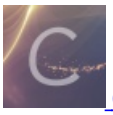
【moeCTF题解-0x01】Reverse

原创

框架主义者 于 2020-10-09 23:37:00 发布 643 收藏 4

分类专栏： CTF

本文链接：https://blog.csdn.net/weixin_47102975/article/details/108988480



---

title: 【moeCTF题解-0x01】Reverse

categories:

- CTF
- moeCTF

tags:

- CTF

【moeCTF题解-0x01】Reverse

一个全新的领域

> **【moeCTF题解】总目录如下：**
>
> 【moeCTF题解-0x00】序 （包括Sign in）
>
> 【moeCTF题解-0x01】Reverse （包括Android、IoT）
>
> 【moeCTF题解-0x02】Pwn
>
> 【moeCTF题解-0x03】Algorithm
>
> 【moeCTF题解-0x04】Crypto
>
> 【moeCTF题解-0x05】Misc
>
> 【moeCTF题解-0x06】Classic Crypto
>
> 【moeCTF题解-0x07】Web

逆向工程入门指北

> 25points

开题，是一篇 Reverier 大大写的逆向工程指引

阅读到reverse.pdf的最后得到flag：

`moectf{0hhhhhhh_I_kn0w_hoW_t0_R3v3rs3!}`

Welcome To Re!

> 50points
> 欢迎来到逆向工程的世界!
> 注: 在题目的压缩包中你会发现两个二进制文件. 没有后缀的为 `Linux x86_64` 平台上的可执行程序, 有后缀的为 `Windows x86_64` 平台上的可执行程序. 两个程序的验证逻辑与解出的 `flag` 均相同, 你选择其中一个进行逆向分析即可. 多平台是考虑到使用不同系统的选手均能在自己的平台上执行题目程序. 部分题目程序可能存在一些系统操作, 如果引起了安全软件报毒请忽略或在虚拟机内进行逆向. 题目程序均不会对系统造成破坏, 请放心食用.
> 点击下方的 `View Hint` 可以查看出题人给出的解题提示哦~ 有时候hint也许能祝你一臂之力!

开了免费的Hint，提示用 `IDA`

用IDA64反汇编，直接F5反编译一下main：

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char Str1; // [rsp+20h] [rbp-60h]
  char Str2[8]; // [rsp+50h] [rbp-30h]

  _main();
  strcpy(Str2, "moectf{W3lc0me-T0_th3-W0rld_Of_R3v3rsE!}");
  puts("Welcome to MoeCTF! --by Reverier\nPlease Input your flag and I will check it:");
  scanf("%41s", &Str1);
  if ( !strcmp(&Str1, Str2) )
    puts("Congratulations!");
  else
    puts("Ruaaaaaaaaaaaaa~~~Wrong!");
  getchar();
  getchar();
  return 0;
}
```

就看到flag了：`moectf{W3lc0me-T0_th3-W0rld_Of_R3v3rsE!}`

Thank you JavaScript

75points

flag请精确提交哦 格式: moectf{xxxx}

下载后看到一个只有一行的奇奇怪怪程序：

```
eval(function(p,a,c,k,e,d){e=function(c){return(c<a?"":e(parseInt(c/a)))+((c=c%a)>35?String.fromCharCode(c+29):c
.toString(36))};if(!''.replace(/^/,String)){while(c--)d[e(c)]=k[c]||e(c);k=[function(e){return d[e]}];e=function
(){return'\\w+'};c=1;};while(c--)if(k[c])p=p.replace(new RegExp('\\b'+e(c)+'\\b','g'),k[c]);return p;}('l 1=m(\'
k-4-2\');i j 6(){1.2(\'q r p --n o b\');1.2(5 1.4());1.2(`a ${5 1.d(\'9 h e?\')}!`);f 3=g;F(!3){1.2(\'D E 7 B 8:
\');3=5 1.4()===\'G{H\'+\'c\'+\'v\'+\'w\'+\'0\'+\'u-\'+\'s\'+\'t\'+\'z\'+\'A\'+\'!}\'}1.2(\'y! x C 7 8!\')}6();'
,44,44,'|io|write|saidHi|read|await|main|the|flag|Who|Hello|Reverier||ask|you|let|false|are|async|function|conso
le|const|require|written|by|ThankYouJavaScript|MoeCTF|2020|Jav|aS||k_|Y|You|Congratulations|cr|ipt|true|find|Ple
ase|input|while|moectf|Fx'.split('|'),0,{}))
```

啊啊啊啊啊这是什么鬼呀，格式化一下也没有用，还是一行语句，也看不懂哪里有输入输出，但node竟然能运行

尝试手动重构，失败*N

观察程序的特征，发现`eval(function(p,a,c,k,e,d{...})`，于是上百度，发现原来是一个js代码混淆工具产生的特征，将原程序前的`eval`换成`console.log`再运行可以轻松反混淆：

