Ciberseguridad en dispositivos móviles

Práctica 4

Luis López Cuerva
Pablo Alcarria Lozano

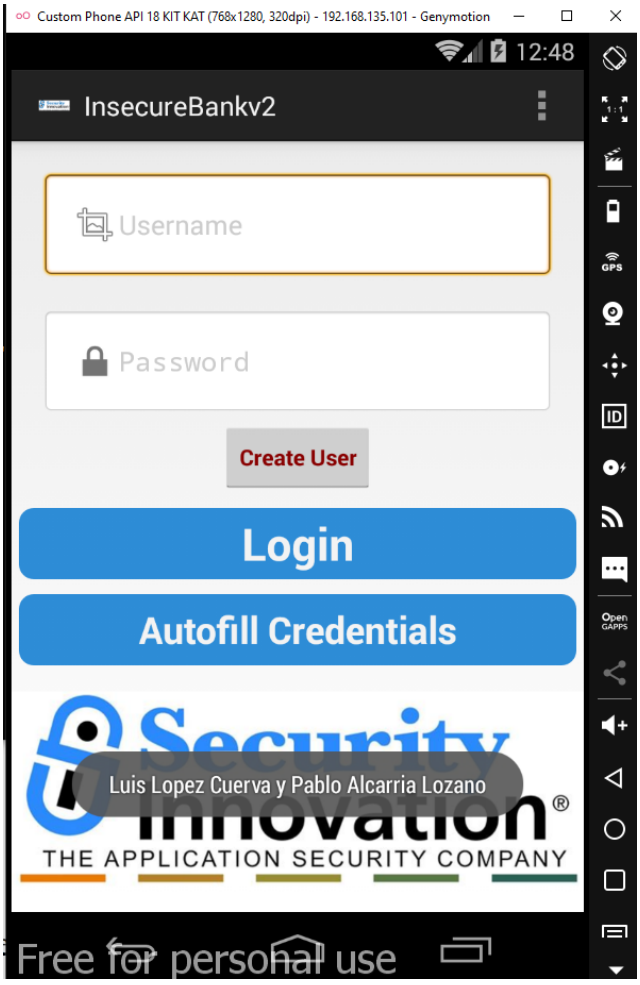
Luis López Cuerva
Pablo Alcarria Lozano

Introducción

En esta práctica se trabaja el análisis dinámico de la aplicación Insecure Bank v2, con el objetivo de identificar las vulnerabilidades o debilidades que encontramos en la aplicación mientras se encuentra en ejecución. Todas las pruebas realizadas se realizan mediante las herramientas jdb y Frida. Durante las tareas planteadas, se explicarán los problemas identificados para facilitar una posterior solución.

