



INFORME DE PRUEBAs DE

VULNERABILIDAD producción

yo cuido lo público Móvil

Dirección de Gobierno Digital

Bogotá, D.C, octubre de 2017

CONTROL DE CAMBIOS

| **VERSIÓN** | **FECHA** | **No. SOLICITUD** | **RESPONSABLE** | **DESCRIPCIÓN** |
| --- | --- | --- | --- | --- |
| 1.0 | 2017-10-09 | No aplica | Servinformación | Actualización del documento |

TABLA DE CONTENIDO

[1. INTRODUCCIÓN 5](#_Toc496474946)

[2. INFORME DE PRUEBAS DE VULNERABILIDAD 6](#_Toc496474947)

[2.1 FORMATO Y HERRAMIENTAS 6](#_Toc496474948)

[2.2 IDENTIFICACIÓN DE VULNERABILIDADES ANDROID 6](#_Toc496474949)

[3. CONCLUSIONES 9](#_Toc496474950)

LISTA DE tablas

[Tabla 1. Formato para presentación de Vulnerabilidades 6](#_Toc496474954)

[Tabla 2. Pruebas Realizadas a Nivel de Aplicación 7](#_Toc496474955)

[Tabla 3. Documentación de Vulnerabilidades 7](#_Toc496474956)

.

1. INTRODUCCIÓN

E

ste documento presenta el informe de los resultados obtenidos en la ejecución de las pruebas de vulnerabilidad realizadas a la aplicación móvil Yo Cuido Lo Público, para Android en el ambiente de producción diseñado para el desarrollo de estas pruebas.

El informe de pruebas de vulnerabilidad presenta las vulnerabilidades encontradas a nivel de las aplicaciones desarrolladas para Android.

El desarrollo de las pruebas de vulnerabilidad incluye: pruebas dinámicas y estáticas del código en la plataforma android. El código completo para Android fue analizado, realizando la simulación de equipos y el funcionamiento del software, junto a revisión de líneas de código.

La presentación de las vulnerabilidades encontradas en la solución muestra el nivel de las mismas y se genera recomendaciones de aseguramiento basado en el riesgo que genera cada una de ellas para la solución.

El informe busca documentar las pruebas realizadas, y los resultados de cada una de ellas. Las pruebas son realizadas con base en las características propias de la aplicación y la plataforma.

1. INFORME DE PRUEBAS DE VULNERABILIDAD

* 1. FORMATO Y HERRAMIENTAS

El formato para presentar las vulnerabilidades se muestra a continuación:

Tabla 1. Formato para presentación de Vulnerabilidades

|  |  |
| --- | --- |
| **Nombre de la Vulnerabilidad** | |
| **Nivel de Riesgo Calculado** | |
| **Critico** | El acceso al sistema puede hacerse desde un sitio remoto sin requerir autenticación. El sistema sería fácil y seriamente comprometido de aprovecharse esta vulnerabilidad. |
| **Alto** | La explotación de la vulnerabilidad proveería acceso al sistema con privilegios de administración. El sistema seria seriamente comprometido. |
| **Medio** | La explotación de la vulnerabilidad permitiría acceso indirecto a datos o archivos de configuración. |
| **Bajo** | La explotación de la vulnerabilidad podría conducir al atacante a obtener estadísticas del sistema, cuentas de usuarios o alguna otra información sensitiva que ayudarían a ejecutar un ataque. |
| **Información** | Información complementaria |
| **Categoría** | Categoría a la que pertenece la vulnerabilidad |
| **Descripción** | Descripción de la vulnerabilidad |
| **Impacto** | Impacto que tendría la vulnerabilidad en caso de explotarse |
| **Sistemas o Código Afectado** | Sistemas, scripts o páginas web afectados por la vulnerabilidad afectados por la vulnerabilidad |

La descripción de las vulnerabilidades es presentada al final del documento, como resumen de los hallazgos realizados, dentro de las tareas de las pruebas de vulnerabilidad.

* 1. IDENTIFICACIÓN DE VULNERABILIDADES ANDROID

Se realizan pruebas específicas relacionadas a la aplicación web, que se enmarcan en el TOP 10 Mobile de OWASP, de vulnerabilidades en aplicaciones. Adicionalmente se realizan pruebas específicas de código enfocadas a pruebas estáticas nativas para la aplicación realizada en Android.