```
require('console-read-write');
async function main() {
    io.write('MoeCTF 2020 ThankYouJavaScript --written by Reverier');
    io.write(await io.read());
    io.write(`Hello ${await io.ask('Who are you?')}!`);
    let saidHi = false;
    while (!saidHi) {
        io.write('Please input the true flag:');
        saidHi = await io.read() === 'moectf{Fx' + 'c' + 'k_' + 'Y' + '0' + 'u-' + 'Jav' + 'aS' + 'cr' + 'ipt' +
 '!}'
    }
    io.write('Congratulations! You find the flag!')
}
```

得到flag：`moectf{Fxck_Y0u-JavaScript!}`

> JavaScript真的是一门可he爱xie的语言呢~

具体原理可参考：密码学笔记——eval(function(p,a,c,k,e,d) 的加密破解

> 其实这个eval(function(p,a,c,k,e,d){}))中自带解码函数e()，“while(c–)if(k[c])p=p.replace(new RegExp(’\b’+e©+’\b’,‘g’),k[c]);return p” while循环产生的每个p就是解码后的函数代码,我们删掉源码中的“return p”,不用将结果返回, 而是直接输出在一个文本区域中

SimpleRe

> 100points
> xor xor xor!!!
> 异或真的是一种美妙的操作呢~ 注: 在题目的压缩包中你会发现两个二进制文件. 没有后缀的为 `Linux x86_64` 平台上的可执行程序, 有后缀的为 `Windows x86_64` 平台上的可执行程序. 两个程序的验证逻辑与解出的 `flag` 均相同, 你选择其中一个进行逆向分析即可. 多平台是考虑到使用不同系统的选手均能在自己的平台上执行题目程序. 部分题目程序可能存在一些系统操作, 如果引起了安全软件报毒请忽略或在虚拟机内进行逆向. 题目程序均不会对系统造成破坏, 请放心食用.

IDA 反编译，发现关键函数 `enc`：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-n2YzSTgb-1602257797665)(【moeCTF题解-0x01】Reverse/image-20201009134249123.png)]

cpoy一下，稍作修改，并把另一个地方找到的

```
.data:0000000000404040 aim             db 'rpz|kydKw^qTl@Y/m2f/J-@o^k.,qkb',0
```

带入：

```c
#include <stdio.h>

int main()
{
    int result;    // eax
    signed int i4; // [rsp+20h] [rbp-40h]
    signed int i3; // [rsp+24h] [rbp-3Ch]
    signed int i2; // [rsp+28h] [rbp-38h]
    signed int i1; // [rsp+2Ch] [rbp-34h]
    signed int nn; // [rsp+30h] [rbp-30h]
    signed int mm; // [rsp+34h] [rbp-2Ch]
    signed int ll; // [rsp+38h] [rbp-28h]
    signed int kk; // [rsp+3Ch] [rbp-24h]
    signed int jj; // [rsp+40h] [rbp-20h]
    signed int ii; // [rsp+44h] [rbp-1Ch]
    signed int n;  // [rsp+48h] [rbp-18h]
    signed int m;  // [rsp+4Ch] [rbp-14h]
    signed int l;  // [rsp+50h] [rbp-10h]
    signed int k;  // [rsp+54h] [rbp-Ch]
    signed int j;  // [rsp+58h] [rbp-8h]
    signed int i;  // [rsp+5Ch] [rbp-4h]
    char out[] = "rpz|kydKw^qTl@Y/m2f/J-@o^k.,qkb";
    for (i = 0; i <= 30; ++i)
        out[i] ^= 0x17;
    for (j = 0; j <= 30; ++j)
        out[j] ^= 0x39u;
    for (k = 0; k <= 30; ++k)
        out[k] ^= 0x4Bu;
    for (l = 0; l <= 30; ++l)
        out[l] ^= 0x4Au;
    for (m = 0; m <= 30; ++m)
        out[m] ^= 0x49u;
    for (n = 0; n <= 30; ++n)
        out[n] ^= 0x26u;
    for (ii = 0; ii <= 30; ++ii)
        out[ii] ^= 0x15u;
    for (jj = 0; jj <= 30; ++jj)
        out[jj] ^= 0x61u;
    for (kk = 0; kk <= 30; ++kk)
        out[kk] ^= 0x56u;
    for (ll = 0; ll <= 30; ++ll)
        out[ll] ^= 0x1Bu;
    for (mm = 0; mm <= 30; ++mm)
        out[mm] ^= 0x21u;
    for (nn = 0; nn <= 30; ++nn)
        out[nn] ^= 0x40u;
    for (i1 = 0; i1 <= 30; ++i1)
        out[i1] ^= 0x57u;
    for (i2 = 0; i2 <= 30; ++i2)
        out[i2] ^= 0x2Eu;
    for (i3 = 0; i3 <= 30; ++i3)
        out[i3] ^= 0x49u;
    for (i4 = 0; i4 <= 30; ++i4)
        out[i4] ^= 0x37u;
    printf("%s",out);
    return 0;

