脱asprotect使用如下脚本：

/\*

Script written by VolX

Script : Aspr2.XX\_unpacker

版本 : v1.14aSC

日期 : 19-May-2008

调试环境 : OllyDbg 1.1, ODBGScript 1.65, WINXP, WIN2000

调试选项 : 设置 OllyDbg 忽略所有异常选项

工具 : OllyDbg, ODBGScript 1.65, Import Reconstructor.

感谢 : Oleh Yuschuk - author of OllyDbg

SHaG - author of OllyScript

Epsylon3 - author of ODbgScript

特别感谢 : fly, linex, machenglin 等兄弟的帮忙测试.

\*/

//support Asprotect 1.32, 1.33, ,1.35, 1.4, 2.0, 2.1, 2.11, 2.2beta, 2.2, 2.3, 2.4

var tmp1

var tmp2

var tmp3

var tmp4

var tmp5

var tmp6

var tmp7

var tmp8

var tmp9

var tmp10

var imgbase

var imgbasefromdisk

var 1stsecbase

var 1stsecsize

var ressecbase

var signVA

var sizeofimg

var dllimgbase

var count

var transit1

var transit2

var func1

var func2

var func3

var func4

var OEP\_rva

var caller

var caller1

//for IAT fixing

var paddr1

var paddr2

var paddr3

var paddr4

var paddr5

var paddr6

var ori1

var ori2

var ori3

var ori4

var ori5

var iatstartaddr

var iatstart\_rva

var iatendaddr

var iatsize

var EBXaddr

var ESIaddr

var lastsecbase

var lastsecsize

var thunkdataloc

var thunkpt

var thunkstop

var type3API

var type3count

var type1API

var E8count

var writept2

var APIpoint3

var crcpoint1

var FF15flag

var ESIpara1

var ESIpara2

var ESIpara3

var ESIpara4

var nortype

var DFCequ

var DFCaddr

var REequ

var REaddr

var GPAequ

var GPAaddr

var v1.32

var v2.0x

var newver

var sttablesize

//for stolencode after API

var SCafterAPIcount

//for dll

var reloc\_rva

var reloc\_size

var isdll

var reloc1

var reloc2

var reloc3

var reloc4

var reloc5

var reloc6

var reloctemp

//for Aspr API

var Aspr1stthunk

var AsprAPIloc

var EmuAddr

//std function

var 55pt

var 55struct1

var 55dataloc

var 55sc

//delphi initialization table

var dataendaddr

var countaddr

var tablea

var tableb

var decryptaddr

var dataloc

//OEP/SDK stolen code

var 57pt

var 57jmppt

var 57struct

var jmptablesize

var scstk

var OEPscaddr

var xtrascloc //dllimgbase+F00

var dualvc

var sdkscaddr

var sdksccount

var vcrefstart

var vcrefend

var findendaddr

var patchaddr

var patchendaddr

var patchinsamesec

var SDKsize

var newphysec

var newphysecsize

var virtualsec

var newzeroVA

var curzeroVA

var virzeroVA

var newpatchaddr

var newpatchendaddr

//VM

var VMcodeloc

var VMstartaddr

var VMlength

cmp $VERSION, "1.64"

jb odbgver

dbh

BPHWCALL //clear hardware breakpoint

GMI eip, MODULEBASE //get imagebase

mov imgbase, $RESULT

//log imgbase

mov tmp1, [imgbase+3C]

add tmp1, imgbase //tmp1=signature VA

mov signVA, tmp1

mov imgbasefromdisk, [signVA+34]

//log imgbasefromdisk

mov sizeofimg, [signVA+50]

mov tmp2, [signVA+88]

add tmp2, imgbase

mov ressecbase, tmp2

mov 1stsecsize, [signVA+100]

//log 1stsecsize

mov 1stsecbase, [signVA+104]

add 1stsecbase, imgbase

//log 1stsecbase

mov tmp1, signVA

add tmp1, f8 //1st section

mov tmp2, 0

mov tmp2, [signVA+6], 2

last:

cmp tmp2, 1

je lab1

add tmp1, 28

sub tmp2, 1

jmp last

lab1:

mov lastsecsize, [tmp1+8]

//log lastsecsize

mov tmp3, [tmp1+0C]

add tmp3, imgbase

mov lastsecbase, tmp3

//log lastsecbase

//check if its an exe or dll

cmp imgbasefromdisk, imgbase

je lab1\_1

mov isdll, 1

jmp lab1\_2

lab1\_1:

GPI EXEFILENAME

mov tmp1, $RESULT

cmp tmp1, 0

je error

GPI PROCESSNAME

mov tmp2, $RESULT

GPI CURRENTDIR

mov tmp3, $RESULT

eval "{tmp3}{tmp2}.exe"

mov tmp4, $RESULT

eval "{tmp3}{tmp2}.dll"

mov tmp5, $RESULT

scmpi tmp1, tmp4

je lab1\_2

scmpi tmp1, tmp5

jne error

mov isdll, 1

lab1\_2:

cob

coe

gpa "GetSystemTime", "kernel32.dll"

bp $RESULT

esto

bc $RESULT

rtr

sti

GMEMI eip, MEMORYOWNER

mov dllimgbase, $RESULT

cmp dllimgbase, 0

je error

//log dllimgbase

find dllimgbase, #3135310D0A#

mov tmp1, $RESULT

cmp tmp1, 0

je wrongver

find dllimgbase, #0F318901895104# //check rdtsc trick

mov tmp1, $RESULT

cmp tmp1, 0

je lab1\_5

sub tmp1, 80

find tmp1, #558BEC#

mov tmp1, $RESULT

cmp tmp1, 0

je error

bp tmp1

eob lab1\_3

eoe lab1\_3

esto

lab1\_3:

cmp eip, tmp1

je lab1\_4

esto

lab1\_4:

bc tmp1

mov eip, [esp]

add esp, 4

lab1\_5:

find dllimgbase, #8B5F048B3383C304# //search "mov ebx,[edi+4]" "mov esi,[ebx]""add ebx,4"

mov tmp2, $RESULT

cmp tmp2, 0

jne lab1\_6

find dllimgbase, #8B6F048B750083C504# //search "mov ebp,[edi+4]" "mov esi,[ebp]""add ebp,4"

mov tmp2, $RESULT

cmp tmp2, 0

jne lab1\_6

find dllimgbase, #8B6?0?8B?50083C504# //search "mov ebp,[e??+0?]" "mov e??,[ebp]""add ebp,4"

mov tmp2, $RESULT

cmp tmp2, 0

je error

lab1\_6:

find dllimgbase, #3138310D0A#

cmp $RESULT, 0

je lab1\_7

sub tmp2, 600

jmp lab1\_8

lab1\_7:

sub tmp2, 200

lab1\_8:

find tmp2, #8BF08973??# //search "mov esi, eax", "mov [ebx+??], esi"

mov tmp3, $RESULT

cmp tmp3, 0

je error

mov 57pt, tmp3

find 57pt, #3130370D0A#

mov tmp5, $RESULT

cmp tmp5, 0

je error

sub tmp5, 57pt

cmp tmp5, 0A0

ja error

lab2:

//log 57pt

mov tmp1, dllimgbase

add tmp1, 010e00

find tmp1, #892D????????3b6C24??#

mov tmp2, $RESULT

cmp tmp2, 0

je error45

find tmp2, #833C240074??#

mov tmp4, $RESULT

cmp tmp4, 0

je error45

add tmp4, 4

find tmp1, #8B5483408BC6# //search "mov edx,[ebx+eax\*4+40]" "mov eax,esi"

mov tmp2, $RESULT //vcpoint

cmp tmp2, 0

je error

find tmp2, #807B740074??# //search "cmp [ebx+74],0" "je xxxxxxxx"

mov tmp3, $RESULT

cmp tmp3, 0

je lab2\_1

mov dualvc, 1

lab2\_1:

bp tmp4

eob lab3

eoe lab3

esto

lab3:

cmp eip, tmp4

je lab4

esto

lab4:

bc tmp4

mov tmp1, eip

sub tmp1, 1000

find tmp1, #F3A566A5# //search "rep movs[edi],[esi]","movs [edi],[esi]"

mov tmp1, $RESULT

cmp tmp1, 0

je error

find tmp1, #0F84??000000#

mov thunkstop, $RESULT

//log thunkstop

bp thunkstop

find dllimgbase, #45894500# //search "inc ebp", "mov [ebp],eax"

mov tmp2, $RESULT

cmp tmp2, 0

je error

sub tmp2, 27

mov APIpoint3, tmp2

//log APIpoint3

find dllimgbase, #40890383C704#

mov tmp1, $RESULT

add tmp1, 1

mov thunkpt, tmp1

//log thunkpt

cmp isdll, 1

jne lab7\_1

mov !zf, 1

mov tmp1, eip

mov tmp2, [tmp1+2], 2

cmp tmp2, 5C03 //chk if "add ebx, [esp+4]"

je lab5

cmp tmp2, 5C8B //chk if "mov ebx, [esp+4]"

jne error

mov reloc\_rva, esi

mov tmp1, esi

jmp lab6

lab5:

mov reloc\_rva, ebx

mov tmp1, ebx

lab6:

add tmp1, imgbase

call ChkRelocSize

lab7:

mov reloc\_size, tmp2

lab7\_1:

bp thunkpt

find dllimgbase, #33C08A433?3BF0# //search "xor eax,eax", "mov al, {ebx+3?]", "cmp esi,eax"

mov paddr1, $RESULT

cmp paddr1, 0

je error

add paddr1, 7

//log paddr1

mov tmp2, [paddr1-3], 1

cmp tmp2, 3F

jne lab8

mov v1.32, 1

lab8:

mov thunkdataloc, dllimgbase

add thunkdataloc, 200 //dllimgbase+200

find dllimgbase, #0036300D0A#

mov tmp1, $RESULT

cmp tmp1, 0

je error

find tmp1, #68????????68????????68????????68????????#

mov tmp1, $RESULT

add tmp1, 14

mov tmp3, [tmp1], 2

cmp tmp3, 35FF

je lab11

mov crcpoint1, tmp1

//log crcpoint1

bp crcpoint1

eob lab9

eoe lab9

esto

lab9:

cmp eip, crcpoint1

je lab10

esto

lab10:

eob

eoe

bc crcpoint1

bc thunkpt

bc thunkstop

rtr

sti

bp thunkpt

bp thunkstop

lab11:

eob lab12

eoe lab12

esto

lab12:

cmp eip, thunkpt

je lab13

cmp eip, thunkstop

je lab18

esto

lab13:

bc thunkpt

mov ESIaddr, esi

//log ESIaddr

mov ori1, [paddr1]

mov ori2, [paddr1+4]

mov tmp1, [signVA+30]

add tmp1, imgbase

find tmp1, #426F726C616E6420432B2B202D# //Search "Borland C++ -"

mov tmp2, $RESULT

cmp tmp2, 0

jne lab13\_1

find tmp1, #436F64654765617220432B2B202D# //Search "CodeGear C++ -"

mov tmp2, $RESULT

cmp tmp2, 0

je lab13\_2

lab13\_1:

mov tmp1, [ebx]

add tmp1, imgbase

GMEMI tmp1, MEMORYBASE

mov tmp2, $RESULT

cmp tmp2, 0

je error

GMEMI tmp1, MEMORYSIZE

mov tmp3, $RESULT

cmp tmp3, 0

je error

fill tmp2, tmp3, 00

lab13\_2:

find eip, #3A5E3?7517#

mov tmp1, $RESULT

cmp tmp1, 0

je error

mov ESIpara1, [tmp1]

//log ESIpara1

add tmp1, 6

find tmp1, #3A5E3?7517#

mov tmp2, $RESULT

cmp tmp2, 0

je error

mov ESIpara2, [tmp2]

//log ESIpara2

add tmp2, 6

find tmp2, #3A5E3?75??#

mov tmp1, $RESULT

cmp tmp1, 0

je error

mov ESIpara3, [tmp1]

//log ESIpara3

add tmp1, 6

//chk version is with AsprAPI ?

find dllimgbase, #3138300D0A#

mov tmp2, $RESULT

cmp tmp2, 0

je lab13\_3

find tmp1, #8A07E8#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 3

mov tmp6, [tmp2]

add tmp6, tmp2

add tmp6, 5

lab13\_3:

find tmp1, #473A5E3?#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 1

mov tmp3, [tmp2], 3

add tmp3, 74000000

mov ESIpara4, tmp3

//log ESIpara4

find eip, #834424080447EB1A# //search "add [esp+8],4", "inc edi"

mov tmp1, $RESULT

cmp tmp1, 0

je lab13\_4

mov nortype, 1

//log nortype

//checking iatendaddr

lab13\_4:

mov tmp7, eip //save eip

mov tmp1, dllimgbase

mov [tmp1], #609CBE740E8C00BD000F8600C74500000286008B4D008B0305000000018901834500048BFB83C70A83C1048939834500#

add tmp1, 30 //30

mov [tmp1], #0433C0B9FFFFFFFFF2AE8A1F3A5E34744B3A5E37750883C707FF45FCEBEC3A5E38750883C705FF45FCEBDF3A5E3A751C#

add tmp1, 30 //60

mov [tmp1], #508D47F58B0089452058C78560F1FFFFEB12909083C704FF45FCEBBE83C703668B0783C00203F8FF45FCEBAE807D0401#

add tmp1, 30 //90

mov [tmp1], #7469478BDF833B000F8575FFFFFFC6450401C7450800026304C745FC000000008B45088B0089450C8945148B45088B40#

add tmp1, 30 //C0

mov [tmp1], #04894510834508088B45088B0083F80074213B450C720E89450C8B5D088B5B04895D10EB083B45147703894514834508#

add tmp1, 30 //F0

mov [tmp1], #08EBD58B7D10E936FFFFFFB8000263048B0883F90074113B4D147407C741FC0000000083C008EBE89D61909000000000#

mov tmp1, dllimgbase

mov tmp2, dllimgbase

add tmp2, 0F00 //dllimgbase+F00

add tmp1, 3 //3

mov [tmp1], ESIaddr

add tmp1, 5 //8

mov [tmp1], tmp2

add tmp1, 7 //F

mov [tmp1], thunkdataloc

add tmp1, A //19

mov [tmp1], imgbase

add tmp1, 23 //3C

mov [tmp1], ESIpara4

add tmp1, 5 //41

mov [tmp1], ESIpara1

add tmp1, D //4E

mov [tmp1], ESIpara2

add tmp1, D //5B

mov [tmp1], ESIpara3

add tmp1, 4A //A5

mov [tmp1], thunkdataloc

add tmp1, 57 //FC

mov [tmp1], thunkdataloc

cmp nortype, 1

je lab14

mov tmp1, dllimgbase

add tmp1, 74 //74

mov [tmp1], #83C705FF#

lab14:

cob

coe

mov tmp4, dllimgbase

add tmp4, 11A //end point

bp tmp4

mov eip, dllimgbase

run

bc tmp4

mov eip, tmp7 //restore eip

mov tmp1, dllimgbase

add tmp1, 0EFC

mov tmp2, [tmp1] //API count of last dll

mov tmp3, [tmp1+10] //last thunk addr

shl tmp2, 2

add tmp3, tmp2

mov iatendaddr, tmp3

//log iatendaddr

mov iatstartaddr, [tmp1+18]

//log iatstartaddr

mov iatstart\_rva, iatstartaddr

sub iatstart\_rva, imgbase

mov [iatendaddr], 0

mov tmp2, iatendaddr

sub tmp2, iatstartaddr

add tmp2, 4

mov iatsize, tmp2

find dllimgbase, #3138300D0A#

cmp $RESULT, 0

je lab14\_1

find tmp6, #BA01000000B9#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 6

mov AsprAPIloc, [tmp2]

log AsprAPIloc

mov tmp2, [tmp1+24]

cmp tmp2, 0

je lab14\_1

add tmp2, imgbase

mov Aspr1stthunk, tmp2

log Aspr1stthunk

lab14\_1:

fill dllimgbase, f30, 00

//force to decrypt all api

mov tmp1, dllimgbase

cmp v1.32, 1

je lab15

mov [tmp1], #570FB67B353BF775040FB673365F3BF00F8500000000E900000000#

jmp lab16

lab15:

mov [tmp1], #570FB67B393BF775040FB6733A5F3BF00F8500000000E900000000#

lab16:

add tmp1, 10

mov tmp2, paddr1

add tmp2, 60

eval "jnz 0{tmp2}"

asm tmp1, $RESULT

add tmp1, 6

mov tmp2, paddr1

add tmp2, 5

eval "jmp 0{tmp2}"

asm tmp1, $RESULT

eval "jmp {dllimgbase}"

asm paddr1, $RESULT

find paddr1, #3B432?74656AFF# //search "cmp eax,[ebx+2?]","je xxxxxx","push -1"

mov paddr2, $RESULT

cmp paddr2, 0

je lab17

add paddr2, 3

//log paddr2

mov ori3, [paddr2]

mov [paddr2], #EB#

lab17:

find paddr1, #3B432?741b6AFF# //search "cmp eax,[ebx+2?]","je xxxxxx","push -1"

mov paddr3, $RESULT

cmp paddr3, 0

je error

add paddr3, 3

//log paddr3

mov ori4, [paddr3]

mov [paddr3], #EB#

find paddr1, #8902B8????????#

mov paddr4, $RESULT

cmp paddr4, 0

je error

add paddr4, 2

//log paddr4

gpa "DllFunctionCall", "MSVBVM60.dll"

mov tmp2, $RESULT

cmp tmp2, 0

je lab17\_1

GMEMI tmp2, MEMORYOWNER

mov tmp3, $RESULT

cmp tmp3, 0

jne lab17\_4

lab17\_1:

gpa "DllFunctionCall", "MSVBVM50.dll"

mov tmp2, $RESULT

cmp tmp2, 0

je lab17\_5

GMEMI tmp2, MEMORYOWNER

mov tmp3, $RESULT

cmp tmp3, 0

je lab17\_5

//如有必要在此加入更多 VB 版本.....

lab17\_4:

mov DFCaddr, tmp2

mov DFCequ, [paddr4+1]

mov tmp1, dllimgbase

add tmp1, 20 //dllimgbase+20

eval "jmp 0{tmp1}"

asm paddr4, $RESULT

mov [tmp1], #B8#

add tmp1, 1 //dllimgbase+21

mov [tmp1], tmp2

mov tmp3, paddr4

add tmp3, 5

add tmp1, 4 //dllimgbase+25

eval "jmp 0{tmp3}"

asm tmp1, $RESULT

lab17\_5:

mov count, 0 //counter

find paddr4, #C21000#

mov tmp1, $RESULT

cmp tmp1, 0

je error

mov tmp2, paddr4

loop2:

find tmp2, #Eb01??B8????????#

mov paddr5, $RESULT

cmp paddr5, 0

je loop2\_1

cmp paddr5, tmp1

ja loop2\_1

add count, 1

mov tmp2, paddr5

add tmp2, 8

jmp loop2

loop2\_1:

//log count

cmp count, 2

je lab17\_6

cmp count, 0

je lab17\_10

cmp count, 1

jne error

mov tmp4, paddr4

jmp lab17\_7

lab17\_6:

find paddr4, #Eb01??B8????????#

mov paddr5, $RESULT

cmp paddr5, 0

je error

add paddr5, 3

//log paddr5

mov tmp4, paddr5

gpa "RaiseException", "kernel32.dll"

mov tmp2, $RESULT

cmp tmp2, 0

je lab17\_7

GMEMI tmp2, MEMORYOWNER

mov tmp3, $RESULT

cmp tmp3, 0

je lab17\_7

mov REaddr, tmp2

mov REequ, [paddr5+1]

mov tmp1, dllimgbase

add tmp1, 30 //dllimgbase+30

eval "jmp 0{tmp1}"

asm paddr5, $RESULT

mov [tmp1], #B8#

add tmp1, 1 //dllimgbase+31

mov [tmp1], tmp2

mov tmp3, paddr5

add tmp3, 5

add tmp1, 4 //dllimgbase+35

eval "jmp 0{tmp3}"

asm tmp1, $RESULT

lab17\_7:

find tmp4, #Eb01??B8????????#

mov paddr6, $RESULT

cmp paddr6, 0

je error

add paddr6, 3

//log paddr6

mov tmp1, [paddr6+1]

mov tmp2, 0

mov tmp2, [tmp1], 1

cmp tmp2, 0E8

jne lab17\_8

mov tmp2, [tmp1+5], 2

cmp tmp2, 0E0FF

jne lab17\_10

gpa "RaiseException", "kernel32.dll"

mov tmp2, $RESULT

cmp tmp2, 0

je lab17\_10

GMEMI tmp2, MEMORYOWNER

mov tmp3, $RESULT

cmp tmp3, 0

je lab17\_10

mov REaddr, tmp2

mov REequ, [paddr6+1]

cmp count, 1

jne lab17\_9

mov paddr5, paddr6

jmp lab17\_9

lab17\_8:

mov tmp2, [tmp1+5], 1

cmp tmp2, 0C

jne lab17\_10

mov tmp2, [tmp1+8], 1

cmp tmp2, 08

jne lab17\_10

gpa "GetProcAddress", "kernel32.dll"

mov tmp2, $RESULT

cmp tmp2, 0

je lab17\_10

GMEMI tmp2, MEMORYOWNER

mov tmp3, $RESULT

cmp tmp3, 0

je lab17\_10

mov GPAaddr, tmp2

mov GPAequ, [paddr6+1]

lab17\_9:

mov tmp1, dllimgbase

add tmp1, 40 //dllimgbase+40

eval "jmp 0{tmp1}"

asm paddr6, $RESULT

mov [tmp1], #B8#

add tmp1, 1 //dllimgbase+41

mov [tmp1], tmp2

mov tmp3, paddr6

add tmp3, 5

add tmp1, 4 //dllimgbase+45

eval "jmp 0{tmp3}"

asm tmp1, $RESULT

lab17\_10:

mov count, 0

eob lab12

eoe lab12

esto

lab18:

bc thunkstop

bphwc thunkpt

mov [paddr1], ori1

mov [paddr1+4], ori2

cmp DFCequ, 0

je lab18\_1

mov [paddr4], #B8#

mov [paddr4+1], DFCequ

lab18\_1:

cmp REequ, 0

je lab18\_2

mov [paddr5], #B8#

mov [paddr5+1], REequ

lab18\_2:

cmp GPAequ, 0

je lab18\_3

mov [paddr6], #B8#

mov [paddr6+1], GPAequ

lab18\_3:

cmp paddr2, 0

je lab19

mov [paddr2], ori3

lab19:

mov [paddr3], ori4

fill dllimgbase, 60, 00

find dllimgbase, #8B432C2BC583E805#

mov tmp1, $RESULT

cmp tmp1, 0

je error

add tmp1, 8

mov writept2, tmp1

//log writept2

bphws writept2, "x"

find eip, #C700D4000000# //Search dword ptr [eax], 0D4"

mov 55pt, $RESULT

cmp 55pt, 0

add 55pt, 8

jne lab19\_2

find eip, #C600D485# //Search "mov byte ptr [eax], 0D4"

mov 55pt, $RESULT

cmp 55pt, 0

je lab19\_1

add 55pt, 5

jmp lab19\_2

lab19\_1:

find eip, #C600D4837D??00# //Search "mov byte ptr [eax], 0D4", "cmp [ebp-8], 0"

mov 55pt, $RESULT

cmp 55pt, 0

je error

add 55pt, 7

lab19\_2:

//log 55pt

bp 55pt

BPHWS APIpoint3, "x"

eoe lab20

eob lab20

esto

lab20:

cmp eip, APIpoint3

je lab21

cmp eip, writept2

je lab23

cmp eip, 55pt

je lab25

esto

lab21:

mov type3API, 1

cmp EBXaddr, 0

jne lab22

mov EBXaddr, ebx

//log EBXaddr

mov tmp1, [EBXaddr+4A], 1

mov FF15flag, tmp1

//log FF15flag

lab22:

bphwc APIpoint3

eob lab22\_1

eoe lab22\_1

esto

lab22\_1:

cmp eip, writept2

je lab23

cmp eip, 55pt

je lab25

esto

lab23:

bphwc writept2

cmp EBXaddr, 0

jne lab24

mov EBXaddr, ebx

//log EBXaddr

mov tmp1, [EBXaddr+4A], 1

mov FF15flag, tmp1

//log FF15flag

lab24:

mov type1API, 1

//log type1API

eob lab24\_1

eoe lab24\_1

esto

lab24\_1:

cmp eip, APIpoint3

je lab21

cmp eip, 55pt

je lab25

esto

lab25:

bphwc APIpoint3

bphwc writept2

bc 55pt

cmp !zf, 0

jne lab27\_1

sti

sti

sti

sti

mov tmp1, eax

mov tmp2, [tmp1]

//log tmp2, "55 struct = "

cmp tmp2, 0

je lab25\_1

cmp tmp2, 1

je lab25\_2

msg "未知的 55 数据结构"

//pause

//old

lab25\_1:

mov tmp2, eax

mov tmp6, [tmp2+4] //data size

add tmp6, tmp2

sub tmp6, 8 //ending address of data

add tmp2, 8

jmp lab25\_3

//new

lab25\_2:

mov 55struct1, 1

mov tmp2, eax

mov tmp6, [tmp2+6] //data size

add tmp6, tmp2

sub tmp6, 8 //ending address of data

add tmp2, 0C

lab25\_3:

alloc 1000

mov 55dataloc, $RESULT

mov tmp3, 55dataloc

loop3:

cmp tmp2, tmp6

jae lab26

mov tmp4, [tmp2]

add tmp4, imgbase

mov [tmp3], tmp4

add tmp2, 4

mov tmp5, [tmp2]

add tmp2, tmp5

add tmp2, 4

add tmp3, 4

add count, 1

cmp 55struct1, 1

je loop3\_1

jmp loop3

loop3\_1:

add tmp2, 2

jmp loop3

lab26:

coe

cob

rtr

//log count

cmp count, 1

je onefunc

cmp count, 2

je twofunc

cmp count, 5

je fivefunc

cmp count, 6

je sixfunc

cmp count, 7

je sevenfunc

lab26\_1:

sti

mov 55sc, 1

jmp lab27\_1

onefunc:

log "1 个标准函数"

mov tmp1, 55dataloc

mov tmp2, [tmp1]

mov [tmp2], #6AFF5064A100000000508B44240C64892500000000896C240C8D6C240C50C3#

jmp lab27

twofunc:

mov tmp1, 55dataloc

mov tmp2, [tmp1]

mov tmp3, [tmp1]

sub tmp3, A

mov tmp4, [tmp3]

cmp tmp4, A6F3D189

je twofunc\_1

sub tmp3, 1

mov tmp4, [tmp3]

cmp tmp4, A6F3D189

jne lab26\_1

twofunc\_1:

log "2 个标准函数"

mov [tmp2], #56575389C689D709C074038B40FC09D274038B52FC89C139D1760289D139C9F3A6742A8A5EFF80FB61720880FB7A7703#

add tmp2, 30

mov [tmp2], #80EB208A7FFF80FF61720880FF7A770380EF2038FB74D80FB6C30FB6D729D05B5F5EC3#

add tmp1, 4

mov tmp2, [tmp1]

mov [tmp2], #575689D789C6B9FFFFFFFF31C0F2AEF7D189D731D2F3A68A46FF8A57FF29D05E5FC3#

jmp lab27

fivefunc:

log "5 个标准函数"

jmp lab26\_1

sixfunc:

log "6 个标准函数"

mov tmp1, 55dataloc

mov tmp2, [tmp1]

mov tmp3, [tmp1]

sub tmp3, 30

find tmp3, #0FB646FF0FB657FF#

mov tmp4, $RESULT

cmp tmp4, 0

je lab26\_1

//log tmp4

cmp tmp4, tmp2

ja lab26\_1

mov [tmp2], #56575389C689D709C074038B40FC09D274038B52FC89C139D1760289D139C9F3A6742A8A5EFF80FB61720880FB7A7703#

add tmp2, 30

mov [tmp2], #80EB208A7FFF80FF61720880FF7A770380EF2038FB74D80FB6C30FB6D729D05B5F5EC3#

add tmp1, 4 //2nd

mov tmp2, [tmp1]

mov [tmp2], #89FA89C7B9FFFFFFFF30C0F2AEB8FEFFFFFF29C889D7C3#

add tmp1, 4 //3rd

mov tmp2, [tmp1]

mov [tmp2], #89FA89C7B9FFFFFFFF30C0F2AE8D47FF89D7C3#

add tmp1, 4 //4th

mov tmp2, [tmp1]

mov [tmp2], #575689C689D7B9FFFFFFFF30C0F2AEF7D189F789D689CA89F8C1E902F3A589D183E103F3A45E5FC3#

add tmp1, 4 //5th

mov tmp2, [tmp1]

mov [tmp2], #575689D789C6B9FFFFFFFF31C0F2AEF7D189D731D2F3A68A46FF8A57FF29D05E5FC3#

add tmp1, 4 //6th

mov tmp2, [tmp1]

mov [tmp2], #568BF08BD0AC08C074123C614172F680F87A77F180E8208846FFEBE9925EC3#

jmp lab27

sevenfunc:

log "7 个标准函数"

mov tmp1, 55dataloc

mov tmp2, [tmp1]

mov tmp3, [tmp1]

sub tmp3, B

mov tmp4, [tmp3]

cmp tmp4, A6F3D189

jne lab26\_1

mov [tmp2], #56575389C689D709C074038B40FC09D274038B52FC89C139D1760289D139C9F3A6742A8A5EFF80FB61720880FB7A7703#

add tmp2, 30

mov [tmp2], #80EB208A7FFF80FF61720880FF7A770380EF2038FB74D80FB6C30FB6D729D05B5F5EC3#

add tmp1, 4 //2nd

mov tmp2, [tmp1]

mov [tmp2], #89FA89C7B9FFFFFFFF30C0F2AEB8FEFFFFFF29C889D7C3#

add tmp1, 4 //3rd

mov tmp2, [tmp1]

mov [tmp2], #89FA89C7B9FFFFFFFF30C0F2AE8D47FF89D7C3#

add tmp1, 4 //4th

mov tmp2, [tmp1]

mov [tmp2], #565789D689C789CA39F77711742BC1E902F3A589D183E103F3A45F5EC38D740EFF8D7C0FFF83E103FDF3A483EE0383EF#

add tmp2, 30

mov [tmp2], #0389D1C1E902F3A5FC5F5EC3#

add tmp1, 4 //5th

mov tmp2, [tmp1]

mov [tmp2], #575689C689D7B9FFFFFFFF30C0F2AEF7D189F789D689CA89F8C1E902F3A589D183E103F3A45E5FC3#

add tmp1, 4 //6th

mov tmp2, [tmp1]

mov [tmp2], #575689D789C6B9FFFFFFFF31C0F2AEF7D189D731D2F3A68A46FF8A57FF29D05E5FC3#

add tmp1, 4 //7th

mov tmp2, [tmp1]

mov [tmp2], #57565309C0744409D2744089C389D730C0B9FFFFFFFFF2AEF7D149742E89CE89DFB9FFFFFFFFF2AEF7D129F1761D89DF#

add tmp2, 30

mov [tmp2], #8D5EFF89D6ACF2AE751189C85789D9F3A65F89C175ED8D47FFEB0231C05B5E5FC3#

lab27:

sti

lab27\_1:

cob

coe

find dllimgbase, #0036300D0A#

mov tmp6, $RESULT

cmp tmp6, 0

je error

mov tmp3, tmp6

sub tmp3, 90

find tmp3, #C600??#

mov tmp2, $RESULT

cmp tmp2, 0

je lab27\_2

cmp tmp2, tmp6

jb lab27\_3

lab27\_2:

find tmp3, #C700D?000000#

mov tmp2, $RESULT

cmp tmp2, 0

je error

cmp tmp2, tmp6

ja error

lab27\_3:

find tmp2, #74??#

mov tmp4, $RESULT

cmp tmp4, 0

je error

cmp tmp4, tmp6

ja error

mov transit1, tmp4

//log transit1

find eip, #C700D5000000#

mov tmp3, $RESULT

cmp tmp3, 0

add tmp3, 8

jne lab27\_4

find eip, #C600D5#

mov tmp1, $RESULT

cmp tmp1, 0

je error

find tmp1, #74??#

mov tmp3, $RESULT

cmp tmp3, 0

je error

lab27\_4:

eob lab27\_5

eoe lab27\_5

bp tmp3

esto

lab27\_5:

cmp eip, tmp3

je lab27\_6

esto

lab27\_6:

bc tmp3

cmp !zf, 0

jne lab28

//Collect SDK stolen code

find dllimgbase, #C603E98D5301#

mov 57jmppt, $RESULT

cmp 57jmppt, 0

je error

bp 57jmppt

mov xtrascloc, dllimgbase

add xtrascloc, 0F00 //dllimgbase+F00

//log xtrascloc

//log 57pt

bp 57pt

mov tmp4, xtrascloc

mov tmp5, dllimgbase

add tmp5, 300 //dllimgbase+300

mov tmp9, dllimgbase

add tmp9, 500 //dllimgbase+500

mov tmp8, dllimgbase

mov tmp7, 0 //counter

lab28:

bp transit1

eob lab28\_1

eoe lab28\_1

esto

lab28\_1:

cmp eip, 57pt

je lab29

cmp eip, 57jmppt

je lab30

cmp eip, transit1

je lab31

esto

//Get total SDK sections and collect address of scstk

lab29:

cmp sdksccount, 0

jne lab29\_9

find eip, #8BE55DC2??00#

mov tmp1, $RESULT

cmp tmp1, 0

je error

mov tmp2, [tmp1+4], 1

cmp tmp2, 08

jne lab29\_1

mov sdksccount, [ebp-0c]

log sdksccount, "SDK 偷代码区段总数 = "

mov tmp1, [esp]

GMEMI tmp1, MEMORYBASE

mov tmp10, $RESULT

jmp lab29\_2

lab29\_1:

cmp tmp2, 0c

jne error

mov sdksccount, [ebp-10]

log sdksccount, "SDK 偷代码区段 = "

mov tmp1, [esp+4]

GMEMI tmp1, MEMORYBASE

mov tmp10, $RESULT

lab29\_2:

cmp tmp7, 0

jne lab29\_9

mov tmp1, [tmp10+4], 2

cmp tmp1, 0

je lab29\_6

cmp tmp1, 1

jne lab29\_3

add tmp10, 0E

jmp lab29\_4

//Aspr 2.3 Build6.26

lab29\_3:

mov tmp1, [tmp10+4]

mov tmp2, [tmp10+0E]

cmp tmp1, tmp2

jne error //unknown aspr version

mov tmp1, [tmp10+8], 2

cmp tmp1, 1

jne error //unknown aspr version

mov tmp2, [tmp10+12], 2

cmp tmp1, tmp2

jne error //unknown aspr version

add tmp10, 12

lab29\_4:

mov tmp1, [tmp10], 2

cmp tmp1, 01

jne lab29\_9

mov tmp2, [tmp10+6]

cmp tmp2, 0

je lab29\_9

mov tmp1, [tmp10+2]

cmp tmp1, 0

je lab29\_9

add tmp1, imgbase

mov [tmp8], tmp1

add tmp8, 4

add tmp10, tmp2

add tmp10, 0A

cmp tmp2, 1000

ja lab29\_5

add SDKsize, 1000

jmp lab29\_4

lab29\_5:

and tmp2, FFFFF000

add tmp2, 1000

add SDKsize, tmp2

jmp lab29\_4

lab29\_6:

add tmp10, 0C

lab29\_7:

mov tmp2, [tmp10+4]

cmp tmp2, 0

je lab29\_9

mov tmp1, [tmp10]

cmp tmp1, 0

je lab29\_9

add tmp1, imgbase

mov [tmp8], tmp1

add tmp8, 4

add tmp10, tmp2

add tmp10, 08

cmp tmp2, 1000

ja lab29\_8

add SDKsize, 1000

jmp lab29\_7

lab29\_8:

and tmp2, FFFFF000

add tmp2, 1000

add SDKsize, tmp2

jmp lab29\_7

lab29\_9:

mov [tmp4], eax

add tmp7, 1 //counter

mov tmp1, [ebx]

add tmp1, imgbase

mov [tmp5], tmp1

add tmp4, 4

add tmp5, 4

eob lab28\_1

eoe lab28\_1

esto

lab30:

mov tmp1, dllimgbase

add tmp1, 500 //dllimgbase+500

mov tmp2, [tmp1]

cmp tmp2, 0

jne lab30\_3

//Decide the structure of jmp table and dump it

mov tmp2, edi

mov jmptablesize, 0

mov tmp1, [edi], 2

cmp tmp1, 1

je lab30\_2

mov tmp1, [edi]

mov tmp3, [edi+8]

cmp tmp1, tmp3

jne lab30\_1

mov 57struct, "57A"

jmp lab30\_3

lab30\_1:

mov 57struct, "57C"

jmp lab30\_3

lab30\_2:

mov 57struct, "57B"

//copy data

lab30\_3:

scmp 57struct, "57A"

je lab30\_4

scmp 57struct, "57B"

je lab30\_6

scmp 57struct, "57C"

je lab30\_8

jmp error

lab30\_4:

bc 57jmppt

cob

coe

mov tmp1, dllimgbase

add tmp1, 100

mov [tmp1], #609C8BF7BF0005C0008B06394608750F8B4E04890F83C60883C704F2A4EBEA893D400122019D61909090#

mov tmp1, dllimgbase

add tmp1, 100

add tmp1, 5 //105

mov tmp2, dllimgbase

add tmp2, 500

mov [tmp1], tmp2

add tmp1, 1C //121

mov tmp2, dllimgbase

add tmp2, 140

mov [tmp1], tmp2

add tmp1, 6 //127--end point

bp tmp1

mov ori1, eip

mov tmp2, dllimgbase

add tmp2, 100

mov eip, tmp2

run

cmp eip, tmp1

jne error

bc tmp1

mov tmp2, [dllimgbase+140]

mov tmp3, dllimgbase

add tmp3, 500

sub tmp2, tmp3

mov jmptablesize, tmp2

mov eip, ori1

mov tmp2, dllimgbase

add tmp2, 100

fill tmp2, 44, 00

jmp lab30\_12

lab30\_6:

bc 57jmppt

cob

coe

mov tmp1, dllimgbase

add tmp1, 100

mov [tmp1], #609C8BF7BF0005C9008B460283F800741439460A750F8B4E06890F83C60A83C704F2A4EBE4893D4001C9009D61909000#

mov tmp1, dllimgbase

add tmp1, 100

add tmp1, 5 //105

mov tmp2, dllimgbase

add tmp2, 500

mov [tmp1], tmp2

add tmp1, 22 //127

mov tmp2, dllimgbase

add tmp2, 140

mov [tmp1], tmp2

add tmp1, 6 //12D--end point

bp tmp1

mov ori1, eip

mov tmp2, dllimgbase

add tmp2, 100

mov eip, tmp2

run

cmp eip, tmp1

jne error

bc tmp1

mov tmp2, [dllimgbase+140]

mov tmp3, dllimgbase

add tmp3, 500

sub tmp2, tmp3

mov jmptablesize, tmp2

mov eip, ori1

mov tmp2, dllimgbase

add tmp2, 100

fill tmp2, 44, 00

jmp lab30\_12

lab30\_8:

mov tmp2, [edi]

add tmp2, imgbase

cmp tmp2, ebx

jne lab30\_12

mov ori1, edi

find ori1, #0000000000000000#

mov tmp3, $RESULT

cmp tmp3, 0

je error

sub tmp3, ori1

mov tmp2, tmp3

shr tmp2, 2

shl tmp2, 2

cmp tmp3, tmp2

je lab30\_9

shr tmp3, 2

add tmp3, 1

shl tmp3, 2

lab30\_9:

add jmptablesize, tmp3 //bytes to copy

add jmptablesize, 0C

mov tmp2, tmp3

add tmp2, 8

mov [tmp9], tmp2

add tmp9, 4

lab30\_10:

cmp tmp3, 0

je lab30\_11

mov tmp1, [ori1]

mov [tmp9], tmp1

add ori1, 4

add tmp9, 4

sub tmp3, 4

jmp lab30\_10

lab30\_11:

add tmp9, 8 //add 8 bytes for differentiation

lab30\_12:

eob lab28\_1

eoe lab28\_1

esto

lab31:

cmp sdksccount, 0

je lab32

//log SDKsize

//log jmptablesize

mov tmp1, dllimgbase

add tmp1, 500

dm tmp1, jmptablesize, "jmptable.bin"

cmp sdksccount, tmp7 //tmp7=number of section with scstk

je lab31\_1

log tmp7, "带 scstk 的 SDK 区段 = "

mov tmp1, dllimgbase //Location of full set address

mov tmp2, tmp1

add tmp2, 300 //Location of section with scstk

mov tmp9, xtrascloc //store SDK section without scstk

add tmp9, 80

//find out which SDK section need dumping

loop4:

mov tmp3, [tmp1]

cmp tmp3, 0

je lab31\_1 //compare finished

loop4\_1:

mov tmp4, [tmp2]

cmp tmp4, 0

je loop4\_2 //not found

cmp tmp3, tmp4

je loop4\_3 //jmp if found

add tmp2, 4

jmp loop4\_1

//section need to be dump manually found

loop4\_2:

mov tmp6, [tmp1]

mov tmp5, [tmp6+1]

add tmp5, tmp6

add tmp5, 5

log tmp5, "SDK 偷代码区段地址 = "

mov [tmp9], tmp6 //store SDK section without scstk

add tmp9, 4

mov [tmp9], tmp5

add tmp9, 4

add tmp1, 4

mov tmp2, dllimgbase

add tmp2, 300 //Location of section with scstk

jmp loop4

loop4\_3:

add tmp1, 4

mov tmp2, dllimgbase

add tmp2, 300 //Location of section with scstk

jmp loop4

//end compare

lab31\_1:

fill dllimgbase, B00, 00

lab32:

bc 57pt

bc 57jmppt

bc transit1

cmp !zf, 0

jne lab41

sti

sti

sti

mov countaddr, [eax]

add countaddr, imgbase

log countaddr, "Delphi 初始化表的地址 "

find dllimgbase, #55FFD784C07504#

mov tmp1, $RESULT

cmp tmp1, 0

je error

find tmp1, #837D0?0075E5#

mov tmp3, $RESULT

cmp tmp3, 0

je error

sub tmp3, 2

mov tmp2, dllimgbase

bp tmp3

mov tmp4, 0 //counter

eob lab32\_1

eoe lab32\_1

esto

lab32\_1:

cmp eip, tmp3

je lab32\_2

esto

lab32\_2:

mov [tmp2], edx

cmp tmp4, 2

je lab32\_3

add tmp2, 4

add tmp4, 1

esto

lab32\_3:

bc tmp3

cob

coe

rtr

sti

rtr

sti

rtr

mov tablea, [dllimgbase]

mov tableb, [dllimgbase+4]

mov decryptaddr, [dllimgbase+8]

fill dllimgbase, 10, 00

alloc 4000

mov dataloc, $RESULT

//log dataloc

find decryptaddr, #81??????????0F84????00005?5?#

mov tmp1, $RESULT

cmp tmp1, 0

je error

add tmp1, 0C

mov paddr1, tmp1

//log paddr1

mov ori1, [paddr1]

mov ori2, [paddr1+4]

