ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

VICERRECTORADO DE INVESTIGACIÓN INNOVACIÓN Y TRANSFERENCIA TECNOLÓGICA

DIRECCIÓN DE POSGRADOS

TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE MAGÍSTER EN GERENCIA DE SISTEMAS

TEMA: CASO DE ESTUDIO - PRUEBAS NO FUNCIONALES EN LA NUBE Y EN SITIO

AUTOR: FLORES RAMÍREZ, CARLOS JAVIER

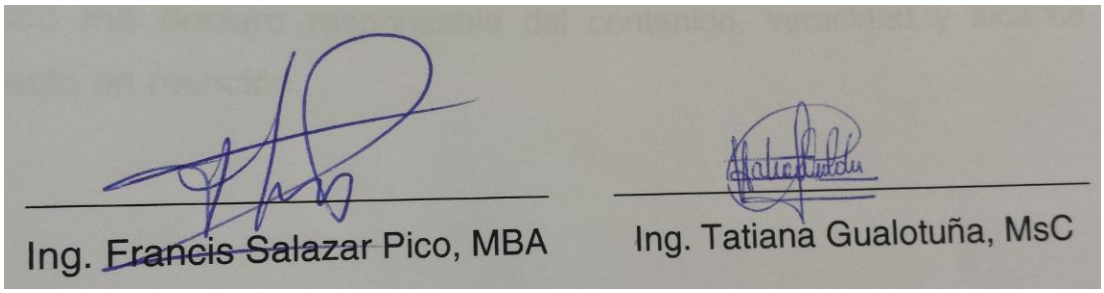
DIRECTOR: ING. MBA FRANCIS SALAZAR PICO

SANGOLQUÍ

2015

CERTIFICADO

Certificamos que el presente proyecto titulado “Caso de Estudio – Pruebas no Funcionales en la nube y en sitio”, fue desarrollado en su totalidad por el Ing. Carlos Javier Flores Ramírez, bajo nuestra dirección.



Ing. Francis Salazar Pico, MBA

Ing. Tatiana Gualotuña, MsC

AUTORÍA DE RESPONSABILIDAD

El presente proyecto titulado “Caso de Estudio – Pruebas no Funcionales en la nube y en sitio”, ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado el derecho intelectual de terceros considerándolos en citas a pie de página y como fuentes en el registro bibliográfico.

Consecuentemente declaro que este trabajo es de mi autoría, en virtud de ello me declaro responsable del contenido, veracidad y alcance del proyecto en mención.

_____

Ing. Carlos J. Flores Ramírez

AUTORIZACIÓN

Yo, Carlos Javier Flores Ramírez, autorizo a la Universidad de las Fuerzas Armadas “ESPE” a publicar en la biblioteca virtual de la institución el presente trabajo “Caso de Estudio – Pruebas no Funcionales en la nube y en sitio”, cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Sangolquí, julio del 2015

_____

Ing. Carlos J. Flores Ramírez

DEDICATORIA

El esfuerzo invertido en este proyecto académico quiero dedicarlo a mi familia que siempre ha sido mi motivación y fuerza para culminar con todos los proyectos emprendidos, quienes pacientemente han sabido entender mis momentos de ausencia con ellos, comprendiendo que han sido con un fin positivo.

A mi amada esposa Patricia y mi adorada hija Paula quiero decirles que este nuevo peldaño que se ha subido no hubiera sido posible sin su apoyo.

AGRADECIMIENTO

Mi principal agradecimiento es a Dios por haberme permito culminar este importante proyecto académico y poder aplicar los conocimientos adquiridos en mi labor profesional, después a los ingenieros Francis Salazar y Tatiana Gualotuña quienes con su guía han dado realce al contenido de este proyecto.

A la Universidad de las Fuerzas Armadas “ESPE” y sus colaboradores que han demostrado ser excelentes profesionales y poseer un alto espíritu de compromiso y apoyo al estudiante.

A mi familia por ser siempre un soporte y aliento en las decisiones que he tomado a lo largo de mi vida personal, académica y profesional. Mi esposa, mi hija, mis padres, mi hermano MUCHAS GRACIAS por su valiosa e incondicional ayuda en todo momento.

Índice de contenidos

Índice de Ilustraciones

Índice de Tablas

RESUMEN

Este proyecto es una guía de cómo configurar y ejecutar pruebas no funcionales de carga y seguridad en la nube y en sitio con un grupo de herramientas licenciadas, versiones comunitarias y libres de la nube.

Se optó por este par de pruebas no funcionales debido a que son las más conocidas en el medio tecnológico, fueron ejecutadas para aplicaciones de intranet, internet y móvil, todas con una arquitectura diferente.

El ambiente de ejecución es una importante variable a considerar porque de esto depende que tipo de herramienta se va a utilizar en las pruebas, por lo general la mayoría de herramientas que se ejecutan en sitio poseen más características y componentes que permiten una más amplia configuración de la herramienta con la aplicación, así como requiere de una persona que posea el conocimiento para poder utilizar correctamente la herramienta y elaborar reportes que aporten con el objetivo de las pruebas. Por otro lado en la nube no se tiene el problema de poseer una infraestructura propia para las pruebas no funcionales que se puede convertir en un ahorro debido a que no se requiere dar mantenimiento a este ambiente adicional.

La arquitectura de la aplicación es otra pieza clave en la evaluación de la herramienta que se va a utilizar ya que algunos componentes tecnológicos no son soportados por todas las herramientas y esto se convierte en una limitante al momento de grabar la prueba.

PALABRAS CLAVES:
CONFIGURACIÓN
EJECUCIÓN
SITIO
NUBE
PRUEBAS NO FUNCIONALES

ABSTRACT

This project is a guide on how to set up and run non-functional testing like load and security; on the cloud and on-site with a group of licensed tools, community and free versions of the cloud.

These non-functional tests were chosen because they are the best known in the technological environment, they were executed for: Intranet, Internet and Mobile applications, all of them with a different architecture.

Environment of execution is an important variable to consider because the type of tool to be used in testing depends of it, usually most tools that run on site have more features and components that allow a wider configuration and requires a person with the knowledge to properly use the tool and produce reports that contribute to the objective of the tests. Furthermore, executing tests on the cloud removes the problem of having their own infrastructure for non-functional testing that can be converted into savings because it does not requires maintenance for this environment.

The application architecture is another key element in the assessment of the tool to be used, because some technological components are not supported by all tools and this becomes a limiting factor when recording the test.

KEY WORDS:

SETTING UP

EXECUTION

ON-SITE

CLOUD

NON-FUNCTIONAL TESTING

1. CAPÍTULO I
INTRODUCCIÓN

1.1. Antecedentes

La fase inicial de este proyecto cubrió en primera instancia la parte teórica sobre los conceptos de calidad durante el ciclo de vida de un producto de software en cualquier tipo de organización, siempre referenciando a mejores prácticas y estándares del mercado actual, esto permitió crear una base de conocimiento que posteriormente se utilizó para hacer énfasis en el proceso de las pruebas no funcionales de software o pruebas técnicas como también se las conoce. Entre las más conocidas se pueden mencionar pruebas de carga, estrés, rendimiento, seguridad.

Posterior al fundamento teórico se realizó una investigación de mercado que permitió conocer un poco más el estado de algunas de las organizaciones afiliadas a la AESOFT[1] con el fin de identificar si cumplen con procedimientos de calidad durante la elaboración del software, si tienen equipos de trabajo para tareas exclusivas de calidad o si estas tareas las realizan los propios desarrolladores y que tipos de pruebas ejecutan. Respecto a esto se evidenció que la mayoría de las organizaciones conoce y tiene establecido algún procedimiento de calidad en la elaboración del software, independiente de su tamaño, sin embargo no todas tienen equipos exclusivos para realizar estas actividades de manera continua lo que se puede traducir a que realizan control de calidad más no llegan al aseguramiento de calidad.

El enfoque principal del anterior proyecto era direccionar hacia las pruebas no funcionales de software y sus ambientes de ejecución, este tipo de pruebas requieren conocimiento técnico y experiencia por parte del analista de pruebas, aparte de las herramientas respectivas, para poder detectar ciertos defectos en las aplicaciones que en muchas ocasiones el usuario final desconoce cómo evidenciarlas, incluso desconoce cómo estas pueden afectar gravemente al giro del negocio. Por ejemplo, si no se prueba

[1] Asociación Ecuatoriana de Software

la concurrencia de usuarios en una página web de ventas en línea por medio de pruebas de carga al momento que se encuentre operacional en producción conforme vayan accediendo usuarios puede degradarse la aplicación provocando caídas del servicio y como resultado final que los clientes no deseen volver a comprar en ese sitio web, este es uno de los muchos aspectos que pueden solucionar la planificación de pruebas no funcionales en las aplicaciones.

1.2. Justificación

En nuestro medio las organizaciones que desarrollan software si conocen o si están familiarizadas con el concepto de aseguramiento de la calidad en software y el concepto de pruebas no funcionales.

Independiente del tamaño de la organización, la mayoría de ellas realiza pruebas no funcionales, esto indica la gran responsabilidad de cada organización por elaborar productos de software apegados a estándares de calidad.

Las organizaciones medianas y grandes son aquellas que pueden tener equipos de personas dedicadas exclusivamente a tareas de aseguramiento de la calidad, y por lo tanto, personas que se puedan especializar en el uso de herramientas para realizar pruebas técnicas.

Los resultados de las pruebas no funcionales no siempre son fáciles de interpretar por parte de los usuarios finales de las aplicaciones, por esto se necesita la ayuda de un analista de pruebas con experiencia para determinar posibles sectores defectuosos en el software y a medida que el tiempo transcurre se puede ir comprobando la robustez del mismo.

En base a lo anterior mencionado se desea aportar con una guía de cómo se pueden configurar, ejecutar y obtener resultados en algunas herramientas que realizan pruebas no funcionales de software tanto en sitio como en la nube para identificar qué tipo de ambiente puede ser el mejor para cada organización.

1.3. Objetivo General

Documentar una secuencia de pasos ordenados para elaborar pruebas no funcionales de software en aplicaciones de diferentes estructuras tecnológicas y que se despliegan en distintas redes dentro de una organización como intranet e internet.

1.4. Objetivos Específicos

- Configurar las herramientas en sitio y en la nube para las pruebas con las condiciones que se han definido en el proyecto precedente.
- Identificar si es necesario o no la intervención de un analista de pruebas especializado para la configuración de las distintas herramientas y la interpretación de sus resultados.
- Ejecutar las pruebas no funcionales en cada una de las herramientas y ambientes planificados.
- Evidenciar el alcance de los resultados que se puede obtener en cada ambiente y con cada herramienta probada para que ayude a cualquier organización a seleccionar una herramienta y ambiente que cubra sus necesidades.
- Señalar las limitaciones que tienen las herramientas considerando que algunas de ellas fueron utilizadas en sus versiones gratuitas o comunitarias.

2. CAPÍTULO II
CONFIGURACIÓN DE AMBIENTES

2.1. Configuración de herramienta en sitio para pruebas de Carga

La herramienta que se utilizó para realizar las pruebas de carga es SilkPerformer de la suite de Borland, a continuación se explica cómo se debe configurar la herramienta para ejecutar la prueba de carga, esto sirve para cualquier tipo de aplicación web.

En la ilustración 1 se puede apreciar que una vez que la herramienta se ha desplegado y se ha seleccionado el tipo de aplicación como web (HTTP) aparece una ventana donde se especifica el perfil de la aplicación que para este caso va a ser el navegador con el que se grabó la prueba (Mozilla Firefox), a continuación pide que se llene el campo URL, esto es muy importante debido a que es la dirección de donde va a escuchar el puerto 8080 para empezar a grabar la herramienta los eventos del mouse.

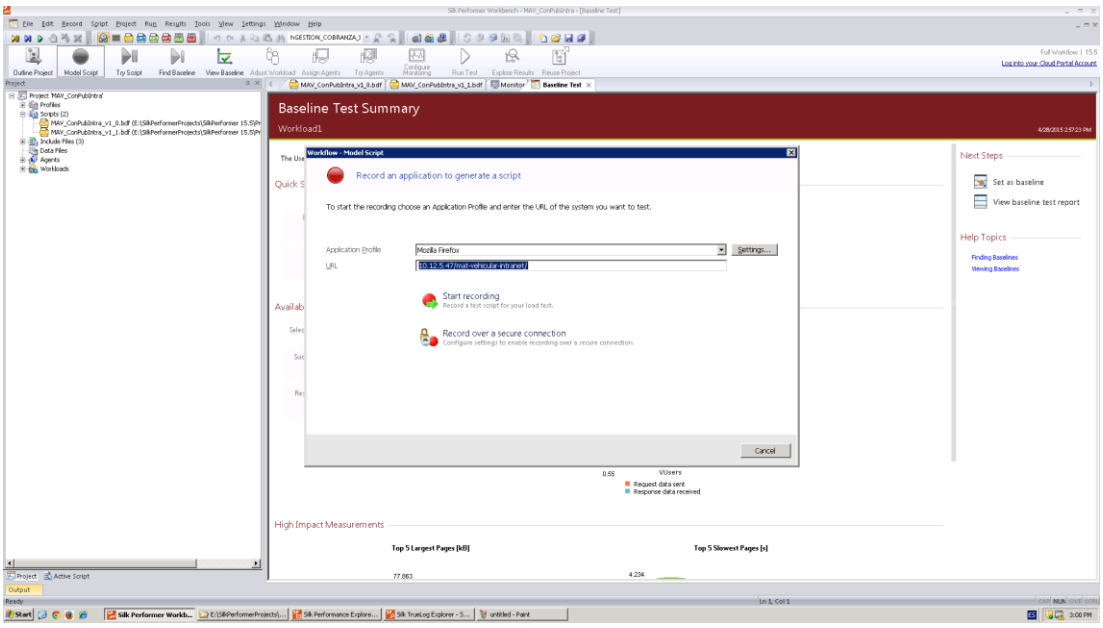


Ilustración 1 Grabación de la prueba

Una vez que se han llenado los dos campos, perfil y URL se procede a presionar el botón “Start Recording”, de inmediato se inicia el navegador Mozilla con la dirección ingresada y se sobrepone en la esquina superior

derecha una pequeña ventana que indica que los movimientos se están grabando en la herramienta.

Cuando se finaliza la grabación se debe presionar el botón “Stop” de la ventana esquinera y se detendrá la grabación, posterior aparece el script que se ha grabado como se muestra en la ilustración 2.

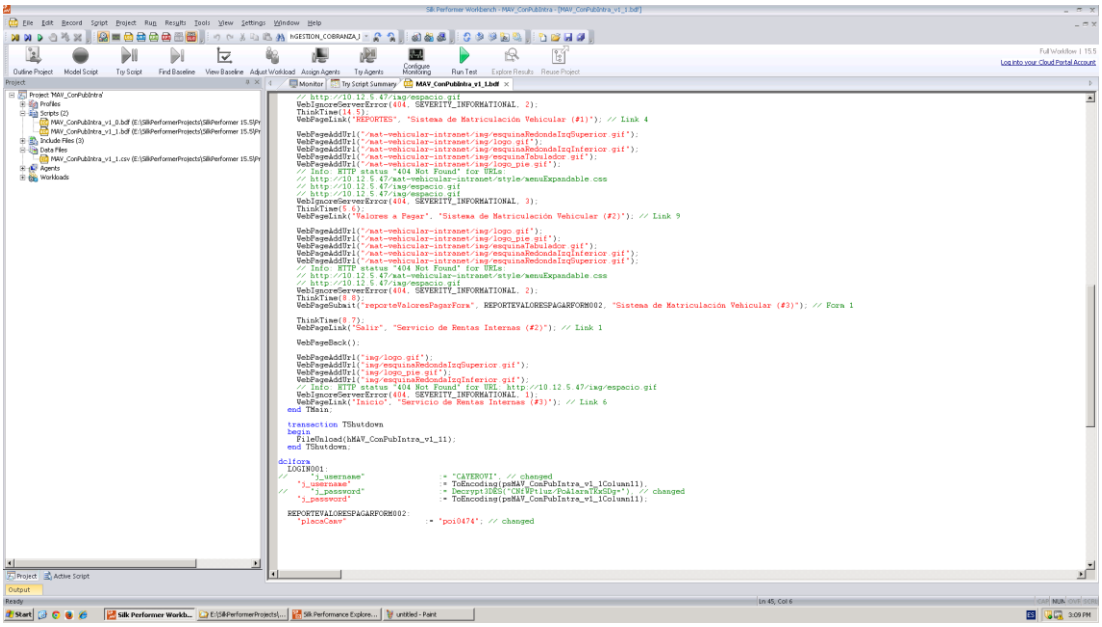


Ilustración 2 Código del script

Después que se tiene un script de la prueba grabada y al cual se ha asignado un nombre, se debe presionar en la barra principal el botón “Try Script” que es una utilidad que permite validar que el script que se grabó esté correcto para que el robot pueda ejecutarlo las veces que se lo configure con los parámetros originales o con nuevos.

En la ventana del Try Script (ilustración 3) muestra cual script se va a validar y con qué perfil, a continuación se puede seleccionar si se requiere que la revisión muestre la secuencia grabada en imágenes (Animated run with TrueLog Explorer) y si se quiere incluir los tiempos muertos que también se han grabado (Enable think times). Por defecto los tiempos muertos no están marcados con el fin de que la validación del script sea inmediata.

El último paso para la validación del script es presionar el botón “Run” de la ventana.

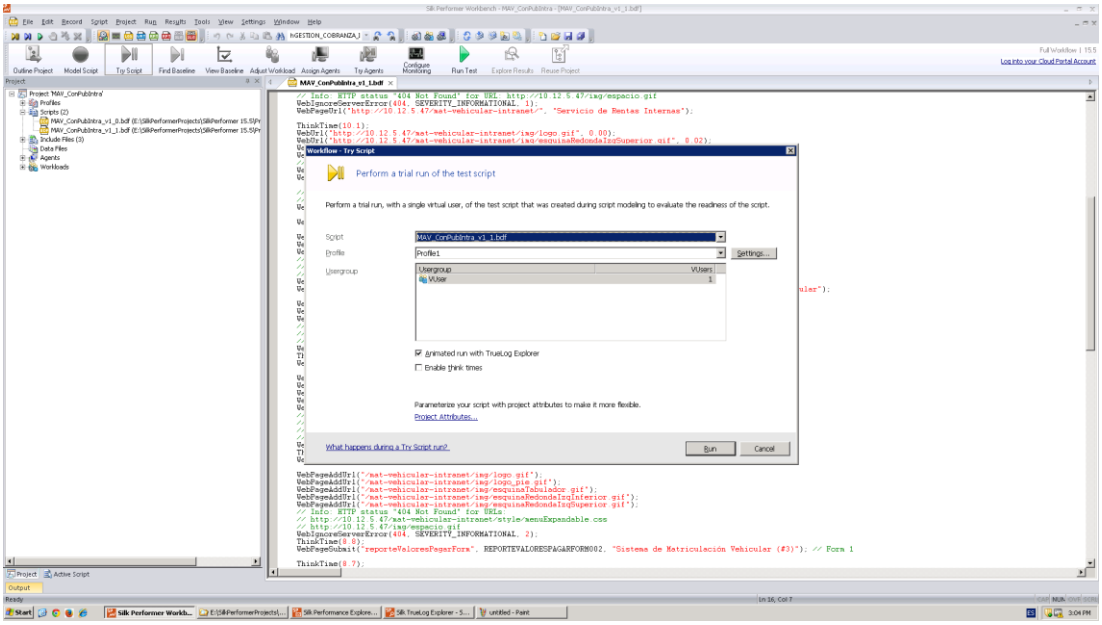


Ilustración 3 Ejecutar la validación del script

Para la validación del script se abre una nueva ventana correspondiente a “SilkTrueLog Explorer” que es un componente de la herramienta SilkPerformer que ayuda a realizar una revisión visual e interactiva de lo que se ha grabado en la prueba. Esto va marcando con un visto en verde si cada paso del script es exitoso y marca en rojo si tiene algún problema.

Como se muestra en la ilustración 4 todos los pasos del script han sido exitosos por lo que se puede realizar a continuación la grabación de la línea base de la prueba.

El concepto de la línea base es muy importante ya que permite tener una referencia para los resultados que se obtendrán con los diferentes tipos de prueba de carga o estrés que se configuren. Por ejemplo: si en la línea base obtenemos tiempos de transacción de 2 segundos se espera tener los mismos tiempos de transacción en las otras pruebas de carga.

Un ejemplo del reporte de una línea base se muestra en la ilustración 5.

