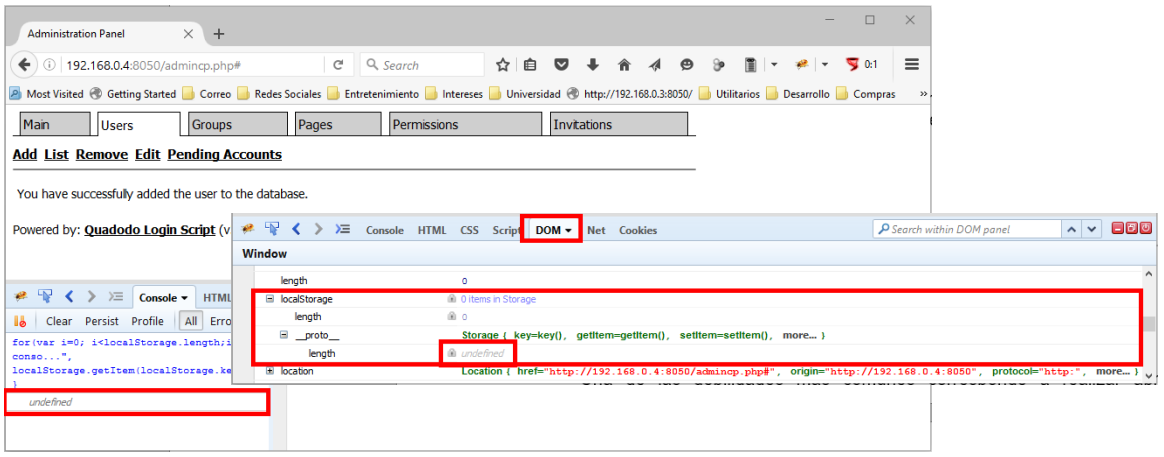
Al realizar el acceso de usuario, una creación de usuario y una creación de grupo por ejemplo se ejecutó el script indicado y no se detectó almacenamiento alguno. Por otro lado, se verificó también con la herramienta Firebug en Firefox pero no se halló almacenamiento local por parte de Quadodo.

Figura. 98. Ejecución de javascript y validación de localStorage en Firebug en el ingreso de usuario



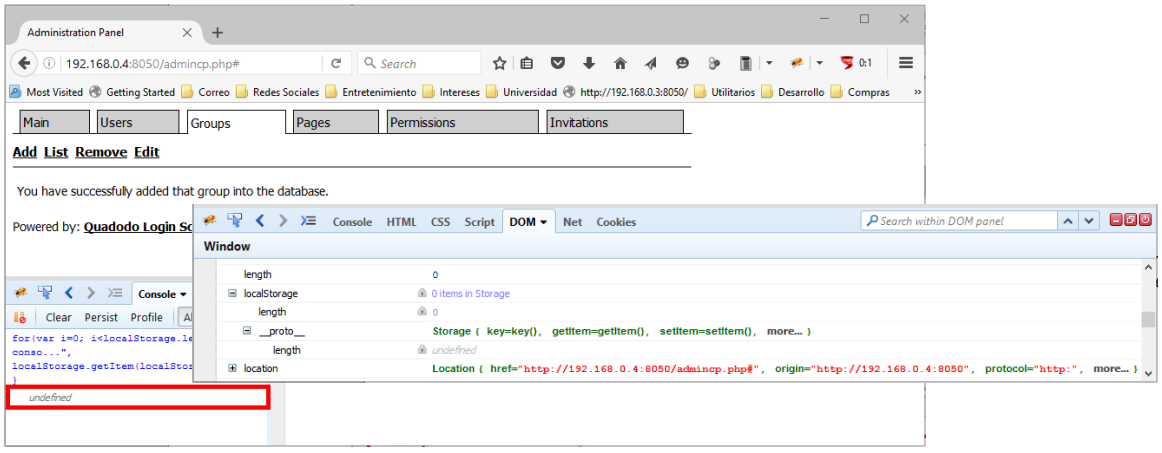
Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

Figura. 99. Ejecución de javascript y validación de localStorage en Firebug la creación de usuario



Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

Figura. 100. Ejecución de javascript y Validación de localStorage en Firebug en la creación de grupo



Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

11.4.6.1 Resultado

Como se puede observar, no se detectó almacenamiento local por parte de Quadodo, por lo cual no hay contenido que pueda ser usado y manipulado. Es importante tener en cuenta que “en promedio los Browsers permiten almacenar cerca de 5MB de almacenamiento local por dominio, lo cual comparado con los 4KB de las cookies es una gran diferencia” y que según lo indica el proyecto OWASP “los datos almacenados persisten después de que la ventana ha sido cerrada, por lo cual es una mala idea almacenar de esta manera datos sensibles o identificadores de sesión.” (OWASP, 2014).

Por lo anterior, Quadodo al no utilizar almacenamiento local disminuye el acceso o almacenamiento de datos e información sensible, lo cual es muy importante si va a ser integrado a otras aplicaciones para la administración de usuarios.

11.5 AUTHORIZATION TESTING

Las pruebas de autorización permiten entender cómo funciona el proceso de validación de acceso a recursos permitidos y usar esta información para evitar el mecanismo de autorización, encontrar una ruta transversal de vulnerabilidad o encontrar formas de escalar privilegios asignados.

Las pruebas que se abarcan en el presente documento para este campo son las siguientes:

- ***Testing Directory traversal/file include (OTG-AUTHZ-001):*** Busca detectar métodos o funcionalidades implementadas incorrectamente al permitir el acceso de lectura o escritura a archivos no permitidos y que pueden ser usados por un agresor, debido a la ejecución arbitraria de código o de comandos.
- ***Testing for Bypassing Authorization Schema (OTG-AUTHZ-002):*** Esta prueba se enfoca en verificar como ha sido implementado el esquema de autorización del sistema para cada rol o privilegio de acceso a funcionalidades o recursos específicos.
- ***Testing for Privilege escalation (OTG-AUTHZ-003):*** Evalúa la presencia de escalamiento de privilegios de un estado a otro, es decir, el acceso que obtiene un usuario a uno o más recursos o funcionalidades por elevación de permisos que deben ser prevenidos por la aplicación.
- ***Testing for Insecure Direct Object References (OTG-AUTHZ-004):*** Tiene como objetivo identificar si la aplicación o el sistema permite el acceso a objetos por entradas suministradas por el usuario, lo cual puede representar un riesgo al permitir a atacantes acceder a recursos o registros modificando el valor o parámetro de referencia al objeto.

11.5.1 Testing Directory traversal/file include (OTG-AUTHZ-001)

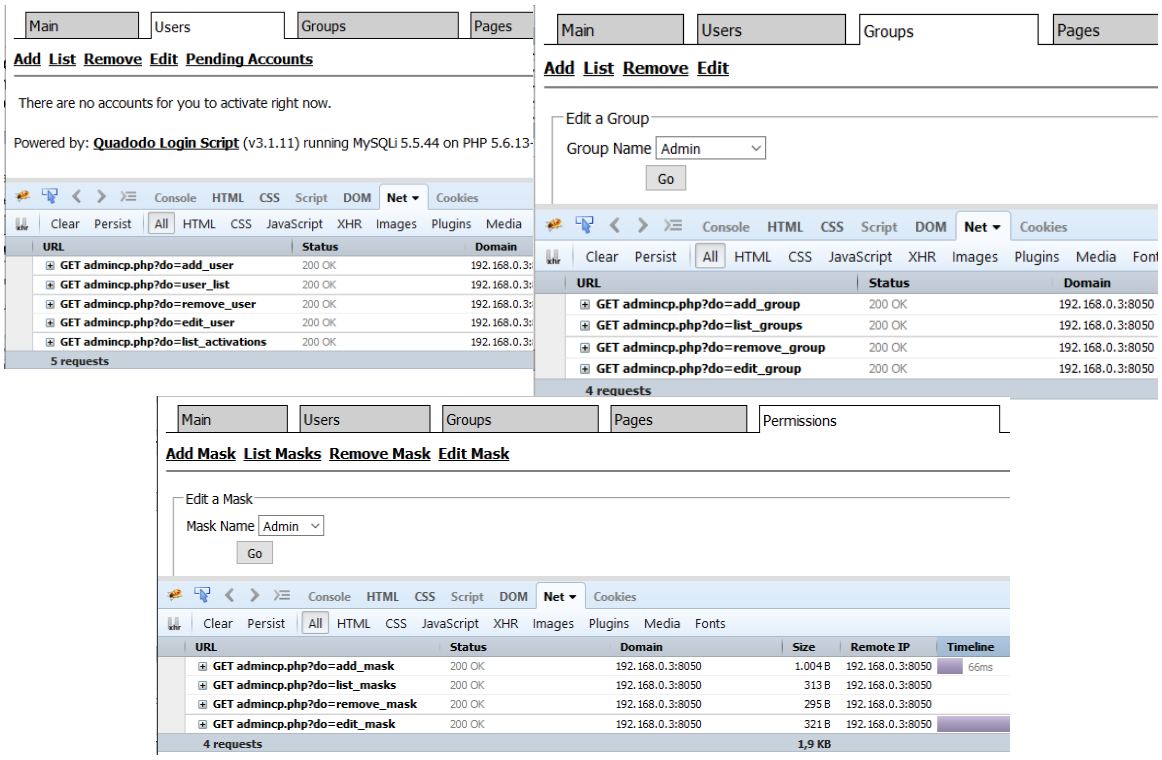
Para la ejecución de ese test es necesario usar una herramienta fuzzer, es decir, una herramienta usada para probar uno o más parámetros de una aplicación determinada. Para este caso se hace uso de la herramienta DotDotPwn, la cual según señala la documentación disponible en la página oficial de Kali Linux es flexible e inteligente para descubrir vulnerabilidades de directorios transversales in software alojado en servidores bajo protocolo HTTP, FTP y TFTP.

11.5.1.1 Ejecución de la prueba

Según lo identificado a lo largo de estas pruebas, se identifica que la aplicación Quadodo hace uso del parámetro ***do***, para mostrar de acuerdo a la opción seleccionada la página correspondiente, como consta en el siguiente análisis realizado a través de ***Firebug*** (plugin de desarrollo establecido para firebug).

A continuación, se muestra los valores que se asignan a la variable ***do*** al hacer clic en cada una de las opciones de las pestañas Users, Groups y Permissions.

Figura. 101. Parámetro do en las opciones de las pestañas Users, Groups y Permissions de Quadodo



Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

Para probar si este parámetro presenta una vulnerabilidad de seguridad, se procede a ejecutar DotDotPwn indicándole los siguientes parámetros:

- ***Módulo:*** Indica el protocolo ó la manera como será realizada la prueba. Ya que lo que se quiere es evaluar específicamente el parámetro do, se usa el módulo http-url.

- ***Host:*** a través del parámetro **–h** se indica el host donde se encuentra desplegada la aplicación, para este caso es el equipo local (127.0.0.1).

- ***Puerto:*** a través del parámetro –x se indica el puerto de acceso a la aplicación, se configura por lo tanto **8050** en el cual se encuentra disponible el sitio web de Quadodo.

- ***URL:*** ya que se usa para esta prueba el módulo **http-url** como se indicó, mediante el parámetro **–u** se indica la URL a testear. Sin embargo, para indicar que será el valor que se asigne al parámetro **do** el que se desea evaluar, se usa la palabra **TRANSVERSAL** para que DotDotPwn haga la asignación de valores de prueba correspondiente. Así que la configuración quedaría con la URL de esta manera:

  http://localhost:8050/admincp.php?do=TRAVERSAL

- ***Sistema operativo:*** Lo que hará DotDotPwn será asignar al parámetro indicado posibles rutas de sistema que puedan corresponder a carpetas o recursos, usando patrones de acceso con puntos, slashes y comodines. Se señala a través del parámetro **–o** que el sistema operativo es **unix,** en el cual se basa Linux y a su vez la distribución **Kali**.

- ***Patrón de texto:*** En caso tal que DotDotPwn logre identificar un recurso o ruta del sistema, tratará a su vez de encontrar los accesos o información del usuario ***root***. Para ello se señala dicho valor en el parámetro ***–k*** de la herramienta.

La instrucción entonces queda de la siguiente manera para ejecutarla:

Figura. 102. Instrucción de ejecución para DotDotPwn

```
root@6garzon:/# dotdotpwn.pl -m http-url -h 127.0.0.1 -x 8050 -u
http://localhost:8050/admincp.php?do=TRAVERSAL -o unix -k root
```

Fuente: El autor. Screenshot de Firebug ejecutándose en Firefox.

Figura. 103. Pantalla inicial de la herramienta DotDotPwn al ejecutarse desde la terminal de Kali Linux

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

La herramienta procede a realizar la asignación de valores posibles como se muestra a continuación:

Figura. 104. Fragmentos de ejecución de DotDotPwn sobre Quadodo

```
[+] Report name: Reports/localhost_04-24-2016_17-58.txt

[========== TARGET INFORMATION ==========]
[+] Hostname: localhost
[+] Setting Operating System type to "unix"
[+] Protocol: http
[+] Port: 8050

[=========== TRAVERSAL ENGINE ===========]
[+] Creating Traversal patterns (mix of dots and slashes)
[+] Multiplying 6 times the traversal patterns (-d switch)
[+] Creating the Special Traversal patterns
[+] Translating (back)slashes in the filenames
[+] Adapting the filenames according to the OS type detected (unix)
[+] Including Special sufixes
[+] Traversal Engine DONE ! - Total traversal tests created: 9840

[=========== TESTING RESULTS ============]
[+] Ready to launch 3.33 traversals per second
[+] Press Enter to start the testing (You can stop it pressing Ctrl + C)

[+] Replacing "TRAVERSAL" with the traversals created and sending
[*] Testing URL: http://localhost:8050/admincp.php?do=../etc/passwd
[*] Testing URL: http://localhost:8050/admincp.php?do=../etc/issue
[*] Testing URL: http://localhost:8050/admincp.php?do=../../etc/passwd
[*] Testing URL: http://localhost:8050/admincp.php?do=../../etc/issue
[*] Testing URL: http://localhost:8050/admincp.php?do=../../../etc/passwd
[*] Testing URL: http://localhost:8050/admincp.php?do=../../../etc/issue
[...]
[*] Testing URL: http://localhost:8050/admincp.php?do=..%5c..%5c..%5c..%5cetc%5cpasswd
[*] Testing URL: http://localhost:8050/admincp.php?do=..%5c..%5c..%5c..%5cetc%5cissue
[...]
```

```
[*] Testing URL: http://localhost:8050/admincp.php?do=..\..\..\..\..\etc\issue%00
[*] Testing URL:
http://localhost:8050/admincp.php?do=..\..\..\..\..\etc\issue%00index.html
[*] Testing URL:
http://localhost:8050/admincp.php?do=..\..\..\..\..\etc\issue%00index.htm
[*] Testing URL: http://localhost:8050/admincp.php?do=..\..\..\..\..\etc\issue;index.html
[*] Testing URL: http://localhost:8050/admincp.php?do=..\..\..\..\..\etc\issue;index.htm
[*] Testing URL: http://localhost:8050/admincp.php?do=..\..\..\..\..\..\etc\issue%00
[*] Testing URL:
http://localhost:8050/admincp.php?do=..\..\..\..\..\..\etc\issue%00index.html
[*] Testing URL:
http://localhost:8050/admincp.php?do=..\..\..\..\..\..\etc\issue%00index.htm
[*] Testing URL:
http://localhost:8050/admincp.php?do=..\..\..\..\..\..\etc\issue;index.html
[*] Testing URL:
http://localhost:8050/admincp.php?do=..\..\..\..\..\..\etc\issue;index.htm
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

11.5.1.2 Resultado obtenido

Al finalizar, la herramienta arroja el siguiente resultado después de la evaluación realizada:

Figura. 105. Resultado ejecución DotDotPwn

```
[+] Fuzz testing finished after 50.90 minutes (3054 seconds)
[+] Total Traversals found: 0

[+] Report saved: Reports/localhost_04-24-2016_17-58.txt
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

Como se puede observar, no se encontró ninguna inclusión o acceso a recursos del sistema, realizando la ejecución en 50,9 minutos.

11.5.2 Testing for Bypassing Authorization Schema (OTG-AUTHZ-002)

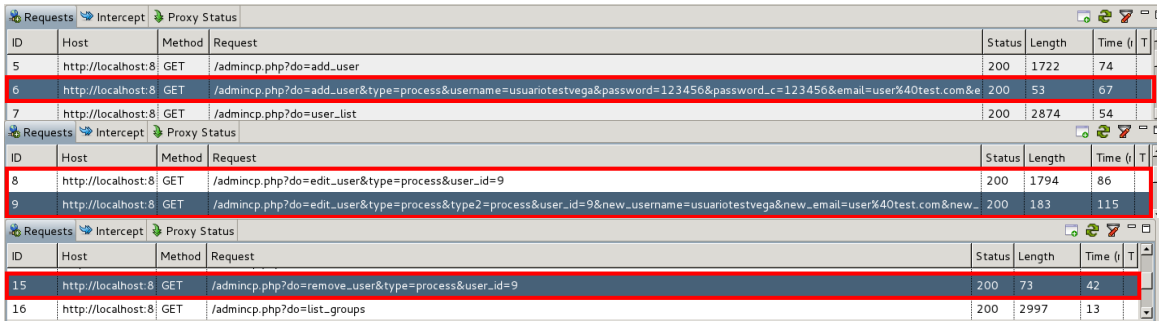
El desarrollo de este test que se muestra a continuación se enfoca en responder dos preguntas que plantea OWASP en su guía de pruebas concernientes al acceso de funcionalidades administrativas en la aplicación que se está evaluando. Ya que Quadodo provee dichas funcionalidades permitiendo al usuario la creación, eliminación y edición de usuarios y grupos del sistema entre otras entidades dentro del sistema es necesario resolver dichas incógnitas, las cuales son:

- ¿Es posible acceder a funciones administrativas incluso cuando el usuario ha ingresado con privilegios estándar?
- ¿Es posible usar estas funciones administrativas como un usuario con un rol diferente y para quien para la acción debe ser denegada?

Para responder a estas dos incógnitas se hace referencia a los resultados obtenidos con la herramienta VEGA en las pruebas de Client-Side Testing.

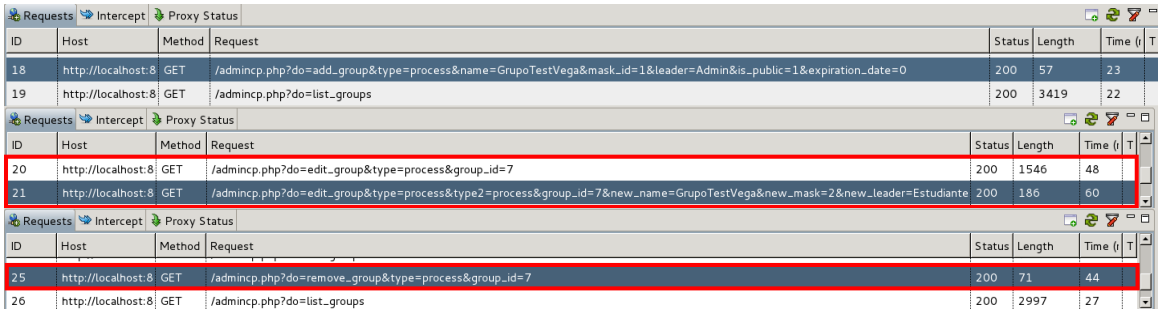
Como se puede observar en las siguientes imágenes, fue posible capturar las peticiones enviadas por URL en las acciones de creación, eliminación y edición de grupos y usuarios.

Figura. 106. URLs de administración de usuarios



Fuente: El autor. Screenshot de pantalla de VEGA.

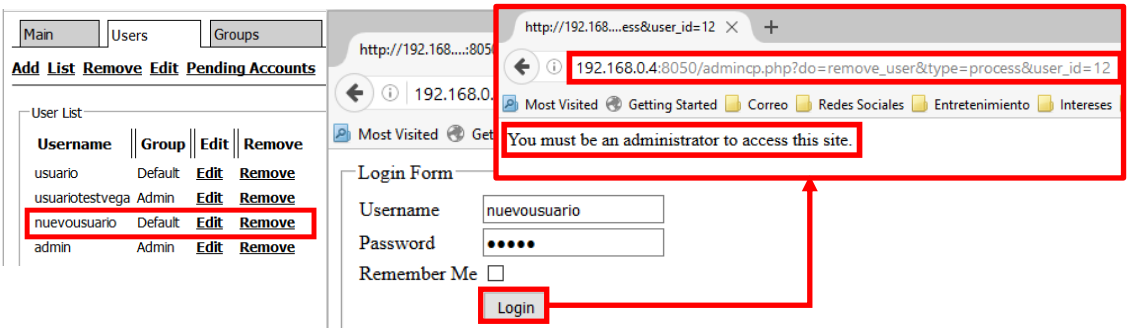
Figura. 107. URLs de administración de grupos



Fuente: El autor. Screenshot de pantalla de VEGA.

Dichas acciones solamente pueden ser realizadas por un usuario con privilegios de administrador. Para comprobarlo, se crea el usuario ***nuevousuario*** con privilegios estándar (Default) y se intenta acceder, por ejemplo en este caso, directamente a la URL de eliminación de usuarios enviando los parámetros correspondientes.

