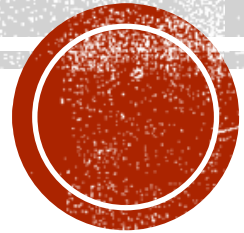
# WINAPI HOOKİNG WİTH C++



# OVERVIEW

- WHOAMI
- QUICK INTRODUCTION TO WINAPIs
- USE-CASE EXAMPLES
- HOW TO DO IN THEORY
- HOW TO DO IN PRACTICE
- ALTERNATIVES
- QUESTIONS

# whoami

## Yusuf Kocadaş

- Final year Computer Engineering Student
- Reverse engineering enthusiast
- Formerly intern at TUBITAK BILGEM
- Facebook & ASUS HOF

# WINAPI

- **Definition:** The Windows API, informally WinAPI, is Microsoft's core set of application programming interfaces (APIs) available in the Microsoft Windows operating systems.(Wikipedia)
- Well-documented by Microsoft.
- __stdcall in x86
- __fastcall in x64

# SOME USE-CASES

- Anti-Debugging & Anti-Anti-Debugging
  - CheckRemoteDebuggerPresent, NtQueryInformationProcess, GetTickCount, SeDebugPrivilege, IsDebuggerPresent…
- Malware Analysis
  - GetAsyncKeyState, InternetOpen, CreateRemoteThread, CreateProcess, CreateMutex, CreateFileA, RegCreateKeyA…
- Cheating & Anti-Cheating in Games
  - CreateToolhelp32Snapshot, Process32First,Process32Next, Module32First, Module32Next, CreateServiceA, GetModuleFileName…
- Packing & Unpacking
  - VirtualAlloc, VirtualProtect, LoadLibrary, GetProcAddress…

# TRAMPOLINE HOOK IN THEORY

# connect function

12/05/2018 • 8 minutes to read

The **connect** function establishes a connection to a specified socket.

## Syntax

C++

```
int WSAAPI connect(
  SOCKET         s,
  const sockaddr *name,
  int            namelen
);
```

## Parameters

`s`

A descriptor identifying an unconnected socket.

`name`

A pointer to the sockaddr structure to which the connection should be established.

`namelen`

The length, in bytes, of the sockaddr structure pointed to by the *name* parameter.

## Return value

If no error occurs, **connect** returns zero. Otherwise, it returns SOCKET_ERROR, and a specific error code can be retrieved by calling WSAGetLastError.

CHECK THE DOCUMENTATION IF IT EXISTS. KNOWING THE FUNCTION SIGNATURE WOULD HELP A LOT.

| Address | Bytes | Opcode | | Comment |
|---|---|---|---|---|
| **WS2_32.connect** | | | | |
| WS2_32.connect | 8B FF | mov | edi,edi | |
| WS2_32.connect+2 | 55 | push | ebp | |
| WS2_32.connect+3 | 8B EC | mov | ebp,esp | |
| WS2_32.connect+5 | 83 EC 18 | sub | esp,18 | 24 |
| WS2_32.connect+8 | 8D 45 F4 | lea | eax,[ebp-0C] | |
| WS2_32.connect+B | 56 | push | esi | |
| WS2_32.connect+C | 8B 35 00407675 | mov | esi,[WS2_32.DLL+44000] | (7572E730) |
| WS2_32.connect+12 | 57 | push | edi | |
| WS2_32.connect+13 | 50 | push | eax | |
| WS2_32.connect+14 | 8D 45 F8 | lea | eax,[ebp-08] | |
| WS2_32.connect+17 | 50 | push | eax | |

Determine how many instruction will be overridden by our detouring instruction(jmp), in this case, 3 instruction will be overriden but the first one has no point, so we can consider there are just 2 instructions.

```
__declspec(naked) int realConnect(SOCKET s, const sockaddr* name,int namelen)
{
    __asm
    {
        push ebp
        mov ebp,esp
        mov edx, connectAdr
        add edx, 5
        jmp edx
    }
}
```

