看雪上有人分享的一个注册机的代码分析

原创

普通网友 于 2014-06-03 21:01:02 发布 604 收藏

最近看到有人在看雪上分享了一个apk，是关于注册机的闯关游戏。但是由于代码都在JAVA层，所以没有实际价

反编译apk，得到smali，当然也可以得到java源代码。

```
 const/4 v0, 0x0

.line 18
.local v0, answer:I
const/4 v3, 0x0


.local v3, x:I
:goto_0
invoke-virtual {p1}, Ljava/lang/String;->length()I


move-result v4


if-ge v3, v4, :cond_0


.line 20
invoke-virtual {p1, v3}, Ljava/lang/String;->charAt(I)C


move-result v1


.line 21
.local v1, current:C
mul-int v4, v1, v1


add-int/2addr v0, v4


.line 22
xor-int/2addr v0, v1


.line 18
add-int/lit8 v3, v3, 0x1



goto :goto_0
```

```
.line 26
    .end local v1           #current:C
    :cond_0
    :try_start_0
<span style="white-space:pre"> </span>
<span style="white-space:pre"> </span>
<span style="white-space:pre"> </span>
    invoke-static {p2}, Ljava/lang/Integer;->parseInt(Ljava/lang/String;)I
    :try_end_0
    .catch Ljava/lang/Exception; {:try_start_0 .. :try_end_0} :catch_0
<span style="white-space:pre"> </span>
<span style="white-space:pre"> </span>


    move-result v2
<span style="white-space:pre"> </span>
<span style="white-space:pre"> </span>


    .line 27
    .local v2, numericSerial:I


<span style="white-space:pre"> </span>
    if-ne v2, v0, :cond_1#判断注册码
<span style="white-space:pre"> </span>


    .line 29
    const/4 v4, 0x1


    .line 32
    .end local v2           #numericSerial:I
    :goto_1
<span style="white-space:pre"> </span>
    return v4


    .line 31
    :catch_0
    move-exception v4


    .line 32
    :cond_1
    const/4 v4, 0x0


    goto :goto_1
```

可以看到，算法主要思想是根据输入的用户名计算得到一个值，存在寄存器v0里面，然后判断v0和v2（输入的密码）是否相等，相等则通过，返回值v4为1，否则不通过，v4值为0。通过返回V4记

现在两种方法，一是暴力破解，直接将 v4寄存器的值改为0x01,不管v0和v2的判断结果是什么，都是返回1，那

方法二，让程序自己吐出密码。

```
    const/4 v0, 0x0

    .line 18
    .local v0, answer:I
    const/4 v3, 0x0

    .local v3, x:I
    :goto_0
    invoke-virtual {p1}, Ljava/lang/String;->length()I

    move-result v4

    if-ge v3, v4, :cond_0

    .line 20
    invoke-virtual {p1, v3}, Ljava/lang/String;->charAt(I)C

    move-result v1

    .line 21
    .local v1, current:C
    mul-int v4, v1, v1

    add-int/2addr v0, v4

    .line 22
    xor-int/2addr v0, v1

    .line 18
    add-int/lit8 v3, v3, 0x1

invoke-static {v0}, Lcrack;->log_int(I)V

    goto :goto_0



    .line 26
    .end local v1           #current:C
    :cond_0
    :try_start_0



    invoke-static {p2}, Ljava/lang/Integer;->parseInt(Ljava/lang/String;)I
    :try_end_0
    .catch Ljava/lang/Exception; {:try_start_0 .. :try_end_0} :catch_0



    move-result v2



    .line 27
    .local v2, numericSerial:I
```

```
    if-ne v2, v0, :cond_1#判断注册码


    .line 29
    const/4 v4, 0x1

    .line 32
    .end local v2           #numericSerial:I
    :goto_1

    return v4

    .line 31
    :catch_0
    move-exception v4

    .line 32
    :cond_1
    const/4 v4, 0x0

    goto :goto_1
```

这里的invoke-static {v0}, Lcrack;->log_int(I)V是我自己定义的一个类，作用是方便打印调试信息，因为大家知道，在smali里面添加简单的一句log都要好几行，特别麻烦，在这里我将它包装成了一个类，具体实现如下：

```
.method public static log_int(I)V
    .locals 2

    .prologue

    const-string v0, "info"

    invoke-static {p0}, Ljava/lang/String;->valueOf(I)Ljava/lang/String;

    move-result-object v1

    invoke-static {v0, v1}, Landroid/util/Log;->d(Ljava/lang/String;Ljava/lang/String;)I

    return-void
.end method
```

这样我们只要调用invoke-static {v0}, Lcrack;->log_int(I)V就可以打印了，我随便讲该代码放在了程序的一部分，结果如下：

可以看到，经过一系列运算后，67300就是我们最后需要的密码。