Figura. 108. Acceso directo a URL de eliminación de usuarios con usuario "nuevousuario" autenticado

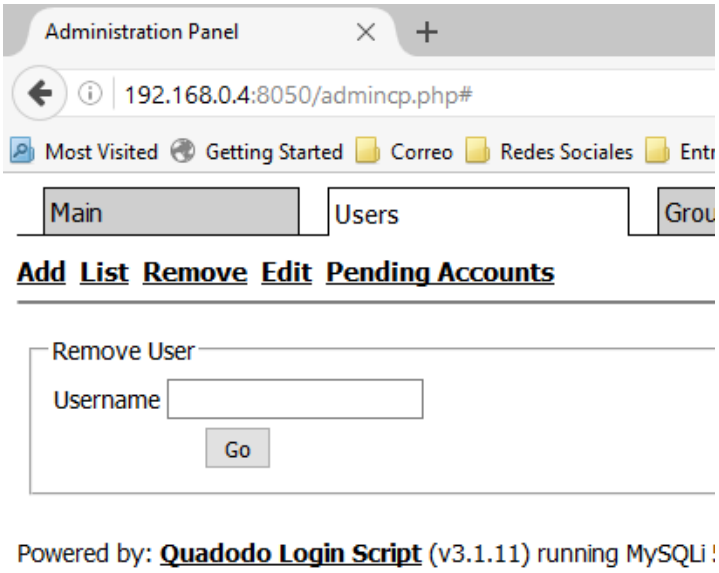


Fuente: El autor. Screenshot de pantalla de Quadodo.

Como se puede observar el mensaje indica que el usuario debe tener privilegios de administrador para poder acceder a la funcionalidad.

Ahora bien, otra prueba válida es abrir dos pestañas, en la primera se ingresa con privilegios de administrador y se accede a la opción de eliminación de usuario, por ejemplo.

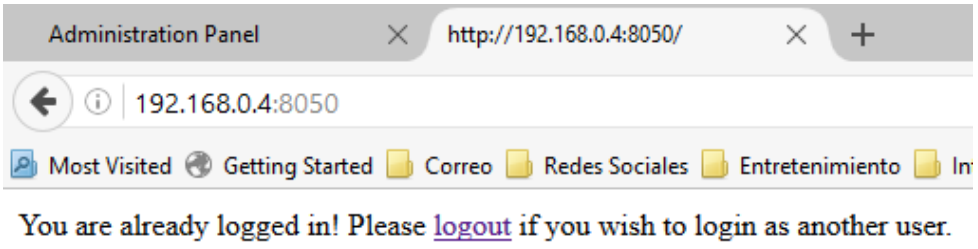
Figura. 109. Acceso a opción de eliminación de usuarios como administrador



Fuente: El autor. Screenshot de pantalla de Quadodo.

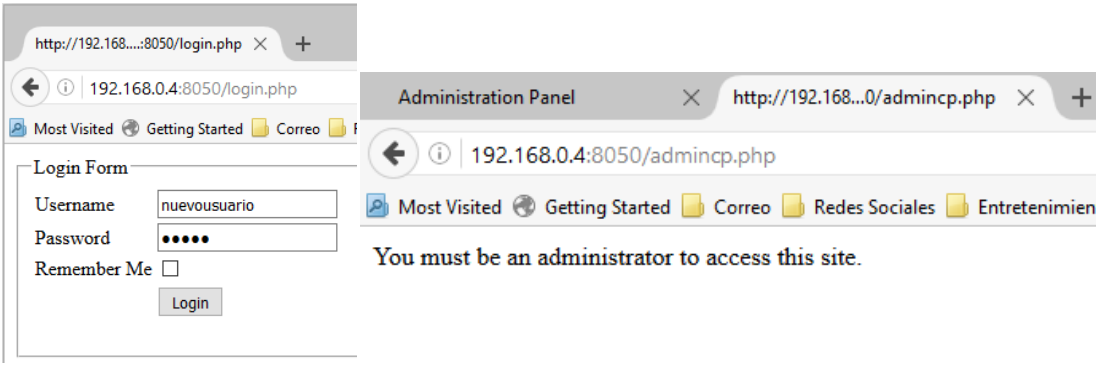
Y por otro lado en la otra pestaña se accede con un usuario con privilegio estándar. Como se puede observar en la segunda pestaña el sistema detecta que ya hay un usuario ingresado y pide que el usuario se autentique nuevamente.

Figura. 110. Acceso desde segunda pestaña donde se solicita autenticación nuevamente



Fuente: El autor. Screenshot de pantalla de Quadodo.

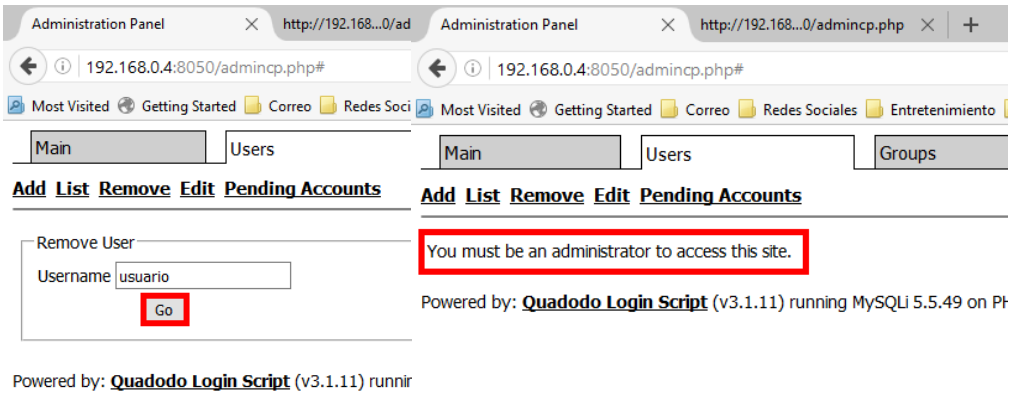
Figura. 111. Acceso desde segunda pestaña con usuario que posee permisos estándar



Fuente: El autor. Screenshot de pantalla de Quadodo.

Como paso final para esta prueba, en la primera pestaña se ingresa el nombre del usuario a eliminar y se hace clic en el botón de eliminación y se verifica el resultado.

Figura. 112. Mensaje de respuesta al intentar eliminar usuario con sesión de usuario sin privilegios



Fuente: El autor. Screenshot de pantalla de Quadodo.

Se observa que el sistema detecta efectivamente un usuario con privilegios estándar y no permite la eliminación del usuario.

Como punto a observar, todas las peticiones pasan por la página de administración (admincp.php), por lo cual sin importar la petición, centraliza las validaciones de acceso a las páginas de administración.

11.5.2.1 Resultado

Quadodo cuenta con una validación de permisos sobre los recursos, es por ello que para cada acción a ejecutar valida el usuario y el rol del mismo. Cubriendo de esta manera el acceso o ejecución no autorizada sobre la ejecución.

Por lo anterior y respondiendo a las incógnitas planteadas, no es posible acceder a funciones administrativas incluso cuando el usuario ha ingresado con privilegios estándar y como las acciones no pudieron ser utilizadas por dicha restricción, tampoco es posible usar funciones administrativas a través de un usuario para el cual la acción ha sido denegada.

11.5.3 Testing for Privilege escalation (OTG-AUTHZ-003)

Esta prueba es similar a la realizada anteriormente, sin embargo la guía de OWASP menciona otros elementos que pueden ser usados para suplantar los privilegios de los usuarios. Una de estas revisiones se realizó anteriormente con el acceso de dos pestañas diferentes, pero por otro lado, OWASP expone el ejemplo de campos ocultos que pueden estar almacenando como valor el rol del usuario y que al ser modificados pueda realizarse un ataque.

Para esta prueba la guía de OWASP sugiere la herramienta WebScarab, la cual se configura a modo de Proxy similar a la herramienta Vega pero con más opciones de configuración y rastreo. Sin embargo, adicional a esto, la herramienta permite mostrar en todas las páginas los campos ocultos mostrando su valor, de esta manera se verifica su contenido y el almacenamiento de algún indicador de rol del usuario.

Para iniciar esta herramienta se accede a través de una terminal en Kali Linux con el comando webscarab.

