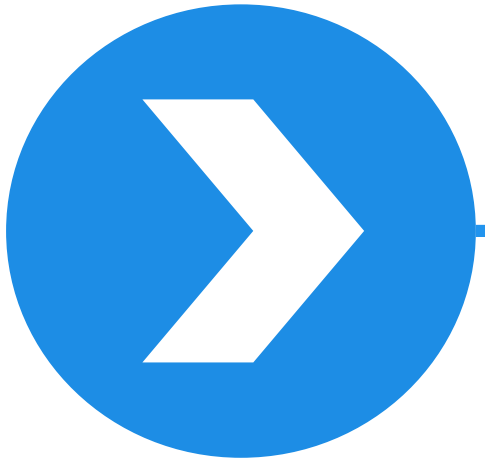
北京网信办网站安全比赛(2017)题解

安天 李柏松



第一轮

- PE结构
- 加密算法
- 流量分析
- 格式识别

1.基础知识题

- PE文件结构是逆向分析中不可或缺的知识，现在以PE文件ATScanner.exe为例进行填写。
- 此文件IMAGE_NT_HEADERS结构体由三个成员组成，第一个成员为签名结构体，其值为00004550h，CPU常值为014Ch，块（节）数目的值为_____，文件时间戳值为_____，文件入口点地址为_____，用于文件信息标识（文件头属性特征）的值为_____。

1.基础知识题


```
000000F8   50 45 00 00  ASCII "PE"          PE signature (PE)
000000FC   4C01         DW 014C             Machine = IMAGE_FILE_MACHINE_I386
000000FE   0400         DW 0004              NumberOfSections = 4
00000100   CC5C1D59     DD 591D5CCC          TimeDateStamp = 591D5CCC
00000104   00000000     DD 00000000          PointerToSymbolTable = 0
00000108   00000000     DD 00000000          NumberOfSymbols = 0
0000010C   E000         DW 00E0              SizeOfOptionalHeader = E0 (224.)
0000010E   0301         DW 0103              Characteristics = EXECUTABLE_IMAGE|32BIT_MACHINE|RELOCS_STRIPPED
00000110   0B01         DW 010B             MagicNumber = PE32
00000112   08           DB 08                MajorLinkerVersion = 8
00000113   00           DB 00                MinorLinkerVersion = 0
00000114   00200800     DD 00082000          SizeOfCode = 82000 (532480.)
00000118   00200400     DD 00042000          SizeOfInitializedData = 42000 (270336.)
0000011C   00000000     DD 00000000          SizeOfUninitializedData = 0
00000120   89B70500     DD 0005B789          AddressOfEntryPoint = 5B789
00000124   00100000     DD 00001000          BaseOfCode = 1000
00000128   00300800     DD 00083000          BaseOfData = 83000
```

2.简单的加密分析

题目分值	15分
待分析文件名称	Warmup.exe_85B593CC6E0A1826109FD10716D63C76
待分析文件MD5值	85B593CC6E0A1826109FD10716D63C76
运行测试环境	Windows XP、Windows 7
考察的技术点	Windows平台、异或算法、x86汇编
题目描述	当输入FLAG之后，会出现明确的提示信息，所输入的字符串即为所要提交的FLAG。
备注	提交正确答案即获得相应的分数

```
地址      Hex 转存          反汇编                                  注释
004010FC  > BE E0404100     mov esi,Warmup.004140E0                 ldyvlqmzhuy:|cq[^qyo|cq{^qyo\cq[^/s
00401101  .  8D4424 20      lea eax,dword ptr ss:[esp+20]
00401105  >  8A10           mov dl,byte ptr ds:[eax]
00401107  .  8ACA           mov cl,dl
00401109  .  3A16           cmp dl,byte ptr ds:[esi]
0040110B  .v 75 1C          jnz short Warmup.00401129
0040110D  .  3ACB           cmp cl,bl
0040110F  .v 74 14          je short Warmup.00401125
00401111  .  8A50 01        mov dl,byte ptr ds:[eax+1]
00401114  .  8ACA           mov cl,dl
00401116  .  3A56 01        cmp dl,byte ptr ds:[esi+1]
00401119  .v 75 0E          jnz short Warmup.00401129
0040111B  .  83C0 02        add eax,2
```

2.简单的加密分析


```
.data:004140B0 ; char aPause[]
.data:004140B0 aPause          db 'pause',0            ; DATA XREF: _main:loc_4011F3↑o
.data:004140B6                 align 4
.data:004140B8 aSorryTryAgain_ db 'Sorry! Try again.',0 ; DATA XREF: _main:loc_40118F↑o
.data:004140CA                 align 4
.data:004140CC aContratulation db 'Contratulations!',0 ; DATA XREF: _main+132↑o
.data:004140DD                 align 10h
.data:004140E0 aLdyvlqmzhuyCqQ db 'LDYVLQMZHuY:|cQ[^Qyo|cQ{~QYO\CQ[^/s',0
.data:004140E0                                         ; DATA XREF: _main:loc_4010FC↑o
.data:00414104 aPleaseInputThe db 'Please Input the Flag: ',0 ; DATA XREF: _main+1E↑o
```

```
memset(&v18, 0, 0x100u);
v19 = 0;
v21 = 0;
v20 = 0;
sub_401990((int)&dword_417D88, &v17);
v6 = 0;
v7 = strlen(&v17) + 1;
if ( (signed int)(v7 - 1) > 0 )
{
  do
  {
    *(&v17 + v6) ^= 0xEu;
    ++v6;
  }
  while ( v6 < (signed int)(v7 - 1) );
}
if ( !strcmp(&v17, aLdyvlqmzhuyCqQ) )
{
  v8 = sub_401700(&unk_417CF8, aContratulation);
  sub_401280(10);
```

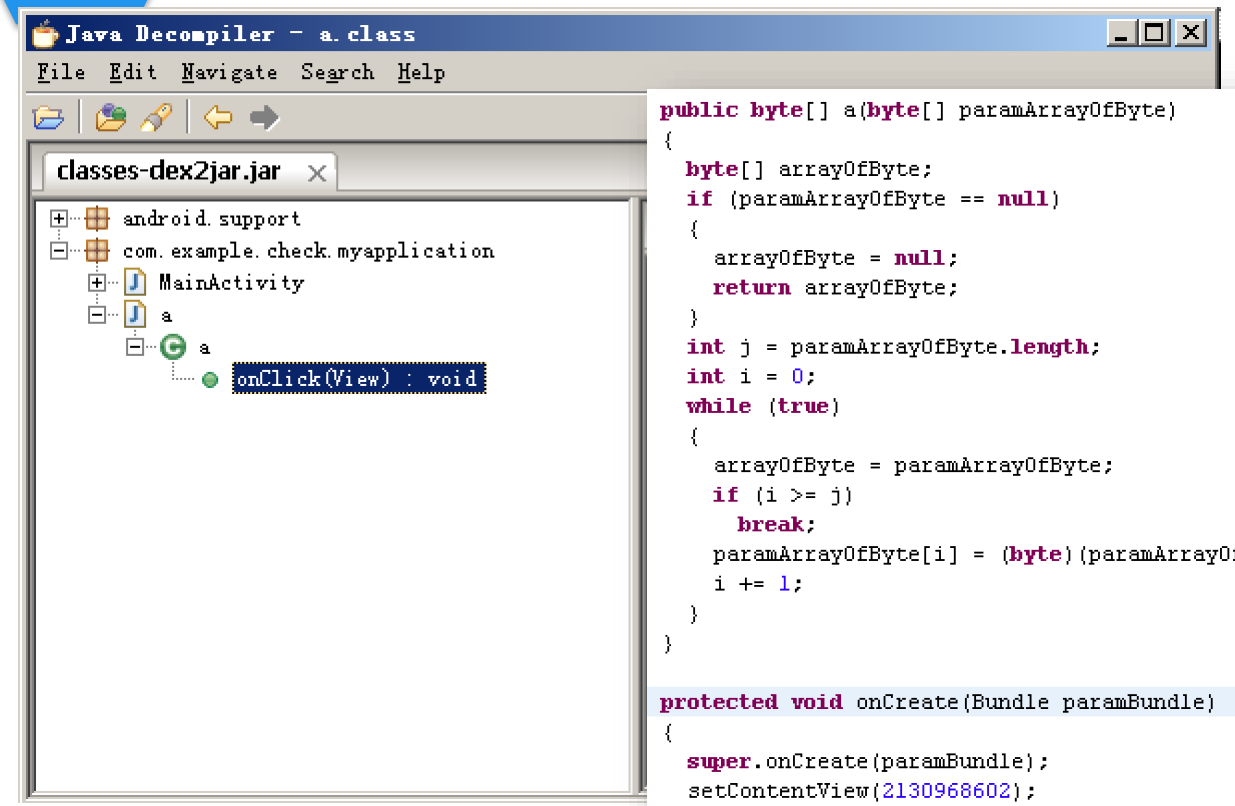
```
Python 2.7.10 (default, Jul 30 2016, 18:31:42)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.34)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> for x in 'LDYVLQMZHuY:|cQ[^Qyo|cQ{~QYO\CQ[^/s': print chr(ord(x)^ 0x0e),
...
B J W X B _ C T F { W 4 r m _ U P _ w a r m _ u p _ W A R M _ U P ! }
>>>
```

