Anexo de documentación

Como premisa para este proyecto, se tomó el problema de los analistas de seguridad web y la primera toma de contacto con un equipo a analizar. Cuando se encarga un trabajo de este tipo, se debe realizar una primera toma de contacto con el dispositivo que conlleva realizar un breve análisis para saber cuál es el estado, a grandes rasgos, del objetivo. Para este tipo de análisis general se utilizan herramientas como *OpenVAS,* que a menudo resultan pesadas de ejecutar, puesto que se necesitan equipos con suficiente capacidad para ejecutar la mayoría de las pruebas, o en el caso de la utilización de herramientas de pago, costosas.

Lightpen, trata de solucionar ambos problemas otorgando al pentester, una manera fácil y rápida de realizar esta primera toma de contacto. Con su propio dispositivo móvil y desde cualquier lugar con una conexión a internet, podrá realizar una suite de pruebas básicas y no intrusivas para tener una idea preliminar del equipo. La idea original del proyecto prioriza pruebas simples y livianas, puesto que hay que ser conscientes de la limitación de los dispositivos móviles. Esto no quita utilidad a las mismas puesto que, por ejemplo, un análisis de las cabeceras de una página web resulta de mucha ayuda aunque no suponga una gran carga de trabajo.

La informática, y en especial la seguridad, crece día tras día con la aportación de un gran número de personas en distintas partes del mundo, por lo que, si negásemos a la aplicación la capacidad de nutrirse de ello, estaríamos cometiendo un gran error. Debido a esto, una parte importante del proyecto es la modularización de la aplicación. Lightpen permite el diseño de pruebas en forma de plugin de forma que la inclusión de nuevos análisis sea lo más fácil posible para los desarrolladores. De esta manera, la aplicación analizará automáticamente los plugins que cumplan ciertas condiciones, incluyéndolos así automáticamente en la suite de pruebas listas para ejecución.

Otra parte importante de este tipo de aplicaciones es la presentación de la información al cliente. Con el principio base que promueve iniciativas como Ionic, es decir, la utilización de tecnologías web para la creación de aplicaciones multiplataforma, Lightpen aprovecha HTML para mostrar los resultados de sus pruebas, de tal manera que se facilite así la tarea de compartir o almacenar los resultados de las mismas.

Manuales del Sistema

1.1 Manual de Instalación

Lightpen es una aplicación Android y como tal requiere poca configuración previa para su instalación, no obstante, explicaremos todos los pasos necesarios para su instalación incluyendo capturas para facilitar su instalación.

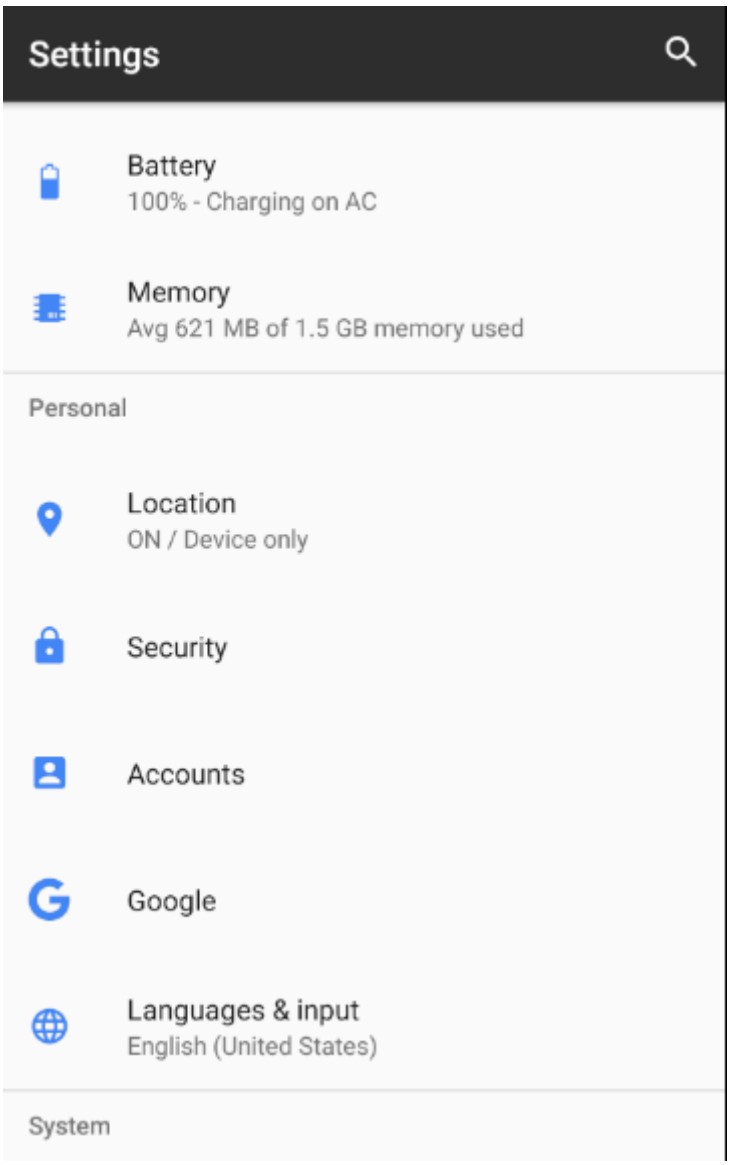

Ilustración 1. Pantalla de configuración de Android

En primer lugar, deberemos activar en nuestro dispositivo la característica que nos permite instalar el software de orígenes desconocidos. Para ello iremos al menú del terminal, pestaña **Seguridad (Security)** y allí activaremos la característica **Orígenes desconocidos (Unknown Sources)**, esto nos permitirá instalar la aplicación ya que no es una fuente reconocida por la tienda de aplicaciones.