//log ori1

//log ori2

find paddr1, #E8????0000#

mov tmp1, $RESULT

cmp tmp1, 0

je error

mov tmp9, tmp1

mov tmp2, [tmp1+1]

add tmp2, tmp1

add tmp2, 5

find tmp2, #3B??0F82??FFFFFF#

mov tmp3, $RESULT

cmp tmp3, 0

je error

mov paddr2, tmp3

//log paddr2

mov tmp2, [tmp3+4]

add tmp2, tmp3

add tmp2, 8

mov tmp1, [tmp2], 1

cmp tmp1, 2B

je lab32\_4

find tmp2, #2B??#

mov tmp1, $RESULT

cmp tmp1, 0

je error

cmp paddr2, tmp1

jb error

opcode tmp1

mov tmp5, $RESULT\_2

add tmp5, tmp1

jmp lab32\_9

lab32\_4:

opcode tmp2

mov tmp5, $RESULT\_2

add tmp5, tmp2

lab32\_9:

mov ori3, [paddr2]

mov tmp1, dllimgbase

mov [tmp1], #609CB800004000B900104000BA00204000BB00304000BD00404000BE00504000BF00604000E80001300090909D619090#

mov tmp1, dllimgbase

mov tmp6, imgbase

add tmp1, 3 //3

mov [tmp1], tmp6

add tmp6, 1000

add tmp1, 5 //8

mov [tmp1], tmp6

add tmp6, 1000

add tmp1, 5 //D

mov [tmp1], tmp6

add tmp6, 1000

add tmp1, 5 //12

mov [tmp1], tmp6

add tmp6, 2000

add tmp1, 5 //17

mov [tmp1], tmp6

add tmp6, 1000

add tmp1, 5 //1C

mov [tmp1], tmp6

add tmp6, 1000

add tmp1, 5 //21

mov [tmp1], tmp6

add tmp1, 4 //25

eval "call 0{tmp5}"

asm tmp1, $RESULT

mov [paddr2], #C390#

mov tmp7, eip

mov tmp6, esp

mov eip, dllimgbase

bp paddr2

eob lab33

eoe lab33

run

lab33:

cmp eip, paddr2

je lab33\_1

jmp error

lab33\_1:

bc paddr2

mov tmp1, tmp6

sub tmp1, 28

mov esp, tmp1

sti

mov tmp1, imgbase

cmp eax, tmp1

je ecxchk

mov tmp8, eax

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

ecxchk:

add tmp1, 1000

cmp ecx, tmp1

je edxchk

mov tmp8, ecx

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

edxchk:

add tmp1, 1000

cmp edx, tmp1

je ebxchk

mov tmp8, edx

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

ebxchk:

add tmp1, 1000

cmp ebx, tmp1

je ebpchk

mov tmp8, ebx

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

ebpchk:

add tmp1, 2000

cmp ebp, tmp1

je esichk

mov tmp8, ebp

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

esichk:

add tmp1, 1000

cmp esi, tmp1

je edichk

mov tmp8, esi

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

edichk:

add tmp1, 1000

cmp edi, tmp1

je edxchk

mov tmp8, edi

sub tmp8, tmp1

cmp tmp8, 10

jbe lab34

jmp error

lab34:

cob

coe

mov tmp1, dllimgbase

add tmp1, 2e

bp tmp1

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7

mov [paddr2], ori3 //restore code

fill dllimgbase, 50, 00

mov tmp7, eip

mov tmp1, dllimgbase

mov [tmp1], #609CB90000FD01BA00001602BD00001802BE0000170233C08B3983FF00743281FF72E9EFB9741F8BDE03322B312B0390#

add tmp1, 30 //30

mov [tmp1], #909090909090909090909090903BDE72EC03C789450083C50883C10883C208EBC0833DA000BA0001741BB90400FD01BA#

add tmp1, 30 //60

mov [tmp1], #04001602BD04001802C705A000BA0001000000EB9C9D61909000000000000000#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov [tmp1], tablea

add tmp1, 5 //8

mov [tmp1], tableb

add tmp1, 5 //D

mov [tmp1], dataloc

add tmp1, 5 //12

mov [tmp1], decryptaddr

find tablea, #0000000000000000#

mov tmp2, $RESULT

cmp tmp2, 0

je error

mov dataendaddr, tmp2

sub tmp2, 8

mov tmp3, [tmp2] //data limit

add tmp1, 0F //21

mov [tmp1], tmp3

add tmp1, 10 //31

eval "add ebx, 0{tmp8}"

asm tmp1, $RESULT

mov tmp3, dllimgbase

add tmp3, A0

add tmp1, 22 //53

mov [tmp1], tmp3

add tmp1, 8 //5B

mov tmp2, tablea

add tmp2, 4

mov [tmp1], tmp2

add tmp1, 5 //60

mov tmp2, tableb

add tmp2, 4

mov [tmp1], tmp2

add tmp1, 5 //65

mov tmp2, dataloc

add tmp2, 4

mov [tmp1], tmp2

add tmp1, 6 //6B

mov [tmp1], tmp3

mov tmp5, dllimgbase

add tmp5, 77 //end point

mov eip, dllimgbase

bp tmp5

eob lab34\_1

eoe lab34\_1

esto

lab34\_1:

cmp eip, tmp5

je lab34\_2

esto

lab34\_2:

bc tmp5

mov eip, tmp7

fill dllimgbase, 100, 00

find paddr2, #5?5?5?E9??F?FFFF#

mov tmp1, $RESULT

cmp tmp1, 0

je error

mov paddr3, tmp1

//log paddr3

find paddr1, #FFD0# //"call eax" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryecx

cmp paddr4, paddr2

jb iscalleax

tryecx:

find paddr1, #FFD1# //"call ecx" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryedx

cmp paddr4, paddr2

jb iscallecx

tryedx:

find paddr1, #FFD2# //"call edx" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryebx

cmp paddr4, paddr2

jb iscalledx

tryebx:

find paddr1, #FFD3# //"call ebx" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryesp

cmp paddr4, paddr2

jb iscallebx

tryesp:

find paddr1, #FFD4# //"call esp" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryebp

cmp paddr4, paddr2

jb iscallesp

tryebp:

find paddr1, #FFD5# //"call ebp" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryesi

cmp paddr4, paddr2

jb iscallebp

tryesi:

find paddr1, #FFD6# //"call esi" ?

mov paddr4, $RESULT

cmp paddr4, 0

je tryedi

cmp paddr4, paddr2

jb iscallesi

tryedi:

find paddr1, #FFD7# //"call edi" ?

mov paddr4, $RESULT

cmp paddr4, 0

je hexfind2

cmp paddr4, paddr2

jb iscalledi

hexfind2:

log tmp9

mov tmp1, [tmp9+1]

add tmp1, tmp9

sub tmp1, 50

mov tmp4, 50

loop5:

cmp tmp4, 0

je error

mov tmp2, [tmp1]

and tmp2, f0ff

cmp tmp2, 0000D0ff

je hexfound2

sub tmp4, 1

add tmp1, 1

jmp loop5

hexfound2:

mov paddr4, tmp1

//log paddr4

mov tmp2, [paddr4+1]

and tmp2, 0f

cmp tmp2, 0

je iscalleax

cmp tmp2, 1

je iscallecx

cmp tmp2, 2

je iscalledx

cmp tmp2, 3

je iscallebx

cmp tmp2, 4

je iscallesp

cmp tmp2, 5

je iscallebp

cmp tmp2, 6

je iscallesi

cmp tmp2, 7

je iscalledi

jmp error

iscalleax:

mov caller1, "eax"

jmp lab35

iscallecx:

mov caller1, "ecx"

jmp lab35

iscalledx:

mov caller1, "edx"

jmp lab35

iscallebx:

mov caller1, "ebx"

jmp lab35

iscallesp:

mov caller1, "esp"

jmp lab35

iscallebp:

mov caller1, "ebp"

jmp lab35

iscallesi:

mov caller1, "esi"

jmp lab35

iscalledi:

mov caller1, "edi"

lab35:

mov paddr5, paddr1

sub paddr5, 4

mov ori6, [paddr5]

mov tmp1, dllimgbase

mov tmp2, dllimgbase

add tmp2, 100 //dllimgbase+100

mov [tmp2], dataloc

mov tmp3, tmp2

add tmp3, 4 //dllimgbase+104

mov tmp5, dataloc

add tmp5, 2008

mov [tmp3], tmp5

mov tmp4, dllimgbase

add tmp4, 7A //dllimgbase+7A

mov [tmp1], #609C68000040006800001602680000FD01E8EAFF5C01832D0401BA0004C6057A00BA002DC605D800BA002DC7050001BA#

add tmp1, 30 //30

mov [tmp1], #000400180268000040006804001602680400FD01E8B2FF5C01EB5590000000008B050001BA008B00909083050001BA00#

add tmp1, 30 //60

mov [tmp1], #0890E92C015D01000000000000009090538B1D0401BA00890383050401BA00085B909090909090909090909090909090#

add tmp1, 30 //90

mov [tmp1], #00000000000000000000000000000000BE00201802BFD8214D00B92E010000F2A5B8D8214D00C70096000000C74004E0#

add tmp1, 30 //C0

mov [tmp1], #214D009D61909000000000000000009083050001BA000883050401BA0008E9B8005D0100000000000000000000000000#

mov tmp1, dllimgbase

add tmp1, 3

mov [tmp1], imgbase

add tmp1, 5 //8

mov [tmp1], tableb

add tmp1, 5 //0D

mov [tmp1], tablea

add tmp1, 4 //11

eval "call 0{decryptaddr}"

asm tmp1, $RESULT

add tmp1, 7 //18

mov [tmp1], tmp3

add tmp1, 7 //1F

mov [tmp1], tmp4 //tmp4=dllimgbase+7A

add tmp1, 7 //26

add tmp4, 5E //tmp4=dllimgbase+D8

mov [tmp1], tmp4

add tmp1, 7 //2D

mov [tmp1], tmp2

add tmp1, 4 //31

mov tmp5, dataloc

add tmp5, 4

mov [tmp1], tmp5

add tmp1, 5 //36

mov [tmp1], imgbase

add tmp1, 5 //3B

mov tmp5, tableb

add tmp5, 4

mov [tmp1], tmp5

add tmp1, 5 //40

mov tmp5, tablea

add tmp5, 4

mov [tmp1], tmp5

add tmp1, 4 //44

eval "call 0{decryptaddr}"

asm tmp1, $RESULT

add tmp1, 0E //52

mov [tmp1], tmp2

add tmp1, A //5C

mov [tmp1], tmp2

add tmp1, 5 //61

eval "jmp 0{paddr3}"

asm tmp1, $RESULT

add tmp1, 12 //73

mov [tmp1], tmp3

add tmp1, 8 //7B

mov [tmp1], tmp3

mov tmp5, dllimgbase

add tmp5, 50

eval "jmp 0{tmp5}"

asm paddr1, $RESULT

mov tmp1, dllimgbase

add tmp1, 50 //50

scmpi caller1, "eax"

je lab35\_1

scmpi caller1, "ecx"

je writeecx

scmpi caller1, "edx"

je writeedx

scmpi caller1, "ebx"

je writeebx

scmpi caller1, "esp"

je writeesp

scmpi caller1, "ebp"

je writeebp

scmpi caller1, "esi"

je writeesi

scmpi caller1, "edi"

je writeedi

jmp error

writeecx:

mov [tmp1], #8B0D#

add tmp1, 6 //56

asm tmp1, "mov ecx, [ecx]"

add tmp1, 21 //77

mov [tmp1], #890B#

jmp lab35\_1

writeedx:

mov [tmp1], #8B15#

add tmp1, 6 //56

asm tmp1, "mov edx, [edx]"

add tmp1, 21 //77

mov [tmp1], #8913#

jmp lab35\_1

writeebx:

mov [tmp1], #8B1D#

add tmp1, 6 //56

asm tmp1, "mov ebx, [ebx]"

add tmp1, 1A //70

asm tmp1, "push eax"

add tmp1, 1 //71

mov [tmp1], #8B05#

add tmp1, 6 //77

mov [tmp1], #8918#

add tmp1, 9 //80

asm tmp1, "pop eax"

jmp lab35\_1

writeesp:

mov [tmp1], #8B25#

add tmp1, 6 //56

asm tmp1, "mov esp, [esp]"

add tmp1, 21 //77

mov [tmp1], #8923#

jmp lab35\_1

writeebp:

mov [tmp1], #8B2D#

add tmp1, 6 //56

mov [tmp1], #8B6D0090#

add tmp1, 21 //77

mov [tmp1], #892B#

jmp lab35\_1

writeesi:

mov [tmp1], #8B35#

add tmp1, 6 //56

asm tmp1, "mov esi, [esi]"

add tmp1, 21 //77

mov [tmp1], #8933#

jmp lab35\_1

writeedi:

mov [tmp1], #8B3D#

add tmp1, 6 //56

asm tmp1, "mov edi, [edi]"

add tmp1, 21 //77

mov [tmp1], #893B#

lab35\_1:

mov tmp1, dllimgbase

add tmp1, 83 //83

mov ori3, [paddr4]

mov ori4, [paddr4+4]

mov ori5, [paddr4+8]

mov tmp5, paddr4

add tmp5, 2

opcode tmp5

mov tmp4, $RESULT\_2 //length of 1st cmd after call reg

cmp tmp4, 3

jae lab35\_14

cmp tmp4, 1

je lab35\_3

//length of 1st cmd = 2

mov tmp6, [tmp5], 2

cmp tmp6, 1EB

je lab35\_2

cmp tmp6, 2EB

jne lab35\_4

lab35\_2:

mov tmp3, [tmp5+1], 1

add tmp4, tmp3

add tmp4, tmp5

eval "jmp 0{tmp4}"

asm tmp1, $RESULT

jmp lab36\_1

//length of 1st cmd = 1

lab35\_3:

mov tmp3, [tmp5]

and tmp3, 00F0FFF0

cmp tmp3, 0EBF0 //"prefix ??", "jmp ???????"

jne lab35\_4

mov tmp3, [tmp5+2], 1

add tmp3, tmp5

add tmp3, tmp4

add tmp3, 2

eval "jmp 0{tmp3}"

asm tmp1, $RESULT

jmp lab36\_1

//2nd cmd after call reg

lab35\_4:

mov tmp6, tmp5

add tmp6, tmp4

opcode tmp6

mov tmp8, $RESULT\_2 //length of 2nd cmd after call reg

mov tmp2, tmp4

add tmp4, tmp8

cmp tmp8, 2

je lab35\_5

cmp tmp8, 3

je lab35\_7

cmp tmp4, 3

jae copybyte

jmp lab35\_9

//length of 2nd cmd = 2

lab35\_5:

mov tmp3, [tmp6], 2

cmp tmp3, 1EB

je lab35\_6

cmp tmp3, 2EB

je lab35\_6

cmp tmp4, 3

jae copybyte

jmp lab35\_9

lab35\_6:

opcode tmp5

mov tmp3, $RESULT\_1

eval "{tmp3}"

asm tmp1, $RESULT

add tmp1, tmp8

mov tmp3, 0 //For Odbgscript compatibility

mov tmp3, [tmp6+1], 1

add tmp2, tmp3

add tmp2, tmp8

add tmp2, tmp5

eval "jmp 0{tmp2}"

asm tmp1, $RESULT

jmp lab36\_1

//length of 2nd cmd = 3

lab35\_7:

mov tmp3, [tmp6+1], 2

cmp tmp3, 1EB

je lab35\_8

cmp tmp3, 2EB

je lab35\_8

cmp tmp4, 3

jae copybyte

jmp lab35\_9

lab35\_8:

opcode tmp5

mov tmp3, $RESULT\_1

eval "{tmp3}"

asm tmp1, $RESULT

add tmp1, tmp8

mov tmp3, 0 //For Odbgscript compatibility

mov tmp3, [tmp6+2], 1

add tmp2, tmp3

add tmp2, tmp8

add tmp2, tmp5

eval "jmp 0{tmp2}"

asm tmp1, $RESULT

jmp lab36\_1

//3rd cmd after call reg

lab35\_9:

mov tmp7, tmp6

add tmp7, tmp8

opcode tmp7

mov tmp9, $RESULT\_2 //length of 3rd cmd after call reg

add tmp4, tmp9

cmp tmp9, 2

je lab35\_10

cmp tmp9, 3

je lab35\_12

jmp copybyte

//length of 3rd cmd = 2

lab35\_10:

mov tmp3, [tmp7], 2

cmp tmp3, 1EB

je lab35\_11

cmp tmp3, 2EB

je lab35\_11

jmp copybyte

lab35\_11:

mov tmp3, [tmp5], 2

mov [tmp1], tmp3

add tmp1, 2

mov tmp3, [tmp7+1], 1

add tmp2, tmp3

add tmp2, tmp8

add tmp2, tmp9

add tmp2, tmp5

eval "jmp 0{tmp2}"

asm tmp1, $RESULT

jmp lab36\_1

//length of 3rd cmd = 3

lab35\_12:

mov tmp3, [tmp7+1], 2

cmp tmp3, 1EB

je lab35\_13

cmp tmp3, 2EB

je lab35\_13

jmp copybyte

lab35\_13:

mov tmp3, [tmp5], 2

mov [tmp1], tmp3

add tmp1, 2

mov tmp3, [tmp7+2], 1

add tmp2, tmp3

add tmp2, tmp8

add tmp2, tmp9

add tmp2, tmp5

eval "jmp 0{tmp2}"

asm tmp1, $RESULT

jmp lab36\_1

//one command to copy

lab35\_14:

cmp tmp4, 3

jne copybyte

//length of 1st cmd = 3

mov tmp3, [tmp5+1]

and tmp3, 0F0FF

cmp tmp3, EB

je lab35\_15

jmp copybyte

lab35\_15:

mov tmp3, [tmp5+2], 1

add tmp3, tmp5

add tmp3, tmp4

eval "jmp 0{tmp3}"

asm tmp1, $RESULT

jmp lab36\_1

copybyte:

mov tmp6, tmp5 //paddr4+2

mov tmp7, tmp1 //patch addr in dllimgbase

mov tmp3, tmp4 //ttl bytes to copy

shr tmp3, 2

mov tmp2, tmp3

shl tmp2, 2

cmp tmp4, tmp2

je copybyte\_1

add tmp3, 1

copybyte\_1:

cmp tmp3, 0

je lab36

mov tmp2, [tmp6]

mov [tmp7], tmp2

sub tmp3, 1

add tmp6, 4

add tmp7, 4

jmp copybyte\_1

lab36:

add tmp1, tmp4

add tmp5, tmp4

eval "jmp 0{tmp5}"

asm tmp1, $RESULT

lab36\_1:

mov tmp1, dllimgbase

add tmp1, 70

eval "jmp 0{tmp1}"

asm paddr4, $RESULT

//

mov tmp1, dllimgbase

add tmp1, D2

mov tmp2, dllimgbase

add tmp2, 100

mov [tmp1], tmp2

add tmp1, 7 //D9

add tmp2, 4

mov [tmp1], tmp2

add tmp1, 5 //DE

mov tmp2, paddr5

sub tmp2, 2

mov tmp3, tmp2

add tmp2, ori6

add tmp2, 6

eval "jmp 0{tmp2}"

asm tmp1, $RESULT

mov tmp1, dllimgbase

add tmp1, D0

eval "jz 0{tmp1}"

asm tmp3, $RESULT

//for move data

mov tmp1, dllimgbase

add tmp1, 0A1 //A1

mov tmp2, dataloc

add tmp2, 2000

mov [tmp1], tmp2

add tmp1, 5 //A6

mov [tmp1], countaddr

add tmp1, 5 //AB

mov tmp2, dataendaddr

sub tmp2, tablea

add tmp2, 8

shr tmp2, 2

mov [tmp1], tmp2

add tmp1, 7 //B2

mov [tmp1], countaddr

add tmp1, 6 //B8

mov tmp2, dataendaddr

sub tmp2, tablea

shr tmp2, 3

mov [tmp1], tmp2

add tmp1, 7 //BF

mov tmp2, countaddr

add tmp2, 8

mov [tmp1], tmp2

mov tmp7, eip

mov eip, dllimgbase

mov tmp1, dllimgbase

add tmp1, C5 //end point

bp tmp1

eob lab36\_2

eoe lab36\_2

esto

lab36\_2:

cmp eip, tmp1

je lab36\_3

esto

lab36\_3:

//msg "Delphi 初始化表修复完毕"

bc tmp1

//Restore original code

mov tmp2, paddr1

mov [tmp2], ori1

add tmp2, 4

mov [tmp2], ori2

mov tmp2, paddr4

mov [tmp2], ori3

add tmp2, 4

mov [tmp2], ori4

add tmp2, 4

mov [tmp2], ori5

mov [paddr5], ori6

mov caller1, "nil"

mov eip, tmp7

fill dllimgbase, 110, 00

jmp lab41\_1

lab41:

cob

coe

rtr

lab41\_1:

cmp type3API, 0

je lab46

//fix type3 API

mov tmp4, APIpoint3

sub tmp4, 100

find tmp4, #05FF000000508BC3#

mov tmp1, $RESULT

cmp tmp1, 0

je error

add tmp1, 8

opcode tmp1

mov func1, $RESULT\_1

//log func1

add tmp1, 5

find tmp1, #8BC3E8??#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 2

opcode tmp2

mov func2, $RESULT\_1

//log func2

add tmp2, 5

find tmp2, #8BC3E8??#

mov tmp1, $RESULT

cmp tmp1, 0

je error

add tmp1, 2

opcode tmp1

mov func3, $RESULT\_1

//log func3

mov tmp3, [tmp1-D], 1

cmp tmp3, 50

je lab42

mov v1.32, 1

//log v1.32

lab42:

mov tmp1, dllimgbase

mov [tmp1], #60BB6806CA00BD000DC4008B73548D7B408B43188945608B83E000000089453433C08A078D04408B4C83688BC6FFD18B#

add tmp1, 30 //30

mov [tmp1], #C8034B24038BE000000033C08A47098D04408B5483688BC6FFD2807B20000F854C0100003C010F8544010000894D7033#

add tmp1, 30 //60

mov [tmp1], #C08A47078D04408B5483688BC6FFD289452433C08A47088D04408B5483688BC6FFD289452833C08A47028D04408B5483#

add tmp1, 30 //90

mov [tmp1], #688BC6FFD289453C33C08A47068D04408B5483688BC6FFD28845408B83E000000001453C8B453C5033C08A454005FF00#

add tmp1, 30 //C0

mov [tmp1], #0000508BC3E85A6A03008BC88B53108BC3E8725803008B552403553403D08955248B55282B55342BD089552833C08A47#

add tmp1, 30 //F0

mov [tmp1], #038D04408B5483688BC6FFD28945348B83E000000001453433C08A47018D04408B5483688BC6FFD28845388D452C5066#

add tmp1, 30 //120

mov [tmp1], #8B4D24668B55288BC3E8126503008B552C0393E0000000909090909060E82E00000066B9FF153E8A4538363A434A7405#

add tmp1, 30 //150

mov [tmp1], #6681C100108B457066890883C002893061EB3A00000000000000000000000090BEE02150003916740D83C60481FE3C2A#

add tmp1, 30 //180

mov [tmp1], #0210770FEBEF81EE0000400081C600004000C390900000000000000000FF4568FF4D6003B3E4000000837D60000F876D#

add tmp1, 30 //1B0

mov [tmp1], #FEFFFF6190#

mov tmp1, dllimgbase

mov tmp2, dllimgbase

add tmp2, 0D00 //dllimgbase+D00

mov tmp3, dllimgbase

add tmp3, 0D68 //Dllimgbase+D68

add tmp1, 2 //2

mov [tmp1], EBXaddr

add tmp1, 5 //7

mov [tmp1], tmp2

add tmp1, BE //C5

eval "{func1}"

asm tmp1, $RESULT

add tmp1, 0C //D1

eval "{func2}"

asm tmp1, $RESULT

add tmp1, 58 //129

eval "{func3}"

asm tmp1, $RESULT

add tmp1, 48 //171

mov [tmp1], iatstartaddr

add tmp1, D //17E

mov [tmp1], iatendaddr

add tmp1, A //188

mov [tmp1], imgbase

add tmp1, 6 //18E

mov [tmp1], imgbasefromdisk

add tmp1, 5 //193 error point

mov tmp5, tmp1

bp tmp5

add tmp1, 21 //1B4 end point

mov tmp6, tmp1

bp tmp6

mov tmp7, eip //store eip

cmp v1.32, 1

jne lab43

mov tmp1, dllimgbase

add tmp1, 11B //dllimgbase+11B

mov [tmp1], #90909090#

add tmp1, 13 //dllimgbase+12E

mov [tmp1], #8BD090909090909090#

lab43:

mov eip, dllimgbase

eob lab44

eoe lab44

run

lab44:

cmp eip, tmp5 //error

je lab60

cmp eip, tmp6 //OK

je lab45

jmp error

lab45:

bc tmp5

bc tmp6

//msg "type3 API 修复完毕"

//pause

mov type3count, [tmp3]