3.简单的Android程序逆向分析

题目分值	15分
待分析文件名称	app-release.apk_C25A37D991E40EC2371285093BB4E78F
待分析文件MD5值	C25A37D991E40EC2371285093BB4E78F
运行测试环境	Android 7.0
考察的技术点	Android平台、异或算法、Smali语法
题目描述	当输入FLAG之后，会出现明确的提示信息，所输入的字符串即为所要提交的FLAG。
备注	提交正确答案即获得相应的分数

使用Android逆向工具（dex2jar+Java Decompiler），得到源代码，查看字符串，发现有明确的提示信息，如“Contratulations!”与“Sorry! Try again”。查看它们的调用，即可找到FLAG比较的位置，向上找，就可以看到异或算法（与0x12做xor运算）。再次XOR，即可得到正确的FLAG。

3.简单的Android程序逆向分析




```
public byte[] a(byte[] paramArrayOfByte)
{
  byte[] arrayOfByte;
  if (paramArrayOfByte == null)
  {
    arrayOfByte = null;
    return arrayOfByte;
  }
  int j = paramArrayOfByte.length;
  int i = 0;
  while (true)
  {
    arrayOfByte = paramArrayOfByte;
    if (i >= j)
      break;
    paramArrayOfByte[i] = (byte)(paramArrayOfByte[i] ^ 0x12);
    i += 1;
  }
}

protected void onCreate(Bundle paramBundle)
{
  super.onCreate(paramBundle);
  setContentView(2130968602);
  this.m = ((Button)findViewById(2131427413));
```

```
public void onClick(View paramView)
{
  MainActivity.a(this.a, "PXEJPMQFTiS|v`\"#vMDw`KMA3_b~w3o");
  MainActivity.a(this.a, (EditText)this.a.findViewById(2131427412));
  MainActivity.b(this.a, MainActivity.a(this.a).getText().toString());
  paramView = new String(this.a.a(MainActivity.b(this.a).getBytes()));
  Log.i(paramView, MainActivity.b(this.a));
  t localt = new t(this.a);
  if (paramView.equals(MainActivity.c(this.a)) == true)
  {
    localt.a("提示");
    localt.b("Contratulations! You Got Correct Flag.");
    localt.c();
    return;
  }
  localt.a("提示");
  localt.b("Sorry! You Got Error Flag.");
  localt.c();
}
```

```
>>> for x in 'PXEJPMQFTiS|v`\"#vMDw`KMA3_b~w3o': print chr(ord(x)^ 0x12),
...
B J W X B _ C T F { A n d r 0 1 d _ V e r Y _ S ! M p l e ! }
>>>
```

4.采用VM技术的简单加密程序分析

题目分值	25分
待分析文件名称	CrackMe.exe_DE1901B00C7D3C1020635723C24D4601
待分析文件MD5值	DE1901B00C7D3C1020635723C24D4601
运行测试环境	Windows XP、Windows 7（命令行测试）
考察的技术点	虚拟机指令、算法、
题目描述	当输入FLAG之后，会出现明确的提示信息，所输入的字符串即为所要提交的FLAG。
备注	提交正确答案即获得相应的分数

- 程序基于VM实现，加密代码由VM指令编写，无加壳和反跟踪。
- VM是用软件实现的哈佛结构（代码存储器与数据存储器分开）
  - CPU： 32位
  - 字长：32位
  - 指令存储器地址32位
  - 数据存储器地址32位
  - 指令长度32位
  - 基于堆栈结构的虚拟 CPU，堆栈最大长度32个操作数，操作数从右向左入栈（类似JVM或.NET CLR虚拟机）
- 参考书：大学教材《计算机组成原理》

4.采用VM技术的简单加密程序分析


```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  printf(aCrackMe);
  printf(aEnter);
  sub_401030((int)&word_408254, (int)&unk_408030);
  return 0;
}
```

```
dword_408254    dd      104h,     118h,     305h,     104h,       0
                                        ; DATA XREF: _main+19↑o
                dd      104h,       0,      404h,     104h,      2Fh
                dd      104h,       0,      304h,     201h,     303h
                dd      0D4h,     104h,      18h,     104h,       0
                dd      304h,     101h,     504h,     104h,      10h
                dd      104h,       0,      304h,     104h,       7
                dd      102h,     101h,     504h,     402h,     104h
                dd      118h,     104h,       0,      304h,     101h
                dd      504h,     201h,     403h,     0D4h,     104h
                dd        0,      304h,     301h,     104h,       0
                dd      404h,     203h,      20h,     104h,      2Fh
                dd      104h,       0,      304h,     201h,     403h
                dd      104h,     104h,     218h,     405h,     103h
                dd      104h,     21Ch,     405h,     103h
; char aEnter[]
aEnter          db 'Enter: ',0          ; DATA XREF: _main+A↑o
```

4.采用VM技术的简单加密程序分析


```
sub_401390();
dword_40AC60 = 0;
while ( 1 )
{
  while ( 1 )
  {
    while ( 1 )
    {
LABEL_2:
      while ( 1 )
      {
        v2 = *(_DWORD *)(a1 + 4 * dword_40AC60);
        v3 = dword_40AC60++ + 1;
        if ( v2 > 0x201 )
          break;
        if ( v2 == 0x201 )
        {
          v19 = sub_4013E0();
          v7 = sub_4013E0();
          sub_4013A0(v19 - v7);
        }
```

; EIP ?

```
switch ( v2 )
{
  case 0x401u:
    v19 = sub_4013E0() - 1;
    sub_4013A0(v19);
    break;
  case 0x402u:
    v19 = sub_4013E0();
    v14 = sub_4013E0();
    sub_4013A0(v19 ^ v14);
    break;
  case 0x403u:
    v15 = *(_DWORD *)(a1 + 4 * v3);
    dword_40AC60 = v3 + 1;
    if ( sub_4013E0() )
      dword_40AC60 = v15 >> 2;
    break;
  case 0x404u:
```

; DEC?

; XOR

; JMP?

