TSAM-P3

Brynjar Örn Grétarsson

October 2019

1 Introduction

In project 3, students are set the task of creating a small network of independent computers, a botnet. The bots on the network should be able to relay messages to their nearest neighbors and hopefully not crash too often while doing that seemingly trivial task. The estimated time to complete this is 50-60 hours.

2 Specification description

The following list details my interpretation of the given specifications and how the server is intended to behave if some host tries to do something not in spec. This list is built on the original specifications which can be found in the project description.

2.1 Server specifications

LISTSERVERS,<FROM_GROUP_ID>

Is the command that is used to identify hosts that are connected 1-hop away. That is, hosts that are connected to a host that my host is connected to. This is used to identify the host that my server connects to when setting up the initial connection. If a host does not send the SERVERS response within a reasonable time frame it is disconnected.

SERVERS

Is the response to the LISTSERVERS message. It is exactly as detailed in the original spec. This message starts with the SERVERS identifier followed by the senders id, ip and port comma seperated. This block then ends with a semicolon. Then comes the list of all connected servers. Each block in the list is seperated by a semicolon and each item within a block is seperated by a comma. Each block should contain the servers id, ip and port in this order.

KEEPALIVE,<NO_MESSAGES>

Is used to determine whether a server is still up or if it has crashed in addition to how many messages are waiting to be fetched. This works by periodically sending out a message containing KEEPALIVE followed by how many messages you have for the recipient. If the server does not receive this heartbeat within a reasonable time limit, it can disconnect.

GET_MSG,<GROUP_ID>

Is used to fetch messages from a connected host for the group specified in the <GROUP_ID> field.

SEND_MSG<FROM_ID><TO_ID><MESSAGE>

Is used to deliver a message from anyone to anyone. My implementation only sends this message to a directly connected server that matches the description of the server in the <TO_ID> field. Otherwise it stores the message locally. This is done in order to not flood the network and duplicate messages. If everyone just broadcasted these messages, they would get duplicated and the network would probably crash from all the messages. It would at least get very slow because of all the traffic.

LEAVE,<SERVER_IP>,<PORT>

Is used to disconnect a server from the network. This message could be used to send to a connected server and ask him to disconnect a specific server in order for my server to connect to that server. This is how a server could crawl across the network to try to look for a specific server. This could be used to completely disconnect a server from the network if it does not have any protection from that. This command could also unwillingly and accidentally disconnect parts of the botnet that could then not connect again, effectively creating two smaller subnets.

STATUSREQ,<FROM_GROUP>

Is used to get the total number of messages for each server stored on a server that is one hop away. The <FROM_GROUP> is the sender of this message. I also submitted a spec clarification on piazza on this matter(id @390 - STATUS* command proposal).

STATUSRESP,<FROM>,<TO>,<(SERVER,N_MSG)>

Is the response to STATUSREQ where <FROM> is the sender of the message, <TO> is the recipient. Following that is a "tuple" containing a server id and how many messages are stored. This only gets sent if the number of messages is larger than 0. See the same piazza post for clarification(id @390 - STATUS* command proposal). When my implementation receives this command and it

contains my server id, it sends out a GET_MSG request to fetch the messages and store them.

2.2 Client specifications

The client expanded upon the server specifications in order to be able to test all the required features of the server. There is a non-critical bug in the client where it can end up getting behind the server by a single message. This happens because the client dumbly waits for input from the client. This stops the code and an incoming message is not displayed right away but instead when the user is done typing the input. This can be reproduced by sending nothing from the client which does not prompt the server to send a response and the client is now stuck a single message behind the server.

GETMSG,<GROUP_ID>

Is used to get the newest message addressed to group with the id in <GROUP_ID>

SENDMSG,<FROM_GROUP>,<TO_GROUP>,<MESSAGE>

Is used to send a specific message to any group. This is changed from the spec to be able to send a message by hand from any group instead of only your group. The message field was also added because a command that is only used to send messages needs to have a message if it is supposed to be anything else than a static message.

LISTSERVERS

Is used to get the list of servers connected to this server.

CONNECT <IP> <PORT>

Is not in the spec. It is used to connect to a server using a given ip and port.

