# Android 渗透测试 frida——Brida 插件加解密实战演示

Android APP 测试时，经常发生会遇见数据包加密传输，这样就会影响测试。不过，Brida 可以编写加密解密脚本，对加密的数据包进行解密，在加密。工具或者插件都是为了测试方便。

## 环境

```
Android 8.1.0  pixel
Burpsuite 1.7
windows 10
eseBrida.apk （下面给下载地址）
phpstudy
```

## 什么是 Brida

### 介绍 Brida

1、Brida.jar 为 Burpsuite 插件

2、bridaServicePyro 是用于 Frida 适配到 burpsuite 上的 python 脚本，这一部分存储在插件中，在执行 brida 过程中复制到缓存文件夹中。

3、script.js 是要注入到目标应用程序的 javascript 脚本，它会通过 Frida 带有的 rpc.exports 功能将信息返回到拓展程序中。

4、该 script.js 脚本会被 Frida 注入到我们在 Brida 中指定的进程中所以我们可以直接使用 Frida 的 api。

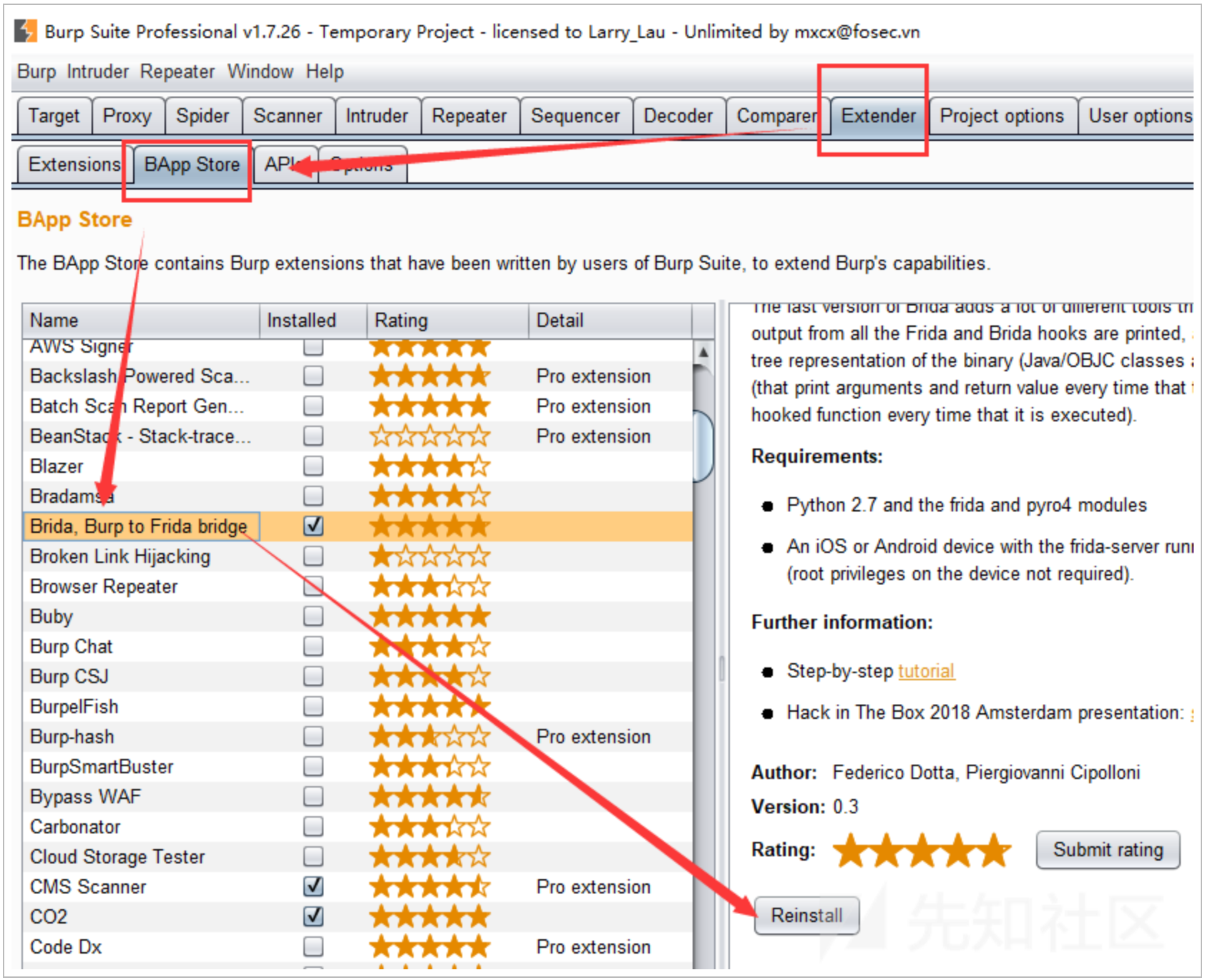
5、目前只支持 python2.7

### 安装 frida

需要安装 frida, 参考文章  https://www.jianshu.com/p/c349471bdef7 (https://www.jianshu.com/p/c349471bdef7)

## 安装 Brida

在 Burpsuite 安装 Brida



(https://xzfile.aliyuncs.com/media/upload/picture/20200417115806-a4dfb36a-805f-1.png)

