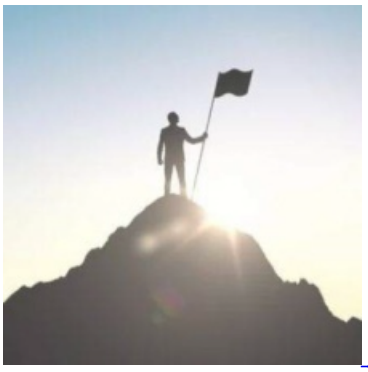
XCTF MOBILE 新手 app1、app2、easy-apk

原创

A_dmins 于 2019-09-30 17:18:32 发布

分类专栏： CTF题 XCTF

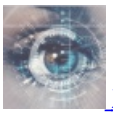
本文链接：https://blog.csdn.net/qq_42967398/article/details/101779509



CTF题 同时被 2 个专栏收录

115 篇文章 11 订阅

订阅专栏



XCTF

24 篇文章 0 订阅

订阅专栏

XCTF MOBILE 新手 app1、app2、easy-apk

app1

使用APKIDE打开apk文件，查看java源码，找到主类：

```
implements View.OnClickListener
{
  MainActivity$1(MainActivity paramMainActivity) {}

  public void onClick(View paramView)
  {
    for (;;)
    {
      int i;
      try
      {
        paramView = this.this$0.text.getText().toString();
        PackageInfo localPackageInfo = this.this$0.getPackageManager().getPackageInfo("com.example.yaphetshan.tencentgreat", 16384)
        String str = localPackageInfo.versionName;
        int j = localPackageInfo.versionCode;
        i = 0;
        if ((i < paramView.length()) && (i < str.length()))
        {
          if (paramView.charAt(i) != (str.charAt(i) ^ j)) {
            Toast.makeText(this.this$0, "再接再厉，加油~", 1).show();
          }
        }
        else if (paramView.length() == str.length())
        {
          Toast.makeText(this.this$0, "恭喜开启闯关之门！", 1).show();
          return;
        }
      }
      catch (PackageManager.NameNotFoundException paramView)
      {
        Toast.makeText(this.this$0, "年轻人不要耍小聪明噢", 1).show();
        return;
      }
      i += 1;
```


经过分析发现是异或，我们就去找这两个常数：

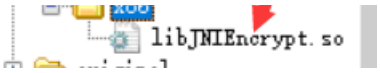
```
public final class BuildConfig
{
  public static final String APPLICATION_ID = "com.example.yaphetshan.tencentgreat";
  public static final String BUILD_TYPE = "debug";
  public static final boolean DEBUG = Boolean.parseBoolean("true");
  public static final String FLAVOR = "";
  public static final int VERSION_CODE = 15;
  public static final String VERSION_NAME = "X<cP[?PHNB<P?aj";
}
```

直接编写脚本进行处理：

```
s = "X<cP[?PHNB<P?aj"
n = 15
flag = ""
for i in s:
 j = ord(i)
 flag += chr(j ^ n)
print(flag)
```

app2

使用APKIDE打开apk文件，我们发现这里多了一个libc的文件夹，libc好像是什么函数动态链接库



查看一下，里面好像有个什么解密的？？？？
先不管，继续往下走~~
主函数里面没看出什么鬼东西，不过发现了一个类中有一个特殊的字符串，需要解密啥的：

```
import android.os.Bundle;
import android.widget.TextView;
import com.tencent.testvuln.c.Encryto;

public class FileDataActivity
  extends a
{
  private TextView c;

  protected void onCreate(Bundle paramBundle)
  {
    super.onCreate(paramBundle);
    setContentView(2130903042);
    this.c = ((TextView)findViewById(2131165184));
    this.c.setText(Encryto.decode(this, "9YuQ2dk8CSaCe7DTAmaqAA=="));
  }
}
```


看见import com.tencent.testvuln.c.Encryto;，直接进入其中：

```
public class Encryto
{
  static
  {
    System.loadLibrary("JNIEncrypt");
  }

  public static native int checkSignature(Object paramObject);

  public static native String decode(Object paramObject, String paramString);

  public static native String doRawData(Object paramObject, String paramString);

  public static native String encode(Object paramObject, String paramString);

  public native String HelloLoad();
}
```