BCMSG,<FROM>,<TO>,<MESSAGE>

Is not in the spec. It is used to bradcast a message to all connected servers. This message has the from and to fields set the same in all messages.

LEAVE,<IP>,<PORT>

Is not in the spec. It is used to tell the server to send the LEAVE message to a specific host using a provided ip and port.

STATUSREQ,<TO_ID>

Is not in the spec. It is used to send the STATUSREQ message to a host that is connected to my server. If the host is not found, it does not do anything.

error message

If the command sent to the client is not found, it sends a short error message as a response.

3 Sending and receiving messages

Sent messages

Messages I successfully sent to other groups can be seen in figure 1. The groups I sent messages to were groups : 96, 17, 39, 5, 47, 24, 29, 7, 25, 9 and 8. A total of 11 groups. The data in the image is obviously not the entire log since that would have taken way too much space. This log can be found in the logs folder and is called local_log.log. These are only the messages sent with the SEND_MSG message. There are many more messages that were sent using the LISTSERVERS command. These are below in the bonus section.

```
13-11:06:19 - SEND_MSG,P3_GROUP_72,P3_GROUP_96,Hi from group 72!
13-11:08:37 - SEND_MSG,P3_GROUP_72,P3_GROUP_17,Hi group 17 from group 72!
13-11:12:14 - SEND_MSG,P3_GROUP_72,P3_GROUP_39,Hi group 39 from group 72
13-11:14:49 - SEND_MSG,P3_GROUP_72,P3_GROUP_5,Hi from group 72!
13-12:03:36 - SEND_MSG,P3_GROUP_72,P3_GROUP_47,Hi group 47!
13-12:04:43 - SEND_MSG,P3_GROUP_72,P3_GROUP_24, Hi gorup 24!
13-12:05:47 - SEND_MSG,P3_GROUP_72,P3_GROUP_29,Hi group29! Good to see ya on the network!
13-12:15:55 - SEND_MSG,P3_GROUP_72,P3_GROUP_7,Hi group 7! welcome to the botnet
13-12:18:02 - SEND_MSG,P3_GROUP_72,P3_GROUP_25,HELLO 25!
13-12:34:43 - SEND_MSG,P3_GROUP_72,P3_Group_9,Hi group 9. I RECOMENND having the GROUP all caps!
13-12:43:31 - SEND_MSG,P3_GROUP_72,P3_GROUP_8,Sup wassup
```

Figure 1: Sent messages with a timestamp

Received messages

Figure 2 contains the the messages I received only from the SEND_MSG command. They were from groups 5, 113, 96, 14 and 42 in figure 2. Another 5 messages are located in the same log file (skel_log.log) and is from the SERVERS message seen in figure 3.

```
12-14:43:50 - SEND_MSG,P3_GROUP_5,P3_GROUP_72,Hey hey group 72. This is group 5.SEND_MSG,P3_GROUP_5,P3_GROUP_72,Hey hey group 72. This is group 5.
12-15:43:36 - SEND_MSG,P3_GROUP_5,P3_GROUP_72,Hey group 72. Can you send me any messages?
12-16:21:18 - SEND_MSG,ORACLE,ORACLE,ORACLE
12-20:49:10 - SEND_MSG,P3_GROUP_113,P3_GROUP_72,Hi
13-00:46:43 - SEND_MSG,P3_GROUP_96,P3_GROUP_72,Hallo
13-11:31:08 - SEND_MSG,P3_GROUP_113,P3_GROUP_72,hello
13-11:32:29 - SEND_MSG,P3_GROUP_113,P3_GROUP_72,Wrong format of SENDMSG, Assignemnt API says SEND_MSG for server commands.
13-11:39:04 - SEND_MSG,P3_GROUP_14,P3_GROUP_72,Testing testing
13-13:30:20 - SEND_MSG,P3_GROUP_14,P3_GROUP_72,Testing testing
13-14:11:02 - SEND_MSG,P3_GROUP_42,P3_GROUP_72,Hey there group 72;
13-14:12:13 - SEND_MSG,P3_GROUP_42,P3_GROUP_72, What is the ultimate question?;
```

