Homework 4 - Metasploit Modules

Introduction

You will develop a full-chain for access using the Metasploit framework to exploit a given vulnerable service, BockServe 2.0a. First, you will write a Metasploit auxiliary module to scan a given target to check for the presence of this vulnerable service. Second, you will write a Metasploit exploitation module to "exploit" the service and get a meterpreter shell. Lastly, you will write a Metasploit post module to do explorative tasks on the compromised machine over your meterpreter shell.

Requirements

- Successful Metasploit auxiliary, exploitation, and post modules that identify, exploit, and use the vulnerable service to accomplish tasks on the target.
- You are prohibited from using *any* outside libraries that do not come installed with Ruby or Python.
- Metasploit is (almost) entirely in Ruby, so the code for this project must be written in either Ruby or Python.

Vulnerable Service

The given vulnerable service is a simple Python server running on a target machine that acts as a remote Python shell. It runs on port 3285. It allows you to send Python commands (after some initial interaction) and it returns the result after executing them, much like a local (but stateless) top-level Python interpreter would. Since the service runs untrusted code, it can be leveraged to gain execution on the machine.

Auxiliary -

The auxiliary module is designed to check for the presence of a vulnerability. Your module should take an IP address and check for the presence of the vulnerable service. Note that there is another version of this service (BockServe 3.0 and BockServe 2.0b) that does not execute code but have the same banner, so checking for an open port is not enough to confirm the service is vulnerable. Minor interactions with the service is sufficient to confirm suspected vulnerability. RPORT and RHOST are the only options your auxiliary module needs to support.
Your auxiliary module should be named "auxiliary_<directoryID>.rb". For example, my module is named "auxiliary_kbock.rb"

Exploit -

The exploitation module is designed to take advantage of a vulnerability. Your module should take a target IP address and obtain a meterpreter shell on the target. RPORT and RHOST are the only options your exploit module needs to support.
Your auxiliary module should be named “exploit_<directoryID>.rb”. For example, my module is named “exploit_kbock.rb”

Post -

The post module is designed to run over a meterpreter shell and elicit information about the target. You can assume the target server will be running on Linux. Your post module should obtain the following information, and print it to the user:

- List of users present on the system
- Kernel version
- Operating System major version
- Retrieve password hashes

Your post module should be named “post_<directoryID>.rb”. For example, my module is named “post_kbock.rb”

Testing

You will be supplied with an test environment on aceslab. This environment is functionally identical to what we will use when grading your implant. You are not required to use this environment, but it is there for your benefit. The modules will be run automatically through an orchestration script to test that they can accurately determine whether or not a machine is vulnerable to the exploit, achieve a meterpreter shell, and retrieve the required information from the target.

Submission

Please submit your code to the ELMS submit link. Your code should be in 3 .rb/.py files.

Helpful Resources

<u>Commands</u>
reload - reloads current Metasploit modules
reload_all - reloads all Metasploit modules from all sources

<u>Example Modules</u>
https://github.com/rapid7/metasploit-framework/blob/master/modules/