//log type3count

fill dllimgbase, 0E00, 00

mov eip, tmp7 //restore eip

lab46:

cmp AsprAPIloc, 0

je lab52

cmp Aspr1stthunk, 0 //VB app ?

je lab52

mov count, 120 //Need free space 120 bytes for 2.xx

call FindEMUAddr

//$$$ fix Asprotect API $$$

lab46\_1:

//chk number of API

mov tmp5, 0 //counter

mov tmp6, Aspr1stthunk

mov tmp1, AsprAPIloc

add tmp1, 4

mov caller, "lab46\_1"

lab46\_2:

mov tmp2, [tmp1]

GMEMI tmp2, MEMORYOWNER

mov tmp3, $RESULT

cmp tmp3, dllimgbase

jne lab46\_3

add tmp5, 1

add tmp1, 4

jmp lab46\_2

lab46\_3:

log tmp5, "这版的 Asprotect 其 SDk API 总数 = "

lab47:

mov tmp10, 0

cmp tmp5, 0B

je loop8

cmp tmp5, 0C

je loop9

cmp tmp5, 0D

je loop10

msg "未知的 Asprotect SDK API"

jmp error

//Asprotect 2.3 build01.14

loop8:

mov tmp7, AsprAPIloc

scmp caller, "lab84"

je loop8\_2

mov tmp1, [tmp6]

GMEMI tmp1, MEMORYOWNER

mov tmp2, $RESULT

cmp tmp2, dllimgbase

jne lab48

mov tmp8, 0 //reset counter

loop8\_1:

cmp tmp8, tmp5 //compare all the API in AsprAPIloc?

ja error

mov tmp2, [tmp7] //AsprAPIloc

cmp tmp1, tmp2

je loop8\_3

add tmp7, 4

add tmp8, 1

jmp loop8\_1

loop8\_2:

mov tmp1, [tmp6]

cmp tmp1, 0

je lab48

mov tmp8, [tmp6+4]

//0-GetRegistrationKeys,1-GetRegistrationInformation,2-CheckKey,3-CheckKeyAndDecrypt

//4-GetKeyDate,5-GetKeyExpirationDate,6-GetTrialDays,7-GetTrialExecs

//8-GetExpirationDate,9-GetModeInformation,A-GetHardwareID,B-SetUserKey

loop8\_3:

cmp tmp8, 1

je B\_GRI

cmp tmp8, 2

je B\_CK

cmp tmp8, 3

je B\_CKAD

cmp tmp8, 4

je B\_GKD

cmp tmp8, 5

je B\_GKED

cmp tmp8, 6

je B\_GTD

cmp tmp8, 7

je B\_GTE

cmp tmp8, 8

je B\_GED

cmp tmp8, 9

je B\_GMI

cmp tmp8, 0A

je B\_GHI

msg "这个 API 没有模拟"

//pause

scmp caller, "lab84"

je loop8\_4

add tmp6, 4

jmp loop8

loop8\_4:

add tmp6, 8

jmp loop8

//GetRegistrationInformation

B\_GRI:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C700909090008B44240CC70090909000B801000000C20C00#

add tmp3, 6

mov tmp4, EmuAddr

add tmp4, 20

mov [tmp4], #313131313232323233333333# //111122223333

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne B\_GRI\_1

mov tmp9, EmuAddr

add tmp9, 6

call DLLASPRAPI

B\_GRI\_1:

add tmp3, 0A

mov tmp4, EmuAddr

add tmp4, 30

cmp isdll, 1

jne B\_GRI\_2

mov tmp9, EmuAddr

add tmp9, 10

call DLLASPRAPI

B\_GRI\_2:

mov [tmp4], #04000000566F6C58#

add tmp4, 4

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetRegistrationInformation "

scmp caller, "lab84"

je B\_GRI\_3

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 40

add tmp6, 4

jmp loop8

B\_GRI\_3:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 40

add tmp6, 8

jmp loop8

//CheckKey

B\_CK:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20C00#

log EmuAddr, "CheckKey "

scmp caller, "lab84"

je B\_CK\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop8

B\_CK\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop8

//CheckKeyAndDecrypt

B\_CKAD:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20C00#

log EmuAddr, "CheckKeyAndDecrypt "

scmp caller, "lab84"

je B\_CKAD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop8

B\_CKAD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop8

//GetKeyDate

B\_GKD:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C70001008B44240C66C70001008B44241066C700D707B801000000C21000#

log EmuAddr, "GetKeyDate "

scmp caller, "lab84"

je B\_GKD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop8

B\_GKD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop8

//GetKeyExpirationDate

B\_GKED:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C7001E008B44240C66C7000C008B44241066C7006B08B801000000C21000#

log EmuAddr, "GetKeyExpirationDate "

scmp caller, "lab84"

je B\_GKED\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop8

B\_GKED\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop8

//GetTrialDays

B\_GTD:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C7001E0000008B44240CC7001E000000B801000000C20C00#

log EmuAddr, "GetTrialDays "

scmp caller, "lab84"

je B\_GTD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop8

B\_GTD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop8

//GetTrialExecs

B\_GTE:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C7001E0000008B44240CC7001E000000B801000000C20C00#

log EmuAddr, "GetTrialExecs "

scmp caller, "lab84"

je B\_GTE\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop8

B\_GTE\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop8

//GetExpirationDate

B\_GED:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C7001E008B44240C66C7000C008B44241066C7006B08B801000000C21000#

log EmuAddr, "GetExpirationDate "

scmp caller, "lab84"

je B\_GED\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop8

B\_GED\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop8

//GetModeInformation

B\_GMI:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C700909090008B44240CC70090909000B801000000C20C00#

add tmp3, 6

mov tmp4, EmuAddr

add tmp4, 20

mov [tmp4], #53697465204C6963656E7365# //Site license

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne B\_GMI\_1

mov tmp9, EmuAddr

add tmp9, 6

call DLLASPRAPI

B\_GMI\_1:

add tmp3, 0A

mov tmp4, EmuAddr

add tmp4, 30

mov [tmp4], #030000000#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne B\_GMI\_2

mov tmp9, EmuAddr

add tmp9, 10

call DLLASPRAPI

B\_GMI\_2:

log EmuAddr, "GetModeInformation "

scmp caller, "lab84"

je B\_GMI\_3

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 40

add tmp6, 4

jmp loop8

B\_GMI\_3:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 40

add tmp6, 8

jmp loop8

//GetHardwareID

B\_GHI:

mov tmp3, EmuAddr

mov [tmp3], #B890909000C3#

add tmp3, 1

mov tmp4, EmuAddr

add tmp4, 10

mov [tmp4], #31323334353637382D34343434#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetHardwareID "

cmp isdll, 1

jne B\_GHI\_1

mov tmp9, EmuAddr

add tmp9, 1

call DLLASPRAPI

B\_GHI\_1:

scmp caller, "lab84"

je B\_GHI\_2

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop8

B\_GHI\_2:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop8

//Asprotect v2.11

loop9:

mov tmp7, AsprAPIloc

scmp caller, "lab84"

je loop9\_2

mov tmp1, [tmp6]

GMEMI tmp1, MEMORYOWNER

mov tmp2, $RESULT

cmp tmp2, dllimgbase

jne lab48

mov tmp8, 0 //reset counter

loop9\_1:

cmp tmp8, tmp5 //compare all the API in AsprAPIloc?

ja error

mov tmp2, [tmp7] //AsprAPIloc

cmp tmp1, tmp2

je loop9\_3

add tmp7, 4

add tmp8, 1

jmp loop9\_1

loop9\_2:

//log tmp6

mov tmp1, [tmp6]

cmp tmp1, 0

je lab48

mov tmp8, [tmp6+4]

//0-GetRegistrationKeys,1-GetRegistrationInformation,2-SaveKey,3-CheckKey

//4-CheckKeyAndDecrypt,5-GetKeyDate,6-GetKeyExpirationDate,7-GetTrialDays

//8-GetTrialExecs,9-GetExpirationDate,A-GetModeInformation,B-GetHardwareID

//C-SetUserKey

loop9\_3:

cmp tmp8, 1

je C\_GRI

cmp tmp8, 3

je C\_CK

cmp tmp8, 4

je C\_CKAD

cmp tmp8, 5

je C\_GKD

cmp tmp8, 6

je C\_GKED

cmp tmp8, 7

je C\_GTD

cmp tmp8, 8

je C\_GTE

cmp tmp8, 9

je C\_GED

cmp tmp8, 0A

je C\_GMI

cmp tmp8, 0B

je C\_GHI

msg "这个 API 没有模拟"

//pause

scmp caller, "lab84"

je loop9\_4

add tmp6, 4

jmp loop9

loop9\_4:

add tmp6, 8

jmp loop9

//GetRegistrationInformation

C\_GRI:

mov tmp3, EmuAddr

mov [tmp3], #8B442404C700909090008B442408C70090909000B801000000C20800#

add tmp3, 6

mov tmp4, EmuAddr

add tmp4, 20

mov [tmp4], #313131313232323233333333# //111122223333

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne C\_GRI\_1

mov tmp9, EmuAddr

add tmp9, 6

call DLLASPRAPI

C\_GRI\_1:

add tmp3, 0A

mov tmp4, EmuAddr

add tmp4, 30

cmp isdll, 1

jne C\_GRI\_2

mov tmp9, EmuAddr

add tmp9, 10

call DLLASPRAPI

C\_GRI\_2:

mov [tmp4], #04000000566F6C58#

add tmp4, 4

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetRegistrationInformation "

scmp caller, "lab84"

je C\_GRI\_3

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 40

add tmp6, 4

jmp loop9

C\_GRI\_3:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 40

add tmp6, 8

jmp loop9

//CheckKey

C\_CK:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20800#

log EmuAddr, "CheckKey "

scmp caller, "lab84"

je C\_CK\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop9

C\_CK\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop9

//CheckKeyAndDecrypt

C\_CKAD:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20C00#

log EmuAddr, "CheckKeyAndDecrypt "

scmp caller, "lab84"

je C\_CKAD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop9

C\_CKAD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop9

//GetKeyDate

C\_GKD:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C70001008B44240C66C70001008B44241066C700D707B801000000C20C00#

log EmuAddr, "GetKeyDate "

scmp caller, "lab84"

je C\_GKD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop9

C\_GKD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop9

//GetKeyExpirationDate

C\_GKED:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C7001E008B44240C66C7000C008B44241066C7006B08B801000000C20C00#

log EmuAddr, "GetKeyExpirationDate "

scmp caller, "lab84"

je C\_GKED\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop9

C\_GKED\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop9

//GetTrialDays

C\_GTD:

mov tmp3, EmuAddr

mov [tmp3], #8B442404C7001E0000008B442408C7001E000000B801000000C20800#

log EmuAddr, "GetTrialDays "

scmp caller, "lab84"

je C\_GTD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop9

C\_GTD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop9

//GetTrialExecs

C\_GTE:

mov tmp3, EmuAddr

mov [tmp3], #8B442404C7001E0000008B442408C7001E000000B801000000C20800#

log EmuAddr, "GetTrialExecs "

scmp caller, "lab84"

je C\_GTE\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop9

C\_GTE\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop9

//GetExpirationDate

C\_GED:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C7001E008B44240C66C7000C008B44241066C7006B08B801000000C20C00#

log EmuAddr, "GetExpirationDate "

scmp caller, "lab84"

je C\_GED\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop9

C\_GED\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop9

//GetModeInformation

C\_GMI:

mov tmp3, EmuAddr

mov [tmp3], #8B442404C700909090008B442408C70090909000B801000000C20C00#

add tmp3, 6

mov tmp4, EmuAddr

add tmp4, 20

mov [tmp4], #53697465204C6963656E7365# //Site license

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne C\_GMI\_1

mov tmp9, EmuAddr

add tmp9, 6

call DLLASPRAPI

C\_GMI\_1:

add tmp3, 0A

mov tmp4, EmuAddr

add tmp4, 30

mov [tmp4], #030000000#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne C\_GMI\_2

mov tmp9, EmuAddr

add tmp9, 10

call DLLASPRAPI

C\_GMI\_2:

log EmuAddr, "GetModeInformation "

scmp caller, "lab84"

je C\_GMI\_3

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 40

add tmp6, 4

jmp loop9

C\_GMI\_3:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 40

add tmp6, 8

jmp loop9

//GetHardwareID

C\_GHI:

mov tmp3, EmuAddr

mov [tmp3], #B890909000C3#

add tmp3, 1

mov tmp4, EmuAddr

add tmp4, 10

mov [tmp4], #31323334353637382D34343434#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetHardwareID "

cmp isdll, 1

jne C\_GHI\_1

mov tmp9, EmuAddr

add tmp9, 1

call DLLASPRAPI

C\_GHI\_1:

scmp caller, "lab84"

je C\_GHI\_2

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop9

C\_GHI\_2:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop9

//Asprotect 2.3 build04.26

loop10:

mov tmp7, AsprAPIloc

scmp caller, "lab84"

je loop10\_2

mov tmp1, [tmp6]

GMEMI tmp1, MEMORYOWNER

mov tmp2, $RESULT

cmp tmp2, dllimgbase

jne lab48

mov tmp8, 0 //reset counter

loop10\_1:

cmp tmp8, tmp5 //compare all the API in AsprAPIloc?

ja error

mov tmp2, [tmp7] //AsprAPIloc

cmp tmp1, tmp2

je loop10\_3

add tmp7, 4

add tmp8, 1

jmp loop10\_1

loop10\_2:

//log tmp6

mov tmp1, [tmp6]

cmp tmp1, 0

je lab48

mov tmp8, [tmp6+4]

//0-GetRegistrationKeys,1-GetRegistrationInformation,2-RemoveKey,3-CheckKey

//4-CheckKeyAndDecrypt,5-GetKeyDate,6-GetKeyExpirationDate,7-GetTrialDays

//8-GetTrialExecs,9-GetExpirationDate,A-GetModeInformation,B-GetHardwareID

//C-GetHardwareIDEx,D-SetUserKey

loop10\_3:

cmp tmp8, 1

je D\_GRI

cmp tmp8, 2

je D\_RK

cmp tmp8, 3

je D\_CK

cmp tmp8, 4

je D\_CKAD

cmp tmp8, 5

je D\_GKD

cmp tmp8, 6

je D\_GKED

cmp tmp8, 7

je D\_GTD

cmp tmp8, 8

je D\_GTE

cmp tmp8, 9

je D\_GED

cmp tmp8, 0A

je D\_GMI

cmp tmp8, 0B

je D\_GHI

cmp tmp8, 0C

je D\_GHIE

msg "这个 API 没有模拟"

//pause

scmp caller, "lab84"

je loop10\_4

add tmp6, 4

jmp loop10

loop10\_4:

add tmp6, 8

jmp loop10

//GetRegistrationInformation

D\_GRI:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C700909090008B44240CC70090909000B801000000C20C00#

add tmp3, 6

mov tmp4, EmuAddr

add tmp4, 20

mov [tmp4], #313131313232323233333333# //111122223333

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne D\_GRI\_1

mov tmp9, EmuAddr

add tmp9, 6

call DLLASPRAPI

D\_GRI\_1:

add tmp3, 0A

mov tmp4, EmuAddr

add tmp4, 30

cmp isdll, 1

jne D\_GRI\_2

mov tmp9, EmuAddr

add tmp9, 10

call DLLASPRAPI

D\_GRI\_2:

mov [tmp4], #04000000566F6C58#

add tmp4, 4

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetRegistrationInformation "

scmp caller, "lab84"

je D\_GRI\_3

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 40

add tmp6, 4

jmp loop10

D\_GRI\_3:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 40

add tmp6, 8

jmp loop10

//RemoveKey

D\_RK:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20C00#

log EmuAddr, "RemoveKey "

scmp caller, "lab84"

je D\_RK\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop10

D\_RK\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop10

//CheckKey

D\_CK:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20C00#

log EmuAddr, "CheckKey "

scmp caller, "lab84"

je D\_CK\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop10

D\_CK\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop10

//CheckKeyAndDecrypt

D\_CKAD:

mov tmp3, EmuAddr

mov [tmp3], #B801000000C20C00#

log EmuAddr, "CheckKeyAndDecrypt "

scmp caller, "lab84"

je D\_CKAD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 10

add tmp6, 4

jmp loop10

D\_CKAD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 10

add tmp6, 8

jmp loop10

//GetKeyDate

D\_GKD:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C70001008B44240C66C70001008B44241066C700D707B801000000C21000#

log EmuAddr, "GetKeyDate "

scmp caller, "lab84"

je D\_GKD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop10

D\_GKD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop10

//GetKeyExpirationDate

D\_GKED:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C7001E008B44240C66C7000C008B44241066C7006B08B801000000C21000#

log EmuAddr, "GetKeyExpirationDate "

scmp caller, "lab84"

je D\_GKED\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop10

D\_GKED\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop10

//GetTrialDays

D\_GTD:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C7001E0000008B44240CC7001E000000B801000000C20C00#

log EmuAddr, "GetTrialDays "

scmp caller, "lab84"

je D\_GTD\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop10

D\_GTD\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop10

//GetTrialExecs

D\_GTE:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C7001E0000008B44240CC7001E000000B801000000C20C00#

log EmuAddr, "GetTrialExecs "

scmp caller, "lab84"

je D\_GTE\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop10

D\_GTE\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop10

//GetExpirationDate

D\_GED:

mov tmp3, EmuAddr

mov [tmp3], #8B44240866C7001E008B44240C66C7000C008B44241066C7006B08B801000000C21000#

log EmuAddr, "GetExpirationDate "

scmp caller, "lab84"

je D\_GED\_1

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 30

add tmp6, 4

jmp loop10

D\_GED\_1:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 30

add tmp6, 8

jmp loop10

//GetModeInformation

D\_GMI:

mov tmp3, EmuAddr

mov [tmp3], #8B442408C700909090008B44240CC70090909000B801000000C20C00#

add tmp3, 6

mov tmp4, EmuAddr

add tmp4, 20

mov [tmp4], #53697465204C6963656E7365# //Site license

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne D\_GMI\_1

mov tmp9, EmuAddr

add tmp9, 6

call DLLASPRAPI

D\_GMI\_1:

add tmp3, 0A

mov tmp4, EmuAddr

add tmp4, 30

mov [tmp4], #030000000#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

cmp isdll, 1

jne D\_GMI\_2

mov tmp9, EmuAddr

add tmp9, 10

call DLLASPRAPI

D\_GMI\_2:

log EmuAddr, "GetModeInformation "

scmp caller, "lab84"

je D\_GMI\_3

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 40

add tmp6, 4

jmp loop10

D\_GMI\_3:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 40

add tmp6, 8

jmp loop10

//GetHardwareID

D\_GHI:

mov tmp3, EmuAddr

mov [tmp3], #B890909000C20400#

add tmp3, 1

mov tmp4, EmuAddr

add tmp4, 10

mov [tmp4], #31323334353637382D34343434#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetHardwareID "

cmp isdll, 1

jne D\_GHI\_1

mov tmp9, EmuAddr

add tmp9, 1

call DLLASPRAPI

D\_GHI\_1:

scmp caller, "lab84"

je D\_GHI\_2

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop10

D\_GHI\_2:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop10

//GetHardwareIDEx

D\_GHIE:

mov tmp3, EmuAddr

mov [tmp3], #B890909000C3#

add tmp3, 1

mov tmp4, EmuAddr

add tmp4, 10

mov [tmp4], #31323334353637382D34343434#

sub tmp4, imgbase

add tmp4, imgbasefromdisk

mov [tmp3], tmp4

log EmuAddr, "GetHardwareIDEx "

cmp isdll, 1

jne D\_GHIE\_1

mov tmp9, EmuAddr

add tmp9, 1

call DLLASPRAPI

D\_GHIE\_1:

scmp caller, "lab84"

je D\_GHIE\_2

mov tmp3, EmuAddr

sub tmp3, imgbase

add tmp3, imgbasefromdisk

mov [tmp6], tmp3

add EmuAddr, 20

add tmp6, 4

jmp loop10

D\_GHIE\_2:

eval "jmp 0{EmuAddr}"

asm tmp1, $RESULT

add EmuAddr, 20

add tmp6, 8

jmp loop10

DLLASPRAPI:

cmp tmp10, 0

je reloc1

cmp tmp10, 1

je reloc2

cmp tmp10, 2

je reloc3

cmp tmp10, 3

je reloc4

cmp tmp10, 4

je reloc5

cmp tmp10, 5

je reloc6

msg "DLLASPRAPI error"

//pause

jmp error

reloc1:

sub tmp9, imgbase

mov reloc1, tmp9

jmp DLLASPRAPI\_1

reloc2:

sub tmp9, imgbase

mov reloc2, tmp9

jmp DLLASPRAPI\_1

reloc3:

sub tmp9, imgbase

mov reloc3, tmp9

jmp DLLASPRAPI\_1

reloc4:

sub tmp9, imgbase

mov reloc4, tmp9

jmp DLLASPRAPI\_1

reloc5:

sub tmp9, imgbase

mov reloc5, tmp9

jmp DLLASPRAPI\_1

reloc6:

sub tmp9, imgbase

mov reloc6, tmp9

DLLASPRAPI\_1:

add tmp10, 1

ret

lab48:

cmp isdll, 1

jne lab51

mov tmp1, reloc\_rva

add tmp1, imgbase

mov tmp2, tmp1

add tmp2, 08

mov tmp3, [tmp2], 2

and tmp3, 0F000

cmp tmp3, 3000 //type 3 relocation ?

