Homework 6 - Exploitation

Introduction

You have been tasked with developing a simple exploit for a target communications platform. There exist multiple vulnerabilities in the program; fortunately, we have recovered the source code. You will write an exploit for operators to use to attack this service in the field that presents the operator with a shell on the target machine through the service.

The service will be running on a target default Ubuntu machine, with its firewall enabled such that every port is blocked for both outgoing and incoming traffic with the exception of the port the service is running on (default: 12345). This means you are **NOT** allowed to open up any new ports on the machine or even call out from the target machine itself.

Deliverables

- Single, self-contained Python script that presents the user with a shell they can interact with. As usual, you are not permitted to use any outside libraries that require you to install anything.
- Executable should take two command line parameters, the remote host IP address and remote port.
- Executable should create an interactive shell, where we can type commands and see the output. It is up to you how you want this shell to look, but you should give us some output to know when we can begin typing commands.
- Report of how you developed your exploit and how it works, with enough detail for us to recreate your exploit. Your report should read like a tutorial for exploiting the server.

Example

Example 1

```
# python exploit_kbock.py 10.1.0.1 1337
Setting up for execution
Setting up shell:
$
```

<here, the user can interact with the shell>

```
$ whoami
student
$ echo “hi”
hi
$ pwd
/root
$ ^D
```

```
Disconnecting from shell
#
```

Example 2

```
# python exploit_ghughey.py 10.1.0.1 1337
Could not connect to server - server is not listening.
#
```

Testing

This is not an exercise in dealing with network latencies. You can, and should, simply test this by running the server on your local machine and running your exploit on the same machine. You should remember that latencies will vary by a very small amount on local interfaces. It should also be quite easy to disable the parts of the server you do not need to lower the amount of time it takes to develop the exploit - just make sure to test the whole thing at the end!

Notes

- This is not an exercise on how to create reverse or bind shells. You need to create a shell using an exploit and manually send commands and receive their output.
- You may assume that the server is being run as root, but you do not need to.
- There are probably other possible exploits in this server that you might not need for this exercise. If you manage to do any of them, feel free to write them up in your report!
- The prettier your shell, the happier George will be :)
- You can assume that commands run on your shell will have output.

Submission

Please submit your code and report in a .zip file to the ELMS submit link. Your code should be a single file, named exploit_uid.py. For example, Kevin's would be exploit_kbock.py.