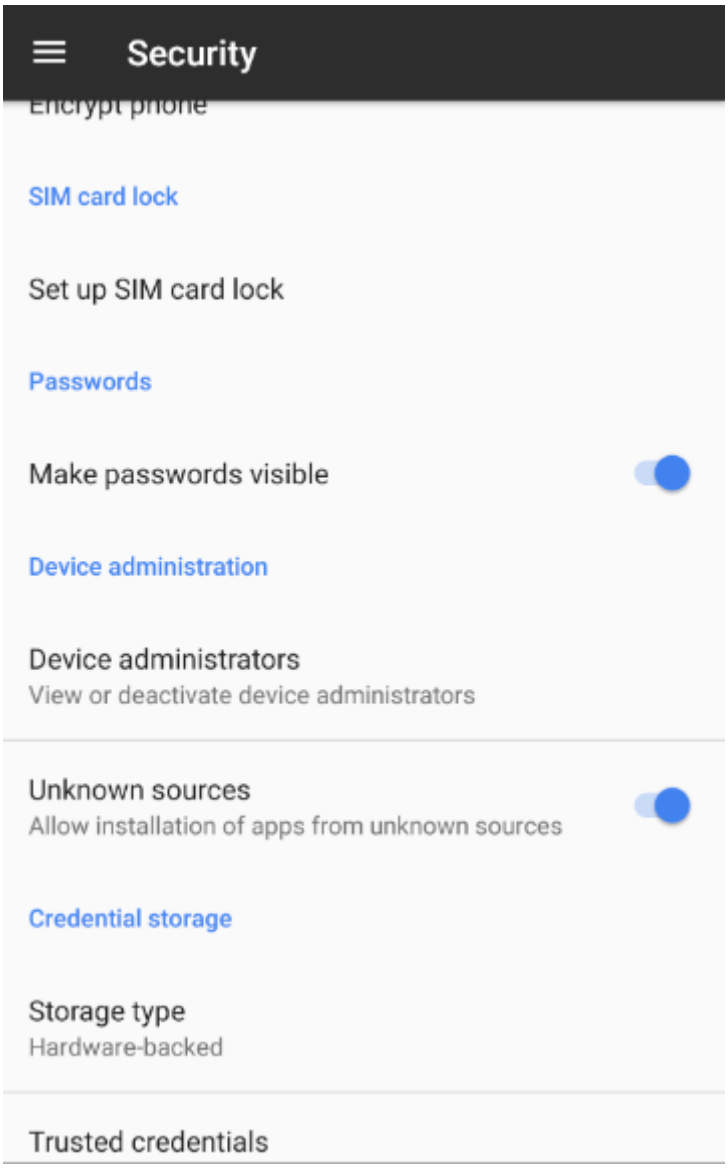

Ilustración 2. Pantalla de configuración de parámetros de Seguridad

Una vez activada esta característica ya estamos listos para copiar nuestra aplicación al dispositivo. Una vez descargado el fichero *Lightpen.apk* lo lanzaremos desde el gestor de carpetas de nuestro dispositivo y se abrirá una ventana informándonos de los permisos que esta aplicación requiere. En las últimas versiones de Android, los permisos de las aplicaciones se irán consultando en el momento del uso, no tenemos más que aceptarla para poder utilizar la aplicación.