jne lab51

GMEMI tmp1, MEMORYSIZE

mov tmp2, $RESULT

alloc tmp2

mov reloctemp, $RESULT

//log reloctemp

cmp tmp10, 0 //no relocation of item in emulation code

je lab49\_1

//add relocate item for dll

mov tmp1, dllimgbase

mov [tmp1], #609CBD00038D00C745040000E200C7450800D00010C7450C5C040000C7451001000000B917010000B8003000008B7D08#

add tmp1, 30 //30

mov [tmp1], #8BD7F2AF83F9000F85730000008BFA8B0F83F9000F84160200003BC877078B4F0403F9EBEA8BCF8BD12B4D088B5D0C2B#

add tmp1, 30 //60

mov [tmp1], #D98BCB53578B7D048BF2F3A433C05F8BCBF3AA8BFAC7070090000083C20483C708E87A010000E89502000085C0740383#

add tmp1, 30 //90

mov [tmp1], #C70283C108890A598B7504F3A4E94701000090909090909090909090909090908BD783EA04031766837AFE007507C745#

add tmp1, 30 //C0

mov [tmp1], #0001000000578B0F83E90833C083C7048BD7668B07663DFD32771183C70283E90283F9000F84A6010000EBE690909090#

add tmp1, 30 //F0

mov [tmp1], #8BD78BCF2B4D088B5D0C2BD98BCB53578B7D048BF2F3A433C05F8BCBF3AA8BFAE8EB000000598B7504F3A45AE8FF0100#

add tmp1, 30 //120

mov [tmp1], #00890A8BFA9C33C98B4510A8010F94C19D83F9010F84AF000000837D0000747090909090909090909090909090909090#

add tmp1, 30 //150

mov [tmp1], #8B0F83E90403F98BD783C7028BCF2B4D088B5D0C2BD98B4D10D1E103D98BCB8BF78B7D04F3A433C08BCB8BFAF3AA8BFA#

add tmp1, 30 //180

mov [tmp1], #8B75048BCBF3A4EB60909090909090909090909090909090909090909090909090909090909090909090909090909090#

add tmp1, 30 //1B0

mov [tmp1], #8B0F83E90403F98BD783EF028BD78BCF2B4D088B5D0C2BD98B4D10D1E103D98BCB8B7D048BF2F3A48BFA66C70700008B#

add tmp1, 30 //1E0

mov [tmp1], #CB8B750483C702F3A49D619090909090000000000000000000000000000000008B4D1066C707063649E33E83C70266C7#

add tmp1, 30 //210

mov [tmp1], #07103649E33383C70266C707803A49E32883C70266C707803A49E31D83C70266C707803A49E31283C70266C707803A49#

add tmp1, 30 //240

mov [tmp1], #83F9000F850500000083C702C390909000000000000000000000000000000000C70700B000008BD783C20483C708E88D#

add tmp1, 30 //270

mov [tmp1], #FFFFFFE8A800000083C108890AE967FFFFFF00000000000000000000000000008BCF2B4D088B5D0C2BD98BCB578BF78B#

add tmp1, 30 //2A0

mov [tmp1], #7D04F3A45A837D0001750383EA028BFAE84BFFFFFF5AE865000000890A85C0740866C707000083C7028BCB8B7504F3A4#

add tmp1, 30 //2D0

mov [tmp1], #E914FFFFFF9000000000000000000000#

add tmp1, 50 //320

mov [tmp1], #8B4D10D1E18BF28B0683F800740B837D0000740383E80203C88BC1C1E902C1E1023BC8740A83C0028BC833C040EB0233#

add tmp1, 30 //350

mov [tmp1], #C0C30000000000000000000000000000#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov tmp2, dllimgbase

add tmp2, 400

mov [tmp1], tmp2

add tmp1, 7 //A

mov [tmp1], reloctemp

add tmp1, 7 //11

mov tmp2, reloc\_rva

add tmp2, imgbase

mov [tmp1], tmp2

add tmp1, 7 //18

mov [tmp1], reloc\_size

add tmp1, 7 //1F

mov [tmp1], tmp10

add tmp1, 5 //24

mov tmp3, reloc\_size

shr tmp3, 2

mov [tmp1], tmp3 //reloc no.

add tmp1, 5 //29

mov tmp5, reloc1

and tmp5, 0FFFFF000

mov [tmp1], tmp5

add tmp1, 4E //77

mov [tmp1], tmp5

add tmp1, 60 //D7

mov tmp3, [tmp1+2]

mov tmp2, reloc1

sub tmp2, tmp5

add tmp2, 3000

mov [tmp1], tmp2

add tmp1, 2 //D9

mov [tmp1], tmp3

add tmp1, 12D //206

mov tmp6, reloc1

sub tmp6, tmp5

add tmp6, 3000

mov tmp3, [tmp1+2]

mov [tmp1], tmp6

add tmp1, 2

mov [tmp1], tmp3

cmp tmp10, 1

je lab48\_1

mov tmp1, dllimgbase

add tmp1, 211 //211

mov tmp6, reloc2

sub tmp6, tmp5

add tmp6, 3000

mov tmp3, [tmp1+2]

mov [tmp1], tmp6

add tmp1, 2

mov [tmp1], tmp3

cmp tmp10, 2

je lab48\_1

mov tmp1, dllimgbase

add tmp1, 21C //21C

mov tmp6, reloc3

sub tmp6, tmp5

add tmp6, 3000

mov tmp3, [tmp1+2]

mov [tmp1], tmp6

add tmp1, 2

mov [tmp1], tmp3

cmp tmp10, 3

je lab48\_1

mov tmp1, dllimgbase

add tmp1, 227 //227

mov tmp6, reloc4

sub tmp6, tmp5

add tmp6, 3000

mov tmp3, [tmp1+2]

mov [tmp1], tmp6

add tmp1, 2

mov [tmp1], tmp3

cmp tmp10, 4

je lab48\_1

mov tmp1, dllimgbase

add tmp1, 232 //232

mov tmp6, reloc5

sub tmp6, tmp5

add tmp6, 3000

mov tmp3, [tmp1+2]

mov [tmp1], tmp6

add tmp1, 2

mov [tmp1], tmp3

cmp tmp10, 5

je lab48\_1

mov tmp1, dllimgbase

add tmp1, 123D //23D

mov tmp6, reloc6

sub tmp6, tmp5

add tmp6, 3000

mov tmp3, [tmp1+2]

mov [tmp1], tmp6

add tmp1, 2

mov [tmp1], tmp3

cmp tmp10, 6

jne error

lab48\_1:

mov tmp1, dllimgbase

add tmp1, 262 //262

mov [tmp1], tmp5

mov tmp1, dllimgbase

add tmp1, 1EB //1EB--end point

mov tmp2, tmp1

add tmp2, 63 //24E--error point

mov tmp7, eip

mov eip, dllimgbase

bp tmp1

bp tmp2

eob lab48\_2

eoe lab48\_2

esto

lab48\_2:

cmp eip, tmp1

je lab48\_3

cmp eip, tmp2

je lab48\_4

jmp error

lab48\_3:

bc tmp1

bc tmp2

mov eip, tmp7

fill dllimgbase, 320, 00

mov tmp1, reloc\_rva

add tmp1, imgbase

call ChkRelocSize

jmp lab49

lab48\_4:

msg "修复重定位表出错"

//pause

jmp error

lab49:

mov reloc\_size, tmp2

//log reloc\_size

//relocate addr in IAT

lab49\_1:

coe

cob

find Aspr1stthunk, #00000000#

mov tmp10, $RESULT

sub tmp10, Aspr1stthunk

shr tmp10, 2

mov tmp2, tmp10

shl tmp2, 2

cmp tmp1, tmp2

je lab49\_2

add tmp10, 1

lab49\_2:

mov tmp1, dllimgbase

mov [tmp1], #609CBD00038D00C745040000E200C7450818900010C7450C00900010C7451000D00010C7451460040000B917010000B8#

add tmp1, 30 //30

mov [tmp1], #009000008B7D108BD7F2AF85C90F85FD0000008BFA8B0F83F9000F84900000003BC877078B4F0403F9EBEA8BCF8BD12B#

add tmp1, 30 //60

mov [tmp1], #4D108B5D142BD98BCB53578B7D048BF2F3A433C05F8BCBF3AA8BFAC7070090000083C7088BD7B9030000008B5D088BF3#

add tmp1, 30 //90

mov [tmp1], #2B750C81C6003000006689374983F900740883C70283C304EBE483C7028BCF2BCA83C1088BD9C1E902C1E1023BCB7406#

add tmp1, 30 //C0

mov [tmp1], #83C70283C302895AFC5B8BCB8B7504F3A4E99D01000000000000000000009090C70700B0000083C7088BD7B903000000#

add tmp1, 30 //F0

mov [tmp1], #8B5D088BF32B750C81C6003000006689374983F900740883C70283C304EBE483C7028BCF2BCA83C1088BD9C1E902C1E1#

add tmp1, 30 //120

mov [tmp1], #023BCB740683C70283C302895AFCE940010000000000000000000000000000908BD783EA04031766837AFE00750A832F#

add tmp1, 30 //150

mov [tmp1], #02C7450001000000578B0F83E90833C083C7048BD7668B07663D1830770883C70283E902EBEF83F900740D8B42FC83E8#

add tmp1, 30 //180

mov [tmp1], #083BC1740383EF028BD78BCF2B4D108B5D142BD98BCB53578B7D048BF2F3A433C05F8BCBF3AA8BFAB9030000008B5D08#

add tmp1, 30 //1B0

mov [tmp1], #8BF32B750C81C6003000006689374983F900740883C70283C304EBE483C7025B8BCB8B7504F3A45FB903000000D1E101#

add tmp1, 30 //1E0

mov [tmp1], #0F8BC18BD783EA0403178BCA2BCF83E9048BD9C1E902C1E1023BCB7443830702578BFA8BCF2B4D108B5D142BD903D88B#

add tmp1, 30 //210

mov [tmp1], #CB578B7D048BF2F3A433C05F66C707000083C7028BCB8B7504F3A45FEB45000000000000000000000000000000009090#

add tmp1, 30 //240

mov [tmp1], #837D0001752D8BFA8BCF2B4D108B5D142BD903D88BCB578B7D0483C2028BF2F3A433C05F578BCB8BFAF3AA5F8BCB8B75#

add tmp1, 30 //270

mov [tmp1], #04F3A49D619090909090909000000000#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov tmp2, dllimgbase

add tmp2, 300

mov [tmp1], tmp2

add tmp1, 7 //0A

mov [tmp1], reloctemp

add tmp1, 7 //11

mov [tmp1], Aspr1stthunk

add tmp1, 7 //18

GMEMI Aspr1stthunk, MEMORYBASE

mov tmp3, $RESULT

mov [tmp1], tmp3

add tmp1, 7 //1F

mov tmp3, reloc\_rva

add tmp3, imgbase

mov [tmp1], tmp3

add tmp1, 7 //26

mov [tmp1], reloc\_size

add tmp1, 5 //2B

mov tmp3, reloc\_size

shr tmp3, 2

mov [tmp1], tmp3

add tmp1, 5 //30

GMEMI Aspr1stthunk, MEMORYBASE

mov tmp6, $RESULT

sub tmp6, imgbase

mov [tmp1], tmp6

add tmp1, 4D //7D

mov [tmp1], tmp6

add tmp1, A //87

mov [tmp1], tmp10

add tmp1, 5B //E2

mov [tmp1], tmp6

add tmp1, A //EC

mov [tmp1], tmp10

add tmp1, 7E //16A

mov tmp4, Aspr1stthunk

sub tmp4, tmp6

add tmp4, 3000

mov tmp2, [tmp1+2]

mov [tmp1], tmp4

add tmp1, 2 //16C

mov [tmp1], tmp2

add tmp1, 3D //1A9

mov [tmp1], tmp10

add tmp1, 30 //1D9

mov [tmp1], tmp10

add tmp1, 9C //275 -- end point

mov tmp7, eip

mov eip, dllimgbase

bp tmp1

eob lab49\_3

eoe lab49\_3

run

lab49\_3:

cmp eip, tmp1

je lab49\_4

jmp error

lab49\_4:

bc tmp1

mov eip, tmp7

fill dllimgbase, 320, 00

mov tmp1, reloc\_rva

add tmp1, imgbase

call ChkRelocSize

lab49\_5:

mov reloc\_size, tmp2

//log reloc\_size

GMEMI reloctemp, MEMORYSIZE

mov tmp2, $RESULT

free reloctemp, tmp2

lab51:

scmp caller, "lab46\_1"

je lab52

scmp caller, "lab84"

je lab85

jmp error

//Search and fix CRC check

lab52:

mov caller, "nil"

cob

coe

mov tmp9, eip //save eip

mov tmp1, dllimgbase

mov [tmp1], #609CBE00104000B9FCAF28008B1681E2F0F0FF0081FA5050E8000F85100100008A1680E20F80FA0873688A560180E20F#

add tmp1, 30 //30

mov [tmp1], #80FA08735D8B5E0481E3FFFFFF0083FB00754F515683C607B90001000033C08B1681E2FFF0F0F081FAC35050E0740846#

add tmp1, 30 //60

mov [tmp1], #4985C975EAEB03408BD65E5983F80175218D5E038B1B03DE83C3073BDA73138A42013C58720C8A42023C587205E90E00#

add tmp1, 30 //90

mov [tmp1], #0000E9A90100009090909090909090904250515756B8E9000000B9000100008BFE33F6F2AEE3193BFA77158BDF031F83#

add tmp1, 30 //C0

mov [tmp1], #C3043BDA75ED46EBEA9090909090909083FE01742B83FE0274095E5F5958E95D0100005E8BC683C002C600B8C7400101#

add tmp1, 30 //F0

mov [tmp1], #00000083C005EB0E00000000000000005E8BC683C002C600E98BCA2BC883E9058948015F5958E9250100009000000000#

add tmp1, 30 //120

mov [tmp1], #000000000000000000000000000000008B1681E2F0F0FFFF81FA50500F84754066817E06FFFF75388B5EF381E3FFFF00#

add tmp1, 30 //150

mov [tmp1], #FF81FB0F8200FF75278B56F981E2F0FFF00081FA5081F000751666C7460290E9E9CB0000000000000000000000000090#

add tmp1, 30 //180

mov [tmp1], #803EE90F85B70000008B560183FA000F85AB00000033DB668B5E056681E3F0F06681FB50500F859500000033D28A5605#

add tmp1, 30 //1B0

mov [tmp1], #80E20F80FA080F82840000008A560680E20F80FA087279807E07E975738B560881E200FFFFFF83FA007565575150B80F#

add tmp1, 30 //1E0

mov [tmp1], #000000B9400000008BFE83EF40F2AE85C97448803F847407803F857417EBEE8BC70347013BC6753366C747FF90E9EB2B#

add tmp1, 30 //210

mov [tmp1], #000000008BC70347018038E9751D8A580180E3F080FB1077129090909066837803007507C747010000000058595F9090#

add tmp1, 30 //240

mov [tmp1], #83C60183E90185C90F85BEFDFFFF9D619090#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov [tmp1], 1stsecbase

add tmp1, 5 //08

mov tmp3, sizeofimg

sub tmp3, 2004

mov [tmp1], tmp3

mov tmp3, dllimgbase

add tmp3, 250 //end point

mov eip, dllimgbase

bp tmp3

run

cmp eip, tmp3

jne error

bc tmp3

lab53:

fill dllimgbase, 260, 00

mov eip, tmp9

//get all call xxxxxxxx

lab54:

cmp type1API, 0

je lab78

fixtype1:

find dllimgbase, #3130320D0A# //search "102"

mov tmp6, $RESULT

cmp tmp6, 0

je error

find tmp6, #05FF00000050# //"Add eax,FF" "push eax"

mov tmp1, $RESULT

cmp tmp1, 0

je error

find tmp1, #8B45F4E8#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 3

opcode tmp2

mov func1, $RESULT\_1

//log func1

add tmp2, 5

find tmp2, #8B45F4E8#

mov tmp1, $RESULT

cmp tmp1, 0

je error

add tmp1, 3

opcode tmp1

mov func2, $RESULT\_1

//log func2

add tmp1, 5

find tmp1, #8B45F4E8????????#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 3

opcode tmp2

mov func3, $RESULT\_1

//log func3

mov tmp1, tmp2

add tmp1, 5

mov tmp3, [tmp1]

find tmp1, #8B55FCE8#

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 3

opcode tmp2

mov func4, $RESULT\_1

//log func4

cmp tmp3, A1FC4589

jne lab55

find tmp1, #8B83080100008B401C#

mov tmp2, $RESULT

cmp tmp2, 0

je lab54\_1

mov v2.0x, 1

jmp lab55

lab54\_1:

mov v1.32, 1

lab55:

//log v1.32

//log v2.0x

mov tmp1, dllimgbase

mov [tmp1], #609CBB000E0201BE00104000803EE875188B460103C683C0053B432C750B893500C09E00E8170000004681FE00705900#

add tmp1, 30 //30

mov [tmp1], #72DA9D6190909000000000000000009060BD0009FB00A100C09E00894510BB000E02018B480103C883C1053B4B2C7421#

add tmp1, 30 //60

mov [tmp1], #61C3909090909090909090909090909090909090909090909090909090909090908B45102B43148B55102B53242B93E0#

add tmp1, 30 //90

mov [tmp1], #0000008955F83B43280F83600400008D53408955E48B53188955F48B551083C2058A123293E00000008BFA81E7FF0000#

add tmp1, 30 //C0

mov [tmp1], #0025FF00000033F83B7DF40F87AE0100008B83E4000000F7EF0343548945FC8B45E40FB6008D04408B7483688B45FCFF#

add tmp1, 30 //F0

mov [tmp1], #D68BF03B75F80F8574010000807B2000741B8B45E40FB640098D04408B5483688B45FCFFD23C010F843B0200008D75FC#

add tmp1, 30 //120

mov [tmp1], #33C08A43428D04408BD38B7C82688B06FFD78945B833C08A43438D04408BD38B7C82688B06FFD78BF833C08A43458D04#

add tmp1, 30 //150

mov [tmp1], #408BD38B5482688B06FFD28845B733C08A43418D04408BD38B5482688B06FFD28845BF8B83E00000000345B88945D433#

add tmp1, 30 //180

mov [tmp1], #C08A43478D04408BD38B5482688B06FFD28945E003BBE00000005733C08A45B705FF000000508BC3E88BB102008BC88B#

add tmp1, 30 //1B0

mov [tmp1], #53108BC3E80B9F02008945D033C08A43488D04408BD38B7C82688B06FFD78B55D00155E08B5510422B022B45D08B5510#

add tmp1, 30 //1E0

mov [tmp1], #0FB61203C28BD38B522C2B551083EA0503C28D55CC52668B4DE08BD08BC3E8E9AB02008B83E00000000145CC837DD4FF#

add tmp1, 30 //210

mov [tmp1], #740E8B45108B5D14890383C304895D148B5DCCE978020000909090909090909090909090909090909090909090909090#

add tmp1, 30 //240

mov [tmp1], #BE00705900391E741183C60481FE747A59000F87A7020000EBEB81EE0000400081C600004000C3000000000000000090#

add tmp1, 30 //270

mov [tmp1], #81C7FF0000003B7DF40F8652FEFFFF8B83080100008B401C488945F48B43188B55F4423BC27405E9630200008B45F485#

add tmp1, 30 //2A0

mov [tmp1], #C00F8C58020000408945E0C745EC000000008B83080100008B55ECE8800000008BF88B45E40FB6008D04408B7483688B#

add tmp1, 30 //2D0

mov [tmp1], #4704FFD68BF03B75F8753F807B200074178B45E40FB640098D04408B5483688B4704FFD23C01746883C7048BF7E91EFE#

add tmp1, 30 //300

mov [tmp1], #FFFF909090900000000000000000000000000000000090909090FF45ECFF4DE07590E9D8010000909090909000000000#

add tmp1, 30 //330

mov [tmp1], #0000000000000000000000000000000033C985D27C0B3B501C7D068B40188B0C908BC1C3909090908D75FCEB08909090#

add tmp1, 30 //360

mov [tmp1], #83C7048BF733C08A43478D04408BD38B7C82688B06FFD78945EC33C08A43488D04408BD38B7C82688B06FFD78945E833#

add tmp1, 30 //390

mov [tmp1], #C08A43428D04408BD38B7C82688B06FFD78BF833C08A43468D04408BD38B5482688B06FFD28845DF03BBE00000005733#

add tmp1, 30 //3C0

mov [tmp1], #C08A45DF05FF000000508BC3E867AF02008BC88B53108BC3E8E79C02008945D833C08A43438D04408BD38B7C82688B06#

add tmp1, 30 //3F0

mov [tmp1], #FFD78BF803BBE00000008B45EC03C70345D88945EC8B45E82BC72B45D88945E833C08A43418D04408BD38B5482688B06#

add tmp1, 30 //420

mov [tmp1], #FFD28845BF895D208BD88D45B450668B4DEC668B55E88B4520E8AEA902008B45208B80E00000000345B48945FC8945CC#

add tmp1, 30 //450

mov [tmp1], #576A008D4DE08B45208B403C8B55FCE8106D02008945FC8B45E08B00E81F0000000045BF8B5DCCEB2700000000000000#

add tmp1, 30 //480

mov [tmp1], #00000000000000000000000000000090516689C1C1C0106601C828E059C3000081FB909090907507BB90909090EB2181#

add tmp1, 30 //4B0

mov [tmp1], #FB909090907507BB90909090EB1281FB90909090750ABB909090009090909090E86BFDFFFF66B9FF158B5DE48A430A3A#

add tmp1, 30 //4E0

mov [tmp1], #45BF74056681C100108B5D1066890B83C3028933FF05000E900061C390909090#

mov tmp1, dllimgbase

mov tmp2, tmp1

add tmp1, 3 //3

mov [tmp1], EBXaddr

add tmp1, 5 //8

mov [tmp1], 1stsecbase

add tmp1, 18 //20

mov tmp4, dllimgbase

add tmp4, 0E04 //dllimgbase+0E04

mov [tmp1], tmp4

add tmp1, 0C //2C

mov tmp3, sizeofimg

sub tmp3, 1000

add tmp3, imgbase

mov [tmp1], tmp3

add tmp1, 16 //42

mov tmp2, dllimgbase

add tmp2, 900 //dllimgbase+900

mov [tmp1], tmp2

add tmp1, 5 //47

mov [tmp1], tmp4

add tmp1, 8 //4F

mov [tmp1], EBXaddr

add tmp1, 159 //1A8

eval "{func1}"

asm tmp1, $RESULT

add tmp1, C //1B4

eval "{func2}"

asm tmp1, $RESULT

add tmp1, 4A //1FE

eval "{func3}"

asm tmp1, $RESULT

add tmp1, 43 //241

mov [tmp1], iatstartaddr

add tmp1, D //24E

mov [tmp1], iatendaddr

add tmp1, E //25C

mov [tmp1], imgbase

add tmp1, 6 //262

mov [tmp1], imgbasefromdisk

add tmp1, 16A //3CC

eval "{func1}"

asm tmp1, $RESULT

add tmp1, C //3D8

eval "{func2}"

asm tmp1, $RESULT

add tmp1, 61 //439

eval "{func3}"

asm tmp1, $RESULT

add tmp1, 26 //45F

eval "{func4}"

asm tmp1, $RESULT

add tmp1, 97 //4F6

mov tmp2, dllimgbase

add tmp2, E00 //dllimgbase+E00 for storing E8count

mov [tmp1], tmp2

mov tmp2, dllimgbase

add tmp2, 914 //dllimgbase+900

mov [tmp2], lastsecbase //loc for storing sc after API

mov tmp2, dllimgbase

add tmp2, 34 //34 -- end point

bp tmp2

mov tmp3, dllimgbase

add tmp3, 4FF //4FF -- error point

bp tmp3

cmp v1.32, 1

jne lab56

mov tmp4, dllimgbase

add tmp4, 203 //203

mov [tmp4], #8945CC83C404909090#

add tmp4, 7C //27F

mov [tmp4], #8B830401#

add tmp4, 33 //2B2

mov [tmp4], #8B830401#

add tmp4, 18C //43E

mov [tmp4], #83C404909090909090909090#

find dllimgbase, #3136300D0A#

mov tmp4, $RESULT

cmp tmp4, 0

jne lab56\_1

find dllimgbase, #3B7DF40F83????FFFF8B4354#

mov tmp4, $RESULT

cmp tmp4, 0

je error

mov tmp4, dllimgbase

add tmp4, 270 //270

mov [tmp4], #81C7FF0000003B7DF40F8652FEFFFF8B43548945FC8B7B1885FF0F866F0200008B45E40FB6008D04408B7483688B45FC#

add tmp4, 30 //2A0

mov [tmp4], #FFD68BF03B75F87571807B2000741B8B45E40FB640098D04408B5483688B45FCFFD23C010F848E0000008D75FCE94EFE#

add tmp4, 30 //2D0

mov [tmp4], #FFFF00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000#

add tmp4, 30 //300

mov [tmp4], #00000000000000000000000000000000000000000000909090904F8B83E40000000145FC85FF0F8764FFFFFFE9CE01000090#

jmp lab56\_1

lab56:

cmp v2.0x, 1

jne lab56\_1

mov tmp4, dllimgbase

add tmp4, 203 //203

mov [tmp4], #8945CC83C404909090#

add tmp4, 23b //43E

mov [tmp4], #83C404909090909090909090#

lab56\_1:

cmp DFCequ, 0

je lab56\_2

mov tmp1, dllimgbase

add tmp1, 4A2 //4A2

mov [tmp1], DFCequ

add tmp1, 7 //4A9

mov [tmp1], DFCaddr

jmp lab56\_3

lab56\_2:

mov tmp1, dllimgbase

add tmp1, 4A0

mov [tmp1], #EB0D#

lab56\_3:

cmp REequ, 0

je lab56\_4

mov tmp1, dllimgbase

add tmp1, 4B1 //4B1

mov [tmp1], REequ

add tmp1, 7 //4B8

mov [tmp1], REaddr

jmp lab56\_5

lab56\_4:

mov tmp1, dllimgbase

add tmp1, 4AF

mov [tmp1], #EB0D#

lab56\_5:

cmp GPAequ, 0

je lab56\_6

mov tmp1, dllimgbase

add tmp1, 4C0 //4C0

mov [tmp1], GPAequ

add tmp1, 7 //4C7

mov [tmp1], GPAaddr

jmp lab57

lab56\_6:

mov tmp1, dllimgbase

add tmp1, 4BE

mov [tmp1], #EB0B#

lab57:

mov tmp6, eip

mov eip, dllimgbase

eob lab58

eoe lab58

esto

lab58:

cmp eip, tmp2

je lab59

cmp eip, tmp3

je lab60

esto

lab59:

bc tmp2

bc tmp3

mov eip, tmp6

mov E8count, 0

mov E8count, [dllimgbase+0E00]