}
```

因为异或加密与解密是对称的，所以我们可以直接用`gcc`编译出可执行文件，运行得到flag：

`moectf{ThAnKs_F0r-y0U2_pAt13nt}`

Protection

> 100points
> 曾经Reverier写了个小软件, 结果很轻易的就被别人破解, 然后贴了个其他的标志就拿出去卖了. 从那以后, Reverier就开始四处寻找方法来保护自己的程序. 比如...给程序套一层衣服.

先IDA一下，没头绪，于是百度……

得知Linux下的程序加壳的很少（开源精神嘛），常用的加壳也只有几种方法，比如`upx`

观察原程序的hex，返现确实是使用upx加壳的

```
0004e6f0: 00 0A 00 24 49 6E 66 6F 3A 20 54 68 69 73 20 66    ...$Info:.This.f
0004e700: 69 6C 65 20 69 73 20 70 61 63 6B 65 64 20 77 69    ile.is.packed.wi
0004e710: 74 68 20 74 68 65 20 55 50 58 20 65 78 65 63 75    th.the.UPX.execu
0004e720: 74 61 62 6C 65 20 70 61 63 6B 65 72 20 68 74 74    table.packer.htt
0004e730: 70 3A 2F 2F 75 70 78 2E 73 66 2E 6E 65 74 20 24    p://upx.sf.net.$
0004e740: 0A 00 24 49 64 3A 20 55 50 58 20 33 2E 39 36 20    ..$Id:.UPX.3.96.
0004e750: 43 6F 70 79 72 69 67 68 74 20 28 43 29 20 31 39    Copyright.(C).19
0004e760: 39 36 2D 32 30 32 30 20 74 68 65 20 55 50 58 20    96-2020.the.UPX.
0004e770: 54 65 61 6D 2E 20 41 6C 6C 20 52 69 67 68 74 73    Team..All.Rights
0004e780: 20 52 65 73 65 72 76 65 64 2E 20 24 0A 00 90 90    .Reserved..$....
```

解铃还须系铃人，upx也可以用于脱壳，Linux环境下运行`upx -d`得到脱壳后的程序

上IDA64分析源码：

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  signed int i; // [rsp+Ch] [rbp-34h]
  char v5[40]; // [rsp+10h] [rbp-30h]
  unsigned __int64 v6; // [rsp+38h] [rbp-8h]

  v6 = __readfsqword(0x28u);
  printf((unsigned __int64)"please input your flag: ");
  _isoc99_scanf((unsigned __int64)"%28s");
  for ( i = 0; i <= 27; ++i )
  {
    if ( ((unsigned __int8)x[i] ^ (unsigned __int8)v5[i]) != y[i] )
    {
      puts("wrong!", v5);
      return 0;
    }
  }
  puts("right!", v5);
  return 0;
}
```

编写相应的解密脚本：

```
x = "aouv#@!V08asdozpnma&*#%!$^&*"
y = [0x0C, 0, 0x10, 0x15, 0x57, 0x26, 0x5A, 0x23, 0x40 ,0x40,0x3E,
0x42, 0x37, 0x30, 9, 0x19, 3, 0x1D, 0x50, 0x43, 7, 0x57, 0x15,
0x7E, 0x51, 0x6D, 0x43, 0x57, 0,0,0,0]

# [0x0C, 0, 0x10, 0x15, 0x57, 0x26, 0x5A, 0x23, 2,0x3E,
# 0x42, 0x37, 0x30, 9, 0x19, 3, 0x1D, 0x50, 0x43, 7, 0x57, 0x15,
# 0x7E, 0x51, 0x6D, 0x43, 0x57, 4]
# ((unsigned __int8)x[i] ^ (unsigned __int8)v5[i]) != y[i] )
flag = ''
for i in range(28):
    flag = flag + chr(ord(x[i]) ^ y[i])

print(flag)
```

得到flag：`moectf{upx_1S_simp1e-t0_u3e}`

Real EasyPython

> 100points
> 人生苦短, 我用 `pyyyyyyyyyython`
> 环境: `Python 3.7.8 x86_64`
> **Q&A**
> - 本题是什么？
> Python逆向。
> - 怎么逆？
> uncompyle，祝你一路顺风。

根据提示，使用`uncompyle6`进行反编译：

```
uncompyle6 -o puzz.py puzzle.pyc
```

```
# uncompyle6 version 3.7.4
# Python bytecode 3.7 (3394)
# Decompiled from: Python 3.8.2 (default, Jul 16 2020, 14:00:26)
# [GCC 9.3.0]
# Embedded file name: ./source.py
# Compiled at: 2020-08-03 20:55:47
# Size of source mod 2**32: 515 bytes
key = [
 115, 76, 50, 116, 90, 50, 116, 90, 115, 110, 48, 47, 87, 48, 103, 50, 106, 126, 90, 48, 103, 116, 126, 90, 85,
126, 115, 110, 105, 104, 35]
print('Input your flag: ', end='')
flag = input()
out = []
for i in flag:
    out.append(ord(i) >> 4 ^ ord(i))

if len(out) != len(key):
    print('TRY AGAIN!')
    exit()
for i in range(len(out)):
    if out[i] != key[i]:
        print('TRY AGAIN!')
        exit()

print('you are right! the flag is : moectf{%s}' % flag)
```

