PROYECTO DE FIN DE CICLO

DESARROLLO DE APLICACIONES MULTIPLATAFORMA

ARP Spoof Detector

DETECTOR DE SUPLANTACION DE INTERFAZ/DISPOSITIVO DE RED

Autor: Leonardo Martins

leonardo.martins@email.com

Tutor: Daniel Santiago Martinez

PAGINA INTENCIONALMENTE EN BLANCO

TABLA DE CONTENIDO

TABLA DE IMAGENES

1 INTRODUCCIÓN

1.1. RESUMEN

ARP Spoof Detector es una aplicación que permite detectar ataques de "mitm" (man in the middle) que fuesen efectuados mediante la explotación de vulnerabilidades en el protocolo de resolución de direcciones. (ARP address resolution protocol).

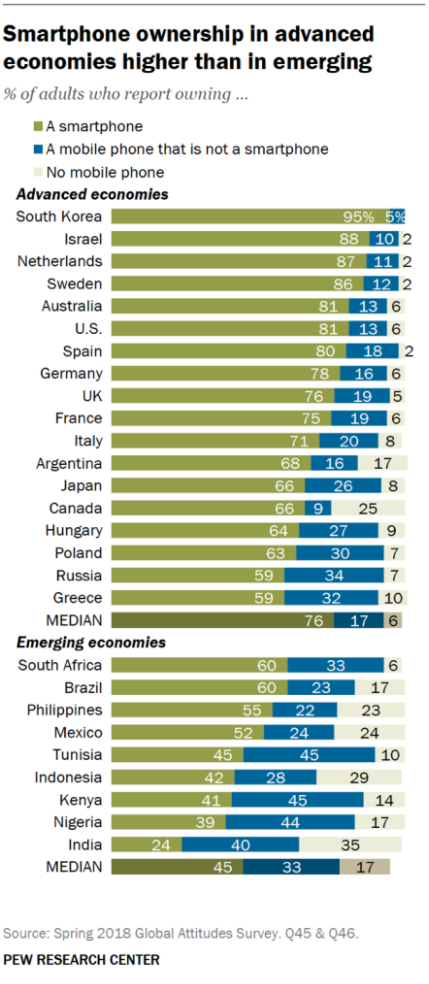
Un porcentaje elevado de la población actual cuenta hoy día con un dispositivo móvil con capacidad de conexión a internet, la cual utiliza diariamente por cuestiones personales o de trabajo. El uso de redes comprometidas por un agente malicioso puede devenir en perdida de privacidad, robo de credenciales o información sensible, entre otras.

El objetivo de este proyecto es la de proveer una herramienta que nos permita identificar redes comprometidas y prevenir los inconvenientes que el uso de la misma posibilitaría.

1.2. MOTIVACIÓN

Hoy día, cuando hablamos de Europa, resulta difícil imaginar a una persona sin un dispositivo móvil con capacidad de acceso a internet, cualquier sea su tipo (Smartphone, Tablet, Laptop). Utilizando España como ejemplo podemos decir que ya en 2018 el porcentaje de utilización de Smartphones entre adultos era del 80% según estudios publicados en Pew Research Centre. (https://www.pewresearch.org).

Esta alta disponibilidad de dispositivos no se ve reflejada en el costo de acceso a internet móvil que los diferentes proveedores de servicios ofrecen y, en general, lo que se ofrece dentro de la línea "económica" representa una cantidad limitada de datos (en Gigabytes) entre periodos de facturación. Esta cantidad finita de datos deviene en escases si tenemos en cuenta los servicios actuales de Streaming o la compartición de archivos multimedia que utilizamos día a día cuando nos encontramos fuera de casa tanto por ocio o por trabajo.



1 Porcentaje de utilización de dispositivos telefónicos. (Smartphone u otros) en países desarrollados y determinadas economías emergentes.

No es extraño entonces que tengamos por costumbre la utilización de redes "publicas" durante este tiempo, siendo estas redes públicas las que encontramos

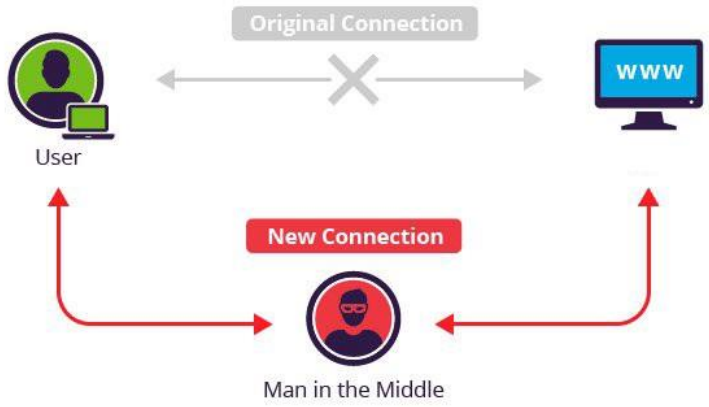
en un Café, Restaurant, etc.; que a su vez publicitan el acceso a su red Wi-fi como método de captación de clientes.

¿Podemos considerar como seguras a estas redes?

Para la mayoría de los casos la respuesta a esta pregunta es NO. En general se puede afirmar que este tipo de lugares poseen una configuración similar a una red hogareña o de pequeña oficina. Este tipo de redes son particularmente vulnerables a ataques de Man-in-the-Middle, siendo este tipo de ataque uno de los más populares a nivel mundial. Una forma de ejecución de este ataque se realiza mediante la explotación de vulnerabilidades dentro del protocolo ARP (Address Resolution Protocol) con la cual es posible impostar el MAC ID de un dispositivo y obtener así cierto control en el flujo de datos. Con este control es posible comprometer la integridad de los datos que manejamos o robar información, entre otros riesgos a los que nos exponemos.

1.3. MITM Y ARP SPOOFING / POISONING

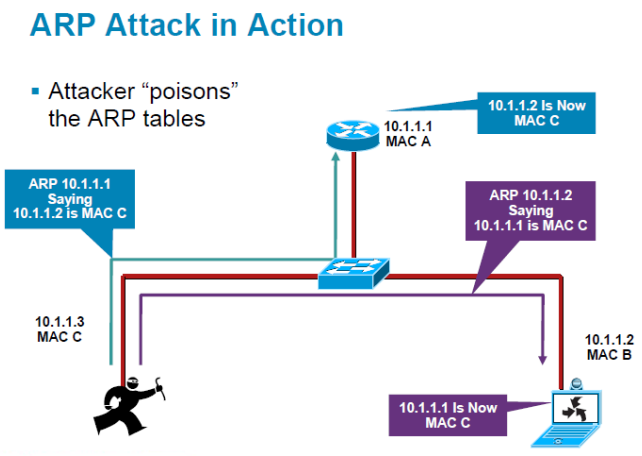
Con motivo de establecer una mejor comprensión sobre el software desarrollado es importante especificar que es un ataque Man in the middle y particularmente via ARP Spoofing.



2 Representación gráfica de un ataque MITM

Un ataque Man in the middle (o Janus Attack) es un tipo de ataque en el cual se adquiere la capacidad de leer, insertar y modificar la información entre dos nodos (usuarios, dispositivos, etc) sin que estos noten tal violación. Una de las formas de lograr esto es mediante la explotación de vulnerabilidades en el protocolo ARP (Address Resolution Protol). Cada dispositivo cuenta con una codigo de identificación único conocido como MAC (Media Access Control) Address o ID. El MAC ID está conformado por una combinación de 6 pares de dígitos o caracteres, separados entre sí por ":" o "- "; los primeros 3 pares corresponden al llamado OUI (Organizationally Unique Identifer) que nos permite identificar quien es el fabricante de dicho dispositivo. Los restantes 3 pares de dígitos corresponden al número único de dispositivo fabricado por esa empresa.

Sin entrar en detalle podemos afirmar que es posible, sin demasiado esfuerzo, el modificar el MAC ID de un dispositivo y de esta forma inducir a otro dispositivo a enviarnos información. que originalmente debía ser entregada a un tercer dispositivo.



3 Diagrama de un ataque ARP Spoofing

1.4. MÉTODOS DE DETECCIÓN

Si bien la mayoría de los dispositivos Router que se utilizan hoy día son capaces de neutralizar este tipo de ataques esta opción suele no estar configurada por defecto. Además, es normal encontrar dispositivos sin esta funcionabilidad aun operando en bares, restaurantes e incluso oficinas.