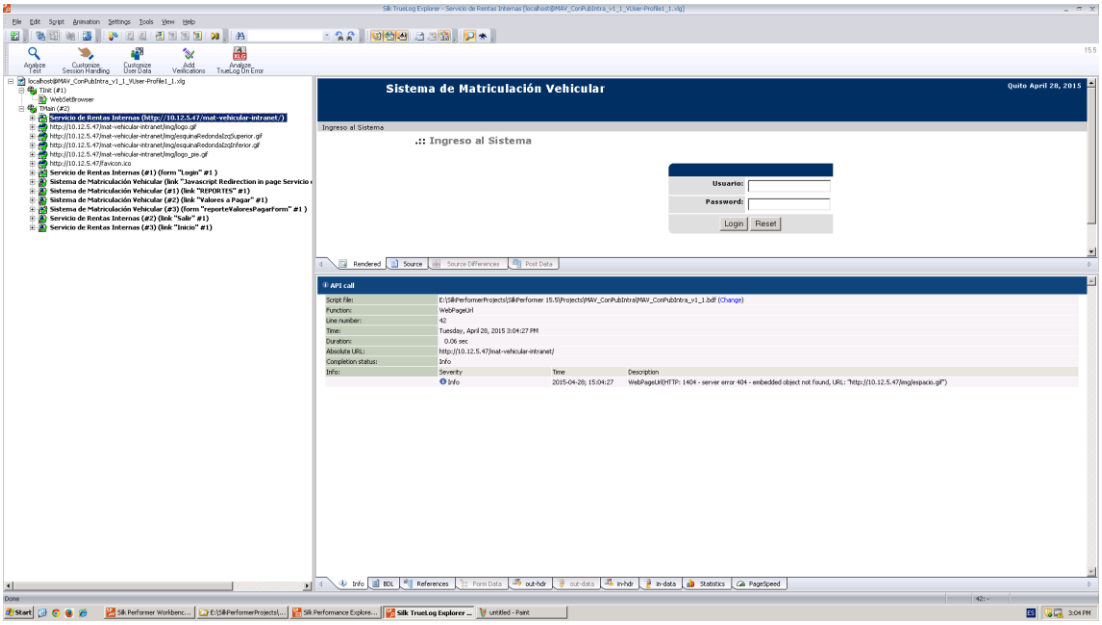


Ilustración 4 Resultado de la validación del script

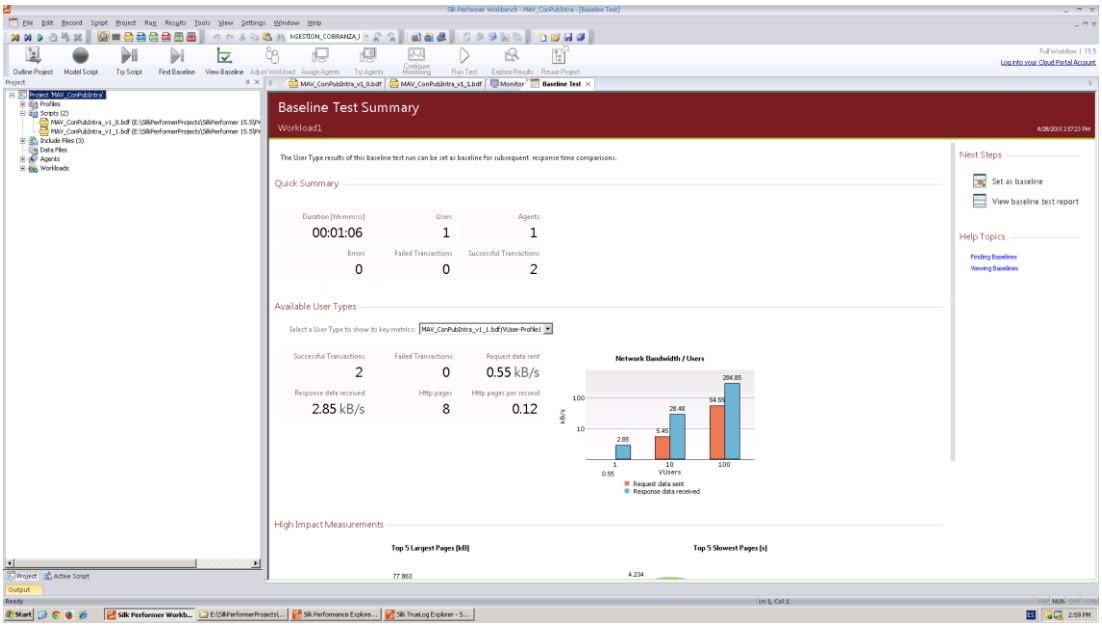


Ilustración 5 Resumen de la prueba de una línea base

Cuando se requiere modificar valores dentro de un script como por ejemplo el usuario y contraseña para que se pueda simular N cantidad de usuarios se requiere ir a la ventana del Try Script y seleccionar en la barra principal la opción “Customize User Data” y a continuación en el panel de la derecha escogemos el objeto HTML que su contenido debe ser reemplazado y se presiona el botón derecho del mouse para escoger la opción

“Customize value”, seguido de esto se abre un asistente para la creación del nuevo parámetro como se muestra en la ilustración 6.

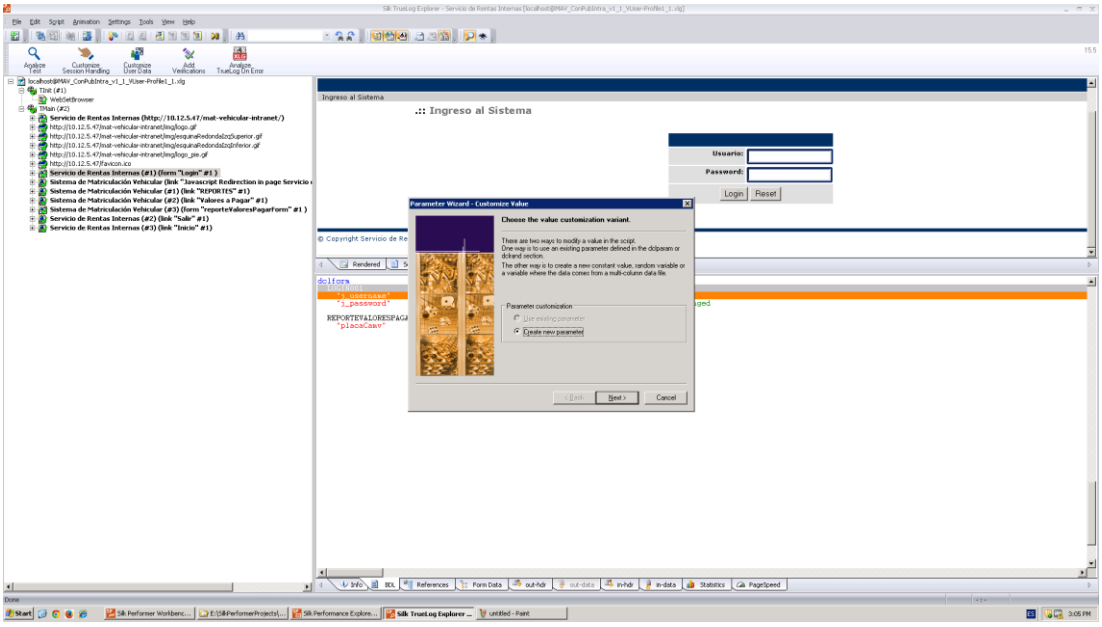


Ilustración 6 Personalización de valores

Al presionar el botón “Next” para crear un nuevo parámetro aparece una ventana que permite seleccionar que tipo de parámetro se va a utilizar (ilustración 7), para el ejemplo se utiliza la última opción que permite ingresar parámetros multicolumna, o sea usuario y contraseña.

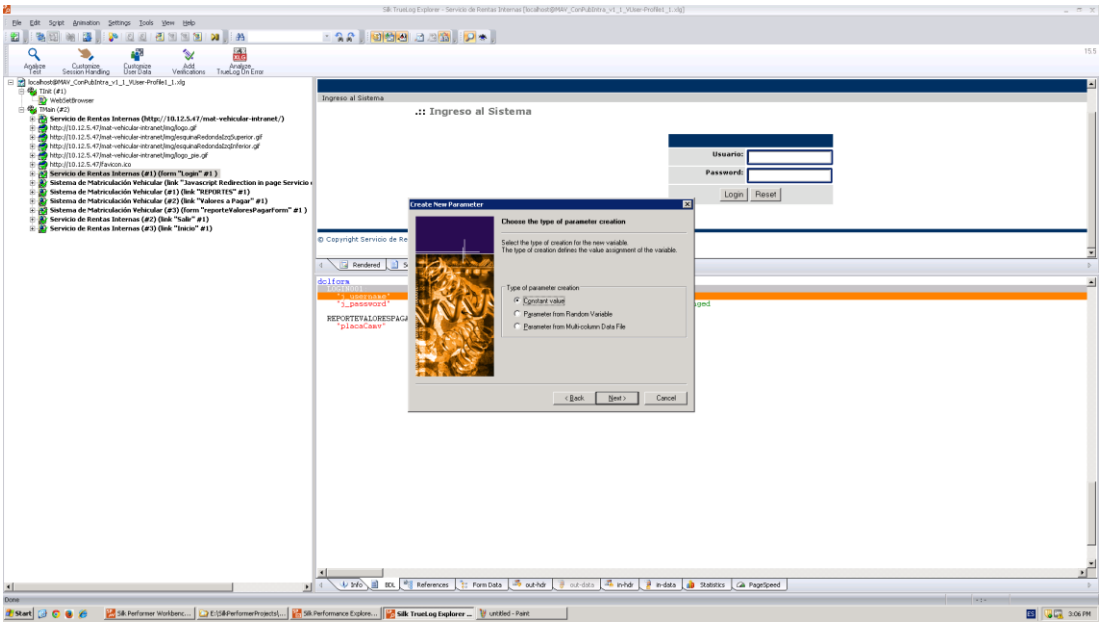


Ilustración 7 Tipo de parámetro

En la ilustración 8 se muestra la nueva ventana donde se ingresan los valores que tomarán los nuevos parámetros definidos por el ingeniero de pruebas, esta ventana permite el ingreso de información como una hoja de cálculo así que se puede copiar y pegar los datos de forma sencilla. Una vez cargada la información se selecciona la columna y se presiona el botón Next.

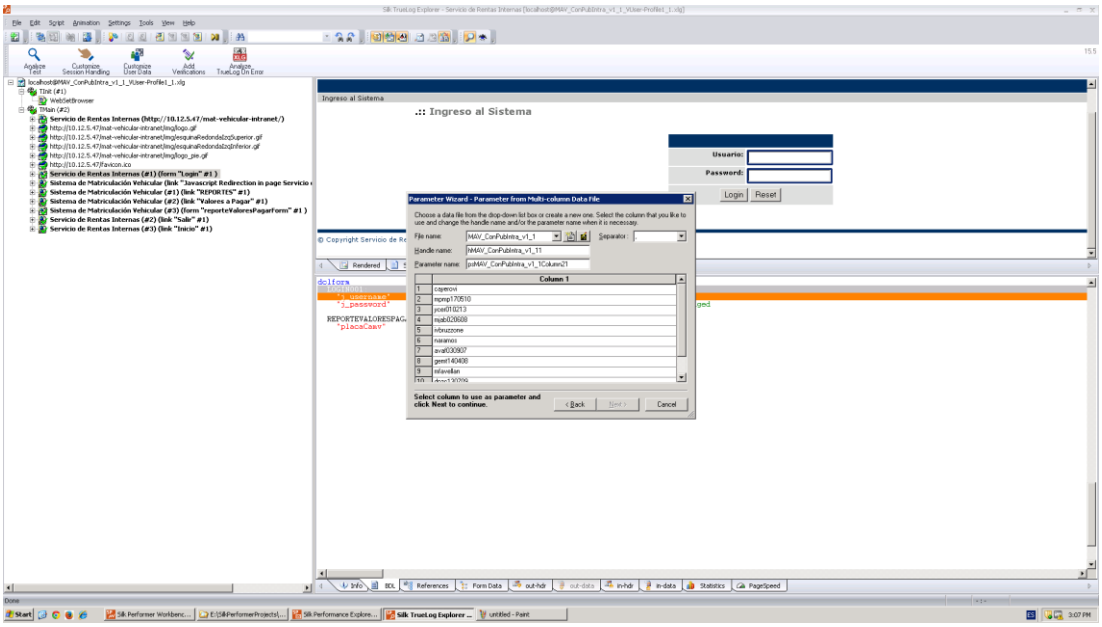


Ilustración 8 Carga de datos para los parámetros

La ventana que aparece a continuación (ilustración 9) requiere que se personalice si el parámetro lo asignará la herramienta de forma aleatoria o secuencial, en este punto son definiciones del ingeniero de pruebas.

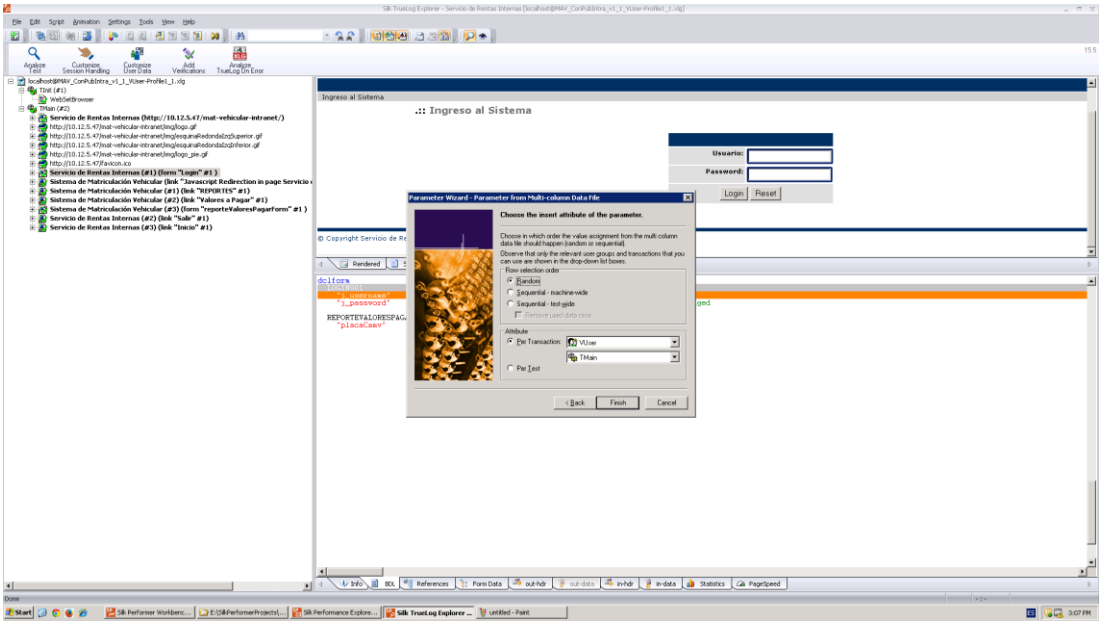


Ilustración 9 Escoger la inserción de los atributos

Al momento de presionar el botón “Finish” la herramienta realizará automáticamente el reemplazo de los valores que se grabaron originalmente por el nuevo parámetro que se definió en la ventana Try Script. Los valores reemplazados se podrán identificar fácilmente en el script ya que la herramienta sólo comenta los originales como se puede observar en la ilustración 10 en la sección del recuadro rojo.

Es recomendable que para cualquier modificación que se realice en el script de pruebas, sea este con los asistentes de SilkPerformer o manualmente, se ejecute nuevamente la opción Try Script de la barra principal de la herramienta ya que con esto garantizamos que las pruebas que contengan más usuarios virtuales no tendrán problemas de ejecución del script y el robot seguirá la secuencia de pasos sin defectos.

Con esto se puede esclarecer la lectura de los errores en el informe final ya que no aparecerán errores por la mala grabación del script sino sólo errores propios de las aplicaciones.

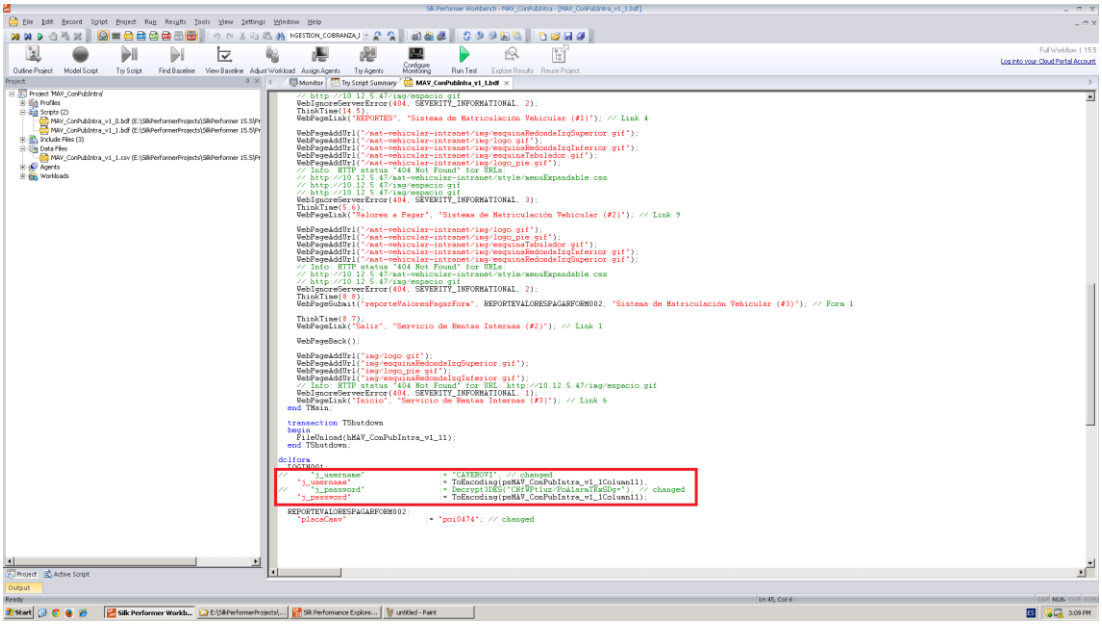


Ilustración 10 Código modificado mediante asistente

Una vez que ya se tiene un script validado y limpio se procede a escoger que tipo de prueba es la que se va a ejecutar, las más comunes son las siguientes:

- **Increasing**, esta prueba permite realizar cargas incrementales de usuarios durante el tiempo de duración de la prueba, llegar a una cantidad de usuarios constantes e ir reduciendo gradualmente la cantidad de usuarios hasta finalizar la prueba. Como se muestra en la ilustración 11.
- **Steady State**, permite realizar una prueba con un número fijo de usuarios virtuales durante todo el tiempo de duración de la prueba.
- **Dynamic**, este tipo de prueba habilita una barra de desplazamiento en la ventana de ejecución de la prueba con el fin de que el ingeniero de pruebas pueda ir simulando cargas muy dinámicas entre los intervalos de usuarios virtuales previamente definidos.
- **All Day**, ejecuta pruebas con el comportamiento de un día de transacciones definido. Por lo general estas pruebas ayudan a generar estrés en las aplicaciones.

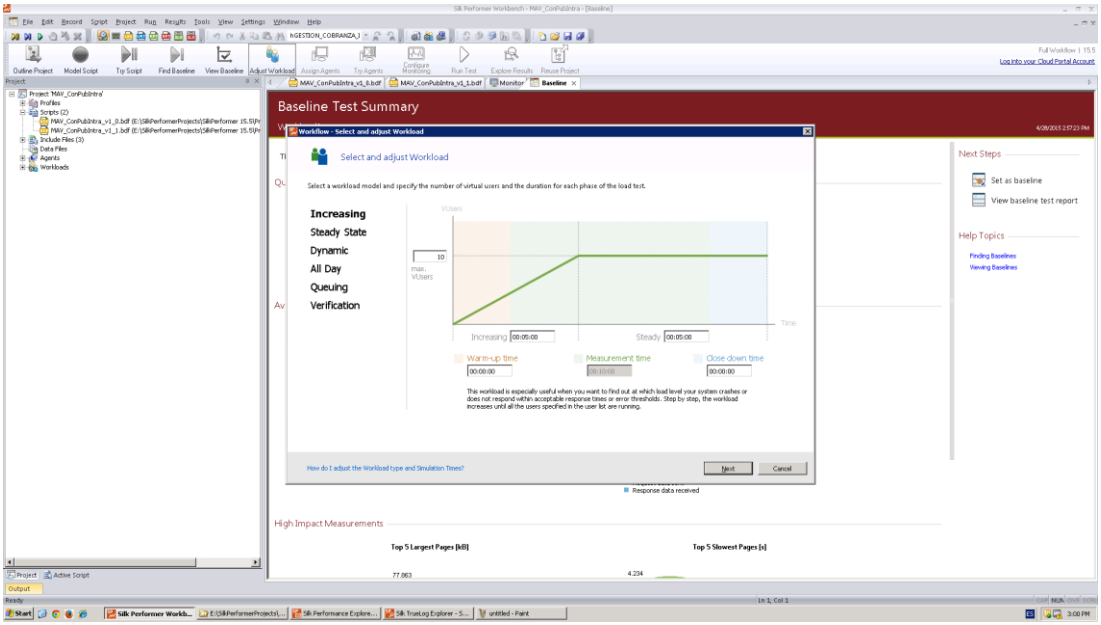


Ilustración 11 Tipos de pruebas de carga

Después de seleccionar el modelo de prueba de carga se debe asignar el número de usuarios virtuales con los que se va a trabajar en la ejecución de la prueba. Es importante tomar en cuenta la cantidad de usuarios virtuales que soporta la licencia instalada.