Figura. 113. Comando Webscarab

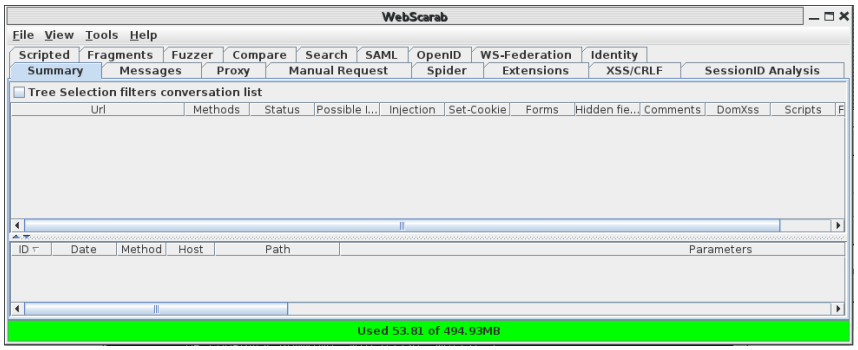
```
root@6garzon:/home/henrygarzon# webscarab
No plugins found!
Using WebScarab.whitelistRegex pattern : null. Will not save any data
for requests not matching this pattern
org.owasp.webscarab.ui.swing.EnabledBooleanTableCellRenderer[,0,0,0x0,i
nvalid,alignmentX=0.0,alignmentY=0.5,border=javax.swing.plaf.BorderUIRe
```

```
source$CompoundBorderUIResource@ecb67f,flags=296,maximumSize=,minimumSi
ze=,preferredSize=,defaultIcon=,disabledIcon=,disabledSelectedIcon=,mar
gin=javax.swing.plaf.InsetsUIResource[top=2,left=2,bottom=2,right=2],pa
intBorder=true,paintFocus=true,pressedIcon=,rolloverEnabled=true,rollov
erIcon=,rolloverSelectedIcon=,selectedIcon=,text=]
Help set not found
19:46:36 main(Proxy.parseListenerConfig): No proxies configured!?
19:46:36 main(SSLSocketFactoryFactory.<init>): Generating CA key
19:46:38 main(SearchModel.addSearch): Adding search Body search
19:46:38 main(SearchModel.addSearch): Adding search Request search
19:46:38 main(SearchModel.addSearch): Adding search Response search
19:46:38 main(SearchModel.addSearch): Adding search Request parameter
search
19:46:39 Listener-127.0.0.1:8008(Listener.listen): Proxy listening on
127.0.0.1:8008
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

Al ingresar este comando se abrirá la siguiente interfaz.

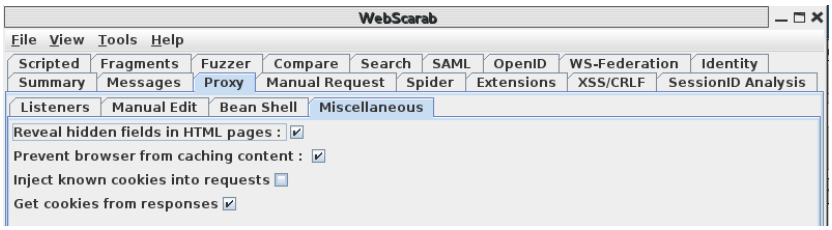
Figura. 114. Interfaz de Webscarab



Fuente: El autor. Screenshot de pantalla de WebScarab.

Para iniciar, se accede a la pestaña ***Proxy*** y a su vez a la pestaña ***Miscellaneous*** que se muestra y se seleccionan las siguientes opciones:

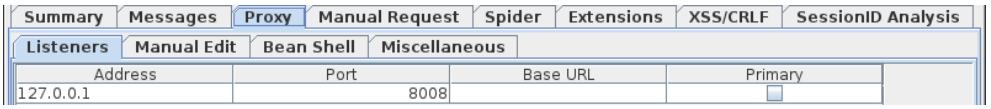
Figura. 115. Opciones seleccionadas Webscarab



Fuente: El autor. Screenshot de pantalla de WebScarab.

La primera opción muestra los campos ocultos (hidden fields) en las páginas HTML, la siguiente opción previene que el navegador almacene contenido y la última obtiene todas las cookies de las respuestas (responses) generadas. A continuación se verifica la configuración de proxy a utilizar en el navegador web. Para ello se accede a la pestaña Proxy y la pestaña Listeners, donde se muestra la IP y puerto de escucha del proxy.

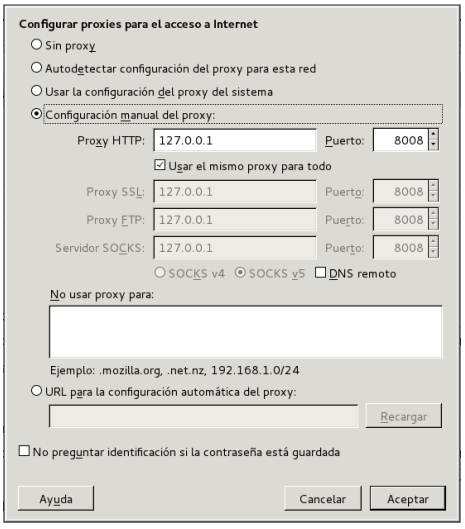
Figura. 116. Configuración de proxy Webscarab



Fuente: El autor. Screenshot de pantalla de WebScarab.

Tomando esta configuración se configura en el navegador estos parámetros.

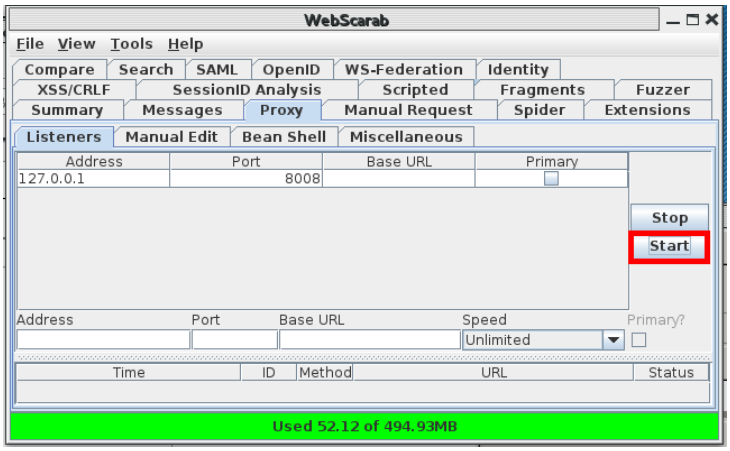
Figura. 117. Configuracion de proxy Webscarab en el navegador



Fuente: El autor. Screenshot de pantalla de Firefox.

Una vez realizado hecho esto se procede a hacer clic sobre el botón ***start*** que se encuentra en Webscarab y se comienza a navegar por la aplicación para detectar los campos ocultos y su valor.

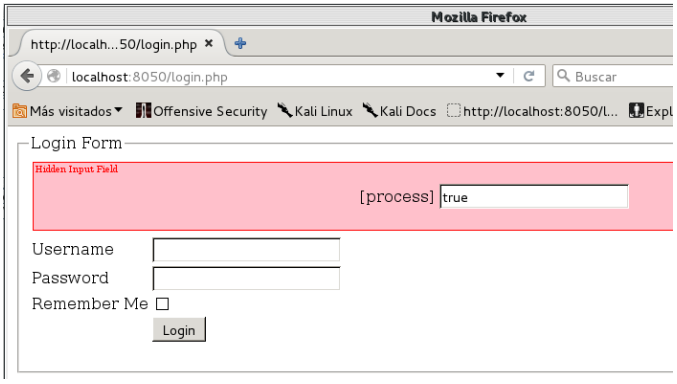
Figura. 118. Botón Start para iniciar el proxy de Webscarab



Fuente: El autor. Screenshot de pantalla de WebScarab.

A continuación se muestra el ingreso a la aplicación, se puede observar resaltado en rojo el campo oculto ***process*** el cual tiene el valor ***true***.

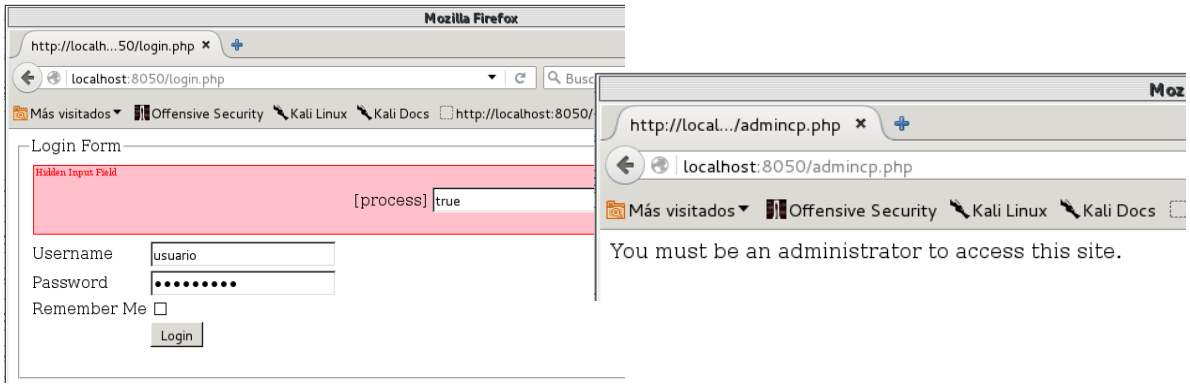
Figura. 119. Campo oculto process mostrado por Webscarab



Fuente: El autor. Screenshot de pantalla de Quadodo.

Ingreso a la aplicación con un usuario sin privilegios de administrador, se puede observar que los campos mostrados hasta el momento no muestran ningún rol asignado.

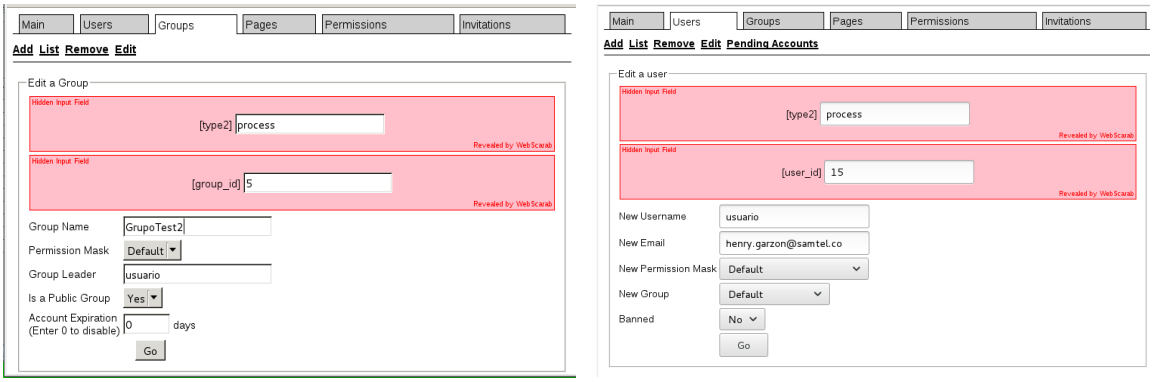
Figura. 120. Acceso de aplicación con usuario no administrador



Fuente: El autor. Screenshot de pantalla de Quadodo.

A continuación se muestra el formulario de edición de grupo y se despliegan dos campos ocultos: ***type2*** y ***group_id***, correspondiente al grupo que se está editando.

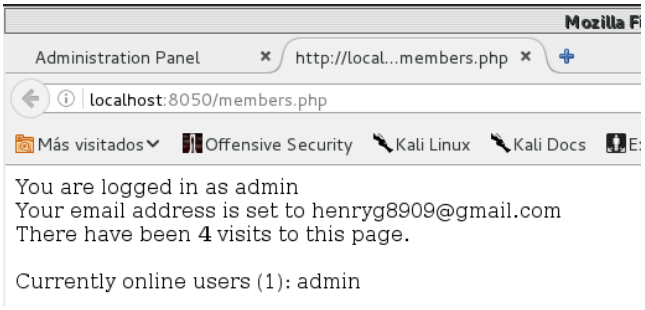
Figura. 121. Campos ocultos en el formulario de edición de grupo y de edición de usuario.



Fuente: El autor. Screenshot de pantalla de Quadodo.

Acceso a la página ***members.php*** donde se muestra la información de sesión del usuario ingresado, no se despliega ningún campo oculto con información del rol.

Figura. 122. Página members.php sin campos ocultos revelados



Fuente: El autor. Screenshot de pantalla de Quadodo.

Al continuar navegando en la aplicación no se detecta algún campo oculto con la información del rol del usuario actual, por otro lado los campos ocultos que se manejan almacenan los identificadores del registro que está editando.

11.5.3.1 Resultado

Los resultados arrojados para esta prueba no revelan campos que expongan o almacenen la información del rol del usuario que pueda suponer una vulnerabilidad para que un atacante lo modifique para realizar un robo de sesión.

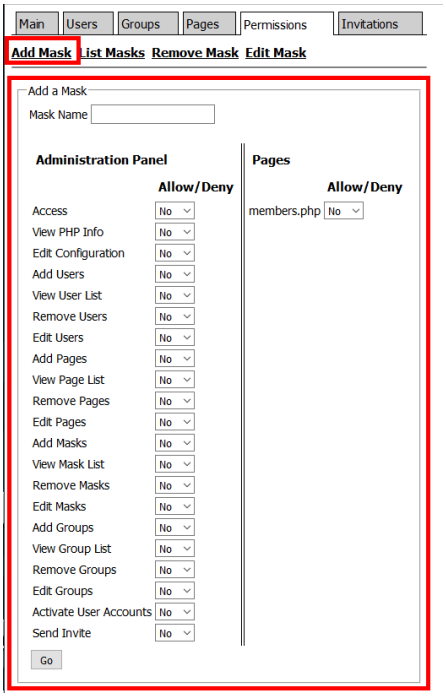
11.5.4 Testing for Insecure Direct Object References (OTG-AUTHZ-004)

Para esta prueba la guía de OWASP expone varios ejemplos de revisión a ser aplicados, generalmente son accesos realizados a través de la URL que se desea acceder suministrando un identificador que corresponde a un registro u objeto de base de datos sobre el cual se realiza una acción, muchas aplicaciones utilizan este método en el ámbito web. Sin embargo, la vulnerabilidad de seguridad surge cuando se accede a un objeto donde el usuario se supone no tiene acceso o permisos.

Para el caso de Quadodo, se han revisado los accesos de URL en las pruebas anteriores, sin embargo, para este caso se evalúan permisos sobre opciones específicas. Para ello se crean dos máscaras (***Mask***) de permisos bajo la funcionalidad que provee Quadodo a través de la pestaña ***Permissions***. Se hace clic en la opción ***AddMask*** para iniciar.

En la imagen se muestran los permisos que pueden ser configurados, bajo la columna ***Allow/Deny*** se selecciona la opción ***No*** para indicar que no hay privilegios sobre la opción indicada.

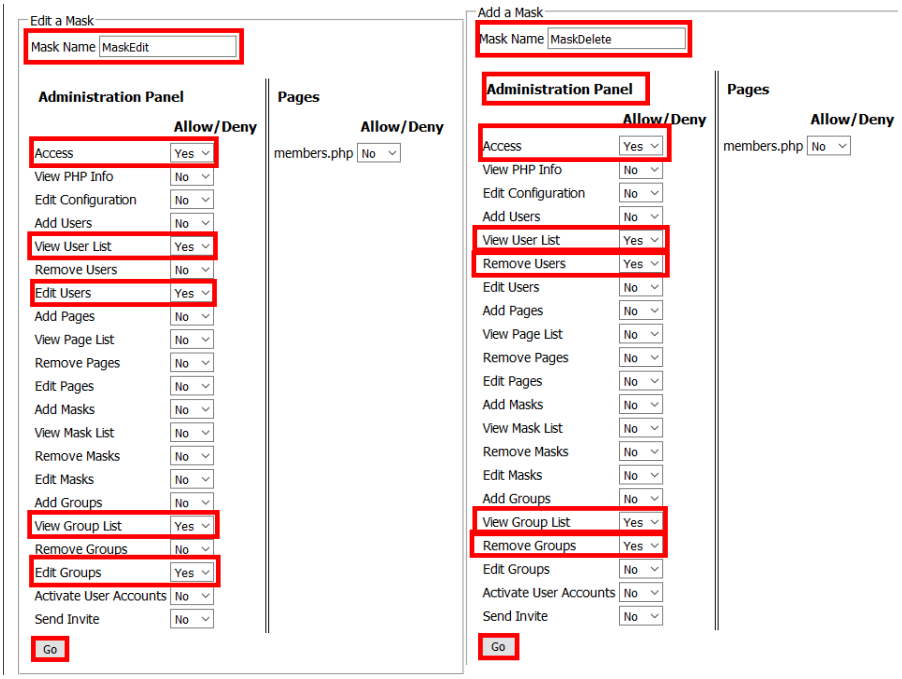
Figura. 123. Opción de agregar máscara de permisos en Quadodo



Fuente: El autor. Screenshot de pantalla de Quadodo.

Se crean a continuación dos máscaras de permisos, la primera se llamara ***MaskEdit*** la cual proveerá permisos de acceso al panel de administración solamente para listar y editar usuarios y grupos y la segunda se llamara ***MaskDelete*** que proveerá acceso al panel de de administración solamente para listar y eliminar usuarios y grupos de la aplicación.

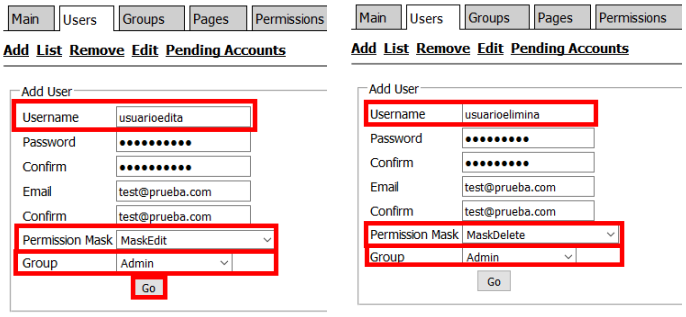
Figura. 124. Máscaras de permisos MaskEdit y MaskDelete creadas para prueba



Fuente: El autor. Screenshot de pantalla de Quadodo.

Una vez creadas las máscaras se crean dos usuarios administradores: el primero llamado ***usuarioedita***, al cual se le asigna la máscara de permisos ***MaskEdit*** y el otro llamado ***usuarioelimina*** que tendrá asignada las máscara de permisos ***MaskDelete***. En la figura se puede observar la máscara de permisos y el grupo asignados.

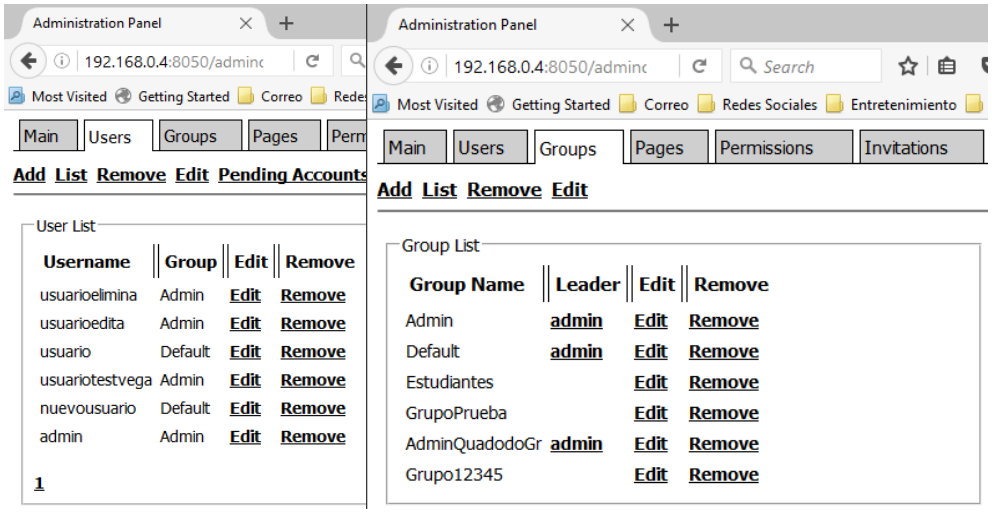
Figura. 125. Usuarios de eliminación y creación de usuarios y grupos creados para prueba



Fuente: El autor. Screenshot de pantalla de Quadodo.

Los usuarios y grupos creados al momento de la prueba son:

Figura. 126. Listado de usuarios y grupos existentes al momento de hacer la prueba

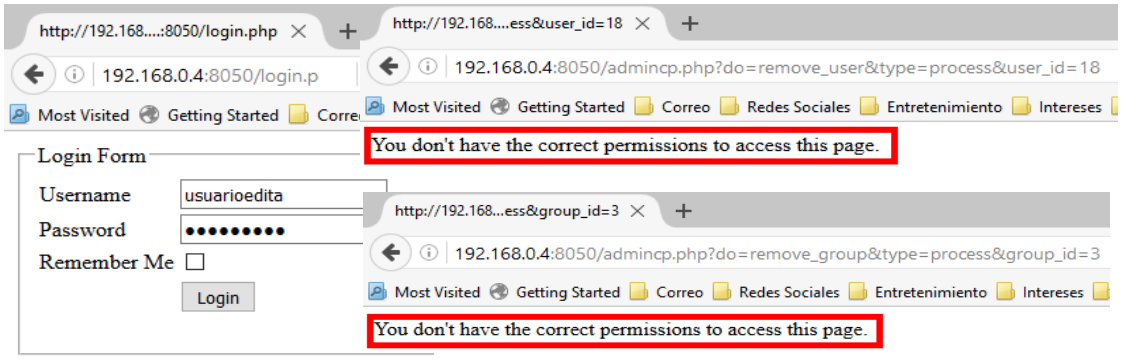


Fuente: El autor. Screenshot de pantalla de Quadodo.

Ahora bien, ya conocidas por pruebas anteriores (como es el caso de las pruebas realizadas con la herramienta Vega) las URL de edición y eliminación de usuarios y grupo, se procede a realizar lo siguiente:

Ingresar con el usuario ***usuarioedita***, que tiene permisos de edición de usuarios y grupos e intentar acceder a la URL de eliminación de usuarios.

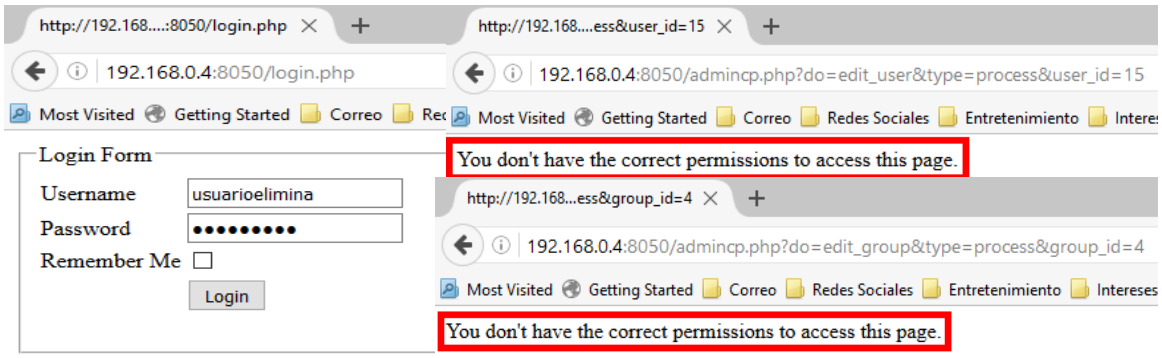
Figura. 127. Mensaje al acceder a página de eliminación de usuarios y de grupos con usuario sin permisos



Fuente: El autor. Screenshot de pantalla de Quadodo.

Ahora se ingresa a la aplicación con el usuario usuarioelimina y se accede directamente a las URL de edición de usuario donde no se le han asignado permisos.

Figura. 128. Mensaje al acceder a página de edición de usuarios y de grupos con usuario sin permisos



Fuente: El autor. Screenshot de pantalla de Quadodo.

11.5.4.1 Resultado

La aplicación Quadodo tiene implementadas validaciones para las acciones que realiza el usuario, evitando que este realice operaciones para las cuales no tiene permisos asignados.

11.6 INPUT VALIDATION TESTING

Una de las debilidades más comunes corresponde a realizar apropiadamente las validaciones de los datos de entrada provenientes de un cliente o de un entorno (OWASP, 2014). Estas vulnerabilidades pueden ser víctimas de los siguientes ataques:

- Cross site scripting
- SQL injection
- Interpreter injection
- Ataques Unicode
- Ataques de Sistema de archivos
- Sobrecarga de buffer (buffer overflows)

Los ataques realizados sobre Quadodo son los siguientes:

- ***Testing for Reflected Cross site scripting (OTG-INPVAL-001):*** Ocurre cuando un atacante inyecta código en una respuesta HTTP sencilla, el cual no

es almacenado en la aplicación, no es persistente y solamente impacta a los usuarios que acceden a un enlace malicioso o que enlaza a una página externa.

- ***Testing for SQL Injection (OTG-INPVAL-005):*** El objetivo es realizar una inyección parcial o completa de sentencias SQL a través de los datos de entrada de la aplicación o a través de los datos transmitidos desde el cliente a la aplicación web con el fin de leer, manipular o modificar datos de la base de datos que la aplicación esté utilizando. Para el caso de Quadodo Script, el autor indica que se implementaron controles contra el SQL Injection, por lo cual se realizan las verificaciones correspondientes.

11.6.1 Testing for Reflected Cross site scripting (OTG-INPVAL-001):

Esta vulnerabilidad como bien se describió, está presente en Quadodo teniendo en cuenta los resultados de las pruebas realizadas para los test Client-Side o del lado del cliente (OTG-CLIENT-001, OTG-CLIENT-002, OTG-CLIENT-003, OTG-CLIENT-004 y OTG-CLIENT-005).

11.6.1.1 Resultado

Como resultado de esta prueba se puede concluir que esta vulnerabilidad también abarca e impacta lo relacionado con vulnerabilidades de validaciones de entrada (Input Validation) haciéndola critica para el sistema y reduciendo su seguridad considerablemente. Es indispensable por lo tanto que Quadodo reestructure la manera como los script son construidos para reducir cualquier riesgo de seguridad relacionado con cross-site scripting.

11.6.2 Testing for SQL Injection (OTG-INPVAL-005):

Para la realización de este ataque se utilizará la herramienta **SQLMap** disponible en Kali Linux 2.0 aplicando el máximo nivel de ataque y especificando el motor de base de datos en algunos casos para identificar si es posible realizar inyección de código SQL. Para ello se utilizarán las URL identificadas en cada módulo en las pruebas de registro de usuarios:

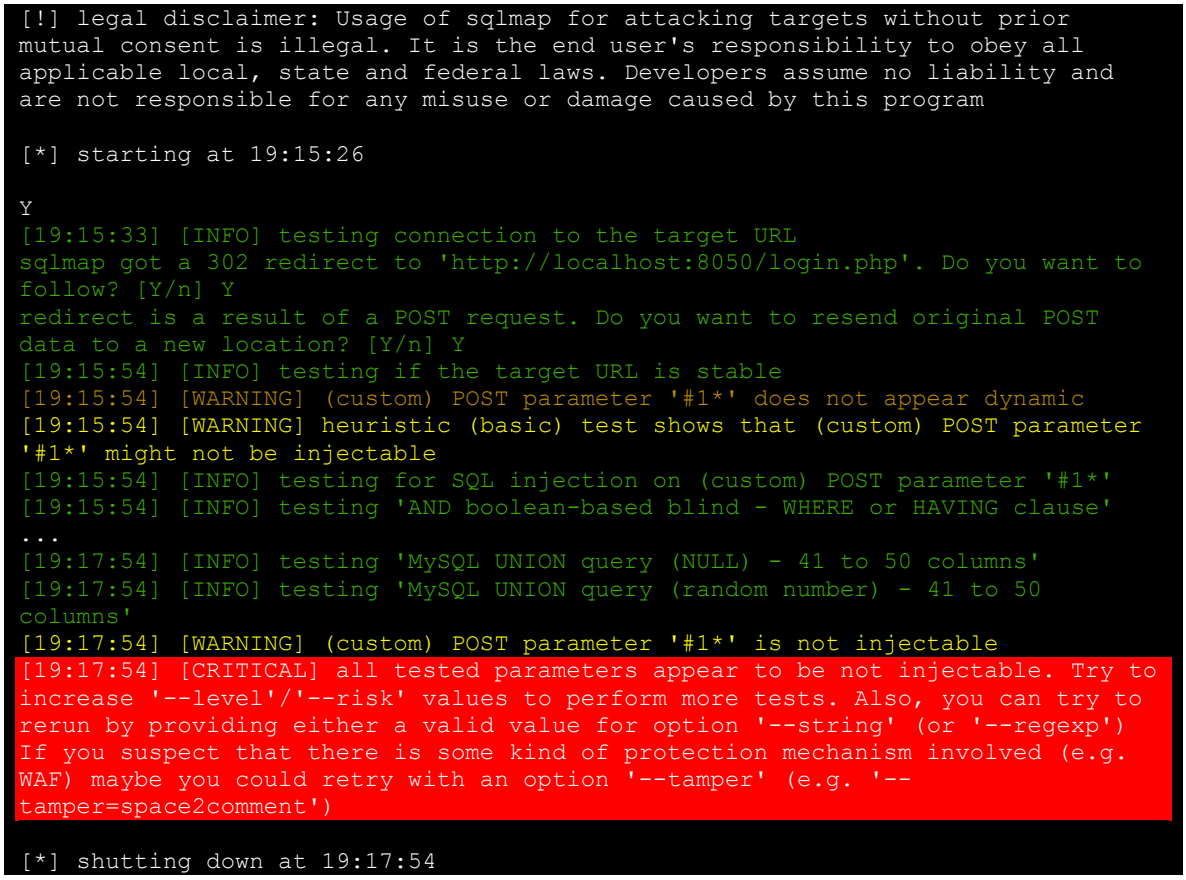
11.6.2.1 Ejecución sobre el módulo de Seguridad

La URL destino identificada al realizar ingreso de usuario se revisó en SQLMap para identificar si hay vulnerabilidades:
http://localhost:8050/login_process.php?process=true&username=admin&password=abcd1234*. Ejecución en SQLMap:

Figura. 129. Ejecución de SQLMap en página de Ingreso

Fuente: El autor. Screenshot de pantalla de Quadodo.

Figura. 130. Ejecución de SQLMap en página de Ingreso



Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

Según el resultado para esta prueba no se logró realizar la inyección de SQL usando los parámetros enviados en la URL como se indica en el texto subrayado en rojo.

11.6.2.2 Ejecución sobre el módulo de Registro

Se procede a lanzar una prueba de SQL Injection enviando datos de un usuario nuevo usando la URL identificada al realizar registro de usuarios: http://localhost:8050/register.php

Parámetros a través de método POST:

```
process=true&random_id=&username=usertest&password=1234abcd*&
password_c=1234abcd*&email=usuario%40hotmail.com&email_c=usua
rio%40hotmail.com
```

Se ejecuta commando sqlmap indicando mediante la opción ***–u*** la URL a validar y con la instrucción ***–data*** los datos a enviar mediante método POST, se especifica por otro lado se indica el nivel de validación mediante la opción ***–level***, se indica el nivel 5 el cual ejecuta de una forma exhaustiva un largo número de carga de parámetros. Finalmente se indica mediante la opción ***–dbms*** el motor de base de datos (MySQL en este caso) para que se ejecuten cargas de parámetros específicos para este motor y aumentar la posibilidad de encontrar vulnerabilidades.

Figura. 131. Ejecución de SQLMap en página de Registro de Usuarios

```
[19:11:38] [CRITICAL] target URL is heavily dynamic. sqlmap is going to retry
the request
...
[19:13:36] [WARNING] (custom) POST parameter '#2*' is not injectable
[19:13:36] [CRITICAL] all tested parameters appear to be not injectable. Try to
increase '--level'/'--risk' values to perform more tests. Also, you can try to
rerun by providing either a valid value for option '--string' (or '--regexp')
If you suspect that there is some kind of protection mechanism involved (e.g.
WAF) maybe you could retry with an option '--tamper' (e.g. '--
tamper=space2comment')

[*] shutting down at 19:13:36
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

Según el resultado para esta prueba no se logró realizar la inyección de SQL usando los parámetros enviados en la URL como se indica en el texto subrayado en rojo.

11.6.2.3 Ejecución sobre el módulo Panel de Control de Grupos

Se ejecuta entonces el SQLMap con las mismas opciones indicadas en la ejecución aplicada al módulo de Registro y según el resultado para esta prueba no se logró realizar la inyección de SQL usando los parámetros enviados en la URL como se indica en el texto subrayado en rojo:

Figura. 132. Ejecución de SQLMap en página de Creación de Grupos

```
henrygarzon@6garzon:~$ sqlmap -u
"http://localhost:8050/admincp.php?do=add_group&type=process&name=Estudiantes&mask_id=2&l
eader=usuario&is_public=1&expiration_date=0" --dbms=MySQL --level 5
...
[19:21:50] [INFO] testing if GET parameter 'do' is dynamic
[19:21:50] [WARNING] GET parameter 'do' does not appear dynamic
[19:21:50] [WARNING] heuristic (basic) test shows that GET parameter 'do' might not be
injectable
...
 [19:29:31] [INFO] testing 'MySQL UNION query (NULL) - 41 to 50 columns'
[19:29:31] [INFO] testing 'MySQL UNION query (random number) - 41 to 50 columns'
[19:29:31] [WARNING] Referer parameter 'Referer' is not injectable
[19:29:31] [CRITICAL] all tested parameters appear to be not injectable. Try to increase
'--level'/'--risk' values to perform more tests. Also, you can try to rerun by providing
either a valid value for option '--string' (or '--regexp') If you suspect that there is
some kind of protection mechanism involved (e.g. WAF) maybe you could retry with an
option '--tamper' (e.g. '--tamper=space2comment')

[*] shutting down at 19:29:31
```

Fuente: El autor. Resultado ejecución de comandos en terminal Kali Linux.

11.6.2.4 Resultado

El resultado arrojado por SQLMap en todos los casos se indicó que los parámetros enviados no son inyectables. La ejecución se realizó indicando el número máximo de nivel de proceso (5) y especificando incluso explícitamente el motor de base de datos. Sin embargo no fue posible ejecutar una inyección de SQL satisfactoria, por

lo cual se evidencia que lo indicado por el autor respecto a esta protección a nivel de seguridad es verdadera.

11.7 SESSION MANAGEMENT TESTING

El manejo de sesión (Session Management) es el mecanismo y uno de los componentes centrales de aplicaciones web, mediante el cual se mantiene el estado de un usuario, se realiza identificación y control del mismo mientras interactúa con la aplicación. Los entornos de aplicación más populares, como ASP o PHP proveen rutinas integradas de sesión, este último como ya se ha observado provee lo que se conoce como “Session ID” en una Cookie.

De acuerdo a lo anterior, es importante entonces que las aplicaciones realicen un buen manejo de sesión ya que con la misma un usuario no sólo se identifica, sino también accede y navega a través de contenidos en la aplicación.

Las pruebas a aplicar a Quadodo respecto al manejo de sesión son las siguientes:

1. ***Testing for Session Management Schema (OTG-SESS-001):*** El objetivo para esta prueba es evaluar que las cookies y otras claves de sesión sean creadas de una manera segura e impredecible, ya que de lo contrario si la forma como se hace no es robusta, un atacante puede fácilmente “secuestrar” (Hijack) la sesión de usuarios autenticados en la aplicación y dependiendo del privilegio que tengan estos sobre la misma acceder a recursos no autorizados.
2. ***Testing for cookies attributes (OTG-SESS-002):*** Ya que las cookies pueden ser vulneradas por vectores de ataque usándolas con el objetivo de suplantar usuarios, el objetivo de este test es comprobar que estas vulnerabilidades no están presentes en una aplicación web, lo cual es de suma importancia para evitar por ejemplo robo d einformación, accesos no autorizados.