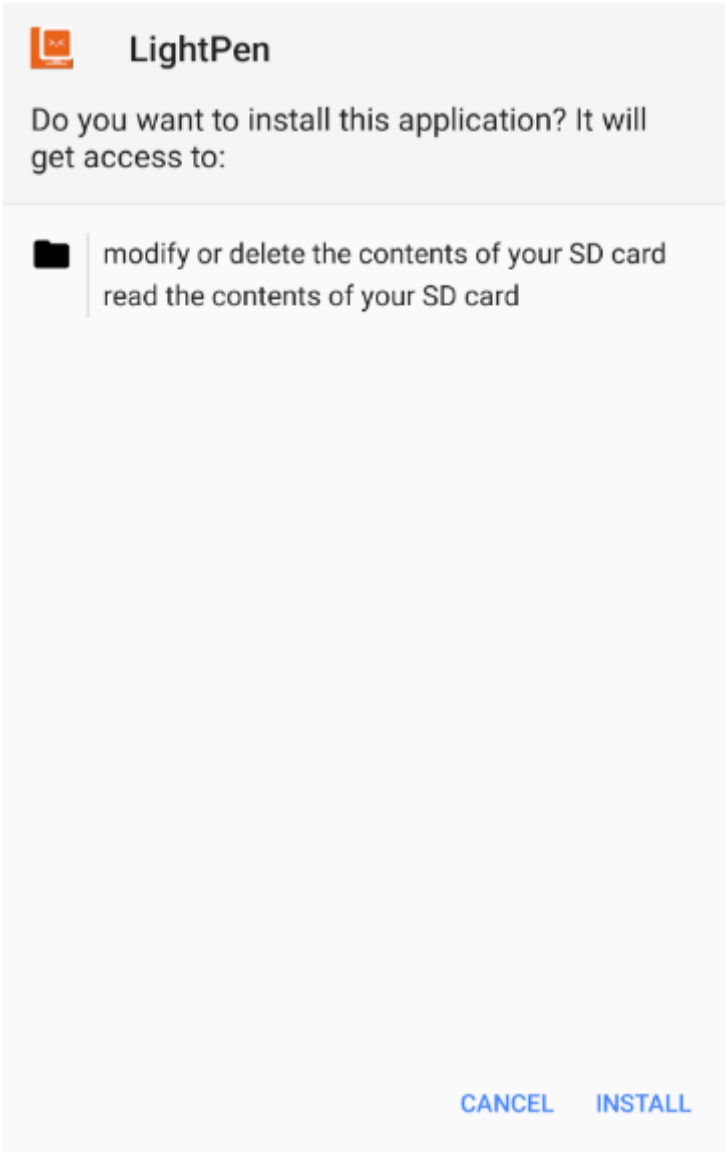

Ilustración 3. Pantalla de instalación de Lightpen

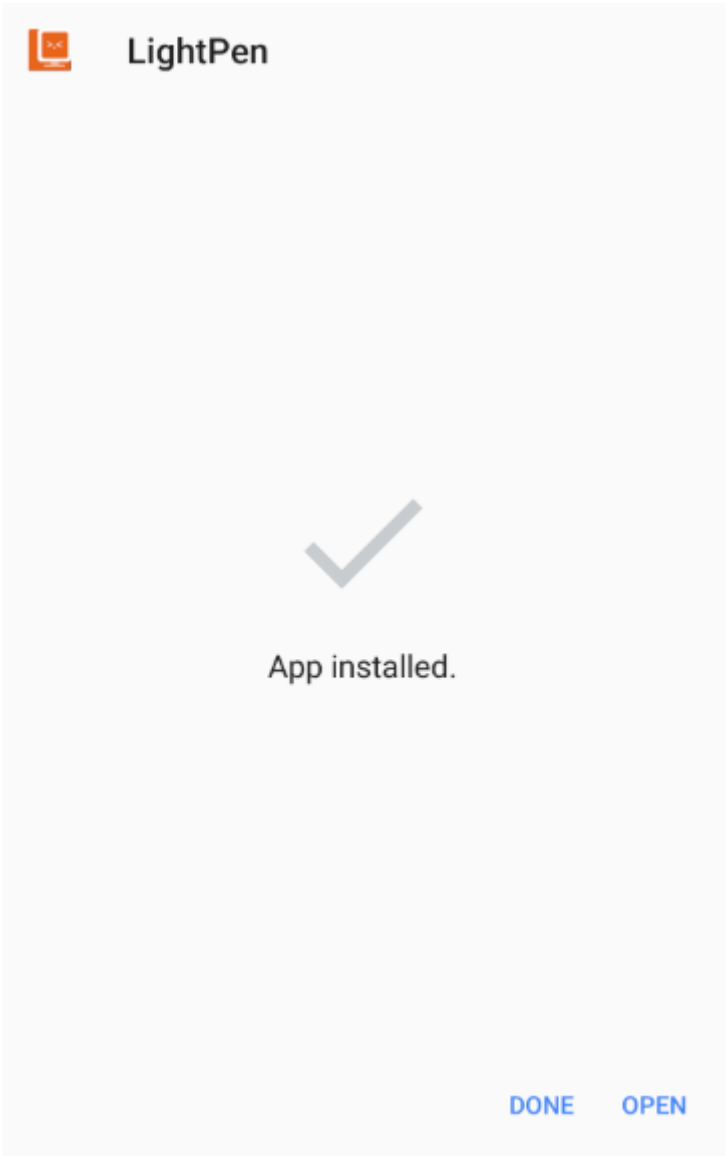

Ilustración 4. Pantalla de finalización de instalación

Manual de Usuario

Al ejecutar por primera vez Lightpen en nuestro dispositivo se mostrará la pantalla que definiremos como menú principal. En ella podremos ver las tres diferentes opciones que corresponderán con las tres posibles pantallas accesibles desde el menú, que en orden vertical son:

- Pantalla de análisis
- Pantalla de historial
- Pantalla de configuración

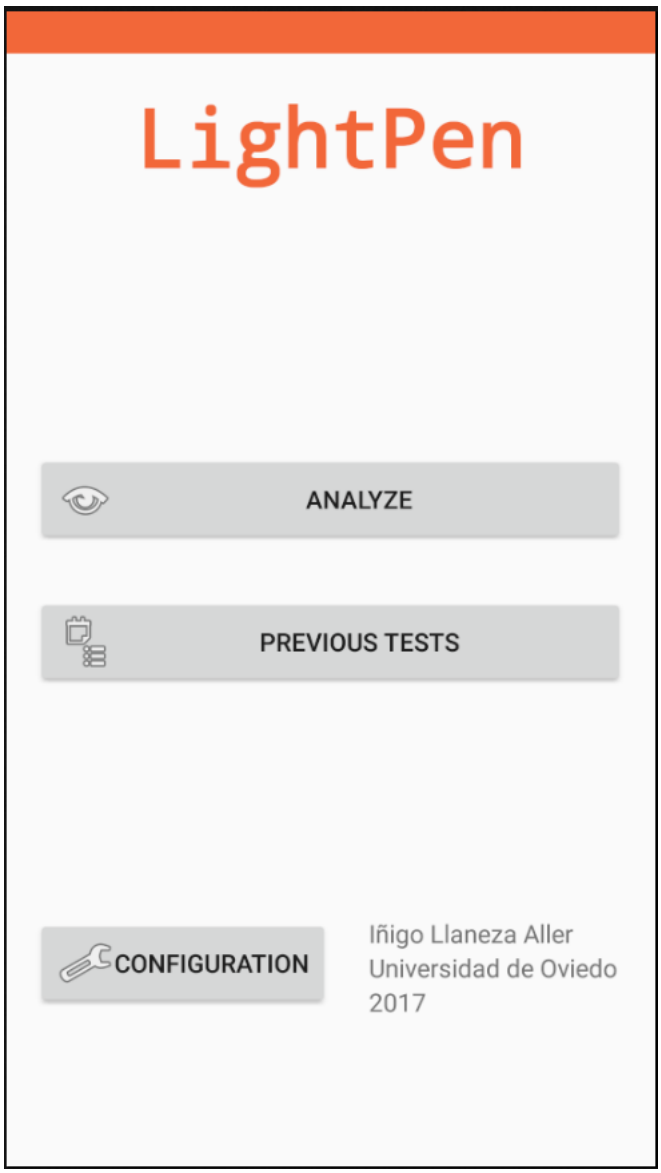

Ilustración 5. Pantalla del menú Principal

Desde la primera opción, podremos navegar a la pantalla del análisis, que consta de los siguientes elementos:

- Botón de vuelta al menú principal

- Botón de validación de URL
- Botón de comienzo del análisis
- Lista de plugins a utilizar, estos se indicarán mediante el uso de los checkbox

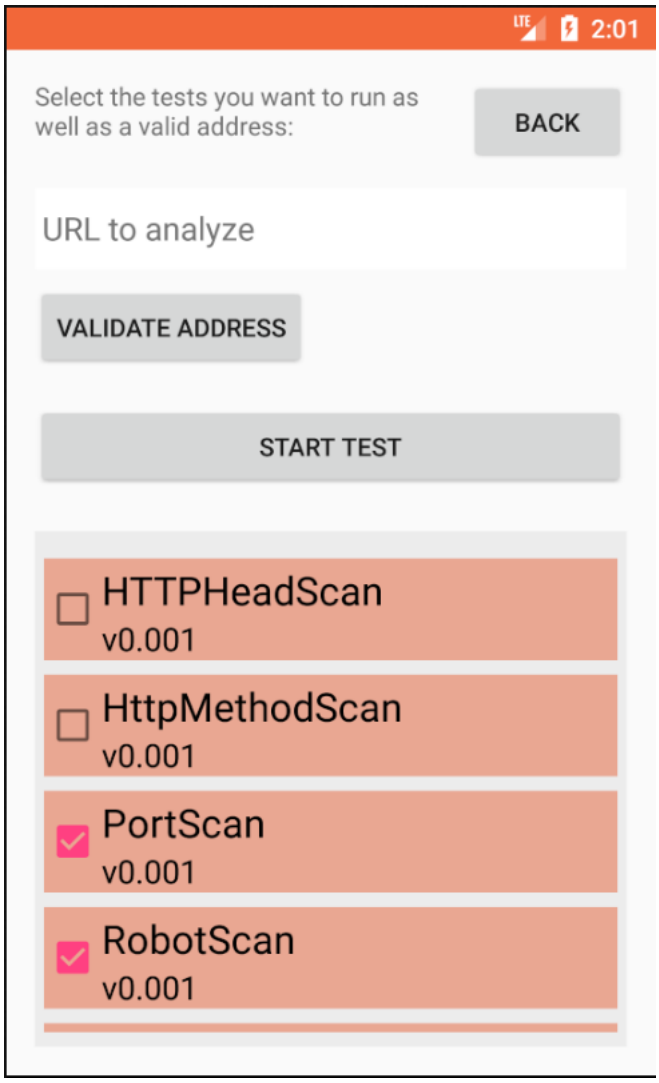

Ilustración 6. Pantalla de Análisis

Una vez comenzado el análisis, podremos ver la pantalla de carga, donde un gif de carga nos permitirá saber que la ejecución sigue en curso y una barra de progreso nos informará del porcentaje finalizado de la prueba.