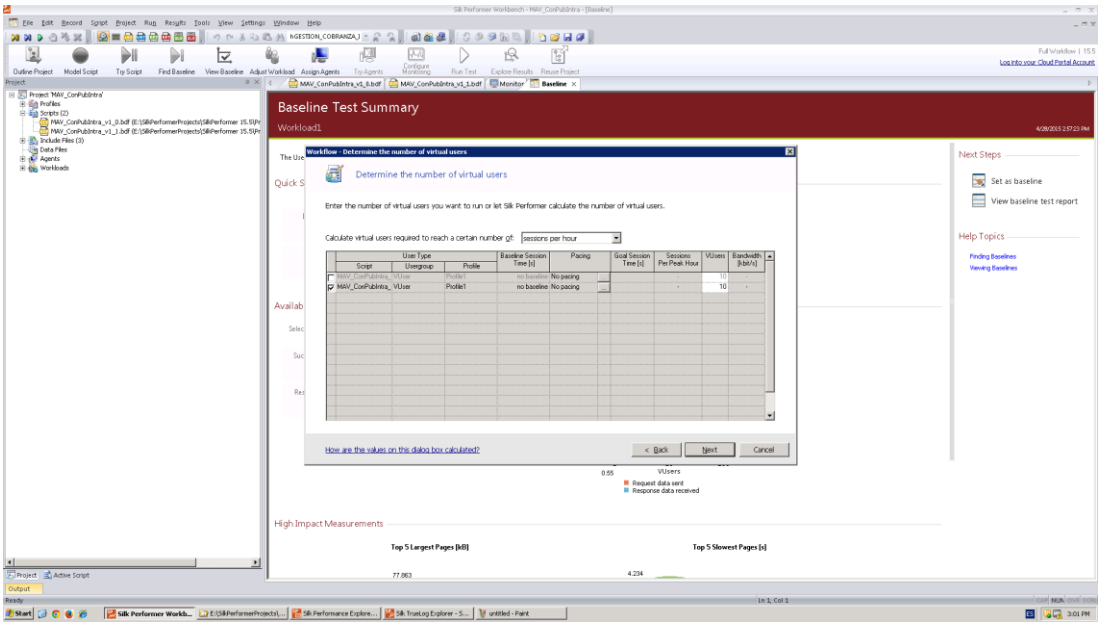


Ilustración 12 Ingreso de usuarios virtuales

La ilustración 13 presenta las opciones que utiliza la herramienta para consumir los usuarios virtules:

- Asignación de agentes manualmente
- Usar el cluster de agentes de SilkCentral
- Usar los agentes de la nube

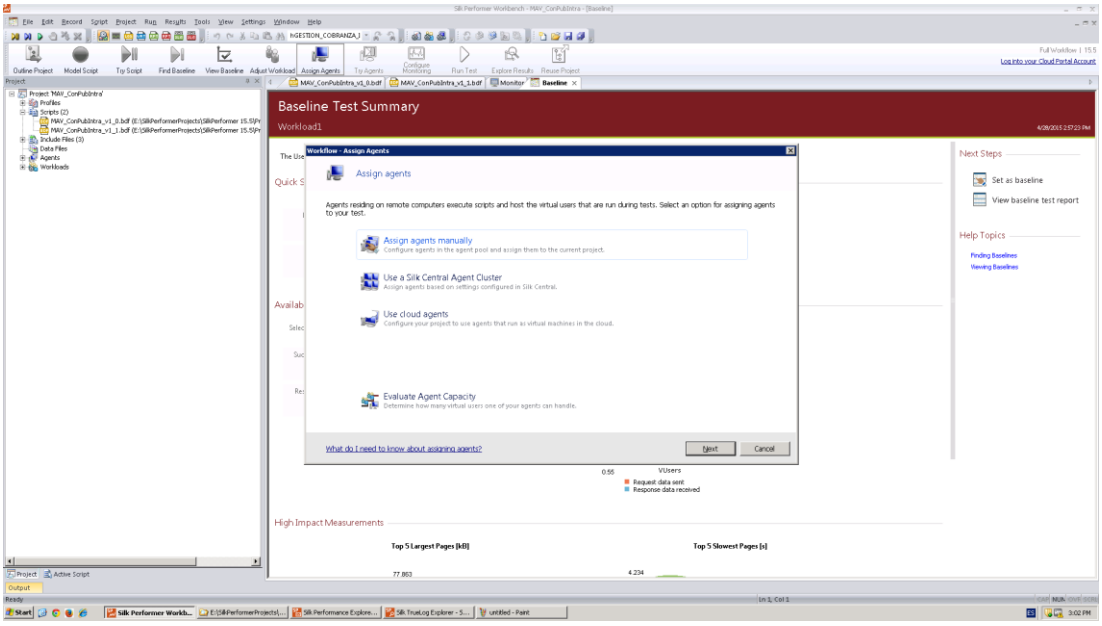


Ilustración 13 Configuración de agentes

La ilustración 14 muestra una ventana donde podemos escoger el script que se desea probar y con qué agente se lo va a ejecutar, con el fin de realizar una validación de los recursos del agente antes de realizar la prueba final.

Un agente se define como un equipo que va a aportar con sus recursos para la ejecución de la prueba de carga, esto sirve para descongestionar los recursos del servidor donde se encuentra instalada la herramienta. Este hecho de descongestionar los recursos es muy importante ya que en los resultados finales se puede presentar falsos negativos debido a consumo de recursos del servidor como saturación de la memoria o procesador sin embargo el servidor de la aplicación a donde se dirige la prueba de carga no ha sido exigido hasta el punto de degradar sus recursos.

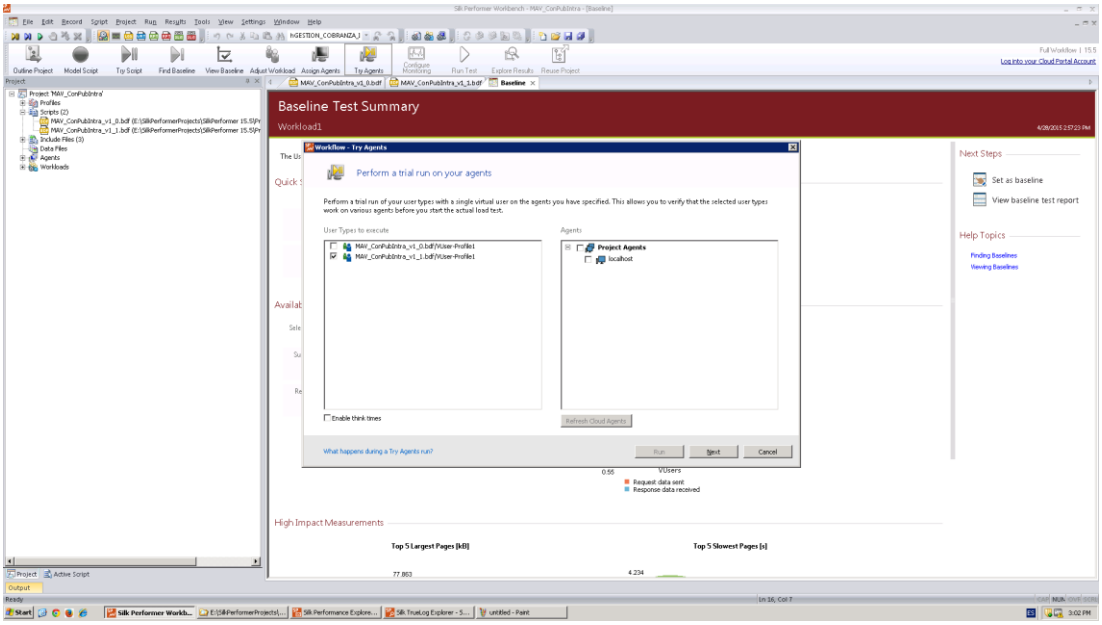


Ilustración 14 Prueba de agentes

El penúltimo paso en el proceso de la configuración de la prueba de carga es la ejecución en sí del script que se ha grabado y validado previamente, como se muestra en la ilustración 15 ya aparece qué modelo de prueba se va a ejecutar, el script, con cuantos agentes y cuantos usuarios virtuales, en este paso se revisa que todo concuerde con lo planificado y se presiona el botón “Run”.

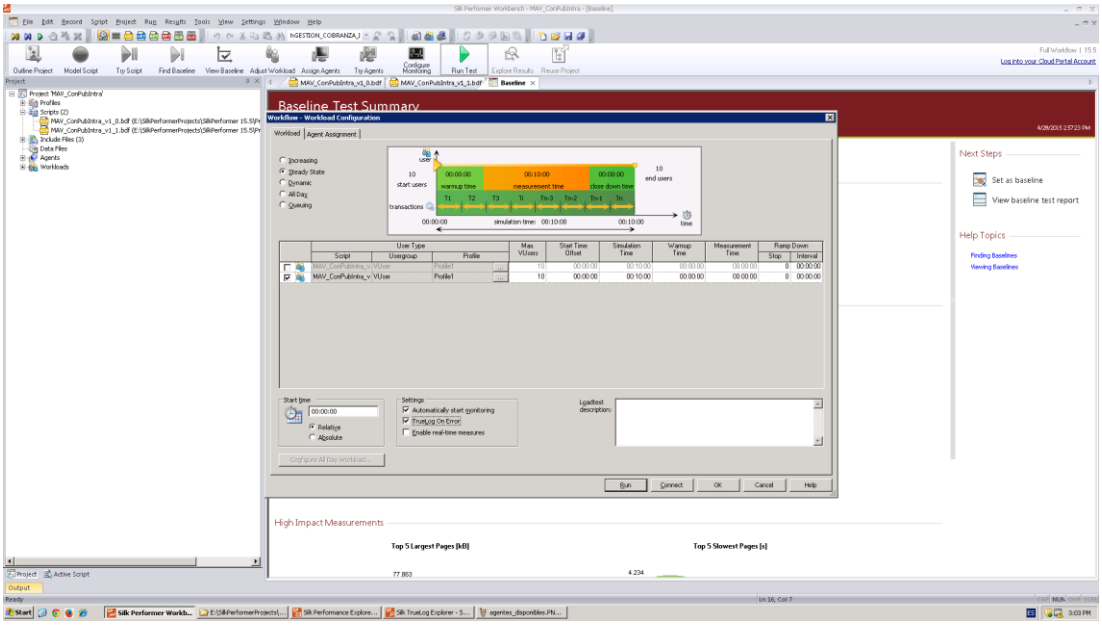


Ilustración 15 Ventana de ejecución de la prueba

Cuando se está ejecutando la prueba se puede ir monitoreando el consumo de recursos del servidor donde está instalada la herramienta, el estado de los usuarios virtuales y agentes e ir revisando si aparecen o no errores durante la ejecución. La ilustración 16 muestra la consola de monitoreo de la prueba de carga.

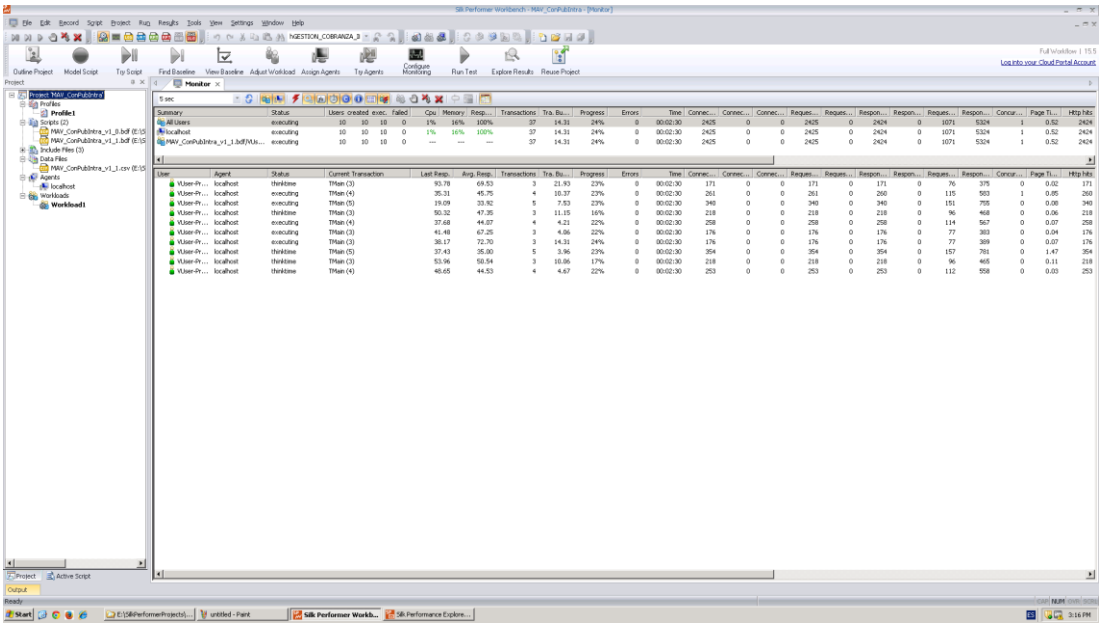


Ilustración 16 Consola de monitoreo

Como parte final en la configuración de la herramienta es importante la interpretación de los reportes que exporta la herramienta, como se puede ver en la ilustración 17 existen datos de la configuración de la prueba como: tiempo de duración, usuarios virtuales, transacciones exitosas, transacciones fallidas, tasa de transferencia y gráficas de errores y tipos de usuarios; esto nos permite ir conociendo el comportamiento de una aplicación en diversos ambientes y condiciones con el fin de ir previniendo cualquier tipo de incidentes al momento de poner la aplicación en el ambiente de producción.

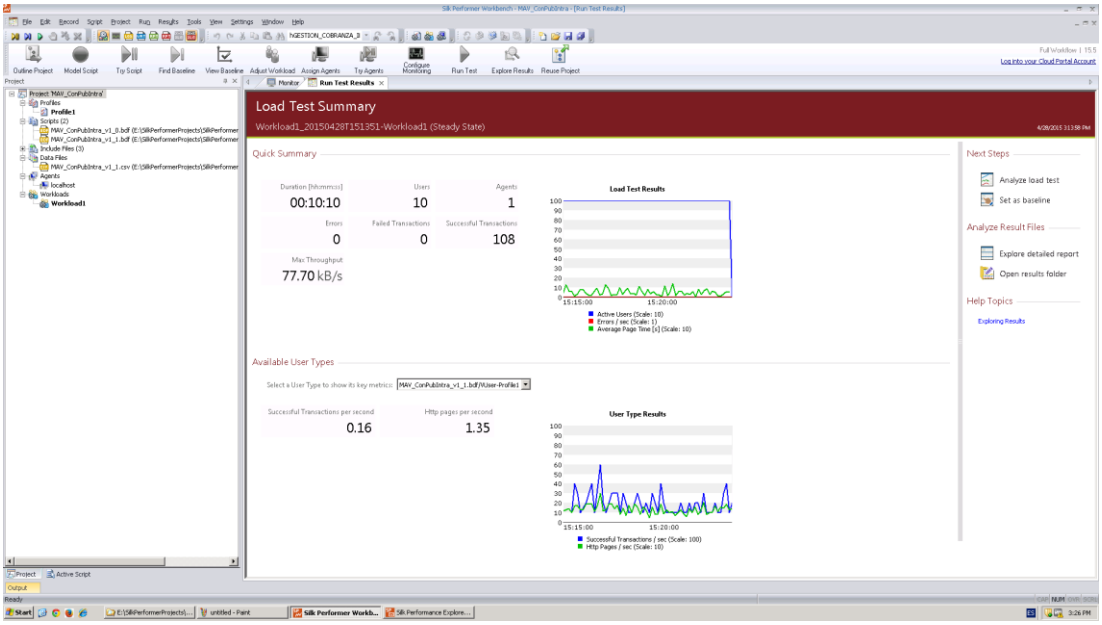


Ilustración 17 Resumen de la prueba de carga

2.2. Configuración de la herramienta en la nube para pruebas de Carga

La herramienta que se utilizó para pruebas de carga en la nube es LoadStorm, con referencia a la configuración de la herramienta SilkPerformer esta herramienta es mucho más sencilla de configurar pero no ofrece la robustez ni la flexibilidad para realizar parametrizaciones en los valores constantes de la prueba.

La ilustración 18 muestra el esquema que se debe seguir para la configuración de una prueba de carga en la nube con la herramienta LoadStorm:

Ilustración 18 Pasos para realizar una prueba de carga en la nube

Al momento de ingresar a la aplicación LoadStorm en la web se debe crear un nuevo plan para la ejecución de la prueba como se muestra en la ilustración 19.

Se ingresa el nombre, una descripción y se presiona el botón “Save”.

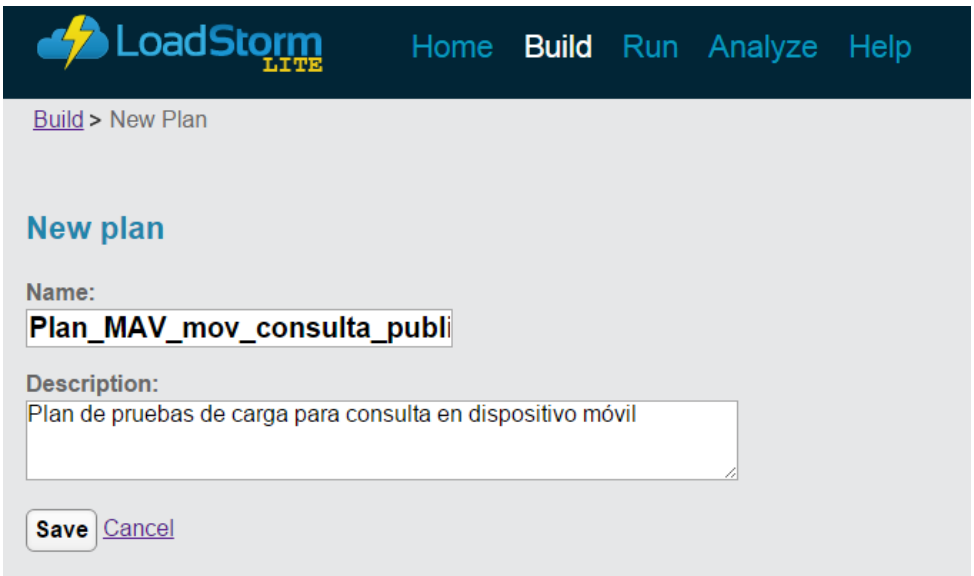


Ilustración 19 Definición del plan

La ilustración 20 muestra la ventana donde se indica que el nuevo plan ha sido grabado exitosamente.

Ilustración 20 Grabación del plan

Una vez que el plan ha sido grabado se debe crear un nuevo escenario de prueba, esto consiste en llenar un formulario donde se ingresa el nombre del escenario, una descripción, ponderación, tiempos de pausa, la cantidad de datos que utiliza la página que vamos a probar y si se ejecutan javascripts y descargan imágenes.

La ilustración 21 muestra el formulario antes mencionado.

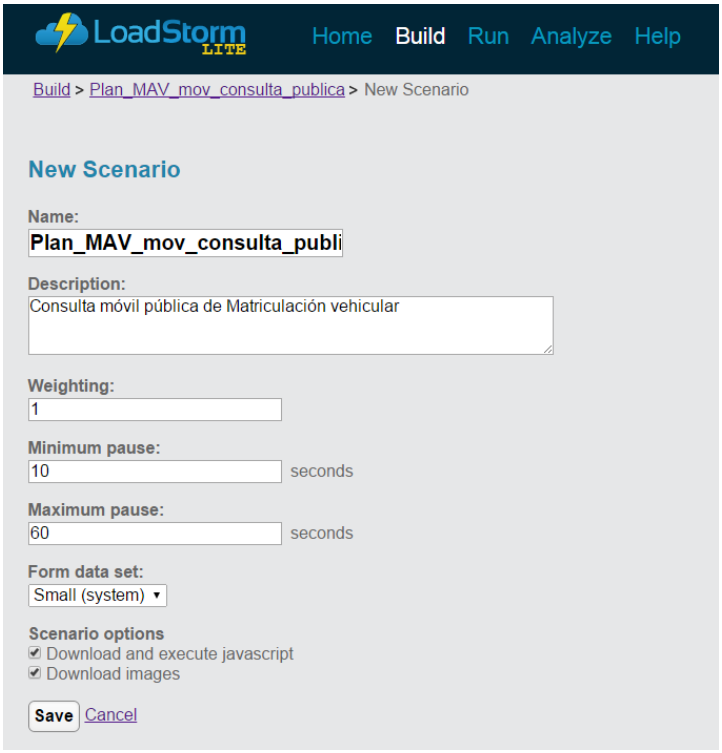


Ilustración 21 Creación del escenario

La ilustración 22 muestra que el escenario se ha grabado correctamente y cuál fue su configuración.