4.采用VM技术的简单加密程序分析

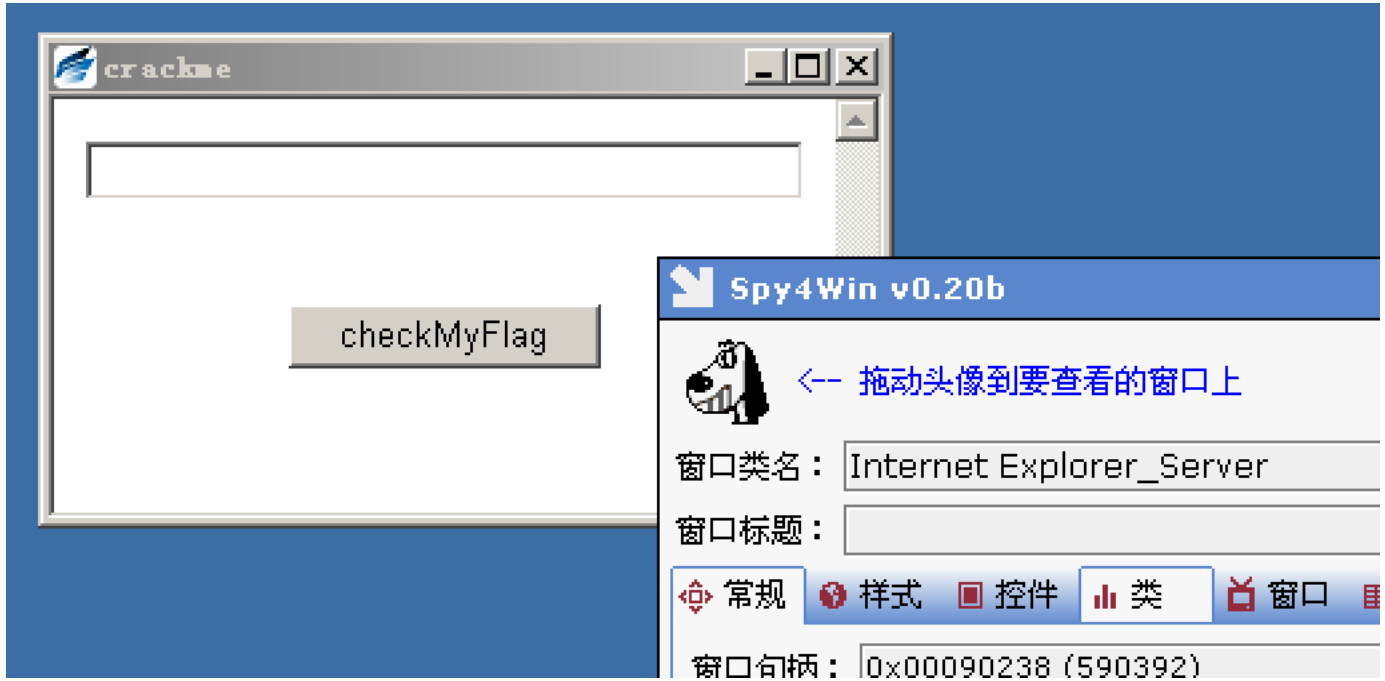

```
00408040 key            db 'AntiyLab'
00408048 ciphertext     db 3, 24h, 23h, 31h, 3Bh, 13h, 22h, 36h, 7, 15h, 37h, 5Ch; 0
00408048                db 4Ah, 7Ch, 56h, 26h, 75h, 58h, 59h, 2Ch, 4Eh, 7Ch, 24h; 0Ch
00408048                db 4Fh, 75h, 5Eh, 47h, 51h, 54h, 0Eh, 57h, 24h, 78h, 43h; 17h
00408048                db 4Ch, 2Ah, 4Ah, 0Ah, 57h, 5Bh, 79h, 2Ch, 43h, 2Ah, 4Ch; 22h
00408048                db 7Fh                  ; 2Dh
00408076                db  1Ch
00408077                db    0
```

```
swordlea@Sword_MBP Scripts$ more xor_antiy.py
i = 0
key='AntiyLab'
ciphertext='''\
\x03\x24\x23\x31\x3B\x13\x22\x36\x07\x15\x37\x5C\x4A\x7C\x56\x26\
\x75\x58\x59\x2C\x4E\x7C\x24\x4F\x75\x5E\x47\x51\x54\x0E\x57\x24\
\x78\x43\x4C\x2A\x4A\x0A\x57\x5B\x79\x2C\x43\x2A\x4C\x7F\x1C\
'''
for x in ciphertext:
    print chr(ord(x)^ord(key[i])),
    i += 1
    if i >= len(key): i = 0
swordlea@Sword_MBP Scripts$ python xor_antiy.py
B J W X B _ C T F { C 5 3 0 7 D 4 6 - E 7 0 E - 4 0 3 8 - B 6 F 9 - 8 C 3 F 6 9 8 B 7 C 5 3 }
swordlea@Sword_MBP Scripts$
```

5.加壳的Delphi 程序分析

题目分值	35分
待分析文件名称	check.exe_2AEA761866C1CE91CCB5AF8510C3C56C
待分析文件MD5值	2AEA761866C1CE91CCB5AF8510C3C56C
运行测试环境	Windows XP、Windows 7
考察的技术点	脱壳、javascript加密、混淆
题目描述	当输入FLAG之后，会出现明确的提示信息，所输入的字符串即为所要提交的FLAG。
备注	提交正确答案即获得相应的分数

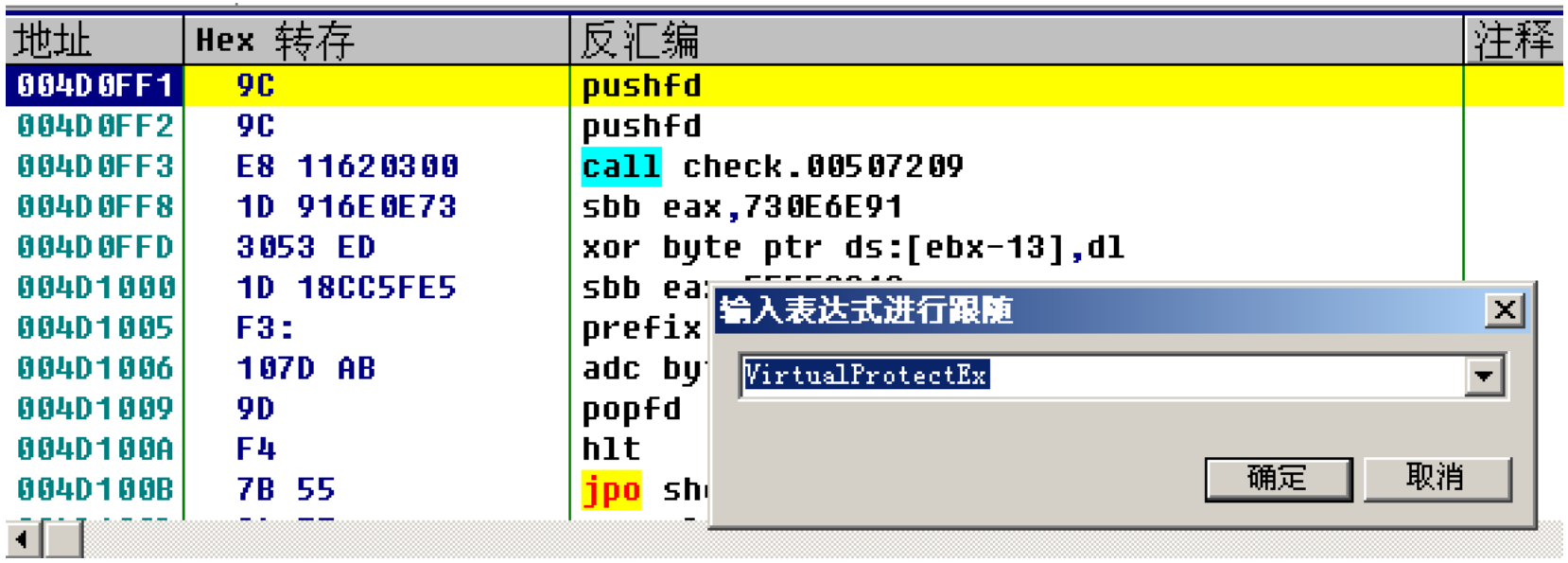
- 采用DELPHI语言开发
- 采用VMP加壳
- 使用混淆JS脚本判断输入

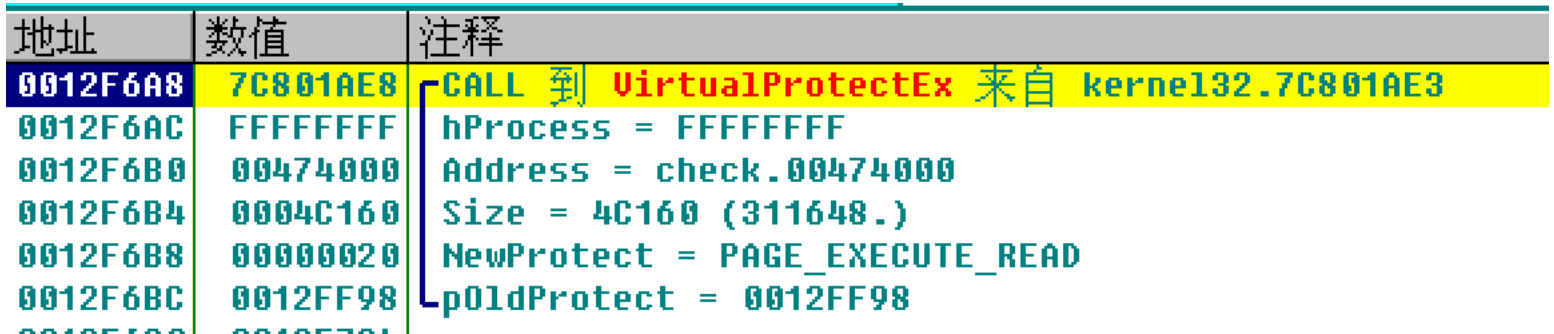


5.加壳的Delphi 程序分析

VMP 脱壳

OD + StrongOD ， 设置 VirtualProtectEx断点，执行约10次，看到PAGE_EXECUTE_READ属性时，使用Dump插件脱壳。脱壳后无法正常执行（需修复PE），但可使用十六进制编辑工具（如ＵＥ）查找脚本或使用IDA静态分析。