De igual forma podríamos detectar este proceso de forma manual utilizando la herramienta arp de nuestro sistema operativo (en Windows y MacOS esta herramienta está incluida por defecto, pero no así en todas las distribuciones de Linux, donde es parte del paquete net-tools). La imagen 2.3 corresponde a una consulta arp donde se aprecia que el MAC ID de uno de los dispositivos ha sido impostada y por ende se encuentra duplicada en la tabla).

```
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\el_le>arp -a -N 192.168.0.39
ARP: bad argument: 192.168.0.39

C:\Users\el_le>arp -a -N 192.168.0.38

Interface: 192.168.0.38 --- 0xb
  Internet Address      Physical Address      Type
  192.168.0.1           f4-6b-ef-d3-ff-ff     dynamic
  192.168.0.39          28-e3-47-22           dynamic
  192.168.0.86          f4-6b-ef-d3-ff-ff     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\el_le>_
```

4 Tabla arp. Con ataque ARP spoofing en proceso. (Nota: la imagen ha sido alterada para preservar el ID real de los dispositivos)

1.5. OBJETIVO

La intención de este proyecto es la de desarrollar una aplicación que realice un scan de la red donde el dispositivo se encuentra conectado y verifique si existe un MAC ID duplicado dentro de la tabla ARP. Tal aplicación podrá ser utilizada bajo sistemas operativos Linux y Windows, además de una versión Android. Si el resultado fuese positivo el usuario recibiría una alerta que le permitiría entender que la privacidad e integridad de sus datos están en peligro.

1.6. PÚBLICO OBJETIVO

ARP Spoof Detector ha sido ideado como herramienta para aquellas personas que, aunque consientes de los riesgos que la utilización de redes públicas conlleva, no cuentan con la capacidad técnica para protegerse. De igual forma la aplicación puede ser útil incluso para usuarios avanzados debido a la automatización del proceso y su posibilidad de funcionamiento en segundo plano.

1.7. ESTRUCTURA DEL DOCUMENTO

El presente documento recoge las diferentes fases de realización del proyecto. El orden de aparición de los mismos intenta establecer una representación cronológica de como se ha estructurado el trabajo.
En los siguientes capítulos encontraremos entonces información acerca de la elección de los métodos de desarrollo, así como de los procesos que resultan del mismo y finalizando con la presentación de una conclusión y propuestas a futuro.

2 DESARROLLO

2.1. METODOLOGÍA DE DESARROLLO

La metodología de desarrollo refiere a un entorno o marco de desarrollo en el cual se consigue estructurar, planificar y controlar el proceso de desarrollo.

Para la realización de este proyecto utiliza un modelo de cascada como metodología de desarrollo. Este mismo es el primer modelo en ser aplicado a operaciones de desarrollo de software y puede ser considerado antiguo. No obstante, analizando pros y contras de la implementación del mismo podemos considerar que se puede ajustar a este proyecto sin inconvenientes.

5 Modelo "cascada"

2.2. VENTAJAS E INCONVENIENTES GENERALES DEL DESARROLLO EN CASCADA.

A. Ventajas

- El tiempo que se pasa en diseñar el producto en las primeras fases del proceso puede evitar problemas que serían más costosos cuando el proyecto ya estuviese en fase de desarrollo.
- Ideal para proyectos estables, donde los requisitos son claros y no van a cambiar a lo largo del proceso de desarrollo

B. Inconvenientes

- Para proyectos a largo plazo, este modelo puede suponer un problema al cambiar las necesidades del usuario a lo largo del tiempo.

- Los diseñadores pueden no tener en cuenta todas las dificultades que se encontrarán cuando estén diseñando un software, lo que conllevará rediseñar el proyecto para solventar el problema.
- No se va mostrando al cliente el producto a medida que se va desarrollando, sino que se ve el resultado una vez ha terminado todo el proceso. Esto cual provoca inseguridad por parte del cliente que quiere ir viendo los avances en el producto
- En muchas ocasiones, los clientes no saben bien los requisitos que necesitarán antes de ver una primera versión del software en funcionamiento.

2.3. REQUERIMIENTOS DE SOFTWARE

2.3.1 REQUERIMIENTOS FUNCIONALES SOFTWARE

- La aplicación debe detectar entradas duplicadas dentro de la tabla arp
- La aplicación es capaz de correr en múltiples plataformas (Linux, Windows, Android).
- La aplicación puede correr en segundo plano (minimizada)
- La aplicación debe generar una alerta en caso de detectar una posible amenaza.
- La frecuencia de búsqueda es configurable.
- La alerta puede ser desactivada sin que el programa deje de funcionar.
- La aplicación permite el envió de un email de alerta en caso de detección.

2.3.2 REQUERIMIENTOS NO FUNCIONALES SOFTWARE

- La IU es simple y no ejerce una disminución de la performance.
- Disponibilidad para diferentes dispositivos Android. (Smartphones o tablets)

3 ANÁLISIS Y DISEÑO

3.1. INTRODUCCIÓN

En esta sección se intentará describir el proceso de análisis y diseño la aplicación intentando otorgar una idea completa del software desarrollado.

3.2. HERRAMIENTAS UTILIZADAS

3.2.1. APACHE NETBEANS

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE[1] es un producto libre y gratuito sin restricciones de uso.
NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos (Actualmente Sun Microsystems es administrado por Oracle Corporation).

3.2.2. JAVA (JDK8)

Java es un lenguaje de programación y una plataforma informática que fue comercializada por primera vez en 1995 por Sun Microsystems. Hay muchas aplicaciones y sitios web que, probablemente, no funcionarán a menos que tenga Java instalado y cada día se crean más. Java es rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde consolas para juegos hasta computadoras avanzadas, desde teléfonos móviles hasta Internet, Java está en todas partes, que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra.

3.2.3. JAVA FX

JavaFX es una familia de productos y tecnologías de Oracle Corporation (inicialmente Sun Microsystems), para la creación de Rich Internet Applications (RIAs), esto es, aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio, incluyendo aplicaciones multimedia interactivas. Las tecnologías incluidas bajo la denominación JavaFX son JavaFX Script y JavaFX Mobile, aunque hay más productos JavaFX planeados

3.2.4. ANDROID STUDIO

Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014.
Está basado en el software IntelliJ IDEA de JetBrains y ha sido publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, macOS y GNU/Linux. Ha sido diseñado específicamente para el desarrollo de Android.

3.2.5. MAC ADDRESS LOOK UP API

"Mac address look up" es un sitio web que proporciona una REST API con la cual es posible obtener información sobre los diferentes fabricantes de dispositivos de red. Dicha API puede utilizarse de forma gratuita y sin necesidad de una llave API.
https://maclookup.app/

3.2.6. ORACLE VIRTULBOX

Oracle VM VirtualBox es un software de virtualización para arquitecturas x86/amd64. Actualmente es desarrollado por Oracle Corporation como parte de su familia de productos de virtualización. Por medio de esta aplicación es posible instalar sistemas operativos adicionales, conocidos como «sistemas invitados», dentro de otro sistema operativo «anfitrión», cada uno con su propio ambiente virtual.

3.2.7. SCENE BUILDER

Scene Builder es una herramienta de despliegue visual para aplicaciones JavaFX.
Usuarios pueden utilizar funciones de "drag and drop" de componentes visuales hacia un área de trabajo, modificar sus propiedades y aplicar hojas de estilo. La aplicación genera un fichero FXML con la información de despliegue del Proyecto y este fichero puede luego ser combinado con la aplicación java vinculando la lógica del Proyecto con la interfaz de usuario.

3.2.8. STARUML

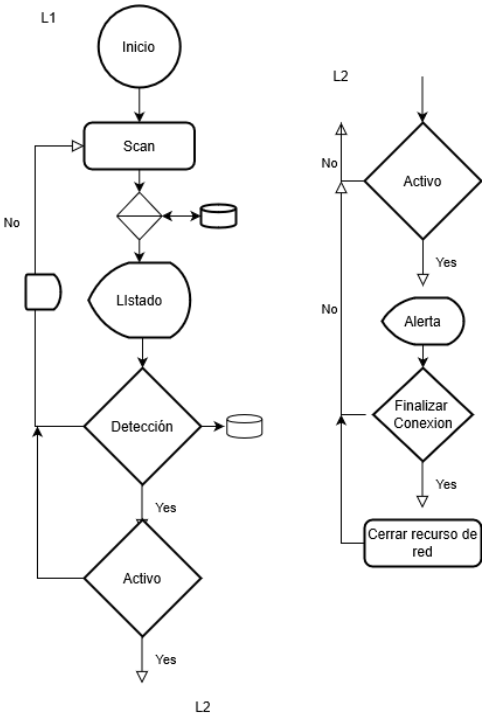
StarUML es una herramienta de Desarrollo de diagramas UML desarrollada por MKLab. Soporta la mayoría de los diagramas especificados en UML 2.0.

3.2.9. WIX TOOLSET

Wix Toolset (conocido como Wicks) es una aplicación de software para la creación de paquetes de instalación de Windows a partir de ficheros XML.

3.3. LÓGICA DE LA APLICACIÓN

De acuerdo con los requisitos de la aplicación podemos elaborar el diagrama de flujos que servirá como base para el proceso de codificación. Este diagrama corresponde a la idea original del proyecto y el resultado final tiene leves variaciones en cuanto a la logica de la aplicacion.