11.7.1 Testing for Session Management Schema (OTG-SESS-001):

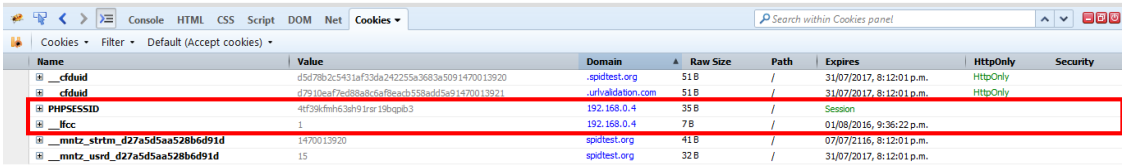
Para esta prueba o revisión, OWASP provee en su guía de pruebas (Testing Guide) algunos puntos a evaluar a nivel de cookies y de sesión que deben corresponder a características seguras de las mismas para reducir cualquier vulnerabilidad de información en las mismas. Para este caso se toman los resultados de las pruebas ya realizadas y que sirven como evidencia para estas evaluaciones:

1. ***Respecto a las cookies:*** Indica evaluar cuantas cookies son creadas y usadas en la aplicación y que información es almacenada en las mismas.

   Para ello se navega en Quadodo, accediendo, creando y eliminando un usuario y creando y eliminando un grupo. Como resultado y a través de la herramienta Firebug se detecta como constantes las siguientes cookies, de las cuales se

detectan como parte del dominio de Quadodo (192.168.0.4) las que se indican en el cuadro rojo:

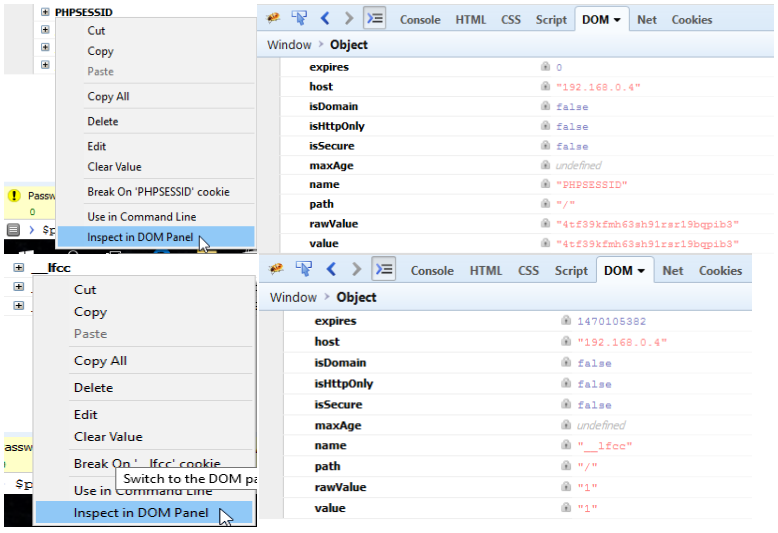
Figura. 133. Cookies generadas en Quadodo



Fuente: El autor. Screenshot de pantalla de Firebug.

Como se puede observar en las mismas se está almacenando el Identificador de sesión (***PHPSESSID***) y la cookie ***__lfcc*** con un valor 1. Al hacer clic derecho en cada una de ellas y hacer clic en la opción "***Inspect In DOM Panel***" del menú que se despliega se pueden observar las características detalladas de cada una de ellas, las cuales se analizarán en el siguiente punto:

Figura. 134. Detalles de las Cookies PHPSESSID y __lfcc



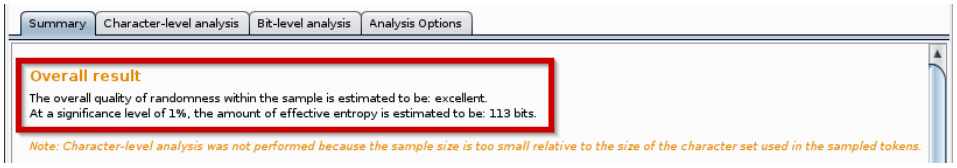
Fuente: El autor. Screenshot de pantalla de Firebug.

2. ***Respecto a la sesión:*** Hace énfasis en que tenga entre otras las siguientes características:

    a. ***Que sea impredecible:*** Como se pudo observar en los resultados del test de autorización OTG-AUTHN-004 de predicción de Id de sesión realizado

a través de la herramienta **Burpsuite**, el valor que se asigna a la cookie PHPSESSID no tiene un patrón específico que pueda determinarse fácilmente. A modo de aclaración hay que tener en cuenta que esta variable de sesión es provista por PHP y Quadodo a su vez la usa en la aplicación, por otro lado se puede determinar que cumple con esta característica de sesión indicada por OWASP.

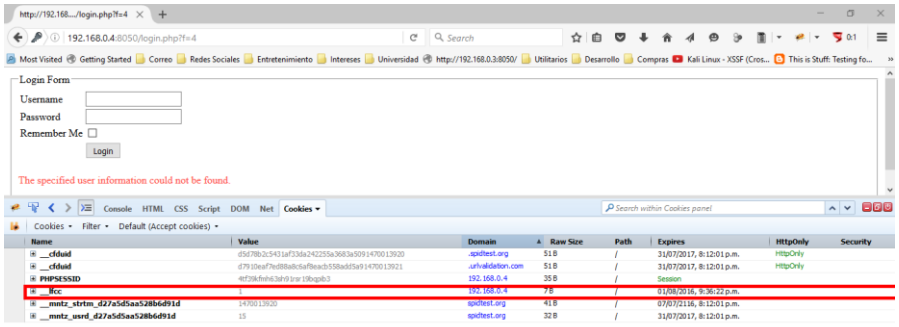
Figura. 135. Resultado de Sequencer de Burp Suite



Fuente: El autor. Screenshot de pantalla de Burpsuite.

b. ***Que tenga resistencia al sabotaje:*** Esta característica se refiere a la resistencia de la cookie a ser modificada. Y por otro lado coloca un ejemplo en el que NO se ***incluyan*** propiedades o indicadores de acceso o de permisos (por ejemplo un indicador IsAdmin para determinar si es administrador de la página) (OWASP, 2014). Para este caso, las dos cookies generadas no presentan este tipo de indicadores, incluso la cookie ***__lfcc*** mantiene su valor cuando hay un usuario inexistente intentando acceder, por lo cual esta característica se cumple en las cookies utilizadas en Quadodo.

Figura. 136. Cookie __lfcc después de un intento de acceso fallido en Quadodo



Fuente: El autor. Screenshot de pantalla de Burpsuite.

c. ***Expiración***: Esta característica debe ser aplicada a cookies críticas para que deban ser validadas y eliminadas solamente por un periodo determinado de tiempo. Cuando no se asigna una fecha determinada la

cookie existirá únicamente durante la sesión, es decir hasta que se cierre la ventana del navegador.

Para el caso de las cookies detectadas, el Identificador de sesión (PHPSESSID) solamente estará presente durante la sesión.

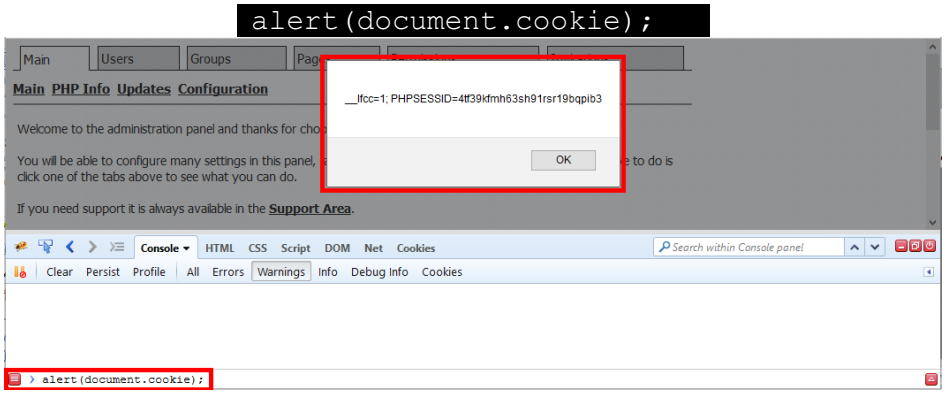
Por otro lado para la cookie se determina la fecha de expiración ***01/08/2016, 9:36:22 p.m.*** A pesar de no ser crítica esta cookie cumple con esta característica al igual que la anterior.

d. ***Propiedad "Secure" activada:*** En el detalle mostrado para las dos cookies identificadas se observa que la propiedad ***IsSecure*** a la cual se hace referencia están desactivadas. La misma es importante activarla para cookies con valor crítico como lo es el Id de sesión y permite trasmitir la información de la misma solamente por un canal encriptado. Sin embargo, hay que tener en cuenta que según el protocolo sobre el cual se despliegue acceda Quadodo (HTTP o HTTPS) se debe configurar correctamente esta propiedad.

Por otro lado, y adicional a lo anterior en la guía OWASP se indica que al menos el identificador de sesión tenga una longitud de 50 caracteres y que se use active la propiedad HTTPOnly (asignando a la misma el valor ***true***), respecto a esta última se detalla el análisis en Quadodo en el siguiente test (OTG-SESS-01).

Se accede a Quadodo y se ejecuta el siguiente script en la consola de Firebug, el cual retorna las cookies asignada al documento y las muestra en una alerta.

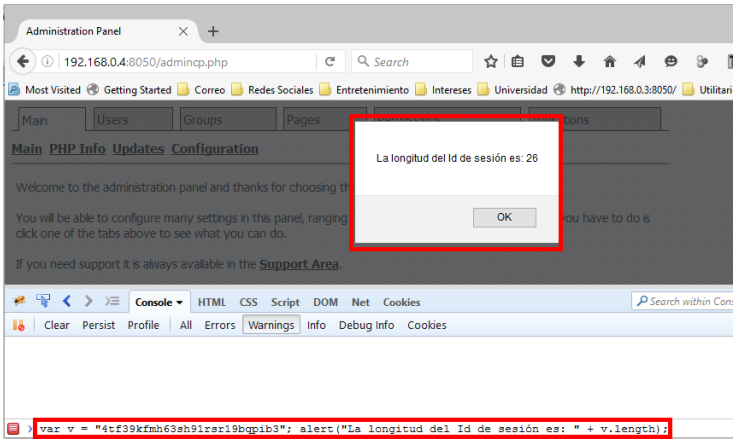
Figura. 137. Ejecución de script que muestra cookie del documento.



Fuente: El autor. Screenshot de pantalla de Firebug ejecutándose en Firefox.

A continuación se determina la longitud del Id de sesión copiando el valor de la cookie PHPSESSID y ejecutando el siguiente script en la consola de Firebug.

Figura. 138. Longitud de Id de sesión mostrada a través de código Javascript




```
var v = "4tf39kfmh63sh91rsr19bqpib3";
alert("La longitud del Id de sesión es: " + v.length);
```

Fuente: El autor. Screenshot de pantalla de Firebug ejecutándose en Firefox.

El resultado indica que el Identificador de sesión no cumple con el mínimo de caracteres recomendados para su longitud, así que es importante considerar la implementación de una opción de mejora que mitigue este aspecto.

11.7.1.1 Resultado

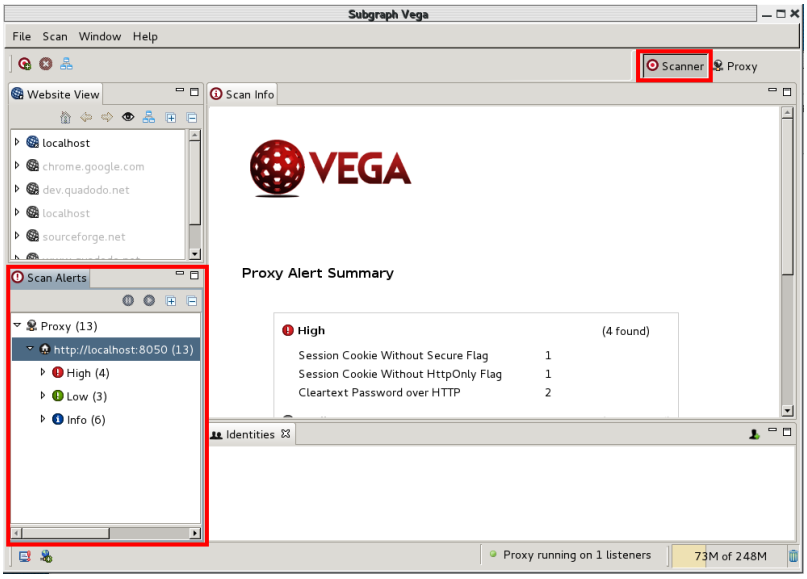
Quadodo presenta algunos puntos a mejorar en cuanto a las cookies, a pesar de solamente utilizar un par para su funcionamiento, entre las cuales está el identificador de sesión debe hacer más robusta esta validación, por ejemplo no limitándose con la que provee PHP para lograr ajustar a la longitud de caracteres recomendada, por supuesto atendiendo una encripción también robusta.

Por otro lado el acceso a las cookies fue posible a través de JavaScript lo cual supone una vulnerabilidad como se detalla en el test siguiente.

11.7.2 Testing for cookies attributes (OTG-SESS-002)

En las pruebas realizadas en los test aplicados del lado del cliente (Client-Side Testing) evidenciados en este documento, y al script ejecutado en el análisis anterior se logró identificar que fue posible acceder a la cookie del documento HTML a través de javascript. Volviendo a la herramienta Vega, se retoman los resultados obtenidos en las pruebas mencionadas y se accede a la opción ***Scanner*** la cual despliega el panel ***Scan Alerts***.

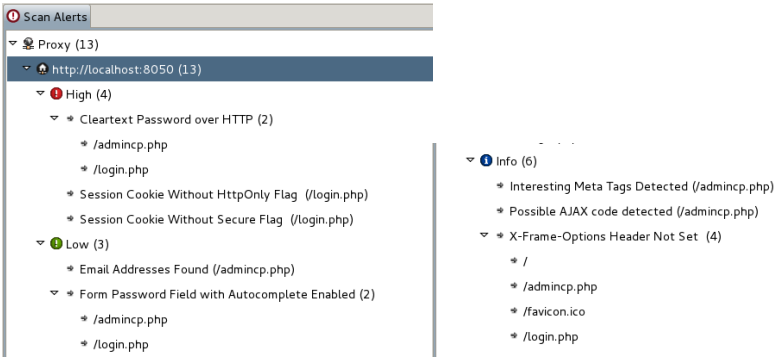
Figura. 139. Opción Scanner de la herramienta Vega y panel Scan Alerts



Fuente: El autor. Screenshot de pantalla de VEGA.

En dicho panel se listan algunas vulnerabilidades según la criticidad de las mismas como se muestra en la imagen a continuación. Son tres categorías: Alta (High), Baja (Low) e Informativa (Info) y para cada uno de los ítems mostrados bajo cada categoría, Vega suministra algunas recomendaciones para optimizar o dar solución a las vulnerabilidades halladas.

Figura. 140. Listado de vulnerabilidades y hallazgos panel Scan Alerts en la herramienta Vega



Fuente: El autor. Screenshot de pantalla de VEGA.

A continuación se describen las vulnerabilidades encontradas:

1. **Categoría Baja (Low):** Se muestra esta categoría aspectos relacionados con autocompletado de campos de formularios que contienen cuentas de correo electrónico y passwords, los cual tiene que ver con un comportamiento del lado del navegador web, por lo cual no se abarcará el tema ya que no es de aplicación.

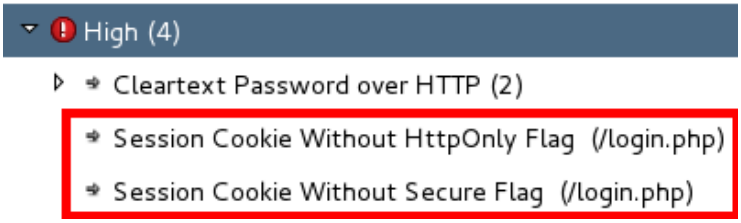
2. **Categoría Informativa (Info):** En esta categoría se listan etiquetas de interés en la cabecera, en este caso el nombre del autor, código AJAX detectado (el cual se explotó en la prueba del test OTG-CLIENT-004) y cabeceras de marcos no configuradas.

Figura. 141. Detalle de uno de las posibles vulnerabilidades en la categoría Informativa (Vega)

Fuente: El autor. Screenshot de pantalla de VEGA.

3. **Categoría Alta:** En esta categoría se hace especial énfasis. Según los ítems listados bajo la misma se indica una vulnerabilidad con referencia a Passwords con texto plano sobre HTTP (la cual fue detectada ya en el test OTG-IDENT-002) y dos relacionadas con las cookies de sesión, a la cual corresponde este test:

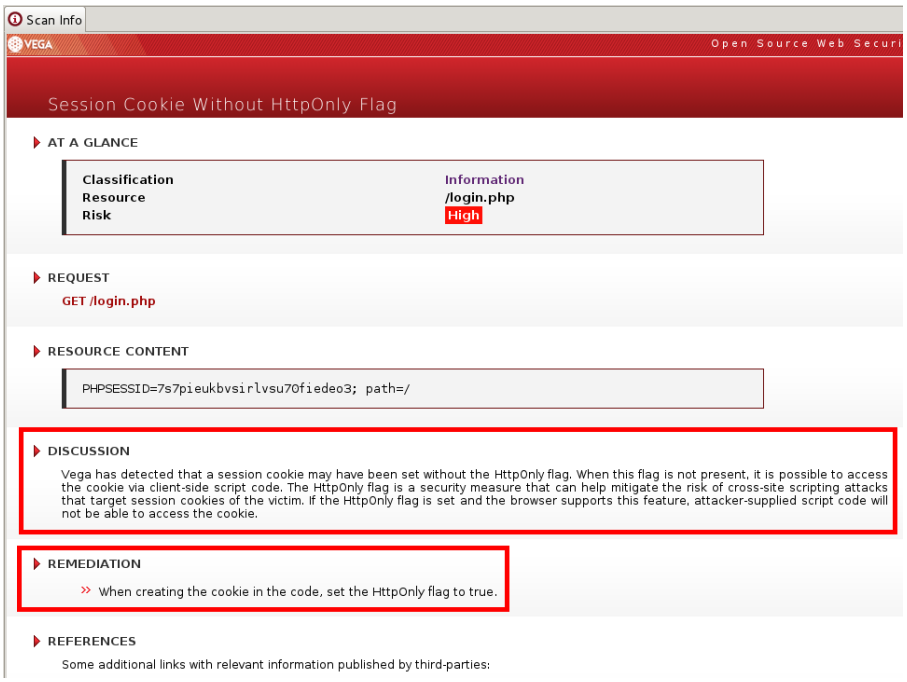
Figura. 142. Vulnerabilidades de Cookies detectadas mediante la herramienta Vega



Fuente: El autor. Screenshot de pantalla de VEGA.

1. La primera indica que no se ha asignado el flag o propiedad HttpOnly. Respecto a esta vulnerabilidad, la herramienta expone (bajo la sección **Discussion**) que cuando esta propiedad NO está presente es posible acceder a las cookies de la aplicación a través de un código del lado del cliente, como se identificó en la pruebas del test OTG-CLIENT-003. Si esta propiedad está presente puede ayudará a mitigar el riesgo de ataques de cross-site que use las cookies de sesión de la víctima.

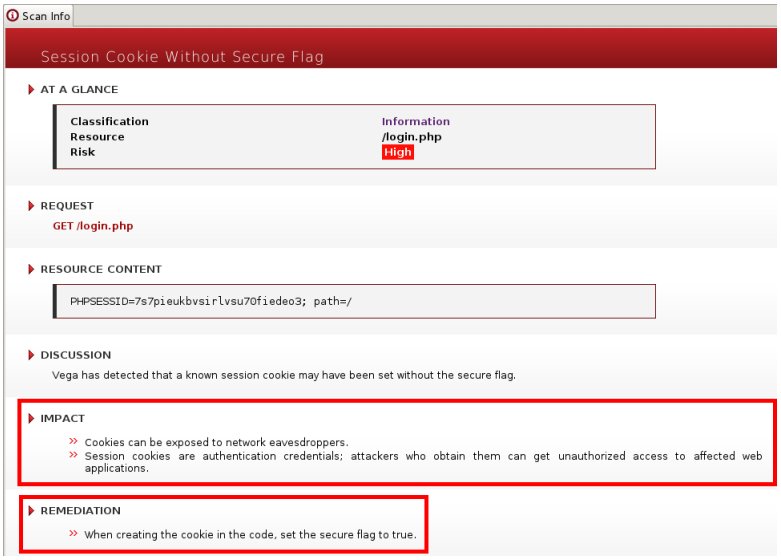
Figura. 143. Descripción y solución expuesta en la herramienta Vega para cookies sin propiedad HttpOnly



Fuente: El autor. Screenshot de pantalla de VEGA.

2. La segunda vulnerabilidad indicada en el cuadro señala que no se ha asignado la propiedad Secure a la cookie. Según lo indica la misma herramienta el impacto de esta vulnerabilidad implica que las cookies puedan ser expuestas a atacantes o vigilantes en la red. La solución, también indicada en la herramienta bajo la sección **Remediation**, es asignar el valor true a dicha propiedad.

Figura. 144.Descripción y solución expuesta en la herramienta Vega para cookies sin propiedad Secure



Fuente: El autor. Screenshot de pantalla de VEGA.

11.7.2.1 Resultado

Quadodo presenta dos vulnerabilidades correspondientes a las cookies, las cuales impactan el acceso a información sensible del manejo de sesión y de información del usuario. La primera vulnerabilidad indica que no se asigna el flag o propiedad HttpOnly y la segunda indica que no se ha asignado la propiedad Secure.

12. CONSOLIDACIÓN DE RESULTADOS Y GENERACIÓN DE REPORTE

Una vez realizadas las pruebas correspondientes, se procede a consolidar a continuación los resultados de cada una de ellas. Para ello y según lo señalado en el marco teórico se aplicará el formato CVE.

CVE (Common Vulnerabilities and Exposures) es un diccionario de nombres comunes como identificadores propios de CVE asignados a vulnerabilidades de seguridad detectadas en sistemas específicos que fue y que alimenta la base de datos nacional de los estados unidos (U.S. National Vulnerability Database [NVD]) (CVE, 2016). Este diccionario "provee puntos de referencia para el intercambio de datos con los cuales diferentes productos y servicios de seguridad se puedan entender", según lo expuesto en la página oficina de CVE (CVE, 2016).

