



Department of Computer Science
Computer Networks

Due: Sunday 13th October 2019

Marking: 10% and up to 5 Bonus Marks

Your name:

TA Name:

Time Taken:

This is the third of three programming projects for Computer Networks, and is worth a total of 10% of your final mark. Additionally 4 bonus marks may be awarded for this project.

You may work on this individually or as part of a team of 2.

You may reuse your code from Project 1, or use the very simple, sample client-server project provided as a base.

Optional: Please include a rough estimate of how long it took you do the assignment so that we can calibrate the work being assigned for the course. (The estimated time is provided purely as a guideline.)

Important: Please read the instructions carefully.

In particular:

- The programming language is C or C++
- Although you can use boost for string handling, you may not use Boost.Asio (The BSD Socket API lies under all networking frameworks including Boost.Asio. We want to expose you to networking fundamentals, so that you understand where some of the issues with using networking frameworks come from.)
- **You must join a Group in Canvas, even if you are the only person in the group. The group ID of your canvas group, will also be the ID of your server.**
- You must provide clear instructions in a README file on how to compile and run your program, and what OS it was compiled on. If your program does not compile, please provide a detailed explanation of why you think that is, and why you decided not to use a source code management system with frequent commits for your project.
- Code should be well commented.
- Although you must develop your own code for this assignment, it is at the same time a class project, and relies on co-operation to succeed. It is perfectly ok to co-ordinate servers, discuss issues with inter-server messaging, etc.

Programming Assignment

In this project you will write a simple store and forward botnet message server, with accompanying Command and Control (C&C) client, following the specification below.

The goal is to link your server and client into a class wide botnet, and use them to fulfill the assignment todo list. In order to do that, you will need to give some thought to routing commands through the network, connecting to and leaving the botnet, storing messages for disconnected servers, message expiry, and handling commands for other groups. There is also a mysterious Oracle to be found.

In particular, remember that there is other information besides that purely in the messages - for example, knowing which link to your server the message arrived on.

Rules of Engagement:

0. Don't crash the (bot) network.
1. Try not to crash the campus network either.
2. Messages should not be longer than 5000 characters. You may truncate any messages you receive longer than that.
3. "Be strict in what you send, be tolerant in what you receive"
4. The executable for your server should be named as tsam<Group ID> eg. tsamvgroup1, the first parameter should be the port it is accepting connections on.
5. Your server should identify itself within the Botnet using your group ID, eg. V_Group_1 (Note, Instructor servers will also be running, prefixed by I_)
6. While part of the botnet, your server must maintain connections with at least 2 other servers and no more than 5 servers.
7. TCP ports 4000-4100 are available on skel.ru.is for external connections.
8. Your clients must connect to your server only, and issue any commands to other servers on the botnet via your server
9. You must use two token characters to indicate the start and end of each message between servers: ASCII character, 01 (SOH) for start, and 04(EOT) for end of message, with bitstuffing as necessary.
10. You are encouraged to reach out to other groups to be on the botnet with them at the same time, and co-operate in getting things working.

You may host the server on your machine, or on skel.ru.is, and run the clients from your local laptop. If you are not able to do this for any reason, please bring this to our attention *the first week of the assignment*.

Note: The `nohup` and `disown` commands can be used from a `bash/zsh` shell to run a command independently from the terminal, such as a server. If you do this be careful to clean up any stray processes during testing and debugging.

Server Specification

You do not have to run your server on `skel.ru.is`, as long as it can connect to at least one server that is running on `skel.ru.is`. If you are having problems connecting to `skel`, contact an Instructor in the first week.

Your server should listen on one TCP port for other servers to connect to it, and continue to listen to a separate port for your clients to connect.

The server port you are listening on should be included in the command line, so that servers running on `skel.ru.is` can be found. For example:

```
./tsamvgroup1 4044
```

where 4044 is the TCP port for server connections. You should allow a maximum of 5 servers to connect to you.

Server Communication

Messages between servers should be sent using the following format:

```
<SOH><Command>,< comma separated parameters ><EOT>
```

Server Commands

The server should support at least the following commands with other servers.

LISTSERVERS,<FROM_GROUP_ID> Reply with servers response (below)

SERVERS Provide a list of *directly connected* - i.e. *1-hop*, servers to this server.

The first IP address in the list should be the IP of the connected server.

Servers should be specified as GROUP_ID, the HOST IP, and PORT they will accept connections on, comma separated within the message, each server separated by ;

eg. SERVERS,V_GROUP_1,130.208.243.61,8888;V_GROUP_2,10.2.132.12,888;

KEEPALIVE,<No. of Messages> Periodic message to 1-hop connected servers, indicating still alive and the no. of messages waiting for the server at the other end. Do not send more than once/minute.

GET_MSG,<GROUP_ID> Get messages for the specified group. This may be for your own group, or another group.

SEND_MSG,<FROM_GROUP_ID>,<TO_GROUP_ID>,<Message content>
Send message to another group

LEAVE,SERVER_IP,PORT Disconnect from server at specified port.

STATUSREQ,FROM_GROUP Reply with STATUSRESP as below

STATUSRESP,FROM_GROUP,TO_GROUP,<server, msgs held>,...
Reply with comma separated list of servers and no. of messages you have for them

eg. STATUSRESP,V_GROUP_2,1,1,V_GROUP4,20,V_GROUP71,2

Client Commands

Communication between the Client and Server should use the protocol below. You may implement additional commands if you wish. The client should timestamp, (day, minute and second, please no nanoseconds) all messages sent and received, when they are printed out to the console.

GETMSG, GROUP_ID	Get a single message from the server for the GROUP_ID
SENDMSG, GROUP_ID	Send a message to the server for the GROUP_ID
LISTSERVERS	List servers your server is connected to

Assignment

1. Implement client and server as described above. All local commands to the server must be implemented by a separate client over the network. (4 points)
2. Provide a wireshark trace of communication between *your* client and server for all commands implemented (2 points)
3. Have been successfully connected to by an Instructor's server. (1 point)
4. Successfully receive messages from at least 10 other groups (Provide timestamped log) (1 point)
5. Successfully send messages to at least 10 other groups (Provide timestamped log) (1 point)
6. Code is submitted as a single tar file, with README and Makefile. (Do not include hg/git/etc. repositories.) (1 point)

Bonus Points

Note: The maximum grade (including bonus points) for the assignment is 15 points.

- Obtain bonus points for connectivity (maximum 3 points):
 - 0.5 points for messages from servers from students in Akureyri (The Akureyi group whose messages are reported will receive a matching 0.5 points, if they are reported at least 2 times by non-Akureyri groups.)
 - 0.5 points per 10 different groups you can show messages received from
- Provide a time stamped plaintext communication from the Oracle. (Time stamp should be when you received it. The oracle will respond to any messages addressed to it and delivered. It may move.)
- Decode a hashed message from the oracle (1 point)

Note: A number of sites exist that allow you to lookup known/cracked MD5 hashes. <https://hashkiller.co.uk/md5-decrypter.aspx> was used to verify the hashes in this assignment.