安装完成, 在 python2.7 安装 Pyro4

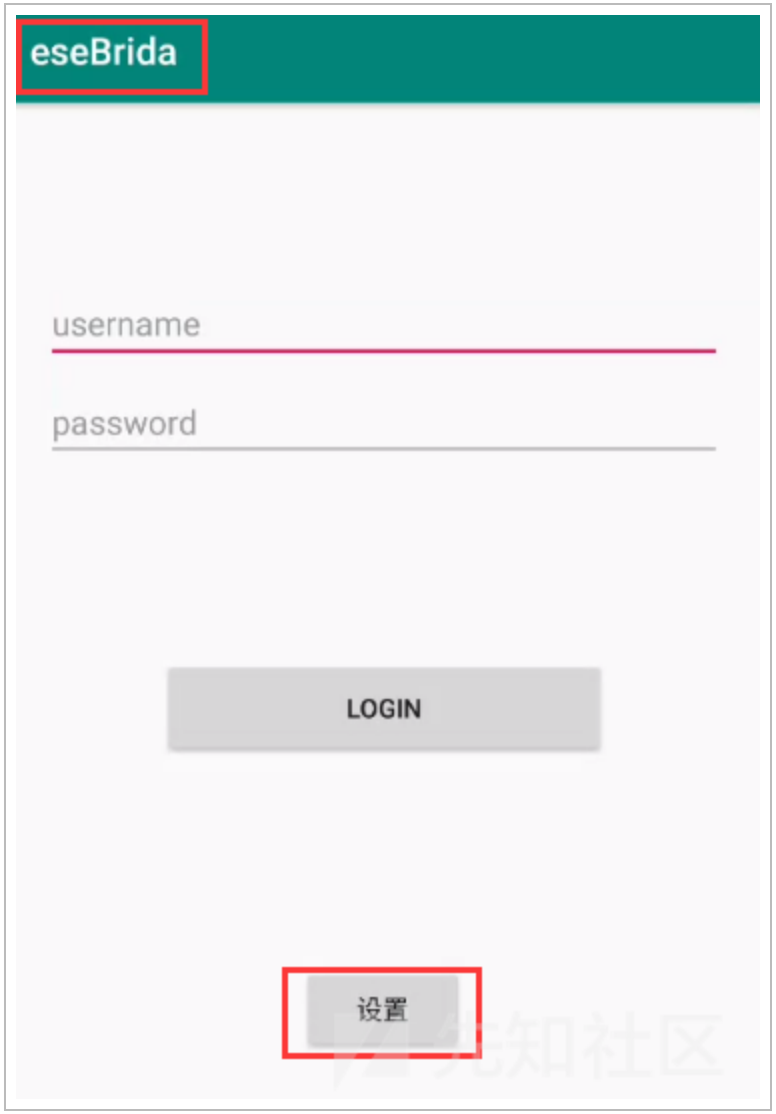
`pip install Pyro4`

# 实战

### apk 运行环境

本文提供一个自己写的 eseBrida.apk (https://pan.baidu.com/s/1r2pKkbsB22FMfu_bD2vhtw)，拿到 apk，因为我这里是测试版本，安装需要加 – t 参数，

`adb install -t esebrida.apk`，安装运行如下:

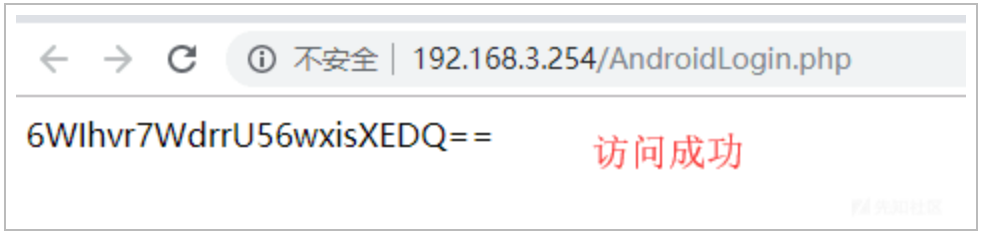



(https://xzfile.aliyuncs.com/media/upload/picture/20200417115839-b8c9be66-805f-1.png)

