**Demba Moussa Gaharo B2**

User manual for my Python program and Excel file.

1. How To use my Python program Python

The entire code reads a packet capture file (DumpFile.txt) and extracts information such as source and destination IP addresses, packet length, flags, sequences, timestamps, etc. It also uses counters to count the number of packets with different flag types, the number of requests and responses via the ICMP protocol, etc. This information is then stored in lists for future use. I will do it step by step.

Une image contenant texte

Description générée automatiquement

This code imports several modules (or libraries) to be used in the program.

The first line "import csv" imports the csv module, which is used for reading and writing CSV files.

The second line "import webbrowser" imports the webbrowser module, which allows the program to open a web browser or interact with one.

The third line "import matplotlib.pyplot as plt" imports the pyplot module from matplotlib library which is used for creating plots and visualizations.

The fourth line "import numpy as np" imports the numpy module, which provides support for large, multi-dimensional arrays and matrices of numerical data, as well as a large library of mathematical functions to operate on these arrays.



This code opens a file named "DumpFile.txt" in read-only mode ("r"). The variable "fichier" is used to store a reference to the opened file, so that other operations (such as reading data from the file) can be performed on the file.

Une image contenant texte

Description générée automatiquement

This code creates several empty lists to store information that will be extracted from the packet capture file.

* "ipsr" is a list to store the source IP addresses.
* "ipde" is a list to store the destination IP addresses.
* "longueur" is a list to store the packet length.
* "flag" is a list to store the flag types of the packets.
* "seq" is a list to store the sequence numbers of the packets.
* "heure" is a list to store the timestamps of the packets.

Une image contenant texte

Description générée automatiquement

This code declares several variables that are used to count different types of information in the captured packets. Each of the variables is a counter that is initialized to 0. The comments indicate the purpose of each variable.

* "flagcounterP" is a counter for the number of flags [S].
* "flagcounterS" is a counter for the number of flags [.]
* "flagcounter" is a counter for the number of flags.
* "framecounter" is a counter for the number of frames exchanged on the network.
* "requestcounter" is a counter for the number of requests.
* "replycounter" is a counter for the number of replies.
* "seqcounter" is a counter for the number of sequences.
* "ackcounter" is a counter for the number of acknowledgments.
* "wincounter" is a counter for the number of win.

Une image contenant texte, plaque, argent

Description générée automatiquement

Une image contenant texte, plaque

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

These codes are a loop that iterates through each line of the "DumpFile.txt" file that was opened in read-only mode previously. It uses the "split()" method to separate each line using a space as a delimiter. It checks if the lines contain useful information by checking if certain keywords (such as "IP", "seq", "win", "ack" etc.) are present in the lines. Then, it fills the different lists declared previously with the information extracted from the lines. It also uses the counters declared previously to count the number of packets with different flag types, the number of requests and responses via the ICMP protocol, etc.

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

These codes use variables that are counters (flagcounter, flagcounterP, flagcounterS, replycounter, requestcounter, etc.) to create graphs using the Python Matplotlib library. It starts by defining a global variable called "globalflagcounter" which is equal to the sum of flagcounter, flagcounterP, and flagcounterS. It also defines variables P, S, and A which are each equal to their respective counter divided by globalflagcounter.

Then, it defines a global variable called "globalreqrepcounter" which is equal to the sum of replycounter and requestcounter, and also defines variables req and rep which are each equal to their respective counter divided by globalreqrepcounter.

Then it converts all counters into lists to view them in a CSV file. Finally, it creates circular graphs for flags and request and replies using the Matplotlib library and saves the graphs as "graphe1.png" and "graphe2.png" respectively.

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

This code represents a webpage that contains graph 1 and 2 and the statistics of flags [S], [P] and [ACK].

Une image contenant texte

Description générée automatiquement

This code opens two CSV files, 'moussa.csv' and 'demba.csv', in write mode. It uses the CSV library to write data into these files.

In the first block of code, it writes a header row with the columns 'Heure', 'IP source', 'IP destination', 'Flag', 'Seq', 'Length' then writes the data using the writerows function and the zip function to combine data from different variables (heure, ipsr, ipde, flag, seq, longueur). It then closes the 'moussa.csv' file using the close() method.

In the second block of code, it writes a header row with the columns 'Flag[P] (PUSH)', 'Flag[S] (SYN)', 'Flag[.] (ACK)', 'Nombre total de trames',"nombre de request","nombre de reply","nombre de sequence","nombre de acknowledg","nombre de window" then writes the data using the writerows function and the zip function to combine data from different variables (flagcounterP, flagcounterS, flagcounter, framecounter, requestcounter, replycounter, seqcounter.

1. Data analysis collected on Excel.

Une image contenant table

Description générée automatiquement

The analysis of these data allowed me to know that it is a DDoS attack because not only is there a recurrence of connection requests with the same source address and at the same time, but also we are supposed to be on a local network where there is no need to connect remotely, but we observe a remote connection request with SSH, which proves that it is an intruder.

A DDoS attack occurs when dozens, or even thousands of computers (referred to as "zombies" or "bots") are used to simultaneously send requests to the same server or network. These requests can be of different types, such as network connections, HTTP requests, or data packets. The goal of this attack is to overload the server or network resources, making the service in question inaccessible to legitimate users.

There are several techniques used to launch a DDoS attack, including the use of botnets (networks of computers infected with malware), request amplification (where a small request is amplified to become a larger one), or the use of vulnerabilities in network protocols. DDoS attacks can also be targeted at specific parts of an online service, such as DNS servers or database servers.

It's worth noting that DDoS attacks can be very difficult to block, as they typically come from many different computers and can be hard to identify. Companies and organizations can use DDoS protection services to minimize the risks and impacts of a DDoS attack.

**To process any CSV file with this code, you would simply need to provide the name of the file**.

Une image contenant texte

Description générée automatiquement

In case of need, you can visit my Github account, you will find the Python code, the HTML page, the graphs, the CSV files, and this same notice and do not hesitate to contact me in case of need.

https://github.com/dmg227/SAE15-Orale