Implementing basic networking utilities

Lab Manual

Objectives

- Create simple clients and servers
- Write python code to replace netcat
- Build a TCP proxy
- Create ssh server
- Apply ssh tunneling

Table of Contents

1.

Background Reading

Textbook:

Black Hat Python, chapter 2

Other resources:

https://www.sans.org/reading-room/whitepapers/malicious/basic-reverse-engineering-immunity-debugger-36982
https://sgros-students.blogspot.ca/2014/05/immunity-debugger-basics-part-1.html
http://tf.nist.gov/tf-cgi/servers.cgi
https://tools.ietf.org/html/rfc4330

Notes common to all lab and home assignment problems

For every lab and home assignment, all work should go into your personal repository, subdirectory named mXX, where XX stands for the module number. For each problem, carefully name the program as described. The programs are extracted from your repository by a Python script, and errors in the program name will result in the instructor never seeing your program, and your mark for it will be ZERO!

Anything to record and report in this lab is to be written in plain text file (Word document is not a plain text file). The file MUST be named mXXpXX.txt. Mis-naming this file, or not having it in the proper location will result in mark ZERO for anything to be recorded or reported.

There are always many ways how to solve a programming problem, and usually one or two ways which are fast, compact and elegant.

Make sure to push your work to the server often, and have pushed the working version of the program by the deadline specified. The script extracting your programs from your repository will be run at any time after the deadline.

Lab specific intro

In this lab, we will explore the basics of networking with Python. We will learn about network sockets, create clients and servers, and write code for other networking utilities in Python. The core Python module for network programming is the **`socket`** module, and we will explore how to create socket, connect to remote host, listen on a socket on a specific port, send and receive data packets.

Problem 1

Write simple TCP web client in Python, name the program **`m11tcpcli.py.`** The client will connect to website http://itss.biomea.com, and get page /cgi-bin/echo.cgi. Report the information received.

Problem 2

Write simple TCP server **`m11tcpsrv.py`** using the chapter “TCP Server” from the Black Hat Python textbook. Run the server, point your browser and record what the server received.

Problem 3

Write daytime server in Python, name it **`m11timed.py.`** Have it listen on port 2013 and use the UDP protocol The server will reply to any packet received with packet containing the UTC current date and time in the following format:

YYYY-MM-DD HH:MM:SS <crlf>

Use linux command **`script`** to save your interactive terminal session. Connect to your server using **`netcat`** in linux environment, and report the relevant part of your console log.

Problem 4

Write simple UDP client in Python, name it **`m11udpc.py.`** The client will get two parameters – the name or ip address of the server, and the port to send one UDP packet to. Run the client and request the time from the server implemented in problem 2, and report the time corrected for local time zone. Report console log.

Problem 5

Improve the daytime server from problem 2, call it **`m11synctimed.py.`** Have the server get accurate time information by sending UDP packet to time.nist.gov on port 37, using the SNTP protocol described in RFC4330. Connect to the server using the udp client written in problem 3. Report console log.

Problem 6

Study the chapter Replacing Netcat, page 13 of the Black Hat Python textbook. Enter the code for bhnet.py, start session logging with linux **`script`** command and kick the tires.

Problem 7

Build simple TCP proxy using the code from the textbook as an example, name it **`m11proxy.py`**. Run the proxy on server designated by your instructor, and use it to manually get a web page from problem 1. Report the information received.

Problem 8

Install python-paramiko package, and write code for simple SSH server in Python, naming the file **`m11sshd.py`**. Configure the server to authenticate when receiving username “hacker” and password “secret”. Generate pair of keys named **`m11_rsa.key`** using **`ssh-keygen`**. Test the server by logging into it remotely, and report the session.

Problem 9

Create two scripts, **`m11revsshc.py`**, which will be a client running on a virtual machine (the "victim"), and **`m11sshd.py`**, which will be a server running on your laptop. Start the server, then remotely execute commands on the victim system. Log and report the session.

Problem 10

Using the code in SSH Tunelling chapter on page 30 of the Black Hat Python textbook, create tunnel from your laptop to the server designated by your instructor. Log the session. Login to the server and execute a simple command. Report the session.