Los identificadores CVE (CVE identifiers) cumplen un estándar o sintaxis especifico que se detalla en el sitio oficial de CVE. Estos identificadores cumplen la siguiente estructura:

CVE (prefijo) + Año + Dígitos Arbitrarios

Cada uno de las partes del identificador se separa con un guión **(-)**, los dígitos arbitrarios son 4 inicialmente, ya en caso de ser necesario pueden aumentarse hasta en 7 dígitos (CVE, 2015). La forma como deben asignarse es como se muestran en los ejemplos a continuación:

Figura. 145. Ejemplo de asignación de CVE identifiers (tomado de la página de CVE)

IDs with 4 digits	**IDs with 5 digits** (when needed)	**IDs with 6 digits** (when needed)	**IDs with 7 digits** (when needed)
CVE-2014-0001	CVE-2014-10000	CVE-2014-100000	CVE-2014-1000000
CVE-2014-3127	CVE-2014-54321	CVE-2014-456132	CVE-2014-7654321
CVE-2014-9999	CVE-2014-99999	CVE-2014-999999	CVE-2014-9999999

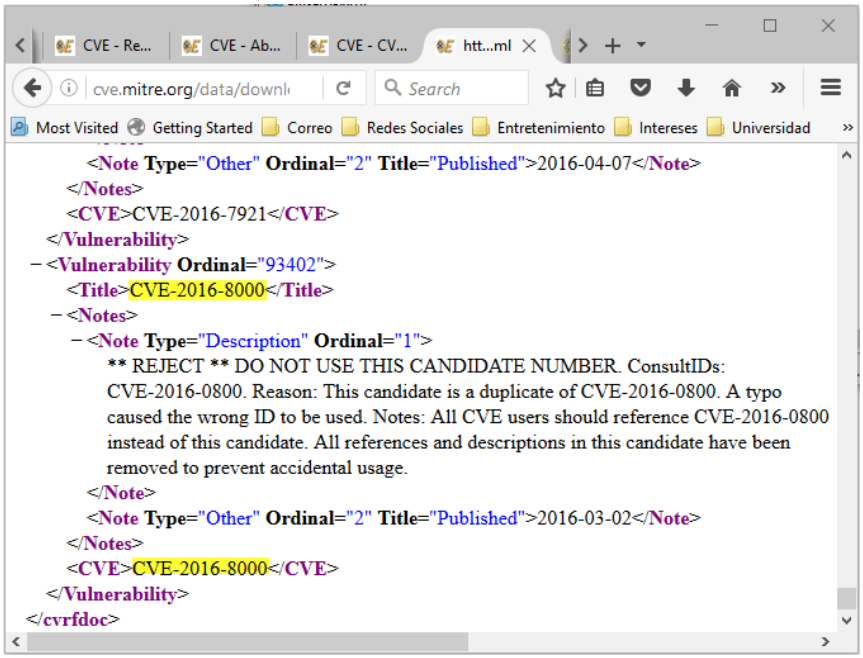
Fuente: CVE Common Vulnerabilities and Exposures. CVE ID Syntax Change.

Para obtener un identificador CVE es necesario realizar un proceso de solicitud descrito en la página de CVE, el cual indica verificar como prerrequisito que el producto o la fuente de datos sobre la cual se vaya a reportar la vulnerabilidad esté cubierta (CVE, 2016). Por ejemplo, CVE cubre productos de grandes empresas como Apple, Android, Cisco, entre otras. La URL donde pueden ser consultados los productos que CVE cubre es: http://cve.mitre.org/cve/data_sources_product_coverage.html

Ahora bien, ya que Quadodo es un proyecto Open Source hecho por un desarrollador particular (Douglas Rennehan) no figura entre los productos cubiertos por CVE para registrar vulnerabilidades, para estos casos CVE provee un método alternativo que consiste en contactarse a la dirección de correo electrónico cve-assign@mitre.org. Sin embargo, advierte en su página web que la asignación del identificador CVE depende de la disponibilidad de los analistas.

Por lo anterior, se aplicará el identificador CVE de forma independiente y para efectos de este proyecto tomando como referencia el último identificador generado para el año 2016 y que figura en el listado oficial de CVE, siguiendo la secuencia correspondiente. (CVE, 2016)

Figura. 146. Último CVE generado para el año 2016 al momento de realizar este proyecto



Fuente: CVE. CVE Output in CVRF 1.1: 20161110

12.1.1 Consolidación de datos en diccionario CVE

Se indica en el cuadro relacionado a continuación cada uno de los test ejecutados basados en la guía OWASP versión 4, con su correspondiente nombre y código, además de la herramienta utilizada para la ejecución. Cabe aclarar que algunas de las pruebas se realizaron de forma manual, es decir, sin ninguna herramienta automatizada alguna como lo sugería OWASP para algunos de estos casos.

Por otro lado, se asigna el correspondiente identificador del diccionario CVE solamente a los escenarios con resultado **FALLIDO**. Entiéndase como **FALLIDO** el estado asignado a los escenarios de prueba que tuvieron un resultado que expone una vulnerabilidad de seguridad en la aplicación. Por el contrario el estado **EXITOSO** es asignado a los escenarios de prueba que tuvieron como resultado una vulnerabilidad de seguridad no presente o no expuesta en la aplicación. El consolidado es el siguiente:

Tabla 5. Consolidación de resultados y CVE IDs asignados a vulnerabilidades encontradas

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
CVE-2016-8001	**Testing Identity Management**	OTG-IDENT-002	Test user registration process	LiveHTTPHeaders	**FALLIDO**	La información de los formularios de registro y de ingreso es enviada de forma clara y básica, permitiendo de esta manera ser monitoreada y capturada.
CVE-2016-8002	**Testing Identity Management**	OTG-IDENT-004	Testing for Account Enumeration and Guessable User Account	*Manual*	**FALLIDO**	Según los mensajes obtenidos para cada escenario, los mismos evidencian que la aplicación revela si el usuario ingresado existe o no en el sistema.
CVE-2016-8003	**Testing for Authentication Testing**	OTG-AUTHN-001	Testing for Credentials Transported over an Encrypted Channel	LiveHTTPHeaders	**FALLIDO**	Al interceptar las cabeceras enviadas en cada petición, las credenciales viajan en texto plano sin ningún tipo de encripción previa y por protocolo HTTP. Claro que esto se debe también a que a la configuración del sitio trabaja sobre este protocolo y no por HTTPS, sin embargo los datos pueden visualizarse a través del método GET y ser manipulados fácilmente para alterar cualquier información.

Fuente: El autor.

Tabla 2. (Continuación)

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
--	**Testing for Authentication Testing**	OTG-AUTHN-003	Testing for Weak lock out mechanism	*Manual*	**EXITOSO**	El número de intentos depende de la configuración establecido en el aplicativo por parte del administrador del sistema. Sin embargo, al tener esta funcionalidad, el script puede adaptar la limitación de intentos de accesos al sistema según la política que crea conveniente cada organización y administrador, dependerá que la primera esté fuertemente establecida estipulando el parámetro más conveniente.
--	**Testing for Authentication Testing**	OTG-AUTHN-004 - 1	Testing for Bypassing Authentication Schema *Direct page request (forced browsing)*	*Manual*	**EXITOSO**	Los resultados indican que al intentar acceder sin autenticación o con un usuario sin privilegios, el sistema evita el ingreso validando cada escenario, lo cual asegura que el mismo no acceda a opciones no autorizadas
--	**Testing for Authentication Testing**	OTG-AUTHN-004 - 2	Testing for Bypassing Authentication Schema *Session ID Prediction*	Burp Suite	**EXITOSO**	No se identifica un patrón de generación o por lo menos no una secuencia específica de generación de identificador de sesión, por lo cual se puede asegurar, también según el resultado arrojado en la herramienta Burp Suite que Quadodo no presenta un riesgo de seguridad informática en la generación del identificador de sesión que usa en la aplicación.

Tabla 2. (Continuación)

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
--	**Authorization Testing**	OTG-AUTHZ-001	Testing Directory traversal/file include	DotDotPwn	**EXITOSO**	No se encontró ninguna inclusión o acceso a recursos del sistema, realizando la ejecución en la herramienta DotDotPwn 50,9 minutos
--	**Authorization Testing**	OTG-AUTHZ-002	Testing for Bypassing Authorization Schema	Vega/Manual	**EXITOSO**	Quadodo cuenta con una validación de permisos sobre los recursos, es por ello que para cada acción a ejecutar valida el usuario y el rol del mismo. Cubriendo de esta manera el acceso o ejecución no autorizada sobre la ejecución. Por lo anterior y respondiendo a las incógnitas planteadas, no es posible acceder a funciones administrativas incluso cuando el usuario ha ingresado con privilegios estándar y como las acciones no pudieron ser utilizadas por dicha restricción, tampoco es posible usar funciones administrativas a través de un usuario para el cual la acción ha sido denegada.
--	**Authorization Testing**	OTG-AUTHZ-003	Testing for Privilege escalation	Webscarab	**EXITOSO**	Los resultados arrojados para esta prueba no revelan campos que expongan o almacenen la información del rol del usuario que pueda suponer una vulnerabilidad para que un atacante lo modifique para realizar un robo de sesión.

Tabla 2. (Continuación)

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
--	**Authorization Testing**	OTG-AUTHZ-004	Testing for Insecure Direct Object References	Manual	**EXITOSO**	La aplicación Quadodo tiene implementadas validaciones para las acciones que realiza el usuario, evitando que este realice operaciones para las cuales no tiene permisos asignados.
CVE-2016-8004	**Client-Side Testing**	OTG-CLIENT-001	Testing for DOM-based Cross site scripting	Vega/Firebug	**FALLIDO**	Es posible la manipulación del DOM en las opciones de menú.
CVE-2016-8005	**Client-Side Testing**	OTG-CLIENT-002	Testing for JavaScript Execution	Vega/Firebug	**FALLIDO**	El manejo de construcción y redirección de los enlaces del menú no es seguro, permitiendo la inyección de código Javascript no seguro.
CVE-2016-8006	**Client-Side Testing**	OTG-CLIENT-003	Testing for HTML Injection	Vega/Firebug	**FALLIDO**	La vulnerabilidad de inyección de código HTML está presente mediante la construcción de los menús que ya se ha indicado, por otro lado en la creación de grupos por ejemplo, es permitido guardarlos con código HTML, por lo cual se debe implementar una restricción con el fin de evitar este tipo de inyección. Se detecta sin embargo, que se puede establecer el formato de nombre de usuarios bajo la opción de configuración, así que puede contemplarse la misma configuración para la creación de grupos.

Tabla 2. (Continuación)

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
CVE-2016-8007	**Client-Side Testing**	OTG-CLIENT-004	Testing for Client Side URL Redirect	Vega/Firebug	**FALLIDO**	La vulnerabilidad de redirección de URL del lado del cliente está presente, así que es necesario que se implemente un mejor control de acceso en los enlaces para que la redirección o retorno de contenido de páginas externas sea validado con el fin de evitar cualquier riesgo de seguridad o robo de datos.
CVE-2016-8008	**Client-Side Testing**	OTG-CLIENT-005	Testing for CSS Injection	Vega/Firebug	**FALLIDO**	Fue posible realizar la inyección de CSS, por lo cual hace posible modificar la interfaz del usuario haciendo que este ejecute acciones sobre opciones que sean resaltadas para a su vez realizar acciones inesperadas.
CVE-2016-8009	**Input Validation Testing**	OTG-INPVAL-001	Testing for Reflected Cross site scripting	Vega/Firebug	**FALLIDO**	Esta vulnerabilidad, está presente en Quadodo teniendo en cuenta los resultados de las pruebas realizadas para los test Client-Side o del lado del cliente (OTG-CLIENT-001, OTG-CLIENT-002, OTG-CLIENT-003, OTG-CLIENT-004 y OTG-CLIENT-005). Esta vulnerabilidad también abarca e impacta lo relacionado con vulnerabilidades de validaciones de entrada (Input Validation).

Tabla 2. (Continuación)

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
--	**Client-Side Testing**	OTG-CLIENT-012	Test Local Storage	Firebug	**EXITOSO**	No se detectó almacenamiento local por parte de Quadodo, por lo cual no hay contenido que pueda ser usado y manipulado. Es importante tener en cuenta que "en promedio los Browsers permiten almacenar cerca de 5MB de almacenamiento local por dominio, lo cual comparado con los 4KB de las cookies es una gran diferencia" y que según lo indica el proyecto OWASP "los datos almacenados persisten después de que la ventana ha sido cerrada, por lo cual es una mala idea almacenar de esta manera datos sensibles o identificadores de sesión." Por lo anterior, Quadodo al no utilizar almacenamiento local disminuye el acceso o almacenamiento de datos e información sensible, lo cual es muy importante si va a ser integrado a otras aplicaciones para la administración de usuarios.
--	**Input Validation Testing**	OTG-INPVAL-005	Testing for SQL Injection	SQLMap	**EXITOSO**	El resultado arrojado por SQLMap en todos los casos se indicó que los parámetros enviados no son inyectables. La ejecución se realizó indicando el número máximo de nivel de proceso (5) y especificando incluso explícitamente el motor de base de datos. Sin embargo no fue posible ejecutar una inyección de SQL satisfactoria, por lo cual se evidencia que lo indicado por el autor respecto a esta protección a nivel de seguridad es verdadera.

Tabla 2. (Continuación)

CVE-ID	Campo de evaluación	Código	Test Name	Herramienta	Resultado	Descripción del resultado o de la vulnerabilidad
CVE-2016-8010	**Session Management Testing**	OTG-SESS-001	Testing for Session Management Schema	Firebug	**FALLIDO**	Quadodo presenta algunos puntos a mejorar en cuanto a las cookies, a pesar de solamente utilizar un par para su funcionamiento, entre las cuales está el identificador de sesión debe hacer más robusta esta validación.
CVE-2016-8011	**Session Management Testing**	OTG-SESS-002	Testing for cookies attributes	Vega/Firebug	**FALLIDO**	Quadodo presenta dos vulnerabilidades correspondientes a las cookies, las cuales impactan el acceso a información sensible del manejo de sesión y de información del usuario. La primera vulnerabilidad indica que no se asigna el flag o propiedad HttpOnly y la segunda indica que no se ha asignado la propiedad Secure.

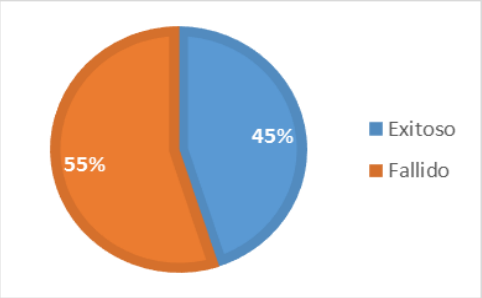
12.2 ANÁLISIS DE RESULTADOS

12.2.1 Resultados generales

El resumen general de resultados fallidos y exitosos es el siguiente:

Figura. 147. Resumen y gráfico de resultados generales

Resultado	Cantidad
Exitoso	9
Fallido	11
Total general	**20**



Fuente: El autor.

Quadodo presenta casi que un escenario dividido entre resultados fallidos y resultados exitosos. Es importante entonces que las primeras sean solucionadas, ya que Quadodo es un script hecho para ser reutilizado por aplicaciones que requieran de una funcionalidad que les provea administración de roles y de permisos.

12.2.2 Resultados obtenidos por campo de evaluación

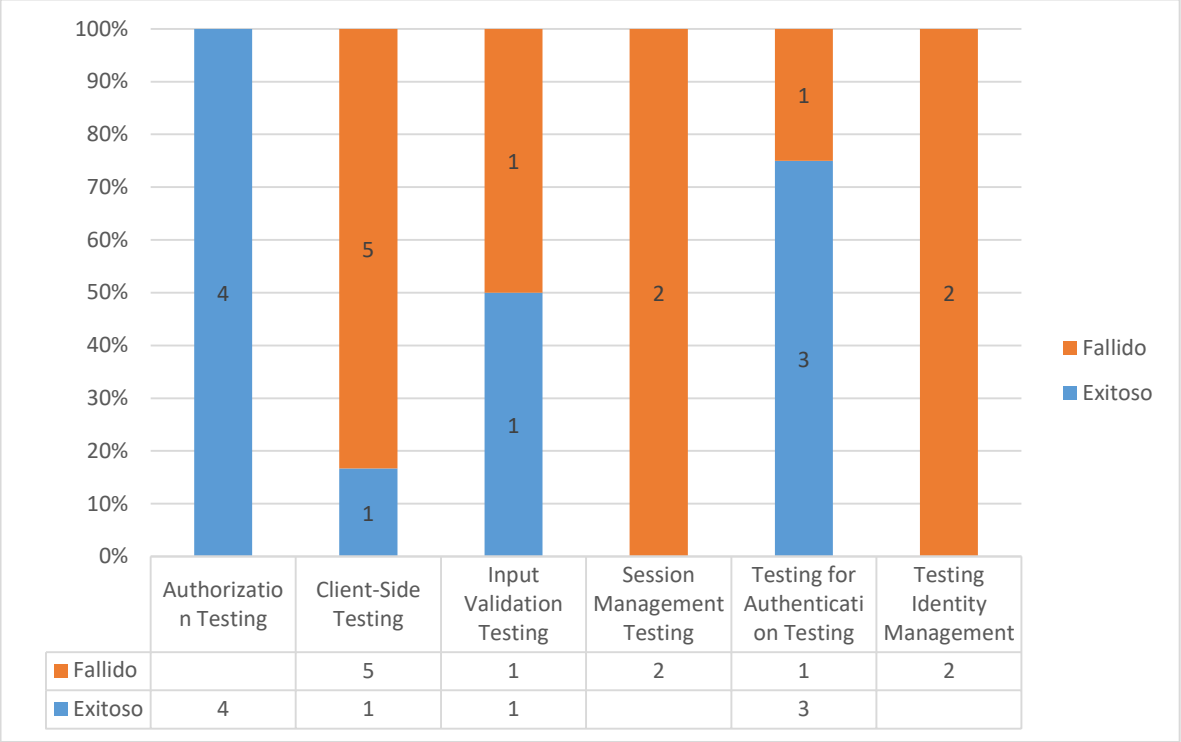
El resumen de los resultados exitosos y fallidos obtenidos en los test aplicados por cada uno de los campos de evaluación y que se consolidan en el cuadro anterior es el siguiente:

Tabla 6. Resumen resultados exitosos y fallidos de los test aplicados por campo de evaluación

Campo de evaluación	Exitoso	Fallido	Total general
Authorization Testing	4		4
Client-Side Testing	1	5	6
Input Validation Testing	1	1	2
Session Management Testing		2	2
Testing for Authentication Testing	3	1	4
Testing Identity Management		2	2
Total general	**9**	**11**	**20**

Fuente: El autor.

Figura. 148. Gráfico de resumen de resultados de test aplicados por campo de evaluación



Fuente: El autor.

Como se puede observar en el gráfico anterior, Quadodo presenta fortalezas en lo que a autorización de usuarios (Authorization) en la aplicación se refiere, teniendo resultados exitosos en las pruebas aplicadas dirigidas a la inclusión de archivos transversales, manejo del esquema de autorización, escalamiento de privilegios y referencias a objetos potencialmente inseguros.

Por otro lado y el siguiente campo en el que presenta fortalezas es en el de autenticación de usuarios (Authentication), aquí se hace un paréntesis aclarando que este campo es diferente a la autorización de usuarios. Al hablar de autenticación, se refiere al proceso o a la forma como el sistema verifica la identidad del usuario al acceder o al intentar acceder a la aplicación, en cambio al hablar de autorización se refiere a como el sistema valida y gestiona los accesos a recursos del sistema y los permisos asignados al usuario.

Continuando con el análisis y a lo que se refiere a autenticación de usuarios, Quadodo presenta fortalezas en el esquema de autenticación implementado el cual fue revisado a través de las pruebas específicas de acceso forzoso a páginas directas y predicción del identificador de sesión, adicional a esto Quadodo permite parametrizar el número de intentos de accesos fallidos a la aplicación, por lo cual

esto debe ser determinado cuidadosamente por el usuario administrador de la aplicación.

Sin embargo, presenta no conformidades principalmente en el campo de manejo de identidad del usuario teniendo vulnerabilidades en el proceso de registro del usuario en la aplicación y en la verificación de usuarios existentes en la aplicación; en el campo del manejo de sesión donde fue posible acceder a los valores de las cookies correspondientes donde además se identificó la ausencia de atributos de seguridad importantes para evitar el robo de sesión.

Pero, uno de los campos en los cuales se realizaron cinco pruebas con resultados fallidos sobre seis en total, fue el correspondiente al lado del cliente (Client-Side) donde los scripts implementados del lado de la interfaz del usuario presentaron una no conformidad mayor al vulnerar incluso los datos del sistema, permitiendo inyectar código malicioso para que el usuario realizara acciones de alto impacto sin intención.

12.3 GESTIÓN DE LOS RIESGOS GENERADOS APLICANDO LAS NORMAS ISO 31000:2009 E ISO/IEC 31010:2009

La norma ISO 31000 del año 2009 se enfoca en el manejo del riesgo, por lo cual “provee principios, marcos de trabajo y un proceso de manejo de riesgos.” (ISO, 2009). Es usada en cualquier tipo de empresa u organización para evaluar los riesgos que puedan estar presentes en las mismas para generar acciones de mejora o planes de acción que ayuden a mitigarlos y/o eliminarlos.

Como parte de la gestión de riesgos la norma indicada establece entre sus procesos cuatro procesos para poder realizar dicha gestión los cuales son (Meléndez González, 2015):

1. Identificación de los riesgos
2. Análisis de los riesgos
3. Evaluación de riesgos
4. Tratamiento de los riesgos

Ahora bien, haciendo un paréntesis en este punto lo que se ha evaluado en la presente monografía no es un proceso organizacional sino un componente específico de software que puede ser integrado a otras aplicaciones para proveer funcionalidades de administración de usuarios y accesos. Sin embargo, dicha norma puede ser aplicada a este contexto ya que cada vulnerabilidad encontrada supone un riesgo hacia la seguridad informática del sistema que implemente el componente.