写出相应的解密脚本：

```
key = [
 115, 76, 50, 116, 90, 50, 116, 90, 115, 110, 48, 47, 87, 48, 103, 50, 106, 126, 90, 48, 103, 116, 126, 90, 85,
126, 115, 110, 105, 104, 35]

out1 = ''
for i in key:
    ii = i
    out1 = out1 + chr(ii >> 4 ^ ii)

print(out1)
```

得到flag：

```
moectf{tH1s_1s_th3-R3a1ly_3asy_Python!}
```

关于python编译的原理，参考：（*这个找不到最初的原创作者了...侵删*）

> **1. Python是一门解释型语言？**
> 我初学Python时，听到的关于Python的第一句话就是，Python是一门解释性语言，我就这样一直相信下去，直到发现了.pyc文件的存在。
> 如果是解释型语言，那么生成的.pyc文件是什么呢？c应该是compiled的缩写才对啊！
> 为了防止其他学习Python的人也被这句话误解，那么我们就在文中来澄清下这个问题，并且把一些基础概念给理清。
> **2. 解释型语言和编译型语言**
> 计算机是不能够识别高级语言的，所以当我们运行一个高级语言程序的时候，就需要一个“翻译机”来从事把高级语言转变成计算机能读懂的机器语言的过程。这个过程分成两类，第一种是编译，第二种是解释。
> 编译型语言在程序执行之前，先会通过编译器对程序执行一个编译的过程，把程序转变成机器语言。运行时就不需要翻译，而直接执行就可以了。最典型的例子就是C语言。
> 解释型语言就没有这个编译的过程，而是在程序运行的时候，通过解释器对程序逐行作出解释，然后直接运行，最典型的例子是Ruby。
> 通过以上的例子，我们可以来总结一下解释型语言和编译型语言的优缺点，因为编译型语言在程序运行之前就已经对程序做出了“翻译”，所以在运行时就少掉了“翻译”的过程，所以效率比较高。但是我们也不能一概而论，一些解释型语言也可以通过解释器的优化来在对程序做出翻译时对整个程序做出优化，从而在效率上超过编译型语言。
> 此外，随着Java等基于虚拟机的语言的兴起，我们又不能把语言纯粹地分成解释型和编译型这两种。
> 用Java来举例，Java首先是通过编译器编译成字节码文件，然后在运行时通过解释器给解释成机器文件。所以我们说Java是一种先编译后解释的语言。
> **3. Python到底是什么**
> 其实Python和Java/C#一样，也是一门基于虚拟机的语言，我们先来从表面上简单地了解一下Python程序的运行过程吧。
> 当我们在命令行中输入python hello.py时，其实是激活了Python的“解释器”，告诉“解释器”：你要开始工作了。可是在“解释”之前，其实执行的第一项工作和Java一样，是编译。
> 熟悉Java的同学可以想一下我们在命令行中如何执行一个Java的程序：
> `javac hello.java`
> `java hello`
> 只是我们在用Eclipse之类的IDE时，将这两部给融合成了一部而已。其实Python也一样，当我们执行python hello.py时，他也一样执行了这么一个过程，所以我们应该这样来描述Python，Python是一门先编译后解释的语言。
> **4. 简述Python的运行过程**
> 在说这个问题之前，我们先来说两个概念，PyCodeObject和pyc文件。
> 我们在硬盘上看到的pyc自然不必多说，而其实PyCodeObject则是Python编译器真正编译成的结果。我们先简单知道就可以了，继续向下看。
> 当python程序运行时，编译的结果则是保存在位于内存中的PyCodeObject中，当Python程序运行结束时，Python解释器则将PyCodeObject写回到pyc文件中。
> 当python程序第二次运行时，首先程序会在硬盘中寻找pyc文件，如果找到，则直接载入，否则就重复上面的过程。
> 所以我们应该这样来定位PyCodeObject和pyc文件，我们说pyc文件其实是PyCodeObject的一种持久化保存方式。

亦可参考官方文档

RxEncode

> 150points
> Reverier为了防止软件被轻易破解, 就费尽心思写了一个密码验证程序. 可这么简单的程序真的有用嘛? 作为逆向大神的你应该可以很轻易的破解这个问题.

~~不会做~~时间不够，没来得及做

等~~官方题解出来~~做出来了再更叭

`_(:з」∠)_`

EasyCommonLISP

> 150points
> Reverier买了两个脚踏板.
> 当luoqi@n问他要做什么的时候, 他回答道: `"编程啊"`
>
> - 如何运行:
>
> `Linux x86_64`环境下安装`clisp`, 运行命令

```
clisp ./puzzle.lisp
```

