Heartbeats Design Principles

Gary Khodayari 27th, Feb 2022

Github Link

You will need two separate machines, a server and client to run this application combo. Both client and server programs need to be run as root on both machines. I will setup an Ubuntu server on one of my proxmox nodes as a server and use my laptop with Ubuntu WSL as the client.

Client Side:

Client will only handle one session at a time. I will use scapy to both send and receive messages.

Client.py program runs as root. Client will initiated the TCP handshake on randomized port with server which should be already listening on a designated port (11414). The client will initiate a handshake and then send a custom user chosen payload. Due to iptables setting, Linux machine will probably send a RST automatically so the session is close. That is ok, we have to do the whole handshake every single time. When the program is running, there are two threads handling the functions of the program.

Thread 1: User Interface

This thread will handle the interactions with the user. I will create a menu that give multiple options to the user such as terminating the connection to the server or sending a customized message.

Thread 2: I NEED A DOCTOR

This thread will listen on a designated port (11415) for pulses from a server. These pulses are there to make sure the client is still able to contact the server. After receiving the pulse, client will send a message to the server to reset the timeout timer.

Server Side:

Server will be able to receive messages and monitor heartbeat timer for multiple sources. I will use scapy to send and receive packets on the server

Before client.py is ran, server.py should be already running and listening on a designated port (14144). Server.py accepts the 3 way TCP handshake and analyzes the packets. If the packets has TERMINATE in it the heartbeat timeout timer for the source IP will stop and rest and the theoretical session is closed. Any other payload is outputted to the terminal with some useful information. The first time a handshake is initiated with the server, it will store the IP address of the sender to a dictionary. There is a second dictionary that keeps a record time passed the last time server heard from the client like a timer. If the timer passes 60 seconds for any IPs the server will send the PULSE TCP message to the client. If the client responds, the timer resets. There will be two threads that handle these tasks.

Thread 1: A Very Good Listener

This is always listening for packets and analyzing them as soon as they are received. It will also handle adding IPs to the dictionaries and resetting the timer

Thread 2: A Caring Friend

This thread will keep the timer value increasing (unless its reset by thread 1) for all the clients. If the timer goes above 60 seconds, it will send a pulse message the IP.

I will use Berkley's packet filtering scheme to filter the proper flag to print
https://www.ibm.com/docs/en/qsip/7.3.2?topic=queries-berkeley-packet-filters

NOTES

The reason why I chose in program inputs rather than system arguments is, this makes this program even cross plafrom as Windows and BSD handel them diffrently and also this allows me further acces for making a better user interface