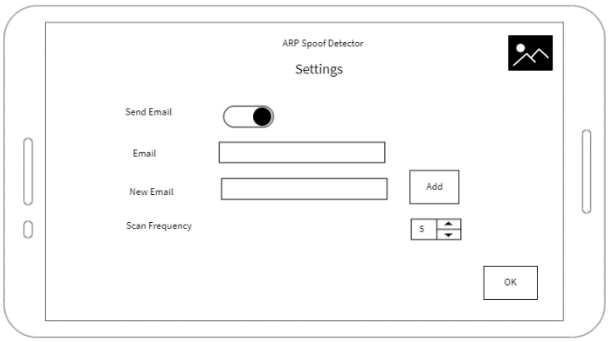
6 Diagrama de flujo

3.4. ELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN

Como se ha mencionado en el apartado 3.2 el lenguaje utilizado para la codificación es Java, particularmente en su versión 1.8 (Conocido también como Java8).
La elección del mismo se debe a que el mismo corre sin problemas independientemente de la plataforma en la que se utiliza. El compilador de Java convierte el codigo en bytecode que es un lenguaje intermedio. Este es luego ejecutado por el SO utilizando Java Virtual Machine. El único prerrequisito es la instalación de JRE que es fácilmente accesible online.
Además, siendo que Java es uno de los lenguajes oficiales de desarrollo para Android por lo que la reutilización de codigo es posible en gran parte. Adecuación del mismo será necesario, no obstante, de acuerdo a particularidades en el sistema Android.

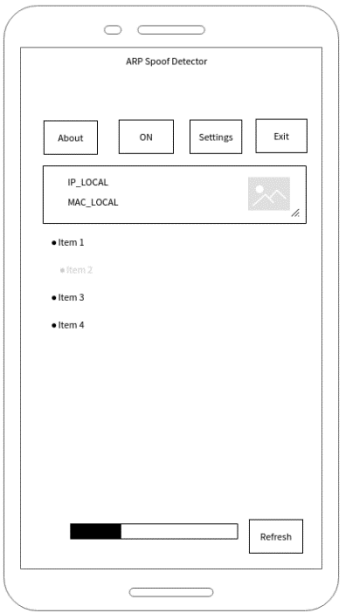
3.5. DISEÑO DE INTERFAZ DE USUARIO

3.5.1. WIREFRAMES ANDROID



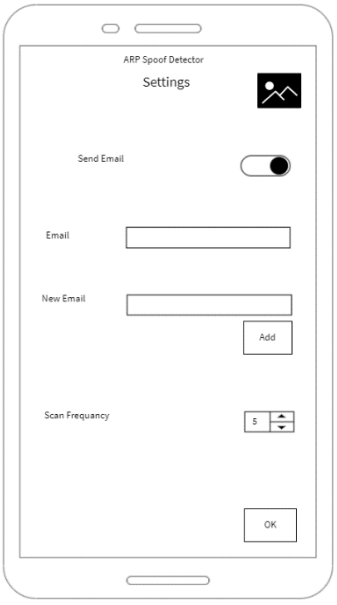
8 Wireframe Landscape Settings

7 Wireframe Landscape Main



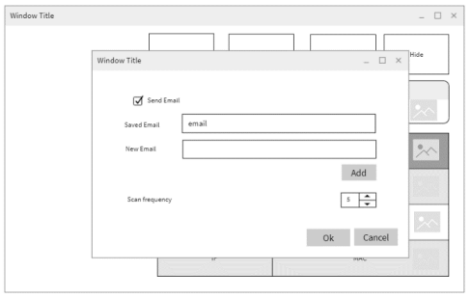
10 Wireframe Portrait Main

11 Wireframe Portrait Alert

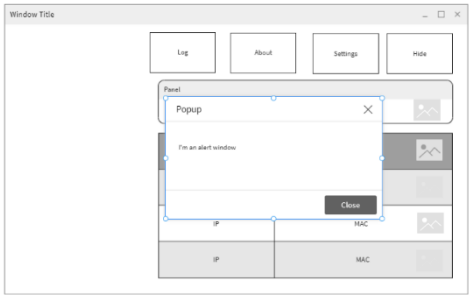


9 Wireframe Portrait Settings

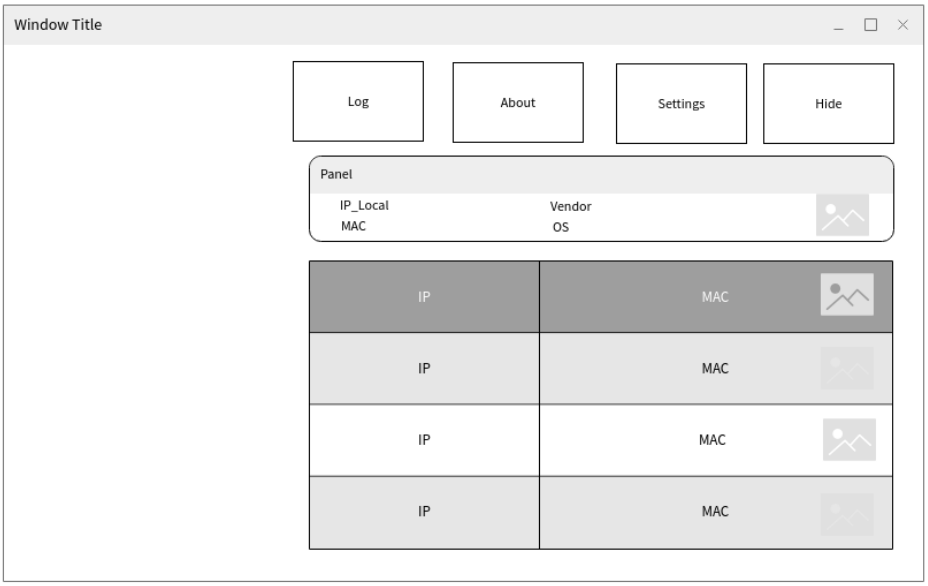
3.5.2. WIREFRAMES PC



12 WireFrame PC Settings



13 WireFrame PC Alert



14 12 WireFrame PC Main

3.5.3. DISEÑO DE LA INTERFAZ (PC) CON JAVAFX

La implementación de JavaFX como el generador de la interfaz gráfica de la aplicación es uno de los cambios más notorios en la realización del proyecto. Originalmente la interfaz del usuario para la aplicación en Windows / Linux seria realizada utilizando JavaSwing.

En una comparación 1:1 encontramos:

	JavaSwing	JavaFX
Componentes	Swing contiene mas componentes.	Menor cantidad de componentes en comparación con componentes Legacy de Swing.

UI	Componentes estándar pueden ser creados. Los componentes son poco flexibles.	Componentes GUI avanzados pueden crearse, con componentes menos estáticos.
Desarrollo	La codificación de componentes se realiza a través de APIs Swing.	El desarrollo de componentes se puede realizar con Scripts.
Funcionalidad	No se esperan nuevas funcionalidades en el futuro.	Se estima que el kit de herramientas JavaFX crecerá en el futuro.
MVC	La implementación de la arquitectura MVC carece de consistencia en algunos componentes.	Totalmente compatible con la arquitectura. Permite la creación de componentes a partir de un FMXL

JavaFX se encuentra incluido dentro del paquete Java SDK 8. A partir de su versión 11 Java SDK y JavaFX forman parte de módulos separados. Además, la integración para su uso dentro de diferentes IDE puede ocasionar problemas de compatibilidad o errores en el funcionamiento.

3.5. CASOS DE USO

2.3.1 ESTRUCTURA DE CASO DE USO

Las diferentes versiones del software han sido desarrolladas para seguir una misma lógica, por lo que salvo especificado en el CU el funcionamiento es el mismo para cualquier de sus versiones.
Nombre: Identificación del caso de uso.
- Descripción: Descripción de la funcionalidad
- Actores: Actores relacionados con el caso de uso.
- Precondiciones: Acciones o hechos que se han de cumplir para que el caso de uso se pueda iniciar.
- Flujo normal: serie de eventos que se desarrollan si el caso de uso se ejecuta sin problemas.
- Flujo Alternativo: serie de eventos que se desarrollan si se produce algún fallo al ejecutar el caso de uso o si el mismo depende de variables condicionales.
- Postcondiciones: Acciones o hechos que se cumplirán si el flujo de eventos normal se ha ejecutado correctamente.

