Dictamen final sobre Auditoría de Seguridad de la Información

Reporte de Pruebas de Penetración

En este reporte describe a que sistemas se pudo tener acceso y con qué herramientas se hizo dicha prueba. Así también se detalla el método, plataforma y escenario bajo el cual se pudo penetrar a las Plataformas.

Marcos Piñeiro Villegas, Yair Rodríguez Aparicio
marcos.pineiro@infotec.mx, yair.rodriguez@infotec.mx

Índice

Versión	Fecha	Descripción
2.0	04/Diciembre/2020	Resultados de las pruebas de penetración

Dictamen elaborado por Yair Rodríguez Aparicio en coordinación con Marcos Piñeiro Villegas.

1. **Resumen Ejecutivo**

En este documento se presenta las herramientas y metodologías que se utilizó en las pruebas de penetración al sistema solicitado por la Dirección ejecutiva.

2. **Metodología**

Para llevar acabo estas pruebas se baso en el esquema de Penetration Testing Execution Standard (**PTES -** http://www.pentest-standard.org/index.php/Main_Page) adecuado para los requerimientos y alcances de tiempo que se tenían en este análisis. De este modo se siguieron los pasos establecidos de acuerdo al estándar:

- Interacciones de enlace – Se atiende la solicitud para el monitoreo y pruebas de penetración y calidad de software del sistema firco v2.
- Reunir inteligencia – Se reunió datos sobre el proyecto firco v2, en cuanto el código del sistema y la base de datos del mismo.
- Modelación de riesgos – Mediante entrevistas y cuestionarios, se pudo modelar las amenazas, vulnerabilidades y determinar los riesgos de que se presentaran dichas actividades bajo un esquema cuantitativo.
- Análisis de vulnerabilidades – Se realizó un escaneo con la plataforma Sonarqube.
- Explotación de vulnerabilidades – Se utilizaron plataformas automatizadas mediante las cuales se pudiese explotar las vulnerabilidades surgidas en el análisis. Estas herramientas permitieron que, dado el tiempo con el que se contaba, se pudiera tener un análisis automatizado para poder revisar el proyecto firco v2.
- Pos Explotación – Durante esta fase se retiene control de los equipos comprometidos para valorar su uso en alguna fase posterior de estudio. Esta fase no se llevo acabo
- Reporte – El reporte que se compone a esta fase es esta sección.

2.1 **Herramientas**

Las herramientas que se escogieron para este proceso son todas de fuente abierta (open-source) las cuales permiten flexibilizar su configuración,

Los programas que se utilizaron para estas pruebas fueron los siguientes:

- **Docker -** Proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos.
- **Sonarqube** – es una plataforma para evaluar código fuente. Es software libre y usa herramientas de análisis estático de código fuente para obtener métricas que pueden mejorar la calidad del software.

2.2 **Ambiente**

Las pruebas de penetración se llevaron acabo con el análisis estático del código fuente con un ambiente en Docker, montando una máquina virtual de sonarqube.

3. **Pruebas de penetración**

Se realizó el análisis estático del código fuente en la máquina virtual de Sonarqube, haciendo un análisis de todos los archivos del proyecto con el siguiente resultado:

4. **Conclusiones**

Se concluye la prueba de penetración con **8722 bugs**, 8 **tipos de vulnerabilidades** detectadas, distribuidas en **1228 puntos de acceso** de seguridad para revisar y **60621 issues** de code smell.

4.1 **Trabajos futuros**

Cómo trabajo futuro, se recomienda realizar una re-ingeniería del proyecto o en su defecto, solventar todas las vulnerabilidades, bugs y el code smell del proyecto.

