

Ciberseguridad en
dispositivos móviles

Práctica 5

Luis López Cuerva
Pablo Alcarria Lozano



Introducción	2
Configuración del entorno de trabajo	2
Captura de tráfico con Wireshark	3
Proxy con Burp Suite	3
Conclusiones	5

Introducción

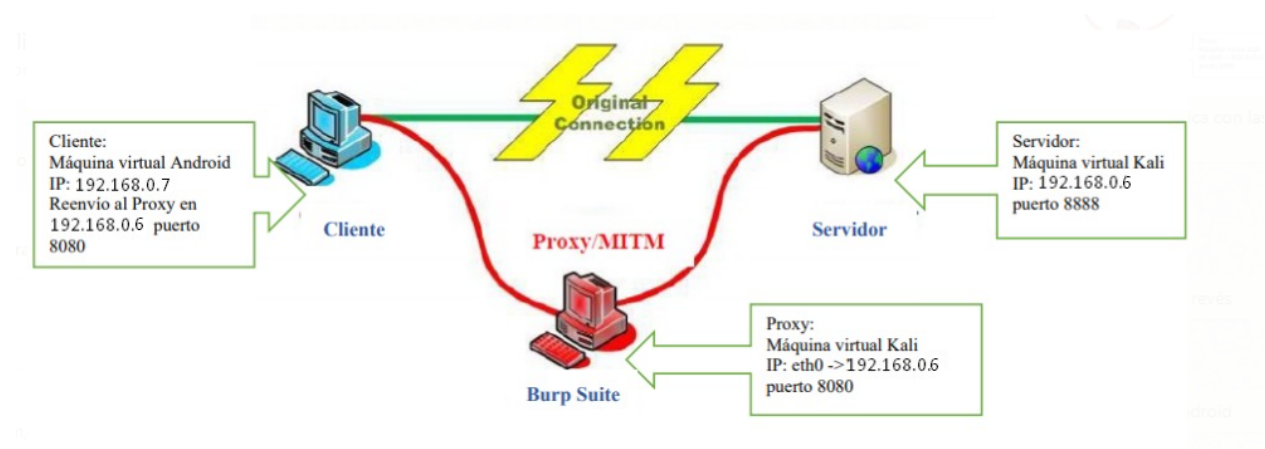
En esta práctica se trabaja el análisis dinámico de la aplicación Insecure Bank v2, con el objetivo de identificar las vulnerabilidades o debilidades que encontramos en la aplicación mientras se encuentra en ejecución. Todas las pruebas realizadas se realizan bajo un entorno controlado con dos máquinas virtuales, una para el cliente de la aplicación, y otra para el servidor y el proxy; diseñado específicamente para el análisis dinámico de la aplicación. Durante las tareas planteadas, se explicarán los problemas identificados para facilitar una posterior solución.

Configuración del entorno de trabajo

El primer paso para la configuración de las dos máquinas virtuales ha sido la creación de una red NAT y la configuración del cliente y el servidor proxy para que la utilicen.

También ha sido necesario modificar la iptables de la máquina virtual que actúa como cliente para que su tráfico sea redireccionado al proxy Burp Suite.

La máquina virtual Kali Linux será la encargada de ejecutar el servidor de Insecure Bank v2 en el puerto 8888, y posteriormente también ejecutará un proxy con Burp Suite (reflejado como el camino rojo) en el puerto 8080, de manera que todo lo que pase hacia el servidor de Insecure Bank antes pasará por nuestro proxy. Junto con WireShark, tenemos el software necesario para monitorizar el tráfico que pase. Por el otro lado, la máquina cliente ejecuta una versión de Android a la que se le ha instalado la APK "InsecureBankv2".



Captura de tráfico con Wireshark

Para realizar la captura de tráfico entre InsecureBank en Android y el servidor, ejecutamos Wireshark en la máquina del servidor, y al filtrar el tráfico, de forma que sólo nos salgan los paquetes entrantes y salientes del puerto 8888, nos encontramos con que las transferencias se envían como una petición POST sin ningún tipo de encriptación, de forma que es visible para cualquier usuario que monitoriza el tráfico. Vemos que este tipo de peticiones se hacen sin encriptar. También resulta curioso cómo para el login hay una protección del usuario y la contraseña, ya que se codifican con la clase CryptoClass.java, y aunque no sea acertado dejar en “shared prefs” el contenido de la aplicación, existía una protección. Es verdad que esta protección era como poner una puerta a un terreno sin valla, ya que los parámetros del cifrado también estaban incluidos en la clase, pero tiene menos sentido aún si a la hora de hacer una transferencia, se envía por texto plano el usuario y su contraseña además de los datos de la transferencia.

```
▼ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▼ Form item: "username" = "jack"
    Key: username
    Value: jack
  ▼ Form item: "password" = "Jack@123$"
    Key: password
    Value: Jack@123$
  ▼ Form item: "from_acc" = "999999999"
    Key: from_acc
    Value: 999999999
  ▼ Form item: "to_acc" = "555555555"
    Key: to_acc
    Value: 555555555
  ▼ Form item: "amount" = "666"
    Key: amount
    Value: 666
```

