Lab 7: Remote Memory Exploits

Due on April 16th (T) 2019

1 Overview

In this lab, your goal is to exploit one network service that we have set up for you. The network service includes a specific exploitable vulnerability that you will need to identify. Once you have identified the vulnerability, you will need to construct a program that delivers malicious payload to the service, triggering and exploiting the vulnerability.

You can use any programming language you want for this lab.

2 The Server

There is one service running on our server that you will need to attack. The service and its port is:

- Encrypted Echo Server (port 9907)

The archive contains the source code and build scripts for the service. The exact binary that is running on our server is also provided in the archive. The service is compiled as 32-bit executables without stack protection (-fno-stack-protector), and it is statically linked with libraries. The service is executing with ASLR disabled.

3 Attack Approach

3.1 Understanding the Service

The first thing to do is to understand how the server works. What does it do? How do you communicate with it? Understand the data flow in the server code.

3.2 Server Stack Frame

The second thing is to draw the server's stack frame for the *OnClient* function on a paper. Where are *cookie*, *buf*, *hdr* variables located in the stack frame? And the most important one, where is *saved eip* located in the stack frame? Your goal is to overwrite the *saved eip* with the address of your malicious payload so that the server jumps to your malicious payload and executes it.

3.3 Stack Buffer Overflow Bug Hunting

You may notice that the data you send will be stored into *buf*, and this is where you can overflow the server's stack. To start, you should look at how the server receives your data and stores into *buf*, and identify where the server fails to check the length of your input data so that the length of your input data is greater than the size of *buf*, which is a buffer-overflow bug.

3.4 Attack

Once you have identified the bug, you need to deliver a malicious payload to trigger vulnerability so that the server performs some actions you have specified in the payload. Your ultimate goal is to steal the secret flag stored in the server. The idea is to use reverse-shell: actions specified in the malicious payload will have the server sends a remote shell to you and therefore you can controll a remote shell (just like ssh shell). With the remote shell, you can open the secret file and read the secret.

Here is a great tutorial on the reverse-shell: https://www.hackingtutorials.org/networking/hacking-netcat-part-2-bind-reverse-shells/. You can setup the Netcat reverse-shell listener on our server or COE gateway. Before the attack, make sure your reverse shell is working by executing your malicious payload. Here is a reverse-shell payload you can use: http://shell-storm.org/shellcode/files/shellcode-883.php. Note that you need to change the IP address to where your Netcat reverse-shell listener is.

3.5 Secret Flag

The service is running in a directory that includes a secret flag. After you have exploited the service you must collect the secret flag.

4 Accessibility

For security reasons, the service in our server only accepts TCP connections from machines in the Northeastern network. Thus, you must be either physically present on campus, or SSH into a Northeastern machine and run attack on that machine, or get tunneled/VPNed, in order to access the service.

5 What You Need to Turn In

To receive credit on this lab, you will need to turn in all your code, and a brief report with following items:

1. Report the secret.
2. Why is canary countermeasure in the server code not effective against stack buffer overflow attack? and How would you make it effective?
3. Suggest and discuss two countermeasures preventing stack buffer overflow attacks.