Cerrando el paréntesis anterior y continuando entonces con la aplicación de la norma ISO 31000:2009 en lo que concierne a los procesos de gestión de riesgos,

se debe hablar también de la norma ISO/IEC 31010:2009. Esta última sirve como apoyo a la primera, proveyendo artefactos y técnicas para realizar la evaluación de riesgos con el objetivo que los mismos sean claros para la toma de decisiones y para la realización de acciones de mejora o planes de acción que logren minimizarlos, prevenirlos o solucionarlos (ISO, 2009).

12.3.1 Identificación de los riesgos

A continuación se relaciona un riesgo a cada vulnerabilidad encontrada (es decir, se relaciona solamente a los resultados que se indicaron como Fallidos):

Tabla 7. Matriz de riesgos asociados a las vulnerabilidades

CVE-ID	Campo de evaluación	Código Test OWASP	Test Name	Descripción de la vulnerabilidad (Causa de Riesgo)	Riesgo(s)
CVE-2016-8001	**Testing Identity Management**	OTG-IDENT-002	Test user registration process	La información de los formularios de registro y de ingreso es enviada de forma clara y básica, permitiendo de esta manera ser monitoreada y capturada.	- Intercepción de credenciales - Robo de credenciales
CVE-2016-8002	**Testing Identity Management**	OTG-IDENT-004	Testing for Account Enumeration and Guessable User Account	Según los mensajes obtenidos para cada escenario, los mismos evidencian que la aplicación revela si el usuario ingresado existe o no en el sistema.	- Identificación usuarios existentes
CVE-2016-8003	**Testing for Authenticati on Testing**	OTG-AUTHN-001	Testing for Credentials Transported over an Encrypted Channel	Al interceptar las cabeceras enviadas en cada petición, las credenciales viajan en texto plano sin ningún tipo de encripción previa y por protocolo HTTP. Claro que esto se debe también a que a la configuración del sitio trabaja sobre este protocolo y no por HTTPS, sin embargo los datos pueden visualizarse a través del método GET y ser manipulados fácilmente para alterar cualquier información.	- Intercepción de credenciales - Robo de credenciales

Fuente: El autor.

Tabla 4. (Continuación)

CVE-ID	Campo de evaluación	Código Test OWASP	Test Name	Descripción de la vulnerabilidad (Causa de Riesgo)	Riesgo(s)
CVE-2016-8004	**Client-Side Testing**	OTG-CLIENT-001	Testing for DOM-based Cross site scripting	Es posible la manipulación del DOM en las opciones de menú.	- Manipulación de controles - Ejecución de flujos no deseados
CVE-2016-8005	**Client-Side Testing**	OTG-CLIENT-002	Testing for JavaScript Execution	El manejo de construcción y redirección de los enlaces del menú no es seguro, permitiendo la inyección de código Javascript no seguro	- Ejecución de flujos no deseados - Edición y borrado de información
CVE-2016-8006	**Client-Side Testing**	OTG-CLIENT-003	Testing for HTML Injection	La vulnerabilidad de inyección de código HTML está presente en la aplicación mediante la construcción de los menús que ya se ha indicado, por otro lado en la creación de grupos por ejemplo, a pesar de que se codifican los caracteres especiales que se ingresan en el nombre es permitido guardarlos con dichos caracteres, por lo cual se debería implementar una restricción para ello con el fin de disminuir la inyección de HTML. Se detecta sin embargo que se puede establecer el formato de nombre de usuarios bajo la opción de configuración, así que podría contemplarse la misma configuración para la creación de grupos.	- Manipulación de controles - Ejecución de flujos no deseados:

Fuente: El autor.

Tabla 4. (Continuación)

CVE-ID	Campo de evaluación	Código Test OWASP	Test Name	Descripción de la vulnerabilidad (Causa de Riesgo)	Riesgo(s)
CVE-2016-8007	**Client-Side Testing**	OTG-CLIENT-004	Testing for Client Side URL Redirect	La vulnerabilidad de redirección de URL del lado del cliente está presente, así que es necesario que se implemente un mejor control de acceso en los enlaces para que la redirección o retorno de contenido de páginas externas sea validado con el fin de evitar cualquier riesgo de seguridad o robo de datos.	- Redirección a sitios no deseados:
CVE-2016-8008	**Client-Side Testing**	OTG-CLIENT-005	Testing for CSS Injection	Fue posible realizar la inyección de CSS, por lo cual hace posible modificar la interfaz del usuario haciendo que este ejecute acciones sobre opciones que sean resaltadas para a su vez realizar acciones inesperadas.	- Cambio en la presentación de controles
CVE-2016-8009	**Input Validation Testing**	OTG-INPVAL-001	Testing for Reflected Cross site scripting	Esta vulnerabilidad, está presente en Quadodo teniendo en cuenta los resultados de las pruebas realizadas para los test Client-Side o del lado del cliente (OTG-CLIENT-001, OTG-CLIENT-002, OTG-CLIENT-003, OTG-CLIENT-004 y OTG-CLIENT-005). esta vulnerabilidad también abarca e impacta lo relacionado con vulnerabilidades de validaciones de entrada (Input Validation).	- Redirección a sitios no deseados - Robo de información:

Fuente: El autor.

Tabla 4. (Continuación)

CVE-ID	Campo de evaluación	Código Test OWASP	Test Name	Descripción de la vulnerabilidad (Causa de Riesgo)	Riesgo(s)
CVE-2016-8010	**Session Management Testing**	OTG-SESS-001	Testing for Session Management Schema	Quadodo presenta algunos puntos a mejorar en cuanto a las cookies, a pesar de solamente utilizar un par para su funcionamiento, entre las cuales está el identificador de sesión debe hacer más robusta esta validación.	- Robo de sesión
CVE-2016-8011	**Session Management Testing**	OTG-SESS-002	Testing for cookies attributes	Quadodo presenta dos vulnerabilidades correspondientes a las cookies, las cuales impactan el acceso a información sensible del manejo de sesión y de información del usuario. La primera vulnerabilidad indica que no se asigna el flag o propiedad HttpOnly y la segunda indica que no se ha asignado la propiedad Secure.	- Lectura de cookies - Manipulación de cookies

Fuente: El autor.

12.3.2 Análisis de Riesgos

Una vez identificados los riesgos generados a partir de las vulnerabilidades encontradas se procede a realizar un análisis del impacto que tienen dichos riesgos y a categorizarlos para dar una prioridad en la gestión de los mismos bien sea para minimizarlos o erradicarlos. Esto tiene como objetivo realizar toma de decisiones y consecuentes acciones de mejora o planes de acción para mitigarlos.

Para el análisis indicado la norma ISO/IEC 31010:2009 provee diversas técnicas y herramientas que permiten realizar este proceso, como los que se muestran en la siguiente imagen (Meléndez González, 2015):

Figura. 149. Técnicas de evaluación y riesgos.

TÉNCICAS DE EVALUACIÓN DE RIESGOS					
MÉTODOS DE CONSULTA	MÉTODOS DE SOPORTE	ANÁLISIS DE ESCENARIOS	ANÁLISIS DE FUNCIÓN	EVALUACIÓN DE CONTROLES	MÉTODOS ESTADÍSTICOS
1. Check – List 2. Análisis Preliminar de riesgos 3. Listas de ejemplos	4. Lluvia de ideas 5. Entrevista estructurada o semi-estructurada 6. Técnica Delphi 7. Técnica estructurada What if? (SWIFT) 8. Evaluación de la fiabilidad humana (HRA) 9. Análisis de Riesgos Preliminar	10. Análisis Causa Raíz (RCA) 11. Evaluación de Toxicidad 12. Análisis de Impacto al negocio (BIA) 13. Análisis de árbol de fallas (FTA) 14. Análisis de árbol de acontecimientos (ETA) 15. Análisis de causa – consecuencia 16. Análisis causa – efecto	17. Análisis de modo de fallos y efectos (AMEF – FMEA) 18. Fiabilidad de centro de mantenimiento (RCM) 19. Análisis de errores de diseño (Sneak) 20. Análisis de Peligros de Operabilidad (HAZOP) 21. Análisis de Peligros y Puntos Críticos de Control (HCCAP)	22. Análisis de capas de protección (LOPA) 23. Análisis de fallos y sucesos iniciadores (Bow Tie) 24. Análisis de circuitos de fugas	25. Análisis Markov 26. Simulación Monte Carlo 27. Estadística y redes Bayesianas 28. Curvas FN 29. Índices de Riesgo 30. Matrices de probabilidad y consecuencia 31. Análisis de decisión multi-criterio (MCDA)

Fuente: (Meléndez González, 2015). ISO/IEC 31010: Gestión de riesgos –Técnicas de evaluación de riesgos

Para este caso se aplican los siguientes:

1. ***Análisis preliminar de riesgos (Método de Consulta):*** Tomando como base la matriz de identificación de riesgos aplicada en el punto anterior se realiza un análisis preliminar de riesgos que permite identificar las consecuencias y generar recomendaciones preventivas o correctivas a cada uno, también se asigna un nivel de impacto cuantitativo.

2. ***Matrices de probabilidad y consecuencia (Método estadístico):*** Este método es útil para establecer un perfil de riesgo, por medio del cual se mide el impacto y por ende se establece una prioridad según la probabilidad de que un riesgo ocurra y la consecuencia que puede tener en este caso sobre un sistema que implemente Quadodo, todo esto basado en el método de análisis preliminar anteriormente indicado.

12.3.2.1 Análisis preliminar de riesgos (Método de Consulta):

Se aplica la siguiente matriz para este proceso:

Tabla 8. Tabla de análisis preliminar de riesgos (Descripción de campos)

Descripción del riesgo	Causa	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia
Describe el riesgo encontrado	Indica la causa del riesgo, corresponde a las vulnerabilidades detectadas, las cuales se identifican con el Id de diccionario CVE asignado a cada una.	Describe el impacto del riesgo en el sistema	Se establecen tres niveles de impacto: ***1 – Bajo:*** El riesgo debe gestionarse, pero puede establecerse una opción de mejora en un release programado de la aplicación. ***2 – Medio:*** El riesgo está latente y puede convertirse en un riesgo alto, por ahora solamente identifica acciones que pueden ser usadas para ataques más dañinos e impactantes. ***3 - Alto:*** Requiere una solución pronta ya que inevitablemente genera un impacto negativo en la información del sistema.	Relaciona la probabilidad que un riesgo ocurra. Para ello se establecen tres niveles: ***A – Raro***: No es frecuente que se presente y solo para casos fortuitos y por ende no es usado por atacantes como primera medida para atacar. ***B – Probable***: Sucede en situaciones específicas o en algunos escenarios haciendo que el atacante ejecute pasos adicionales para aprovecharlo. ***C – Probable***: Siempre se presenta y/o puede ser fácilmente identificado por un atacante.

Fuente: El autor.

Inicialmente se asigna un nivel de impacto y probabilidad de ocurrencia para definir el perfil de riesgo y se le asigna a cada riesgo un número precedido con la letra R a modo de identificador:

Tabla 9. Mariz de riesgos y consecuencias

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia
R01	Intercepción de credenciales	CVE-2016-8001 CVE-2016-8003	Identificar las claves de acceso al sistema de los diferentes usuarios que ingresan y hacen uso del mismo.	**3 - Alto**	**C – Probable**
R02	Robo de credenciales	CVE-2016-8001 CVE-2016-8003	Suplantación de identidad de usuarios para manipulación indebida de información	**3 - Alto**	**C – Probable**
R03	Identificación usuarios existentes	CVE-2016-8002	Permite a terceros identificar los usuarios que existen en la aplicación y por ende realizar ataques con identidades de usuario específicas que le permitan acceder a la aplicación.	**2 - Medio**	**C – Probable**
R04	Manipulación de controles	CVE-2016-8004 CVE-2016-8006	Cambiar el flujo o el comportamiento de la página inyectando código que puede hacer que el usuario ejecute acciones sin intención o envíe datos no deseados que puedan afectar la información.	**2 - Medio**	**B – Posible**
R05	Ejecución de flujos no deseados	CVE-2016-8004 CVE-2016-8005 CVE-2016-8006	Manipulación de información sensible y atribución de acciones no deseadas dentro del sistema a usuarios autorizados.	**3 - Alto**	**B – Posible**
R06	Borrado de información	CVE-2016-8005	Quitar privilegios, eliminación de usuarios administradores, denegación y monopolización de acceso por parte de un atacante.	**3 - Alto**	**B – Posible**

Fuente: El autor.

Tabla 6. (Continuación)

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia
R07	Redirección a sitios no deseados	CVE-2016-8007 CVE-2016-8009	Transporte de información de la aplicación a sitios externos y redireccionamiento a sitios maliciosos con miras al robo de información o infección con virus informáticos.	**2 - Medio**	**B – Posible**
R08	Cambio en la presentación de controles	CVE-2016-8008	Un atacante puede guiar al usuario a opciones con comportamientos no deseados combinando ataques de inyección de código.	**1 - Bajo**	**B – Posible**
R09	Robo de información	CVE-2016-8009	Es posible almacenar la información para usarla en ataques posteriores o identificar usuarios existentes en el sistema que se conviertan en victimas por parte de terceros según su rol.	**3 - Alto**	**B – Posible**
R10	Robo de sesión	CVE-2016-8010	Permite detectar que se está usando el Identificador de sesión de PHP, un atacante podría robar dicho Id y suplantar al usuario.	**3 - Alto**	**C – Probable**
R11	Lectura de cookies	CVE-2016-8011	Permite que un atacante pueda tener acceso inmediato a la información que contiene las cookies y que por lo tanto pueda hacer uso de ella. Una ventaja, es que además del Id de sesión no hay información sensible almacenada en las mismas como claves o perfiles.	**2 - Medio**	**C – Probable**
R11	Lectura de cookies	CVE-2016-8011	Permite que un atacante pueda tener acceso inmediato a la información que contiene las cookies y que por lo tanto pueda hacer uso de ella. Una ventaja, es que además del Id de sesión se detecta que no hay información sensible almacenada en las mismas como claves o perfiles. Es por esto que se indica un impacto medio.	**2 - Medio**	**C – Probable**

Fuente: El autor.

Tabla 6. (Continuación)

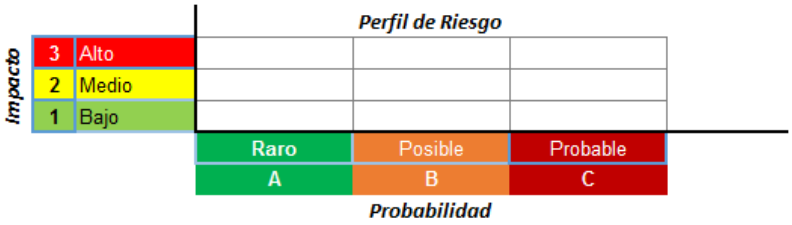
Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia
R12	Manipulación de cookies	CVE-2016-8011	El atacante puede identificar el nombre de cada una de las cookies y la información que almacenan. Ya dependiendo del criterio del atacante, puede hallar útil o no manipular la información contenida en las mismas para realizar un ataque.	**2 - Medio**	**B – Posible**

Fuente: El autor.

12.3.2.2 Matrices de probabilidad y consecuencia (Método estadístico)

Ya identificados a partir del análisis preliminar los niveles de impacto y la probabilidad de ocurrencia para cada riesgo, se procede a realizar el perfil de los riesgos identificados, usando la siguiente matriz, la cual tiene las siguientes características:

Figura. 150. Matriz de perfil de riesgo



Fuente: El autor.

El perfil de riesgo responderá a los siguientes resultados entre impacto y probabilidad del riesgo:

Figura. 151. Definición de perfil de riesgo según impacto y probabilidad del riesgo

Fuente: El autor

Tomando en cuenta lo anterior, serán considerados como riesgos a gestionar de inmediato los que se encuentren en la franja de la matriz que se señala en la siguiente imagen y que corresponden por lo tanto a los riesgos con perfil Moderado, Importante e Intolerable:

Figura. 152. Matriz de perfil de riesgo - franja de riesgos de gestión inmediata

Fuente: El autor.

Los resultados entonces son los siguientes teniendo en cuenta los parámetros indicados:

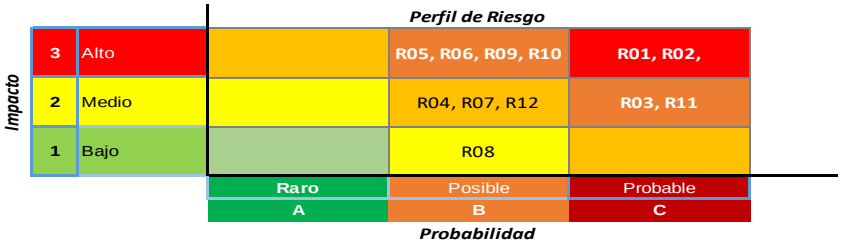
Tabla 10. Relación de riesgos encontrados con su respectivo perfil de riesgo

Id	Riesgo	Causas (CVE-ID)	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R01	Intercepción de credenciales	CVE-2016-8001 CVE-2016-8003	**3 - Alto**	**C – Probable**	**Intolerable**
R02	Robo de credenciales	CVE-2016-8001 CVE-2016-8003	**3 - Alto**	**C – Probable**	**Intolerable**
R03	Identificación usuarios existentes	CVE-2016-8002	**2 - Medio**	**C – Probable**	**Importante**
R04	Manipulación de controles	CVE-2016-8004 CVE-2016-8006	**2 - Medio**	**B – Posible**	**Moderado**
R05	Ejecución de flujos no deseados	CVE-2016-8004 CVE-2016-8005 CVE-2016-8006	**3 - Alto**	**B – Posible**	**Importante**
R06	Borrado de información	CVE-2016-8005	**3 - Alto**	**B – Posible**	**Importante**
R07	Redirección a sitios no deseados	CVE-2016-8007 CVE-2016-8009	**2 - Medio**	**B – Posible**	**Moderado**
R08	Cambio en la presentación de controles	CVE-2016-8008	**1 - Bajo**	**B – Posible**	**Tolerable**
R09	Robo de información	CVE-2016-8009	**3 - Alto**	**B – Posible**	**Importante**
R10	Robo de sesión	CVE-2016-8010	**3 - Alto**	**C – Probable**	**Importante**
R11	Lectura de cookies	CVE-2016-8011	**2 - Medio**	**C – Probable**	**Importante**
R12	Manipulación de cookies	CVE-2016-8011	**2 - Medio**	**B – Posible**	**Moderado**

Fuente: El autor.

Al transferirlos a la matriz de riesgo, se refleja el resultado de la siguiente manera:

Figura. 153. Resultado de perfil de riesgo



Fuente: El autor.

Como se puede observar once de los doce riesgos encontrados se ubican en la franja de gestión inmediata debido a su probabilidad de ocurrencia y su impacto en el sistema. Este resultado presenta a Quadodo como un script con riesgos de seguridad para el sistema que lo implemente, ya que las vulnerabilidades encontradas en el mismo a través de las pruebas realizadas no solamente tienen una alta probabilidad de ocurrencia sino que además pueden generar, por atacantes que las aprovechen, un gran impacto negativo afectando la información del sistema.

12.4 RECOMENDACIONES PARA EL TRATAMIENTO DE LOS RIESGOS Y LAS VULNERABILIDADES ENCONTRADAS

Siguiendo con la matriz planteada en el punto anterior en el análisis preliminar de riesgos, se procede a realizar para cada uno de los riesgos y las vulnerabilidades asociadas el planteamiento de medidas correctivas.

12.4.1 Acciones correctivas para los riesgos R01 y R02

Tabla 11. Riesgos R01 y R02

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R01	Intercepción de credenciales	CVE-2016-8001 CVE-2016-8003	Identificar las claves de acceso al sistema de los diferentes usuarios que ingresan y hacen uso del mismo.	**3 - Alto**	**C – Probable**	**Intolerable**
R02	Robo de credenciales	CVE-2016-8001 CVE-2016-8003	Suplantación de identidad de usuarios para manipulación indebida de información	**3 - Alto**	**C – Probable**	**Intolerable**

Fuente: El autor.

Estos dos riesgos están relacionados por las mismas causas por lo cual realizar las acciones correctivas correspondientes puede mitigar los dos. Para este caso, se detectó en las pruebas que es posible identificar en claro las credenciales de acceso de los usuarios, así como los datos que se envían para la creación de usuarios y grupos. Aunque, por ejemplo el envío de las credenciales de acceso se envían por método POST no es suficiente si viaja a través de protocolo HTTP, es por esto que es necesario que esta información viaje encriptada, para ello es indispensable el uso de HTTPS ya que al usar el estándar de seguridad SSL, permite establecer conexiones de confianza y canales seguros para que la información sea transmitida, evitando de esta manera que la misma pueda ser facilmente identificada en ataques "man-in-the-middle" donde por medio de sniffers la información sea interceptada por atacantes.

Ahora bien, también pueden implementarse algunos métodos de prevención desde el navegador, como por ejemplo el deshabilitar la función de desarrollo que permite visualizar las peticiones (Request) y respuestas (Response) que se realizan desde las páginas, esto por ejemplo lo realiza actualmente la página de Bancolombia. También es posible considerar por ejemplo sumar variables a los datos de autenticación de usuario como tokens de seguridad, donde se combine información cifrada de la IP que realiza la petición y el navegador usado por el usuario.

12.4.2 Acciones correctivas para el Riesgo R03

Tabla 12. Riesgo R03

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R03	Identificación usuarios existentes	CVE-2016-8002	Permite a terceros identificar los usuarios que existen en la aplicación y por ende realizar ataques con identidades de usuario específicas que le permitan acceder a la aplicación.	**2 - Medio**	**C – Probable**	**Importante**

Fuente: El autor.

