# SCALE FOR PROJECT FT_MALCOLM (/PROJECTS/FT_MALCOLM)

You should evaluate 1 student in this team

★

Git repository

```
git@vogsphere.42malaga.com:vogsphere/intra-uuid-b1821e43-2ca9-4979-9
```

## Introduction

- Remain polite, courteous, respectful and constructive throughout the evaluation process. The well-being of the community depends on it.

- Identify with the person (or the group) evaluated the eventual dysfunctions of the work. Take the time to discuss and debate the problems you have identified.

- You must consider that there might be some difference in how your peers might have understood the project's instructions and the scope of its functionalities. Always keep an open mind and grade him/her as honestly as possible. The pedagogy is valid only and only if peer-evaluation is conducted seriously.

## Guidelines

- Only grade the work that is in the student or group's GiT repository.

- Double-check that the GiT repository belongs to the student or the group. Ensure that the work is for the relevant project and also check that "git clone" is used in an empty folder.

- Check carefully that no malicious aliases was used to fool you and make you evaluate something other than the content of the official repository.

- 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/95200/en.subject.pdf)

# Mandatory Part

**User permissions check**

Run the program as a non-root user, then as root with no arguments.

- When running as non-root user, the program should exit by telling the user that root privileges are required to run. A help menu/usage is appreciated but not mandatory in this case.
- When running as root with no arguments, the program should exit and provide a help menu/usage.

Yes No

**Spoof between host and guest**

Run the program with the following arguments:
./ft_malcolm [VM ip] aa:bb:cc:dd:ee:ff [host IP] [host mac]
You can get these informations by running the ifconfig command (sudo/root privileges might be needed)
The program should be waiting for an ARP request before sending a reply and exiting.
After the program finishes, if you show the arp table on the host by running 'arp -a', the mac address of the source IP should
be equal to aa:bb:cc:dd:ee:ff.

Yes No

**Program behaviour**

Run the program with the same arguments as the previous question, but this time, open another terminal and listen to the ARP traffic on the network:
'tcpdump -vv -i [interface] arp'
Verify that the program sends an ARP reply only after a request has been made on the network (not necessarily by the target).

Yes No

**Error management**

Run the program with invalid IP addresses, invalid MAC addresses to test the robustness of the program.
The program should exit and output a minimum of information. Nothing too fancy, of course.

Yes No

# Bonus part

**Bonuses**

For the bonus part, any relevant bonus will grant one point on this scale.
IPv6 management, if present, will give two bonus points.

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

# Ratings

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

Ok

Empty work | Incomplete work | Invalid compilation | Cheat | Crash | Incomplete group

Concerning situation | Leaks | Forbidden function

# Conclusion

**Leave a comment on this evaluation**

**Finish evaluation**