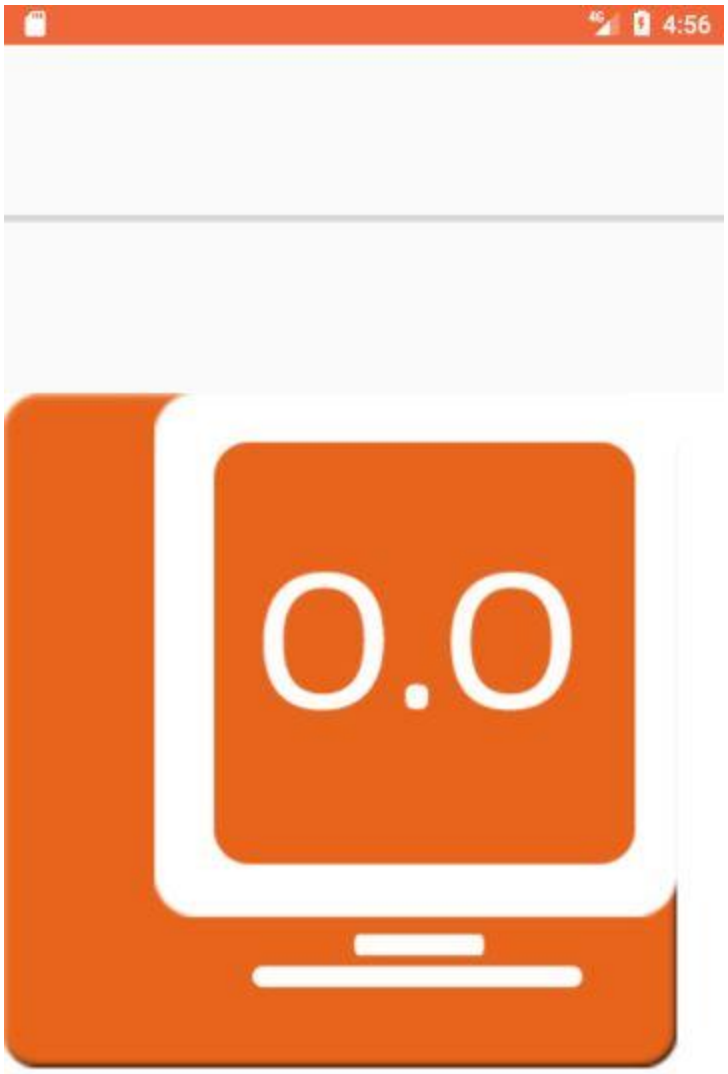

Ilustración 7. Pantalla de carga

Tras la finalización de las pruebas ya podremos ver el resultado de las pruebas y mediante el menú superior escogeremos el destino de la misma. Podremos compartirla mediante la interfaz de Android y guardarla o eliminarla.

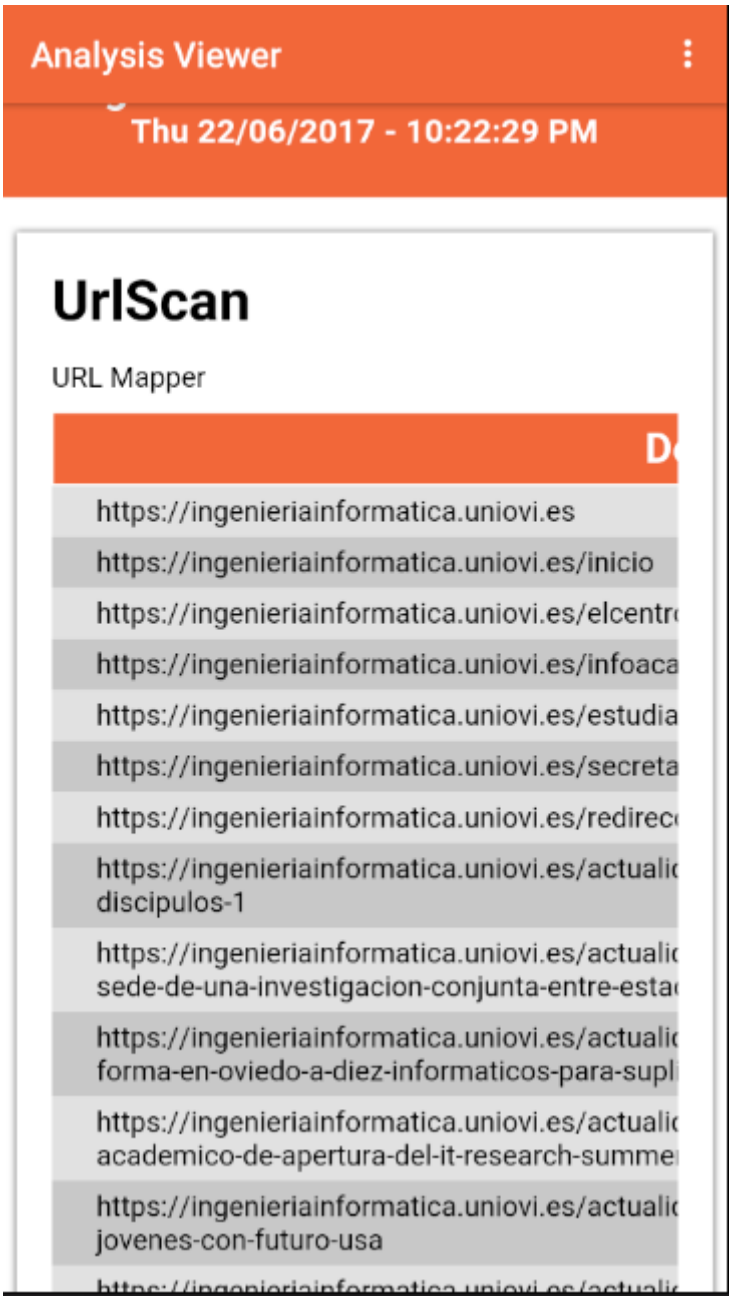

Ilustración 8. Pantalla de visualización de pruebas

Otra opción hubiese sido que el usuario desease revisar una prueba anterior, esto se realizará desde el segundo botón de la pantalla principal, el botón del historial. Esta navegación desembocará en la siguiente ventana:

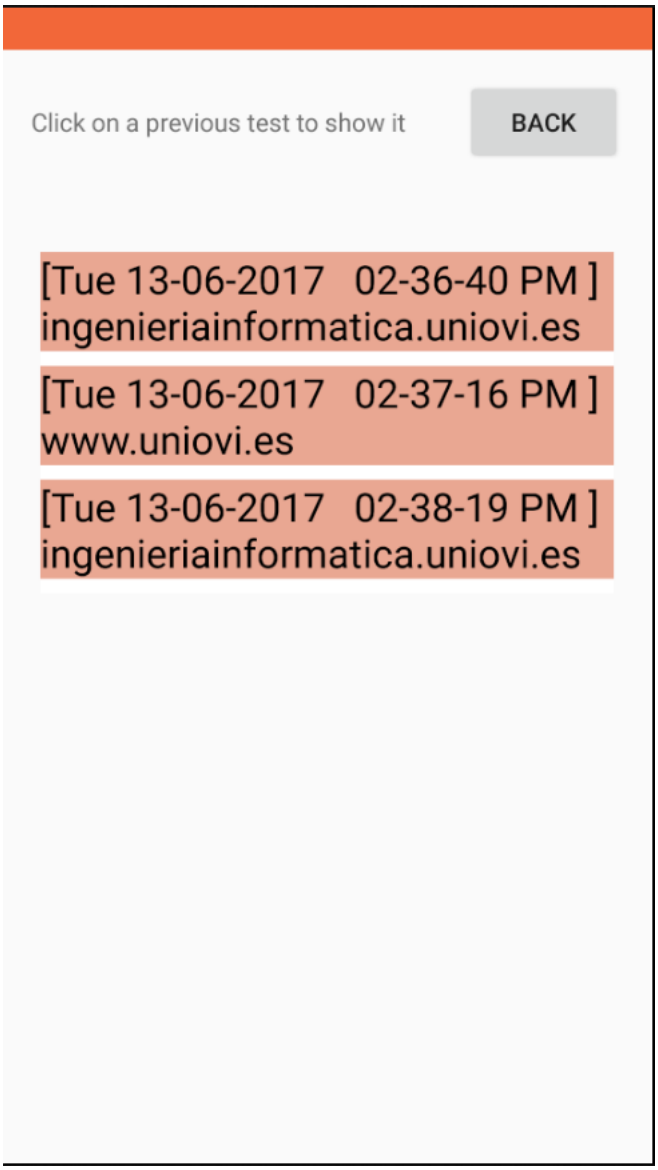

Ilustración 9. Pantalla de Historial

Esta nos permitirá seleccionar una prueba para revisarla, abriendo un visor como el anterior pero también con la opción de menú para eliminar la prueba.

1.2 Manual del Programador

Una característica que forma parte del corazón de este proyecto, es el deseo de que otros programadores pudieran completar la aplicación creando más pruebas y desembocando en que la aplicación pueda mejorar a límites ahora desconocidos. Para esto se diseñó originalmente con el deseo de que esta labor no fuera complicada y otorgando al futuro desarrollador todas las facilidades posibles. Con esta intención, se creó el sistema de plugins, el cual tiene como objetivo aumentar las funcionalidades sin afectar al núcleo de la aplicación Android.

Antes de comenzar a explicar cómo se podría incluir nuevas funcionalidades debo recordar que es **obligatorio** la desactivación de la característica de *Instant Run* de Android Studio, esto puede hacerse desde el menú de configuración del IDE, en la pestaña *Build, Execution, Deployment* bajo la opción *Instant Run*, tal y como se muestra a continuación.