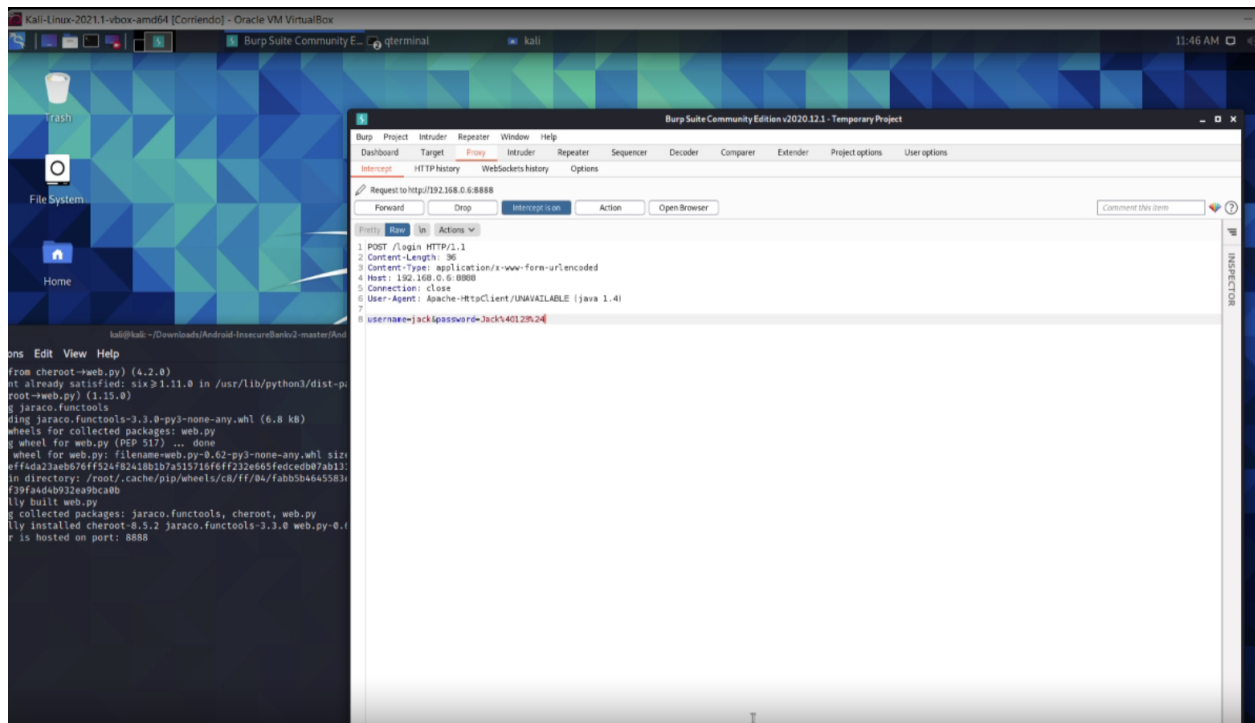
Como principal debilidad, encontramos que la transferencia no se realiza encriptada, y como hemos dicho cualquier usuario conectado a la misma red de cualquiera de ambas partes (servidor o cliente) podría leer el contenido.