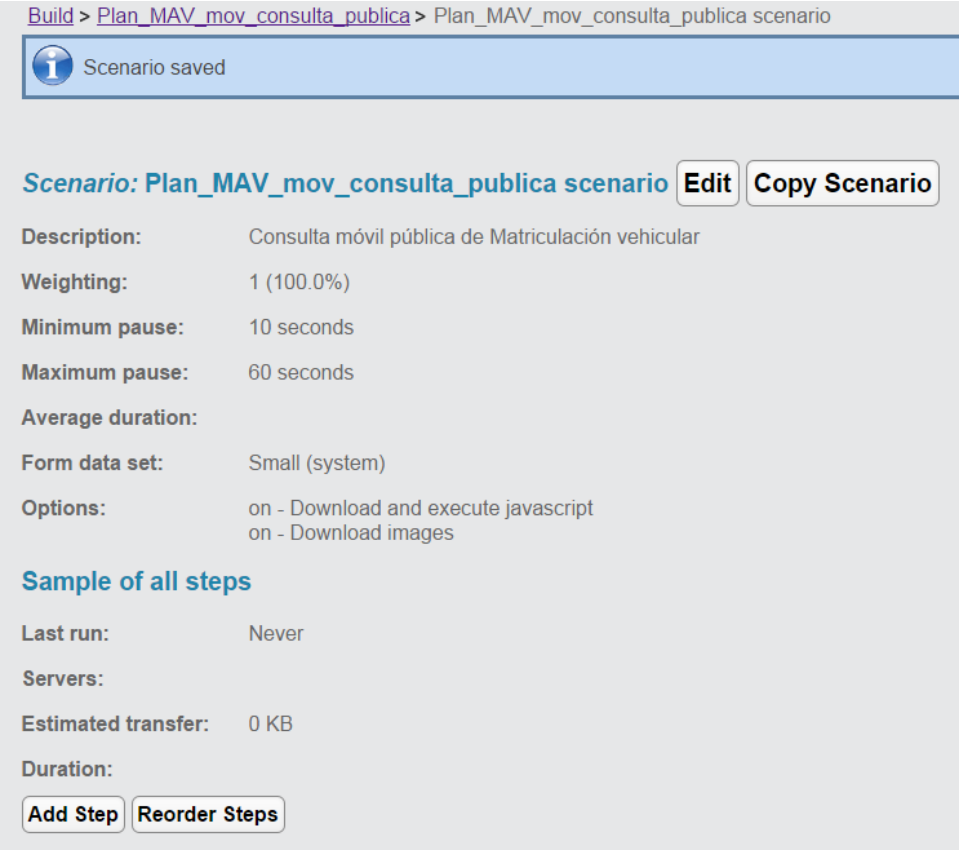


Ilustración 22 Grabación del escenario

El paso siguiente es la configuración del servidor y de la aplicación que se van a probar.

Como se muestra en la ilustración 23 se indica cómo va a arrancar la aplicación, en este caso se despliega en un navegador por lo que se selecciona la opción “Open a page” y a continuación se le indica el contexto de la aplicación y el nombre del servidor donde está alojada para que la herramienta pueda ir leyendo los componentes que van a formar la prueba de carga.

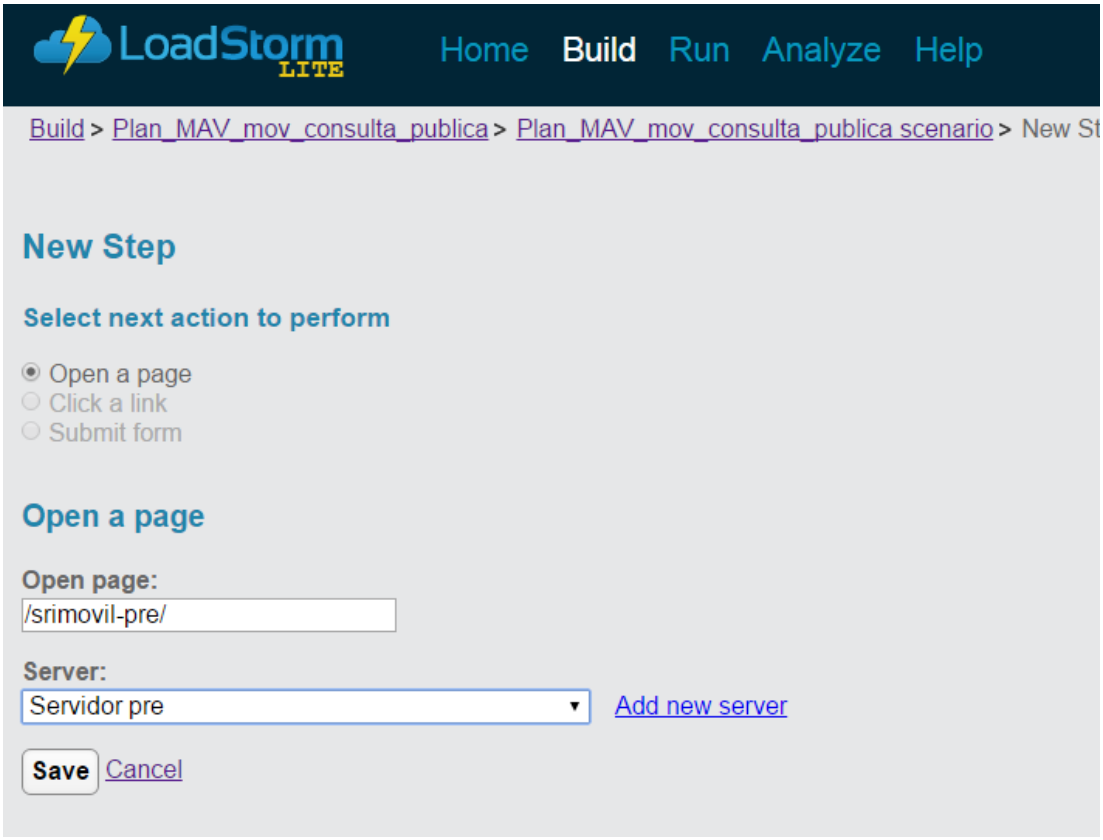


Ilustración 23 Selección de aplicación para la grabación

La ilustración 24 muestra como se despliega los componentes de la página web configurada, los tiempos de carga de la página, el tamaño de la transferencia.

Dentro de la ventana que se muestra en la herramienta se va a ir seleccionando los eventos que memorizará para que sean repetidos en la ejecución de la prueba de carga. Esta información se la puede utilizar como línea base para ir realizando comparaciones con los resultados finales.

Algo interesante en esta herramienta es que cuando se va a grabar un nuevo paso y este se lo va a realizar por medio de el evento clic del mouse ya muestra un listado de enlaces existentes sobre los cuales se puede ejecutar dicho evento con el fin de ser más precisos en la selección de los objetos.

La ilustración 25 presenta los enlaces sobre los cuáles se puede ejecutar el evento de clic.

Ilustración 24 Carga de aplicación

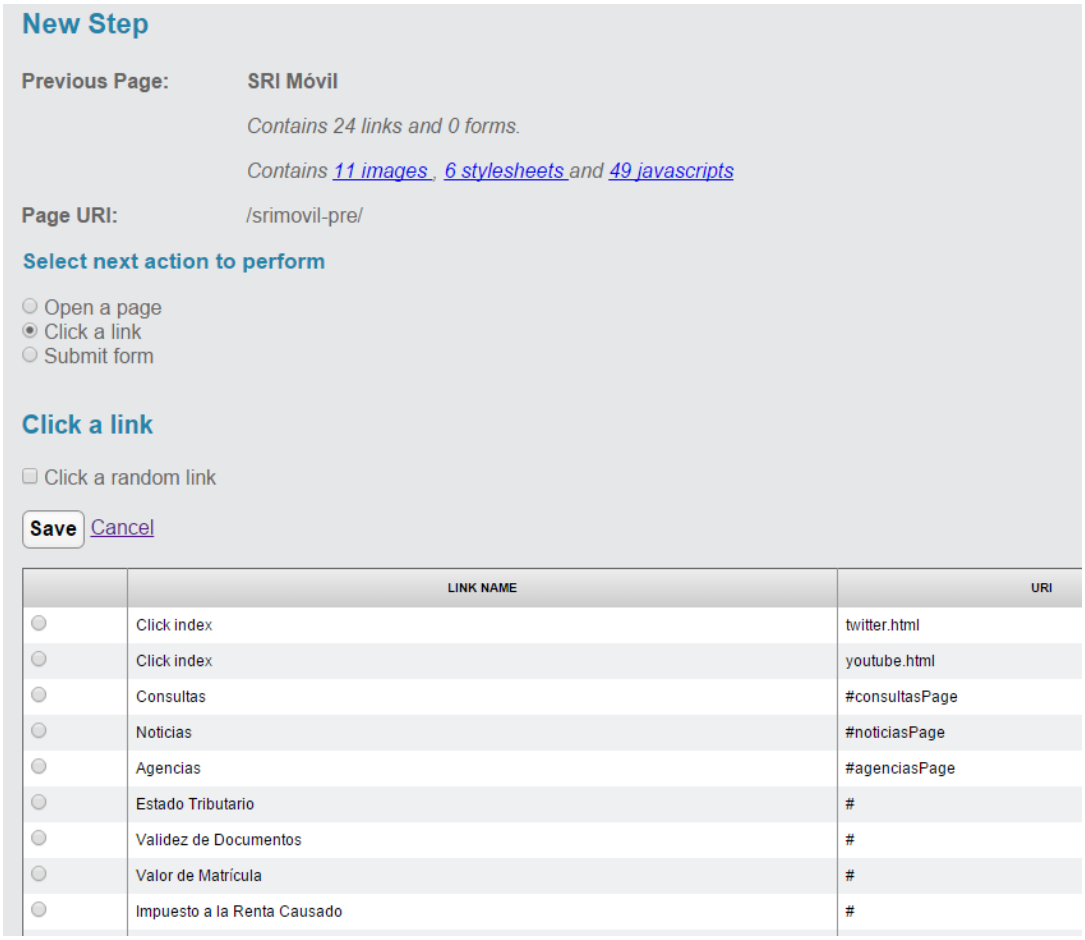


Ilustración 25 Selección del paso a ejecutar

Una vez que se ha finalizado la grabación de todos los eventos de la prueba se procede a cerrar la misma, se la calendariza para cuando se quiere ejecutar y automáticamente se dispara la prueba en la fecha y hora escogida.

La información de los resultados obtenidos se graba en una carpeta de la misma herramienta para que puedan ser leídos en cualquier momento.

2.3. Configuración de la herramienta en sitio para pruebas de Seguridad

La herramienta que se utilizó para realizar pruebas de Seguridad en sitio es IBM Rational Appscan, a continuación se describen algunos de los pasos para la configuración de una prueba.

En la ilustración 26 se observa la ventana inicial para crear un nuevo escaneo para una aplicación, aquí se selecciona el enlace “Create New Scan…”.

Ilustración 26 IBM Rational Appscan – Página inicial

El siguiente paso es escoger una plantilla predeterminada para continuar con la configuración de la prueba, para este caso se seleccionó el escaneo regular, se debe seleccionar el enlace “Regular Scan” como se muestra en la ilustración 27.

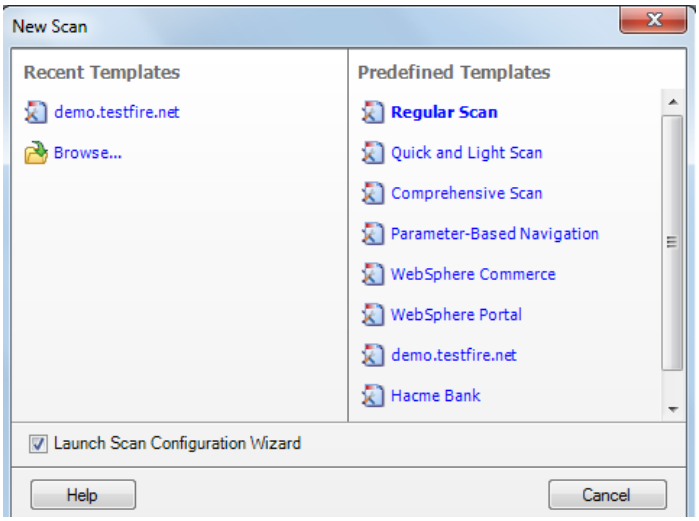


Ilustración 27 IBM Rational Appscan – Plantilla predeterminada

Una vez escogida la plantilla se procede a ingresar la dirección de la página web que se desea evaluar, adicional se configura si existen

servidores adicionales con los que se interactúa para que la herramienta no los detecte como intrusos. La ilustración 28 muestra la página mencionada.

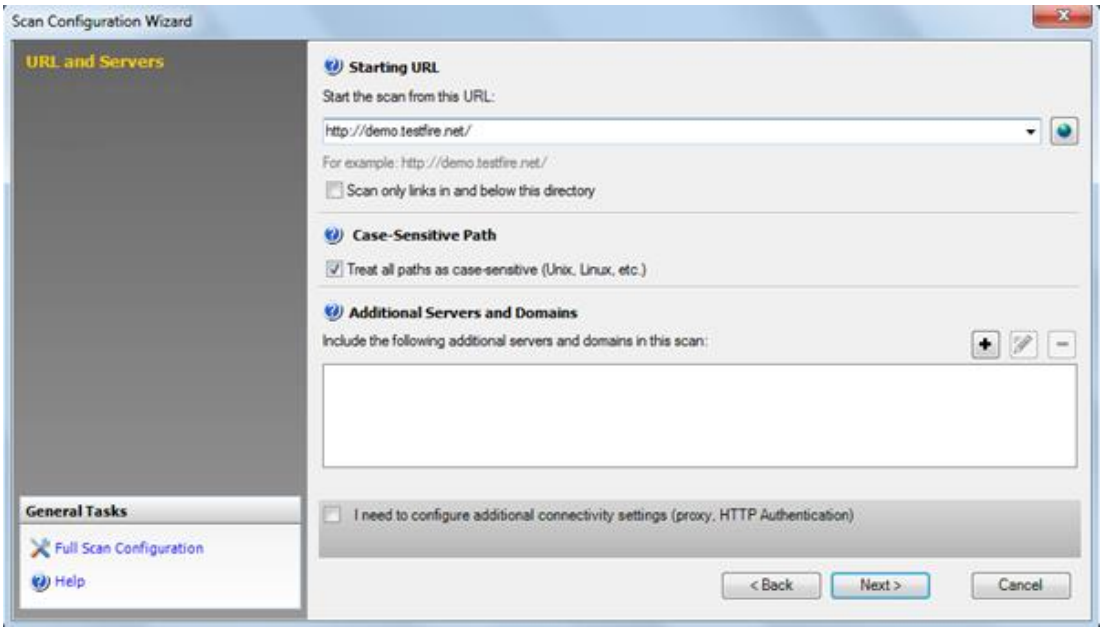


Ilustración 28 IBM Rational Appscan – Ingreso de aplicación

El siguiente paso como se muestra en la ilustración 29 es el tipo de ingreso que tiene el usuario en la aplicación, se lo deja en la opción “Recorded” porque se va a utilizar un ingreso validado por usuario y contraseña.

La herramienta detecta automáticamente las páginas que utiliza la aplicación para poder ingresar y las muestra a modo de lista en un panel de la derecha.

En casos que la aplicación tenga implementados códigos CAPTCHA se recomienda utilizar la opción “Prompt”.

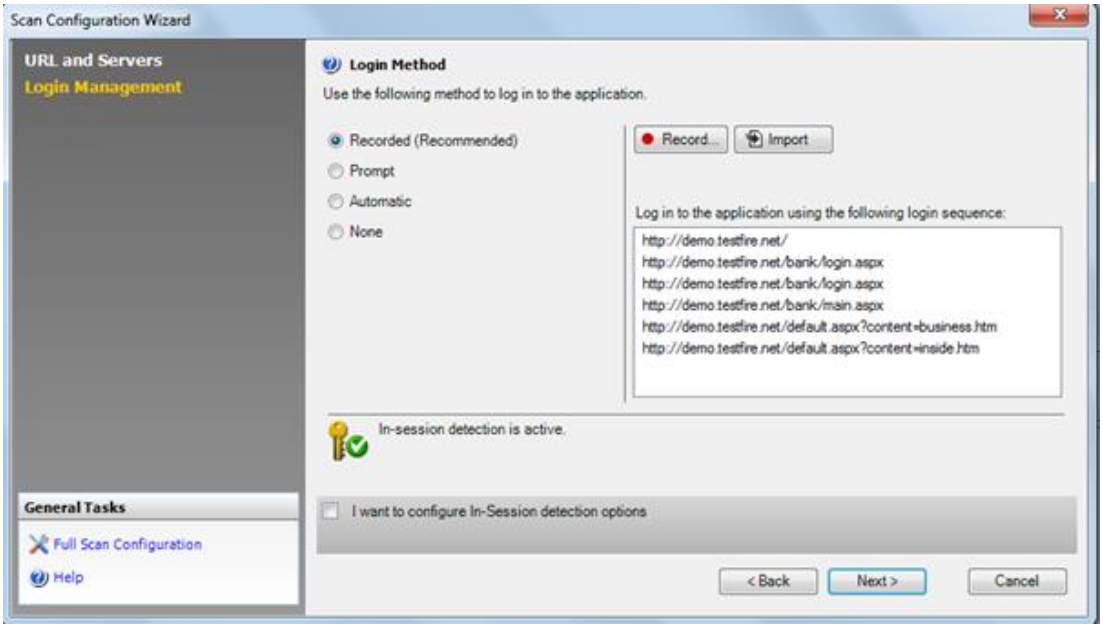


Ilustración 29 IBM Rational Appscan – Método de ingreso

El siguiente paso es indicar que tipo de políticas va a revisar la aplicación, en este caso se selecciona la opción por defecto, ya que en esta se encuentran validaciones exigidas a nivel internacional y que son estándar en las revisiones de seguridad de las aplicaciones. La ilustración 30 muestra esta ventana.

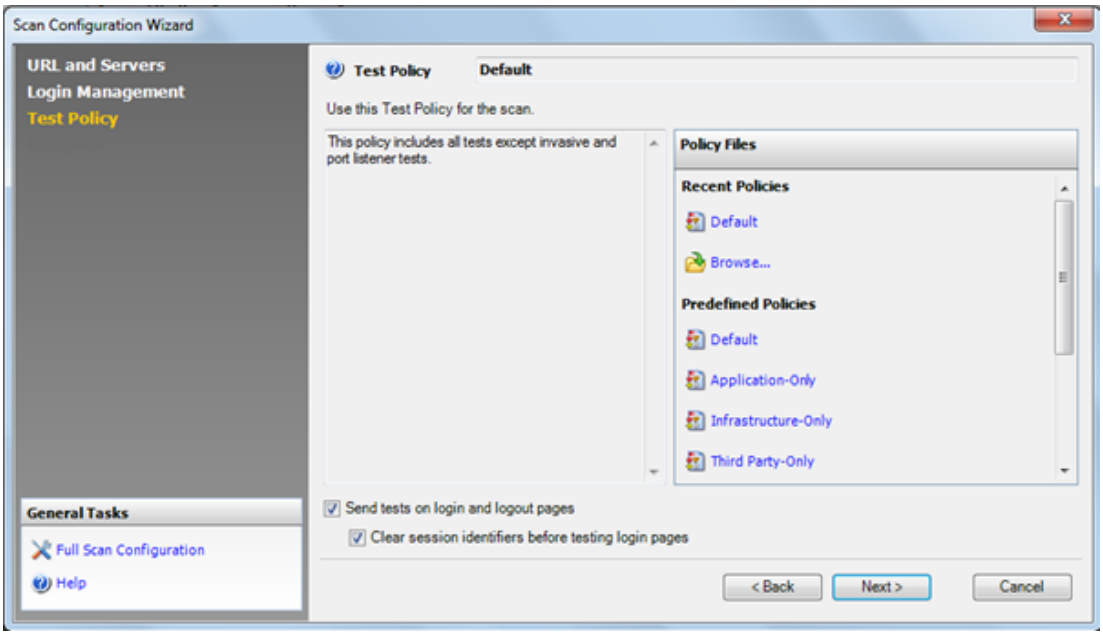


Ilustración 30 IBM Rational Appscan – Políticas de prueba

Como paso final se define como va a iniciar el análisis, si va a ser automático, manual o si se lo va a ejecutar en otro momento. Se selecciona el que se requiera y se presiona el botón “Finish”. La ilustración 31 muestra la pantalla final para la configuración del escaneo de vulnerabilidades.

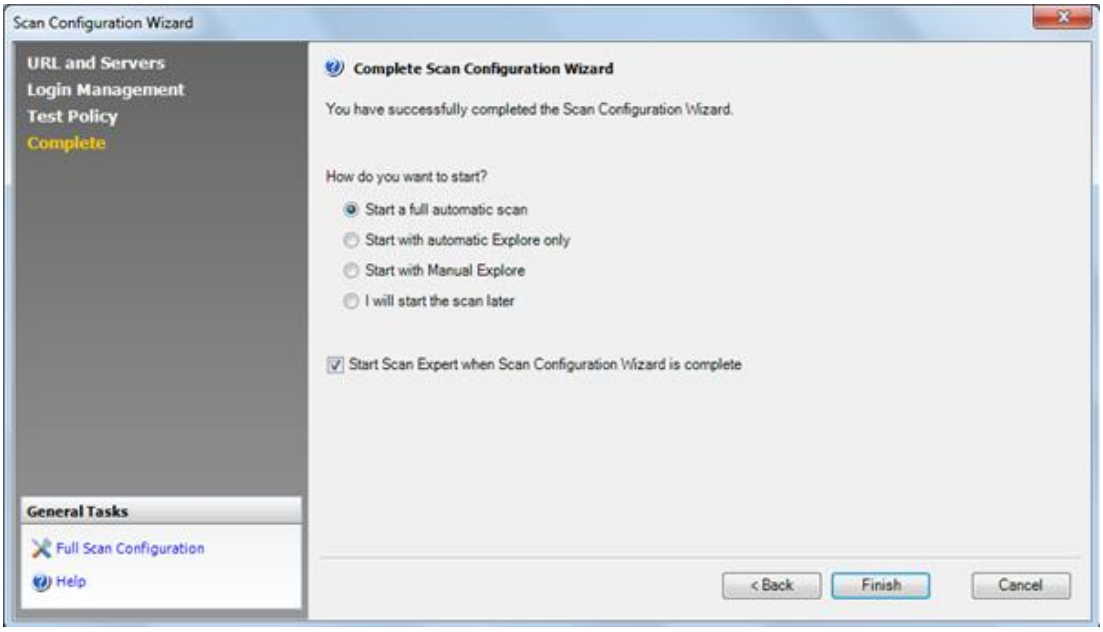


Ilustración 31 IBM Rational Appscan – Finalizar configuración

3. CAPÍTULO III

PRUEBAS NO FUNCIONALES APLICACIÓN INTRANET

3.1. Pruebas de carga en sitio – Intranet

El producto de software que se utilizó para la prueba de carga en sitio corresponde a una aplicación web que se encuentra levantada en una Intranet empresarial, y consta de las siguientes características:

Tabla 1 Características de aplicación Intranet

Característica	Nombre
Servidor de aplicaciones	JBOSS 4.3
Servidor WEB	Apache
Base de datos	Oracle 11g
Lenguaje de programación	Java
JDK	5 release 22

El flujo que se grabó de esta aplicación es el siguiente:

- Ingreso a la aplicación con usuario con perfil de administrador y su contraseña.
- Selección del menú de Reportes.
- Selección de la opción Consulta de Valores a Pagar.
- Ingreso de la placa de un vehículo.
- Presionar el botón consultar.
- Visualización de los valores a pagar correspondientes a matrícula e impuestos del vehículo.
- Presionar en el enlace Salir para que se libere la sesión del usuario caso contrario no se puede ingresar nuevamente con el mismo usuario.