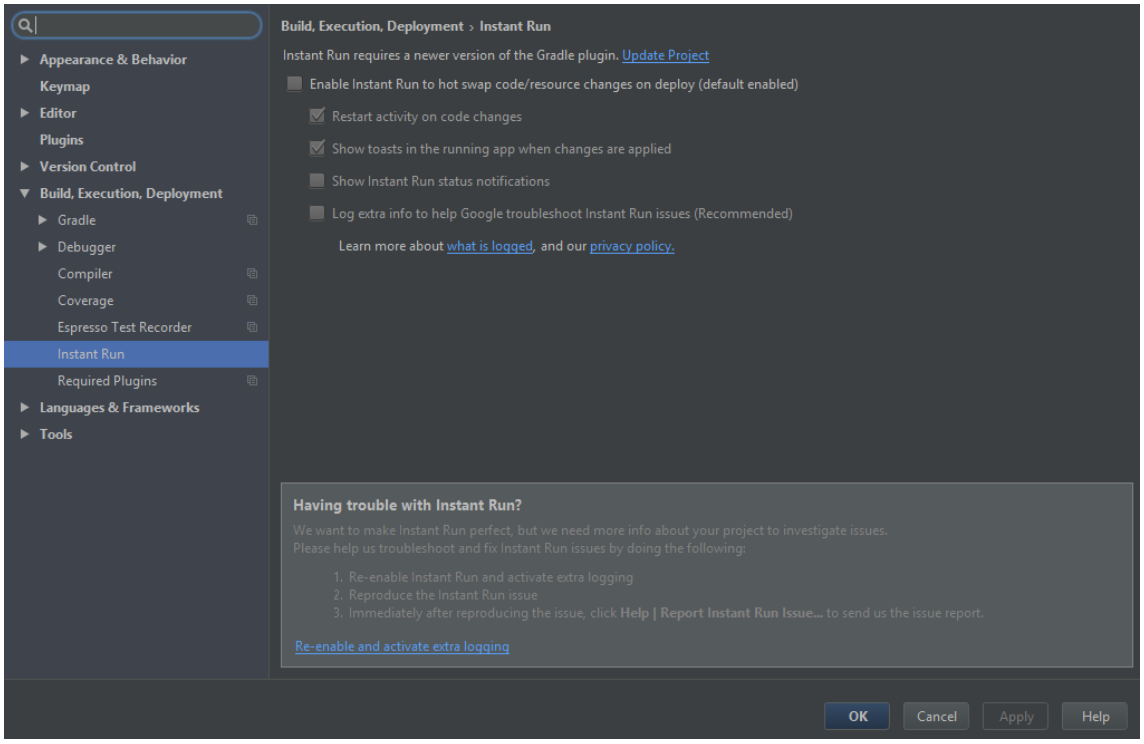

Ilustración 10. Menú de opciones de la aplicación Android Studio

Si un desarrollador externo quisiese añadir una funcionalidad a la aplicación, este deberá seguir los siguientes pasos:

1. Creación del plugin siguiendo el estándar de nombre implementado, *plg_NombrePlugin.java*, este nombre no es obligatorio, puesto que este prefijo, así como la ruta de plugins está definido en los ficheros de configuración, más concretamente en el fichero *config.properties* dentro del paquete *raw*. Actualmente si requiere que el sistema tenga esta extensión, así que lo tomaremos como obligatorio.

Lo anterior fue comentado para que el desarrollador sepa que podría cambiarlo a voluntad con un mínimo cambio.

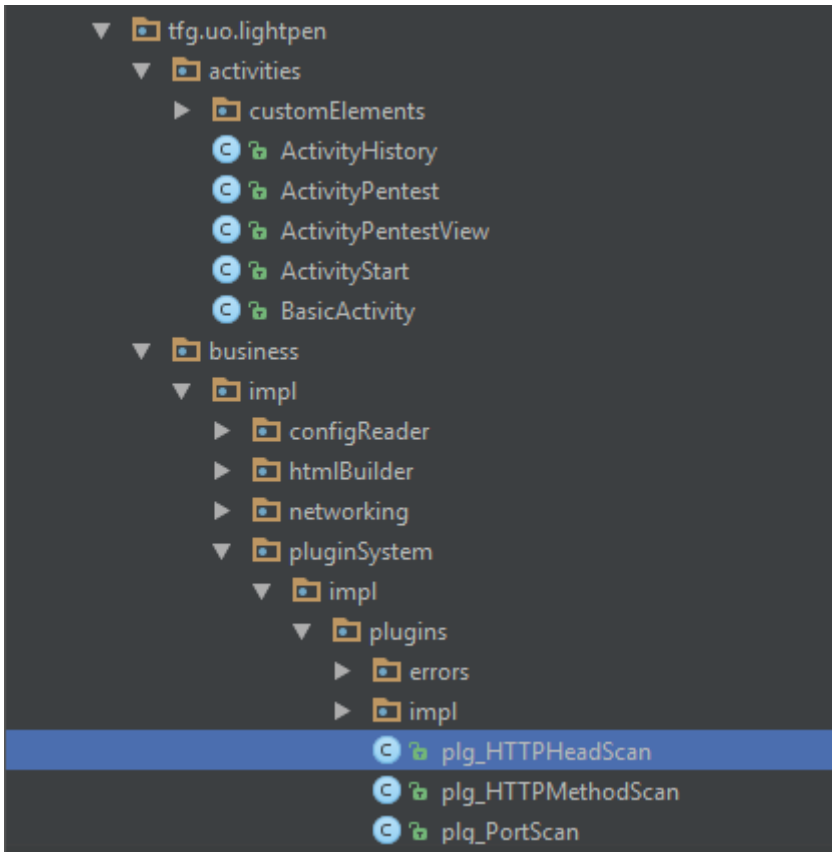

Ilustración 11. Estructura y localización de los plugins

2. El plugin, deberá implementar la interfaz *Plugin.java*, la cual permite que el plugin tenga unas nociones básicas sobre él. La explicación en concreto de la clase podemos encontrarla en el apartado **¡Error! No se encuentra el origen de la referencia.** con mayor profundidad. Se recomienda al desarrollador el uso del objeto ContextData tanto para la internacionalización de los resultados como para posibles accesos a ficheros.