Debido a que los mensajes que muestra Quadodo al presentarse un error de credenciales de acceso son dicientes al revelar si es el nombre del usuario o el password el que se ha ingresado de forma incorrecta o si el usuario ingresado se encuentra en base de datos, se deben modificar los mismos por mensajes genéricos que no revelen dicha información como “***Credenciales incorrectas***”, “***Usuario o password incorrectos***” o "***Autenticación incorrecta***", lo cual hará que un atacante

tenga incertidumbre acerca de las credenciales que intente usar. Esto según lo recomendado por OWASP (OWASP, 2014)

12.4.3 Acciones correctivas para el Riesgo R04

Tabla 13. Riesgo R04

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R04	Manipulación de controles	CVE-2016-8004 CVE-2016-8006	Cambiar el flujo o el comportamiento de la página inyectando código que puede hacer que el usuario ejecute acciones sin intención o envíe datos no deseados que puedan afectar la información.	**2 - Medio**	**B – Posible**	**Moderado**

Fuente: El autor.

El hecho que los componentes del DOM puedan ser manipulados da a lugar a inyecciones de código o comportamiento.

En el resultado del test asociado a la causa indicada, se observó que se mitiga una parte codificando los caracteres ingresados para el nombre del usuario y para el nombre de los grupos.

Sin embargo, la forma como se dinamiza la construcción del menú pudo ser manipulada ajustando un par de líneas de código del lado del navegador.

Como solución propuesta para mitigar este riesgo, se puede validar a nivel de código en la lógica de la aplicación, que la misma opción que se haya asignado sea la misma que el usuario utilice. Para esto el desarrollador se puede valer de las variables de sesión, realizando por supuesto la configuración apropiada como se indica más adelante para los riesgos R11 y R12.

12.4.4 Acciones correctivas para los riesgos R05 y R06

Tabla 14. Riesgos R05 y R06

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R05	Ejecución de flujos no deseados	CVE-2016-8004 CVE-2016-8005 CVE-2016-8006	Manipulación de información sensible y atribución de acciones no deseadas dentro del sistema a usuarios autorizados.	**3 - Alto**	**B – Posible**	**Importante**

Fuente: El autor.

Tabla 11. (Continuación)

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R06	Borrado de información	CVE-2016-8005	Quitar privilegios, eliminación de usuarios administradores, denegación y monopolización de acceso por parte de un atacante.	**3 - Alto**	**B – Posible**	**Importante**

Fuente: El autor.

Estos riesgos tienen relación con el anterior e incluso comparten las mismas causas. Sin embargo, este tiene un mayor impacto debido a su afectación a la información del sistema.

Para dar solución se debe mitigar el riesgo anterior con las medidas correctivas indicadas, adicional a ello y para estos riesgos específicos es indispensable que el usuario deba seguir un flujo específico para llegar, acceder y ejecutar una opción específica, esto con el fin que no se haga de forma directa como el llamado de una función de javascript específica. Por ejemplo para el borrado o edición de la información de un usuario en Quadodo, detectar además de los privilegios pertinentes del usuario si el usuario primero accedió a la página de listado de usuarios, hizo click en la opción de eliminación o edición correspondiente, etc.

Por otro lado, se debe revisar que las funciones de borrado o actualización de información no estén expuestas en el código javascript o sean fácilmente identificadas del lado del cliente.

12.4.5 Acciones correctivas para el riesgo R07

Tabla 15. Riesgo R07

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R07	Redirección a sitios no deseados	CVE-2016-8007 CVE-2016-8009	Transporte de información de la aplicación a sitios externos y redireccionamiento a sitios maliciosos con miras al robo de información o infección con virus informáticos.	**2 - Medio**	**B – Posible**	**Moderado**

Fuente: El autor.

Este riesgo, además de las consecuencias indicadas en la columna correspondiente puede desencadenar el riesgo R06 de borrado de información expuesto anteriormente ya que la redirección puede realizarse dentro de páginas validas dentro del sitio que ejecuten acciones sobre la información como borrado, creación o actualización y asignación de permisos.

Un ataque que puede realizar un atacante aprovechando esta vulnerabilidad es la falsificación de petición de sitios cruzados (CSRF por sus siglas en inglés) el cual incluyó OWASP en su Top 10 de vulnerabilidades del año 2007. OWASP indica que este ataque “fuerza el navegador validado de una víctima a enviar una petición a una aplicación Web vulnerable, la cual entonces realiza la acción elegida a través de la víctima” (OWASP, 2008)

Como medidas de protección, OWASP expone las siguientes como parte del artículo expuesto para esta vulnerabilidad:

1. **Asegurarse que no existen vulnerabilidades XSS en su aplicación:** Para ello es necesario que los riesgos anteriores sean mitigados, ya que la ejecución de flujos no deseados, la manipulación de controles y todo lo relacionado con la intercepción de las credenciales de usuario pueden convertirse en ataques XSS.

2. **Introducir testigos aleatorios "a la medida" en cada formulario y URL que no sean automáticamente presentados por el navegador:** Esto es proveer códigos enfocados a funciones particulares o a conjunto de datos, atributos o códigos particulares que puedan ser verificados para el usuario actual. Un ejemplo de esto es el siguiente donde el atributo oculto (hidden) incluye valores únicos para negociar el comportamiento del formulario y la acción indicadas:

Tabla 16. Ejemplo de testigos.

```
<form action="/transfer.do" method="post">
<input type="hidden" name="8438927730"
value="43847384383">
...
</form>
```

Fuente: *(OWASP, 2008)*

3. **Para datos delicados o transacciones de gran valor, re-autenticar o utilizar firmado de transacción:** Es decir, verificar con información adicional propia del usuario que efectivamente es el que solicita ejecutar la acción sobre la información y que desea efectuar la acción solicitada. Por ejemplo, solicitando de nuevo su contraseña o verificando el mail ingresado para el mismo en Quadodo.

4. **No utilice peticiones GET (URLs) para datos delicados o realizar transacciones de valor.** Como se pudo observar durante las pruebas de pentesting realizadas a Quadodo, hay acciones que usan el método GET para la trasferencia de información de acciones como la creación de grupos. Es importante pues que esta información sea transmitida por método POST, incluyendo los valores testigos aleatorios indicados en el punto 3 de este listado para de esta manera realizar una petición de confianza entre el servidor y el cliente.

5. **El método POST aislado es una protección insuficiente:** Por lo cual se debe implementar lo indicado en el punto 4 respecto a la inclusión de ls testigos aleatorios para la acción a ejecutar.

Por otro lado en PHP es posible generar tokens de seguridad para evitar este tipo de ataques. El desarrollador Diego Lázaro muestra un ejemplo del código en la página que origina la petición y la página que la recibe, verificando que el token generado sea válido para el usuario (Lázaro, 2016):

Figura. 154. Ejemplo código PHP - Generación token CSRF y Validación token CSRF

```
$_SESSION["token"] = md5(uniqid(mt_rand(), true));
echo '<a href="process.php?action=logout&csrf=' .
$_SESSION["token"] . '">Logout</a></p>';
```

```
case "logout":
    if (isset($_GET["csrf"]) && $_GET["csrf"] == $_SESSION["token"]) {
        $_SESSION = array();
        session_destroy();
    }
break;
```

Fuente: *(Lázaro, 2016)*

12.4.6 Acciones correctivas Riesgo R8

Tabla 17. Riesgo R08

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R08	Cambio en la presentación de controles	CVE-2016-8008	Un atacante puede guiar al usuario a opciones con comportamientos no deseados combinando ataques de inyección de código.	**1 - Bajo**	**B – Posible**	**Tolerable**

Fuente: El autor.

Este riesgo se clasifica como de bajo impacto ya que no afecta a la información del sistema, más bien permite modificar la presentación del mismo, es decir colores, tamaño de letra, estilo de enlaces o links, etc. De forma aislada no resulta peligroso, sin embargo al combinarse con otros ataques de inyección de código (como los hallados a nivel de HTML y a nivel de Javascript) puede guiar al usuario a acciones con comportamiento no deseado o robar información. Es necesario por lo tanto que se mitiguen los riesgos indicados asociados con inyección de código HTML, Javascript y XSS para que se evite la generación de este riesgo adicional.

12.4.7 Acciones correctivas Riesgo 09

Tabla 18. Riesgo R09

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R09	Robo de información	CVE-2016-8009 (OWASP, 2015)	Es posible almacenar la información para usarla en ataques posteriores o identificar usuarios existentes en el sistema que se conviertan en victimas por parte de terceros según su rol.	**3 - Alto**	**B – Posible**	**Importante**

Fuente: El autor.

La causa de este riesgo como se evidencia en el presente documento, está asociada al test OTG-INPVAL-001 el cual es la convergencia como se indica de otras pruebas asociadas a inyección de código que ocasionan ataques de tipo XSS. El robo de información ocurre cuando se ejecutan acciones producto de inyección de código que redirigen a sitios no deseados que pueden capturar por método GET por ejemplo, información del usuario que se encuentra en sesión (rol, password, nombre, etc.) para posteriores ataques o ataques inmediatos que ataquen al

sistema o al mismo usuario, llevando a cabo por ejemplo borrado de información o ingeniería social respectivamente. Por lo tanto, la ejecución de las acciones correctivas asociadas a las causas indicadas corresponderá a una mejora de seguridad en el sistema y a la reducción de los riesgos que a su vez se asocian a cada una de ellas.

12.4.8 Acciones correctivas Riego 10

Tabla 19. Riesgo R10

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R10	Robo de sesión	CVE-2016-8010	Permite detectar que se está usando el Identificador de sesión de PHP, un atacante podría robar dicho Id y suplantar al usuario.	**3 - Alto**	**C – Probable**	**Importante**

Fuente. El autor.

Este riesgo está relacionado con la gestión de cookies del sitio, ya que el Id de sesión está expuesto a cualquier atacante o usuario con conocimientos básicos de manejo de herramienta de seguimientos en el navegador como Firebug.

Ya que este Identificador se guarda en una cookie se indican las acciones correctivas a aplicar a las mismas en los riesgos R11 y R12 que se describen a continuación del presente riesgo.

Por otro lado, PHP al igual que para las cookies tiene configuraciones que contribuyen a robustecer la gestión de la sesión haciendo uso de las propiedades que provee el archivo PHP.ini. Entre ellas se destacan las siguientes (PHP, 2016):

- ***session.use_strict_mode***: Propiedad booleana que previene que el usuario sea el que provea Identificadores de sesión, validando solo aquellos que hayan sido generados o inicializados por el módulo de sesiones del lenguaje.

- ***session.hash_function:*** Permite asignar la función hash a usar para la generación del Id de session, por lo tanto entre más fuerte sea la function en consecuencia se generarán Identificadores más robustos y menos vulnerables. Textualmente PHP recomienda que “aunque son improbables las colisiones hash incluso con MD5, los desarrolladores deberían utilizar funciones de hash SHA-2 o posterior para esta tarea. Los desarrolladores podrían emplear hash más fuertes, como sha384 y sha512.”

Es notable destacar que PHP indica que estas configuraciones entre las demás que señala para la seguridad y el manejo de sesiones no son suficientes y por lo tanto

deben ser implementadas medidas adicionales para que la seguridad sea asegurada. Integrar por lo tanto algoritmos de validación adicional, generación de tokens y tiempo límite para transacciones sensibles en la información es válido, teniendo en cuenta la navegabilidad y la expiración de identificadores que ya no se usen.

La seguridad de la sesión por lo tanto puede depender de la configuración del administrador del servidor de PHP, sin embargo, desde programación en Quadodo se deben forzar configuraciones y funcionalidades para que la seguridad de la sesión sea más robusta o se puede guiar al usuario a que las asegure mediante instrucciones en pantalla o avisos de seguridad de configuraciones requeridas.

12.4.9 Acciones correctivas Riesgos R11 y R12

Tabla 20. Riesgos R11 y R12

Id	Riesgo	Causas (CVE-ID)	Consecuencia	Nivel de impacto	Probabilidad de Ocurrencia	Perfil de Riesgo
R11	Lectura de cookies	CVE-2016-8011	Permite que un atacante pueda tener acceso inmediato a la información que contiene las cookies y que por lo tanto pueda hacer uso de ella. Una ventaja, es que además del Id de sesión se detecta que no hay información sensible almacenada en las mismas como claves o perfiles. Es por esto que se indica un impacto medio.	**2 - Medio**	**C – Probable**	**Importante**
R12	Manipulación de cookies	CVE-2016-8011	El atacante puede identificar el nombre de cada una de las cookies y la información que almacenan. Ya dependiendo del criterio del atacante, puede hallar útil o no manipular la información contenida en las mismas para realizar un ataque.	**2 - Medio**	**B – Posible**	**Moderado**

Fuente. El autor.

Estos riesgos se originan desde el mismo punto. La causa corresponde a la posibilidad que tiene un atacante de interceptar, leer y manipular las propiedades de las cookies que usa Quadodo.

Para solucionar esto, es indispensable que las cookies cuenten con los siguientes atributos, según lo indica OWASP en su guía de Pentesting (OWASP, 2016):

- ***Secure:*** Es importante asignar este atributo para las cookies que almacenan información sensible como el identificador de sesión. Al activar este atributo se asegura que el envío de esta información se realice a través de un canal seguro y de forma cifrada para evitar que sea leída por un atacante. Es posible activar este atributo en PHP mediante el archivo ***php.ini*** asignando al atributo booleano

session.cookie_secure el valor ***true***. (PHP, 2016). Por otro lado puede ser asignado mediante código usando el método ***session_set_cookie_params()*** antes de que se inicie la sesión (PHP, 2016). Hay que anotar que Quadodo provee la opción de configurar este atributo como se observa en el subtema de Despliegue y configuración de Quadodo del presnete documento, sin embargo, puede considerarse limitar esta opción a cookies que no guarden información sensible y no a todas las cookies.

- ***HttpOnly:*** Este atributo evita que a través de un script ejecutado del lado del navegador o del terminal cliente que accede a la página acceda al contenido de la cookie. Hay que aclarar que "no elimina el riesgo de scripts cross site pero si la explotación de algunos vectores de ataque". También puede ser configurado en PHP, en el archivo php.ini asignando el valor ***true*** al atributo booleano ***session.cookie_httponly***, igualmente es posible configurarlo a través del método ***session_set_cookie_params()***.

- ***Domain:*** La configuración de este atributo la provee Quadodo Esto es válido puesto que el script puede ser integrado a cualquier sitio web hecho en PHP el cual a su vez puede ser desplegado en el dominio que el administrador o desarrollador consideren. Dependerá pues que estos usuarios lo configuren correctamente. Como guía este atributo permite solamente al dominio que se configura utilizar las cookies del script, lo cual evita que otros servidores puedan hacer uso de la misma.

- ***Path:*** Este attributo debe ser configurado a la par del atributo Domain indicado anteriormente, ya que se protege las cookies para que solamente sean usadas por la ruta indicada dentro del dominio. Este atributo también puede ser configurado mediante Quadodo por las razones indicadas en el atributo anterior. Como ejemplo OWASP en su guía de pentesting indica su utilidad textualmente de la siguiente manera: "Si la aplicación reside en la ruta ***/myapp/*** entonces verifique que el atributo esté con el valor ***";path=/myapp/"*** y no ***";path=/"*** ó ***";path=/myapp'"*** (OWASP, 2016). Es importante incluir el carácter "/" debido a que se daría vía libre a que dominios que empiecen con la palabra ***myapp***.

- ***Expires:*** Este atributo puede asignarse a cookies que NO contengan información sensible. Corresponde a una fecha de expiración de la cookie, por lo cual la misma puede reutilizarse hasta la fecha indicada, las veces que se desee.

13. CONCLUSIONES

Se ejecutaron un total de 20 pruebas de pentesting realizadas a Quadodo bajo las configuraciones descritas sobre funcionalidades que se suministran en el módulo de seguridad (acceso y autenticación al sistema), en el módulo de administración y en el módulo de administración de grupos. De dichas pruebas, 11 presentaron un resultado fallido, es decir, que se descubrió mediante las mismas una o más vulnerabilidades presentes en el script a nivel de seguridad de la información.

De las 11 pruebas indicadas como fallidas o con resultados de vulnerabilidades encontradas, 10 de ellas presentaron un perfil de riesgo que debe ser gestionado de forma inmediata y prioritaria, esto fue posible identificarlo aplicando las normas ISO 30001:2009 e ISO/IEC 31010:2009.

Prevalecieron con un perfil de riesgo intolerable aquellos riesgos que abren una puerta inmediata a ataques de alto impacto como es la intercepción y robo de credenciales. Seguido están los riesgos con perfil Importante y moderado que son originados por acciones más elaboradas por parte de un atacante como inyección de código, ataques XSS o manipulación de cookies. Al final de estos solamente se identifica un riesgo con perfil Tolerable, que corresponde a la manipulación de CSS de Quadodo como script web, que de igual forma y como se ha reiterado al ser ejecutado de forma aislada, es decir solamente manipulando la presentación del sitio no resulta peligroso pero combinándolo con otros ataques de inyección de código puede desencadenar acciones no deseadas en el comportamiento del sistema y en la información del mismo.

Finalmente y de acuerdo a los resultados, no es conveniente integrar por el momento y al menos que se realicen las mitigaciones a los riesgos aquí planteados Quadodo Login Script por los niveles de riesgo de las vulnerabilidades identificadas, haciéndolo poco seguro para la gestión de usuarios, grupos y permisos.

BIBLIOGRAFÍA

ALCALDÍA DE BOGOTÁ. "LEY ESTATUTARIA 1266 DE 2008." (31 de Diciembre de 2008). {En línea}. {14 de noviembre de 2016} disponible en: http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=34488

AVANTIUM, Business Consulting. "Gestión de Riesgos (ISO 31000)." (2015). {En línea}. {14 de noviembre de 2016} disponible en: http://www.avantium.es/index.php/gestion-de-riesgos-iso-31000

CVE. "About CVE." (21 de Enero de 2015). {En línea}. {14 de noviembre de 2016} disponible en: https://cve.mitre.org/about/index.html

CVE. "CVE-ID Syntax Change." (1 de Julio de 2015). {En línea}. {14 de noviembre de 2016} disponible en: http://cve.mitre.org/cve/identifiers/syntaxchange.html

CVE. "About CVE." (6 de Abril de 2016). {En línea}. {14 de noviembre de 2016} disponible en: http://cve.mitre.org/about/index.html

CVE. "CVE Output in CVRF 1.1: 20160826." (2016). {En línea}. {14 de noviembre de 2016} disponible en: http://cve.mitre.org/data/downloads/allitems-cvrf-year-2016.xml

CVE. "Frequently Asked Questions." (22 de Abril de 2016). {En línea}. {14 de noviembre de 2016} disponible en: http://cve.mitre.org/about/faqs.html

CVE. "Request a CVE Identifier." (23 de Agosto de 2016). {En línea}. {14 de noviembre de 2016} disponible en: http://cve.mitre.org/cve/request_id.html

ESAÚ. “¿Qué es el Pentesting?” (12 de Junio de 2015). {En línea}. {5 de marzo de 2017} disponible en: https://openwebinars.net/blog/que-es-el-pentesting/

INFOSEC INSTITUTE. "Session Randomness Analysis with Burp Suite Sequencer." (Enero de 2014). {En línea}. {14 de noviembre de 2016} disponible en: http://resources.infosecinstitute.com/session-randomness-analysis-burp-suite-sequencer/

ISO. "ISO 31000 - Risk management." (2009). {En línea}. {14 de noviembre de 2016} disponible en: http://www.iso.org/iso/home/standards/iso31000.htm

KALI. "dotdotpwn Package Description." (18 de Febrero de 2014). {En línea}. {14 de noviembre de 2016} disponible en: http://tools.kali.org/information-gathering/dotdotpwn

Kali. "Kali Linux 2.0 Released." (11 de Agosto de 2015). {En línea}. {14 de noviembre de 2016} disponible en: https://www.kali.org/releases/kali-linux-20-released/

KALI LINUX. "What is Kali Linux?" (2016). {En línea}. {14 de noviembre de 2016} disponible en: http://docs.kali.org/introduction/what-is-kali-linux

KALI TOOLS. "Vega." (18 de Febrero de 2014). {En línea}. {14 de noviembre de 2016} disponible en: http://tools.kali.org/web-applications/vega

LÁZARO, Diego. "Ataques CSRF: Cross-Site Request Forgery en PHP." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://diego.com.es/ataques-csrf-cross-site-request-forgery-en-php

LEY 1273 DE 2009. (5 de Enero de 2009). Obtenido de Alcaldía de Bogotá: http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=34492

MELÉNDEZ GONZÁLEZ, Jonathan. "CAMBIOS EN LA NORMA ISO 9001:2015 (Acción preventiva - Gestión de Riesgos)." (Septiembre de 2015). {En línea}. {14 de noviembre de 2016} disponible en: http://www.cic-ctic.unam.mx/cic/mas_cic/servicios/cgcp/download/Seminario_ISO_9001_2015/Presentacion%20Ing%20Jonathan%20Melendez%20Gonzalez.pdf

MUKKAMALA, Charan. "Identifying Cross Site Scripting Vulnerabilities Using Automated Tools." (s.f.). {En línea}. {14 de noviembre de 2016} disponible en: https://entersoftsecurity.com/blog/identifying-cross-site-scripting-vulnerabilities-using-automated-tools.html

MySQL. "6.2.7 Troubleshooting Problems Connecting to MySQL." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://dev.mysql.com/doc/refman/5.6/en/problems-connecting.html

NATES PARRA, César Francisco. "ISO 31000 GESTIÓN DEL RIESGO. PRINCIPIOS Y DIRECTRICES." (Agosto de 2011). {En línea}. {14 de noviembre de 2016} disponible en: https://jrcontreras.files.wordpress.com/2015/02/21_gestion_riesgo_iso_31000.pdf

NMAP.org. "File mysql-brute." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://nmap.org/nsedoc/scripts/mysql-brute.html

NMAP.org. "File mysql-databases." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://nmap.org/nsedoc/scripts/mysql-databases.html