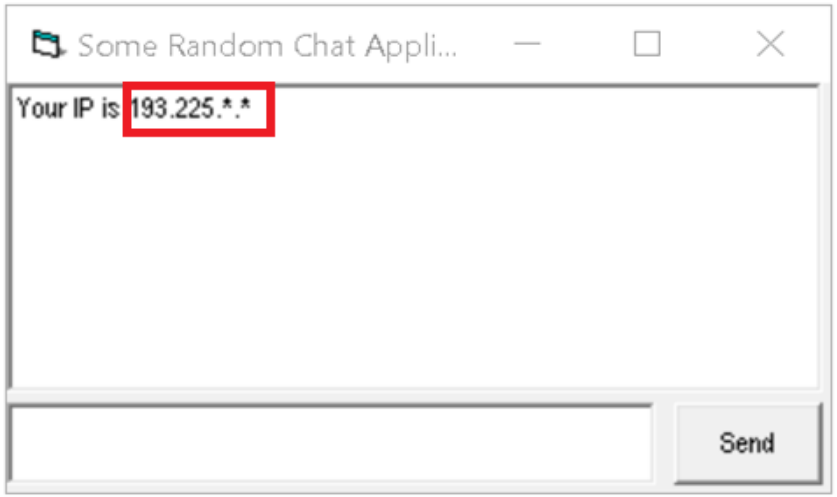
# DON'T FALL INTO YOUR OWN TRAP

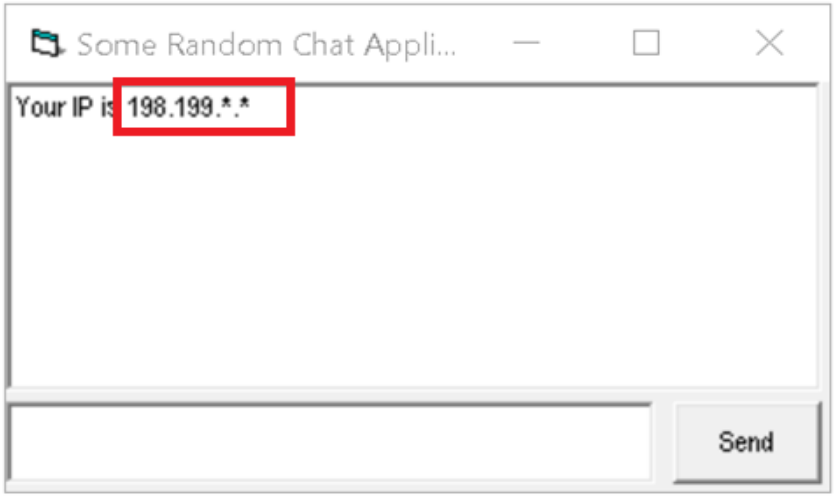
```
__declspec(naked) void connectionHandlerWrapper(SOCKET s, const sockaddr* name, int namelen)
{
    __asm
    {
        push ebp
        mov ebp,esp
        pushad
        push namelen
        push name
        push s
        mov edx, [connectionHandler]
        call edx
        popad
        pop ebp
        ret 0xC
    }
}
```

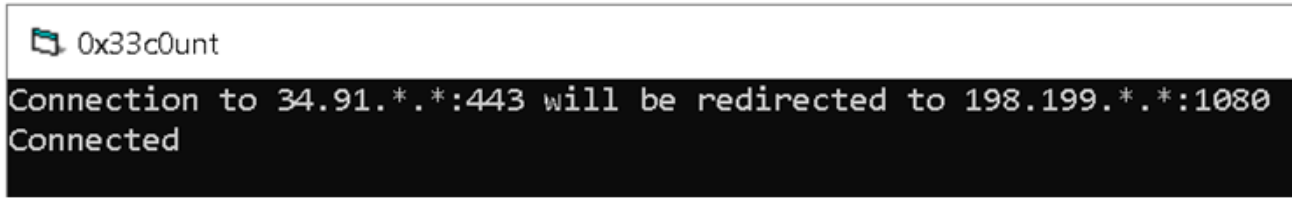
# MAKE YOUR PLAYGROUND COMFORTABLE

```
int WINAPI connectionHandler(SOCKET s, const sockaddr* name, int namelen)
{

    SHORT PORT;
    BYTE IP[4];
    u_long iMode = 0,iResult;

    memcpy(IP, name->sa_data + 2, 4); // read the ip

    memcpy((void*)(name->sa_data + 2), IPArr, 4); // change the ip

    *(BYTE*)&PORT = *(BYTE*)(name->sa_data+1); // read...
    *(BYTE*)((DWORD)&PORT+1) = *(BYTE*)name->sa_data;// the port

    *(BYTE*)(name->sa_data + 1) = *(BYTE*)&port; // change
    *(BYTE*)(name->sa_data) = *(BYTE*)((DWORD)&port + 1); // the port
```

# ACCESS, MODIFY PARAMETERS, CHANGE THE IMPLEMENTATION




- See the effect, in this case we implemented a process-based proxy redirection and tested on a dumb chat application

# TRAMPOLINE HOOK IN PRACTICE

https://github.com/0x33c0unt/process-based-proxy-redirection

# ALTERNATIVES FOR TRAMPOLINE

- Hardware Breakpoints (Limited)
  - Debug Registers, DR0-3
- IAT/EAT Hooking (Not everybody relies on IAT)
  - Import Address Table
- Trigerring an exception (Slow)
  - AddVectoredExceptionHandler, SetUnhandledExceptionFilter

# QUESTIONS