有个设置按钮，可设置服务器地址。这里利用 phpstudy 在 www 目录下运行 **AndroidLogin.php**，

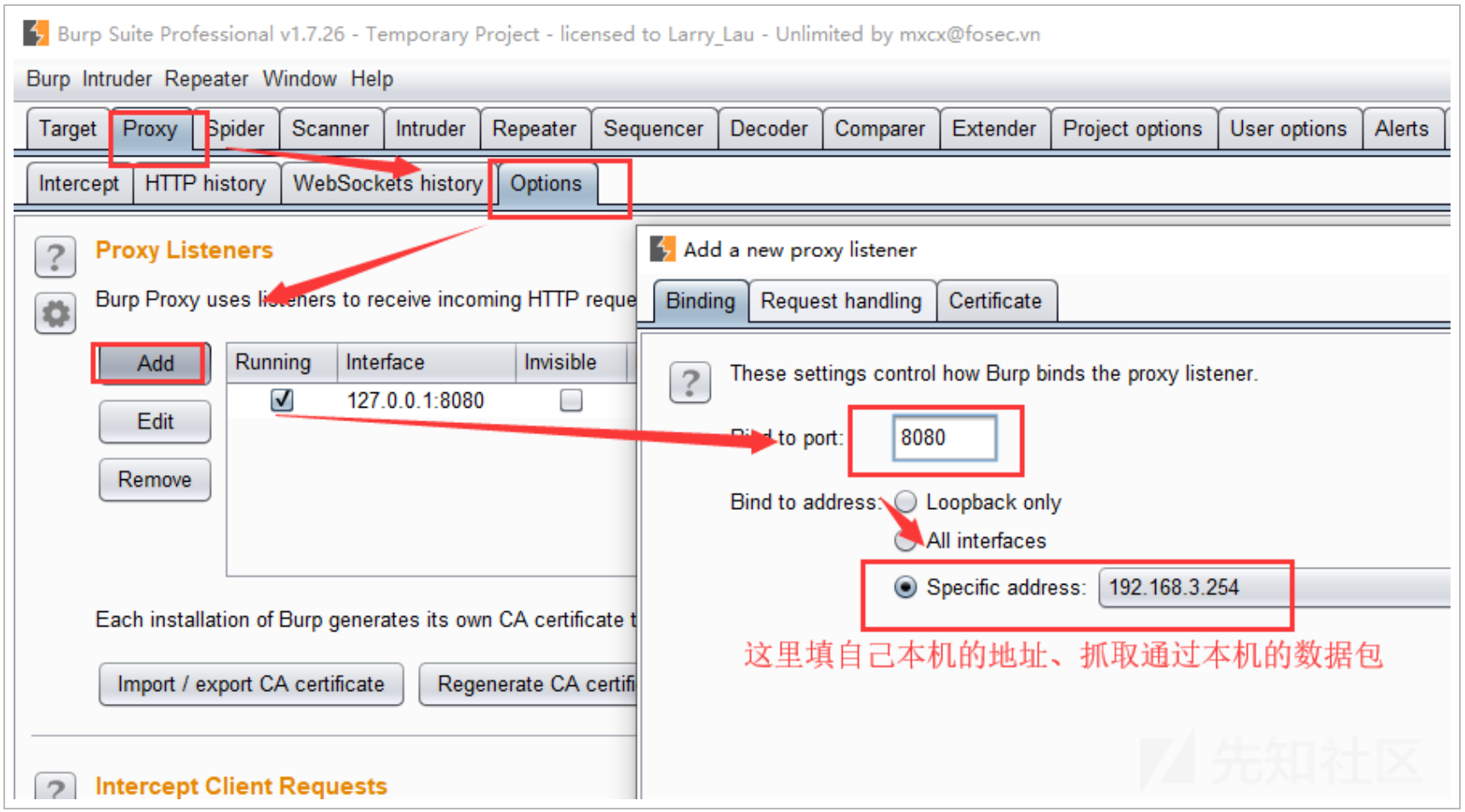

(https://xzfile.aliyuncs.com/media/upload/picture/20200417115858-c3a8e262-805f-1.png)

然后在浏览器访问服务器地址如 http://192.168.3.254/AndroidLogin.php (http://192.168.3.254/AndroidLogin.php) 看是否可以访问成功，如下:




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120241-49029d2c-8060-1.png)

将此地址填入安卓 apk 设置中，burpsuite 设置代理：




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120305-56d911ce-8060-1.png)

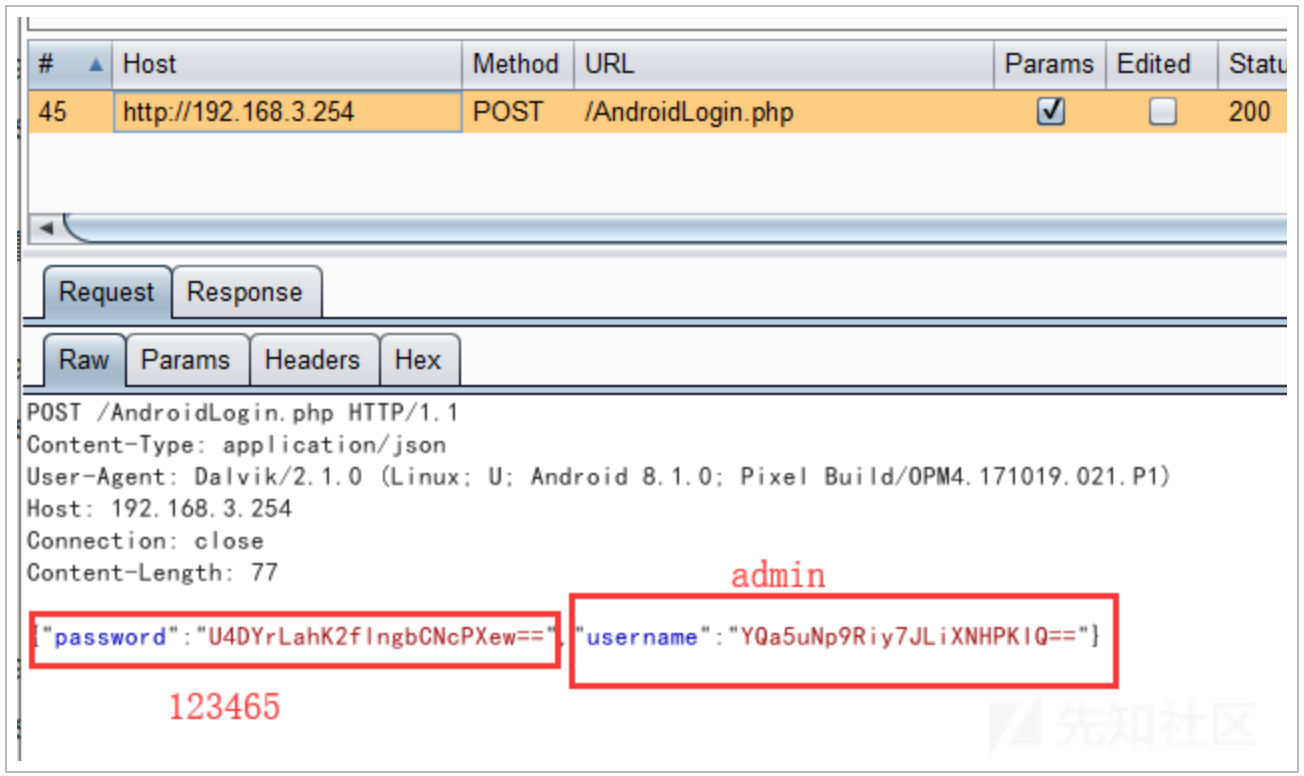
手机 wifi 设置代理，如下：


(https://xzfile.aliyuncs.com/media/upload/picture/20200417120328-64d9870e-8060-1.png)

在 apk 中输入用户名密码，


(https://xzfile.aliyuncs.com/media/upload/picture/20200417115942-ddf25676-805f-1.png)

查看抓取的数据包:




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120358-7699d21e-8060-1.png)

**重点**

这里的数据是密文传输，不利于爆破，这里想对算法进行解密，然后在实现加密传输大奥服务器端。

**分析 apk**

利用 jeb 反编译 apk，发现加解密算法 AesEncryptionBase64 类


(https://xzfile.aliyuncs.com/media/upload/picture/20200417120418-8278bf32-8060-1.png)