Proxy con Burp Suite

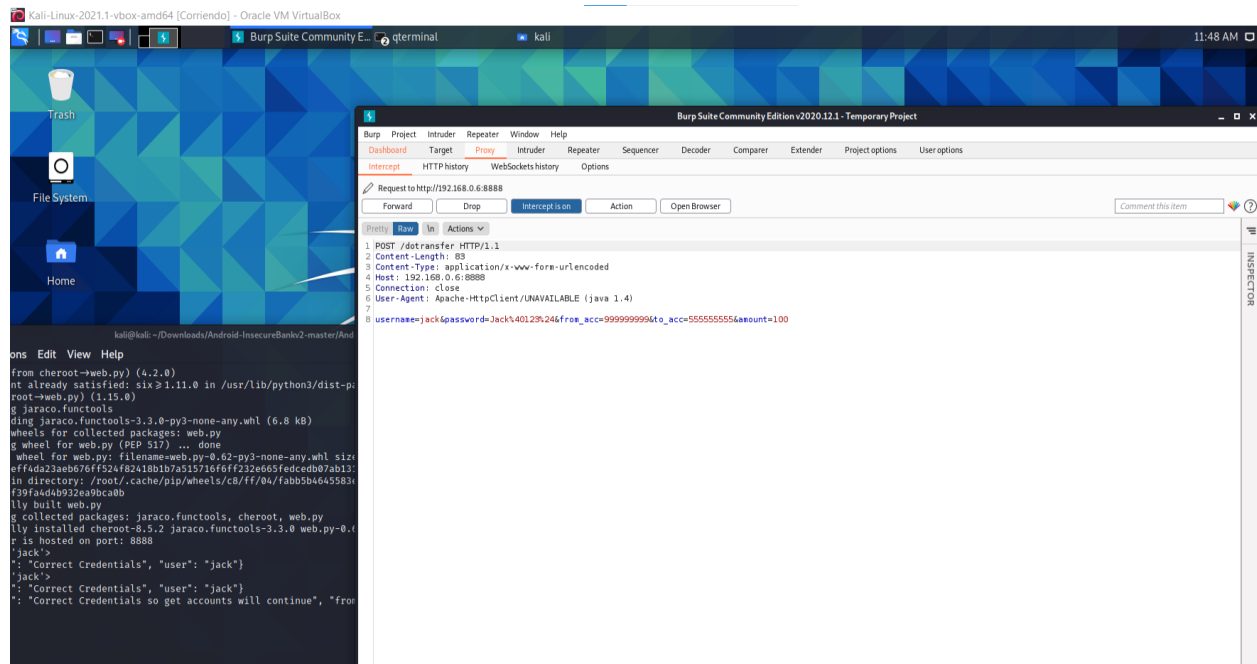
El proxy realizado con Burp Suite, junto a la redirección del tráfico que hay hecha en la máquina Android, nos permite ver el tráfico dirigido hacia la aplicación Insecure Bank, y antes de que atravesase el proxy tenemos la capacidad de verlo, modificarlo, y luego rechazarlo o hacer el

reenvío del paquete. Con el uso conjunto de Burp Suite y Wireshark podemos tener un control total del tráfico del servidor, de la forma que se expone a continuación.

Para comprobar este control en primer lugar se interceptaron las credenciales de inicio de sesión del usuario.




Y a continuación se capturó y modificó el valor de una transacción, haciendo que pasase de ser una transacción de 100 euros a ser una transacción de 2000.



- ▶ Frame 12: 300 bytes on wire (2400 bits), 300 bytes captured (2400 bits) on interface eth0, id 0
- ▶ Ethernet II, Src: PcsCompu_a6:1f:86 (08:00:27:a6:1f:86), Dst: PcsCompu_5b:95:73 (08:00:27:5b:95:73)
- ▶ Internet Protocol Version 4, Src: 192.168.0.6, Dst: 192.168.0.7
- ▶ Transmission Control Protocol, Src Port: 8080, Dst Port: 58798, Seq: 1, Ack: 284, Len: 234
- ▼ Hypertext Transfer Protocol
 - ▶ HTTP/1.1 200 OK\r\n
 - Content-Type: text/html; charset=utf-8\r\n
 - Content-Length: 80\r\n
 - Connection: close\r\n
 - Date: Thu, 01 Apr 2021 15:52:28 GMT\r\n
 - Server: localhost\r\n
 - \r\n
 - [HTTP response 1/1]
 - [Time since request: 20.488754335 seconds]
 - [Request in frame: 5]
 - [Request URI: http://192.168.0.6:8888/dottransfer]
 - File Data: 80 bytes
- ▼ Line-based text data: text/html (1 lines)
 - ["message": "Success", "from": "999999999", "to": "555555555", "amount": "2000"]

Conclusiones

Las dos principales debilidades es que no existe ningún tipo de seguridad ni de verificación de los datos que se transmiten, ya que ni se protegen y cualquier usuario que pueda estar espiando la red lee las comunicaciones entre los clientes y el servidor, y una vez realizado un ataque tipo Man In The Middle, se puede modificar la cantidad de dinero hecha en una transferencia.



En primer lugar, se debería crear algún tipo de seguridad en el tráfico de la red, ya que enviándose todo por texto plano mediante HTTP, se lee. A nivel de diseño de una aplicación en la actualidad debería implementarse HTTPS para cualquier aplicación que tratase alguna información sensible. Se puede discutir el tipo de encriptación a usar en este servicio, pero dadas las características de la aplicación, se requiere uno bastante robusto, ya que se trata con información muy sensible.

Por otra parte, si habiendo un cifrado se interceptase un paquete mediante el ataque, pese a estar cifrado se podría seguir modificando el texto cifrado. En ese caso, lo más probable es que el servidor a la hora de descifrarlo recibiese algo que no comprende, pudiendo causar una denegación del servicio si no hay un manejo de este tipo de errores o excepciones.