NMAP.org. "File mysql-empty-password." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://nmap.org/nsedoc/scripts/mysql-empty-password.html

NMAP.org. "Options Summary." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://nmap.org/book/man-briefoptions.html

OFFENSIVE SECURITY. "¿Qué es Kali Linux?" (2015). {En línea}. {14 de noviembre de 2016} disponible en: http://es.docs.kali.org/introduction-es/que-es-kali-linux

OFFENSIVE SECURITY. "About Kali Linux." (2015). {En línea}. {14 de noviembre de 2016} disponible en: https://www.kali.org/about-us/

OWASP. "Top 10 2007-Vulnerabilidades de Falsificación de Petición en Sitios Cruzados (CSRF)." (7 de Diciembre de 2008). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Top_10_2007-Vulnerabilidades_de_Falsificaci%C3%B3n_de_Petici%C3%B3n_en_Sitios_Cruzados_(CSRF)

OWASP. "OWASP." (1 de Diciembre de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_DOM-based_Cross_site_scripting_(OTG-CLIENT-001)

OWASP. "Sobre OWASP." (11 de Noviembre de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Sobre_OWASP

OWASP. “SQL Injection” (10 de Abril de 2016). {En línea}. (5 de marzo de 2017) disponible en https://www.owasp.org/index.php/SQL_Injection

OWASP. "Test Local Storage (OTG-CLIENT-012)." (1 de Diciembre de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Test_Local_Storage_(OTG-CLIENT-012)

OWASP. "Test User Registration Process (OTG-IDENT-002)." (14 de Mayo de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Test_User_Registration_Process_%28OTG-IDENT-002%29

OWASP. "Test User Registration Process (OTG-IDENT-002)." (14 de Mayo de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Test_User_Registration_Process_(OTG-IDENT-002)

OWASP. "Testing for Account Enumeration and Guessable User Account (OTG-IDENT-004)." (15 de Agosto de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_Account_Enumeration_and_Guessable_User_Account_(OTG-IDENT-004)

OWASP. "Testing for Credentials Transported over an Encrypted Channel (OTG-AUTHN-001)." (30 de Diciembre de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_Credentials_Transported_over_an_Encrypted_Channel_(OTG-AUTHN-001)

OWASP. "Testing for CSS Injection (OTG-CLIENT-005)." (8 de Agosto de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_CSS_Injection_(OTG-CLIENT-005)

OWASP. "Testing for DOM-based Cross site scripting (OTG-CLIENT-001)." (1 de Diciembre de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_DOM-based_Cross_site_scripting_(OTG-CLIENT-001)

OWASP. "Testing for HTML Injection (OTG-CLIENT-003)." (8 de Agosto de 2014). {En línea}. {14 de noviembre de 2016} disponible en: Testing for HTML Injection (OTG-CLIENT-003)

OWASP. "Testing for Input Validation." (8 de Agosto de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_Input_Validation

OWASP. "Testing for JavaScript Execution (OTG-CLIENT-002)." (1 de Diciembre de 2014). {En línea}. {14 de noviembre de 2016} disponible en: Testing for JavaScript Execution (OTG-CLIENT-002)

OWASP. "Testing for Session Management Schema (OTG-SESS-001)." (31 de Julio de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_Session_Management_Schema_(OTG-SESS-001)

OWASP. "Testing Identity Management." (14 de Mayo de 2014). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_Identity_Management

OWASP. "OWASP Testing Guide 4.0." (3 de Agosto de 2015). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/images/5/52/OWASP_Testing_Guide_v4.pdf

OWASP. "Testing for Reflected Cross site scripting (OTG-INPVAL-001)." (29 de Junio de 2015). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_Reflected_Cross_site_scripting_(OTG-INPVAL-001)

OWASP. "Cross-Site Request Forgery (CSRF) Prevention Cheat Sheet." (12 de Octubre de 2016). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet

OWASP. "Testing for cookies attributes (OTG-SESS-002)." (19 de Marzo de 2016). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_for_cookies_attributes_(OTG-SESS-002)

OWASP. "Testing Guide Introduction." (14 de Junio de 2016). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/index.php/Testing_Guide_Introduction#Penetration_Testing

PHP. "Runtime Configuration." (2016). {En línea}. {14 de noviembre de 2016} disponible en: http://php.net/manual/en/session.configuration.php

PHP. "Sesiones y seguridad." (2016). {En línea}. {14 de noviembre de 2016} disponible en: http://php.net/manual/en/session.security.php

PHP. "session_set_cookie_params." (2016). {En línea}. {14 de noviembre de 2016} disponible en: http://php.net/manual/en/function.session-set-cookie-params.php

PORTSWIGGER WEB SECURITY. "How the Randomness Tests Work." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://portswigger.net/burp/help/sequencer_tests.html

RENNEHAN, Douglas. "ChangeLog for the Quadodo Login Script 3.1.x." (2013). {En línea}. {14 de noviembre de 2016} disponible en: http://dev.quadodo.net/qls-3.1.11/CHANGELOG

RENNEHAN, Douglas. "User Guide - Free Login Script." (2013). {En línea}. {14 de noviembre de 2016} disponible en: http://dev.quadodo.net/qls-3.1.11/USERGUIDE.html

RENNEHAN, Douglas. "Downloading the Script." (2015). {En línea}. {14 de noviembre de 2016} disponible en: http://www.quadodo.net/downloads.php

RENNEHAN, Douglas. "Forum Bugs." (2015 de Abril de 2015). {En línea}. {14 de noviembre de 2016} disponible en: http://www.quadodo.net/forum/viewforum.php?f=14&sid=b64b6523a487a04b6c22e21f98bef7d0

RENNEHAN, Douglas. "PHP Login Script?" (26 de Abril de 2015). {En línea}. {14 de noviembre de 2016} disponible en: http://www.quadodo.net/

RENNEHAN, Douglas. "Features." (26 de Abril de 2016). {En línea}. {14 de noviembre de 2016} disponible en: http://www.quadodo.net/features.php

REYES PLATA, Alejandro. “Ethical Hacking” (25 de Octubre de 2010). {En línea}. {4 de marzo de 2017} disponible en https://www.seguridad.unam.mx/descarga.dsc?arch=2776

RENNEHAN, Douglas. "My PHP Login Script." (2015 de Abril de 26). {En línea}. {14 de noviembre de 2016} disponible en: http://www.quadodo.net/login-script.php

ROUSE, Margaret. "Prueba de penetración (pen test)." (Marzo de 2014). {En línea}. {14 de noviembre de 2016} disponible en: http://searchdatacenter.techtarget.com/es/definicion/Prueba-de-penetracion-pen-test

SGSI. “ISO 27001: ¿Qué significa la Seguridad de la Información?” (21 de Mayo de 2015). {En línea}. {5 de marzo de 2017} disponible en: http://www.pmg-ssi.com/2015/05/iso-27001-que-significa-la-seguridad-de-la-informacion/

SOLARTE SOLARTE, Francisco Nicolás Javier. "Lección 30: Legislación en Seguridad Informática en Colombia." (2012). {En línea}. {14 de noviembre de 2016} disponible en: http://datateca.unad.edu.co/contenidos/233005/contenido%20en%20linea%20PELSI_I_2013/leccin_30_legislacin_en_seguridad_informtica_en_colombia.html

UBUNTU. "Apt-Get." (2016). {En línea}. {14 de noviembre de 2016} disponible en: https://help.ubuntu.com/lts/serverguide/apt-get.html

VICEPRESIDENCIA JURÍDICA Y ADMINISTRATIVA DEL FONDO NACIONAL DE GARANTÍAS. "Ley de Hábeas Data Ley 1266 de 2008." (2008). {En línea}. {14 de noviembre de 2016} disponible en: https://www.fng.gov.co/ES/Documentos%20%20Proteccin%20de%20Datos%20Personales/Manual%20Habeas%20Data.pdf

VINICIUS. "Sistema de Login com PHP/MySQL (Quadodo)." (28 de Julio de 2013). {En línea}. {14 de noviembre de 2016} disponible en: http://www.monolitonimbus.com.br/sistema-de-login/

RESUMEN ANALÍTICO ESPECIALIZADO - RAE

1. Información General			
Tema	Ejecución de pruebas de pentesting sobre proyecto web "Quadodo Login Script"		
Título	Pentesting al proyecto web "Quadodo Login Script" desarrollado y soportado en lenguaje php versión 5.5.0		
Autor	Henry Alfonso Garzón Pinzón	***Año***	2017
Director	José Hernando Peña Hidalgo		
Fuente Bibliográfica	Se referencian 65 fuentes bibliográficas, de las cuales se destacan a continuación las principales para el desarrollo del proyecto: • AVANTIUM, Business Consulting. "Gestión de Riesgos (ISO 31000)." (2015). {En línea}. {14 de noviembre de 2016} disponible en: http://www.avantium.es/index.php/gestion-de-riesgos-iso-31000 • CVE. "About CVE." (21 de Enero de 2015). {En línea}. {14 de noviembre de 2016} disponible en: https://cve.mitre.org/about/index.html • ISO. "ISO 31000 - Risk management." (2009). {En línea}. {14 de noviembre de 2016} disponible en: http://www.iso.org/iso/home/standards/iso31000.htm • NATES PARRA, César Francisco. "ISO 31000 GESTIÓN DEL RIESGO. PRINCIPIOS Y DIRECTRICES." (Agosto de 2011). {En línea}. {14 de noviembre de 2016} disponible en: https://jrcontreras.files.wordpress.com/2015/02/21_gestion_riesgo_iso_31000.pdf • OWASP. "OWASP Testing Guide 4.0." (3 de Agosto de 2015). {En línea}. {14 de noviembre de 2016} disponible en: https://www.owasp.org/images/5/52/OWASP_Testing_Guide_v4.pdf • RENNEHAN, Douglas. "User Guide - Free Login Script." (2013). {En línea}. {14 de noviembre de 2016} disponible en: http://dev.quadodo.net/qls-3.1.11/USERGUIDE.html • REYES PLATA, Alejandro. “Ethical Hacking” (25 de Octubre de 2010). {En línea}. {4 de marzo de 2017} disponible en https://www.seguridad.unam.mx/descarga.dsc?arch=2776 • SGSI. “ISO 27001: ¿Qué significa la Seguridad de la Información?” (21 de Mayo de 2015). {En línea}. {5 de marzo de 2017} disponible en: http://www.pmg-ssi.com/2015/05/iso-27001-que-significa-la-seguridad-de-la-informacion/ • SOLARTE SOLARTE, Francisco Nicolás Javier. "Lección 30: Legislación en Seguridad Informática en Colombia." (2012). {En línea}. {14 de noviembre de 2016} disponible en: http://datateca.unad.edu.co/contenidos/233005/contenido%20en%20linea%20PELSI_I_2013/leccin_30_legislacin_en_seguridad_informtica_en_colombia.html		
Resumen	La monografía desarrollada se centra en la práctica de ejecución de pruebas de penetración (pentesting) encaminadas a identificar las vulnerabilidades y ventajas de seguridad de la información de un sistema, componente o programa seleccionado en este caso Quadodo Login Script, un módulo que provee la funcionalidad de autenticación de usuarios, además de otras funcionalidades como administración de usuarios, administración de grupos, permisos de acceso sobre páginas y creación de máscaras de permisos y que puede ser integrado a cualquier aplicación desarrollada en PHP. Las pruebas de pentesting antes mencionadas se aplican en el desarrollo de esta monografía sobre los módulos Security, Administration y Group Control		

	Panel de Quadodo. Para la ejecución de las mismas se toma en cuenta la cuarta versión de la guía de pruebas de OWASP de pentesting para definir las pruebas a ejecutar y se acude a las normas ISO NORMAS ISO 30001:2009 e ISO/IEC 31010:2009 para una vez identificadas las vulnerabilidades, relacionar los riesgos asociados y plantear opciones de mejora para gestionar los mismos.
Palabras Claves	*CVE, Ethical Hacking, ISO/IEC, Kali Linux, OWASP, Pentesting, Seguridad de la Información, SQL Injection,* XSS
Contenidos	La monografía se divide en los siguientes temas principales. 1. TÍTULO 2. INTRODUCCIÓN 3. DESCRIPCIÓN DEL PROBLEMA 4. JUSTIFICACIÓN 5. OBJETIVOS DEL PROYECTO 6. MARCO REFERENCIAL (Contiene los marcos contextual, teórico, legal, conceptual) 7. RECURSOS DEL PROYECTO 8. DISEÑO METODOLÓGICO DEL PROYECTO 9. CONFIGURACIÓN Y ADECUACIÓN DEL AMBIENTE 10. RECONOCIMIENTO PREVIOS A LAS PRUEBAS DE PENTESTING 11. DEFINICIÓN Y EJECUCIÓN DE PRUEBAS DE PENTESTING UTILIZANDO LA METODOLOGÍA OWASP 12. CONSOLIDACIÓN DE RESULTADOS Y GENERACIÓN DE REPORTE (Contiene análisis de resultados, gestión de los riesgos generados aplicando las normas ISO indicadas y recomendaciones para el tratamiento de los riesgos y las vulnerabilidades encontradas) 13. CONCLUSIONES BIBLIOGRAFÍA

2. Descripción del problema de investigación

ANTECEDENTES DEL PROBLEMA: Quadodo se integra con aplicaciones web construidas con lenguaje PHP para reusar funcionalidades previamente construidas de seguridad como autenticación y administración de usuarios y permisos. En caso que una de estas funcionalidades a nivel de seguridad informática presente vulnerabilidades de cualquier tipo, heredará con certeza las mismas al aplicativo que lo implemente, comprometiendo de esta manera su sistema y la información que se almacena, generando también riesgos de diferente magnitud al exponer a atacantes puntos débiles a aprovechar.

FORMULACIÓN DEL PROBLEMA. Actualmente no existe una evaluación formal ejecutada ni documentada de seguridad informática sobre Quadodo Login Script que permita identificar las posibles vulnerabilidades en materia de seguridad de la información que pueda presentar dicho script y que por ende deban corregirse, ni tampoco evidenciar las ventajas que pueda presentar el mismo y que deban mantenerse.

DESCRIPCIÓN Y PREGUNTA DEL PROBLEMA: Por lo anterior, se requiere que se identifiquen tanto las vulnerabilidades presentes en el script como las fortalezas en materia de seguridad informática para garantizar que las aplicaciones que lo integren no sufran o se vean afectadas, es por esto que se plantea la incógnita: *¿Cómo es posible identificar las vulnerabilidades de seguridad informática que puedan estar presentes en Quadodo Login Script y las funcionalidades que este ofrece de manera segura?*

3. Objetivos

OBJETIVO GENERAL: Identificar vulnerabilidades de seguridad presentes en los módulos *Security, Administration y Group Control Panel* del proyecto web "Quadodo Login Script", mediante pruebas de pentesting utilizando la metodología OWASP.

OBJETIVOS ESPECÍFICOS

- Definir las pruebas de pentesting a ejecutar sobre Quadodo Login Script alineadas a la guía OWASP de pruebas de pentesting versión 4 (OWASP Testing Guide v. 4).

- Ejecutar pruebas de pentesting sobre los módulos *Security, Administration y Group Control Panel*.
- Relacionar las vulnerabilidades que se encuentren asignando un identificador basado en la estructura del Common Vulnerabilities and Exposures (CVE).
- Identificar y evaluar los riesgos de seguridad informática asociados a las vulnerabilidades que se encuentren en Quadodo Login Script.
- Generar recomendaciones de acciones de seguridad a ejecutar para mitigar las vulnerabilidades y riesgos que se encuentren en Quadodo Login Script.

4. Metodología

TIPO DE INVESTIGACIÓN: Investigación aplicada encaminada a aplicar los conocimientos adquiridos a lo largo de la especialización en seguridad informática.
TÉCNICAS DE RECOLECCIÓN DE DATOS: Usa las pruebas aplicadas como principal fuente de datos, es aplicado mediante la ejecución de pruebas de pentesting a Quadodo Login Script.
TÉCNICAS DE ANÁLISIS DE DATOS: Una vez obtenidos los resultados y datos de las vulnerabilidades se realiza la clasificación y tabulación de las mismas, haciendo uso también del diccionario CVE (Common Vulnerabilities and Exposures) mediante el cual se asigna un identificador único a cada una de ellas (codificación de resultados). Se establece finalmente una relación de riesgos aplicado el análisis preliminar de riesgos y matrices de probabilidad y consecuencia como métodos de consulta establecidos en la norma ISO/IEC 31010:2009, proponiendo diferentes soluciones basadas en la experiencia y en la documentación técnica encontrada.
ETAPAS DEFINIDAS: Para el desarrollo de la monografía planteada, se definieron las siguientes etapas. ***(1)*** Configuraciones y adecuación de ambiente, **(2)** Planeación, alcance y ejecución de las pruebas de pentesting y ***(3)*** Consolidación de resultados y generación de reporte.

5. Referentes teóricos

Para la ejecución de pruebas se tienen en cuenta las siguientes fases del pentesting (Rouse, 2014):

1. Recolección de información del sistema u objetivo previo a la ejecución de las pruebas a ejecutar.
2. Identificación de los puntos a evaluar del sistema objetivo (puntos de entrada)
3. Ejecución de pruebas, realizando intentos de entrada y explotación de posibles vulnerabilidades.
4. Consolidación y reporte de resultados.

Las pruebas ejecutadas corresponden a pruebas planteadas en la cuarta versión de la guía de pruebas de OWASP.
Se realiza la consolidación y reporte de resultados haciendo uso del diccionario CVE (Common Vulnerabilities and Exposures) el cual es utilizado para publicar vulnerabilidades conocidas de la seguridad de la información, haciendo más fácil compartir datos y evaluar la cobertura de una organización en materia de seguridad.
El análisis de riesgos de seguridad de la información realizado se basa en los resultados obtenidos de las pruebas ejecutadas haciendo uso de las normas ISO 30001:2009 e ISO/IEC 31010:2009. La primera es la normativa para 'proponer unas pautas genéricas sobre cómo gestionar los riesgos de forma sistemática y transparente' (Avantium, Business Consulting, 2015) y la segunda apoya este proceso proporcionando varios artefactos o herramientas para la evaluación de riesgos.

6. Referentes Conceptuales

Se toman los conceptos enmarcados en las palabras clave indicados en este documento:
***CVE*:** Son las siglas de Common Vulnerabilities and Exposures (Vulnerabilidades y Exposiciones comunes en español), es un diccionario centralizado donde se registran vulnerabilidades de seguridad conocidas en la red asignándoles un identificador (CVE ID) teniendo como objetivo la clasificación de vulnerabilidades por herramientas, repositorios y servicios. (CVE, 2015)

Ethical Hacking: Hackeo Ético, es la disciplina de seguridad de la información que se encarga de detectar y explotar las vulnerabilidades sobre sistemas de información con el fin de evaluar y verificar la seguridad tanto física como lógica de los mismos.
ISO/IEC: Estas siglas hacen referencia a las directivas, normas y políticas aplicados a procedimientos, definidos como estándares internacionales. (ISO, 2017).
Kali Linux: Sistema operativo basado en Debian que contiene herramientas para realizar pruebas y ataques de seguridad desde el punto de vista del hacking ético. (Kali Linux, 2016)
OWASP: Es la sigla de Open Web Application Security Project (Proyecto Abierto de Seguridad en Aplicaciones Web), una fundación sin ánimo de lucro cuyo objetivo es promover e incrementar la seguridad del software y la web en general. (OWASP, 2014)
Pentesting: "*…Es la práctica de atacar diversos entornos con la intención de descubrir fallos, vulnerabilidades u otros fallos de seguridad, para así poder prevenir ataques externos hacia esos equipos o sistemas*" (Esaú, 2015).
Seguridad de la Información: Hace referencia a "*la confidencialidad, la integridad y la disponibilidad de la información y los datos importantes para la organización, independientemente del formato que tengan*" como formatos electrónicos, en físico, medios audiovisuales, etc. (SGSI, 2015)
SQL Injection: Tipo de ataque que tiene como objetivo leer y manipular datos sensibles de la base de datos "inyectando" sentencias SQL a través de campos o parámetros vulnerables. (OWASP, 2016)
XSS: Es la sigla de Cross-site Scripting, un tipo de ataque que realiza inyección de scripts o de código malicioso en sitios web. (OWASP, 2016)

7. Resultados

Se ejecutaron un total de 20 pruebas de pentesting sobre Quadodo bajo las configuraciones descritas sobre funcionalidades que se suministran en el módulo de seguridad (acceso y autenticación al sistema), en el módulo de administración y en el módulo de administración de grupos. De dichas pruebas, 11 presentaron un resultado fallido, es decir, que se descubrió mediante las mismas una o más vulnerabilidades presentes en el script a nivel de seguridad de la información.
De las 11 pruebas indicadas como fallidas o con resultados de vulnerabilidades encontradas, 10 de ellas presentaron un perfil de riesgo que debe ser gestionado de forma inmediata y prioritaria, esto fue posible identificarlo aplicando las normas ISO 30001:2009 e ISO/IEC 31010:2009.
Prevalecieron con un perfil de riesgo intolerable aquellos riesgos que abren una puerta inmediata a ataques de alto impacto como es la intercepción y robo de credenciales. Seguido están los riesgos con perfil Importante y moderado que son originados por acciones más elaboradas por parte de un atacante como inyección de código, ataques XSS o manipulación de cookies. Al final de estos solamente se identifica un riesgo con perfil Tolerable, que corresponde a la manipulación de CSS de Quadodo como script web, que de igual forma y como se ha reiterado al ser ejecutado de forma aislada, es decir solamente manipulando la presentación del sitio no resulta peligroso pero combinándolo con otros ataques de inyección de código puede desencadenar acciones no deseadas en el comportamiento del sistema y en la información del mismo.

8. Conclusiones

De acuerdo a los resultados, no es conveniente integrar por el momento y al menos que se realicen las mitigaciones a los riesgos aquí planteados Quadodo Login Script por los niveles de riesgo de las vulnerabilidades identificadas, haciéndolo poco seguro para la gestión de usuarios, grupos y permisos.