Figure 2: Received messages with a timestamp (SEND_MSG)

```
13-19:37:08 - SERVERS,P3_GROUP_18,157.97.10.155,4058
13-18:11:08 - SERVERS,P3_GROUP_7,130.208.243.61,55482;P3_GROUP_72,130.208.243.61,4072;
13-18:12:30 - SERVERS,P3_GROUP_25,130.208.243.61,4025;P3_GROUP_72,130.208.243.61,4002;P3_GROUP_42,130.208.243.61,4042;P3_GROUP_21,130.208.243.61,4053;
13-18:29:24 - SERVERS,P3_GROUP_14,130.208.243.61,4050;
13-18:43:32 - SERVERS,P3_GROUP_39,130.208.243.61,4065;
```

Figure 3: Received messages with a timestamp (SERVERS)

3.1 Wireshark trace

The wireshark trace is both here in figure 4 (pretty pointless if you ask me but a screenshot was requested so I deliver that). The trace from wireshark is also available in the logs folder in a file called wireshark-client-server-comms.pcapng. I used the following filter to get this capture:
tcp.srcport $>$ 3999 and tcp.dstport $>$ 3999

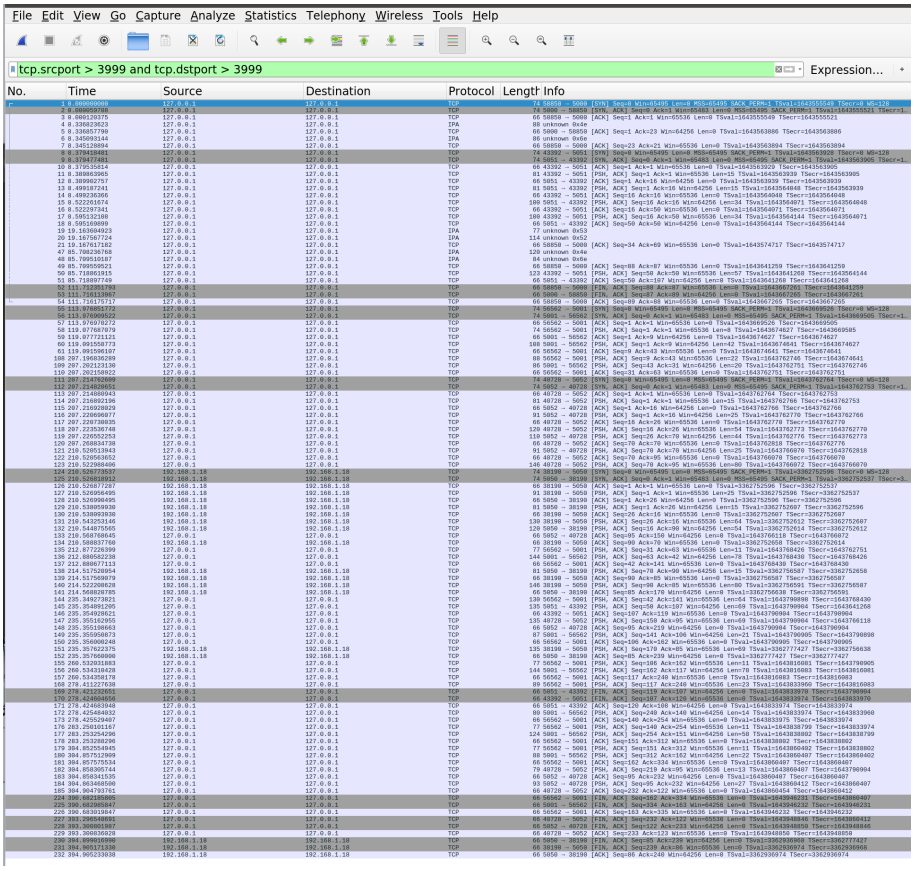

Figure 4: Tiny wireshark trace

MD5 Hash and the all mighty Oracle

The oracle also sent a MD5 hashed phrase. The log of that can be seen in Figure 5. The hash is: $9ed39e2ea931586b6a985a6942ef573e$ and is deciphered into: *state*. This can be found in the file located in the logs folder called local_log.log. In figures 6 and 7 you can see two plain text messages from the Oracle.