Con estos pasos, el sistema ya reconocería automáticamente el plugin, pero otras configuraciones son posibles para el desarrollador, de tal manera que pueda aumentar su personalización tanto como en el apartado de resultados como en el de operaciones de red.

Para el apartado de los resultados, se creó un paquete *errors*, donde se han implementado errores para los distintos tipos de análisis. En caso de que el desarrollador lo estimase necesario podría implementar el suyo siempre y cuando su clase extienda de esa clase Error.

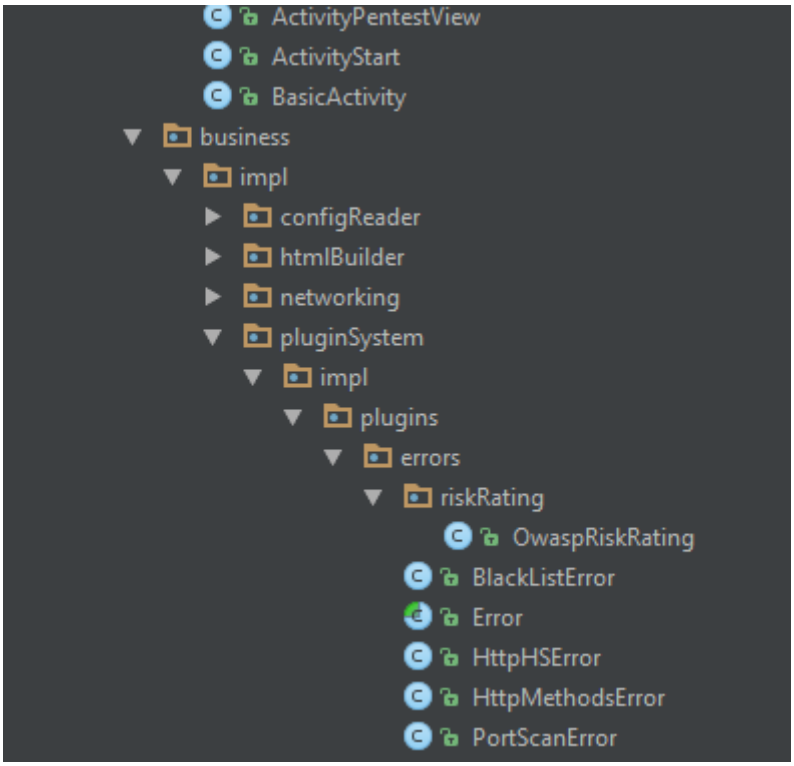

Ilustración 12. Estructura y localización de los distintos errores

Respecto al apartado de conexión de red, se ha implementado una suite de herramientas de red que puede ser ampliada en cualquier momento. Si el desarrollador lo ve adecuado, debido ya sea a que no existe ninguna que se adecúe a su gusto o cumpla sus necesidades puede o bien utilizar la base ya creada o crear sus propias herramientas. La idea original es ir fortaleciendo estas herramientas de conexión, de tal manera, que llegará el momento en el que no haga falta más y los desarrolladores solo tengan que escoger de esa lista.

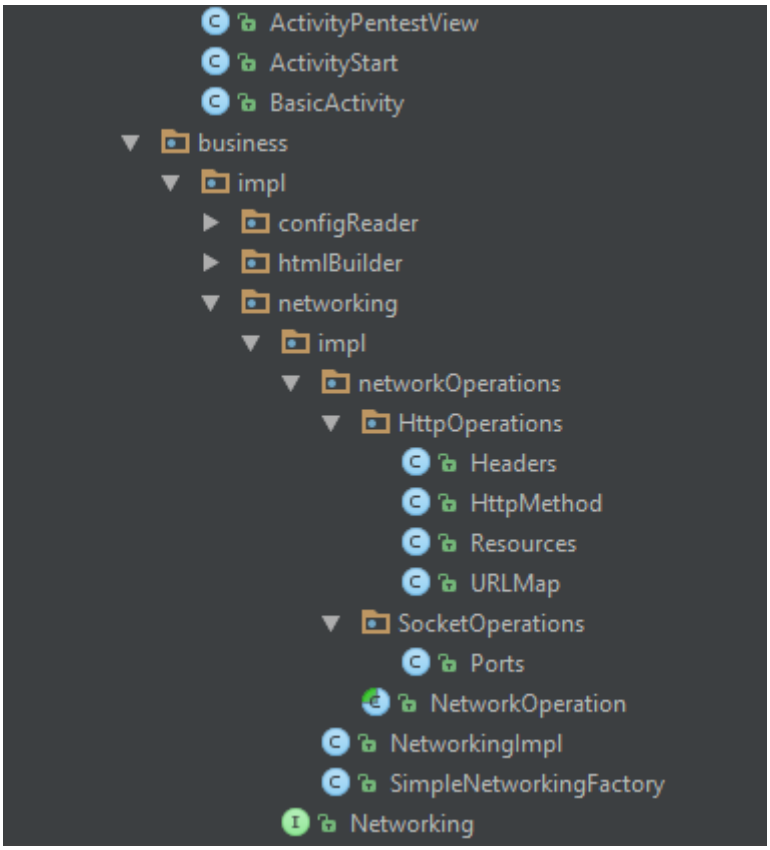

Ilustración 13. Estructura y localización de las operaciones de red