A continuación se presentas las pruebas a nivel de aplicación.

Tabla 2. Pruebas Realizadas a Nivel de Aplicación

| **Prueba** | **Resultado** |
| --- | --- |
| Almacenamiento Inseguro de Datos | No se encontró almacenamiento inseguro de datos sensibles |
| Controles débiles a nivel del servidor | No se ha revisado la conectividad con el servidor. |
| Protección insuficiente a nivel de transporte | Dentro de la plataforma y la programación de los servicios y aplicaciones, se tuvieron en cuenta las protecciones necesarias a nivel de transporte |
| Inyecciones a nivel de cliente | Dentro de las pruebas dinámicas se realizaron pruebas al respecto y se comprobó que no existen ataques exitosos de este tipo |
| Autenticación o Autorización indebida | En la definición de componentes y programación de las aplicaciones estos componentes fueron tenidos en cuenta |
| Mal manejo de sesiones | Se establecieron controles para el correcto manejo de sesiones |
| Decisiones de seguridad realizadas por medio de entradas no confiables | No existen elementos no confiables que puedan realizar decisiones de seguridad dentro de la solución |
| Fuga de información por canales alternos | Las comunicaciones establecidas por la solución solamente confían en puntos específicos de conexión, por lo cual este riesgo no se encuentra asociado a la aplicación. |
| Métodos Criptográficos Inseguros | Se definen métodos estándar a utilizar que están dentro del TOP 10 Mobile de OWASP (Open Web Application Security Project). |
| Fuga de Información Sensible | Dentro de la aplicación no se hace manejo de información considerada como sensible. |

A continuación, se presenta la clasificación de cada una de las vulnerabilidades encontradas. La descripción de las vulnerabilidades incluye la información relacionada a la misma a nivel de resumen.

Tabla 3. Documentación de Vulnerabilidades

|  |
| --- |
| **Uso de Literales en Condicionales** |
| Bajo |
| Manejo de Variables |
| Fue posible identificar literales en algunos condicionales utilizados en algunos condicionales del código |
| **Uso de alcance explicito** |
| Bajo |
| Fuga de Información |
| Algún paquete o método tiene visibilidad de paquetes. Todos los paquetes que se encuentren dentro de las clases deben estar escondidos. |
| Recolección de Información de la aplicación |
| **Cada clase debe tener como mínimo un constructor** |
| Bajo |
| Definición de Constructores en Clases |
| Es necesario definir un constructor como mínimo para la clase. Es posible inicializar un objeto de manera arbitraria si el constructor no se encuentra inicializado en la clase. |
| **Incorrecta Estructura de Métodos** |
| Medio |
| Definición de Métodos |
| Cada uno de los métodos debe tener solamente un punto de salida y debe ser la última sentencia del método, con el fin de evitar posibles saltos a las definiciones propias de los métodos. |

1. CONCLUSIONES

S

egún los hallazgos se generan las siguientes recomendaciones, que permitirán la mitigación de las vulnerabilidades encontradas, las cuales no son críticas para la aplicación, pero generan posibles vectores de ataque.

* Las vulnerabilidades encontradas, son fácilmente remediables y no generan una mayor afectación a la confidencialidad o integridad de los datos manejados por las aplicaciones.
* Se deben mitigar las vulnerabilidades documentadas en este informe y documentar los cambios en las aplicaciones, del mismo modo que revisar la posibilidad de realizar los cambios descritos en el informe.
* Los cambios descritos, son cambios menores, que no deben generar afectación alguna en las aplicaciones.
* Es necesario realizar las revisiones pertinentes dentro del código con el fin de mitigar las vulnerabilidades descritas dentro de este informe.
* Se recomienda evitar el uso de literales en declaraciones de condicionales. Cuando se realice la declaración de variables estáticas el mantenimiento del código se vuelve más complejo.
* Se debe generar una estructuración del código que asegure su correcta codificación, teniendo en cuenta los puntos de salida del código, teniéndolos en cuenta como la última línea de las sentencias, con el fin de evitar posibles adiciones que permitan la manipulación de métodos.