上层定位，发现加密算法的秘钥硬编码，如下

```
}

public void executePOST(Context arg13, UserInfo arg14, String arg15) {
    String v0 = "9876543210123456";
    URLConnection v2 = null;
    InputStream v3 = null;
    String v5 = arg15;
    try {
        Log.e("eseTools", v5);
        if(v5 == null || v5 == "" || (v5.isEmpty())) {
            v5 = "http://192.168.3.254/AndroidLogin.php";
        }

        v2 = new URL(v5).openConnection();
        ((HttpURLConnection)v2).setRequestMethod("POST");
        ((HttpURLConnection)v2).setRequestProperty("Content-Type", "application/json");
        ((HttpURLConnection)v2).getOutputStream();
        ((HttpURLConnection)v2).connect();
        OutputStream v4 = ((HttpURLConnection)v2).getOutputStream();
        String v6 = AesEncryptionBase64.encrypt(v0, arg14.getUsername());
        arg14.setPassword(AesEncryptionBase64.encrypt(v0, arg14.getPassword()));
        arg14.setUsername(v6);
        v4.write(JSON.toJSONString(arg14).getBytes());
```

发现加密秘钥



(https://xzfile.aliyuncs.com/media/upload/picture/20200417120434-8bebcc58-8060-1.png)

自此，apk 的流程已经分析清楚了。接着可以有两种思路

> 方法1、将java代码复制出来，在eclipse实现以下加解密流程，就可以对传输的数据进行解密加密了。
> 方法2、利用Brida调用apk自身加密解密函数，一键实现加密解密操作。

自然，方法 2 相对 1 要简单，而且操作方便。所有便有这篇文章

**编写 Brida js 脚本**

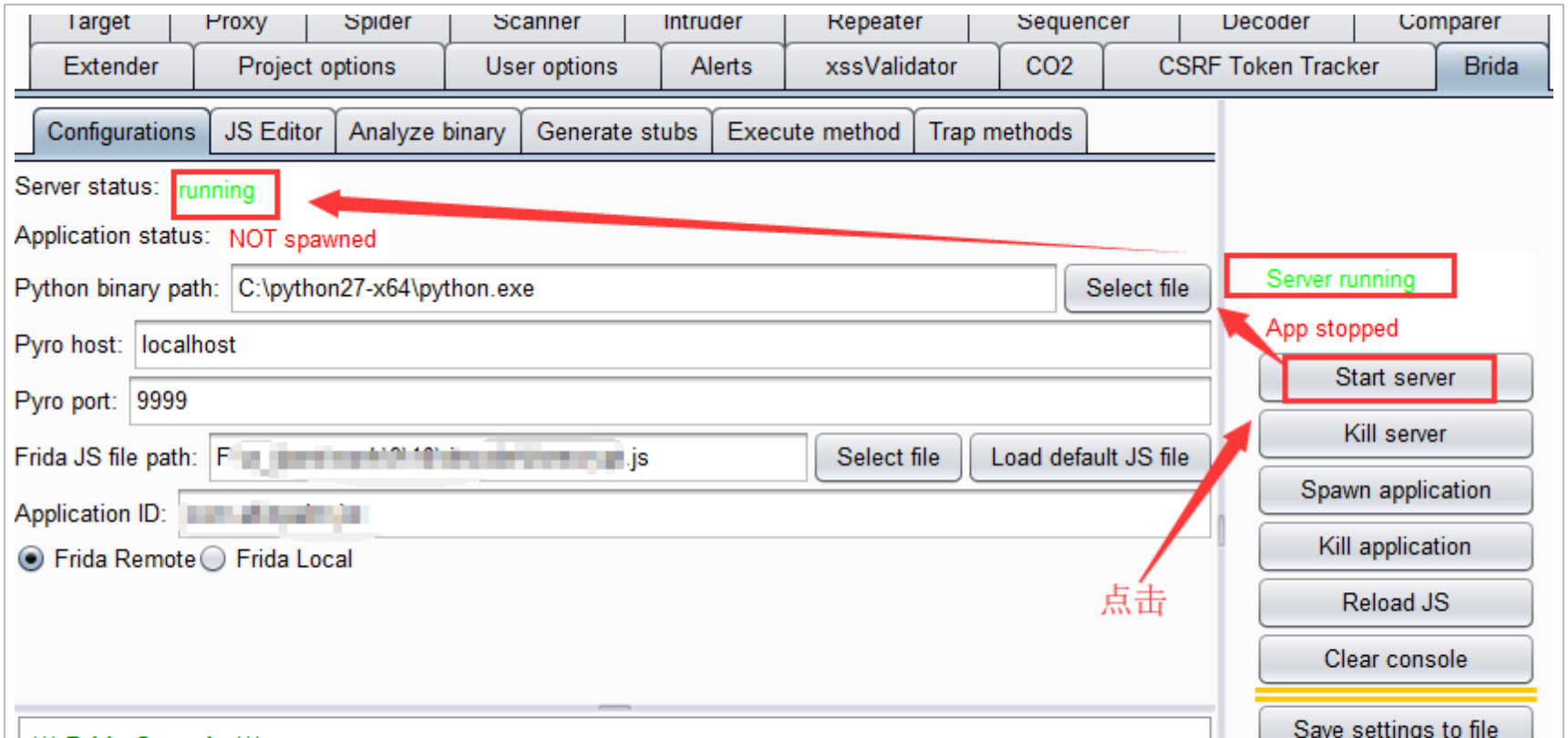
**1、运行 frida**

首先下载 startFridaService.py (https://www.jianshu.com/p/fa422d3b7148)，运行 `python startFridaService.py`

```
λ python startFridaService.py
adb forward tcp:27042 tcp:27042
adb forward tcp:27043 tcp:27043
Android server--->./frida-server64
success-->frida-ps -R
```