Nombre	CU01-INTALL
Descripción	Instalación de la aplicación (PC)

Actores	Usuario
Precondiciones	JRE 1.8 Instalado
Flujo Normal	• Usuario ejecuta el archivo de instalación. • Proceso de instalación en curso. • Finalización del proceso de instalación.
Flujo Alternativo	Ninguno
Postcondiciones	ARP Spoof Detector Instalado

Nombre	CU02-OPEN
Descripción	Se inicializa la aplicación
Actores	Usuario / Sistema
Precondiciones	Ninguna
Flujo Normal	• Usuario abre la aplicación (Inicio en Systray en Windows) • Sistema realiza una comprobación de la disponibilidad de red. • Sistema genera un listado de dispositivos conectados a la red. • La aplicación corre en segundo plano, minimizada
Flujo Alternativo	Sistema muestra una alerta al no identificar conexión red.
Postcondiciones	

Nombre	CU03-FOCUS
Descripción	Foco en aplicación (Solo Windows)
Actores	Usuario
Precondiciones	Ninguna
Flujo Normal	• Usuario abre la selecciona la opción abrir en Systray • Se despliega la vista principal.
Flujo Alternativo	Sistema muestra una alerta al no identificar conexión red.
Postcondiciones	Vista Principal Visible

Nombre	CU04-OPEN-SETTINGS
Descripción	Abre la ventana de Opciones de la aplicación
Actores	Usuario
Precondiciones	Modo visible
Flujo Normal	• Usuario selecciona icono “Opciones” en la vista principal • Carga opciones desde archivo de configuración.
Flujo Alternativo	• Sistema crea archivo de configuración si no existiese.
Postcondiciones	Vista opciones visible

Nombre	CU05-ACCEPT-SETTINGS
Descripción	Cierra Opciones (Aceptar)
Actores	Usuario / Sistema
Precondiciones	Vista opciones visible
Flujo Normal	• Usuario selecciona el botón “Aceptar” • Sistema recupera la información y la documenta en el archivo de configuración. • Sistema actualiza valores en el la clase principal-
Flujo Alternativo	
Postcondiciones	Ventana de opciones cerrada. Vista principal en primer plano.

Nombre	CU06-CANCEL-SETTINGS
Descripción	Cierra Opciones (Cancelar)
Actores	Usuario
Precondiciones	Vista opciones visible
Flujo Normal	• Usuario selecciona el botón “Cancelar”
Flujo Alternativo	
Postcondiciones	Ventana de opciones cerrada. Vista principal en primer plano.

Nombre	CU07-ALERT
Descripción	Detecta
Actores	Sistema

Precondiciones	Notificación Activa, servicio de envió de email activo
Flujo Normal	• Sistema detecta posible amenaza • Sistema muestra ventana de alerta • Sistema envía email de advertencia.
Flujo Alternativo	• Sistema no muestra mensaje de alerta si la opción esta desactivada. • Sistema envía email de notificación si la opción esta seleccionada
Postcondiciones	

Nombre	CU08-OPEN-ABOUT
Descripción	Abre la ventana de información de la aplicación
Actores	Usuario
Precondiciones	Modo visible
Flujo Normal	• Usuario selecciona icono “About” en la vista principal
Flujo Alternativo	
Postcondiciones	Vista “About” visible

Nombre	CU09-HIDE
Descripción	Oculta la vista principal de la aplicación
Actores	Usuario
Precondiciones	Modo visible
Flujo Normal	• Usuario selecciona icono “Hide” en la vista principal
Flujo Alternativo	
Postcondiciones	Vista principal oculta, aplicación sigue corriendo en 2do plano. Icono de modificación activo en Systray.

4 CODIFICACIÓN

4.1 INTRODUCCIÓN

Como ha sido mencionado previamente ambas versiones del software (PC / Android) comparten gran parte del codigo. Esta sección se centrará en describir las clases de uso compartido más importantes y sus métodos para luego enfocarse en aquellas que sean de uso especifico para una u otra versión. Un diagrama de clases esta proporcionado en el Anexo I.

4.2 CLASES COMPARTIDAS

4.2.1 public class OsValidator

Clase utilizada para recuperar el tipo de sistema operativo donde la aplicación se esta ejecutando.

4.2.2 public class VendorLookUp

Clase utilizada para recuperar el nombre del fabricante de un dispositivo utilizando el servicio de https://api.maclookup.app/

4.2.3 public class ArpTable

Se utiliza para limpiar la tabla arp en cache además de leer el nuevo flujo de datos y recuperar la información sobre los dispositivos detectados en la misma.
Siendo que los dispositivos solo son visibles luego de comunicarse entre si dispone de un método public final void pingNetwork(String iface) throws IOException que instancia objetos de la clase PingService mediante la ejecución de hilos.

4.2.4 public class PingService implements Runnable

Clase utilizada para enviar paquetes ICMP a una determinada dirección de red dentro de la red en la que la NIC se encuentra conectada.

4.2.5 public class EmailSender

Se encarga del proceso total de envió de una alerta vía email en caso de detección de una posible amenaza.

4.2.6 public class FXMLSettingsController implements Initializable (PC) / Settings (Android)

Funciona como controlador de la vista “Opciones”. La misma se crea a partir de un fxml en la versión de PC. Está asociada a un recurso xml en la versión Android.

4.3 CLASES ESPECÍFICAS (PC)

De acuerdo a la arquitectura de la aplicación la versión para PC contiene las siguientes clases.

4.3.1 public class Local_IpAddress

Recupera la dirección ip del dispositivo conectado a la red.

Nota: Teniendo en cuenta que un dispositivo puede tener mas de un NIC (ya sean físicas o virtuales) el sistema debe establecer cual es aquel que se encuentra conectado a la red en un momento determinado.

4.3.2 public class ArpTable

Se utiliza para limpiar la tabla arp en cache además de leer el nuevo flujo de datos y recuperar la información sobre los dispositivos detectados en la misma.

Siendo que los dispositivos solo son visibles luego de comunicarse entre si dispone de un método public final void pingNetwork(String iface) throws IOException que instancia objetos de la clase PingService mediante la ejecución de hilos.

4.3.3 public class Local_MacAddress

Recupera el identificador MAC del dispositivo conectado a la red. En Windows bajo la ejecución de una consulta al sistema mediante el uso de Windows Management Instrumentation (wmic); en Linux /Android simplemente leyendo el flujo de datos proporcionado por el comando ifconfig.

4.3.4 public class MainAppLauncher extends Application

Es la clase inicial de la aplicación. Contiene el método *main* y el método start, encargado de iniciar los recursos para una aplicación JavaFX y generar la vista principal a través de un fichero fichero .fmxl.

4.3.5 public class FXMLDocumentController implements Initializable

Es el recurso principal del software, contiene la lógica del sistema y actúa como controlador de la vista principal.

4.3.6 public class FXMLLogController implements Initializable

Funciona como controlador de la vista “About”. La misma se crea a partir de un fxml.

4.4 CLASES ESPECÍFICAS (ANDROID)

4.4.1 public class MainActivity extends AppCompatActivity

Es la clase principal de la aplicación. Contiene el método onCreate y la lógica de la aplicación.

Dentro de esta encontramos dos clases internas:

- Node: Cada instanciación corresponde a un dispositivo diferente en la red.
- TaskReadAdresses extends AsyncTask: clase encargada de refrescar la tabla arp, similar a la clase ArpTable. Actualiza parte de la vista en el Activity Principal.

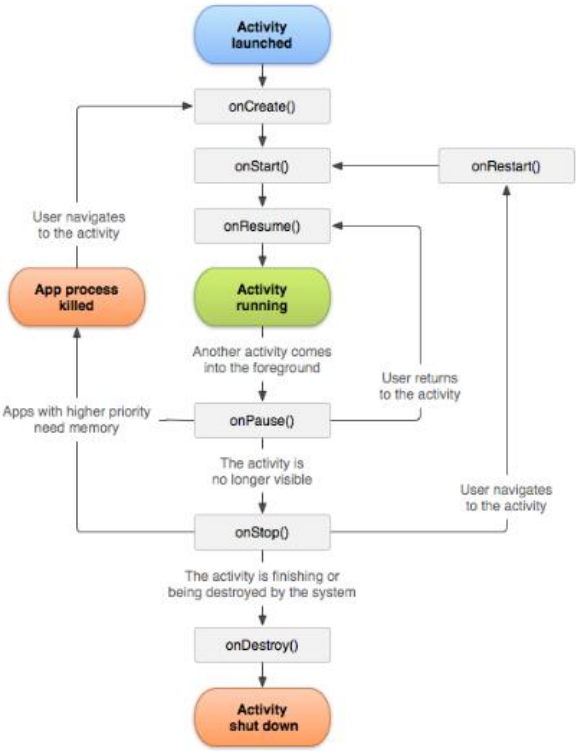
4.5 CONSIDERACIONES PARA EL DESARROLLO EN ANDROID.

4.5.1 Ciclo de vida de una Actividad

En Android una actividad es el componente principal de una aplicación. Se encarga de manejar la interacción con el usuario mediante una interfaz gráfica. Cada pantalla de nuestra aplicación es una actividad.

Cada actividad cuenta con un ciclo de vida y pasa por muchos estados durante el uso de la aplicación. La intención de este segmento no es la de establecer una descripción de cada estado sino la de informar que el desarrollo de este App tiene en cuenta este factor a fin de evitar un funcionamiento defectuoso de la misma.

