Python Project

Scapy functions and encryption

Grade breakdown

Important: The project is individual and will be tested for plagiarism. This work is to be done alone and any instance of copying/collaborative work will be treated harshly.

A. Running 30%:

The program works according to the designed requirements while achieving all goals defined. The program has no "problems" according to the PyCharm problem tab.

B. Comments 30%:

The comments for the project will be in the files themselves. No additional comment/explanation files.

1. A comment at the top of the program with a short description of the program goals and how they are achieved, the general algorithm (step-by-step breakdown), any input/output demands, and any assumptions made.
2. Before every function header (def <func>():) a comment describing what the function does, assumptions, and the algorithm.
   Objective: This comment is supposed to allow the reader (who did not write the code) a short introduction and explanation of the code – for the reader to be able to understand the code.
3. Every variable will have a meaningful name and a short explanation about its usage. i,j,k are used as counters/indexes for loop and the like and don't need comments (for example: for i in range(0,10):)
4. No "magic numbers" – if you use a number multiple times in a program set a constant instead.
5. Use meaningful names for functions, constants, variables, etc.

C. Programming 40%:

Write a proper and modular code:

- Break into files not required but recommended (This will not affect grading)
- Break into functions
- No global variables as much as possible (Remember CONSTANTS don't change during the program run.)
- Use the tools we learned as much as possible and correctly
- Try to write the code as elegantly and simply as possible

Project Objectives

The project is focused on 3 main parts:

- TCP Three-way handshake
- Port scanner and banner grabber
- Data encryption and decryption

Part 1 - Three-way handshake

Create a Scapy function that receives an IP address, port number, and a text message as **arguments**. It then creates a full three-way TCP handshake.
The function must fulfill the following requirements:

a. The function returns 2 variables, one Boolean and one String.
b. If the function successfully connects to the port the Boolean variable returned will be True.
c. The function sends the text message variable to the target after a successful connection and saves the reply to a string variable – returning it at the end of the program as the String variable
d. If the connection fails, the function returns the Boolean variable as False the string will return as None

Note – you can assume the argument input is legal.

Part 2 – Port scanning and banner-grabbing

Use the function from part 1 to create a port scanner/banner grabber.

Port scanner

Create a function that receives an IP address and a list of port numbers as **arguments**. The function then attempts to connect to each port on the list and returns a list of ports that have been confirmed as open by the function.

Banner grabber

Create a banner-grabbing option. The function receives an IP address, and a confirmed open port number as **arguments** and returns the "banner" – I.E the reply message of the port sends back to you after a successful connection/sent message.

Combine

After creating both functions create a third function that allows you to use the "Port scanner" to find the open ports of an IP address and then test the banner of each one of the open ports.

a. The function returns the port and banner of each open port.
b. Ports that are open but return no message return "unknown" instead of the banner
c. This will all be formatted as one string:
   Example:
   IP address – 172.18.0.7
   Port 21 open 220---------- Welcome to Pure-FTPd [privsep] [TLS] ----------
   …

Part 3 – Encryption and Decryption

Use the previous functions and create an encrypted data transfer with the results to another computer.

Encryptor

Create a function that receives a string as an argument and encrypts it using a constant key of your choice. Write your choice of encrypting algorithm

a. The function receives a string
b. The function returns a string that has been encrypted

Decrypter

Create a function that receives an encrypted string as an argument and decrypts it using a constant key of your choice. It should reverse the encryptor

a. The function receives a string that has been encrypted
b. The function returns a string after decrypting it.

Create a function that uses the encryptor and the function from part 1 to connect to an open port and send an encrypted string.