首先看题：

```
(defparameter +alphabet+"AB#DEd@f&hi!klmnLMw3^5678N}PF|HIxyz012JKYZab%Q{S(UVWX-pqrs")(defparameter +len+(length
+alphabet+))(defun divmod(number divisor)(values(floor(/ number divisor))(mod number divisor)))(defun encode(str
)(let((value 0)(rstr(reverse str))(output(make-string-output-stream))(npad 0))(loop for i from 0 to(1- (length s
tr))do(setf value(+ value(*(char-code(elt rstr i))(expt 256 i))))))(loop while(>= value +len+)do(multiple-value-b
ind(new-value mod)(divmod value +len+)(setf value new-value)(write-char(elt +alphabet+ mod) output)))(write-char
(elt +alphabet+ value)output)(loop for char across str do(if(char-equal char #\Nul)(incf npad)(return)))(concate
nate 'string(coerce(loop for i from 1 to npad collecting #\1)'string)(reverse(get-output-stream-string output)))
))(print(encode "moectf{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}"))

;;;; eof
;;;; flag is "&Dx16Y!x3((xYDlShWbQ5hmzWf3EZly6h8UwD#d-1-&#WlDHJaxM5qAzlPP"
```

errr又是一行的程序

稍微格式化一下：

```
(defparameter +alphabet+"AB#DEd@f&hi!klmnLMw3^5678N}PF|HIxyz012JKYZab%Q{S(UVWX-pqrs")
(defparameter +len+(length +alphabet+))
(defun divmod(number divisor)(values(floor(/ number divisor))(mod number divisor)))
(defun encode(str)
    (let ((value 0)(rstr (reverse str ))( output ( make-string-output-stream ) )( npad 0))
        (loop for i from 0 to(1- (length str)) do
            (setf value(+ value(*(char-code(elt rstr i))(expt 256 i))))
        )
        (loop while(>= value +len+)do
            (multiple-value-bind(new-value mod)(divmod value +len+)(setf value new-value)(write-char(elt +alphab
et+ mod) output))
        )
        (write-char(elt +alphabet+ value)output)
        (loop for char across str do
            (if(char-equal char #\Nul)(incf npad)(return))
        )
        (concatenate 'string(coerce(loop for i from 1 to npad collecting #\1)'string)
            (reverse(get-output-stream-string output))
        )
    )
)
(print(encode "moectf{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}"))

;;;; eof
;;;; flag is "&Dx16Y!x3((xYDlShWbQ5hmzWf3EZly6h8UwD#d-1-&#WlDHJaxM5qAzlPP"
```

errrrrrr果然是“上帝的语言”

~~突然好想学lisp啊，其他语言的各种语法结构在lisp竟然用括号完成了大一统 (›´ω`‹) 真 是 太 美 妙 啦~~

既然是上帝的语言那咱凡人岂能读懂，于是我写了一个脚本（嘿嘿嘿）：

```
import os

oFlag = 'woO0Oow_Y0u-ar3_th3_g0D_0f_LIIIISP!}"))'   //手撕部分
for Ichr in range(35,127):
    lisp = r"""
    (defparameter +alphabet+"AB#DEd@f&hi!klmnLMw3^5678N}PF|HIxyz012JKYZab%Q{S(UVWX-pqrs")
    (defparameter +len+(length +alphabet+))
    (defun divmod(number divisor)(values(floor(/ number divisor))(mod number divisor)))
    (defun encode(str)
        (let ((value 0)(rstr (reverse str ))( output ( make-string-output-stream ) )( npad 0))
            (loop for i from 0 to(1- (length str)) do (setf value(+ value(*(char-code(elt rstr i))(expt 256 i)))
))
            (loop while(>= value +len+)do(multiple-value-bind(new-value mod)(divmod value +len+)(setf value new-
value)(write-char(elt +alphabet+ mod) output)))
            (write-char(elt +alphabet+ value)output)
            (loop for char across str do(if(char-equal char #\Nul)(incf npad)(return)))
            (concatenate 'string(coerce(loop for i from 1 to npad collecting #\1)'string)(reverse(get-output-str
eam-string output)))
        )
    )
    (print(encode "moectf{"""

    charE = chr(Ichr) #32~126
    IcharE = 34

    o = list(oFlag)
    o[IcharE] = charE
    oFlag = ''.join(o)

    lisp = lisp + oFlag

    f_write = open("hackfile.lisp", mode='w')
    f_write.writelines(lisp)
    f_write.close()

    flag = "&Dx16Y!x3((xYDlShWbQ5hmzWf3EZly6h8UwD#d-1-&#WlDHJaxM5qAzlPP"
    val = os.popen('clisp hackfile.lisp')
    val.readline()
    out = val.readline()
    out = out[1:-2]
    for i in range(len(flag)):
        # print(out[i] , flag[i])
        if(out[i] != flag[i]):
            break
    print(i,oFlag)
```

本来就是破方法就不详细讲吧，大概原理就是我运行lisp加密flag的时候返现密文是逐节加密的，但相邻的字符还是会对加密结果产生一点影响，于是可以通过深度优先/广度优先搜索的方法解出原文，但又懒，就稍微写一下，然后手撕深度优先搜索了，加上flag是有意义的字符串，手撕也蛮方便。

flag：`moectf{woO0Oow_Y0u-ar3_th3_g0D_0f_LIIIISP!}`

此部分官方writeup出来后还会更新

EzJava

> 200points
> 给大佬递咖啡~
>
> - 链接:
>
> Java是什么? 能喝嘛?
> 能喝 不能喝. Java是一门编程语言, 参考: 链接
> Java有什么专用的逆向工具?