En particular se ha sobrescrito el método onResume() para poder dar una sensación de continuidad aun cuando la actividad se ve interrumpida por pulsaciones en el botón "HOME", llamadas entrantes, cambios de orientación del visor, etc.



15 Ciclo de vida una actividad en Android

4.5.2 ANR

La capacidad de computación de los dispositivos móviles (smartphones, tablets) es evidentemente menor que los ordenadores personales. Debido a esto es importante que procesos que hagan uso de recursos de forma prolongada ejecuten los mismos en un hilo separado del principal, que es aquel que controla la GUI. Particularmente los procesos que utilicen una conexión a un servidor HTTP deben ser tratados de esta forma para evitar un malfuncionamiento de nuestra app.

4.5.3 OutOfMemoryError Exception

Como ha sido mencionado en el punto 4.3 la clase ArpTable instancia objetos de la clase PingService a través de un nuevo hilo por cada dirección de la subred. El uso de este recurso podría devenir en una

excepción que haría que nuestra aplicación se detenga. Sera la clase TaskReadAdresses la que se encargue de impedir que el usuario dispare esta acción de forma indiscriminada, además de invocar al recolector de basura una vez finalizado el proceso de escaneo de la red.

5 MODIFICACIONES

5.1 INTRODUCCIÓN

La intención de esta sección es la de despejar dudas al respecto de aquellos cambios que se hayan producido entre la idea original, presentada en el informe técnico inicial, y el resultado final.

5.2 BASE DE DATOS (VENDORS)

Una de las funcionalidades de ARP Spoof Detector es la de enumerar los dispositivos conectados a la red y la de presentarnos información sobre su dirección de red y su identificado único (MAC). A su vez, el software es capaz de "identificar" el fabricante de tal dispositivo de red mediante una consulta al servicio de https://maclookup.app/.

La idea original incorporaba la utilización de un fichero XML como base de datos de fabricantes, pudiéndose obtener la misma desde https://www.macvendorlookup.com/.

Con este fichero y la posibilidad de acceder a los datos mediante DOM (Document Object Model) o SAX (Simple Api for XML).

Teniendo en cuenta que el existen cerca de veinte mil registros en el fichero anteriormente mencionado esto hace que la opción sea poco viable.

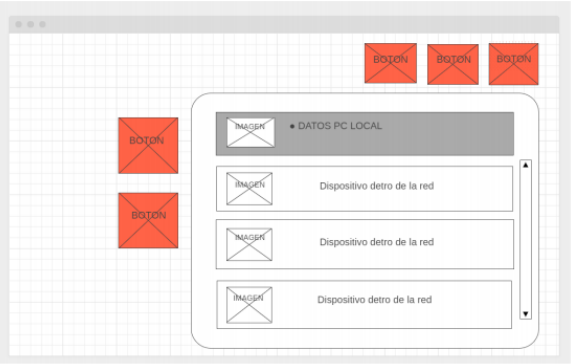
El uso del API de https://maclookup.app/ para la consulta no solo supone una disminución en los recursos utilizados por el sistema y en una menor cantidad de líneas de codigo, dada la simpleza de su implementación.

5.3 INTERFAZ GRÁFICA (PC)

La idea original de la aplicación era la de desarrollar su GUI mediante el uso de Java Swing. La elección de tal recurso estaba basada únicamente en la falta de experiencia con otros recursos.

La implementación de tales componentes supuso 2 problemas:

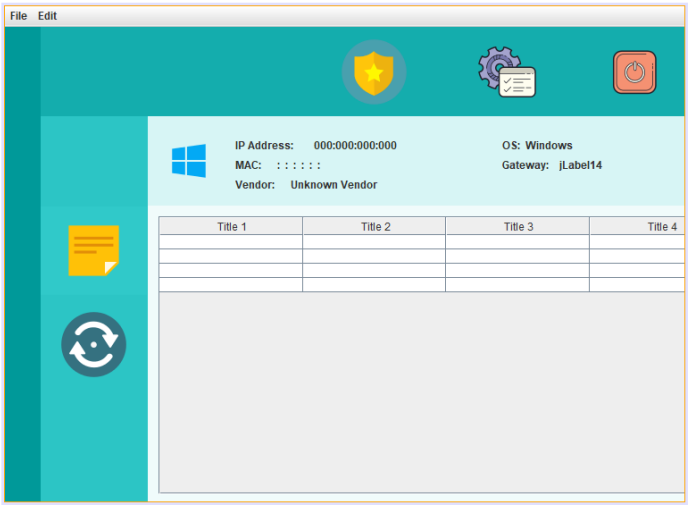
- Algunos de los componentes utilizados podían definirse como "anticuados" en términos de apariencia.
- La falta de flexibilidad de algunos componentes hacia inviable su utilización y



16 Wireframe PC (1er Maquetado)

las alternativas no ofrecían el resultado esperado.

El wireframe de la imagen anterior muestra que cada dispositivo encontrado en la red se encuentra detallado dentro de un panel y estos dentro de un panel con capacidad de "scroll" a medida que la cantidad de dispositivos requiere de mayor espacio para ser visualizada. En Swing el componente JPane no tiene capacidad de scroll y si bien puede ser combinado dentro de un JScrollPane la implementación no es del todo satisfactoria, más aún si tenemos en cuenta que la posibilidad de redimensionar la vista principal ha sido deshabilitada. Una de las variantes hubiese sido la de presentar los resultados dentro de una tabla; nuevamente el resultado visual no era el esperado.



17 Diseno GUI con JavaSwing (Descartado)

5.4 PRECONDICIONES PARA LA UTILIZACIÓN DE LA APLICACION

En Android el comando arp no ejecuta un proceso en particular, sino que muestra por pantalla el contenido del fichero /proc/net/arp de igual forma que si utilizásemos el comando "cat /proc/net/arp".

La información almacenada en tal fichero se crea y se "limpia" automáticamente atendiendo a una configuración del sistema que es inaccesible para un usuario sin privilegios de sistema.

Con esta situación adelante el software enfrenta los siguientes inconvenientes:

- El apartado Publico Objetivo (1.4) describe que la aplicación ha sido mayormente ideada para usuarios que, carentes de capacidad técnica, desean comprobar si la red que utilizan no está siendo afectada por un ataque de ARP Spoofing / Poisoning. El proceso de obtener privilegios de sistema en un sistema Android es conocido como "obtener root" o "rootear" y, dependiendo el modelo del móvil, presenta menor o mayor complejidad, pero siempre frente a un usuario medio/avanzado.

  Según publicado en https://www.kaspersky.com/blog/android-root-faq/17135/ el promedio mundial de dispositivos Android con capacidad root está cerca del 7%, De esta forma la cantidad de personas que podrán hacer uso de la aplicación de forma efectiva se ha reducido considerablemente.
- La ejecución del App por parte de un perfil sin derechos elevados resultaría en información desactualizada o errónea sobre los dispositivos conectados a la red, dificultando la detección al incrementar la posibilidad de falsos positivos y falsos negativos de una amenaza.

En paralelo con lo hasta aquí expuesto cabe destacar que en Windows habilitar o deshabilitar la conexión de red vía línea de comandos también requiere de un perfil con permisos elevados. El comando a utilizar seria netsh interface set interface "iface" enable / disable. Considerando la situación se ha decidido el no ofrecer la opción de realizar esta acción automáticamente en la versión PC.

5.5 CAMBIOS DE ACCESIBILIDAD A LA TABLA ARP EN ANDROID 10 (API 29)

Android 10 incluye restricciones relativas a la privacidad de datos y impedirá el acceso a información que puedan ser utilizados de forma maliciosa, ya sea para colección de datos o establecer un perfil (Fingerprinting)

Dentro de estos cambios se encuentra la imposibilidad de un App de acceder a información dentro del directorio /proc/net, siendo este donde la tabla arp se encuentra localizada.

Una alternativa que ha devuelto un resultado positivo es la de la utilización del paquete iproute2 (/proc/sys) para realizar la consulta. Este comando devuelve la misma información y puede, al momento, ser utilizado para obtener los datos necesarios.

6 PRUEBAS

6.1 INTRODUCCIÓN

Para la evaluación de este proyecto se ha hecho foco en la funcionalidad del sistema y dejar de lado la estructura interna del codigo. En resumen, pruebas de Caja Negra.
En particular se han realizado pruebas de casos de uso para cada una de las diferentes versiones del software.

6.2 EJEMPLO DE PRUEBAS DE CASO DE USO

CP01 -Windows

Paso	Caso de Uso	Resultado Esperado	Resultado actual	Éxito/Falla	Notas
1	CU01-INTALL	Aplicación instalada en dispositivo	Aplicación instalada	Éxito	
2	CU02-OPEN	Aplicación inicia en Systray	Error	Falla	
3	CU03-FOCUS	Despliega la vista principal		N /A	Falla en la iniciación
4	CU09-HIDE	Oculta la vista principal, aplicación continúa funcionando en Systray		N /A	Falla en la iniciación

6.3.1 PC

Las pruebas se han realizado tanto en dispositivos físicos (versión Windows) como en ambientes virtualizados (Ubuntu).
En primera instancia las pruebas realizadas para la versión PC han ayudado a encontrar errores "ajenos" al desarrollo del sistema. (ver CP01-Windows). Esto ha permitido retraer el proceso al estado anterior y avanzar rápidamente (ver 7.2 Problemas en el despliegue). Una vez superado el inconveniente los resultados han sido exitosos.