5.加壳的Delphi 程序分析

查找脚本

```
0005c080h: 00 66 7C 00 FF FF FF FF 08 00 00 00 3C 73 63 72 ; .f|.         ....<scr
0005c090h: 69 70 74 3E 00 00 00 00 01 01 00 48 77 72 69 74 ; ipt>.......Hwrit
0005c0a0h: 65 6C 6E 00 01 00 00 64 6F 63 75 6D 65 6E 74 00 ; eln....document.
0005c0b0h: FF FF FF FF 29 02 00 00 65 76 61 6C 28 66 75 6E ;          )...eval(fun
0005c0c0h: 63 74 69 6F 6E 28 70 2C 61 2C 63 2C 6B 2C 65 2C ; ction(p,a,c,k,e,
0005c0d0h: 64 29 7B 65 3D 66 75 6E 63 74 69 6F 6E 28 63 29 ; d){e=function(c)
0005c0e0h: 7B 72 65 74 75 72 6E 28 63 3C 61 3F 22 22 3A 65 ; {return(c<a?"":e
0005c0f0h: 28 70 61 72 73 65 49 6E 74 28 63 2F 61 29 29 29 ; (parseInt(c/a)))
0005c100h: 2B 28 28 63 3D 63 25 61 29 3E 33 35 3F 53 74 72 ; +((c=c%a)>35?Str
0005c110h: 69 6E 67 2E 66 72 6F 6D 43 68 61 72 43 6F 64 65 ; ing.fromCharCode
```

提取脚本

```
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;}('7 6(){a=8.b.9.5;
1(a=="4{3-2-h-i}"){0("j!")}k{0("g!<"+a+"> d c f e ;-)")}}',21,21,'alert|if|simpower91|by|BJWXB_CTF|value|ckpswd|
function|document|pswd||all|not|is|GUID|my|wrong|antiy|cn|congratulations|else'.split('|'),0,{}))
```

5.加壳的Delphi 程序分析

脚本解密

http://tools.jb51.net/password/evalencode

6. PCAP包分析

题目分值	45分
待分析文件名称	EveryBit.pcap_407571E2E762F532219DBBF4054CA237
待分析文件MD5值	407571E2E762F532219DBBF4054CA237
运行测试环境	Windows XP、Windows 7
考察的技术点	数据包分析、隐写数据分析、数据解密、脱壳、x86汇编
题目描述	当输入FLAG之后，会出现明确的提示信息，所输入的字符串即为所要提交的FLAG。
备注	提交正确答案即获得相应的分数

- 流量分析
- 格式解析
- 脱壳
- 解密

Id	Source	Destination	Captured Length	Packet Length	Protocol	Date Received	Time Delta	Information
1	192.168.201.200	36.104.135.172	62	62	TCP	2017-11-02 10:31:53.187	0.000000	1306 -> HTTP ([SYN], Seq=4167465913, Ack=0, Win=65535)
2	192.168.201.200	180.149.131.98	647	647	TCP	2017-11-02 10:31:53.189	0.001954	1300 -> HTTP ([ACK, PUSH], Seq=3413535336, Ack=777095285, Win...
3	180.149.131.98	192.168.201.200	60	60	TCP	2017-11-02 10:31:53.189	0.002476	HTTP -> 1300 ([ACK], Seq=777095285, Ack=3413535929, Win=64240)
4	180.149.131.98	192.168.201.200	752	752	TCP	2017-11-02 10:31:53.234	0.047540	HTTP -> 1300 ([ACK, PUSH], Seq=777095285, Ack=3413535929, Win...
5	180.149.131.98	192.168.201.200	1514	1514	TCP	2017-11-02 10:31:53.236	0.049349	HTTP -> 1300 ([ACK], Seq=777095983, Ack=3413535929, Win=64240)
6	180.149.131.98	192.168.201.200	1498	1498	TCP	2017-11-02 10:31:53.236	0.049368	HTTP -> 1300 ([ACK, PUSH], Seq=777097443, Ack=3413535929, Win...
7	192.168.201.200	180.149.131.98	54	54	TCP	2017-11-02 10:31:53.236	0.049387	1300 -> HTTP ([ACK], Seq=3413535929, Ack=777098887, Win=65535)
8	180.149.131.98	192.168.201.200	1506	1506	TCP	2017-11-02 10:31:53.237	0.049898	HTTP -> 1300 ([ACK, PUSH], Seq=777098887, Ack=3413535929, Win...
9	180.149.131.98	192.168.201.200	1506	1506	TCP	2017-11-02 10:31:53.237	0.049904	HTTP -> 1300 ([ACK, PUSH], Seq=777100339, Ack=3413535929, Win...
10	180.149.131.98	192.168.201.200	1506	1506	TCP	2017-11-02 10:31:53.237	0.049908	HTTP -> 1300 ([ACK, PUSH], Seq=777101791, Ack=3413535929, Win=...
11	192.168.201.200	180.149.131.98	54	54	TCP	2017-11-02 10:31:53.237	0.049921	1300 -> HTTP ([ACK], Seq=3413535929, Ack=777103243, Win=65535)
12	180.149.131.98	192.168.201.200	1514	1514	TCP	2017-11-02 10:31:53.237	0.050133	HTTP -> 1300 ([ACK], Seq=777103243, Ack=3413535929, Win=64240)
13	180.149.131.98	192.168.201.200	1498	1498	TCP	2017-11-02 10:31:53.237	0.050138	HTTP -> 1300 ([ACK, PUSH], Seq=777104703, Ack=3413535929, Win...

6. PCAP包分析

数据包分析

使用wireshark打开文件，利用“会话->统计”功能，找到传输内容排名靠前的会话（如192.168.201.200->10.255.8.130），发现是图片文件下载（a5.jpg）。

Ethernet · 1 | IPv4 · 11 | IPv6 | TCP · 30 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Packets A → B	Bytes A → B	Pac
192.168.201.200	1315	10.255.8.130	80	279	380 k	23	1689	
192.168.201.200	1300	180.149.131.98	80	40	34 k	13	2005	
192.168.201.200	1308	42.81.93.40	80	17	4615	8	2416	

Wireshark · 追踪 TCP 流 (tcp.stream eq 13) · EveryBit

```
GET /a5.jpg HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash,
application/xaml+xml, application/x-ms-xbap, application/x-ms-application, */*
Accept-Language: zh-cn
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; @1uD/u<yu2GZB8tZ]<L%Umchn8lNzc3RL>=*`;
.NET4.0C; .NET4.0E; .NET CLR 2.0.50727)
Host: 10.255.8.130
Connection: Keep-Alive
```

6. PCAP包分析

图片格式分析

使用十六进制编辑工具（如UE）打开图片文件，根据JPG文件格式（0xFFD8为图像开始，0xFFD9为图像结束），在图片结束位置发现附加数据，其中含有大量类似“yitna”的字符，猜测为KEY，对应XOR内容为0x00，对附加数据解密，得到PE可执行程序。

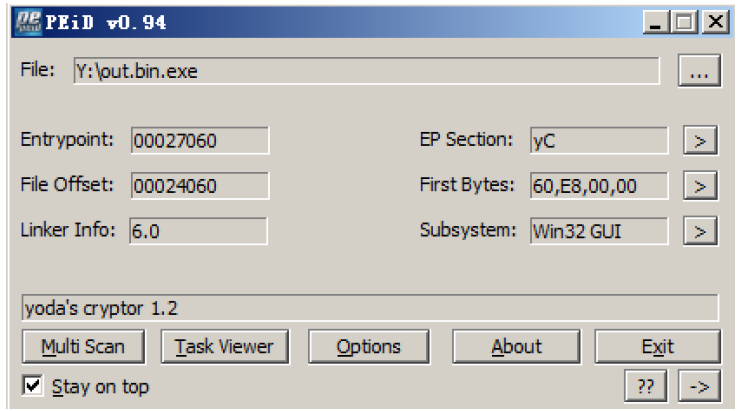
符号	标记代码	说明
SOI	0xFFD8	Start of image
EOI	0xFFD9	End of image
……	……	……