最近在上Java课，老师要求我们用`Eclipse`作为IDE，于是我在其中安装了`JD-Eclipse`插件进行反编译，可参考这个,当然其他的方法还有很多。

反编译得到源码如下：

```java
import java.io.PrintStream;

public class EasyJava {
  public static void main(String[] paramArrayOfString) {
    System.out.println("MoeCTF 2020 EasyJava --by Reverier");
    System.out.println("Input your flag and I will check it:");
    java.io.BufferedReader localBufferedReader = new java.io.BufferedReader(new java.io.InputStreamReader(System
.in));
    String str1 = null;
    int[] arrayOfInt = { 43, 23, 23, 62, 110, 66, 94, 99, 126, 68, 43, 62, 76, 110, 22, 5, 15, 111, 86, 75, 78,
83, 86, 0, 85, 86 };
    try
    {
      str1 = localBufferedReader.readLine();
    } catch (Exception localException) {
      System.out.println("ERROR: Undefined Exception.");
    }
    if (str1.isEmpty()) {
      System.out.println("Nothing received.");
    } else { if (str1.length() != 35) {
        System.out.println("Rua~~~Wrong!");
        return;
      }
      String str2 = str1.substring(0, 7);
      if (!str2.equals("moectf{")) {
        System.out.println("Rua~~~Wrong!");
        return;
      }
      String str3 = str1.substring(7, str1.length() - 1);
      for (int i = 0; i < str3.length() - 1; i++) {
        int j = str3.charAt(i);
        int k = str3.charAt(i + 1);
        int m = j ^ k;
        if (m != arrayOfInt[i]) {
          System.out.println("Rua~~~Wrong!");
          return;
        }
      }
      System.out.println("Congratulations!");
    }
  }
}

/* Location:           D:\code\Javastudio\tes
 * Qualified Name:     EasyJava
 * Java Class Version: 13 (57.0)
 * JD-Core Version:    0.7.1
 */
```

编写相应的解密脚本如下：

```
"""
35==moectf{xxxxxxxxxxxxx}
xxxxxxxxxx==35-8=27
int[] arrayOfInt = { 43, 23, 23, 62, 110, 66, 94, 99, 126, 68, 43, 62, 76, 110, 22, 5, 15, 111, 86, 75, 78, 83,
86, 0, 85, 86 };
  String str3 = str1.substring(7, str1.length() - 1);
/* 28 */       for (int i = 0; i < str3.length() - 1; i++) {
/* 29 */         int j = str3.charAt(i);
/* 30 */         int k = str3.charAt(i + 1);
/* 31 */         int m = j ^ k;
/* 32 */         if (m != arrayOfInt[i]) {
/* 33 */           System.out.println("Rua~~~Wrong!");
/* 34 */           return;
"""
import string

def findit(l,r):
    if (r-l == 1):
        ans = []
        for i in dic:
            for j in dic:
                try:
                    if i ^ j == arrayOfInt[l]:
                        ans.append(bytes(chr(i)+chr(j),encoding='utf-8'))
                except:
                    print(l)
        return ans
    if(r-1 == 0):
        ans = []
        for i in dic:
            ans.append(bytes(chr(i),encoding='utf-8'))
        return ans
    s1 = findit(l,r-2)
    s2 = findit(r-1,r)
    ans = []
    for ans1 in s1:
        for ans2 in s2:
            if (arrayOfInt[r-2] == ans1[-1] ^ ans2[0]):
                ans.append(ans1+ans2)
    return ans
arrayOfInt = [ 43, 23, 23, 62, 110, 66, 94, 99, 126, 68, 43, 62, 76, 110, 22, 5, 15, 111, 86, 75, 78, 83, 86, 0,
 85, 86 ]
print(len(arrayOfInt))
dic = bytes( string.printable,encoding='utf-8')
print(dic)
print(findit(1,26))

# [b'ava_1s-N0t_a-CUP_0f-c0ff3e', b'bub\\2p.M3w\\b.@VS\\3e.`3ee0f', b'ctc]3q/L2v]c/AWR]2d/a2dd1g', b'dsdZ4v(K5qZ
d(FPUZ5c(f5cc6`', b'ere[5w)J4p[e)GQT[4b)g4bb7a', b'fqfX6t*I7sXf*DRWX7a*d7aa4b', b'gpgY7u+H6rYg+ESVY6`+e6``5c', b
'i~iW9{%F8|Wi%K]XW8n%k8nn;m', b"k|kU;y'D:~Uk'I_ZU:l'i:ll9o", b'l{lR<~ C=yRl NX]R=k n=kk>h', b'nynP>|"A?{Pn"LZ_P?
i"l?ii<j', b'oxoQ?}#@>zQo#M[^Q>h#m>hh=k', b'pgpN b<_!eNp<RDAN!w<r!ww"t', b'qfqO!c=^ dOq=SE@O v=s vv#u', b'rerL"`
>]#gLr>PFCL#u>p#uu v', b'sdsM#a?\\"fMs?QGBM"t?q"tt!w', b'tctJ$f8[%aJt8V@EJ%s8v%ss&p', b"ubuK%g9Z$`Ku9WADK$r9w$rr
'q", b"vavH&d:Y'cHv:TBGH'q:t'qq$r", b"w`wI'e;X&bIw;UCFI&p;u&pp%s", b'ynyG)k5V(lGy5[MHG(~5{(~~+}', b'zmzD*h6U+oDz
6XNKD+}6x+}}(~', b'`w`^0r,O1u^`,BTQ^1g,b1gg2d', b'|k|B,n0S-iB|0^HMB-{0~-{{.x', b'~i~@.l2Q/k@~2\\JO@/y2|/yy,z']
# 'moectf{Java_1s-N0t_a-CUP_0f-c0ff3e}'
```