6.3.2 Android

Android Studio cuenta con un sistema de empaquetamiento integrado que facilita el proceso de empaquetado y el mismo no ha devuelto inconvenientes. Las pruebas se han realizado tanto en un dispositivo físico como en un ambiente virtualizado.
Si bien puede decirse que las pruebas han arrojado resultados exitosos es también justo mencionar que el tiempo de respuesta en la detección dispositivos conectados a la red es en ocasiones mayor al esperado.
Por otro lado, estas pruebas han permitido la detección excepciones causadas en el sistema como mencionado en el punto (4.5.3 OutOfMemoryErrorException).

7 DESPLIEGUE

7.1 DESPLIEGUE DE LA APLICACIÓN PARA PC

Como ha sido mencionado en el apartado 3.2.9 se ha hecho uso de Wix Tools en el desarrollo de este proyecto. Esta aplicación, una vez instalada en el SO es reconocida por el IDE y utilizada para generar ficheros de instalación para Windows (.msi o .exe) o paquetes para sistema Linux (.deb / rpm).

Para lograr el objetivo es necesario modificar el fichero build.xml de nuestro proyecto dentro del nodo raíz. Esta información forma parte de la documentación de Oracle JavaFX8 que de acceso público en internet.

```
<target name="-post-jfx-deploy">
    <fx:deploy width="${javafx.run.width}" height="${javafx.run.height}"
            nativeBundles="all"
            outdir="${basedir}/${dist.dir}" outfile="${application.title}">
      <fx:application name="${application.title}"
              mainClass="${javafx.main.class}"/>
      <fx:resources>
        <fx:fileset dir="${basedir}/${dist.dir}"
              includes="*.jar"/>
        <fx:fileset dir="${basedir}/${dist.dir}" />
      </fx:resources>
      <fx:info title="${application.title}"
           vendor="${application.vendor}"/>
    </fx:deploy>
  </target>
```

Luego de modificado el fichero build.xml y desde el IDE solo resta seleccionar la opción Build sobre nuestro proyecto.

7.2 PROBLEMAS EN EL DESPLIEGUE

Los primeros procesos de despliegue efectuados generaban, luego de instalado el software, un error cuando se intentaba ejecutar la aplicación.

Los mensajes de alerta recibidos eran:

- "Error invoking method"
- "Failed to launch jvm"

La clave para solucionar este inconveniente ha sido la mejora en el control de excepciones. Iniciando entonces la versión instalada vía línea de comandos nos encontramos con la excepción FileNotFoundException y NullPointerException.
A partir de entonces y con la ayuda de foros online el problema pudo resolverse. En resumen: Nuestra aplicación hace uso de la librería javax.mail, esta no es parte de las librerías estándar. Debido a esto es importante modificar parte de los datos para los ficheros .jar que componen esta librería se incluyan dentro del paquete (bundle) que el proceso de despliegue creará.
De esta forma incluimos includes="lib/*.jar" en el nodo <fx:fileset dir="${basedir}/${dist.dir}" /> siendo el resultado <fx:fileset dir="${basedir}/${dist.dir}" includes="lib/*.jar" />

8 CONCLUSIÓN

La intención de esta sección es la de expresar conclusiones finales y personales sobre la realización del proyecto, extraídas durante el proceso de desarrollo.

Si bien se han cumplido la gran parte de los objetivos y requisitos mencionadas a lo largo de los diferentes capítulos. En este apartado queda en evidencia que el limitar el uso de la aplicación Android a aquellas personas con un dispositivo "Rooteado" tiene una carga significativa en la efectividad del sistema.

Cabe destacar la importancia del método de desarrollo a utilizar y el impacto que una mala elección puede tener en el proyecto. Podría mencionarse la creación de la GUI como ejemplo.

Come se indicó inicialmente el proyecto sería creado utilizando una interfaz gráfica creada con Java Swing, problemas con la implementación de la misma en nuestro proyecto (de acuerdo a lo que se quería lograr) resultaron costosas en términos de tiempo. No solo debido a la necesidad de aprender a utilizar una tecnología diferente sino también a la adecuación del codigo ya creado para su correcto funcionamiento. Por otro esta misma necesidad me ha ayudado a incorporar nuevas habilidades.

Otro punto a tener en cuenta es la importancia de la documentación que se nos ofrece como desarrolladores, si bien la cantidad de documentos y sus ramificaciones pueden resultar abrumadoras es una gran herramienta para entender el funcionamiento de ciertos procesos.

Por último, destacar la importancia de la comunidad de desarrolladores, sus publicaciones y proyectos sin los cuales este mismo proyecto no podría haber sido posible. Estos aportes me han ayudado tanto a salir de "atascos" como a idear soluciones alternativas a diferentes problemas.

9 EVOLUCIÓN DEL SISTEMA A FUTURO

El objetivo principal de la aplicación a futuro será la de incluir dentro de sus usuarios potenciales a la totalidad de los usuarios de dispositivos Android (sin necesidad de un dispositivo con capacidad Root) y la de establecer una versión para iOS. Con esto en mente se espera también el de establecer una versión para macOS. Este punto es el de alcance inmediato ya que el software puede identificar fácilmente si estuviese corriendo sobre el mismo y ejecutar los comandos en consecuencia.

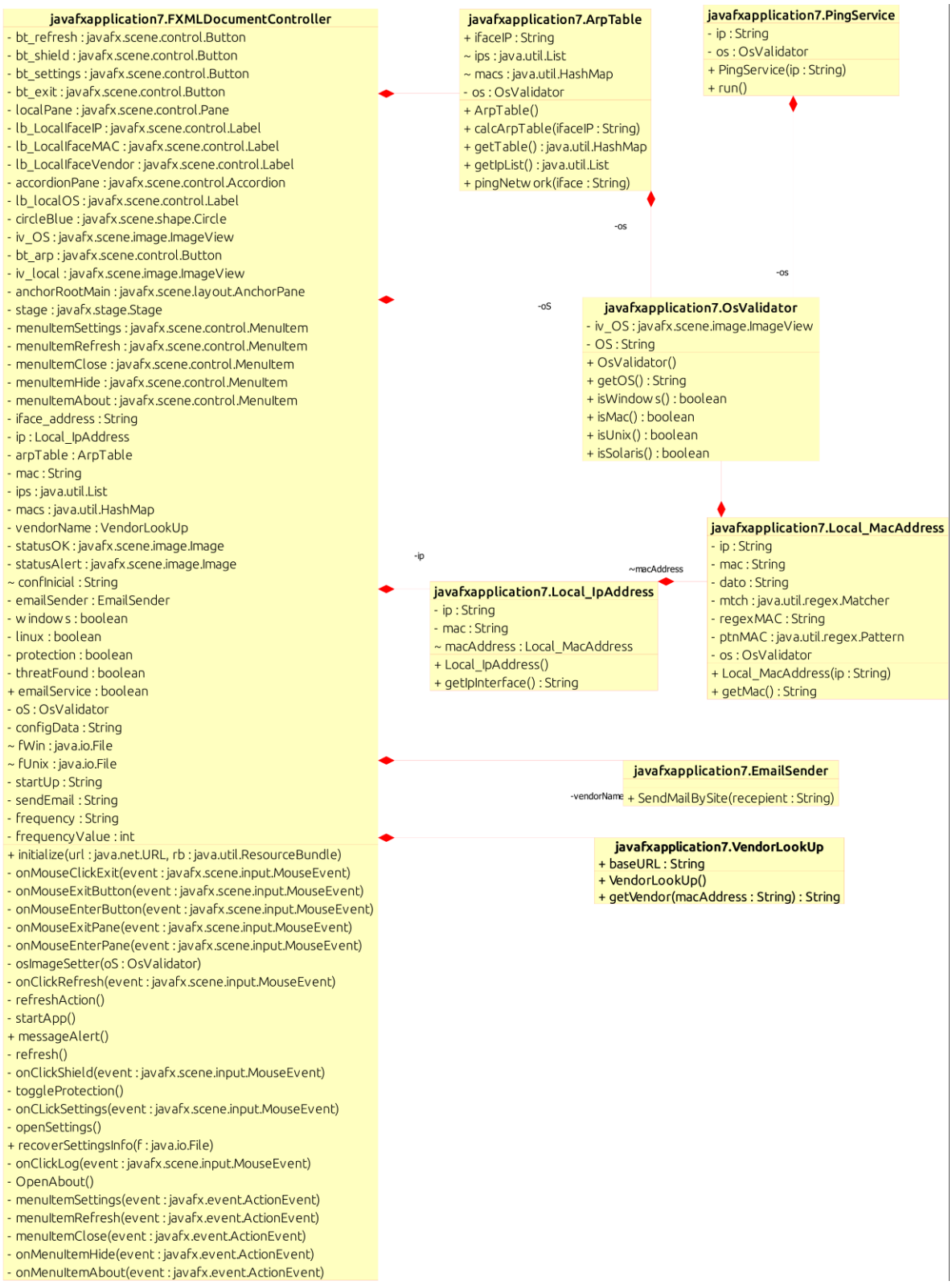
Particularmente este proyecto no recoge esa información ya que no ha sido posible el efectuar pruebas concretas bajo macOS debido a problemas de compatibilidad entre las herramientas utilizadas, el OS y el entorno de virtualización (VMware en este caso). => https://communities.vmware.com/thread/528682

Por otro lado, considero interesante la posibilidad de agregar un escáner de puertos a la aplicación y que esta permita identificar aquellos que se encuentran abiertos, los servicios operando en los mismos y un "listener" que permita, en tiempo real, alertar modificaciones en el estado de estos. En otras palabras, podría decirse que sería importante que el sistema mudase de una aplicación para detectar una intrusión vía arp para gradualmente convertirse en un toolkit de seguridad informática.