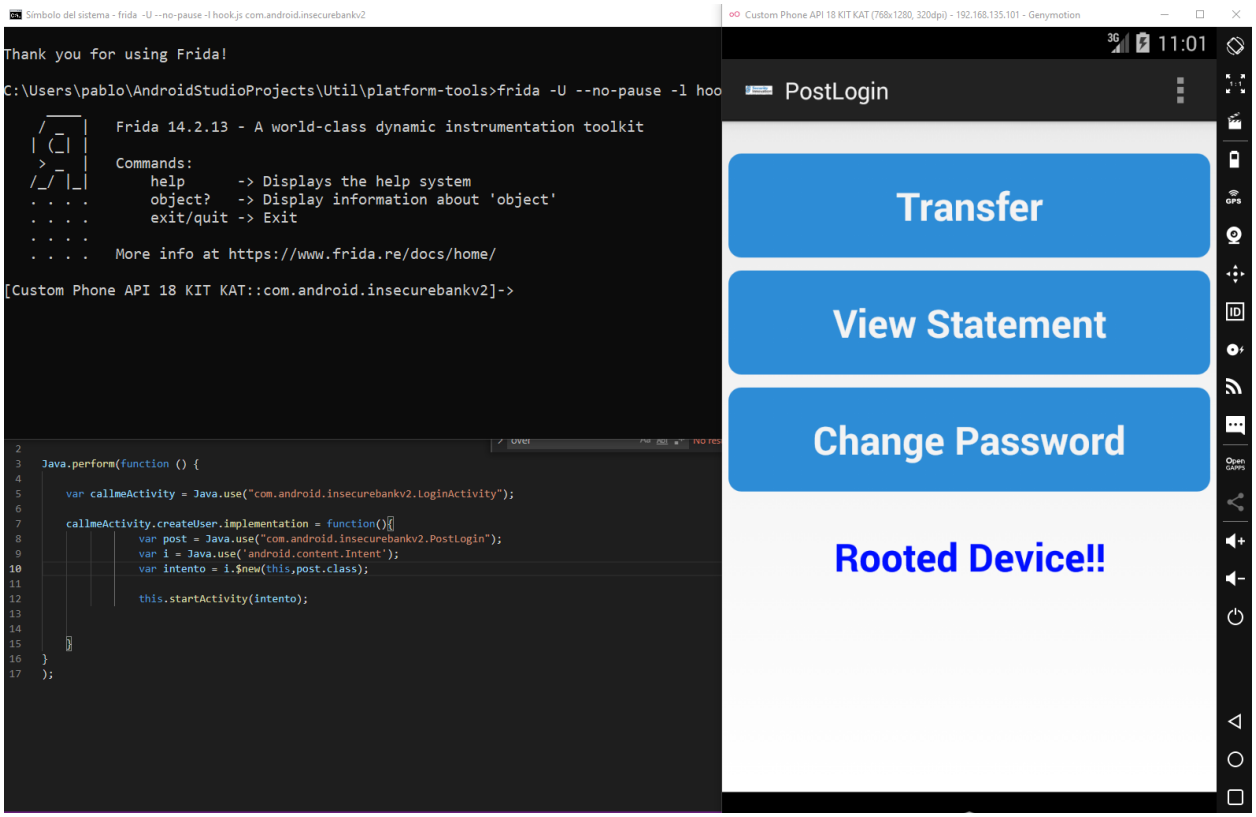
Actuar sobre apps depurables (android:debuggable=”true)

En este primer ejercicio se han seguido los pasos indicados en la práctica y se ha conseguido crear un toast con las características solicitadas al clicar el botón oculto



Baipasear pantalla de login

Con la finalidad de baipasear la pantalla de login se ha modificado en tiempo de ejecución la implementación del método createUser() de la clase LoginActivity, de manera que al pulsar el botón de crear usuario se pase a la ventana de PostLogin. Esto se consigue inyectando el siguiente código JavaScript, que hace una nueva actividad de PostLogin.



Almacenamiento inseguro de credenciales

Con el fin de demostrar que las credenciales se almacenan de forma insegura hemos modificado en tiempo de ejecución el método *aesDeccryptedString()* de la clase CryptoClass, de manera que la nueva implementación llame al método original, imprima por consola la contraseña y finalmente devuelva al método que le ha llamado el valor correcto. El hecho de que se pueda modificar el comportamiento del método en tiempo real y que pueda llamarse a sí mismo almacenarse el resultado dentro de nuestra variable dentro del código inyectado nos abre

muchas posibilidades, pudiendo cambiar el flujo de la ejecución y extraer los datos que se envíen entre las actividades. Además, nos clarifica un poco más el funcionamiento de las ejecuciones de Frida, consiguiendo un mayor control de lo que está pasando en el entorno.