(https://xzfile.aliyuncs.com/media/upload/picture/20200417120450-95c71476-8060-1.png)

**2、运行 Brida，如下**

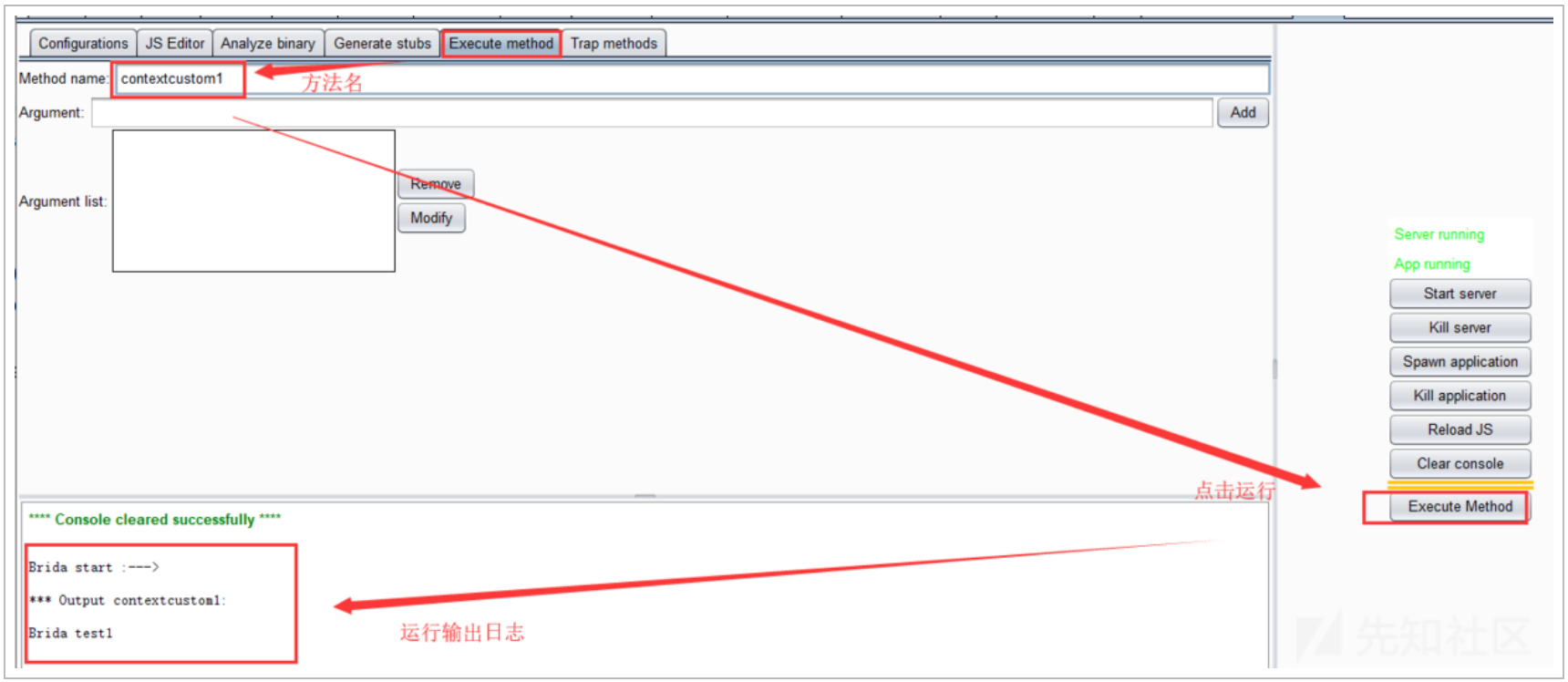



(https://xzfile.aliyuncs.com/media/upload/picture/20200417115822-ae576e9c-805f-1.png)

### 3、先给一个 Brida 简单的 test.js 框架

```
'use strict';

// 1 - FRIDA EXPORTS
rpc.exports = {
    exportedFunction: function() {
    },

    contextcustom1: function(message) {
        console.log("Brida start :--->");
        return "Brida test1";
    },

    getplatform: function () {
        if (Java.available) {
            return 0;
        } else if (ObjC.available) {
            return 1;
        } else {
            return 2;
        }
    }
}
```

### 4、测试方法 contextcustom1




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120553-bb1baf0c-8060-1.png)