注：这里我还傻傻地写了一个巨丑无比的递归来解，列表的边界处理还有一点问题没修。写完了才发现根据异或的对称性是可以直接解密的 `_(:τ」∠)_`

不管啦，有flag就行：`moectf{Java_1s-N0t_a-CUP_0f-c0ff3e}`

> ohhhh我会Jvav啦！

RollCall

> 200points
> 有时候啊, 眼光不能拘泥于一点.
> 在软件设置中把任意一个学生的性别改成 `2`, 保存后重启软件即可获取 `flag`.
> 如果你是通过逆向主程序得到了 `flag`, 请务必联系管理员加分, 顺便接受膜拜.

下载程序打开后是这样的：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-2FkmqKOZ-1602257797667)(【moeCTF题解-0x01】Reverse/image-20201009152004496.png)]

打开`软件设置`，可以修改性别，但只能输入0和1：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-Jkf6vStv-1602257797668)(【moeCTF题解-0x01】Reverse/image-20201009152254678.png)]

看着这~~花里胡哨的~~界面，直觉不好反编译，于是观察该程序所在的文件夹，看看有没有数据库这种东西：

```
.
├── CSkin.dll
├── datedata
├── EntityFramework.dll
├── EntityFramework.SqlServer.dll
├── number
├── RandomNames13.0.exe
├── SQLite-net.dll
├── SQLitePCLRaw.batteries_green.dll
├── SQLitePCLRaw.batteries_v2.dll
├── SQLitePCLRaw.core.dll
├── SQLitePCLRaw.provider.e_sqlite3.dll
├── System.Data.SQLite.dll
├── System.Data.SQLite.EF6.dll
├── System.Data.SQLite.Linq.dll
├── UserData.sqlite                 # <-- 好耶，就是这个
├── x64
│   ├── e_sqlite3.dll
│   ├── sqlite3.dll
│   └── SQLite.Interop.dll
└── x86
    ├── e_sqlite3.dll
    ├── sqlite3.dll
    └── SQLite.Interop.dll
```

是sqlite数据库，Windows下有一个可视化的工具`SQLiteSpy`可以方便编辑它

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-nLDIvhJ9-1602257797669)(【moeCTF题解-0x01】Reverse/153003.png)]

随便修改一个用户的性别信息再打开，跳出flag

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-lG75iu4W-1602257797671)(【moeCTF题解-0x01】Reverse/image-20201009153132456.png)]

> 不能得到加分和接受膜拜惹 `(o´・ェ・`o)`

【moeCTF题解-0x01】Android

Baby Android

> 100points
> 被Rx大哥的逆向题整哭了？ 快来看看友好的android逆向！

用虚拟机打开发现是一个flag验证程序：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-5L1lfCSn-1602257797671)(【moeCTF题解-0x01】Reverse/image-20201009214539242.png)]

使用JEB，定位到com-example-MainActivity，按`TAB`解析，得到下图：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-Suf6lnmw-1602257797672)(【moeCTF题解-0x01】Reverse/image-20201009214722330.png)]

根据加密函数写出解密脚本：

```
# public MainActivity() {
#         super();
#         this.key = "dalaomenbiedale";
#         this.secret = new byte[]{9, 14, 9, 2, 27, 11, 30, 55, 82, 28, 58, 15, 15, 92, 18, 59, 33, 34, 5, 29, 2
, 84, 10, 61, 11, 86, 16, 21, 0x5F, 23, 59, 53, 4, 82, 1, 50, 40, 11, 67, 20};
#     }
#     public boolean checkFlag(byte[] arg7) {
#         int v1 = 40;
#         if(arg7.length == v1) {
#             int v0;
#             for(v0 = 0; v0 < v1; ++v0) {
#                 if((arg7[v0] ^ this.key.getBytes()[v0 % this.key.length()]) != this.secret[v0]) {
#                     return 0;
#                 }
#             }
#             return 1;
#         }
#         return 0;
#     }
import base64
import string

def chackFlag(arg,v0):

    key = b"dalaomenbiedale"
    secret = [9, 14, 9, 2, 27, 11, 30, 55, 82, 28, 58, 15, 15, 92, 18, 59, 33, 34, 5, 29, 2, 84, 10, 61, 11, 86,
 16, 21, 0x5F, 23, 59, 53, 4, 82, 1, 50, 40, 11, 67, 20]

    if((ord(arg) ^ key[v0 % len(key)]) != secret[v0]) :
        return 0
    return 1


dic = string.printable     #各种打印字符
print(dic)
flag = ''
for i in range(0,40):
    for j in dic :
        if(chackFlag(j,i)==1):
            flag = flag + j

print(flag)

# moectf{Y0u_kn0w_@Ndro1d_b3tt3r_Th3n_Me!}
```