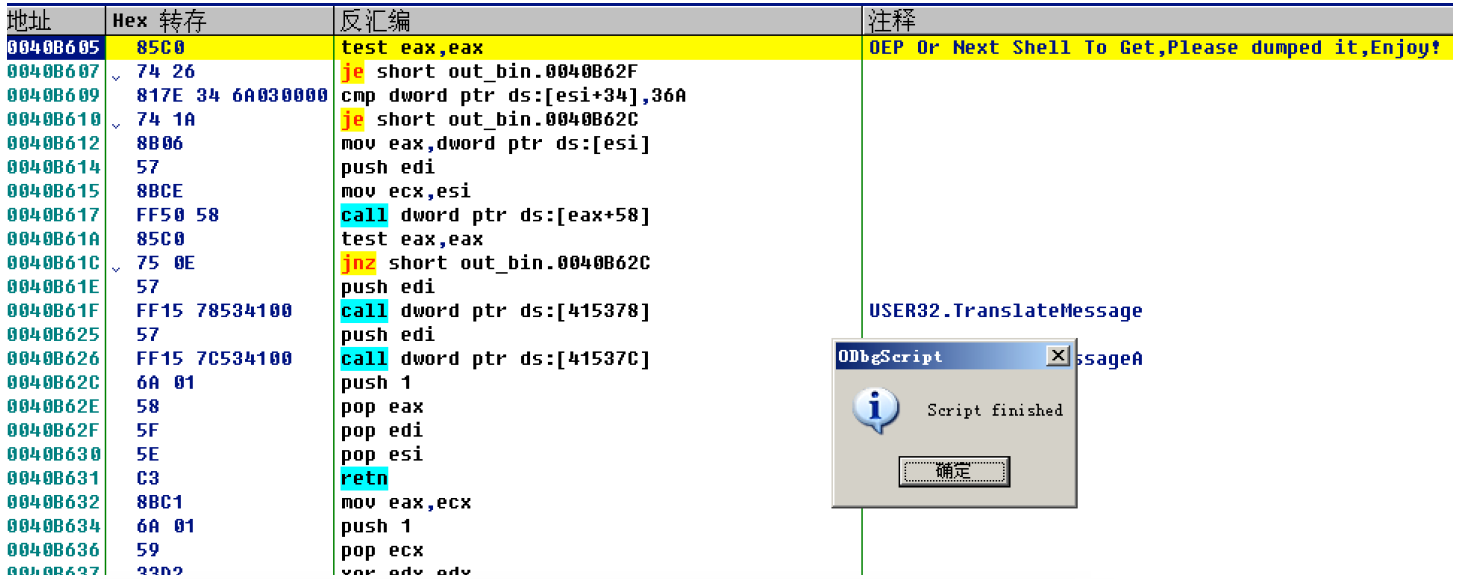
```
000347d0h: 27 08 21 50 96 20 AD C7 81 02 DF 08 1F FD 6A FF ; '.!P? ??.?
000347e0h: 00 FF D9 00 00 00 00 00 00 00 00 00 34 33 E4 6E ; . ?.........43?
000347f0h: 62 79 69 74 6A 61 79 69 8B 91 61 79 D1 74 6E 61 ; byitjayi??ay?na
00034800h: 79 69 74 6E 21 79 69 74 6E 61 79 69 74 6E 61 79 ; yitn!yitnayitnay
00034810h: 69 74 6E 61 79 69 74 6E 61 79 69 74 6E 61 79 69 ; itnayitnayitnayi
00034820h: 74 6E 61 79 69 74 6E 61 99 69 74 6E 6F 66 D3 7A ; tnayitna?tnof?
00034830h: 6E D5 70 A4 55 D6 60 35 A4 55 3A 09 10 1A 54 1E ; n? ?5 :...T.
00034840h: 13 16 0E 06 0F 0C 59 0A 15 00 0F 16 1D 54 0C 04 ; ......Y......T..
00034850h: 59 1B 01 00 41 10 07 54 2A 2E 2A 49 19 01 05 1C ; Y...A..T*.*I....
00034860h: 47 79 63 6B 5D 69 74 6E 61 79 69 74 E8 7B 0A B0 ; Gyck]itnayit?.?
00034870h: B6 15 7C F3 AB 0F 73 EB BB 12 69 E4 A3 02 74 FE ; ?|?.s?.i?.t?
00034880h: A0 1A 64 E3 B6 15 7D F3 3E 0E 73 EB D9 0D 7A E4 ; ?d?.}?.s?.z?
```

6. PCAP包分析

PE程序分析

分析该PE文件可知文件采用yoda加壳，需要使用脱壳工具或脚本（如yodas Crypter 1.2 OEP and Patch IAT v0.1.txt）脱壳后分析。对脱壳后程序修复IAT或直接用IDA Pro静态分析。根据关键字“Congratulation”附近逻辑不难发现以下两段密文。






```
02 3D A3 67 7A 82 FC A2 02 5C 06 18 14 03 26 29 ; .=  z傀?\....&)
00 00 00 00 6F 3D 3A 34 28 AE 65 69 A8 99 0A DA ; ....o=:4(瞽i  .?
9D 09 A2 0D 00 00 00 00 57 72 6F 6E 67 20 21 00 ; ??....Wrong !.
43 6F 6E 67 72 61 74 75 6C 61 74 69 6F 6E 00 00 ; Congratulation..
```

算法

以程
等算
分析
（境

A
动态
6算法。

```
for ( *(&v18
{
  v6 = v4 / 4
  v7 = (*(&v1
  --v4;
}
dword_41C53C[
v8 = &dword_4
do
{
  *v8 = *(v8
  ++v8;
}
while ( (sign
```

国内版 国际版

0xB7E15163

网页 图片 视频 学术 词典 地图

77 条结果 时间不限

RC5_百度百科

对于32位字和64位分组的RC5,
P=0xb7151628aed2a6b Q=0x
https://wapbaike.baidu.com/ite

RC6加密C语言的例

2013-1-8 · 1 #include <stdio.h
P32 0xB7E15163 /* 定义两个常
www.cnblogs.com/yekang/arch

RC6加密解密算法实

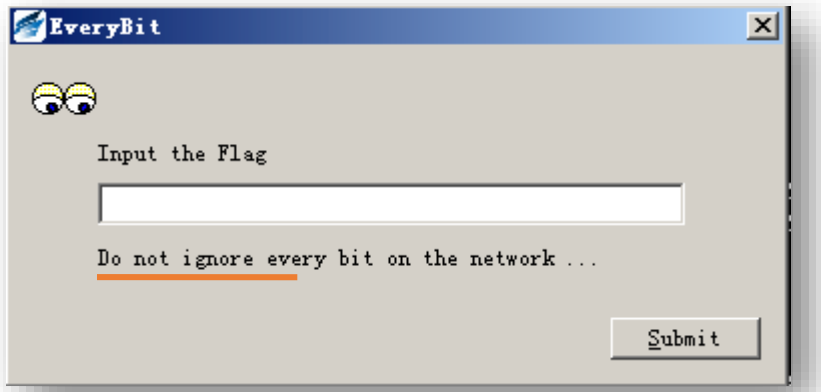
c=4;const unsigned int p=0xb7
moveleft(unsigned int x,unsign
blog.csdn.net/haroroda/article/

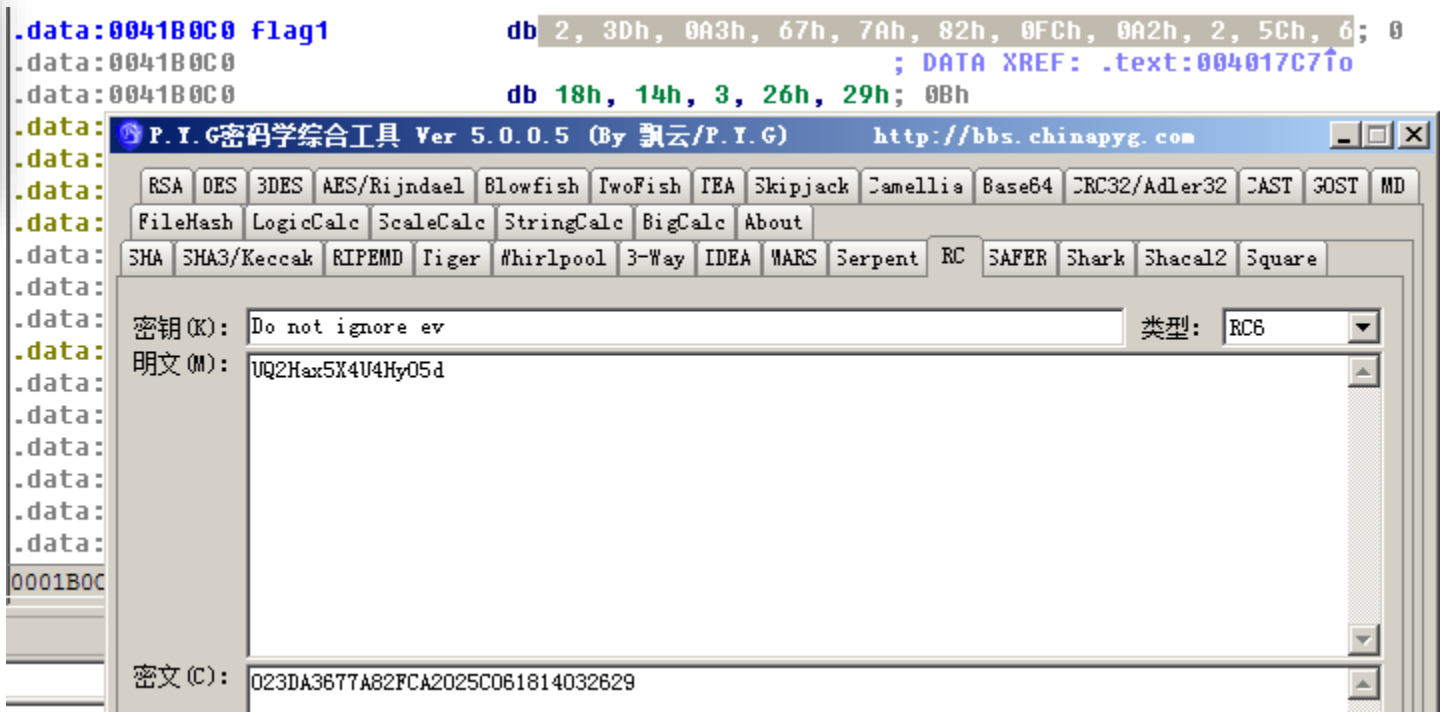
```
void RC6::Base::UncheckedSetKey(const byte *k, unsigned int keylen, const NameValuePairs &params)
{
    AssertValidKeyLength(keylen);

    r = GetRoundsAndThrowIfInvalid(params, this);
    sTable.New(2*(r+2));

    static const RC6_WORD MAGIC_P = 0xb7e15163L;    // magic constant P for wordsize
    static const RC6_WORD MAGIC_Q = 0x9e3779b9L;    // magic constant Q for wordsize
    static const int U=sizeof(RC6_WORD);

    const unsigned int c = STDMAX((keylen+U-1)/U, 1U);  // RC6 paper says c=1 if keylen==0
    SecBlock<RC6_WORD> l(c);

    GetUserKey(LITTLE_ENDIAN_ORDER, l.begin(), c, k, keylen);

    sTable[0] = MAGIC_P;
    for (unsigned j=1; j<sTable.size();j++)
        sTable[j] = sTable[j-1] + MAGIC_Q;

    RC6_WORD a=0, b=0;
    const unsigned n = 3*STDMAX((unsigned int)sTable.size(), c);

    for (unsigned h=0; h < n; h++)
    {
        a = sTable[h % sTable.size()] = rotlConstant<3>((sTable[h % sTable.size()] + a + b));
        b = l[h % c] = rotlMod((l[h % c] + a + b), (a+b));
    }
}
```