El desafío esta puesto entonces en incrementar mis conocimientos como desarrollador no solo con respecto a lenguajes de programación (Swift para iOS por ej.) sino también en cuando a redes, servicios, etc.

ANEXO I

DIAGRAMA DE CLASES PC (STARUML REVERSE ENGEENIERING)



javafxapplication7.FXMLLog

```
- bt_closeAbout : javafx.scene.control.Button
+ initialize(url : java.net.URL, rb : java.util.ResourceBundle)
- onClickClose(event : javafx.scene.input.MouseEvent)
```

javafxapplication7.JavaFXApplication7

```
- iconImageLoc : String
- notificationTimer : java.util.Timer
+ start(stage : javafx.stage.Stage)
- createContent() : javafx.scene.Node
- addAppToTray()
- showStage()
+ main(args : String[])
```

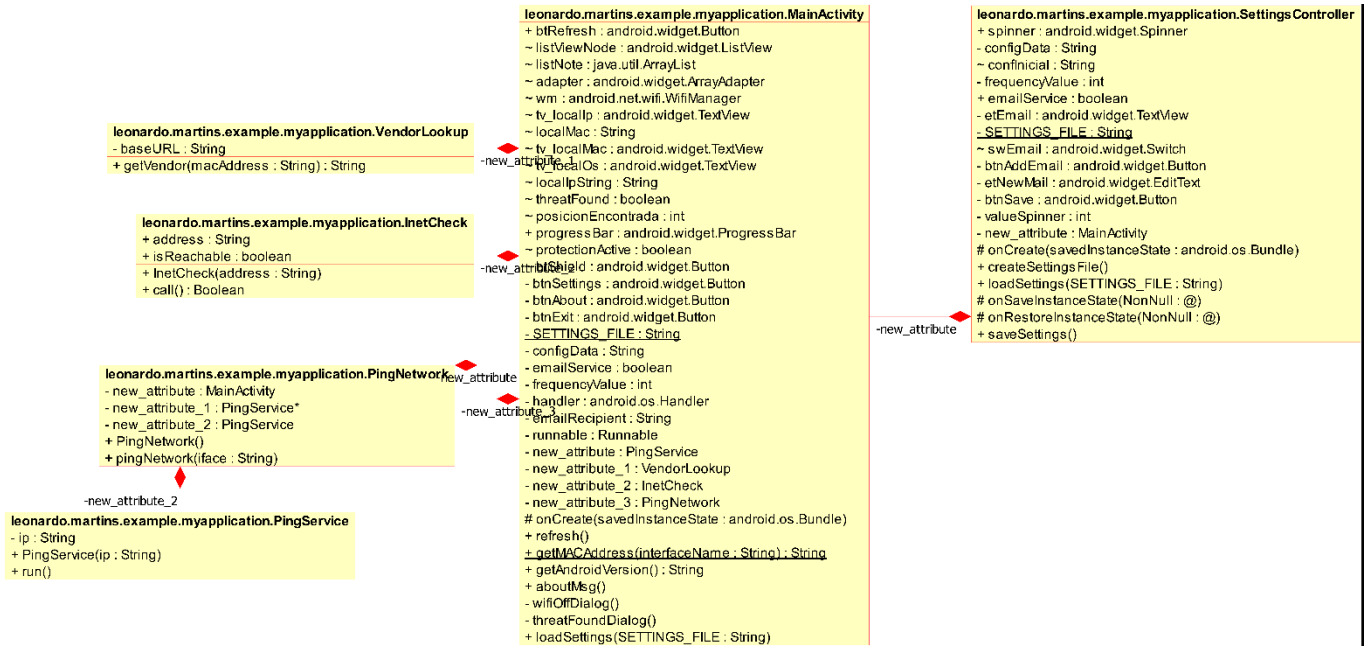
javafxapplication7.FXMLLogController

```
- iv_about : javafx.scene.image.ImageView
~ stage : javafx.stage.Stage
- bt_closeABout : javafx.scene.control.Button
+ initialize(url : java.net.URL, rb : java.util.ResourceBundle)
- bt_onCLoseAbout(event : javafx.scene.input.MouseEvent)
```

javafxapplication7.FXMLSettingsController

```
- cb_sendEmail : javafx.scene.control.CheckBox
- bt_SettingAccept : javafx.scene.control.Button
- bt_settingCancel : javafx.scene.control.Button
- spinner_timeMIn : javafx.scene.control.Spinner
- et_newEmail : javafx.scene.control.TextField
- text_email : javafx.scene.text.Text
- bt_addEmail : javafx.scene.control.Button
- email : String
- stage : javafx.stage.Stage
- configData : String
- defaultSpinner : int
~ fWin : java.io.File
~ fUnix : java.io.File
+ initialize(url : java.net.URL, rb : java.util.ResourceBundle)
+ recoverSettingsInfo(f : java.io.File)
- onClickAcceptSettings(event : javafx.scene.input.MouseEvent)
- onClickCancelSettings(event : javafx.scene.input.MouseEvent)
- onCLickAddEmail(event : javafx.scene.input.MouseEvent)
- writeNewSettings()
```

DIAGRAMA DE CLASES ANDROID (STARUML REVERSE ENGEENIERING)



ANEXO II

MANUAL DEL USUARIO

1 REQUISITOS MINIMOS

Sistema Operativo >= Windows 7 / Linux OS (Cualquier distribución)

net-tools (Linux)

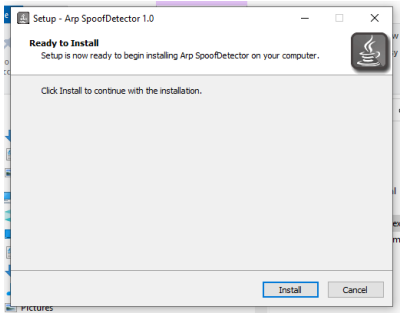
Java Runtime Enviroment >= JRE8

https://www.oracle.com/java/technologies/javase-jre8-downloads.html

2 PROCESO DE INSTALACION

2.1 WINDOWS

El proceso de instalación de la aplicación ha sido simplificado para facilitar la misma a usuarios con capacidades técnicas limitadas. La aplicación esta contenida en un fichero (Arp SpoofDetector-1.0.msi o Arp SpoofDetector.exe) . Solo es necesario iniciar el mismo y seleccionar la opción “Install”. Una vez acabado el proceso la aplicación iniciara automáticamente. (Visible en la **barra de sistemas**). También podremos encontrar el acceso directo en nuestro Menú de Inicio / Windows con el que iniciar la aplicación en caso de que el sistema se encuentre detenido.



2.2 LINUX

manual recoge únicamente el proceso de instalación en distribuciones basadas en Debian.

Para el correcto funcionamiento del software es necesario instalar el paquete net-tools. Este puede adquirirse de con la ejecución de las siguientes sentencias en el terminal.

sudo apt-get update.

sudo apt-get install net-tools

Para la instalación del software ARP_SD en concreto solo basta ejecutar el siguiente comando:

<Directorio donde se encuentra el paquete>\sudo dpkg -i arp-spoofdetector-1.0.deb

El retorno de una operación exitosa se vería similar a:

Selecting previously unselected package arp-spoofdetector.

(Reading database ... 171453 files and directories currently installed.)