La información que muestra la herramienta en sitio es muy completa especifica varios indicadores que permiten realizar un análisis muy completo de los resultados de las pruebas, a continuación en la Ilustración 32 se

muestran tres gráficos de control sobre la prueba realizada. Para mayor información se puede ir al Anexo “8.1. Reporte pruebas de carga en sitio aplicación ambiente Intranet” donde se encuentra el informe completo de la prueba realizada.

La primera gráfica muestra que el tipo de prueba tuvo una concurrencia de 10 usuarios virtuales durante toda su ejecución.

La segunda gráfica muestra la cantidad de transacciones por segundo que se realizaron en el transcurso de la prueba, en total se realizaron 108 transacciones con un promedio de 0,18 transacciones por segundo.

La tercera gráfica muestra la cantidad de errores que aparecieron durante la ejecución de la prueba, en este caso no hubieron errores.

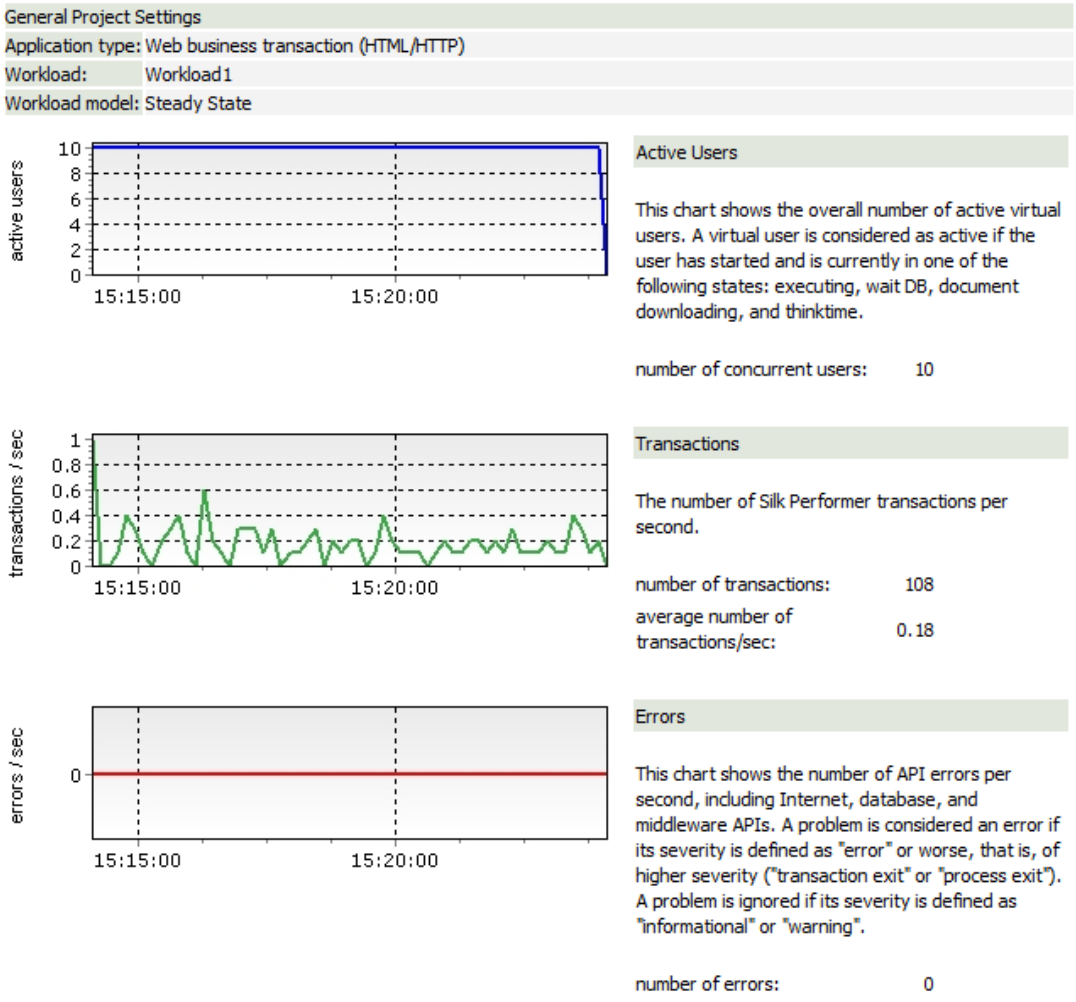


Ilustración 32 Gráfica de control SilkPerformer – Intranet

3.2. Pruebas de carga en la nube - Intranet

La herramienta para pruebas de carga en la nube LoadStorm, no permitió realizar la configuración del registro de los usuarios para validar la aplicación de intranet.

Si no se podía convertir el valor constante del primer usuario a una lista de usuarios la misma sesión no se podía abrir más de una vez de forma simultánea.

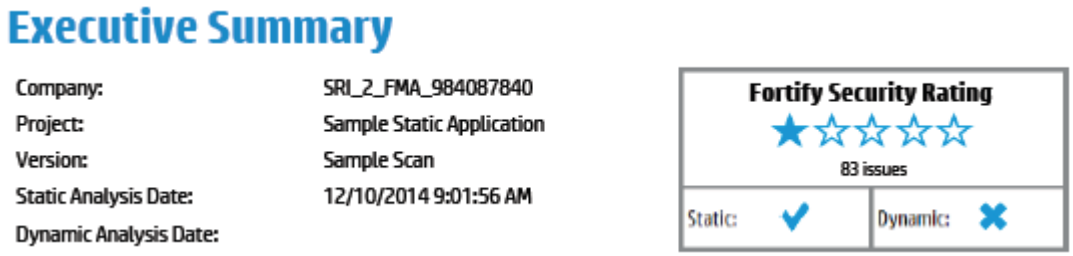
3.3. Pruebas de seguridad en la nube – Intranet

La prueba de seguridad fue realizada a la misma aplicación Intranet que se realizaron las pruebas de carga en el apartado “2.1. Pruebas de carga en sitio – Intranet”.

La herramienta que se utilizó para estas pruebas de seguridad fue HP Fortify on Demand, básicamente se ejecutó el mismo flujo programado en la prueba de carga.

En la ilustración 33 se muestra una parte del reporte que emite la herramienta, si se requiere ver el reporte completo se debe ir al Anexo “8.2. Reporte pruebas de seguridad en sitio aplicación ambiente Intranet”.

En la tabla de la ilustración 33 se puede observar que el informe se encuentra estructurado de la siguiente manera: Rating o criticidad del incidente encontrado (crítico, alto, medio, bajo incluso en colores diferentes para una mejor lectura), la categoría del incidente, el tipo de prueba ejecutada (estática o dinámica) y la cantidad de veces que se han repetido los mismos errores.



Issue Breakdown

Issues are divided based on their impact (potential damage) and likelihood (probability of identification and exploit).

High impact / high likelihood issues represent the highest priority and present the greatest threat.

Low impact / low likelihood issues are the lowest priority and present the smallest threat.

See Appendix for more information.

Rating	Category	Test Type	Instance Count
Critical	Cross-Site Scripting: DOM	Static	4
Critical	Cross-Site Scripting: Reflected	Static	8
Critical	Dangerous File Inclusion	Static	2
Critical	Path Manipulation	Static	2
Critical	Privacy Violation	Static	5
Critical	SQL Injection: Hibernate	Static	4
Critical	SQL Injection	Static	4
High	Command Injection	Static	6
High	Header Manipulation	Static	1
High	Log Forging	Static	2
High	Password Management: Password in Configuration File	Static	1
High	Path Manipulation	Static	1
High	Portability Flaw: File Separator	Static	1
High	Privacy Violation	Static	2
High	Unreleased Resource: Database	Static	13
High	Unreleased Resource: Sockets	Static	3
High	Unreleased Resource: Streams	Static	6
Low	Cookie Security: Cookie not Sent Over SSL	Static	3
Low	Cross-Site Request Forgery	Static	15

Ilustración 33 Informe de issues HP Fortify – Intranet

4. CAPÍTULO IV
PRUEBAS NO FUNCIONALES APLICACIÓN INTERNET

4.1. Pruebas de carga en sitio – Internet

El producto de software que se utilizó para la prueba de carga en sitio corresponde a una aplicación web que se encuentra levantada en Internet, y consta de las siguientes características:

Tabla 2 Características de aplicación Internet

Característica	Nombre
Servidor de aplicaciones	JBOSS 4.3
Servidor WEB	Apache
Base de datos	Oracle 11g
Lenguaje de programación	Java
JDK	5 release 22

El flujo que se grabó de esta aplicación es el siguiente:

- Ingreso a la página web publicada en Internet.
- Selección del menú Consultas Públicas.
- Selección de la opción Consulta de Valores a Pagar.
- Ingreso de la placa de un vehículo.
- Presionar el botón consultar.
- Visualización de los valores a pagar correspondientes a matrícula e impuestos del vehículo.

La información que muestra la herramienta en sitio es muy completa especifica varios indicadores que permiten realizar un análisis muy completo de los resultados de las pruebas, a continuación en la Ilustración 34 se muestran tres gráficos de control sobre la prueba realizada. Para mayor información se puede ir al Anexo “8.3. Reporte pruebas de carga en sitio aplicación ambiente Internet” donde se encuentra el informe completo de la prueba realizada.

La primera gráfica muestra que el tipo de prueba fue incremental, cargando de 5 en 5 los usuarios virtuales hasta estabilizarse en 20.

La segunda gráfica muestra la cantidad de transacciones por segundo que se realizaron en el transcurso de la prueba, en total se realizaron 1440 transacciones con un promedio de 0,72 transacciones por segundo.

La tercera gráfica muestra la cantidad de errores que aparecieron durante la ejecución de la prueba, en este caso no hubieron errores.

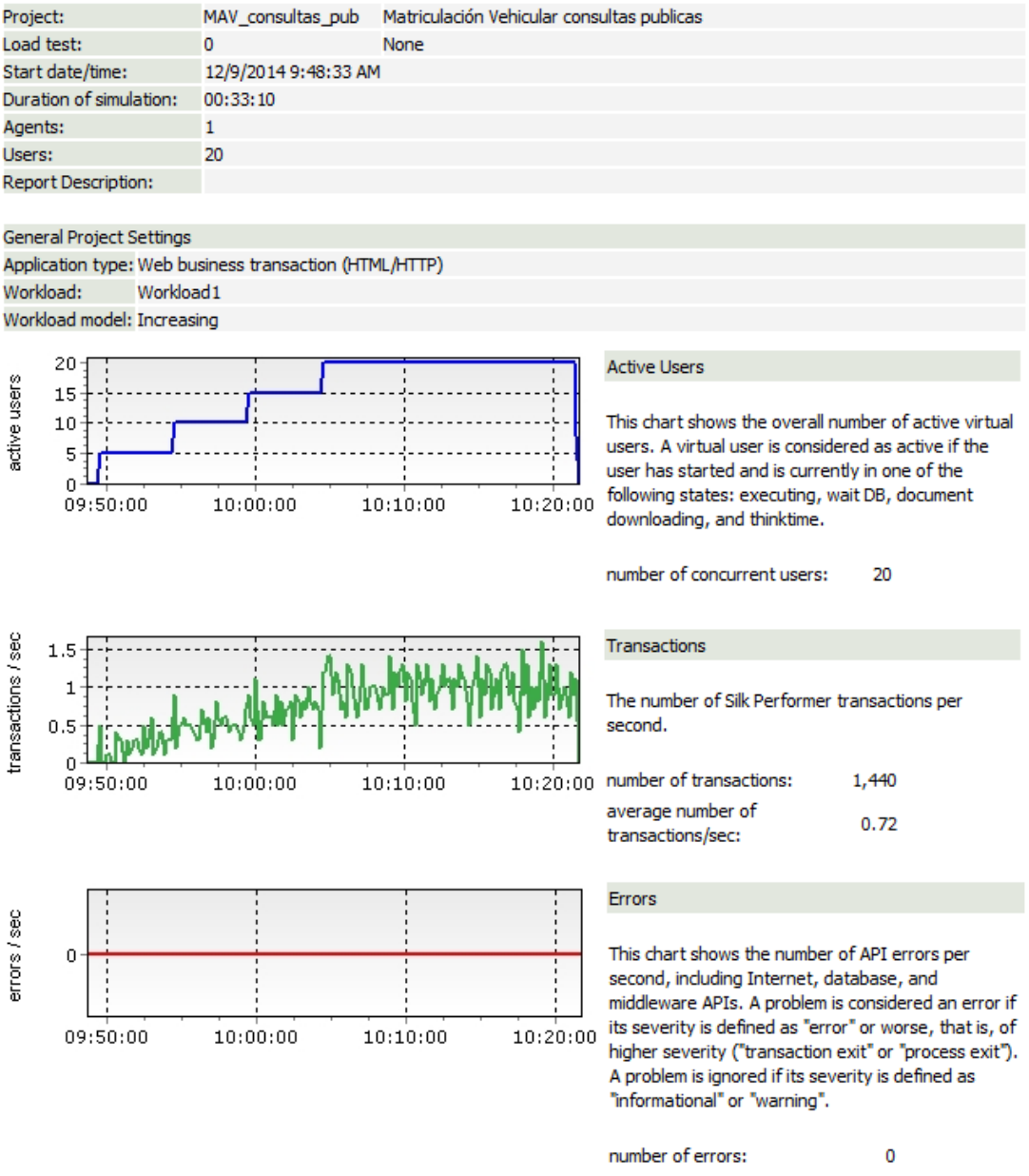


Ilustración 34 Gráfica de control SilkPerformer – Internet

4.2. Pruebas de carga en la nube – Internet

El producto de software utilizado para esta prueba en la nube es el mismo que se utilizó para hacer las pruebas de carga en sitio, sólo se aplicó una configuración adicional en el servidor Apache para que la herramienta LoadStorm pueda grabar los eventos.

En la tabla 3 se describe la configuración que se utilizó al momento de ejecutar la prueba, fue la misma configuración que la prueba de carga con SilkPerformer.

Tabla 3 Configuración de la prueba en la nube - Internet

Característica	Descripción
Duración	20 minutos
Descripción	Prueba de carga, 20 usuarios, de 5 en 5
Patrón de carga	Carga 5 usuarios cada 5 minutos
Carga esperada	Empieza con 5 usuarios y termina con 20 usuarios

En la ilustración 35 se observa un cuadro resumen con los siguientes indicadores: tiempos de respuesta, errores, solicitudes, desempeño, transferencia total, entre otros.

	RESPONSE (AVERAGE)	ERRORS	REQUESTS	RPS (AVERAGE)	RPS (PEAK)	THROUGHPUT (AVERAGE)	THROUGHPUT (PEAK)	TOTAL TRANSFER
HTML	0.809	0	623	0.5	1.0	13 kB/s	25 kB/s	16 MB
Other *	0.436	0	4,459	3.7	7.7	48 kB/s	103 kB/s	56 MB
Total	0.482	0	5,082	4.2	8.7	61 kB/s	128 kB/s	71 MB

Ilustración 35 Tabla resumen LoadStorm – Internet

En la ilustración 36 se muestra una gráfica con la carga de usuarios que se programó para la prueba (incremental), acompañada de las curvas correspondientes al desempeño y solicitudes por segundo que está haciendo la aplicación.

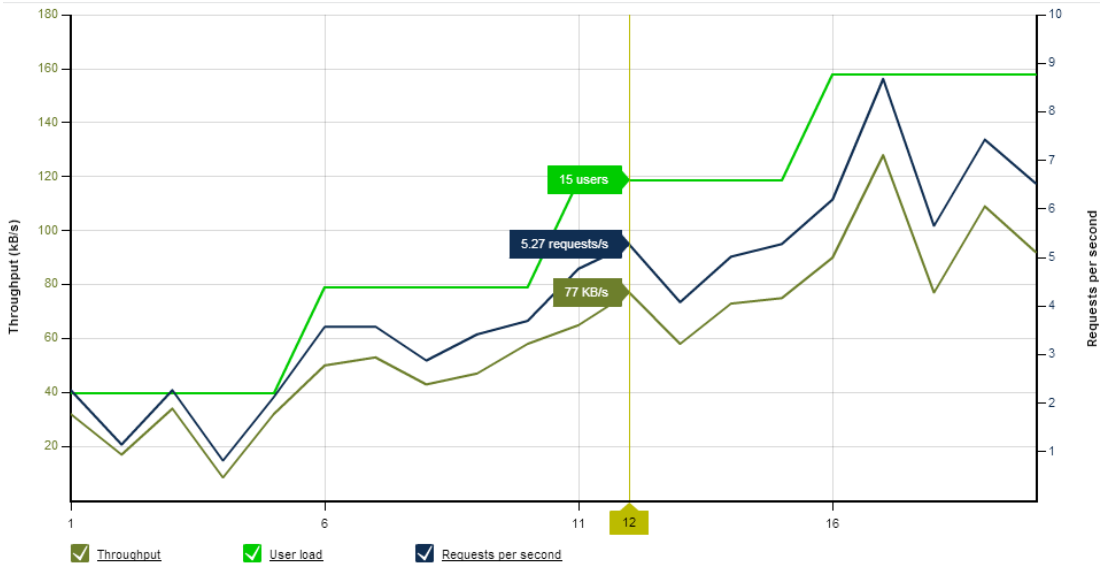


Ilustración 36 LoadStorm – rendimiento vs solicitudes por segundo

En la ilustración 37 se puede observar una gráfica con los tiempos de respuesta que ha tenido la prueba de carga en la nube durante los 20 minutos programados, adicional muestra que durante la prueba no han existido errores.

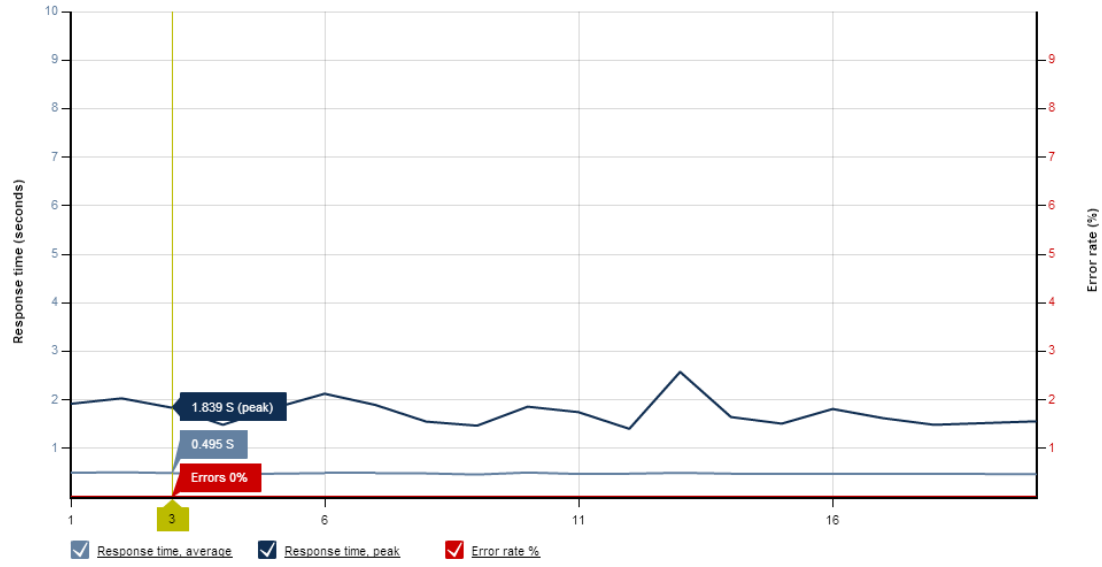


Ilustración 37 LoadStorm – tiempos de respuesta

En la sección de reportes de la herramienta LoadStorm podemos observar que nos ofrece tres tipos de cuadros con la siguiente información:

La ilustración 38 presenta un cuadro con algunas métricas importantes en intervalos de tiempo, las métricas son: cantidad de usuarios, solicitudes,

solicitudes por segundo, promedio de los tiempos de respuesta, máximo tiempo de respuesta, desempeño.