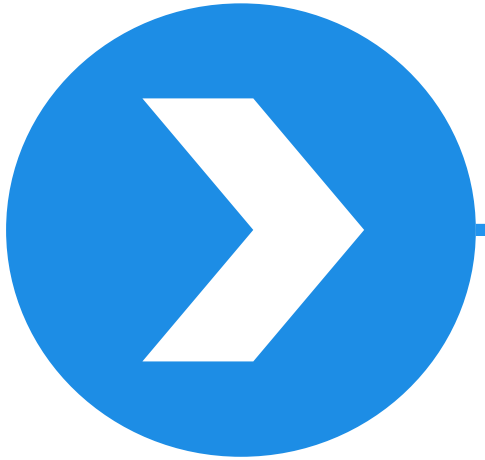
6. PCAP包分析

找到密钥

在密文、算法皆有线索的前提下，通过分析找到密钥。通过分析程序，可发现程序中有两个KEY。
一个由网络下载（http://overseas.weico.cc/share/8770550.html），另一个由对话框界面获取。
通过分析程序，找到正确的KEY（16字节），借助密码工具（如P.Y.G密码学综合工具），即可得出明文。







第二轮

- SPARC架构程序分析

7. Solaris 平台的SPARC架构程序分析

题目分值	55分
待分析文件名称	loop_C09794E6AF8849BCF88FAE133D2091BB
待分析文件MD5值	C09794E6AF8849BCF88FAE133D2091BB
运行测试环境	Solaris 10
考察的技术点	SPARC汇编指令、SPARC调试、
题目描述	通过对文件静态分析，根据出现明确的提示信息，分析出相应的FLAG。
备注	提交正确答案即获得相应的分数

SPARC，“可扩充处理器架构”（Scalable Processor ARChitecture），是RISC微处理器架构之一，具有高性能、可扩展性和稳定性等优点。

7. Solaris 平台的SPARC架构程序分析

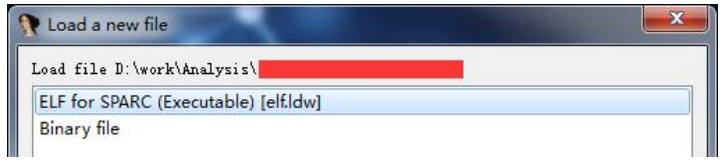
- FILE

```
root@ubuntu:/home/cert# file sample
sample: ELF 32-bit MSB executable, SPARC, version 1 (SYSV), dynamically linked (
uses shared libs), stripped
```

- EXIFTOOL

```
ExifTool Version Number         : 9.39
File Name                       : 2aa9891705977b1
777a727d4ba740764f
Directory                       :
File Size                       : 819 kB
File Modification Date/Time     : 2015:01:18 15:02:30+08:00
File Access Date/Time           : 2015:01:18 15:01:47+08:00
File Creation Date/Time         : 2015:01:18 15:01:47+08:00
File Permissions                : rw-rw-rw-
File Type                       : ELF executable
MIME Type                       : application/octet-stream
CPU Architecture                : 32 bit
CPU Byte Order                  : Big endian
Object File Type                : Executable file
CPU Type                        : SPARC
```