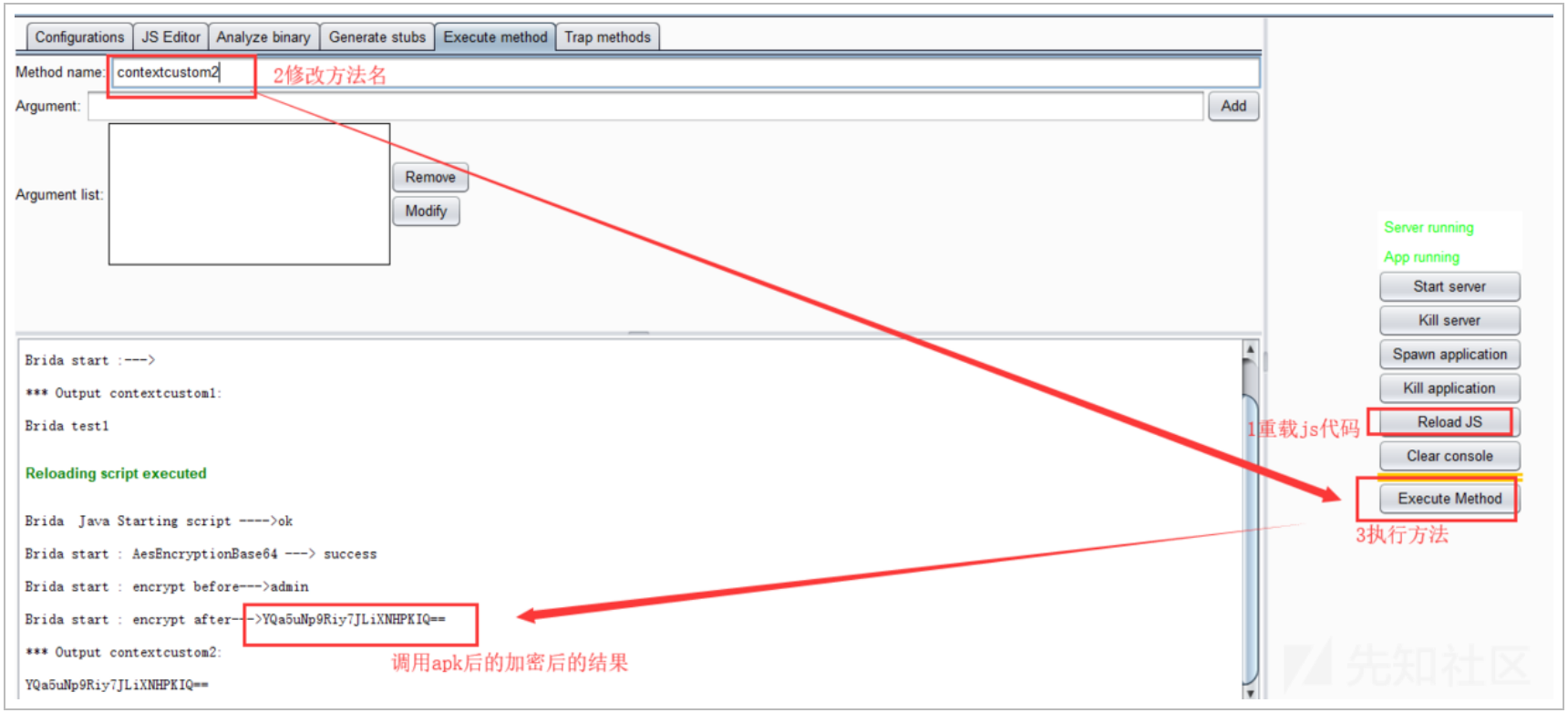
成功执行脚本

**5、编写 Brida 调用 encrypt 加密函数**

```
'use strict';

// 1 - FRIDA EXPORTS
rpc.exports = {
    exportedFunction: function() {
    },

    contextcustom1: function(message) {
        console.log("Brida start :--->");
        return "Brida test1";
    },

    contextcustom2: function(message) {
        console.log("Brida  Java Starting script ---->ok");
        var enc;
        Java.perform(function () {
            try {
                var key = "9876543210123456";
                var text = "admin";
                //hook class
                var AesEncryptionBase64 = Java.use('com.ese.http.encrypt.AesEncryptionBase64');
                console.log("Brida start : encrypt before--->"+text);
                //hook method
                enc = AesEncryptionBase64.encrypt(key,text);
                console.log("Brida start : encrypt after--->"+enc);

            } catch (error) {
                console.log("[!]Exception:" + error.message);
            }
        });
        return enc;
    },

    getplatform: function () {
        if (Java.available) {
            return 0;
        } else if (ObjC.available) {
            return 1;
```

```
        } else {
            return 2;
        }
    }
}
```