Prioridad de revisión

Prioridad de revisión Alta: 56

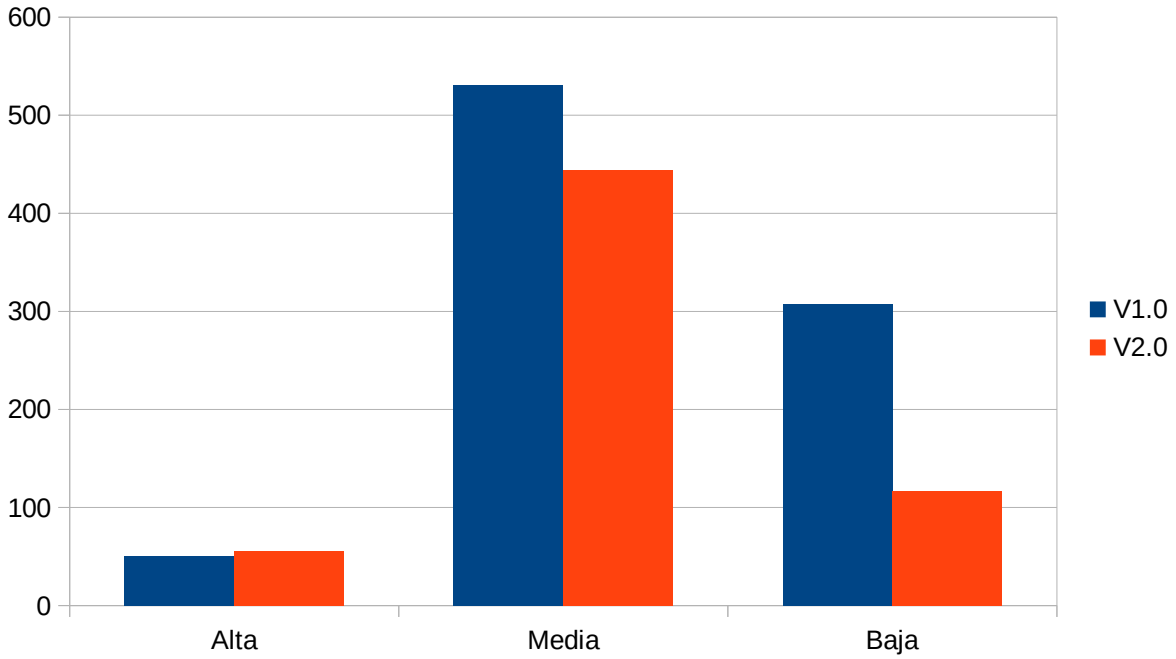
- Authentication
- Command injection
- SQL Injection
- Cross-Site Scripting (XSS)

Prioridad de revisión Mediana: 444

- Denial of Service (DoS)
- Code Injection (RCE)
- Weak Cryptography

Prioridad de revisión Baja: 117

- Insecure Configuration



Gráfica 1. Comparación de vulnerabilidades entre 1er y segundo análisis.

5. **Diccionario de datos**

bug: un error de software o fallo, problema en un programa
issue: se atribuye a la unidad de trabajo para realizar una mejora en un sistema de información.
code smell: es cualquier síntoma en el código fuente de un programa que posiblemente indica un problema más profundo.
Authentication: el acto o proceso de confirmar que algo es quien dice ser.
Command Injection: es un ataque cuyo objetivo es la ejecución de comandos arbitrarios en el sistema operativo a través de una aplicación vulnerable.
SQL Injection: es un método de infiltración de código intruso.
Cross-Site Scripting (XSS): Una secuencia de comandos en sitios cruzados o Cross-site scripting es un tipo de vulnerabilidad informática o agujero de seguridad típico de las aplicaciones Web..
Denial of Service (DoS): ataque de denegación de servicio, cause que el recurso sea inaccesible.
Code Injection (RCE): un defecto de inyección de código, ocurre cuando es posible enviar datos inesperados a un intérprete.
Weak Cryptography: Una vulnerabilidad que se refiera a una técnica de criptografía deficiente o nula.
Insecure Configuration: Vulnerabilidad de una configuración mal implementada.