//log E8count

//msg "修复 type 1 API 完毕"

//pause

jmp lab69

lab60:

msg "Unexpected termination of the process"

//pause

jmp end

//lab61\_lab68

lab69:

mov tmp1, dllimgbase

add tmp1, 914 //dllimgbase+914

mov tmp2, [tmp1]

mov tmp3, lastsecbase //loc for storing sc after API

cmp tmp3, tmp2

je lab76

sub tmp2, tmp3

//dm tmp3, tmp2, "SCafAPI.bin"

shr tmp2, 2

mov SCafterAPIcount, tmp2

//log SCafterAPIcount

//msg "有高级 IAT 保护, 按确定键进行修复"

//pause

fill dllimgbase, 0E10, 00

//Advanced Import protection

find dllimgbase, #3130320D0A# //search "102"

mov tmp6, $RESULT

cmp tmp6, 0

je error

find tmp6, #8B80E4000000E8# //search "mov eax,[eax+E4]" "call xxxxxxxx"

mov tmp1, $RESULT

cmp tmp1, 0

je error

add tmp1, 6

opcode tmp1

mov func1, $RESULT\_1

//log func1

add tmp1 , 6

find tmp1, #8BC7E8????????# //search "mov eax,edi","call xxxxxxx"

mov tmp2, $RESULT

cmp tmp2, 0

je error

add tmp2, 2

opcode tmp2

mov func2, $RESULT\_1

//log func2

add tmp2, 8

mov ori1, [tmp2]

//log ori1

find tmp2, #E8????????#

mov tmp1, $RESULT

cmp tmp1, 0

je error

opcode tmp1

mov func3, $RESULT\_1

//log func3

mov tmp3, [tmp1+1]

add tmp3, tmp1

add tmp3, 5

mov tmp4, [tmp3+09]

cmp tmp4, 01B2D88B

je lab70

mov newver, 1

lab70:

//log newver

mov tmp9, eip //save eip

mov tmp1, dllimgbase

mov [tmp1], #60BB6806F400BD000BEE00BF000BEE008B57048BC3E8860900008945D88D73408B83E4000000E821250000897DDC8BF8#

add tmp1, 30 //30

mov [tmp1], #8B8BE40000008B55D88BC7E87C6000006A10B9C0B7F1008B93E40000008BC7E8E84801009090909033C08A46028D0440#

add tmp1, 30 //60

mov [tmp1], #8BD38B5482688BC7FFD28945F033C08A46038D04408BD38B5482688BC7FFD28945EC33C08A46018D04408BD38B548268#

add tmp1, 30 //90

mov [tmp1], #8BC7FFD23A434A74403A434B74423A434C0F84890000003A434D0F84800000003A434F0F84A70600003A43500F841E07#

add tmp1, 30 //C0

mov [tmp1], #00003A43510F84750700003A43520F84DC070000E907090000E9E208000090908B8BE0000000034DEC034D908B7DDC8B#

add tmp1, 30 //F0

mov [tmp1], #3F8B1F83C3068BC12BC38BD07905F7D283C20481FA81000000770BC603EB83E802884301EB09C603E983E805894301E9#

add tmp1, 30 //120

mov [tmp1], #9C0800009090909090909090909090908845D033C08945AC8945B08945B48945B88945BC8A46078D04408B5483688BC7#

add tmp1, 30 //150

mov [tmp1], #FFD28945B033C08A46058D04408B5483688BC7FFD28BD080EA080F92C280FA01750A8945B0C745B40100000033C08A46#

add tmp1, 30 //180

mov [tmp1], #088D04408B5483688BC7FFD28945B833C08A46068D04408B5483688BC7FFD28BD080EA080F92C280FA01750A8945B8C7#

add tmp1, 30 //1B0

mov [tmp1], #45BC0100000033C08A46098D04408B5483688BC7FFD284C0742EFEC87430FEC87432FEC80F8466010000FEC80F841E02#

add tmp1, 30 //1E0

mov [tmp1], #0000FEC80F8416030000FEC80F84BE030000E9E907000090E9C307000090E9BD0700009057538B7DDC8B3F8B0F83C106#

add tmp1, 30 //210

mov [tmp1], #837DB4010F85B8000000837DBC017547B83900000033D23E8A55B8C0E2033E0255B086F203C2807DB004740E807DB005#

add tmp1, 30 //240

mov [tmp1], #741166890183C102EB18668901C6410224EB0C0500400000668901C641020083C103E9D00000003E8B55B881FA800000#

add tmp1, 30 //270

mov [tmp1], #007307B883380000EB05B88138000033D23E8A55B086F203C2807DB004740E807DB005741466890183C102EB1B668901#

add tmp1, 30 //2A0

mov [tmp1], #C641022483C103EB0F0500400000668901C641020083C1033E8B55B881FA800000007307881183C101EB6C891183C104#

add tmp1, 30 //2D0

mov [tmp1], #EB658B45900145B0837DBC017521B83905000033D23E8A55B8C0E20386F203C26689013E8B55B089510283C106EB383E#

add tmp1, 30 //300

mov [tmp1], #8B55B881FA800000007317B8833D00006689013E8B45B089410288510683C107EB15B8813D00006689013E8B45B08941#

add tmp1, 30 //330

mov [tmp1], #0289510683C10A8BD9E952030000909057538B7DDC8B3F8B0F83C106837DB4010F858A060000837DBC017544B83B0000#

add tmp1, 30 //360

mov [tmp1], #0033D23E8A55B0C0E2033E0255B886F203C2807DB804740E807DB805741166890183C102EB3C668901C6410224EB0C05#

add tmp1, 30 //390

mov [tmp1], #00400000668901C641020083C103EB22B83B05000033D23E8A55B0C0E20386F203C26689013E8B55B803559089510283#

add tmp1, 30 //3C0

mov [tmp1], #C1068BD9E9C702000000000000000000#

add tmp1, 30 //3F0

mov [tmp1], #9090909090909090909090909090909057538B7DDC8B3F8B1F83C306837DB4010F859F000000837DBC017551807DB005#

add tmp1, 30 //420

mov [tmp1], #742AB83800000033D23E8A55B8C0E2033E0255B086F203C266890383C302807DB0047524C6032483C301EB1CB8384500#

add tmp1, 30 //450

mov [tmp1], #0033D23E8A55B8C0E20386F203C2668903C643020083C303E923020000807DB0047423807DB005742BB88038000033D2#

add tmp1, 30 //480

mov [tmp1], #3E8A55B086F203C26689038B55B888530283C303EB5AC703833C24008B55B8885303EB0CC703837D00008A55B8885303#

add tmp1, 30 //4B0

mov [tmp1], #83C304EB3B837DBC017521B83805000033D23E8A55B8C0E20386F203C26689033E8B55B089530283C306EB1466C70380#

add tmp1, 30 //4E0

mov [tmp1], #3D8B55B08953028A45B888430683C307E99B010000909090909090909090909057538B7DDC8B3F8B1F83C306837DB401#

add tmp1, 30 //510

mov [tmp1], #0F85CA040000837DBC017544B83A00000033D23E8A55B0C0E2033E0255B886F203C2807DB804740E807DB80574116689#

add tmp1, 30 //540

mov [tmp1], #0383C302EB39668903C6430224EB0C0500400000668903C643020083C303EB1FB83A05000033D23E8A55B0C0E20386F2#

add tmp1, 30 //570

mov [tmp1], #03C26689033E8B55B889530283C306E90C010000900000000000000000000000#

add tmp1, 30 //5A0

mov [tmp1], #0000000090909090909090909090909057538B7DDC8B3F8B1F83C306837DB4010F851A040000837DBC01751EB83BC000#

add tmp1, 30 //5D0

mov [tmp1], #0033D23E8A55B0C0E2033E0255B886F203C266890383C302EB4B3E8B55B881FA80000000731AB883F8000033C93E8A4D#

add tmp1, 30 //600

mov [tmp1], #B086E903C166890388530283C303EB258B4DB083F900750BC6033D89530183C305EB12B881F8000086E903C166890389#

add tmp1, 30 //630

mov [tmp1], #530283C306EB59909090909090909090#

add tmp1, 30 //660

add tmp1, 30 //690

mov [tmp1], #895DAC5B5F33C08A45D03A434C0F851D0300009090909090909090909090909033C08A46048D04408BD38B5482688BC7#

add tmp1, 30 //6C0

mov [tmp1], #FFD23C06740E3C07740E3C0A740E3C0B740EEB0EB00AEB0AB00BEB06B006EB02B007508B83E00000000345EC0345908B#

add tmp1, 30 //6F0

mov [tmp1], #55AC8BCA2BC87826F7D14980F980720B5883C0708802884A01EB3D5886E0050F80000066890283E904894A02EB2AF7D1#

add tmp1, 30 //720

mov [tmp1], #4181F981000000770E5883C070880283E902884A01EB115886E0050F80000066890283E906894A02E973020000000000#

add tmp1, 30 //750

mov [tmp1], #0000000000000000000000000090909033C08A46058D04408BD38B5482688BC7FFD28845EB33C08A46078D04408BD38B#

add tmp1, 30 //780

mov [tmp1], #5482688BC7FFD28BC88B7DDC8B3F8B1F83C3063D80000000771433C08A45EB86E00583C00000668903884B02EB1E33C0#

add tmp1, 30 //7B0

mov [tmp1], #8A45EB3C007508C60305894B01EB0D86E00581C00000668903894B02E9EF010000000000000000000000000000000090#

add tmp1, 30 //7E0

mov [tmp1], #33C08A46058D04408BD38B5482688BC7FFD28845EB33C08A46068D04408BD38B5482688BC7FFD28845EA8B7DDC8B3F8B#

add tmp1, 30 //810

mov [tmp1], #1F33C08A45EBC1E0030245EA86E0058BC0000066894306E9940100000000000000000000000000000000000000000000#

add tmp1, 30 //840

mov [tmp1], #33C08A46058D04408BD38B5482688BC7FFD28845EB33C08A46078D04408BD38B5482688BC7FFD28BC8034D908B7DDC8B#

add tmp1, 30 //870

mov [tmp1], #3F8B1F83C306807DEB00741733C08A45EBC0E00386E00589050000668903894B02EB06C603A3894B01E9220100000000#

add tmp1, 30 //8A0

mov [tmp1], #0000000000000090909090909090909033C08A46058D04408BD38B5482688BC7FFD28845EB33C08A46068D04408BD38B#

add tmp1, 30 //8D0

mov [tmp1], #5482688BC7FFD28845EA33C08A46078D04408BD38B5482688BC7FFD28BC88B7DDC8B3F8B1F83C306807DEB04743B3D80#

add tmp1, 30 //900

mov [tmp1], #000000771A33C08A45EAC0E0030245EB86E00589400000668903884B02EB5533C08A45EAC0E0030245EB86E005898000#

add tmp1, 30 //930

mov [tmp1], #00668903894B02EB3B3D80000000771B33C08A45EAC0E00386E00589440000668903C6430224884B03EB1933C08A45EA#

add tmp1, 30 //960

mov [tmp1], #C0E00386E00589840000668903C6430224894B03EB4A90909000000000000000#

add tmp1, 30 //990

mov [tmp1], #0000000000000000000000000000009053568BF28BD83B731C7602EB338BC6F7ABE40000000343585E5BC39000000000#

add tmp1, 30 //9C0

mov [tmp1], #8B7DDC8B0783C004833800740A8907FF4704E92AF6FFFF6190900000000000009090#

mov tmp1, dllimgbase

add tmp1, 2 //2

mov [tmp1], EBXaddr

mov tmp2, dllimgbase

add tmp2, 0B00 //dllimgbase+0B00

add tmp1, 5 //7

mov [tmp1], tmp2

add tmp1, 5 //C

mov [tmp1], tmp2

mov [tmp2], lastsecbase //loc for storing sc after API

add tmp1, 1A //26

eval "{func1}"

asm tmp1, $RESULT

add tmp1, 15 //3B

eval "{func2}"

asm tmp1, $RESULT

add tmp1, 8 //43

mov [tmp1], ori1

add tmp1, 0C //4F

eval "{func3}"

asm tmp1, $RESULT

cmp newver, 1

je lab70\_1

mov tmp1, dllimgbase

add tmp1, 54 //54

mov [tmp1], #83C40490#

lab70\_1:

mov tmp1, dllimgbase

mov tmp2, tmp1

mov tmp3, tmp1

mov tmp4, tmp1

mov tmp5, tmp1

add tmp5, A90 //dllimgbase+A90

mov [tmp5], imgbasefromdisk

add tmp3, 1F8 //cmp type 0

bp tmp3

add tmp4, 1FE //cmp type 1

bp tmp4

add tmp1, 9d8 //9d8

bp tmp1 //end point

add tmp2, 9E0 //error point

bp tmp2

mov eip, dllimgbase

eob lab71

eoe lab71

esto

lab71:

cmp eip, tmp1

je lab72

cmp eip, tmp2

je lab73

cmp eip, tmp3

je lab74

cmp eip, tmp4

je lab75

jmp error

lab72:

bc tmp1

bc tmp2

bc tmp3

bc tmp4

//msg "修复高级 IAT 保护完毕"

//pause

mov eip, tmp9 //restore eip

jmp lab76

lab73:

msg "修复高级 IAT 保护出错"

//pause

jmp end

lab74:

msg "cmp type 0"

pause

eob lab71

eoe lab71

esto

lab75:

msg "cmp type 1"

pause

eob lab71

eoe lab71

esto

lab76:

fill dllimgbase, E10, 00

fill lastsecbase, lastsecsize, 00

mov tmp1, type3count

add tmp1, E8count

mov tmp2, [EBXaddr+18]

cmp tmp1, tmp2

je lab78

msg "注意, 有些 API 没修复!"

//pause

lab78:

mov caller, "nil"

mov tmp1, [esp]

mov tmp1, dllimgbase

add tmp1, 1000

find tmp1, #C6463401# //search "mov byte[esi+34], 1"

mov tmp2, $RESULT

cmp tmp2, 0

je error

find tmp2, #68????????68????????68#

mov transit2, $RESULT

cmp transit2, 0

je error

//log transit2

bp transit2

find tmp1, #01049?43# //search "add dword ptr [edi+ebx\*4],edx" "inc ebx"

mov tmp2, $RESULT

cmp tmp2, 0

jne lab80

find tmp1, #01148740# //search "add dword ptr [edi+eax\*4],edx" "inc eax"

mov tmp2, $RESULT

cmp tmp2, 0

jne lab80

find tmp1, #3137300D0A#

cmp $RESULT, 0

jne lab80\_1

mov tmp1, [esp]

mov tmp2, [tmp1]

cmp tmp2, 68

jne lab80\_1

mov tmp2, [tmp1+5], 1

cmp tmp2, 68

jne lab80\_1

mov tmp2, [tmp1+6]

cmp tmp2, tmp1

jne lab80\_1

//Internal VM decrypt

mov VMstartaddr, tmp1

add tmp1, 20

find tmp1, #68????????68????????68#

mov VMlength, $RESULT

cmp VMlength, 0

je lab80\_1

sub VMlength, VMstartaddr

cmp VMlength, 900

ja error

log VMlength

cmp VMcodeloc, 0

jne lab78\_1

alloc 10000

mov VMcodeloc, $RESULT

lab78\_1:

log VMcodeloc

lm VMcodeloc, 4000, "d:\Asprvm8s.bin"

mov tmp1, VMcodeloc

mov tmp2, VMcodeloc

add tmp2, 3f00

add tmp1, 2

mov [tmp1], tmp2

add tmp1, 2821

asm tmp1, "call GetCurrentProcessId"

add tmp1, 56

asm tmp1, "call GetCurrentProcessId"

//copy code

mov tmp1, VMcodeloc

add tmp1, 4500 //VMcodeloc+4500

mov [tmp1], [VMstartaddr], VMlength

coe

cob

mov tmp1, VMcodeloc

mov tmp2, [VMstartaddr+B]

add tmp1, 9 //VMcodeloc+9

mov [tmp1], tmp2

mov tmp2, [VMstartaddr+6]

add tmp1, 7 //VMcodeloc+10

mov [tmp1], tmp2

add tmp1, 2CCE //VMcodeloc+2CDE--end point

bp tmp1

mov tmp9, eip

mov eip, VMcodeloc

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp9

find dllimgbase, #01049?43# //search "add dword ptr [edi+ebx\*4],edx" "inc ebx"

mov tmp2, $RESULT

cmp tmp2, 0

jne lab80

find dllimgbase, #01148740# //search "add dword ptr [edi+eax\*4],edx" "inc eax"

mov tmp2, $RESULT

cmp tmp2, 0

je lab80\_1

lab80:

add tmp2, 9

bp tmp2

lab80\_1:

eob lab80\_2

eoe lab80\_2

esto

lab80\_2:

cmp eip, tmp2

je lab81

cmp eip, transit2

je lab83

esto

lab81:

bc tmp2

mov tmp1, eip

mov tmp2, [tmp1+1]

and tmp2, 0F

cmp tmp2, 6

je lab81\_1

cmp tmp2, 7

je lab81\_2

msg "未知的 Asprotect API 寄存器"

jmp error

lab81\_1:

mov AsprAPIloc, esi

jmp lab81\_3

lab81\_2:

mov AsprAPIloc, edi

lab81\_3:

mov count, 40 //Need free space 40 bytes for 1.3x

call FindEMUAddr

//log EmuAddr

mov tmp1, eip

mov tmp1, [tmp1-3], 1

cmp tmp1, 0E

je lab81\_8

cmp tmp1, 0F

je lab81\_8

msg "未知的 Asprotect SDK API 结构"

//pause

jmp error

lab81\_8:

cmp isdll, 1

jne lab81\_9

cmp imgbasefromdisk, imgbase

je lab81\_9

mov tmp3, tmp1

mov tmp4, AsprAPIloc

loop12:

cmp tmp3, 0

je loop12\_2

mov tmp2, [tmp4]

cmp tmp2, 0

je loop12\_1

mov tmp5, tmp2

sub tmp2, imgbase

eval "{tmp5} {tmp2}(RVA)"

log $RESULT, "Aspr SDK API "

loop12\_1:

sub tmp3, 1

add tmp4, 4

jmp loop12

loop12\_2:

mov tmp3, tmp1

shl tmp3, 2

fill AsprAPIloc, tmp3, 00

jmp lab81\_16

lab81\_9:

//clear dip

mov tmp1, AsprAPIloc

mov [tmp1], 0

add tmp1, 2c

mov [tmp1], 0

//add breakpoint

mov tmp5, 0

mov tmp6, 0

mov tmp7, 0

mov tmp8, 0

mov tmp1, AsprAPIloc

add tmp1, 4

mov tmp5, [tmp1] //GetRegistrationInformation

cmp tmp5, 0

je lab81\_13

mov tmp3, 0

find tmp5, #C20400#, 100

mov tmp2, $RESULT

cmp tmp2, 0

je lab81\_9\_2

mov tmp1, tmp5

lab81\_9\_0:

findop tmp1, #E8????????#

mov tmp1, $RESULT

cmp tmp1, tmp2

ja lab81\_10

mov tmp3, [tmp1+1]

add tmp3, tmp1

add tmp3, 5

cmp tmp3, lastsecbase

ja lab81\_9\_1

cmp tmp3, 1stsecbase

jb lab81\_9\_1

mov tmp4, [tmp3]

cmp tmp4, 0D285C931

je lab81\_9\_2

mov tmp4, [tmp3+2]

cmp tmp4, D88BF28B

jne lab81\_9\_1

mov tmp4, [tmp3+6]

cmp tmp4, D38BC68B

je lab81\_9\_2

lab81\_9\_1:

add tmp1, 5

jmp lab81\_9\_0

lab81\_9\_2:

mov caller, "chkGRI"

lab81\_10:

bp tmp5

lab81\_13:

mov tmp1, AsprAPIloc

add tmp1, 10 //10

mov tmp6, [tmp1] //GetHardwareID

cmp tmp6, 0

je lab81\_14

bp tmp6

lab81\_14:

mov tmp1, AsprAPIloc

add tmp1, 30 //30

mov tmp7, [tmp1] //GetEncryptProc

cmp tmp7, 0

je lab81\_15

bp tmp7

lab81\_15:

mov tmp1, AsprAPIloc

add tmp1, 34 //34

mov tmp8, [tmp1] //GetDecryptProc

cmp tmp8, 0

je lab81\_16

bp tmp8

lab81\_16:

eoe lab82

eob lab82

esto

lab82:

cmp eip, tmp5

je 13xGRI

cmp eip, tmp6

je 13xGHI

cmp eip, tmp7

je 13xGEP

cmp eip, tmp8

je 13xGDP

cmp eip, transit2

je lab90

esto

13xGRI:

bc tmp5

scmp caller, "chkGRI"

jne 13xGRI\_2

coe

cob

mov tmp2, [esp]

mov tmp1, esp

add tmp1, 4

mov tmp3, EmuAddr

add tmp3, 4

mov [tmp1], tmp3 //put blank first

eval "eip == 0{tmp2}"

tocnd $RESULT

13xGRI\_1:

mov caller, "nil"

jmp 13xGRI\_3

13xGRI\_2:

mov tmp2, EmuAddr

add tmp2, 4

mov tmp1, esp

add tmp1, 4

mov [tmp1], tmp2

13xGRI\_3:

mov [EmuAddr], #04000000566F6C58# //"VolX"

log EmuAddr, "GetRegistrationInformation "

add EmuAddr, 10

//msg "13xGRI"