- IDA



SPACR 寄存器

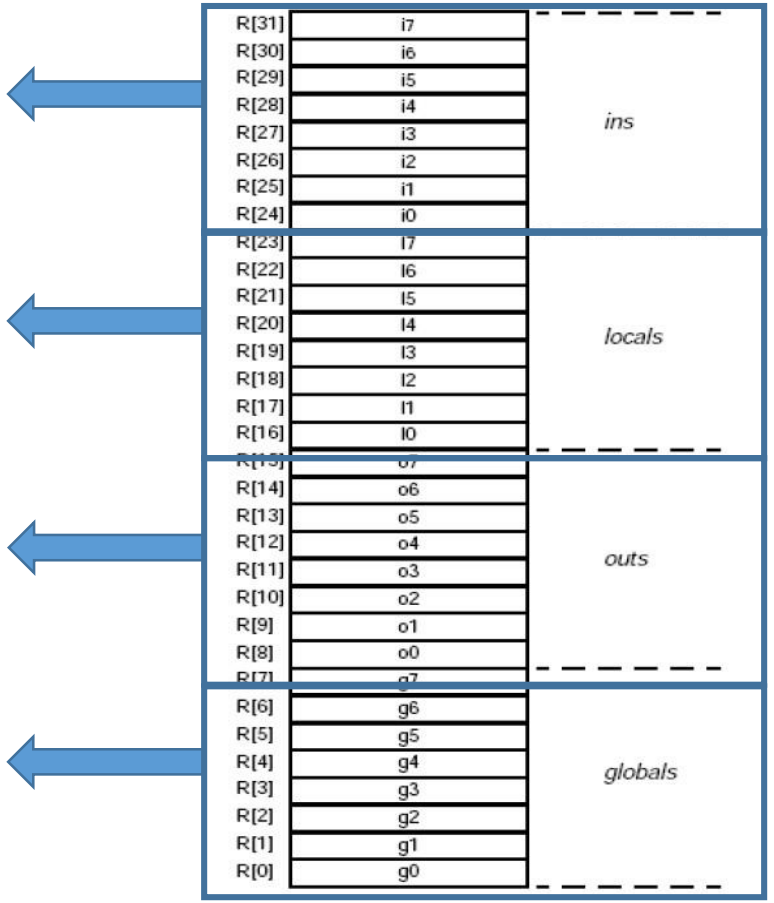
SPARC的通用寄存器不同于X86，在某一时刻有32个寄存器 r0-r31

输入寄存器(8个)－命名为%i0～%i7，
等同于%r24～%r31

局部寄存器(8个)－仅本函数可见
命名为%l0～%l7，等同于%r16～%r23

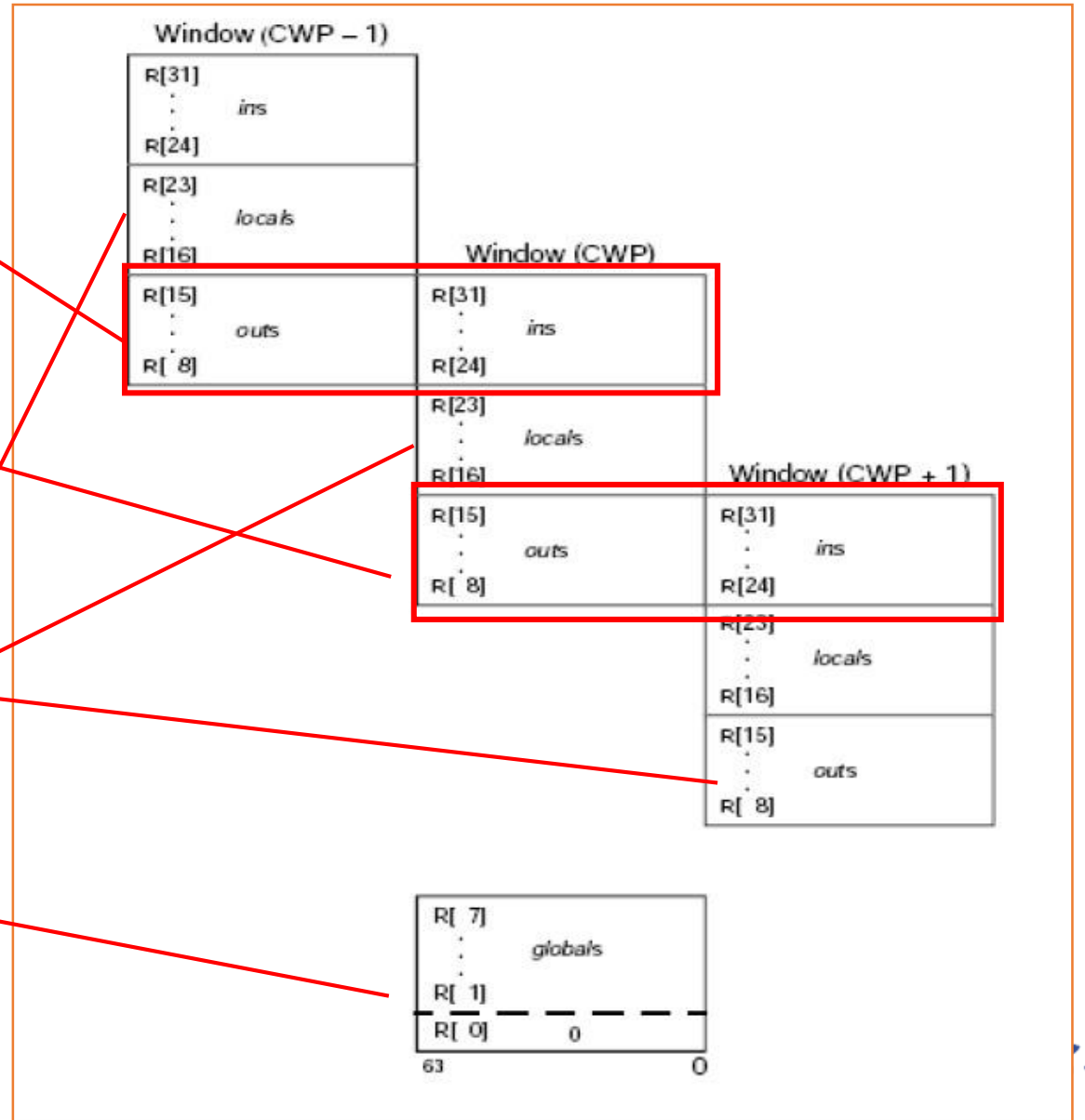
输出寄存器(8个)－函数返回值
命名为%o0～%o7，等同于%r8～%r15

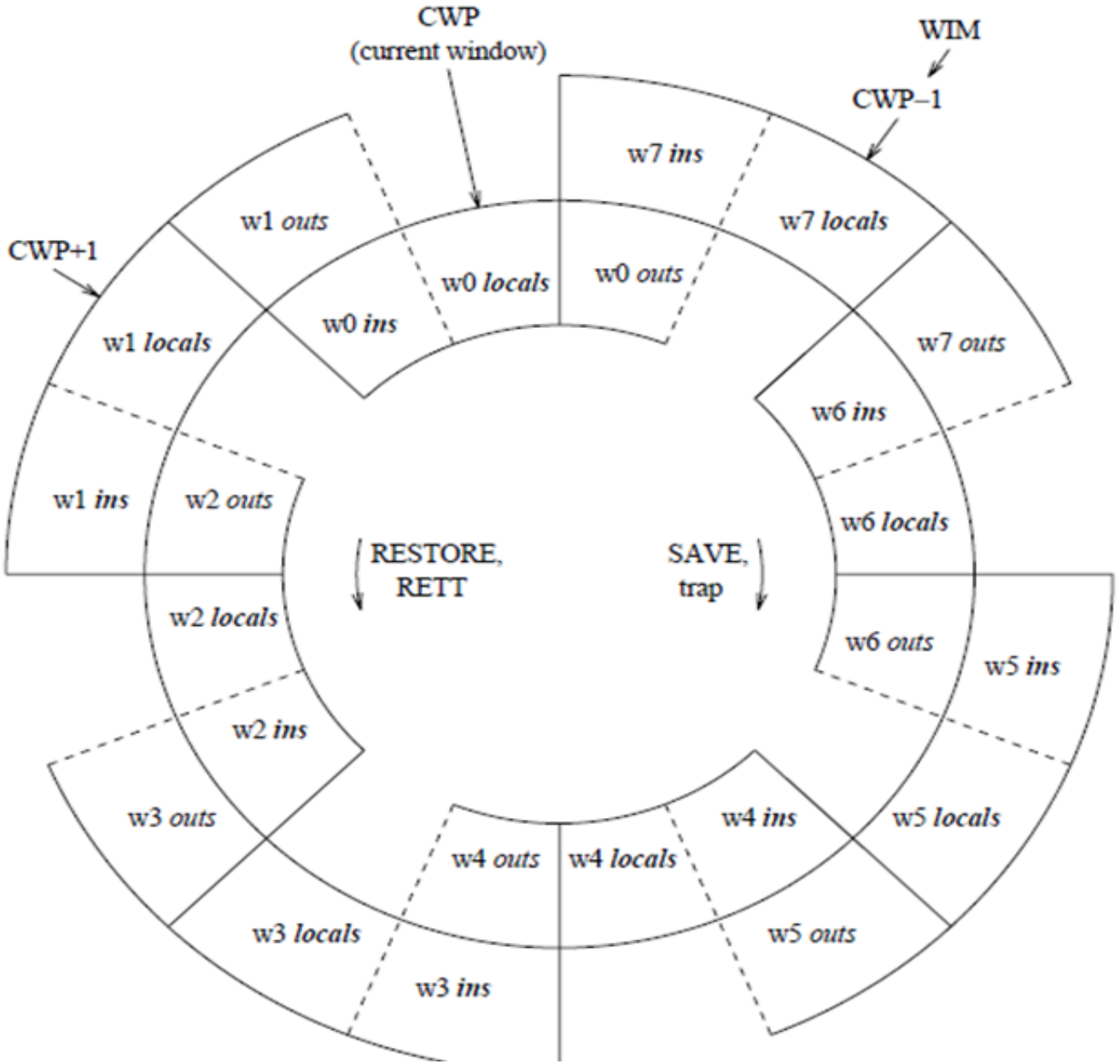
全局寄存器(8个)－对所有函数可见
命名为%g0～%g7 等同于%r0～%r7



窗口管理

- 当前窗口的 i0 ~ i7和上一窗口的 o0 ~ o7 对应于同一组物理寄存器；
- 当前窗口o0 ~ o7作为参数传入下层窗口i0 ~ i7对应于同一组物理寄存器
- 所有窗口的local寄存器绝对独立在自己窗口呢
- global不动




CWP
(current window)
WIM
CWP–1
CWP+1
w7 ins
w7 locals
w7 outs
w0 outs
w0 locals
w0 ins
w1 outs
w1 locals
w1 ins
w2 outs
w6 ins
w6 locals
w6 outs
w5 ins
w5 locals
w5 outs
w4 ins
w4 locals
w4 outs
w3 ins
w3 locals
w3 outs
w2 ins
w2 locals
RESTORE,
RETT
SAVE,
trap

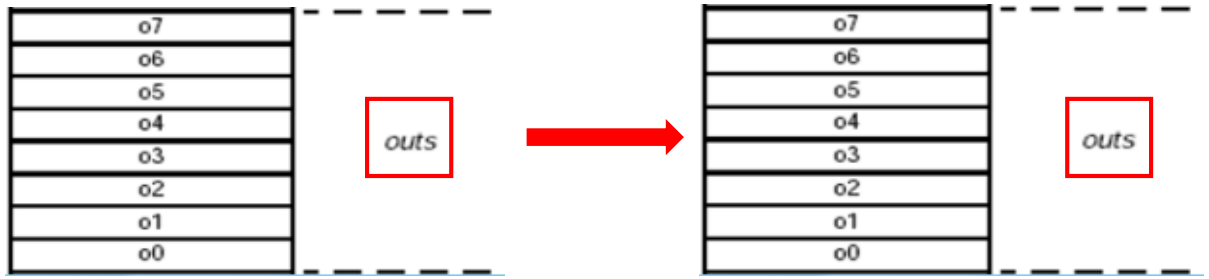
窗口管理-不转动

- 特定不转动窗口

```
sub_12345:
cmp %o0, 0
be  end
```

```
sub_45678:
ld     [%o0], %g1
sethi  %hi(0x8000000), %o3
andn   %g1, %o3, %o3
mov    5, %g1
mov    0, %o4
mov    0, %o5
```

```
sub_12345:
retl
add    %o7, %l7, %l7
```



