看雪CTF 2017 第六题设计思路和解题思路

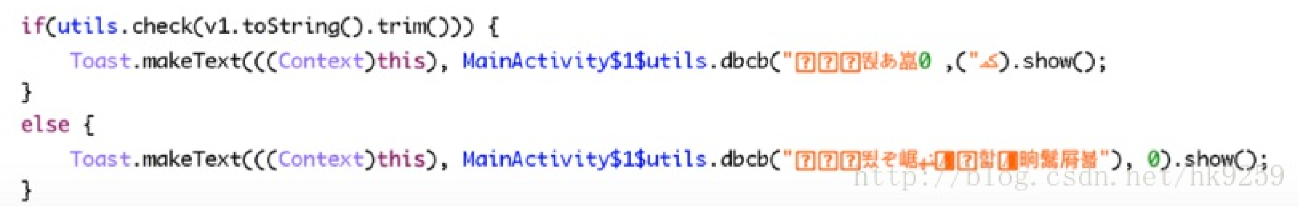
原创

Ericky_ 于 2017-06-14 21:41:34 发布

分类专栏： Android 文章标签： CTF 看雪 逆向 花指令 apk

这道题主要需要花时间搞清楚套路，就迎刃而解了。^_^

1.java层稍作字符串加密和类名方法名混淆处理(本来是打算java层也做点文章的@_@)



```
if(utils.check(v1.toString().trim())) {
    Toast.makeText(((Context)this), MainActivity$1$utils.dbcb("???裂あ嵩0 ,("ؼ").show();
}
else {
    Toast.makeText(((Context)this), MainActivity$1$utils.dbcb("???裂そ崛+??晳?晌鬣屑嵩"), 0).show();
}
```


解题：通过阅读代码，可以知道check函数为关键函数，当返回为真的时候，注册成功。

2.so层用花指令对程序指令进行混淆，在一定程度上防止分析。

```
sub_1ABE                                ; CODE XREF: sub_1AC0+20 p
                B               sub_1AE2
; End of function sub_1ABE


; =============== S U B R O U T I N E =======================================


sub_1AC0                                ; CODE XREF: sub_1AD6-24 p

; FUNCTION CHUNK AT .text:00001AB4 SIZE 0000000A BYTES
; FUNCTION CHUNK AT .text:00001ADC SIZE 00000006 BYTES

                PUSH            {R0,R4,R5,R7,LR}
                B               loc_1AC8
; ---------------------------------------------------------------------------

loc_1AC4                                ; CODE XREF: sub_1AC0+A j
                MOV             R2, R2
                B               loc_1ACC
; ---------------------------------------------------------------------------

loc_1AC8                                ; CODE XREF: sub_1AC0+2 j
                SUB             SP, SP, #8
                B               loc_1AC4
```

解题：编写去花指令脚本，F5就会变得很简单。去花后check函数如下：

```
 do
 {
   v10 = *((_WORD *)&v15 + v7);
   v11 = v7 + 3;
   if ( v10 == -1 )
     goto LABEL_10;
   v12 = (char *)&v15 + 2 * v7;
   v5 = (int *)*((_WORD *)v12 + 1);
   if ( (unsigned __int16)v5 == 0xFFFF )
   {
     v9 = *((_BYTE *)&v15 + 2 * v10);
     byte_20020[2 * v8++] = v9;
ABEL_10:
     v7 = v11;
     continue;
   }
   v7 = *((_WORD *)v12 + 2);
   LOWORD(v9) = *((_WORD *)&v15 + (_DWORD)v5) - *((_WORD *)&v15 + v10);
   *((_WORD *)&v15 + (_DWORD)v5) = v9;
   v9 = (signed __int16)v9;
   if ( (signed __int16)v9 > 0 )
     goto LABEL_10;
 }
 while ( v7 > -1 );
 sub_19FC(v9, v8, 0xFFFF, v5, v15, v16, v17, v18, v19);
 v13 = 0;
 (*(void (__fastcall **)(int, int, _DWORD))(*(_DWORD *)v3 + 676))(v3, v4, 0);
 v14 = (char *)sub_19DA8();
 while ( byte_20020[v13] == (unsigned __int8)v14[v13] )
 {
   if ( ++v13 == 24 )
   {
     result = 1;
     goto LABEL_17;
   }
 }
 sub_27C8(byte_20020, v14);
 result = 0;
ABEL_17:
 if ( _stack_chk_guard != v32 )
   _stack_chk_fail(result, _stack_chk_guard - v32);
```


这样就能很清楚看到注册过程了，第一个while循环释放存放好的字符串到byte_20020。sub_19DA8为加密函数，返回结果到v14，最后v14与byte_20020做比较，一样就行了。算法采用了RC4加密+base64编码。

3.值得一提的是，so中藏关键字符串数据的方式我自己觉得挺有意思的。不知道聪明的你有没有发现呢？如果不会去花也不要紧，还有一个解题的思路就是是找到最后的汇编cmp，下好断点，因为这个算法的缘故。java层可以遍历传入字符串进行遍历暴力破解。

4.总结一下，我自己想到了两个解题方式，一个暴力解题，一个去花解题。从男人刚正面的角度来讲，这道题主要就是考去花，没有加其他的东西了，直接被大佬们秒得体无完肤。。。先行膜拜。最后附上答案：madebyericky94528