Requests by elapsed time

ELAPSED TIME	USER COUNT	REQUESTS	REQUESTS PER SECOND	AVERAGE RESPONSE TIME	MAX RESPONSE TIME	THROUGHPUT (KB/S)
0:01:00	5	136	2.27	0.496 s	1.922 s	32.0
0:02:00	5	69	1.15	0.512 s	2.03 s	17.0
0:03:00	5	136	2.27	0.495 s	1.839 s	34.0
0:04:00	5	49	0.82	0.459 s	1.488 s	8.4
0:05:00	5	128	2.13	0.481 s	1.834 s	32.0
0:06:00	10	214	3.57	0.492 s	2.126 s	50.0
0:07:00	10	215	3.58	0.491 s	1.898 s	53.0
0:08:00	10	173	2.88	0.489 s	1.55 s	43.0
0:09:00	10	205	3.42	0.463 s	1.47 s	47.0
0:10:00	10	222	3.7	0.504 s	1.861 s	58.0

Ilustración 38 LoadStorm – Solicitudes por lapsos de tiempo

La ilustración 39 presenta un cuadro con las solicitudes por código de error, como en la prueba de carga ejecutada no existieron errores esta información se muestra vacía.

Requests by error code

ERROR CODE	RESOURCE	REQUESTS	AVERAGE RESPONSE TIME
There were no errors			

Ilustración 39 LoadStorm – Solicitudes por código de error

En la ilustración 40 se muestra el último cuadro de los reportes, básicamente aquí se muestra los recursos con sus solicitudes, tamaños promedio y tiempos de respuesta promedio.

Esta información es muy útil ya que permite identificar que objetos pueden estar empobreciendo el contenido de una aplicación del lado del cliente al ser estos muy pesados, además de saturar los canales de comunicación del lado del servidor.

Además, analizando la cantidad de solicitudes que realiza un recurso ayuda a comprender si está bien programado el objeto, ya que existen datos que pueden ser almacenados en el cache del cliente para que no realice demasiadas peticiones al servidor.

Requests by response time

RESOURCE	REQUESTS	AVERAGE SIZE	AVERAGE RESPONSE TIME
/tuportal-internet/	164	20,521 bytes	1.412 s
/tuportal-internet/img/bancos/pichincha.png	162	68,304 bytes	0.813 s
/tuportal-internet/img/bancos/interdin.jpg	162	76,335 bytes	0.813 s
/mat-vehicular-internet/re...rVehiculoValoresPagar.jspa	147	43,616 bytes	0.784 s
/tuportal-internet/img/bancos/discover.png	162	51,167 bytes	0.754 s
/mat-vehicular-internet/re...6idGrupo%3D48&esFavorito=N	156	40,546 bytes	0.655 s
/tuportal-internet/img/bancos/produbanco.png	162	33,145 bytes	0.651 s
/tuportal-internet/img/bancos/bgr.png	162	17,164 bytes	0.489 s

Ilustración 40 LoadStorm – Solicitudes por tiempos de respuesta

5. CAPÍTULO V
PRUEBAS NO FUNCIONALES APLICACIÓN MÓVIL

5.1. Pruebas de Seguridad en sitio – Móvil

Debido a problemas con la licencia de desarrollo no se pudo configurar la aplicación web que se requería, así que se explicará la información obtenida de una aplicación demo de la propia herramienta IBM Rational Appscan.

Como se observa en la ilustración 41 una vez que se ha configurado el escaneo la herramienta muestra en el panel lateral izquierdo los componentes que han sido evaluados, y a la derecha presenta las vulnerabilidades detectadas por categorías.

También existe un panel donde se puede navegar más a detalle por la vulnerabilidad que se está analizando.

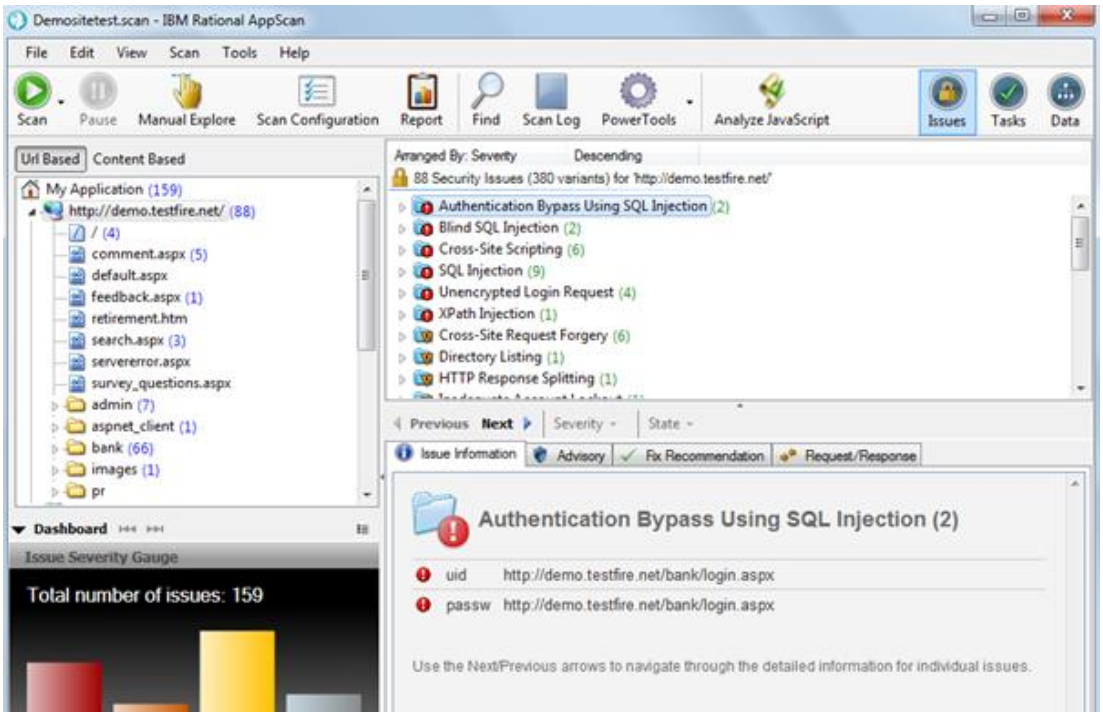


Ilustración 41 IBM Rational Appscan – Análisis inicial de incidentes

Como se muestra en las ilustraciones de la 42 a la 44 los resultados obtenidos por medio de la herramienta son muy precisos y a un nivel de detalle muy bajo porque le muestra las líneas de código que son parte de las vulnerabilidades.

Como cualquier ambiente de desarrollo integrado (IDE por sus siglas en inglés) de programación en cada línea de código indica de forma gráfica si existe alguna advertencia o error. Además que en forma gráfica muestra como están atadas las clases para indagar posibles vulnerabilidades en otras clases.

Incluso la herramienta indica las posibles soluciones que se puede ejecutar para corregir las vulnerabilidades encontradas, esto en base a las buenas prácticas de programación que almacena las políticas de la herramienta, sin embargo antes de aplicar directamente las soluciones propuestas por IBM Rational Appscan es recomendable revisar que esas modificaciones no alteren la lógica del negocio implementada.

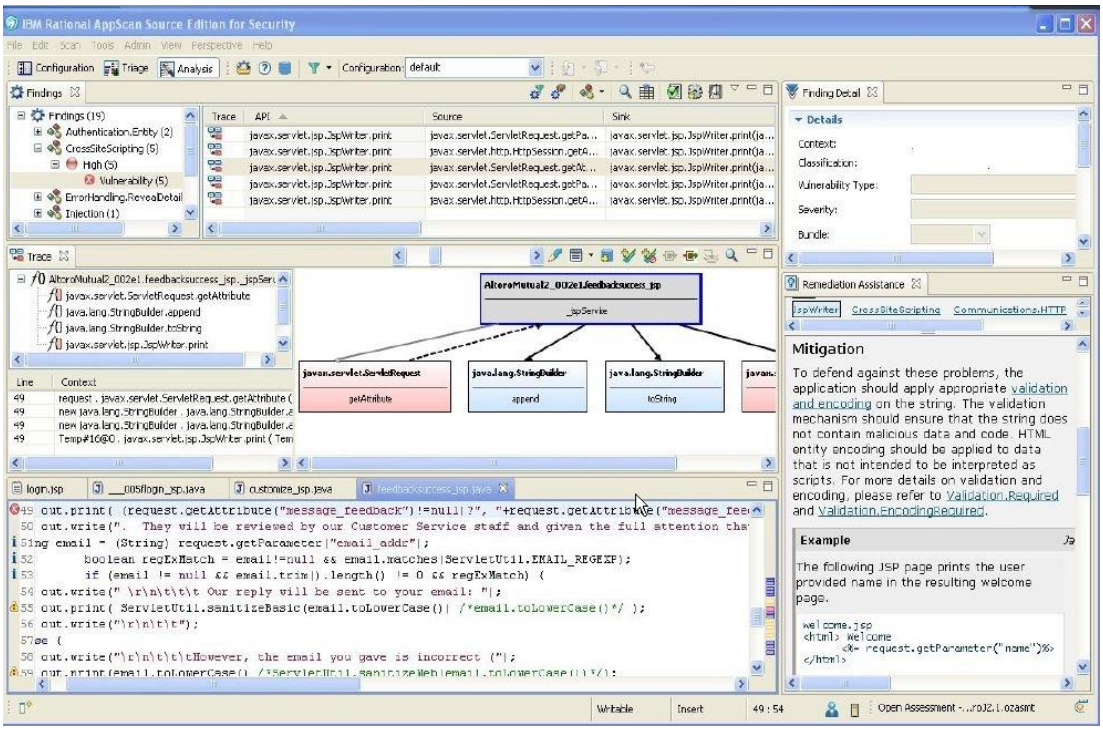


Ilustración 42 IBM Rational Appscan – Resultado 1

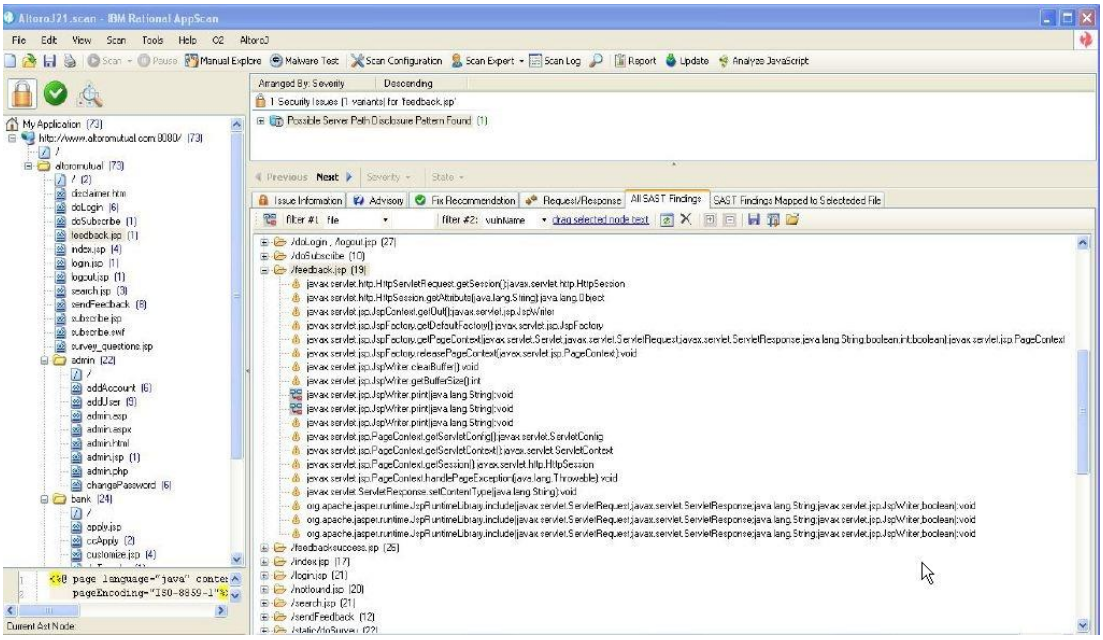


Ilustración 43 IBM Rational Appscan – Resultado 2

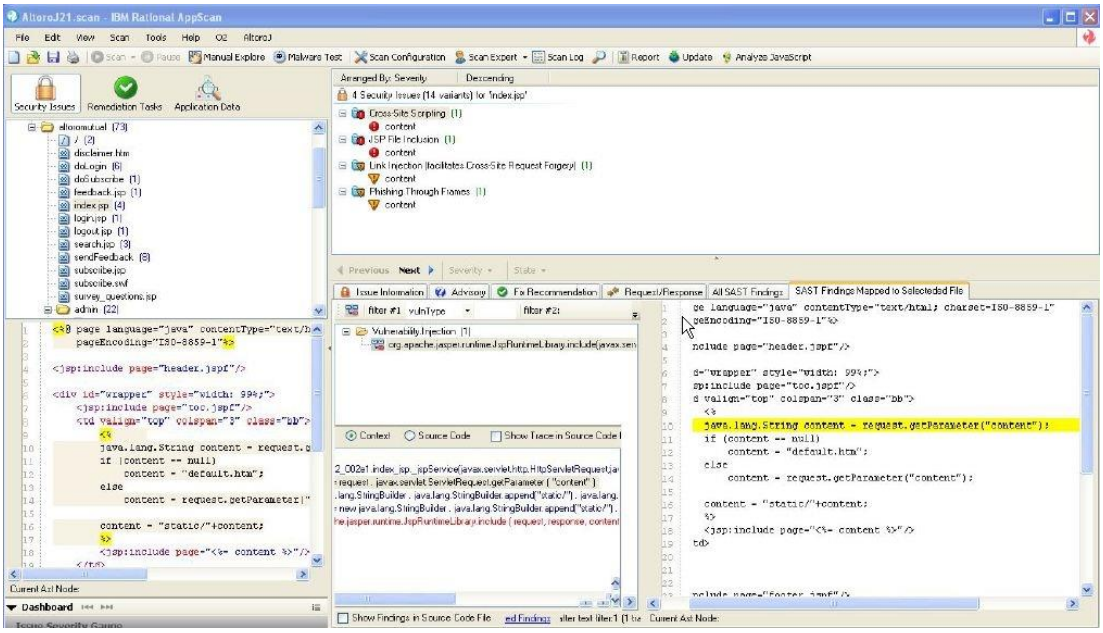


Ilustración 44 IBM Rational Appscan – Resultado 3

5.2. Pruebas de Seguridad en la nube – Móvil

La herramienta HP Fortity on Demand como se indica en la ilustración 45 presenta una gráfica de reporte muy amigable, indica en cinco paneles la información relevante a las vulnerabilidades encontradas de la siguiente manera:

- Panel de riesgos encontrados por severidad, muestra la cantidad de vulnerabilidades clasificadas por severidad: crítica, alta, media y baja.
- Panel de riesgo por impacto y probabilidad de ocurrencia
- Pastel de la mayoría de los problemas preponderantes.

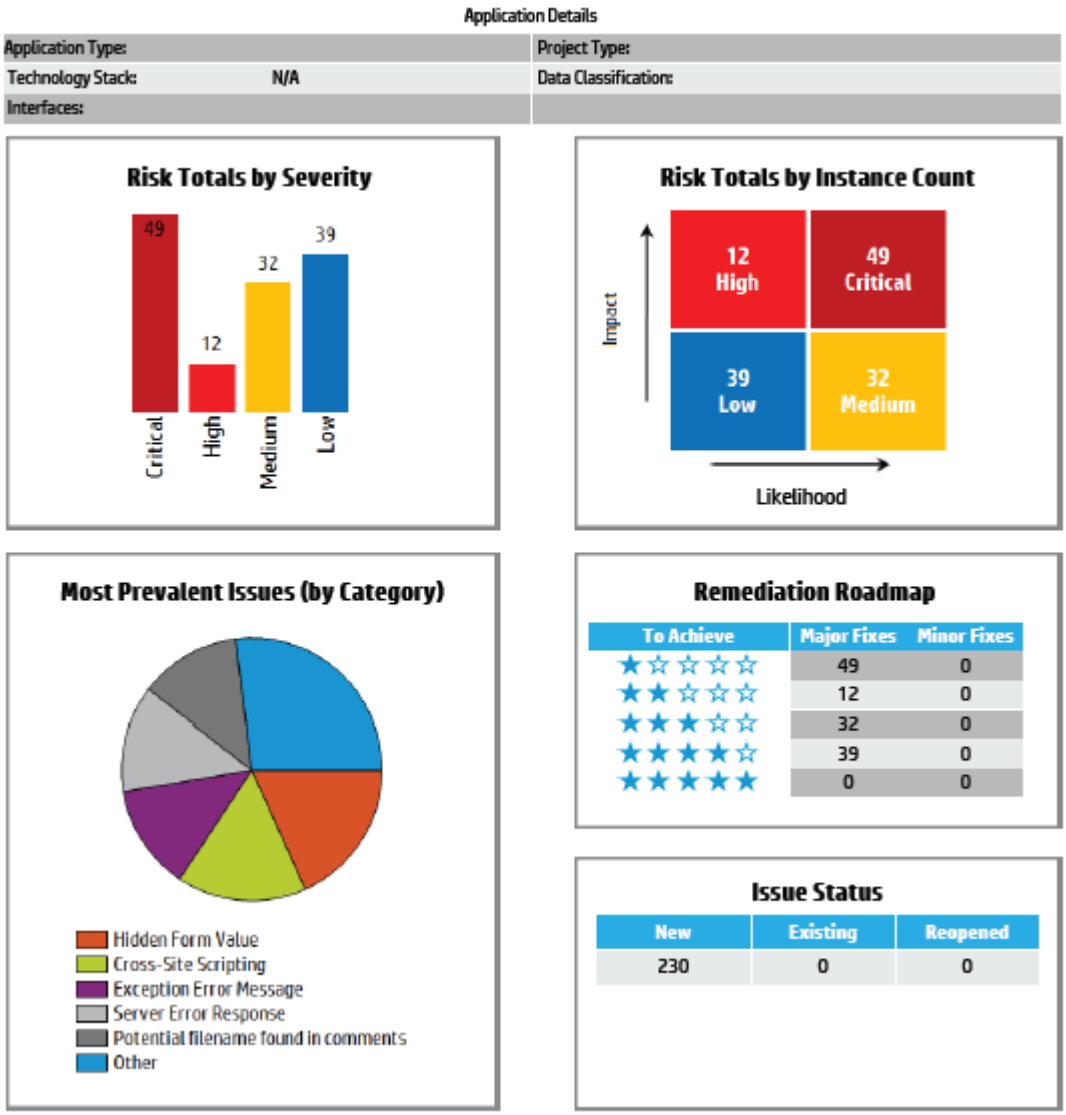


Ilustración 45 HP Fortify on Demand – Resumen vulnerabilidades

En la ilustración 46 se muestra un cuadro del resultado de las vulnerabilidades encontradas con la siguiente estructura: tipo de vulnerabilidad, categoría, tipo de prueba, ocurrencia de las vulnerabilidades.

Issue Breakdown

Issues are divided based on their impact (potential damage) and likelihood (probability of identification and exploit).

High impact / high likelihood issues represent the highest priority and present the greatest threat.

Low impact / low likelihood issues are the lowest priority and present the smallest threat.

See Appendix for more information.

Rating	Category	Test Type	Instance Count
Critical	Cross-Site Scripting	Mobile	37
Critical	Social Security Number Disclosure	Mobile	3
Critical	SQL Injection (confirmed)	Mobile	3
Critical	Universal Arbitrary Command Execution (Backticks)	Mobile	5
Critical	Universal Arbitrary Command Execution (Newline)	Mobile	1
High	Arbitrary File Upload	Mobile	1
High	Arbitrary Remote File Include	Mobile	1
High	Cross-Site Request Forgery (High)	Mobile	4
High	Local File Inclusion/Reading Vulnerability	Mobile	3
High	Logins Sent Over Unencrypted Connection	Mobile	1
High	Password in Query or Cookie Data	Mobile	1
High	Unencrypted Login Form	Mobile	1
Medium	Common Application Test Files	Mobile	1
Medium	Cross-Site Request Forgery	Mobile	1
Medium	Exception Error Message	Mobile	30
Low	ActiveX Control Discovery	Mobile	8
Low	Possible File Upload Capability	Mobile	1
Low	Server Error Response	Mobile	30

Ilustración 46 HP Fortify on Demand – Detalle de vulnerabilidades

6. CAPÍTULO VI
ANÁLISIS DE LOS RESULTADOS OBTENIDOS

6.1. Pruebas de carga en sitio versus la nube

A continuación se indica una tabla con la comparación de las herramientas de carga en la nube y en sitio:

Tabla 4 Comparación herramientas de carga

Características	SilkPerformer	LoadStorm
Ambiente	En sitio	Nube
Configuración	Compleja, requiere de conocimientos técnicos altos, la herramienta es muy robusta y permite hacer varias configuraciones para cada tipo de prueba, es multiplataforma. Se puede limitar el ancho de banda para personalizar cada ejecución.	Sencilla, una persona sin mucha experiencia en configuración de pruebas de carga puede realizarlo.
Flexibilidad	Permite parametrizar los valores ingresados para convertirlos en variables.	La versión Lite de la herramienta no soporta la configuración de valores constantes por variables.
Soporte de tecnologías	Soporta las tecnologías más importantes del mercado, Java, .Net, RichFaces, SAP, Oracle Forms, SOAP, Silverlight, Ajax, entre otras.	Dio problemas con la aplicación móvil que contiene elementos de RichFaces, estos no pudieron ser detectados y no se pudo concluir la prueba.