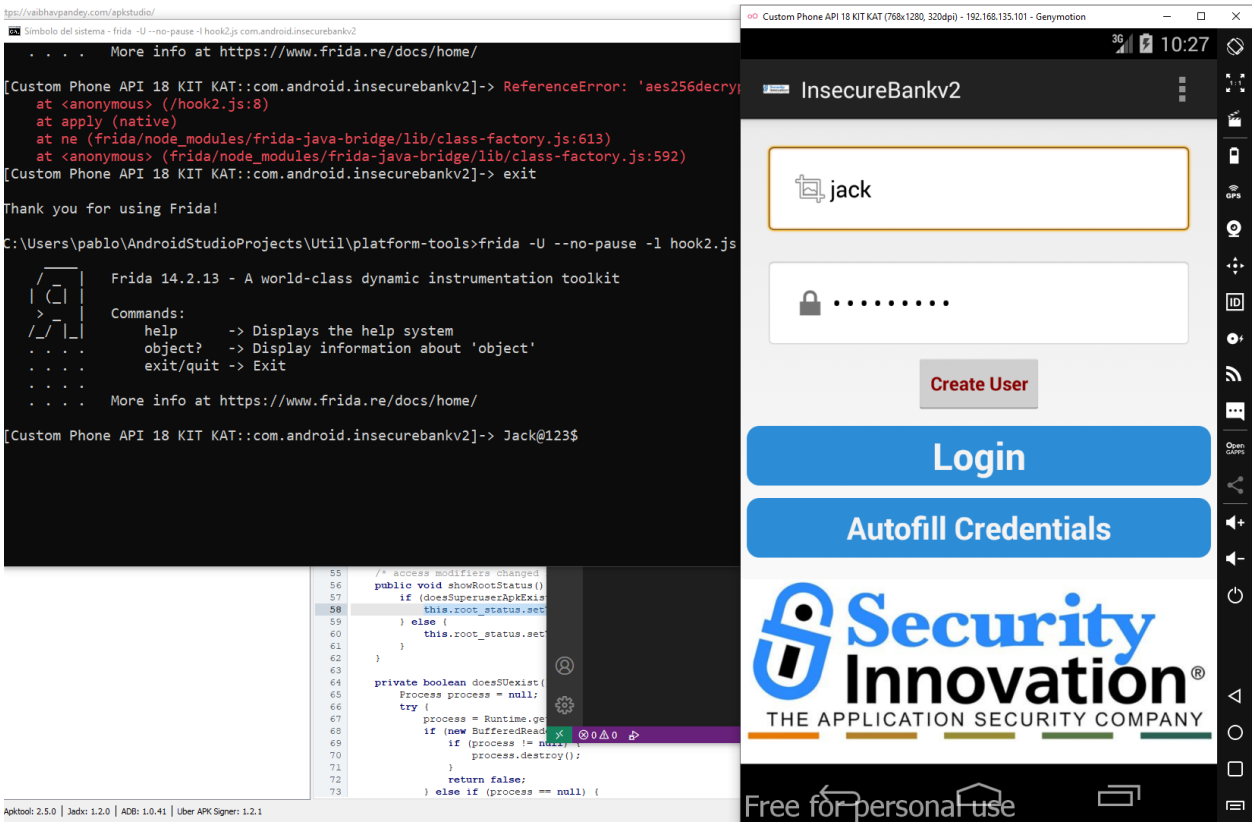
```
Java.perform(function () {


    var callmeActivity = Java.use("com.android.insecurebankv2.CryptoClass");

    callmeActivity.aesDeccryptedString.implementation = function(bundle) {
                var ret =this.aesDeccryptedString(bundle)
                console.log(ret)
                return ret

    };
});
```

Como podemos ver en la siguiente captura en la que acabamos de apretar el botón “Autofill Credentials”, se nos muestra por pantalla la contraseña del usuario jack, con el que ya habíamos iniciado sesión antes en la aplicación, por lo que está almacenado en shared prefs y se puede recuperar. Ahí es donde el código JS inyectado con Frida puede trabajar y recuperar los datos.



Baipasear la detección de root

En este escenario, se han repasado los métodos que controlan si el dispositivo está rooteado o no. Nos encontramos con dos métodos, doesSUexist y doesSuperuserApkExist, a los que modificamos su implementación, retornando el valor “false” en cualquier caso, sin tener en cuenta el funcionamiento anterior a la inyección de código. De esta manera, cualquier intento de detectar si el dispositivo está rooteado por parte del programa será inútil, siempre que esté Frida corriendo de fondo, claro.