过程调用

- 参数

父函数写入o0-o5，子函数使用i0-i5接受。

父函数

```
mov     %l2, %o1
add     %fp, var, i3
mov     i0, %o0
mov     %l2, %o1
call    sub_12345
or      %l4, %l5, %l1
```

1 2

子函数

```
save    %sp, -0x80, %sp
mov     0, %o4
mov     0, %o5
mov     %i0, %o0
mov     %l0, %o1
mov     %i1, %o2
call    sub_45678
```

过程调用

- 返回值

子函数写入i0~i5，父函数o0~o5接受。

子函数

```
mov     0, %i0
ret
restore
```

```
call    sub_56789
mov     1, %i0
ba,a    end
ret
restore
```

父函数

```
call    sub_12345
mov     %i0, %o0
cmp     %o0, 1
bne     end
```

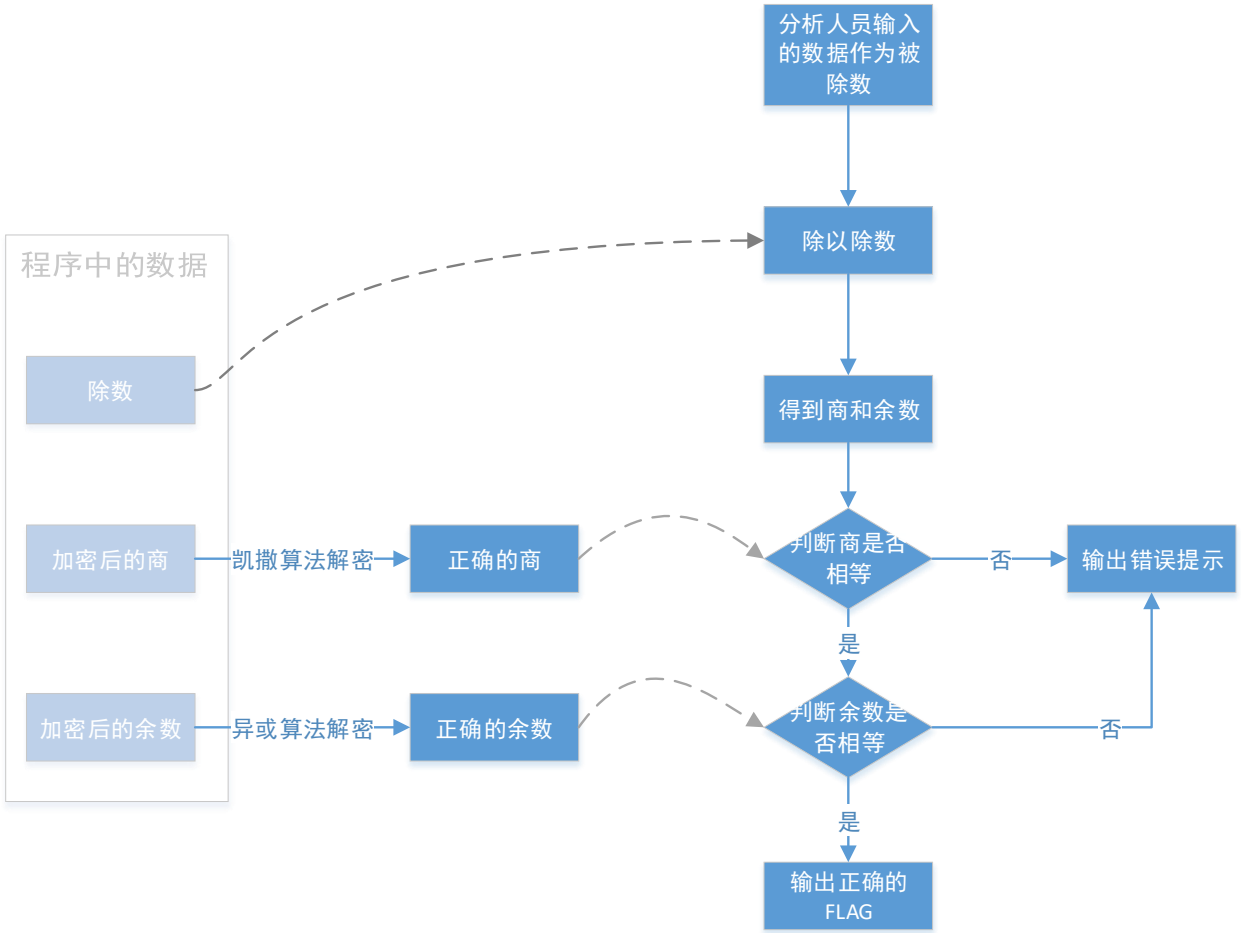
延迟槽 Delay slots

- 分支延迟槽（Branch delay slot），简单地说就是位于分支指令后面的一条指令，不管分支发生与否其总是被执行，**而且位于分支延迟槽中的指令先于分支指令执行**
- 不是SPARC独有的
  - 存在延迟槽的架构：MIPS、PA-RISC、ETRAX CRIS、SuperH、SPARC等
  - 不存在延迟槽的架构：X86、PowerPC、ARM、DEC Alpha

7. Solaris 平台的SPARC架构程序分析

要点

- SPARC架构
- Oracle Solaris系统
- 大数除法
- 凯撒算法
- 异或算法

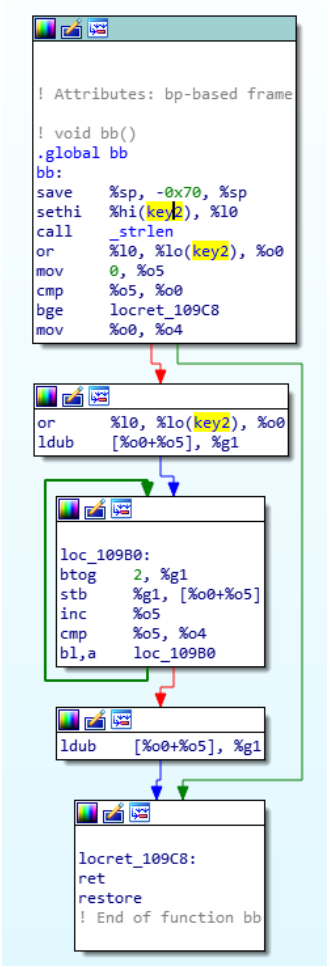


第七题执行流程图

7. Solaris 平台的SPARC架构程序分析


```
loc_10B34:
sethi   %hi(a48222290543499), %o1 ! "48222290543499671471"
sethi   %hi(key), %o0    ! s1
bset    %lo(a48222290543499), %o1 ! "48222290543499671471"
mov     0x15, %o2        ! n
call    _memcpy
bset    %lo(key), %o0
call    aa
nop
set     key2, %o4
mov     0x35, %g1
mov     0x34, %o5
stb     %g1, [%o3+0x220]
stb     %o5, [%o4+1]
mov     0x31, %g1
mov     0x3A, %o5
stb     %g1, [%o4+2]
stb     %o5, [%o4+3]
mov     0x30, %g1
mov     0x3B, %o5
stb     %g1, [%o4+4]
stb     %o5, [%o4+5]
call    bb
clrb    [%o4+6]
sethi   %hi(a), %o5
ld      [%o5+%lo(a)], %g1
cmp     %g1, 0
mov     0, %o3
bne     loc_10BC8
or      %o5, %lo(a), %g1
```

凯撒算法

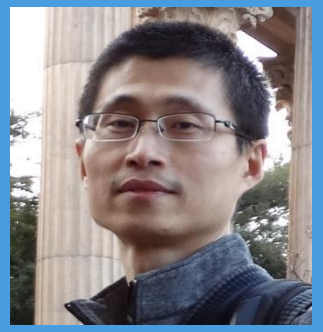


异或算法

```
loc_10C7C:
sethi   %hi(aContratulation), %o0 ! "Contratulations! The Flag is BJWXB_CTF{"
call    _printf
bset    %lo(aContratulation), %o0 ! "Contratulations! The Flag is BJWXB_CTF{"
or      %i3, 0x190, %o1
call    _printf
or      %i4, 0x168, %o0
sethi   %hi(asc_10DD0), %o0 ! "}"
call    _puts
bset    %lo(asc_10DD0), %o0 ! "}"
ba      locret_10CB8
mov     0, %i0
```

Key信息

感谢大家参与本次交流



libaisong@antiy.cn

weibo.com/libaisong75