Continúa

Validación de scripts	Tiene un módulo que permite validar si el script está correcto antes de empezar a ejecutar la prueba.	No posee un módulo adicional, los errores del script aparecen durante la ejecución de las pruebas.
Reportes	Posee un módulo muy dinámico que permite personalizar los reportes con todas las características técnicas que grabó la herramienta como: comportamiento de memoria, uso del CPU, transacciones exitosas, transacciones fallidas, tamaño de los objetos, peticiones realizadas, etc.	Emite un informe muy sencillo con tres gráficos de resultados.
Resultados	Desempeño promedio: 980,10 Kb/s Peticiones/s: 4,43 Errores: 0	Desempeño promedio: 61 Kb/s Peticiones/s: 4,2 Errores: 0

6.2. Pruebas de seguridad en sitio versus la nube

A continuación se indica una tabla con la comparación de las herramientas de seguridad en la nube y en sitio:

Tabla 5 Comparación herramientas de seguridad

Características	IBM Rational Appscan	HP Fortify on Demand
Ambiente	En sitio	Nube
Reporte de vulnerabilidades	Por complejidad	Por complejidad

Continúa

Alertas a nivel de código	Muestra las líneas de código donde existen las vulnerabilidades y sugiere soluciones	No tiene
Categorización de vulnerabilidades	Si, registros de vulnerabilidades a nivel mundial	Si, registros de vulnerabilidades a nivel mundial
Configuración	Nivel de dificultad medio, requiere de una persona que conozca de seguridad y desarrollo de aplicaciones web.	Nivel de dificultad bajo, bastante intuitiva, preferible una persona que entienda de seguridad para mejores parametrizaciones.

7. CAPÍTULO VII

7.1. Conclusiones

- Las organizaciones que realizan pruebas no funcionales en la nube, ya sea pruebas de carga o pruebas de análisis de vulnerabilidades, no necesitan poseer infraestructura adicional a la que tienen debido a que toda la infraestructura para pruebas corre por parte de la herramienta en la nube.
- El perfil de un ingeniero de pruebas con conocimiento en pruebas técnicas o no funcionales si es requerido en cualquiera de los dos casos, ya sean pruebas en sitio o en la nube, ya que para la configuración y ejecución se debe tener muy claro el alcance que se desea cubrir con las pruebas o que vulnerabilidades se deben evitar.
- Para el análisis de los resultados también es necesario que se cuente con el perfil del ingeniero de pruebas ya que ahí se puede entender cuáles son las brechas de calidad que posee la aplicación, además el ingeniero de pruebas debe plantear las posibles soluciones para los incidentes encontrados.
- Para determinar si se compra una herramienta en sitio o se alquila una herramienta en la nube se debe tener muy en cuenta el alcance de las pruebas y la dimensión de la organización. Si se van a tener pruebas esporádicas no hace falta adquirir una herramienta para tenerla subutilizada, por otro lado no todas las organizaciones pueden darse el lujo de adquirir una licencia que sobrepasa los cien mil dólares.
- Para que el cliente de una herramienta en la nube pueda estar tranquilo con la seguridad de su información al momento de hacer pruebas puede utilizar información enmascarada y por medio de un contrato de servicios pasar la responsabilidad de la información a la organización contratada.
- En términos de disponibilidad de las herramientas ninguna puede garantizar un 100% de disponibilidad cada ambiente tiene sus riesgos, por ejemplo las herramientas en la nube disponen de servicios e infraestructura distribuida a nivel mundial que mitiga el riesgo de la indisponibilidad, sin embargo uno de los días de pruebas la herramienta LoadStorm presentó

problemas. Por otro lado el tener una herramienta en sitio puede sufrir de inconvenientes en la red local de la empresa o problemas de actualizaciones que no permitan tenerla siempre disponible.

- Los resultados obtenidos por cada una de las herramientas es independiente del ambiente en el que se la utiliza, la estructura de los resultados está ligada a su proveedor, la claridad, simplicidad o detalle de los resultados son parte de la oferta que tiene cada casa que desarrolla estas suites de pruebas.
- Las organizaciones que desarrollaron sus herramientas en sitio han entendido la importancia que tiene el uso de la nube hoy en día, por lo que se evidenció con el uso de la herramienta SilkPerformer que están implementando componentes para que se ejecuten desde la nube y así liberar los recursos de la organización cliente.

7.2. Recomendaciones

- Cada organización debe tener la idea clara de lo que tiene que cubrir con las pruebas no funcionales en sus productos de software y lo que puede invertir para las herramientas, existen herramientas para todos los bolsillos el problema es que las menos costosas no son tan flexibles o no soportan la cantidad de usuarios necesaria, por lo tanto es necesario realizar un análisis costo beneficio para saber que herramienta conviene a cada organización.
- Antes de adquirir o alquilar una herramienta en cualquier ambiente es necesario solicitar al proveedor un demo de la herramienta para saber si esta va a servir en las aplicaciones de la organización. Este demo debe ser ejecutado de preferencia con las aplicaciones que van a ser configuradas en la organización con eso se conocerá si la herramienta soporta la tecnología de cada aplicación.
- A pesar de que existen buenas aplicaciones en la nube para realizar pruebas de carga, con las pruebas que se realizaron en este proyecto se puede observar que la herramienta en sitio es mucho más robusta y flexible que la herramienta en la nube, por lo tanto se recomienda el uso de esta herramienta para organizaciones grandes.

- Para una organización que esté iniciándose con las pruebas no funcionales se recomienda que empiece con pruebas en la nube, esto le puede ayudar a aprender poco a poco como deben ser las configuraciones en las herramientas hasta ir ganado la experiencia suficiente para adquirir o alquilar una herramienta más compleja y costosa.

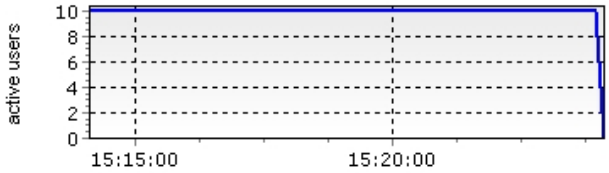
8. BIBLIOGRAFÍA

- Ayuda de la herramienta SilkPerformer de Microfocus. Recuperado de: http://www.borland.com/Products/Software-Testing/Performance-Testing/Silk-Performer
- Ayuda de la herramienta LoadStorm. Recuperado de: http://loadstorm.com/how-to-load-test/
- Ayuda de la herramienta IBM Rational Appscan. Recuperado de: http://www-03.ibm.com/software/products/es/appscan
- Ayuda de la herramienta HP Fortify on Demand. Recuperado de: https://saas.hp.com/es-es/home
- International Software Testing Qualifications Board. (2011). *Certified Tester Foundation Level (V1.10)*. Bélgica.
- Podelko, A. (s.f.). Performance Testing in the Cloud: Look Beyond the Word. *Testing Experience, Diciembre 2012*, 7-8.
- Forno, M. (s.f.). Software Testing: Adapt and Grow with the Cloud. *Testing Experience, Diciembre 2012*, 19-20.
- Chittanai, K. (s.f.). Moving Testing to the Cloud – An Exploration. *Testing Experience, Diciembre 2012*, 27-29.
- Meier, J. D., Farre, C., Bansode, P., Barber, S., Rea, D. (2007). Performance Testing Guidance for Web Applications.
- Ford, C., Gileadi, I., Purba, S., Moerman, M., (2007). Patterns for Performance and Operability: Building and Testing Enterprise. Auerbach Publications, 11-13.
- LOADUIWEB. SmartBear. Análisis de Reportes, métricas más importantes, Recuperado de: http://loaduiweb.org/docs/testing/results/analyzing-the-report.html

9. ANEXOS

9.1. Reporte pruebas de carga en sitio aplicación ambiente Intranet

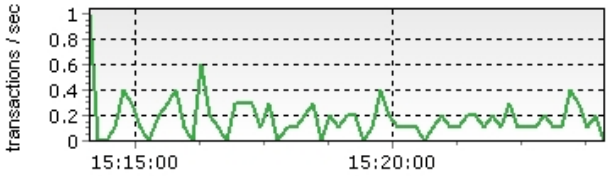
General Project Settings	
Application type:	Web business transaction (HTML/HTTP)
Workload:	Workload1
Workload model:	Steady State



Active Users

This chart shows the overall number of active virtual users. A virtual user is considered as active if the user has started and is currently in one of the following states: executing, wait DB, document downloading, and thinktime.

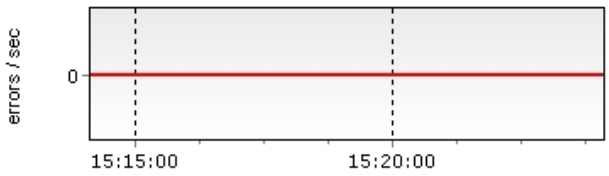
number of concurrent users:	10



Transactions

The number of Silk Performer transactions per second.

number of transactions:	108
average number of transactions/sec:	0.18

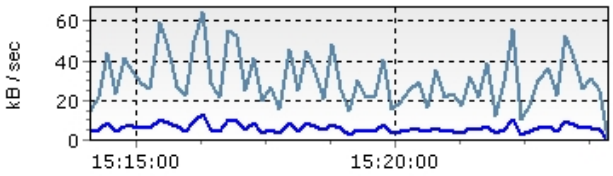


Errors

This chart shows the number of API errors per second, including Internet, database, and middleware APIs. A problem is considered an error if its severity is defined as "error" or worse, that is, of higher severity ("transaction exit" or "process exit"). A problem is ignored if its severity is defined as "informational" or "warning".

number of errors:	0

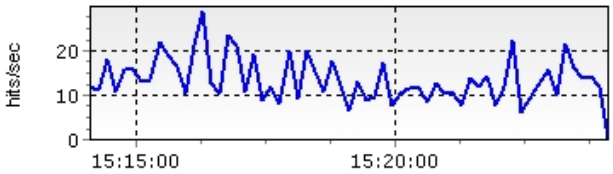
Throughput / Concurrency



Throughput[kB]

The amount of data sent to and received from the server; this includes header and body content information, all TCP/IP-related traffic (HTTP, native TCP/IP, IIOP, POP3, SMTP, FTP, LDAP and WAP), and secure traffic over SSL/TLS. This measurement does not include data overhead caused by SSL/TLS encryption and WTLS encryption in case of WAP.
Request data sent
Response data received

throughput[kB]:	22,181
average throughput[kB]/sec:	36.36

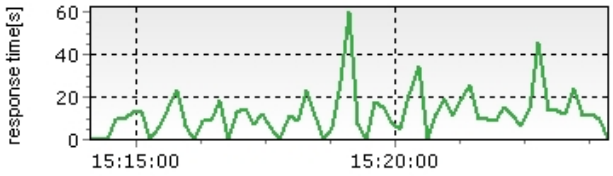


Http Hits

The number of HTTP requests that arrive at the Web server.

number of hits:	8,265
average number of hits/sec:	13.55

Response Times - Transactions

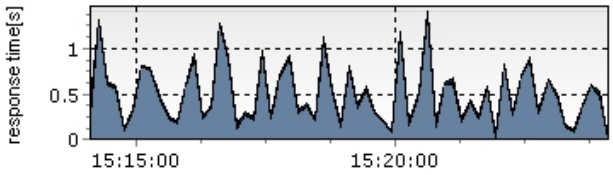


Trans. (busy) ok[s]

The response time of successful transactions, excluding the think times within those transactions. A transaction response time is reported in this type of measurement if all API function calls within the transaction succeed.

minimum[s]:	2.75
average[s]:	14.02
maximum[s]:	67.33
standard deviation[s]:	11.57

Response Times - Page and Action Timers



Page time[s]

The time it takes a virtual user to download a Web page from the server, in seconds. Response times for Web pages are subdivided into server-busy times, document-downloading times, and round-trip times.

average page time[s]:	0.50
average document downloading time[s]:	0.47
average server busy time[s]:	0.46

general information | summary tables | ranking | user types | custom charts | custom tables | detailed charts

The Summary tables contain summarized measurements on a global level. They contain measurement types that aggregate individual measurements from other measurement groups as well as measurement types that represent information on a global level that is not included in other measurement groups.

Summary General

Name	Count	1/sec	1/min	1/h
Transactions	108	0.18	10.62	637
Errors	0	0.00	0.00	0

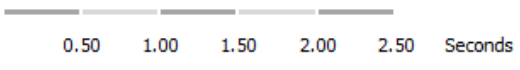
Summary Internet

Name	Count	1/sec	1/min	1/h
Request data sent[kB]	3,680	6.03	361.97	21,718
Response data received[kB]	18,501	30.33	1,819.77	109,186
Requests sent	8,265	13.55	812.95	48,777
Requests failed	0	0.00	0.00	0
Responses received	8,265	13.55	812.95	48,777
Responses failed	0	0.00	0.00	0
Connects successful	8,265	13.55	812.95	48,777
Connects failed	0	0.00	0.00	0
Connects retries	0	0.00	0.00	0

general information | summary tables | ranking | user types | custom charts | custom tables | detailed charts

Slowest Web pages (Page Time)

Name	Seconds	Count	Total	Min	**Avg**	Max	StdDev
Sistema de Matriculación Vehicular		104	235.99	0.93	**2.27**	6.97	1.21
Servicio de Rentas Internas (#1)		104	77.66	0.37	**0.75**	1.99	0.29
Sistema de Matriculación Vehicular (#1)		100	36.89	0.22	**0.37**	1.80	0.18
Sistema de Matriculación Vehicular (#3)		98	34.97	0.22	**0.36**	1.52	0.19
Sistema de Matriculación Vehicular (#2)		99	6.77	0.04	**0.07**	0.35	0.04
Servicio de Rentas Internas		108	5.21	0.02	**0.05**	0.44	0.07
Servicio de Rentas Internas (#3)		98	3.51	0.02	**0.04**	0.11	0.01
Servicio de Rentas Internas (#2)		98	1.20	0.01	**0.01**	0.05	0.01



general information | summary tables | ranking | user types | custom charts | custom tables | detailed charts

The user type section provides detailed tabular results about transactions, individual timers and counters, and interface dependent timers and counters for WEB, database, IIOP and Tuxedo on a per user type level. In addition all API errors, and warnings for each user type are listed.

User Type	Average Page Time [s]	Average Action Time [s]	#Transaction OK	#Transaction Cancelled	#T
MAV_ConPubIntra_v1_1.bdf/VUser-Profile1	0.497	0.000	98		

Time bound histograms

Bound1 75.15% < 1

Bound2 93.88% < 2

In this histogram, response time measurements are grouped into three categories:

Green: The percentage of response times shorter than the value for time bound 1.

Yellow: The percentage of response times longer than the value for time bound 1, but shorter than the value for time bound 2.

Red: The percentage of response times longer than the value for time bound 2.

In the example above,
75% of the response times are shorter than time bound 1, these response times may be considered fully satisfactory;
19% of the response times are between time bound 1 and time bound 2, these response times may be considered slightly problematic, but still acceptable;
6% of the response times are longer than time bound 2, these response times have to be considered seriously problematic.

You can set the time bounds for transactions, custom timers, and page timers automatically using the baseline results of your test. From the Silk Performer workflow bar, select Confirm Baseline/Set response time thresholds. MeasureSetBound functions will be generated into your script to set thresholds for the selected timers.Additionally, you can set time bounds manually in your test script by calling the MeasureSetBound function.

9.2. Reporte pruebas de seguridad en sitio aplicación ambiente Intranet

Executive Summary

Company:	SRI_2_FMA_984087840
Project:	Sample Static Application
Version:	Sample Scan
Static Analysis Date:	12/10/2014 9:01:56 AM
Dynamic Analysis Date:	

Fortify Security Rating

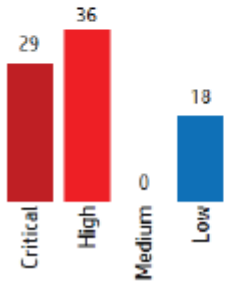
★☆☆☆☆

83 issues

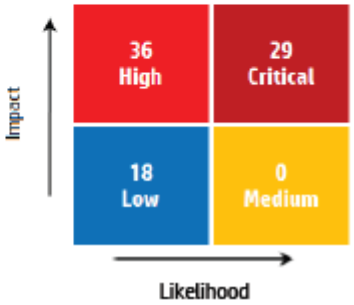
Static: ✔ Dynamic: ✖

Application Details

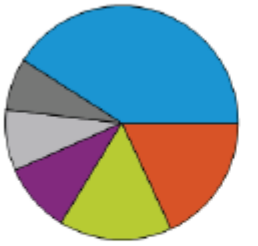
Risk Totals by Severity



Risk Totals by Instance Count



Most Prevalent Issues (by Category)



- Cross-Site Request Forgery
- Unreleased Resource: Database
- Cross-Site Scripting: Reflected
- Privacy Violation
- Unreleased Resource: Streams
- Other

Remediation Roadmap

To Achieve	Major Fixes	Minor Fixes
★☆☆☆☆	0	29
★★☆☆☆	1	35
★★★☆☆	0	0
★★★★☆	15	3
★★★★★	0	0

Issue Status

New	Existing	Reopened
83	0	0

Issue Breakdown

Issues are divided based on their impact (potential damage) and likelihood (probability of identification and exploit).

High impact / high likelihood issues represent the highest priority and present the greatest threat.

Low impact / low likelihood issues are the lowest priority and present the smallest threat.

See Appendix for more information.

Rating	Category	Test Type	Instance Count
Critical	Cross-Site Scripting: DOM	Static	4
Critical	Cross-Site Scripting: Reflected	Static	8
Critical	Dangerous File Inclusion	Static	2
Critical	Path Manipulation	Static	2
Critical	Privacy Violation	Static	5
Critical	SQL Injection: Hibernate	Static	4
Critical	SQL Injection	Static	4
High	Command Injection	Static	6
High	Header Manipulation	Static	1
High	Log Forging	Static	2
High	Password Management: Password in Configuration File	Static	1
High	Path Manipulation	Static	1
High	Portability Flaw: File Separator	Static	1
High	Privacy Violation	Static	2
High	Unreleased Resource: Database	Static	13
High	Unreleased Resource: Sockets	Static	3
High	Unreleased Resource: Streams	Static	6
Low	Cookie Security: Cookie not Sent Over SSL	Static	3
Low	Cross-Site Request Forgery	Static	15

Issue Breakdown by OWASP Top 10 2013 PCI Sections 6.3, 6.5 & 6.6

The OWASP Top Ten represents a broad consensus about what the most critical web application security flaws are. Project members include a variety of security experts from around the world who have shared their expertise to produce this list.

The PCI compliance standards, particularly sections 6.3, 6.5, and 6.6, reference the OWASP Top Ten vulnerability categories as the core categories that must be tested for and remediated.

OWASP 2013 Category	Severity			
	Critical	High	Medium	Low
None	29	36		18
Total	29	36		18

Issue Breakdown by Analysis Type

Issues are divided based on their impact (potential damage) and likelihood (probability of identification and exploit).

High impact / high likelihood issues represent the highest priority and present the greatest threat.

Low impact / low likelihood issues are the lowest priority and present the smallest threat.

See Appendix for more information.

Category	Static	Dynamic
Command Injection	6	0
Cookie Security: Cookie not Sent Over SSL	3	0
Cross-Site Request Forgery	15	0
Cross-Site Scripting: DOM	4	0
Cross-Site Scripting: Reflected	8	0
Dangerous File Inclusion	2	0
Header Manipulation	1	0
Log Forging	2	0
Password Management: Password in Configuration File	1	0
Path Manipulation	3	0
Portability Flaw: File Separator	1	0
Privacy Violation	7	0
SQL Injection	4	0
SQL Injection: Hibernate	4	0
Unreleased Resource: Database	13	0
Unreleased Resource: Sockets	3	0
Unreleased Resource: Streams	6	0
Total	83	0

Appendix - Descriptions of Key Terminology

Security Rating

The Fortify 5-star assessment rating provides information on the likelihood and impact of defects present within an application. A perfect rating within this system would be 5 complete stars indicating that no high impact vulnerabilities were uncovered.

Rating	
★☆☆☆☆	Fortify awards one star to projects that undergo a Fortify security review, which analyzes a project for a variety of software security vulnerabilities.
★★☆☆☆	Fortify awards two stars to projects that undergo a Fortify security review that identifies no high likelihood / high impact issues. Vulnerabilities that are trivial to exploit and have a high business or technical impact should never exist in business-critical software.
★★★☆☆	Fortify awards three stars to projects that undergo a Fortify security review that identifies no low likelihood / high impact issues and meets the requirements needed to receive two stars. Vulnerabilities that have a high impact, even if they are non-trivial to exploit, should never exist in business critical software.
★★★★☆	Fortify awards four stars to projects that undergo a Fortify security review that identifies no high likelihood / low impact issues and meets the requirements for three stars. Vulnerabilities that have a low impact, but are easy to exploit, should be considered carefully as they may pose a greater threat if an attacker exploits many of them as part of a concerted effort or leverages a low impact vulnerability as a stepping stone to mount a high-impact attack.
★★★★★	Fortify awards five stars to projects that undergo a Fortify security review that identifies no issues.

Likelihood and Impact

Likelihood

Likelihood is the probability that a vulnerability will be accurately identified and successfully exploited.