//pause

eoe lab82

eob lab82

esto

13xGHI:

bc tmp6

mov [EmuAddr], #31323334353637382D34343434# //"12345678-4444"

mov tmp1, esp

add tmp1, 4

mov [tmp1], EmuAddr

log EmuAddr, "GetHardwareID "

add EmuAddr, 10

//msg "13xGHI"

//pause

eoe lab82

eob lab82

esto

13xGEP:

bc tmp7

mov tmp1, esp

add tmp1, 4

mov [tmp1], EmuAddr

log EmuAddr, "GetEncryptProc "

add EmuAddr, 10

//msg "13xGEP"

//pause

mov tmp1, AsprAPIloc

add tmp1, 30

mov [tmp1], 0

eoe lab82

eob lab82

esto

13xGDP:

bc tmp8

mov [EmuAddr], #C3#

mov tmp1, esp

add tmp1, 4

mov [tmp1], EmuAddr

log EmuAddr, "GetDecryptProc "

//msg "13xGDP"

//pause

mov tmp1, AsprAPIloc

add tmp1, 34

mov [tmp1], 0

eoe lab82

eob lab82

esto

//Fix VB Aspr SDK API

lab83:

cmp isdll, 1

je lab90

cmp DFCaddr, 0

je lab90

GMEMI iatendaddr, MEMORYBASE

mov tmp1, $RESULT

cmp tmp1, 0

je error

cmp tmp1, 1stsecbase

jne lab90

bc transit2

cob

coe

mov tmp1, dllimgbase

mov [tmp1], #609CB8FF000000BF00104000B900100D00F2AEE376803F2575F78B5F0181FB0010400072EC81FB00204D0077E48B1381#

add tmp1, 30

mov [tmp1], #FA19A0006675DA8BF74E909090909090BD0002EF00BF00104000B900100D00B8B8000000F2AEE333393775F8807FFA68#

add tmp1, 30

mov [tmp1], #75F28B5FFB8B5304833A1077E7837A040075E18BDF83EB11803BA175D7895D008B1A4B895D0483C508EBC99D61909000#

mov tmp1, dllimgbase

add tmp1, 8

mov [tmp1], 1stsecbase

add tmp1, 5 //0D

mov [tmp1], 1stsecsize

add tmp1, 12 //1F

mov [tmp1], 1stsecbase

add tmp1, 8 //27

mov tmp2, 1stsecbase

add tmp2, 1stsecsize

mov [tmp1], tmp2

add tmp1, 0A //31

mov [tmp1], DFCaddr

add tmp1, 10 //41

mov [tmp1], thunkdataloc

add tmp1, 5 //46

mov [tmp1], 1stsecbase

add tmp1, 5 //4B

mov [tmp1], 1stsecsize

add tmp1, 42 //8D -- end point

bp tmp1

mov tmp7, eip

mov eip, dllimgbase

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7

fill dllimgbase, 100, 00

mov count, 160 //Need free space 160 bytes for VB

call FindEMUAddr

lab84:

add EmuAddr, 40 //put extra space

mov tmp5, 0 //counter

mov tmp1, AsprAPIloc

add tmp1, 4

mov tmp6, thunkdataloc

mov caller, "lab84"

jmp lab46\_2

lab85:

mov caller, "nil"

fill thunkdataloc, 100, 00

lab90:

bc transit2

cmp VMstartaddr, 0

je lab90\_1

mov tmp1, [VMcodeloc+4500]

cmp tmp1, 0

je lab90\_1

mov tmp1, VMcodeloc

add tmp1, 4514 //skip first 14 bytes

mov tmp2, VMstartaddr

add tmp2, 14 //skip first 14 bytes

mov tmp3, VMlength

sub tmp3, 14 //skip first 14 bytes

mov [tmp2], [tmp1], tmp3

fill VMcodeloc, 5000, 00

mov VMstartaddr, 0

lab90\_1:

cob

coe

mov caller, "nil"

mov tmp1, dllimgbase

add tmp1, 1000

find tmp1, #3135330D0A# //search ASCII"153"

mov tmp2, $RESULT

sub tmp2, 40

find tmp2, #5?5?C3#

mov tmp3, $RESULT

cmp tmp3, 0

je error

add tmp3, 2

rtr

bp tmp3

eob lab91

eoe lab91

esto

lab91:

cmp eip, tmp3

je lab92

esto

lab92:

bc tmp3

mov tmp1, dllimgbase

add tmp1, 1000

find tmp1, #3130330D0A# //search ASCII"103"

mov tmp2, $RESULT

cmp tmp2, 0

je wrongver

find tmp2, #8D00C3# //search "lea eax,[eax]" "ret"

mov tmp1, $RESULT

cmp tmp1, 0

je wrongver

bphws tmp1, "x"

eob lab93

eoe lab93

esto

lab93:

cmp eip, tmp1

je lab94

esto

lab94:

bphwc tmp1

cob

coe

find eip, #C700E1000000#

mov tmp1, $RESULT

cmp tmp1, 0

jne lab95

find eip, #C600E1#

mov tmp1, $RESULT

cmp tmp1, 0

je error

lab95:

find tmp1, #A1????????894?# //search "mov eax, [xxxxxxxx]","mov [e?p+??],reg32"

mov tmp3, $RESULT

cmp tmp3, 0

je error

mov tmp2, 0

mov tmp2, [tmp3+1]

mov tmp1, [tmp2]

cmp tmp1, 0

jne lab99

lab98:

rtr

sti

GMEMI eip, MEMORYOWNER

mov tmp3, $RESULT

mov tmp2, lastsecbase

add tmp2, lastsecsize

cmp tmp3, tmp2

ja lab98\_1

cmp 1stsecbase, tmp3

jb error

GMEMI eip, MEMORYSIZE

mov tmp1, $RESULT

add tmp3, tmp1

eval "eip > 0{tmp3}"

jmp lab98\_2

lab98\_1:

eval "eip < 0{tmp3}"

lab98\_2:

ticnd $RESULT

mov tmp1, eip

sub tmp1, imgbase

mov OEP\_rva, tmp1

cmp sdksccount, 0

je lab141 //Go to dump file

mov tmp3, eip

jmp lab104

lab99:

bp tmp1

eob lab99\_1

eoe lab99\_1

esto

lab99\_1:

cmp eip, tmp1

je lab99\_2

esto

lab99\_2:

bc tmp1

mov OEPscaddr, eip

find eip, #0000000000000000#

mov patchaddr, $RESULT

mov tmp1, patchaddr

add tmp1, 8

mov tmp4, 10

loop16:

cmp tmp4, 0

je notfound

mov tmp2, [tmp1], 1

cmp tmp2, 0

jne lab100

add tmp1, 1

sub tmp4, 1

jmp loop16

lab100:

add tmp1, 3

mov tmp2, [tmp1], 1

//and tmp2, ff

cmp tmp2, 0

jne error

sub tmp1, b

mov vcrefend, tmp1

sub tmp1, 4

mov tmp4, 200

mov count, 0

loop17:

cmp tmp4, 0

je notfound

mov tmp2, [tmp1]

cmp tmp2, 00000000

je lab101

sub tmp1, 8

sub tmp4, 8

jmp loop17

lab101:

cmp count, 1

je lab102

add count, 1

sub tmp1, 8

sub tmp4, 8

jmp loop17

lab102:

mov tmp4, tmp1

add tmp4, 4

mov vcrefstart, tmp4

loop18:

cmp tmp4, vcrefend

jae lab103

mov tmp1, [tmp4]

add tmp1, imgbase

eval "{tmp1}"

add tmp4, 4

mov tmp2, [tmp4]

add tmp2, OEPscaddr //tmp2== address to put comment

cmt tmp2, $RESULT

add tmp4, 4

jmp loop18

lab103:

mov tmp1, vcrefend

sub tmp1, vcrefstart

mov sttablesize, tmp1

dm vcrefstart, sttablesize, "st\_table.bin"

GCMT eip

mov tmp1, $RESULT

ATOI tmp1

mov tmp2, $RESULT

sub tmp2, imgbase

mov OEP\_rva, tmp2

mov tmp3, $RESULT

lab104:

mov tmp1, lastsecbase

add tmp1, lastsecsize

lab106\_1:

mov virtualsec, tmp1

mov tmp1, 0

cmp SDKsize, 0

je lab106\_2

//With SDK stolen section

mov newphysecsize, SDKsize

lab106\_2:

cmp OEPscaddr, 0

je lab106\_3

//With OEP stolen code

GMEMI OEPscaddr, MEMORYSIZE

mov tmp2, $RESULT

add newphysecsize, tmp2

lab106\_3:

cmp 55sc, 1

jne lab106\_4

//wz std function

add newphysecsize, 1000

lab106\_4:

add newphysecsize, 1000 //extra 1000 bytes

alloc newphysecsize

mov newphysec, $RESULT

//log newphysec

cmp dataloc, 0

jne lab106\_5

alloc 4000

mov dataloc, $RESULT

//log dataloc

jmp lab106\_6

lab106\_5:

fill dataloc, 4000, 00 //clear data

lab106\_6:

cmp OEPscaddr, 0

je lab121

//analyse OEP stolen code

find dllimgbase, #33340D0A#

mov tmp1, $RESULT

cmp tmp1, 0

je error

find tmp1, #FF35????????68#

mov tmp2, $RESULT

cmp tmp2, 0

je error

mov tmp1, [tmp2+2]

mov scstk, [tmp1]

//log scstk

//chk free space

mov patchaddr, vcrefend

add patchaddr, 20

and patchaddr, fffffff0

//log patchaddr

GMEMI OEPscaddr, MEMORYSIZE

mov tmp1, $RESULT

GMEMI OEPscaddr, MEMORYOWNER

mov tmp2, $RESULT

mov tmp3, tmp1

//Assume every 1000 bytes will need A0 bytes of free space

shr tmp3, 0C

mov tmp4, tmp3

shl tmp3, 7

shl tmp4, 5

add tmp3, tmp4

//log tmp3, "Free space need = "

add tmp1, tmp2

sub tmp1, patchaddr

//log tmp1, "Free space exist = "

cmp tmp1, tmp3

ja lab107

mov patchaddr, lastsecbase

jmp lab108

lab107:

mov patchinsamesec, 1

lab108:

call FillSCPatch

lab109:

mov caller, "nil"

mov tmp1, dllimgbase

mov tmp2, dataloc

add tmp2, 800 //dataloc+800

mov tmp3, tmp1

add tmp3, 0D00 //dllimgbase+D00

add tmp1, 5 //5

mov [tmp1], tmp3

add tmp1, 5 //0A

mov [tmp1], scstk

add tmp1, 0D //17

mov [tmp1], tmp2

add tmp1, 2A //41

mov [tmp1], vcrefstart

add tmp1, 19 //5A

mov [tmp1], tmp2

add tmp1, 7 //61

mov [tmp1], patchaddr

add tmp1, 5 //66

mov [tmp1], scstk

add tmp1, 77F //7E5

mov [tmp1], vcrefstart

add tmp1, d //7F2

mov [tmp1], vcrefend

mov tmp4, dllimgbase

add tmp4, C9C

mov tmp1, dataloc

add tmp1, 1000

mov [tmp4], tmp1

add tmp4, 4

mov [tmp4], dataloc

mov tmp4, dllimgbase

add tmp4, 7D9 //end point

bp tmp4

mov tmp5, tmp4

add tmp5, 7 //error point 7E0

bp tmp5

mov tmp7, eip //save eip

mov eip, dllimgbase

eob lab110

eoe lab110

esto

lab110:

cmp eip, tmp5

je patcherr

cmp eip, tmp4

je lab111

jmp error

lab111:

bc tmp4

bc tmp5

mov eip, tmp7

mov tmp1, dllimgbase

add tmp1, CAC

mov patchendaddr, [tmp1]

//msg "OEP 偷代码分析完毕!"

//pause

fill dllimgbase, 0d00, 00 //cleaning location storing call xxxxxxxx address

mov curzeroVA, eip

mov newzeroVA, newphysec

mov virzeroVA, virtualsec

mov tmp1, vcrefend

mov tmp2, [tmp1+0C]

add tmp2, OEPscaddr

mov findendaddr, tmp2

mov caller1, "lab111"

jmp lab160 //copy code to new section

lab113:

mov caller1, "nil"

cmp patchinsamesec, 1

je lab121

fill lastsecbase, lastsecsize, 00

mov patchinsamesec, 0 //restore flag

//Analyse SDK stolen code

lab121:

cmp sdksccount, 0

je lab141

mov count, 0 //counter for fixed sdk stolen code section

mov tmp1, [xtrascloc]

cmp tmp1, 0

je lab150

lab122:

mov tmp1, dllimgbase

add tmp1, EF0 //dllimgbase+EF0

mov [tmp1], xtrascloc

lab123:

mov tmp1, dllimgbase

add tmp1, EF0

mov tmp4, [tmp1]

mov scstk, [tmp4]

cmp scstk, 0

je lab150

//log scstk

add tmp4, 4

mov [tmp1], tmp4 //address point to next stolen code section

mov sdkscaddr, [scstk+18]

cmp sdkscaddr, 0

je lab131

log sdkscaddr, "SDK 偷窃代码区段地址 = "

find sdkscaddr, #0000000000000000#

mov findendaddr, $RESULT

add findendaddr, 8

mov patchaddr, findendaddr

add patchaddr, 10

and patchaddr, fffffff0

//log patchaddr

//Check if the freespace is sufficinet

GMEMI findendaddr, MEMORYOWNER

mov tmp1, $RESULT

GMEMI patchaddr, MEMORYOWNER

mov tmp2, $RESULT

cmp tmp1, tmp2

jne lab124

GMEMI findendaddr, MEMORYSIZE

mov tmp1, $RESULT

//log tmp1, "区段大小 = "

mov tmp3, tmp1

//Assume every 1000 bytes will need C0 bytes of free space

shr tmp3, 0C

mov tmp4, tmp3

shl tmp3, 7

shl tmp4, 6

add tmp3, tmp4

//log tmp3, "Free space need = "

add tmp1, tmp2

sub tmp1, patchaddr

//log tmp1, "Free space exist = "

cmp tmp1, tmp3

ja lab125

lab124:

mov patchaddr, lastsecbase

mov patchinsamesec, 0

jmp lab126

lab125:

mov patchinsamesec, 1

lab126:

call FillSCPatch

lab127:

mov tmp1, dllimgbase

mov tmp2, dataloc

add tmp2, 800 //dataloc+800

mov tmp3, tmp1

add tmp3, 0D00 //dllimgbase+D00

add tmp1, 5 //5

mov [tmp1], tmp3

add tmp1, 5 //0A

mov [tmp1], scstk

add tmp1, 0D //17

mov [tmp1], tmp2

add tmp1, 2A //41

mov [tmp1], findendaddr

add tmp1, 19 //5A

mov [tmp1], tmp2

add tmp1, 7 //61

mov [tmp1], patchaddr

add tmp1, 5 //66

mov [tmp1], scstk

add tmp1, A7 //10D

mov [tmp1], #18#

add tmp1, 6D7 //7E4

mov [tmp1], #C390909090#

mov tmp4, dllimgbase

add tmp4, C9C

mov tmp1, dataloc

add tmp1, 1000

mov [tmp4], tmp1

add tmp4, 4

mov [tmp4], dataloc

mov tmp4, dllimgbase

add tmp4, 7D9 //end point

bp tmp4

mov tmp5, tmp4

add tmp5, 7 //error point 7E0

bp tmp5

mov tmp7, eip //save eip

mov eip, dllimgbase

eob lab128

eoe lab128

esto

lab128:

cmp eip, tmp5

je patcherr

cmp eip, tmp4

je lab129

jmp error

lab129:

bc tmp4

bc tmp5

mov eip, tmp7 //restore eip

//msg "SDk 区段偷代码分析完毕!"

//pause

mov patchendaddr, [dllimgbase+0CAC]

lab130:

add count, 1

fill dllimgbase, 0d00, 00 //cleaning location storing call xxxxxxxx address

lab131:

mov curzeroVA, sdkscaddr

lab132:

cmp newpatchaddr, 0 //1st stolen code section ?

jne lab133

mov virzeroVA, virtualsec

mov newzeroVA, newphysec

jmp lab134

lab133:

mov tmp1, newpatchendaddr

and tmp1, 0FFFFFF00

add tmp1, 200

mov newzeroVA, tmp1

sub tmp1, newphysec //offset

add tmp1, virtualsec

mov virzeroVA, tmp1

lab134:

mov caller1, "lab134"

mov eip, tmp7

jmp lab160 //move code to new section

lab135:

mov caller1, "nil"

lab137:

fill dataloc, 4000, 00 //clear data

cmp patchinsamesec, 1

je lab138

fill lastsecbase, lastsecsize, 00 //clear last sec

lab138:

mov tmp4, [dllimgbase+EF0]

mov scstk, [tmp4]

//log scstk

cmp scstk, 0 //Process all SDK section with scstk ?

jne lab123

//Process SDK section without scstk

mov tmp9, newpatchendaddr

mov tmp1, dllimgbase

add tmp1, 0E00

mov tmp8, xtrascloc

add tmp8, 80

mov [tmp1], tmp8

lab139:

mov tmp1, dllimgbase

add tmp1, 0E00

mov tmp8, [tmp1]

mov tmp6, [tmp8]

cmp tmp6, 0

je lab141

and tmp9, 0FFFFFF00

add tmp9, 200

mov newzeroVA, tmp9

sub tmp9, newphysec //offset

add tmp9, virtualsec

mov virzeroVA, tmp9

mov curzeroVA, [tmp8+4]

mov sdkscaddr, [tmp8+4]

find curzeroVA, #000000000000000000000000#

mov tmp4, $RESULT

cmp tmp4, 0

je error

sub tmp4, curzeroVA //size to copy

mov tmp1, dllimgbase

mov [tmp1], #609CBE0039F600BF00296900B990000000F2A49D619090000000000000000000#

mov tmp1, dllimgbase

add tmp1, 3

mov [tmp1], curzeroVA

add tmp1, 5 //8

mov [tmp1], newzeroVA

add tmp1, 5 //D

mov [tmp1], tmp4

add tmp1, 8 //15 --end point

bp tmp1

mov tmp7, eip

mov eip, dllimgbase

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7

fill dllimgbase, 100, 00

mov tmp9, newzeroVA

add tmp9, tmp4

mov newpatchendaddr, tmp9

mov caller1, "lab139"

jmp lab180

lab140:

mov caller1, "nil"

mov tmp1, dllimgbase

add tmp1, 0E00

mov tmp8, [tmp1]

add tmp8, 8

mov [tmp1], tmp8

mov tmp9, newpatchendaddr

jmp lab139

lab141:

cmp 55sc, 0

je lab143

cmp newphysec, 0

jne lab141\_1

alloc 1000

mov newphysec, $RESULT

mov newzeroVA, newphysec

mov tmp1, lastsecbase

add tmp1, lastsecsize

mov virtualsec, tmp1

mov virzeroVA, virtualsec

mov tmp1, 55dataloc

jmp lab141\_2

lab141\_1:

mov tmp1, newpatchendaddr

and tmp1, 0FFFFFF00

add tmp1, 200

mov newzeroVA, tmp1

cmp virtualsec, 0

je error

sub tmp1, newphysec //offset

add tmp1, virtualsec

mov virzeroVA, tmp1

mov tmp1, 55dataloc

//process std function

lab141\_2:

mov tmp2, [tmp1]

cmp tmp2, 0

je lab143

log tmp2, "标准函数在 "

mov tmp3, 0

mov tmp3, [tmp2], 1

cmp tmp3, 0e9

je lab141\_3

cmp tmp3, 68

jne error

mov tmp4, [tmp2+1]

jmp lab141\_4

lab141\_3:

GCI tmp2, DESTINATION

mov tmp4, $RESULT

lab141\_4:

find tmp4, #0000000000000000#

mov tmp5, $RESULT

cmp tmp5, 0

je error

sub tmp5, tmp4

mov [newzeroVA], [tmp4], tmp5

cmp tmp3, 0e9

je lab141\_5

cmp tmp3, 68

jne error

eval "push 0{virzeroVA}"

asm tmp2, $RESULT

jmp lab141\_6

lab141\_5:

eval "jmp 0{virzeroVA}"

asm tmp2, $RESULT

lab141\_6:

add newzeroVA, tmp5

add newzeroVA, 20

add virzeroVA, tmp5

add virzeroVA, 20

add tmp1, 4

jmp lab141\_2

lab143:

cmp newphysec, 0

je lab144

mov tmp1, lastsecbase

add tmp1, lastsecsize

cmp tmp1, virtualsec

je lab144

eval "All\_{virtualsec}.bin"

DM newphysec, newphysecsize, $RESULT

lab144:

log iatstartaddr, "IAT 的地址 = "

log iatstart\_rva, "IAT 的相对地址 = "

log iatsize, "IAT 的大小 = "

mov tmp3, OEP\_rva

add tmp3, imgbase

GPI PROCESSNAME

mov tmp6, $RESULT

cob

coe

mov tmp1, dllimgbase

mov [tmp1], #609C546A4068001000006800004000E88A160577B80002400033D2668B50068BF081C600010000B9080000008BFE83C7#

add tmp1, 30 //30

mov [tmp1], #08F2A4664A6683FA00740583C620EBE783C618C70661737072C7460800200000C7460C00003D01C7461000200000C746#

add tmp1, 30 //60

mov [tmp1], #1400003D01C74624400000E066FF4006814050002000009D6190900000000000#

mov tmp1, dllimgbase

add tmp1, 0B

mov [tmp1], imgbase

add tmp1, 4 //0F

asm tmp1, "call VirtualProtect"

add tmp1, 6 //15

mov [tmp1], signVA

cmp newphysec, 0 //with stolen code section?