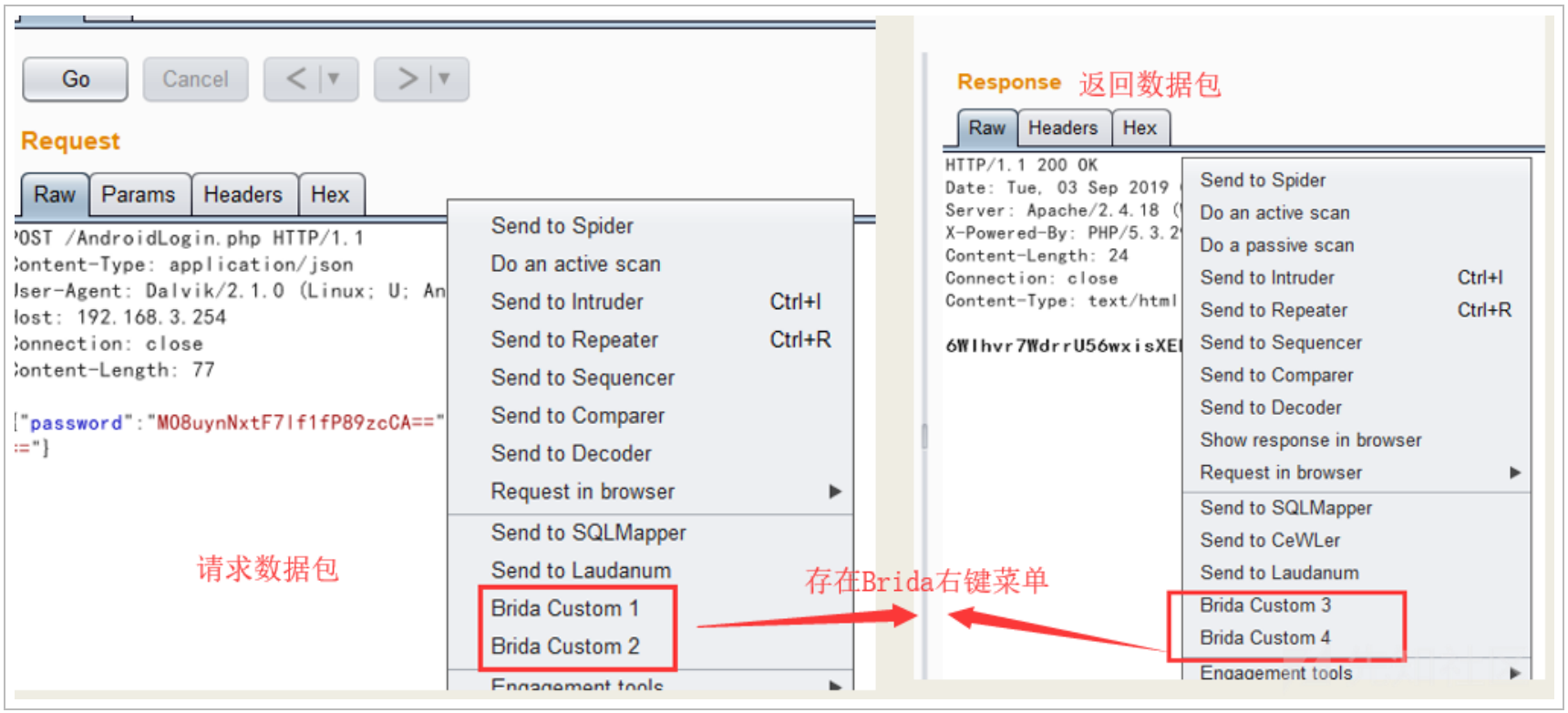
### 6、执行方法 contextcustom2




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120909-302f819c-8061-1.png)

通过签名抓取的数据包， 发现加密数据一致，证实调用 apk 加密算法。

### 7、Burpsuite 右键菜单




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120831-19461dc4-8061-1.png)

发现 4 个方法与请求数据包与返回数据包相互一一对应

1、Brida Custom 1----->contextcustom1

2、Brida Custom 2----->contextcustom2

3、Brida Custom 3----->contextcustom3

4、Brida Custom 4----->contextcustom4

**8、编写对应插件 eseScript.js 脚本**

加载其他脚本，需要重启 burpsuite,

```
'use strict';

// 1 - FRIDA EXPORTS
rpc.exports = {
    exportedFunction: function() {
    },

    //AesEncryptionBase64 encrypt
    contextcustom1: function (message) {
        console.log("Brida start :0--->" + message);
        var data = hexToString(message)
        console.log("Brida start :1--->" + data);
        var enc;
        Java.perform(function () {
            try {
                var key = "9876543210123456";
                var text = data;
                //hook class
                var AesEncryptionBase64 = Java.use('com.ese.http.encrypt.AesEncryptionBase64');
                console.log("Brida start : AesEncryptionBase64 ---> success");
                console.log("Brida start : encrypt before--->"+text);
                //hook method
                enc = AesEncryptionBase64.encrypt(key,text);
                console.log("Brida start : encrypt after--->"+enc);

            } catch (error) {
                console.log("[!]Exception:" + error.message);
            }
        });
        return stringToHex(enc);
    },

    //AesEncryptionBase64 decrypt
    contextcustom2: function (message) {
        console.log("Brida start :0--->" + message);
        var data = hexToString(message)
        console.log("Brida start :1--->" + data);
        var text;
```

```
    Java.perform(function () {
        try {
            var key = "9876543210123456";
            var enc = data;
            //hook class
            var AesEncryptionBase64 = Java.use('com.ese.http.encrypt.AesEncryptionBase64');

            console.log("Brida start : AesEncryptionBase64 ---> success");
            console.log("Brida start : decrypt before--->"+enc);
            //hook method
            text = AesEncryptionBase64.decrypt(key,enc);
            console.log("Brida start : decrypt after--->"+text);
        } catch (error) {
            console.log("[!]Exception:" + error.message);
        }
    });
    console.log("Brida start : decrypt after--->"+stringToHex(text));
    return stringToHex(text);
},

//AesEncryptionBase64 encrypt
contextcustom3: function (message) {
    console.log("Brida start :0--->" + message);
    var data = hexToString(message)
    console.log("Brida start :1--->" + data);
    var enc;
    Java.perform(function () {
        try {
            var key = "9876543210123456";
            var text = data;
            //hook class
            var AesEncryptionBase64 = Java.use('com.ese.http.encrypt.AesEncryptionBase64');
            console.log("Brida start : AesEncryptionBase64 ---> success");
            console.log("Brida start : encrypt before--->"+text);
            //hook method
            enc = AesEncryptionBase64.encrypt(key,text);
            console.log("Brida start : encrypt after--->"+enc);

        } catch (error) {
            console.log("[!]Exception:" + error.message);
        }
    });
    return stringToHex(enc);
},

//AesEncryptionBase64 decrypt
```