发现好熟悉，就显示我们刚刚在libc中看见的那个东西？？？一个解密的函数？？？
直接是看libc中的这个文件，直接解压apk2，找到libc中的这个文件
.os文件直接使用ida打开，发现有一个encode函数，直接查看伪代码：

```
1  int __cdecl encode(int a1, int a2, int a3, int a4)
2  {
3    char *v4; // esi
4    int v5; // ST10_4
5    int result; // eax
6    char *v7; // esi
7    int (__cdecl *v8)(int, char *, size_t); // ST10_4
8    size_t v9; // eax
9    int v10; // [esp+4h] [ebp-28h]
10   int v11; // [esp+8h] [ebp-24h]
11   int v12; // [esp+Ch] [ebp-20h]
12   int v13; // [esp+10h] [ebp-1Ch]
13   char v14; // [esp+14h] [ebp-18h]
14   unsigned int v15; // [esp+18h] [ebp-14h]
15
16   v15 = __readgsdword(0x14u);
17   if ( checkSignature(a1, a2, a3) == 1 )
18   {
```

```
  {
    v14 = 0;
    v13 = '==ye';
    v12 = 'ktse';
    v11 = 'tasi';
    v10 = 'siht';
    v4 = (char *)(*(int (__cdecl **)(int, int, _DWORD))(*(_DWORD *)a1 + 676))(a1, a4, 0);
    v5 = AES_128_ECB_PKCS5Padding_Encrypt(v4, (int)&v10);
    (*(void (__cdecl **)(int, int, char *))(*(_DWORD *)a1 + 680))(a1, a4, v4);
    result = (*(int (__cdecl **)(int, int))(*(_DWORD *)a1 + 668))(a1, v5);
  }
  else
  {
    v7 = UNSIGNATURE[0];
    v8 = *(int (__cdecl **)(int, char *, size_t))(*(_DWORD *)a1 + 652);
    v9 = strlen(UNSIGNATURE[0]);
    result = v8(a1, v7, v9);
  }
  return result;
```


根据名字感觉是AES加密，不过AES需要密钥，我们可以看见上面有一串字符串：thisisatestkey==
猜测这就是密钥，试试：

好像就是，，，，提交正确，，，，
说实话这里面的函数完全没看懂，，，全靠猜，，
一脸懵逼

easy-apk

首先我们使用APKIDE打开apk文件，并且查看java源码，找到主类：

```
package com.testjava.jack.pingan1;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity
  extends AppCompatActivity
{
  protected void onCreate(Bundle paramBundle)
  {
    super.onCreate(paramBundle);
    setContentView(2130968603);
    ((Button)findViewById(2131427446)).setOnClickListener(new View.OnClickListener()
    {
      public void onClick(View paramAnonymousView)
      {
        paramAnonymousView = ((EditText)MainActivity.this.findViewById(2131427445)).getText().toString();
        if (new Base64New().Base64Encode(paramAnonymousView.getBytes()).equals("5rFf7E2K6rqN7Hpiyush7E6S5fJg6rsi5NBf6NGT5rs="))
        {
          Toast.makeText(MainActivity.this, "验证通过!", 1).show();
          return;
        }
        Toast.makeText(MainActivity.this, "验证失败!", 1).show();
      }
    });
  }
}
```


很明显看出有比较，应该是我们输入的字符串进行base64加密之后
与5rFf7E2K6rqN7Hpiyush7E6S5fJg6rsi5NBf6NGT5rs=进行比较
所以我们需要对这个字符串进行base64解密，但是我们去解密会发现错误

￿_￿M￿깕￿zb￿￿!￿N￿￿￿`￿"￿￿_￿f￿	5rFf7E2K6rqN7Hpiyush7E6S5fJg6rsi5NBf6NGT5rs=

所以我们去查看一下码表

```
private static final char[] Base64ByteToStr = { 118, 119, 120, 114, 115, 116, 117, 111, 112, 113, 51, 52, 53, 54, 55, 65, 66, 67, 68, 69, 70,
private static final int RANGE = 255;
private static byte[] StrToBase64Byte = new byte['㘀'];

public String Base64Encode(byte[] paramArrayOfByte)
{
  StringBuilder localStringBuilder = new StringBuilder();
  int j = 0;
  while (j <= paramArrayOfByte.length - 1)
  {
    byte[] arrayOfByte = new byte[4];
    int i = 0;
    int k = 0;
    if (k <= 2)
    {
      if (j + k <= paramArrayOfByte.length - 1) {
        arrayOfByte[k] = ((byte)((paramArrayOfByte[(j + k)] & 0xFF) >>> k * 2 + 2 | i));
      }
      for (i = (byte)(((paramArrayOfByte[(j + k)] & 0xFF) << (2 - k) * 2 + 2 & 0xFF) >>> 2);; i = 64)
      {
        k += 1;
        break;
```


编写脚本查看一下：

```
a = [118, 119, 120, 114, 115, 116, 117, 111, 112, 113, 51, 52, 53, 54,
55, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 121, 122, 48, 49, 50, 80,
81, 82, 83, 84, 75, 76, 77, 78, 79, 90, 97, 98, 99, 100, 85, 86, 87,
88, 89, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 56, 57, 43, 47]

flag = ""

for i in a:
 flag += chr(i);

print(flag)
```

```
python 3.py
vwxrstuopq34567ABCDEFGHIJyzO12PQRSTKLMNOZabcdUVWXYefghijklmn89+/
```

ok，码表不同，所以需要使用另外的base64解密脚本
这里我已经写好了，就直接使用python编写base64加解密脚本（可变换码表）
得到运行结果：

```
python base64.py

      ***************************************
      *     (1)encode          (2)decode     *
      ***************************************

Please select the operation you want to perform:
2
Please enter a string that needs to be decrypted:
5rFf7E2K6rqN7Hpiyush7E6S5fJg6rsi5NBf6NGT5rs=
Decrypted String:
05397c42f9b6da593a3644162d36eb01
```


裹上flag{}提交正确