Impact

Impact is the potential damage an attacker could do to assets by successfully exploiting a vulnerability. This damage can be in the form of, but not limited to, financial loss, compliance violation, loss of brand reputation, and negative publicity.

Fortify Priority Order

Critical

Critical-priority issues have high impact and high likelihood. Critical-priority issues are easy to detect and exploit and result in large asset damage. These issues represent the highest security risk to the application. As such, they should be remediated immediately.

SQL Injection is an example of a critical issue.

High

High-priority issues have high impact and low likelihood. High-priority issues are often difficult to detect and exploit, but can result in large asset damage.
These issues represent a high security risk to the application. High priority issues should be remediated in the next scheduled patch release.

Medium

Medium-priority issues have low impact and high likelihood. Medium-priority issues are easy to detect and exploit, but typically result in small asset damage.
These issues represent a moderate security risk to the application. Medium-priority issues should be remediated in the next scheduled

Low

Low-priority issues have low impact and low likelihood. Low-priority issues can be difficult to detect and exploit and typically result in small asset damage.
These issues represent a minor security risk to the application. Low priority issues should be remediated as time allows.

Issue Status

New

New issues are ones that have been identified for the first time in the most recent analysis of the application.

Existing

Existing issues are issues that have been found in a previous analysis of the application and are still present in the latest analysis.

Reopened

Reopened issues have been discovered in a previous analysis of the application but were not present in subsequent analyses. These issues are now present again in the most recent analysis of the application.

Fortify Remediation Effort

Major Remediation

Major remediation effort issues must often be addressed at multiple locations to fix the root problem.

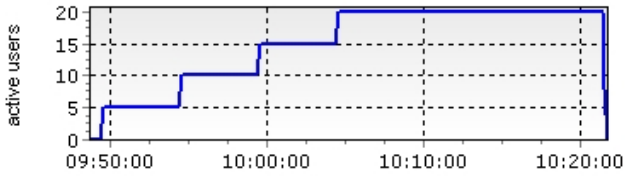
Minor Remediation

Minor remediation effort issues can typically be addressed at the location of the root problem.

9.3. Reporte pruebas de carga en sitio aplicación ambiente Internet

Project:	MAV_consultas_pub	Matriculación Vehicular consultas publicas
Load test:	0	None
Start date/time:	12/9/2014 9:48:33 AM	
Duration of simulation:	00:33:10	
Agents:	1	
Users:	20	
Report Description:		

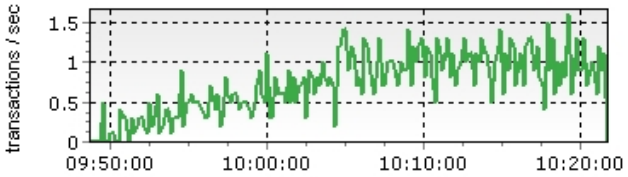
General Project Settings	
Application type:	Web business transaction (HTML/HTTP)
Workload:	Workload1
Workload model:	Increasing



Active Users

This chart shows the overall number of active virtual users. A virtual user is considered as active if the user has started and is currently in one of the following states: executing, wait DB, document downloading, and thinktime.

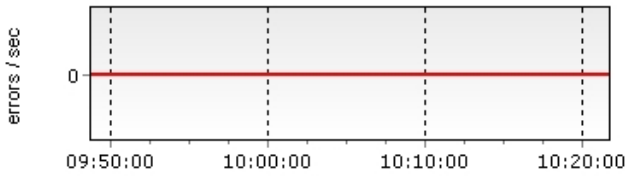
number of concurrent users: 20



Transactions

The number of Silk Performer transactions per second.

number of transactions:	1,440
average number of transactions/sec:	0.72

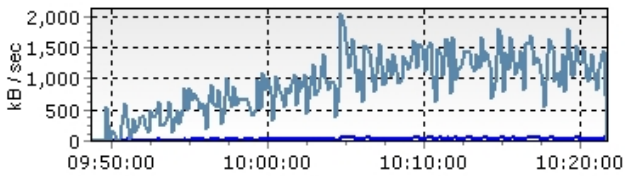


Errors

This chart shows the number of API errors per second, including Internet, database, and middleware APIs. A problem is considered an error if its severity is defined as "error" or worse, that is, of higher severity ("transaction exit" or "process exit"). A problem is ignored if its severity is defined as "informational" or "warning".

number of errors: 0

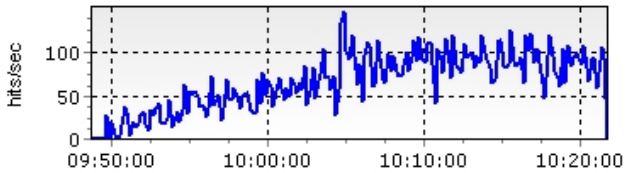
Throughput / Concurrency



Throughput[kB]

The amount of data sent to and received from the server; this includes header and body content information, all TCP/IP-related traffic (HTTP, native TCP/IP, IIOP, POP3, SMTP, FTP, LDAP and WAP), and secure traffic over SSL/TLS. This measurement does not include data overhead caused by SSL/TLS encryption and WTLS encryption in case of WAP.
Request data sent
Response data received

throughput[kB]:	1,891,725
average throughput[kB]/sec:	950.62

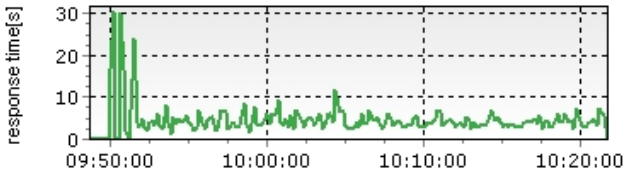


Http Hits

The number of HTTP requests that arrive at the Web server.

number of hits:	131,952
average number of hits/sec:	66.31

Response Times - Transactions

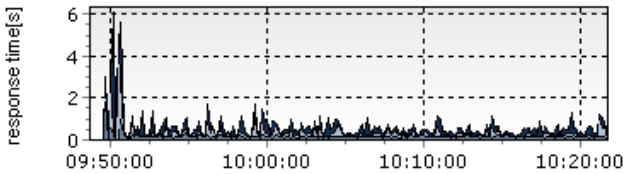


Trans. (busy) ok[s]

The response time of successful transactions, excluding the think times within those transactions. A transaction response time is reported in this type of measurement if all API function calls within the transaction succeed.

minimum[s]:	1.00
average[s]:	4.43
maximum[s]:	57.25
standard deviation[s]:	4.11

Response Times - Page and Action Timers



Page time[s]

The time it takes a virtual user to download a Web page from the server, in seconds. Response times for Web pages are subdivided into server-busy times, document-downloading times, and round-trip times.

average page time[s]:	0.52
average document downloading time[s]:	0.29
average server busy time[s]:	0.16

general information | summary tables | ranking | user types | custom charts | custom tables | detailed charts

The Summary tables contain summarized measurements on a global level. They contain measurement types that aggregate individual measurements from other measurement groups as well as measurement types that represent information on a global level that is not included in other measurement groups.

Summary General

Name	Count	1/sec	1/min	1/h
Transactions	1,440	0.72	43.42	2,605
Errors	0	0.00	0.00	0

Summary Internet

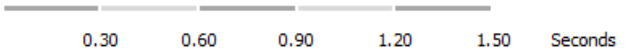
Name	Count	1/sec	1/min	1/h
Request data sent[kB]	61,099	30.70	1,842.18	110,531
Response data received[kB]	1,830,626	919.91	55,194.75	3,311,685
Requests sent	138,505	69.60	4,176.03	250,562
Requests failed	18	0.01	0.54	33
Responses received	131,952	66.31	3,978.45	238,707
Responses failed	0	0.00	0.00	0
Connects successful	9,433	4.74	284.41	17,065
Connects failed	18	0.01	0.54	33
Connects retries	18	0.01	0.54	33

Summary Web

Name	Count	1/sec	1/min	1/h
Http redirections	1,430	0.72	43.12	2,587
Http re-authentications	0	0.00	0.00	0
Http hits	131,952	66.31	3,978.45	238,707
Http pages	7,140	3.59	215.28	12,917
Http cookies received	17,130	8.61	516.48	30,989
Http cookies sent	158,562	79.68	4,780.76	286,846
Hits failed	0	0.00	0.00	0
Http cache hits	8,520	4.28	256.88	15,413
Http cache cond. reloads	0	0.00	0.00	0
Http 1xx responses	0	0.00	0.00	0
Http 2xx responses	121,972	61.29	3,677.55	220,653
Http 3xx responses	1,430	0.72	43.12	2,587
Http 4xx responses	8,550	4.30	257.79	15,467
Http 5xx responses	0	0.00	0.00	0
Http request retries	5,114	2.57	154.19	9,251

Slowest Web pages (Page Time)

Name	Seconds	Count	Total	Min	Avg	Max	StdDev
Servicios en Línea		1,440	1,961.84	0.58	**1.36**	24.59	1.86
Sistema de Matriculación Vehicular (#1)		1,420	957.11	0.05	**0.67**	28.95	1.80
Sistema de Matriculación Vehicular		1,430	677.36	0.05	**0.47**	25.18	1.70
comun.jsp		1,430	68.71	0.03	**0.05**	0.50	0.02
comun.jsp (#1)		1,420	60.64	0.02	**0.04**	3.14	0.20

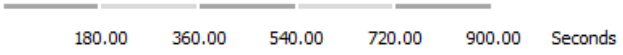


Largest Web pages (Page data)

Name	KBytes	Count	Total	Min	Avg	Max	StdDev
Servicios en Línea		1,440	1,525,848.75	1,059.62	**1,059.62**	1,059.62	0.00
Sistema de Matriculación Vehicular		1,430	189,081.05	131.99	**132.22**	133.32	0.33
Sistema de Matriculación Vehicular (#1)		1,420	82,664.35	57.20	**58.21**	62.43	1.58
comun.jsp		1,430	51,702.55	36.15	**36.16**	36.63	0.03
comun.jsp (#1)		1,420	39,833.64	28.03	**28.05**	29.05	0.11

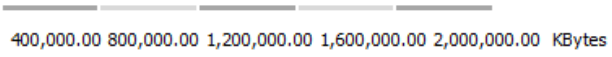
Most server resource consuming Web pages (Total Server Busy time)

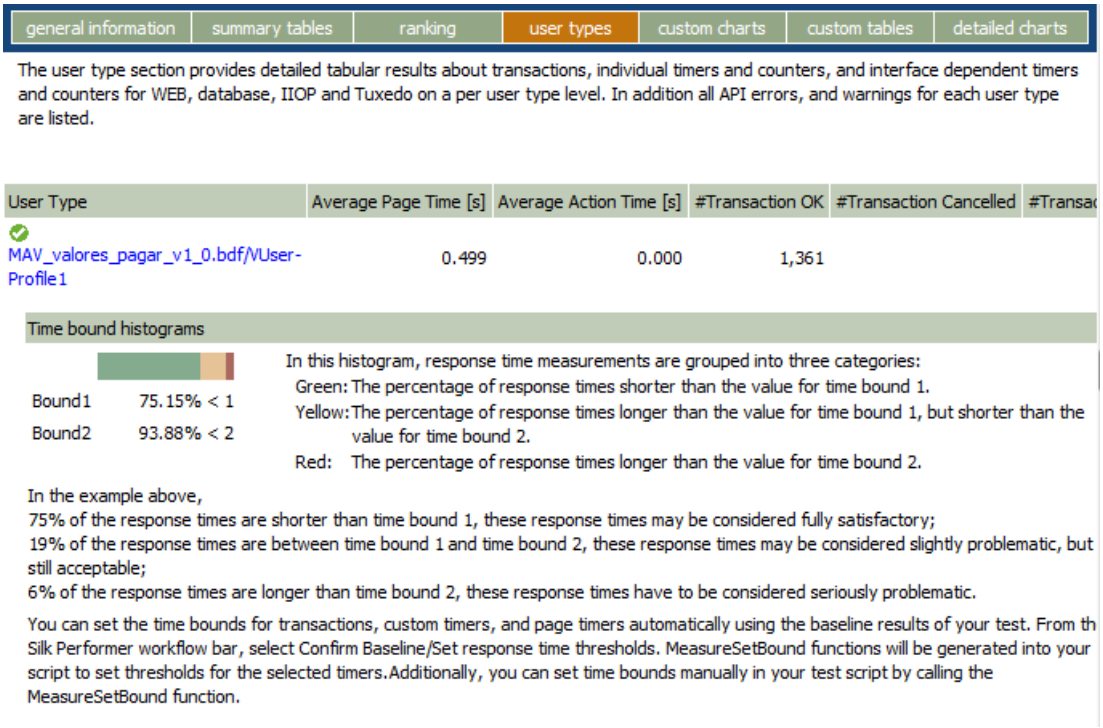
Name	Seconds	Count	Total	Min	Avg	Max	StdDev
Servicios en Línea		1,440	**891.83**	0.39	0.62	3.27	0.18
Sistema de Matriculación Vehicular (#1)		1,420	**196.48**	0.00	0.14	8.02	0.26
comun.jsp (#1)		1,420	**18.05**	0.00	0.01	0.06	0.01
Sistema de Matriculación Vehicular		1,430	**6.75**	0.00	0.00	1.09	0.03
comun.jsp		1,430	**0.25**	0.00	0.00	0.25	0.01



Most network resource consuming Web pages (Total Page data)

Name	KBytes	Count	Total	Min	Avg	Max	StdDev
Servicios en Línea		1,440	**1,525,848.75**	1,059.62	1,059.62	1,059.62	0.00
Sistema de Matriculación Vehicular		1,430	**189,081.05**	131.99	132.22	133.32	0.33
Sistema de Matriculación Vehicular (#1)		1,420	**82,664.35**	57.20	58.21	62.43	1.58
comun.jsp		1,430	**51,702.55**	36.15	36.16	36.63	0.03
comun.jsp (#1)		1,420	**39,833.64**	28.03	28.05	29.05	0.11




general information
summary tables
ranking
user types
custom charts
custom tables
detailed charts
The user type section provides detailed tabular results about transactions, individual timers and counters, and interface dependent timers and counters for WEB, database, IIOP and Tuxedo on a per user type level. In addition all API errors, and warnings for each user type are listed.
User Type
Average Page Time [s]
Average Action Time [s]
#Transaction OK
#Transaction Cancelled
#Transac
MAV_valores_pagar_v1_0.bdf/VUser-Profile1
0.499
0.000
1,361
Time bound histograms
Bound1 75.15% < 1
Bound2 93.88% < 2
In this histogram, response time measurements are grouped into three categories:
Green: The percentage of response times shorter than the value for time bound 1.
Yellow: The percentage of response times longer than the value for time bound 1, but shorter than the value for time bound 2.
Red: The percentage of response times longer than the value for time bound 2.
In the example above,
75% of the response times are shorter than time bound 1, these response times may be considered fully satisfactory;
19% of the response times are between time bound 1 and time bound 2, these response times may be considered slightly problematic, but still acceptable;
6% of the response times are longer than time bound 2, these response times have to be considered seriously problematic.
You can set the time bounds for transactions, custom timers, and page timers automatically using the baseline results of your test. From th Silk Performer workflow bar, select Confirm Baseline/Set response time thresholds. MeasureSetBound functions will be generated into your script to set thresholds for the selected timers.Additionally, you can set time bounds manually in your test script by calling the MeasureSetBound function.

9.4. Reporte pruebas de seguridad en la nube aplicación móvil

Executive Summary

Company:	SRI_2_FMA_984087840
Project:	Sample Mobile Application
Version:	Sample Scan
Mobile Analysis Date:	12/10/2014 9:02:11 AM

Fortify Security Rating

★☆☆☆☆

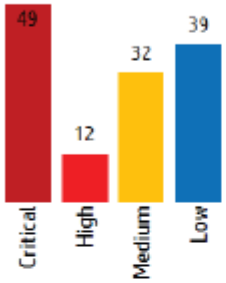
230 issues

Mobile: ✔

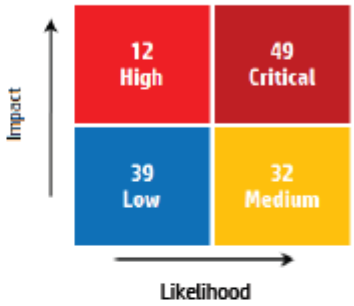
Application Details

Application Type:		Project Type:	
Technology Stack:	N/A	Data Classification:	
Interfaces:			

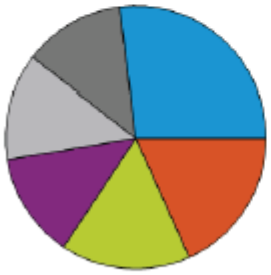
Risk Totals by Severity



Risk Totals by Instance Count



Most Prevalent Issues (by Category)



- Hidden Form Value
- Cross-Site Scripting
- Exception Error Message
- Server Error Response
- Potential filename found in comments
- Other

Remediation Roadmap

To Achieve	Major Fixes	Minor Fixes
★☆☆☆☆	49	0
★★☆☆☆	12	0
★★★☆☆	32	0
★★★★☆	39	0
★★★★★	0	0

Issue Status

New	Existing	Reopened
230	0	0

Issue Breakdown

Issues are divided based on their impact (potential damage) and likelihood (probability of identification and exploit).

High impact / high likelihood issues represent the highest priority and present the greatest threat.

Low impact / low likelihood issues are the lowest priority and present the smallest threat.

See Appendix for more information.

Rating	Category	Test Type	Instance Count
Critical	Cross-Site Scripting	Mobile	37
Critical	Social Security Number Disclosure	Mobile	3
Critical	SQL Injection (confirmed)	Mobile	3
Critical	Universal Arbitrary Command Execution (Backticks)	Mobile	5
Critical	Universal Arbitrary Command Execution (Newline)	Mobile	1
High	Arbitrary File Upload	Mobile	1
High	Arbitrary Remote File Include	Mobile	1
High	Cross-Site Request Forgery (High)	Mobile	4
High	Local File Inclusion/Reading Vulnerability	Mobile	3
High	Logins Sent Over Unencrypted Connection	Mobile	1
High	Password in Query or Cookie Data	Mobile	1
High	Unencrypted Login Form	Mobile	1
Medium	Common Application Test Files	Mobile	1
Medium	Cross-Site Request Forgery	Mobile	1
Medium	Exception Error Message	Mobile	30
Low	ActiveX Control Discovery	Mobile	8
Low	Possible File Upload Capability	Mobile	1
Low	Server Error Response	Mobile	30

Appendix - Descriptions of Key Terminology

Security Rating

The Fortify 5-star assessment rating provides information on the likelihood and impact of defects present within an application. A perfect rating within this system would be 5 complete stars indicating that no high impact vulnerabilities were uncovered.

Rating	
★☆☆☆☆	Fortify awards one star to projects that undergo a Fortify security review, which analyzes a project for a variety of software security vulnerabilities.
★★☆☆☆	Fortify awards two stars to projects that undergo a Fortify security review that identifies no high likelihood / high impact issues. Vulnerabilities that are trivial to exploit and have a high business or technical impact should never exist in business-critical software.
★★★☆☆	Fortify awards three stars to projects that undergo a Fortify security review that identifies no low likelihood / high impact issues and meets the requirements needed to receive two stars. Vulnerabilities that have a high impact, even if they are non-trivial to exploit, should never exist in business critical software.
★★★★☆	Fortify awards four stars to projects that undergo a Fortify security review that identifies no high likelihood / low impact issues and meets the requirements for three stars. Vulnerabilities that have a low impact, but are easy to exploit, should be considered carefully as they may pose a greater threat if an attacker exploits many of them as part of a concerted effort or leverages a low impact vulnerability as a stepping stone to mount a high-impact attack.
★★★★★	Fortify awards five stars to projects that undergo a Fortify security review that identifies no issues.

Likelihood and Impact

Likelihood

Likelihood is the probability that a vulnerability will be accurately identified and successfully exploited.

Impact

Impact is the potential damage an attacker could do to assets by successfully exploiting a vulnerability. This damage can be in the form of, but not limited to, financial loss, compliance violation, loss of brand reputation, and negative publicity.

Fortify Priority Order

Critical

Critical-priority issues have high impact and high likelihood. Critical-priority issues are easy to detect and exploit and result in large asset damage. These issues represent the highest security risk to the application. As such, they should be remediated immediately.

SQL Injection is an example of a critical issue.

High

High-priority issues have high impact and low likelihood. High-priority issues are often difficult to detect and exploit, but can result in large asset damage.
These issues represent a high security risk to the application. High priority issues should be remediated in the next scheduled patch release.

Medium

Medium-priority issues have low impact and high likelihood. Medium-priority issues are easy to detect and exploit, but typically result in small asset damage.
These issues represent a moderate security risk to the application. Medium-priority issues should be remediated in the next scheduled

Low

Low-priority issues have low impact and low likelihood. Low-priority issues can be difficult to detect and exploit and typically result in small asset damage.
These issues represent a minor security risk to the application. Low priority issues should be remediated as time allows.

Issue Status

New

New issues are ones that have been identified for the first time in the most recent analysis of the application.

Existing

Existing issues are issues that have been found in a previous analysis of the application and are still present in the latest analysis.

Reopened

Reopened issues have been discovered in a previous analysis of the application but were not present in subsequent analyses. These issues are now present again in the most recent analysis of the application.

Fortify Remediation Effort

Major Remediation

Major remediation effort issues must often be addressed at multiple locations to fix the root problem.

Minor Remediation

Minor remediation effort issues can typically be addressed at the location of the root problem.