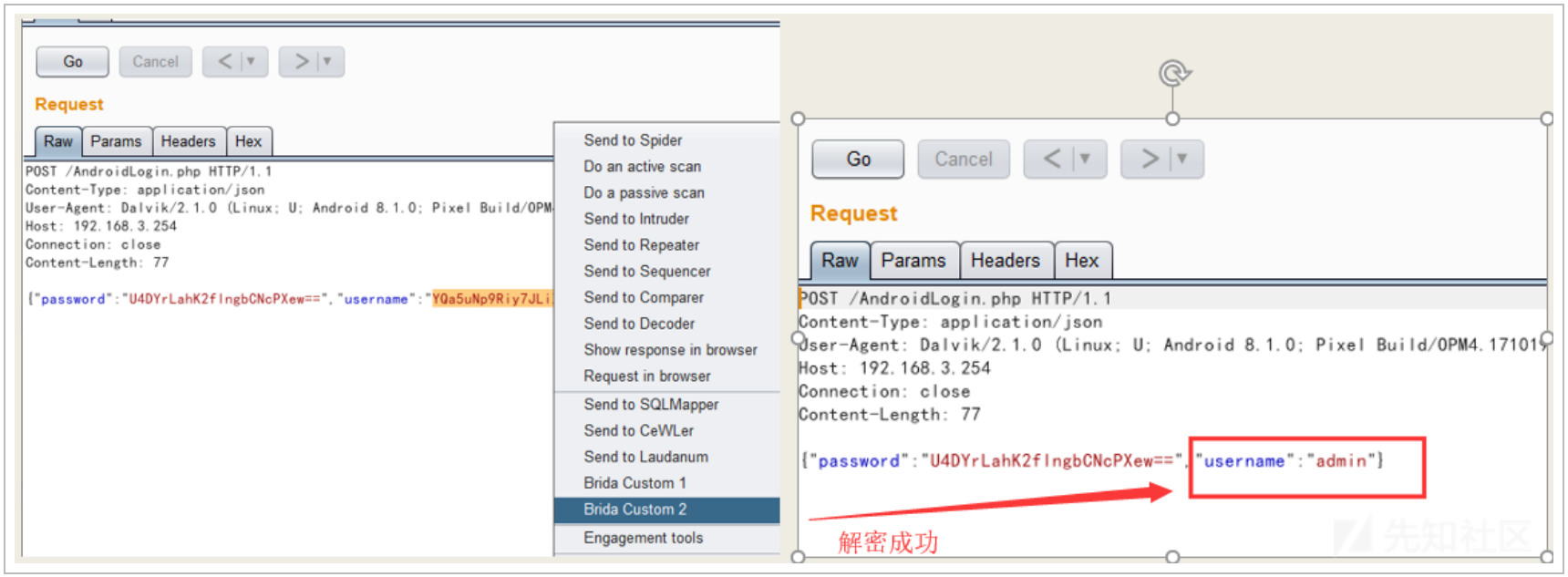
```
    contextcustom4: function (message) {
        console.log("Brida start :0--->" + message);
        var data = hexToString(message)
        console.log("Brida start :1--->" + data);
        var text;
        Java.perform(function () {
            try {
                var key = "9876543210123456";
                var enc = data;
                //hook class
                var AesEncryptionBase64 = Java.use('com.ese.http.encrypt.AesEncryptionBase64');
                console.log("Brida start : AesEncryptionBase64 ---> success");
                console.log("Brida start : decrypt before--->"+enc);
                //hook method
                text = AesEncryptionBase64.decrypt(key,enc);
                console.log("Brida start : decrypt after--->"+text);
            } catch (error) {
                console.log("[!]Exception:" + error.message);
            }
        });
        console.log("Brida start : decrypt after--->"+stringToHex(text));
        return stringToHex(text);
    },

    getplatform: function () {
        if (Java.available) {
            return 0;
        } else if (ObjC.available) {
            return 1;
        } else {
            return 2;
        }

    }

}

// Convert a ASCII string to a hex string
function stringToHex(str) {
    return str.split("").map(function(c) {
        return ("0" + c.charCodeAt(0).toString(16)).slice(-2);
    }).join("");
}

// Convert a hex string to a ASCII string
```

```
function hexToString(hexStr) {
    var hex = hexStr.toString();//force conversion
    var str = '';
    for (var i = 0; i < hex.length; i += 2)
        str += String.fromCharCode(parseInt(hex.substr(i, 2), 16));
    return str;
}
```

注意: 因为从 message 接受的数据是 hex，所有调用 hexToString 转成字符串，然后进行加密操作，最后调用 hexToString 转换成 hex 返回。

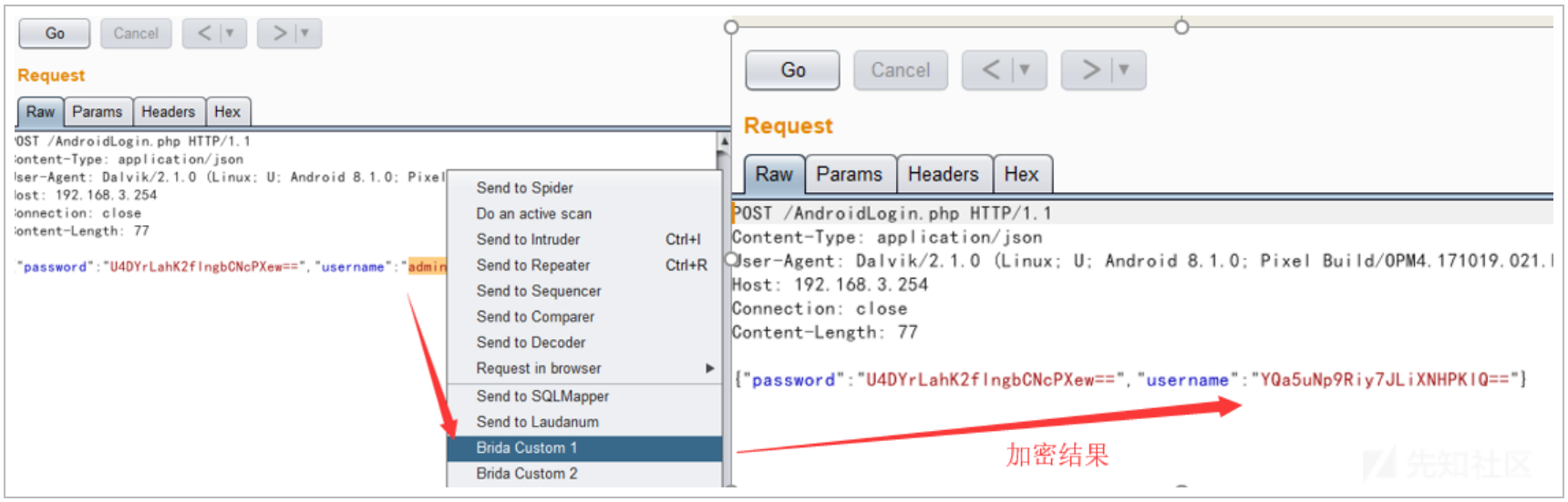
### 9、运行效果

解密




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120707-e7a2c9ca-8060-1.png)

加密




(https://xzfile.aliyuncs.com/media/upload/picture/20200417120731-f59de226-8060-1.png)

自此，解实现了一键加密，一键解密操作。本片文章的目的就达到了。

你以为就结束了吗？No No No .....

当你输入 账号: admin 密码: 654321