这里同样可以利用异或的对称性简化解密，我这里没有出。

运行脚本得到flag：`moectf{Y0u_kn0w_@Ndro1d_b3tt3r_Th3n_Me!}`

Click It!

> 200points
> 动次打次

0x1 在虚拟机里打开：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-wv14fA6r-1602257797672)(【moeCTF题解-0x01】Reverse/image-20201009220237608.png)]

得点1919810下啊（这个数字好奇怪：3

用连点器试一下，突然发现虚拟机变得巨卡无比，放弃

接着尝试使用`JBE`，逆向失败惹...

参考安卓apk反编译、修改、重新打包、签名全过程再次进行尝试，发现在逆向出的源码里很难找关键信息，在汇编程序smali文件中倒是能找到1919810相关的代码，于是尝试直接修改汇编程序中的1919810为1，再打包运行程序获取flag。

具体过程如下：

0x2 用`apktool`解包apk文件

```
java -jar .\apktool_2.4.1.jar -r d .\click.apk -o out
```

这里参考了这篇文章以解决apktool重打包失败的问题

0x3 在smali文件里查找

在smali文件里查找1919810的16进制0x1d4b42，定位到如下位置：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-DyDxbxRh-1602257797673)(【moeCTF题解-0x01】Reverse/image-20201009223728371.png)]

0x4修改0x1d4b42为0x1

这亚只要点一次就可以获取到flag啦

0x5用`apktool`重新打包

```
java -jar .\apktool_2.4.1.jar b out
```

打包的apk生成在out文件夹下的dist文件夹里

0x6 获取一个签名

如果在安卓虚拟机中安装，则提示缺少签名安装失败，于是需要对apk进行签名

> 签名是对要发布的apk文件作标记，确保你的apk文件有唯一的身份归属认证，只有相同签名和相同包名的文件才可以覆盖安装并保留用户信息。

`JDK`自带的签名工具`keytool`和`jarsigner`可以很方便的对apk进行签名

> 首先，签名需要keystore文件，可以使用keytool工具生成，一般Java环境都带有keytool命令，可以在命令行测试。
> 各个参数解释如下：
> -genkey 产生证书文件
> -alias 产生别名
> -keystore 指定密钥库的.keystore文件中
> -keyalg 指定密钥的算法,这里指定为RSA(非对称密钥算法)
> -validity 为证书有效天数，这里我们写的是40000天

shell中执行

```
keytool -genkey -alias demo.keystore -keyalg RSA -validity 40000 -keystore demo.keystore
```

输入上述命令后，会有如下的提示和输入：

```
输入密钥库口令:
再次输入新口令:123456
您的名字与姓氏是什么?
  [Unknown]:  test
您的组织单位名称是什么?
  [Unknown]:  test
您的组织名称是什么?
  [Unknown]:  test
您所在的城市或区域名称是什么?
  [Unknown]:  test
您所在的省/市/自治区名称是什么?
  [Unknown]:  test
该单位的双字母国家/地区代码是什么?
  [Unknown]:  test
CN=test, OU=test, O=test, L=test, ST=test, C=test是否正确?
  [否]:  y

正在为以下对象生成 2,048 位RSA密钥对和自签名证书 (SHA256withRSA) (有效期为 40,000 天):
         CN=test, OU=test, O=test, L=test, ST=test, C=test
```

于是我们获取了一个签名。

0x7 对apk签上签名

> `jarsigner` 也存在于Java JDK的安装包当中，所以安装好了Java环境的话，可以直接在命令行使用。
> `-verbose` 指定生成详细输出
> `-keystore` 指定数字证书存储路径
> 这样，就完成了对一个apk的签名过程，然后就可以安装使用了。注意如果你的手机上原来就有这个apk，需要卸载掉。因为新apk的签名已经改变了。

shell运行：

```
jarsigner -verbose -keystore demo.keystore .\out\dist\click.apk demo.keystore
```

获得如下输出：

```
......
>>> 签名者
    X.509, CN=test, OU=test, O=test, L=test, ST=test, C=test
    [可信证书]

jar 已签名。

警告:
签名者证书为自签名证书。
```

0x8 重新运行修改过的APK!

现在就可以安装运行啦：

[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传(img-3xN2YlUB-1602257797674)(【moeCTF题解-0x01】Reverse/image-20201009220934604.png)]

得到flag：`moectf{y0U_w1n!}`

【moeCTF题解-0x01】IoT

未完待续……