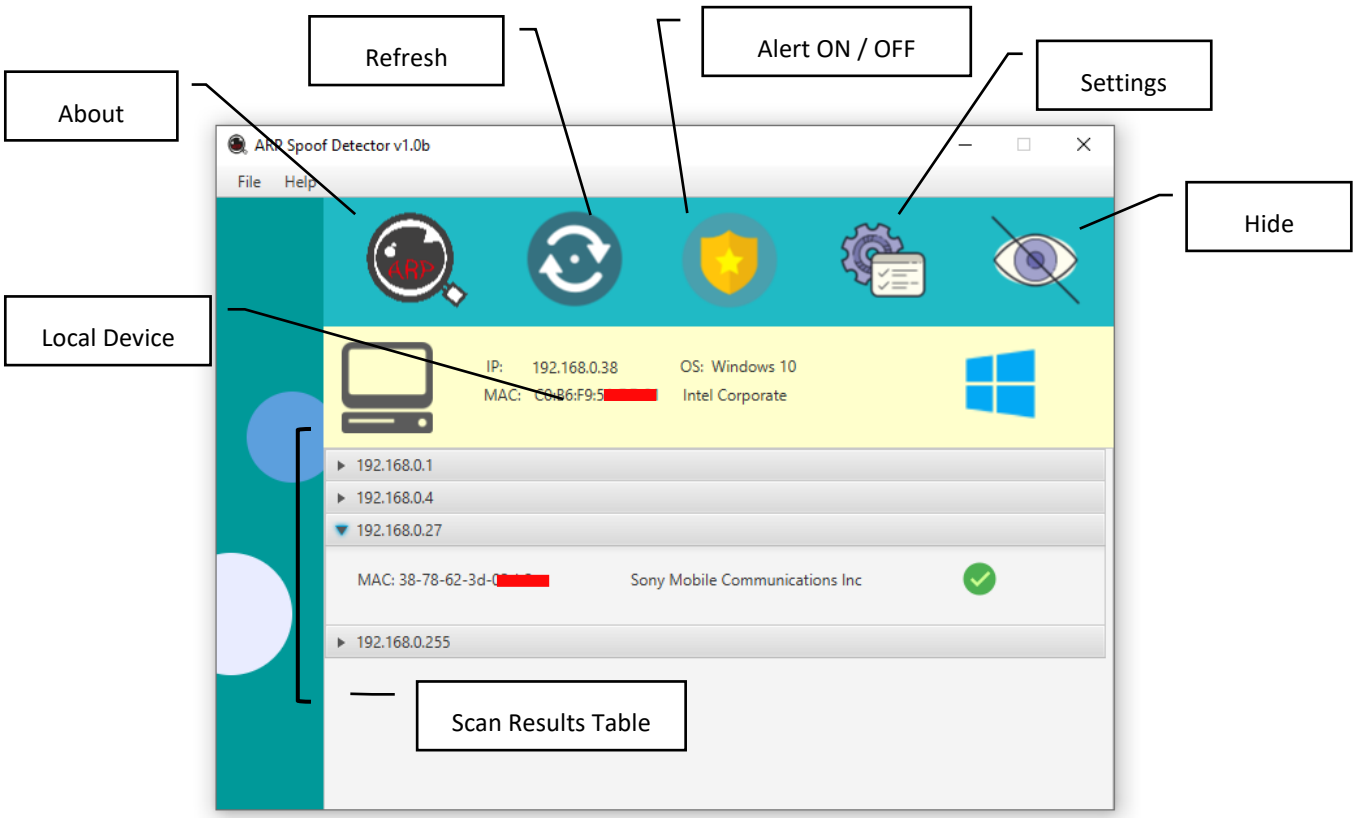
Preparing to unpack arp-spoofdetector-1.0.deb ...

Unpacking arp-spoofdetector (1.0) ...

Setting up arp-spoofdetector (1.0) ...

Adding shortcut to the menu

3 VISTA PRINCIPAL DE LA APLICACIÓN



4 USO DE LA APLICACIÓN

Una vez iniciada la aplicación (Inicio en Systray para Windows) la misma se encargará de ejecutar un análisis a de la red. En caso de que el dispositivo no se encuentre conectado a ninguna red el sistema no devolverá resultado alguno en la tabla de resultados.

El proceso de escaneo es automático y su frecuencia puede ser alterada.

En la parte superior podremos encontrar cinco botones claramente diferenciables. Estos son (izq. A dcha.):

- Sobre / About: Despliega una ventana emergente con datos de la aplicación.

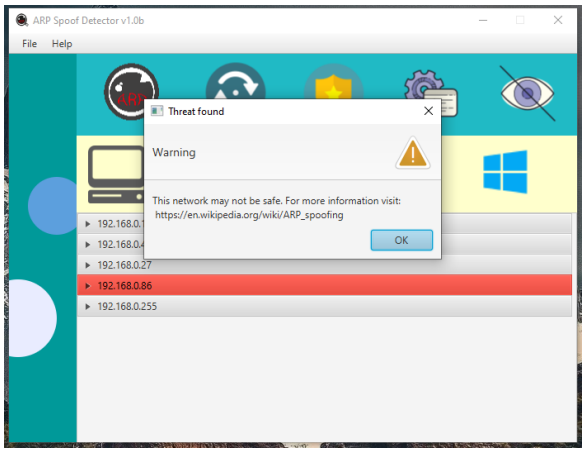
- Refrescar / Refresh: Inicia de manera explícita un escaneo sobre la red, actualizando los valores de la tabla de resultado,
- Alert On / OFF : Sirva para activar o desactivar el proceso de alerta de la aplicación; la aplicación seguirá su curso normal pero no desplegara un mensaje de alerta en caso de detectar una posible amenaza. La notificación podría aun establecerse vía email si tal opción hubiese sido seleccionada.
- Opciones / Settings: Despliega la vista Opciones de la aplicación. Podemos activar el servicio de notificación vía emails e ingresar la casilla donde queremos recibir tal notificación. Además, podemos modificar la frecuencia con la cual el sistema realizara un rastreo sobre la red.
- Ocultar / Hide : Hace invisible la vista principal de la aplicación pero sin cerrar la aplicación. De esta forma se reduce el consumo de recursos del sistema. En Windows podemos hacer visible la aplicación seleccionando la opción “Show ArpSD” del menú contextual que se despliega al hacer “click alternativo” sobre el icono de la aplicación.

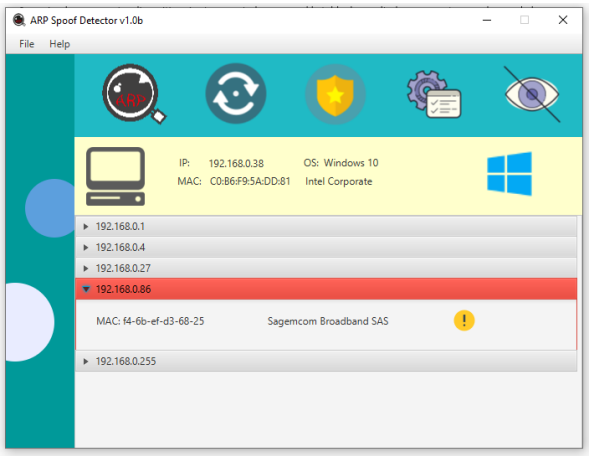
Suponiendo que nuestro dispositivo si está conectado a una red la tabla de resultados nos mostrara cada uno de los dispositivos que se encuentran en la misma. Cada dispositivo detectado será representado por una vinera desplegable titulada con la dirección del dispositivo dentro de la red. Si desplegásemos esta viñeta podremos observar el identificador único del dispositivo y el fabricante del mismo. Sobre la derecha veremos además una señal visual de que el MAC id de ese dispositivo es, al momento del rastreo, un valor único.

5 DETECCION DE POSIBLES ATAQUES

En caso de que la aplicación detecte un posible ataque se desplegara un mensaje de alerta informándonos de esta situación. Además, la viñeta donde se ha encontrado tal duplicación será señalizada alterando el color de la misma.

El mensaje de alerta contiene además un URL donde el usuario puede aprender mas sobre los ataques MITM – ARP en





caso de no estar totalmente seguro de lo que esto implicase.

6 DESISTALACION DE LA APLICACIÓN

El fichero de instalación se encargará de crear un acceso directo al proceso de desinstalación de la aplicación. El mismo se encuentra en el apartado Agregar o Quitar programas.

ANEXO III

BIBLIOGRAFÍA / WEBGRAFÍA

Java

Java APIs,Extensions and Libraries by Kishori Sharan (2da Edición - APRESS)

https://docs.oracle.com/en/java/

https://www.callicoder.com/

https://www.tutorialspoint.com/javafx/index.html

http://www.java2s.com/

Android

Android 9 Development Cookbook by Rick Boyer (3ra Edición – PACKT)

https://developer.android.com/

http://android-er.blogspot.com/

https://guides.codepath.com/android

Miscelánea

https://stackoverflow.com/

https://docs.gluonhq.com/scenebuilder/

https://maclookup.app/api_doc

https://docs.oracle.com/javafx/2/deployment/troubleshooting.htm

https://wixtoolset.org/documentation/

https://communities.vmware.com/thread/528682

https://gist.github.com/jewelsea/e231e89e8d36ef4e5d8a