- To avoid any surprises, carefully check that both the evaluating and the evaluated students have reviewed the possible scripts used to facilitate the grading.

- If the evaluating student has not completed that particular project yet, it is mandatory for this student to read the entire subject prior to starting the defence.

- Use the flags available on this scale to signal an empty repository, non-functioning program, a norm error, cheating etc. In these cases, the grading is over and the final grade is 0 (or -42 in case of cheating). However, with the exception of cheating, you are encouraged to continue to discuss your work (even if you have not finished it) in order to identify any issues that may have caused this failure and avoid repeating the same mistake in the future.

- Remember that for the duration of the defence, no segfault, no other unexpected, premature, uncontrolled or unexpected termination of the program, else the final grade is 0. Use the appropriate flag.
You should never have to edit any file except the configuration file if it exists.
If you want to edit a file, take the time to explicit the reasons with the evaluated student and make sure both of you are okay with this.

- You must also verify the absence of memory leaks. Any memory allocated on the heap must be properly freed before the end of execution.
You are allowed to use any of the different tools available on the computer, such as leaks, valgrind, or e_fence. In case of memory leaks, tick the appropriate flag.

Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/11910/en.subject.pdf)

bircd.tar.gz (/uploads/document/document/1897/bircd.tar.gz)

Mandatory Part

Verifications

- there is a Makefile, the project compiles correctly, is written in C++, the binary file is called ircserv.
- ask and check how many select are present in the code. There must be only one.
- verify that the select is called every time before each accept, read/recv, write/send . After these calls, errno should not be used to trigger specific action (like read again after errno == EAGAIN for ex.).
- verify that each call to fnctl is as follows:
fcntl(fd, F_SETFL, O_NONBLOCK);
Any other use of fcntl is forbidden.
- if those constraints are not fullfilled, stop here the evaluation, and final mark is 0.

✓ Yes ✕ No

Networking

Check the following items :
- server launches and is listening on all network interfaces, on the given port from the command line, and on the port + 1 for TLS communication.
- using the 'nc' tool, you can connect to the server, send commands, and the server is answering.
- using a regular irc client, you can connect on the regular port, and on the TLS port as well.
- the server can handle multiple connections at the same time. At any moment, the server should not block and be able to answer to all demands. Try with multiple irc clients, nc tool, on regular and TLS ports, at the same time.
- send the needed commands to join a channel. Control that all messages from one client on that channel are sent to all other clients of the channel.

✓ Yes ✕ No

Networking Specials

Network communications can be disturbed by many strange situations.
- just like in the subject, using nc, try to send partial commands. Check that the server answers correctly. With a partial command sent, control that other connections are running fine.
- unexpectedly kill a client, and check that the server is still operationnal on other connections and for new incoming clients.
- unexpectedly kill a nc with just half of a command sent. Check again that the server is not in an odd state or blocked.
- stop a client (^-Z) connected on a channel, and then flood the channel using another client. The serveur should not hang. When the client is live again, all stored commands should be processed normally. Also control that there is no memory leak on the server during this operation.

✓ Yes ✕ No

Client Commands

- With both nc and a regular irc client, check that you can authenticate, set a nickname, a username, join a channel. This should be ok, you should have already used this above.
- Control that private messages and notices are fully functionnal, with several parameters (PRIVMSG and NOTICE).
- Check that a regular user does not have privileges to do operator actions, then test with an operator.
All the channel operation commands should be tested.

✓ Yes ✕ No

Server to Server

Now it's time to control that several of your servers can connect to each others and successfully exchange and relay messages.
- Launch a first server, then connect a second, then a third one.
- With a privileged client, check that all 3 servers answer correctly to 8 server commands : statistics, time, info, ...
- Connect many clients on the 3 servers, on several channels, and control that every announces and messages

on a channel propagate correctly through all servers.
- repeat the same actions with at least one public implementation of irc server (like ircd, available with brew).
- When a server is killed, check that other servers are properly warned.

✓ Yes ✕ No

Bonuses

Many bonuses are possible. Only add bonus points if everything else was 100% operationnal. It's up to you to choose how many points you give here, but you don't need to have all proposed bonus in the subject to get the maximum of points.
You can have : an IRC client, a configuration file, handle non mandatory parts of the RFC, a graphic interface, some file transfer system, a bot, an interaction with a real IRC network, ...

Rate it from 0 (failed) through 5 (excellent)

○

Ratings

Don't forget to check the flag corresponding to the defense

✔ Ok ★ Outstanding project

Empty work Invalid compilation Cheat Crash Incomplete group

💧 Leaks ⊘ Forbidden function

Conclusion

Leave a comment on this evaluation

□

Finish evaluation

General term of use of the site (https://signin.intra.42.fr/legal/terms/6)
Privacy policy (https://signin.intra.42.fr/legal/terms/5)
Legal notices (https://signin.intra.42.fr/legal/terms/3)
Declaration on the use of cookies (https://signin.intra.42.fr/legal/terms/2)
Terms of use for video surveillance (https://signin.intra.42.fr/legal/terms/1)
Rules of procedure (https://signin.intra.42.fr/legal/terms/4)