je lab145

mov tmp4, lastsecbase

add tmp4, lastsecsize

cmp tmp4, virtualsec

jne lab145

add tmp1, 37 //4C

mov [tmp1], newphysecsize

mov tmp4, lastsecbase

add tmp4, lastsecsize

sub tmp4, imgbase

add tmp1, 7 //53

mov [tmp1], tmp4

add tmp1, 7 //5A

mov [tmp1], newphysecsize

add tmp1, 7 //61

mov [tmp1], tmp4

add tmp1, 12 //73

mov [tmp1], newphysecsize

add tmp1, 6 //79 -- end point

jmp lab145\_1

lab145:

mov tmp1, dllimgbase

add tmp1, 40

mov [tmp1], #9D619090#

add tmp1, 2 //42 -- end point

lab145\_1:

bp tmp1

mov tmp7, eip

mov eip, dllimgbase

eob lab145\_2

eoe lab145\_2

run

lab145\_2:

cmp eip, tmp1

je lab145\_3

jmp error

lab145\_3:

bc tmp1

mov eip, tmp7

fill dllimgbase, 100, 00

mov tmp1, signVA

add tmp1, 3C //signVA+3C -- FileAlignment

mov [tmp1], 1000

add tmp1, 18 //signVA+54 -- SizeOfHeaders

mov [tmp1], 1000

cmp isdll, 0

je lab146

mov tmp4, 0

mov tmp2, reloc\_rva

add tmp2, imgbase

loop19:

mov tmp5, [tmp2+4]

cmp tmp5, 0

je lab145\_4

add tmp4, tmp5

add tmp2, tmp5

jmp loop19

lab145\_4:

mov reloc\_size, tmp4

add tmp1, 4C //signVA+A0 -- RVA of Relocation Table

mov [tmp1], reloc\_rva

add tmp1, 4 //signVA+A4 -- Size of Relocation Table

mov [tmp1], reloc\_size

log reloc\_rva, "重定位区段相对地址 = "

log reloc\_size, "重定位区段大小 = "

eval "de\_{tmp6}.dll"

mov tmp5, $RESULT

log tmp3, "OEP 地址 = "

log OEP\_rva, "OEP 相对地址 = "

mov tmp1, lastsecbase

add tmp1, lastsecsize

sub tmp1, imgbase

dm imgbase, tmp1, tmp5 //dump file

cmp newphysec, 0 //with stolen code section?

je lab147

mov tmp1, lastsecbase

add tmp1, lastsecsize

cmp tmp1, virtualsec

jne lab147

dma newphysec, newphysecsize, tmp5 //add stolen code section

jmp lab147

lab146:

add tmp1, 4C //signVA+A0 -- RVA of Relocation Table

mov [tmp1], 0

add tmp1, 4 //signVA+A4 -- Size of Relocation Table

mov [tmp1], 0

eval "de\_{tmp6}.exe"

mov tmp5, $RESULT

log tmp3, "OEP 的地址 = "

log OEP\_rva, "OEP 的相对地址 = "

mov tmp1, lastsecbase

add tmp1, lastsecsize

sub tmp1, imgbase

dm imgbase, tmp1, tmp5 //dump file

cmp newphysec, 0 //with stolen code section?

je lab147

mov tmp1, lastsecbase

add tmp1, lastsecsize

cmp tmp1, virtualsec

jne lab147

dma newphysec, newphysecsize, tmp5 //add stolen code section

lab147:

cmp newphysec, 0

je lab148

mov tmp1, lastsecbase

add tmp1, lastsecsize

cmp tmp1, virtualsec

jne lab147\_1

msg "有偷窃代码, 请查看记录窗口内的 IAT 数据"

pause

jmp end

lab147\_1:

msg "有偷窃代码, 先补区段后再修复 IAT"

pause

jmp end

lab148:

msg "没有偷窃代码, 请查看记录窗口内的 IAT 数据"

pause

jmp end

lab150:

msg "lab150"

pause

jmp end

//relocate Call command stolen code

lab160:

//log patchendaddr

mov tmp1, dllimgbase

mov [tmp1], #609CBE34027B02BF00007D01B922040000F2A4BD000259018B45008B0083F800741A8BD881EB3402FE008B530181C234#

add tmp1, 30

mov [tmp1], #D27E0189530183450004EBDC9D619090#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov [tmp1], curzeroVA

add tmp1, 5 //8

mov [tmp1], newzeroVA

add tmp1, 5 //0D

mov tmp2, findendaddr

sub tmp2, curzeroVA //bytes to copy

mov [tmp1], tmp2

add tmp1, 7 //14

mov tmp2, dllimgbase

add tmp2, 200

mov [tmp1], tmp2

mov [tmp2], dataloc

add tmp1, 12 //26

mov tmp2, curzeroVA

sub tmp2, newzeroVA

mov [tmp1], tmp2

mov tmp1, dllimgbase

add tmp1, 2F //2F

cmp curzeroVA, virtualsec

ja lab161

mov tmp2, virzeroVA

sub tmp2, curzeroVA

mov [tmp1], tmp2

mov tmp1, dllimgbase

add tmp1, 2D //2D

mov [tmp1], #81EA#

jmp lab162

lab161:

mov tmp2, curzeroVA

sub tmp2, virzeroVA

mov [tmp1], tmp2

lab162:

coe

cob

mov tmp1, dllimgbase

add tmp1, 3E //end point

mov tmp7, eip //save eip

mov eip, dllimgbase

bp tmp1

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7 //restore eip

fill dllimgbase, 500, 00

scmp caller1, "lab134"

je lab164\_1

//copy and relocate jxx analysed code

//Decide new patch addr

//for Stolen code at OEP

lab163:

cmp patchinsamesec, 1

je lab163\_1

lab163\_1:

mov tmp1, findendaddr

sub tmp1, curzeroVA //offset

add tmp1, newzeroVA

mov tmp2, tmp1

and tmp2, 0ff

cmp tmp2, 0

je lab164

and tmp1, 0FFFFFFF0

add tmp1, 20

jmp lab165

lab164:

and tmp1, 0FFFFFFF0

add tmp1, 10

jmp lab165

//for SDK section

lab164\_1:

cmp patchinsamesec, 1

je lab164\_2

mov tmp1, findendaddr

sub tmp1, curzeroVA

and tmp1, 0FFFFFFF0

add tmp1, 20

add tmp1, newzeroVA

jmp lab165

lab164\_2:

mov tmp1, patchaddr

sub tmp1, curzeroVA //offset

add tmp1, newzeroVA

lab165:

mov newpatchaddr, tmp1

//log newpatchaddr

mov tmp1, dllimgbase

mov [tmp1], #609CBD000DD900BE003ED800BF2018BD01B969000000F2A49090BE0010BE018B0683F8000F84C600000083F8030F844D#

add tmp1, 30 //30

mov [tmp1], #0000008B4DE08B460403C18B55DC8BDA2BD083EA058950018B460803C12BC383E80689430283C3068B460C03C12BC383#

add tmp1, 30 //60

mov [tmp1], #E80589430183C305895DDC83C610EBAF000000000000000000000000000000008B4DE08B460403C18B55DC8BDA2BD083#

add tmp1, 30 //90

mov [tmp1], #EA05895001608BF333D2668B1681E2FFF0000081FA0F800000740346EBEA807E06E975F78975DC618B4DE08B55DC8BDA#

add tmp1, 30 //C0

mov [tmp1], #8B460803C12BC383E80689430283C3068B460C03C12BC383E80589430183C305895DDC83C610E934FFFFFF0000000090#

add tmp1, 30 //F0

mov [tmp1], #9D619090#

mov tmp1, dllimgbase

mov tmp2, dllimgbase

add tmp2, 0D00

add tmp1, 3 //3

mov [tmp1], tmp2

add tmp1, 5 //8

mov [tmp1], patchaddr

add tmp1, 5 //0D

mov [tmp1], newpatchaddr

add tmp1, 5 //12

mov tmp3, patchendaddr

sub tmp3, patchaddr //bytes to copy

mov [tmp1], tmp3

mov newpatchendaddr, tmp3

add newpatchendaddr, newpatchaddr

add tmp1, 9 //1B

mov tmp2, dataloc

add tmp2, 1000

mov [tmp1], tmp2

mov tmp2, dllimgbase

add tmp2, 0CDC

mov [tmp2], newpatchaddr

add tmp2, 4

mov [tmp2], newzeroVA

mov tmp1, dllimgbase

add tmp1, 0F2 //end point

mov tmp7, eip

mov eip, dllimgbase

bp tmp1

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7

fill dllimgbase, D00, 00

fill dataloc, 4000, 00

scmp caller1, "lab134"

je lab180

lab166:

lm dataloc, sttablesize, "st\_table.bin"

mov tmp1, dllimgbase

mov [tmp1], #609CBE0000BE01BB00004000B900906A008B0683F800741603C38B560403D18BFA2BF883EF0589780183C608EBE39D61#

add tmp1, 30

mov [tmp1], #90909000#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov [tmp1], dataloc

add tmp1, 5 //8

mov [tmp1], imgbase

add tmp1, 5 //0D

mov [tmp1], virzeroVA

add tmp1, 23 //30 -- end point

mov tmp7, eip

mov eip, dllimgbase

bp tmp1

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7

fill dllimgbase, 100, 00

fill dataloc, sttablesize, 00

jmp lab190

//For SDK stolen code

//relocate analysed patch code

lab180:

//log sdkscaddr

//log scstk

lm dataloc, jmptablesize, "jmptable.bin"

mov tmp9, dataloc

lab181:

mov tmp2, [tmp9]

cmp tmp2, 0

je error

mov tmp3, [tmp9+4]

add tmp3, imgbase

mov tmp4, [tmp3+1]

add tmp4, tmp3

add tmp4, 5

cmp tmp4, sdkscaddr

je lab182

add tmp9, tmp2

add tmp9, 04

jmp lab181

lab182:

mov tmp6, [tmp9] //length

add tmp9, 04

mov tmp5, dataloc

add tmp5, 800

lab183:

cmp tmp6, 0

je lab189

mov tmp2, [tmp9]

mov [tmp5], tmp2

add tmp9, 4

add tmp5, 4

sub tmp6, 4

jmp lab183

lab189:

mov tmp1, dllimgbase

mov [tmp1], #609CBE0000BE01BB00004000B900906A008B0683F800741603C38B560403D18BFA2BF883EF0589780183C608EBE39D61#

add tmp1, 30

mov [tmp1], #90909000#

mov tmp1, dllimgbase

add tmp1, 3 //3

mov tmp3, dataloc

add tmp3, 800

mov [tmp1], tmp3

add tmp1, 5 //8

mov [tmp1], imgbase

add tmp1, 5 //0D

mov [tmp1], virzeroVA

add tmp1, 23 //30 -- end point

mov tmp7, eip

mov eip, dllimgbase

bp tmp1

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp7

fill dllimgbase, 100, 00

fill dataloc, 1000, 00

lab190:

scmp caller1, "lab111"

je lab113

scmp caller1, "lab134"

je lab135

scmp caller1, "lab139"

je lab140

error:

msg "错误!"

pause

jmp end

wrongver:

find dllimgbase, #0038310D0A#

mov tmp1, $RESULT

cmp tmp1, 0

je wrongver\_1

msg "本脚本不支持这版的 Asprotect, 可能是 Aspr 1.31 或 v2.0 alpha 所加壳."

pause

jmp end

wrongver\_1:

find dllimgbase, #0031350D0A#

mov tmp1, $RESULT

cmp tmp1, 0

je wrongver\_2

msg "本脚本不支持这版的 Asprotect, 可能是 Aspr 1.2x 所加壳."

pause

jmp end

wrongver\_2:

msg "本脚本不支持这版的 Asprotect."

pause

jmp end

error45:

msg "错误 45!"

pause

jmp end

odbgver:

msg "本脚本须配合 ODbgscript 1.64 或以上的版本"

jmp end

notfound:

msg "Not found"

pause

jmp end

patcherr:

msg "分析偷窃代码时出现错误"

pause

end:

ret

//

//

//

ChkRelocSize:

find tmp1, #0000000000000000#

mov tmp2, $RESULT

sub tmp2, imgbase

sub tmp2, reloc\_rva

mov tmp3, tmp2

and tmp3, 0F

mov tmp4, tmp3

shr tmp4, 2

shl tmp4, 2

cmp tmp4, tmp3

je ChkRelocSize\_1

add tmp2, 2

ChkRelocSize\_1:

ret

FindEMUAddr:

//find freespace

cob

coe

mov tmp1, dllimgbase

mov [tmp1], #609CB900040000B800000000BF90909000FDF3AFE30383C70483C704893D3000C9009D61909090000000000000000000#

add tmp1, D //0D

mov tmp2, 1stsecbase

add tmp2, 1stsecsize

sub tmp2, 4

mov [tmp1], tmp2

add tmp1, 11 //1E

mov tmp2, dllimgbase

add tmp2, 30

mov [tmp1], tmp2

add tmp1, 6 //24 -- end point

bp tmp1

mov tmp3, eip

mov eip, dllimgbase

run

cmp eip, tmp1

jne error

bc tmp1

mov eip, tmp3

mov tmp2, [dllimgbase+30]

mov tmp3, tmp2

and tmp3, 0f

mov tmp4, 10

sub tmp4, tmp3

add tmp2, tmp4

add tmp2, 10

mov EmuAddr, tmp2

log EmuAddr

fill dllimgbase, 34, 00

mov tmp1, 1stsecbase

add tmp1, 1stsecsize

cmp EmuAddr, tmp1

jae FindEMUAddr\_3

sub tmp1, tmp2

cmp tmp1, count //freespace compare with count bytes (2.xx=120 bytes, 1.3x=40 bytes)

jae FindEMUAddr\_6

FindEMUAddr\_3:

cmp isdll, 1

je FindEMUAddr\_4

mov tmp1, imgbase

add tmp1, 0D00

mov EmuAddr, tmp1

jmp FindEMUAddr\_6

FindEMUAddr\_4:

ask "请键入存放 Asprotect SDk API 模拟代码的地址 (须最少 120 字节)"

cmp $RESULT, 0

je error

mov EmuAddr, $RESULT

cmp EmuAddr, 1stsecbase

jb FindEMUAddr\_5

mov tmp1, lastsecbase

add tmp1, lastsecsize

cmp tmp1, EmuAddr

jb FindEMUAddr\_5

//log EmuAddr

jmp FindEMUAddr\_6

FindEMUAddr\_5:

msg "这个地址不适用"

jmp FindEMUAddr\_4

FindEMUAddr\_6:

mov count, 0 //clear

ret

FillSCPatch:

mov tmp1, dllimgbase

mov [tmp1], #6083EC60BD000D5901BB000660018B43188945A4C745A8000859018B7DA4803FE875188B4F0103CF83C1053B4B1C750B#

add tmp1, 30 //30

mov [tmp1], #8B75A8893E83C6048975A847897DA481FFA4337B027402EBD290909090909090C745A400000000C745A800085901C745#

add tmp1, 30 //60

mov [tmp1], #AC10347B02BB000660018B75A88B368B45A48B4B6CF7E18B4B3003C833C08A43268B7C83408BC1FFD78BF833C08A4327#

add tmp1, 30 //90

mov [tmp1], #8B5483408BC1FFD28945F433C08A43258B5483408BC1FFD284C00F841D000000FEC80F8478000000FEC80F84B0000000#

add tmp1, 30 //C0

mov [tmp1], #FEC80F8478010000E9130700008B4EFCC606E92BCE83E905894E018B436803F8837B74017503037B70897DF0837DF0FF#

add tmp1, 30 //F0

mov [tmp1], #75110345F4034310837B74017503034370EB0B8B45F0E8D9060000034310C646FBE88D4EFB2BC183E8058946FC8B45A0#

add tmp1, 30 //120

mov [tmp1], #89088345A004E9950600009090909090C606E98B436803F8837B74017503037B70897DF0837DF0FF75080345F4034310#

add tmp1, 30 //150

mov [tmp1], #EB0E8B43180345F02BC683E805894601E95B0600009090909090909090909090E8230000008B459CC700020000008345#

add tmp1, 30 //180

mov [tmp1], #9C048BD6E81F000000E82A000000E92D06000090909090908B55AC2BD683EA05C606E9895601C390522B53188B459C89#

add tmp1, 30 //1B0

mov [tmp1], #1083459C045AC39033C08A43288B5483408BC1FFD2837B7401750733D28A537032C2E8B905000086E0050F8000008B4D#

add tmp1, 30 //1E0

mov [tmp1], #AC6689018B43180345F4034368837B740175030343708BD0E8ABFFFFFF2BD183EA0689510283C106037B18037B68837B#

add tmp1, 30 //210

mov [tmp1], #74017503037B70C601E98BD7E887FFFFFF2BD183EA0589510183C1053E894DACC3909090909090909090909090909090#

add tmp1, 30 //240

mov [tmp1], #E853FFFFFF8B459CC700030000008345#

add tmp1, 10 //250

mov [tmp1], #9C048BD6E84FFFFFFF909090909033C08945B08945B48945B88945BC8A432B8B5483408BC1FFD2837B740175032B4370#

add tmp1, 30 //280

mov [tmp1], #8945B033C08A43298B5483408BC1FFD28BD080EA080F92C280FA01750B3E8945B0C745B40100000033C08A432C8B548340#

add tmp1, 31 //2B1

mov [tmp1], #8BC1FFD2837B740175032B43708945B833C08A432A8B5483408BC1FFD28BD080EA080F92C280FA01750B3E8945B8C745BC0100000033C08A432D8B5483408BC1#

add tmp1, 40 //2F1

mov [tmp1], #FFD285C00F8425000000480F848E010000480F8427020000480F8440030000480F84E9030000E9C404000090909090#

add tmp1, 2F //320

mov [tmp1], #51538B4DAC837DB4010F85B8000000837DBC017547B83900000033D23E8A55B8C0E2033E0255B086F203C2807DB00474#

add tmp1, 30 //350

mov [tmp1], #0E807DB005741166890183C102EB18668901C6410224EB0C0500400000668901C641020083C103E9CA0000003E8B55B8#

add tmp1, 30 //380

mov [tmp1], #81FA800000007307B883380000EB05B88138000033D23E8A55B086F203C2807DB004740E807DB005741466890183C102#

add tmp1, 30 //3B0

mov [tmp1], #EB1B668901C641022483C103EB0F0500400000668901C641020083C1033E8B55B881FA800000007307881183C101EB66#

add tmp1, 30 //3E0

mov [tmp1], #891183C104EB5F837DBC017521B83905000033D23E8A55B8C0E20386F203C26689013E8B55B089510283C106EB383E8B#

add tmp1, 30 //410

mov [tmp1], #55B881FA800000007317B8833D00006689013E8B45B089410288510683C107EB15B8813D00006689013E8B45B0894102#

add tmp1, 30 //440

mov [tmp1], #89510683C10A894DACE9320300009090#

add tmp1, 50 //490

mov [tmp1], #51538B4DAC837DB4010F854103000083#

add tmp1, 10 //4A0

mov [tmp1], #7DBC017544B83B00000033D23E8A55B0C0E2033E0255B886F203C2807DB804740E807DB805741166890183C102EB3966#

add tmp1, 30 //4D0

mov [tmp1], #8901C6410224EB0C0500400000668901C641020083C103EB1FB83B05000033D23E8A55B0C0E20386F203C26689013E8B#

add tmp1, 30 //500

mov [tmp1], #55B889510283C106894DACE970020000#

add tmp1, 30 //530

mov [tmp1], #51538B4DAC837DB4010F859F000000837DBC017551807DB005742AB83800000033D23E8A55B8C0E2033E0255B086F203#

add tmp1, 30 //560

mov [tmp1], #C266890183C102807DB0047524C6012483C101EB1CB83845000033D23E8A55B8C0E20386F203C2668901C641020083C1#

add tmp1, 30 //590

mov [tmp1], #03E983000000807DB0047423807DB005742BB88038000033D23E8A55B086F203C26689018B55B888510283C103EB5AC7#

add tmp1, 30 //5C0

mov [tmp1], #01833C24008A55B8885103EB0CC701837D00008A55B888510383C104EB3B837DBC017521B83805000033D23E8A55B8C0#

add tmp1, 30 //5F0

mov [tmp1], #E20386F203C26689013E8B55B089510283C106EB1466C701803D8B55B08951028A45B888410683C107894DACE95F0100#

add tmp1, 30 //620

mov [tmp1], #009000#

add tmp1, 30 //650

mov [tmp1], #51538B4DAC837DB4010F8581010000837DBC017544B83A00000033D23E8A55B0C0E2033E0255B886F203C2807DB80474#

add tmp1, 30 //680

mov [tmp1], #0E807DB805741166890183C102EB39668901C6410224EB0C0500400000668901C641020083C103EB1FB83A05000033D2#

add tmp1, 30 //6B0

mov [tmp1], #3E8A55B0C0E20386F203C26689013E8B55B889510283C106894DACE9B0000000#

add tmp1, 50 //700

mov [tmp1], #5153837DB4010F85D4000000837DBC017524B83BC0000033D23E8A55B0C0E2033E0255B886F203C28B4DAC66890183C1#

add tmp1, 30 //730

mov [tmp1], #02894DACEB22B881F8000033D23E8A55B086F203C28B4DAC6689013E8B55B889510283C106894DACEB26000000000000#

add tmp1, 50 //780

mov [tmp1], #5B59E831FAFFFFEB37909090909090903C06740E3C07740E3C0A740E3C0B740EEB0EB00AEB0AB00BEB06B006EB02B007C3909090909090909090909090909090#

add tmp1, 40 //7C0

mov [tmp1], #FF45A48345A8048B45A88B0083F8000F8590F8FFFF83C460619090909090909090909090BFD7397A01B9FFFFFFFFF2AF81FF4F3A7A0177E88B47F8C390909090#

//chk version

FillSCP1:

find dllimgbase, #8B5482408BC6FFD22C#

mov tmp1, $RESULT

cmp tmp1, 0

je FillSCP2

add tmp1, 9

mov tmp2, [tmp1], 1

cmp tmp2, 2

je FillSCP3

cmp tmp2, 1

jne patcherr

mov tmp1, dllimgbase

add tmp1, AC //AC

mov [tmp1], #9001#

add tmp1, 8 //B4

mov [tmp1], #15#

add tmp1, 8 //BC

mov [tmp1], #70#

add tmp1, 8 //C4

mov [tmp1], #A800#

add tmp1, 233 //2F7

mov [tmp1], #0504#

add tmp1, 7 //2FE

mov [tmp1], #1E00#

add tmp1, 7 //305

mov [tmp1], #8701#

add tmp1, 7 //30C

mov [tmp1], #2002#

add tmp1, 7 //313

mov [tmp1], #3903#

jmp FillSCP3

//resolve vm code in aspr dll

FillSCP2:

//alloc 10000

//mov VMcodeloc, $RESULT

//log VMcodeloc

//lm VMcodeloc, 4000, "d:\Asprvm8s.bin"

FillSCP3:

ret