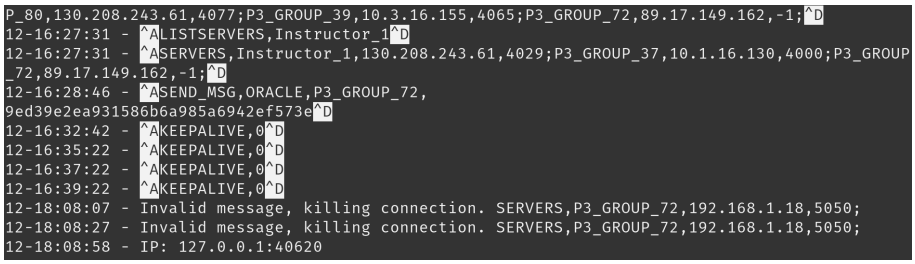

Figure 5: Hash from the Oracle

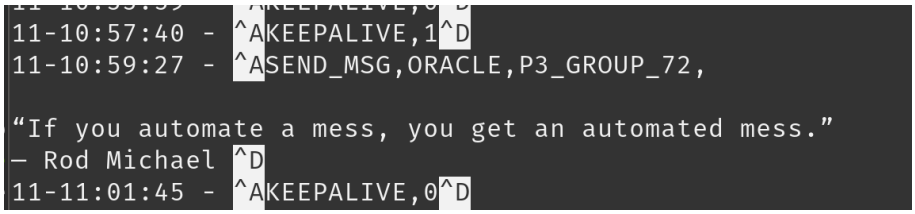

Figure 6: Message 1 from the Oracle

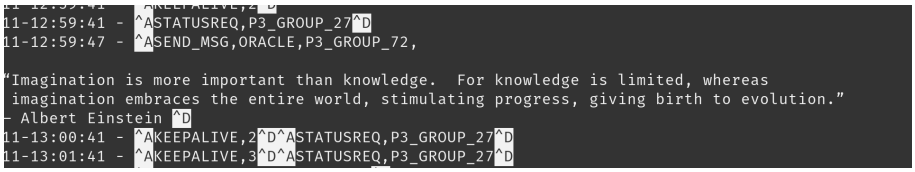

Figure 7: Message 2 from the Oracle

AK bonus

I could only connect to servers 21, 24, 25 and 69. They can be seen here below in figures 8, 9, 10 and 11

Groups 21, 25 and 69 can be found in the logfile skel_log.log and 21 can be found in local_log.log

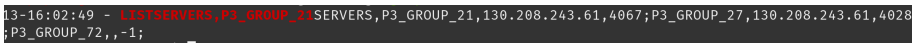

Figure 8: Akureyri 21

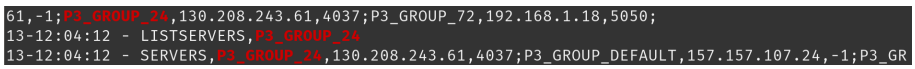

Figure 9: Akureyri 24

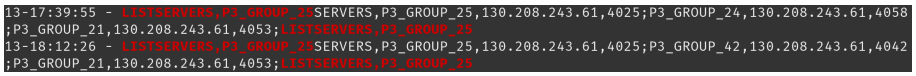

Figure 10: Akureyri 25

Figure 11: Akureyri 69

3.1.1 Messages from other groups (bonus)

In figure 12 are some messages from groups 27, 55, 80, 7, 21, 82, 9, 8, 39, 47 and 17 that count towards the bonus points.

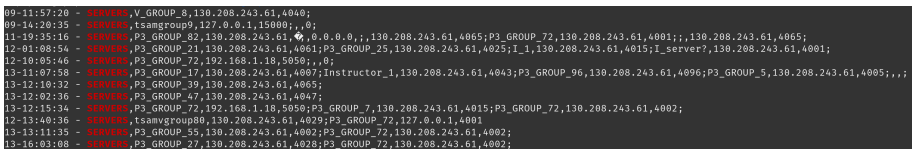

Figure 12: Bonus connections