

SCALE FOR PROJECT FAMINE (/PROJECTS/FAMINE)

Introduction

We ask you for the good progress of this evaluation to respect the following rules:

- Be courteous, polite, respectful and constructive in all situations during this exchange. The bond of trust between the community 42 and you depends on it.
- Highlight to the person (or group) noted the possible malfunctions of work done, and take the time to discuss and discuss.
- Accept that there may sometimes be differences of interpretation on the subject's requests or the scope of the features. Stay open-minded about the other's vision (is he or she right or wrong?), And write down the most honestly possible. The pedagogy of 42 only makes sense if the peer-evaluation is done seriously.


Guidelines

- You should only evaluate what is on the student / group's rendering GiT repository .
- Make sure that the GiT repository is the one corresponding to the student or group and the project.
- Meticulously verify that no malicious alias has been used to mislead you and have you evaluate anything other than the content of the official repository.
- Any meaningful script facilitating the evaluation provided by one of the two parties must be rigorously checked by the other party to avoid unpleasant surprises.

- If the correcting student has not yet done this project, it is mandatory for this student to read the subject in full before starting this defense.

- Use the flags available on this scale to signal an empty rendering, non-functional, a standard fault, a cheat case, etc. In this case, the evaluation is completed and the final grade is 0 (or -42 in the special case of cheating). However, except cheating, you are encouraged to continue to exchange around the work done (or not done precisely) to identify the problems that led to this situation and avoid them for the next rendering.

Attachments

 Subject (<https://cdn.intra.42.fr/pdf/pdf/5698/Famine.en.pdf>)

Source famine

preliminaries

preliminaries

Before starting the defense, please check the following:

- The project is written in C language or assembler,
- A Makefile containing the usual rules is present.
- A virtual machine under the operating system chosen by the answer key is available for evaluation.
- The root rights of this virtual machine are available.

Also check that the project does not use libraries which is detrimental to the educational value of the project, understand that the library (s) do not make coffee to it (s) -seule (s).

If only one of these points is invalid, the correction stops.

 Yes

 No.

Using Famine

Famine - Compilation and execution

Start a make then run the virus once. No error should be visible.

If a condition is not met, then the correction stops here.

 Yes

 No.

Attempt to use Famine

Create the two test and test2 temporary folders in the VM temporary folder.

Compile the program given as an example in the subject named sample.c in 64 bits as indicated in the topic.

The sample file works correctly without displaying errors when it is launched.

Copy the sample binary into the test folder created above.

Just run the virus now.

No error should be visible.

If a condition is not met, then the correction stops here.

 Yes

 No.

Virus infection

Start a simple 'strings' command on your sample located in the previously created test folder.

The sample file works correctly without displaying an error at launch.

If the login of one of the members of the group is not clearly visible, the correction stops here.

Then launch again the virus and watch via the command 'strings' if your sample does not contain a second time the login previously found.

If it is, then the correction stops here.

The sample file works correctly without displaying an error at launch.

If this condition is not respected, then the correction stops here.

 Yes

 No.

Infec-ception virus

Now copy a 64-bit binary of your choice into the test2 folder located in the operating system's temporary folder.

Then start your sample in the test folder of the operating system temporary folder.

No error should be visible.

Look through the command 'strings' that your binary located in the test2 folder contains the login previously found.
If this is not the case, then the correction stops here.

Finally run the infected binary located in the test2 folder and see if it is running without errors.
It must have EXACTLY the same behavior as before infection.

If any of these conditions are not met, then the correction stops here.

✓ Yes

✗ No.

Crash dat virus!

Basic virus

Attempt to use the virus while read / write rights
are not available in the temporary folder.

Attempt to use the virus on files of which one does not have the
rights to read / write (to put in the files test / test2).

At no time should the virus display any error message.

✓ Yes

✗ No.

Advanced virus

The corrector is free to try to crash the virus.

In case of unexpected shutdown, the virus must exit cleanly and no
message should be visible on the computer.

✓ Yes

✗ No.

Bonus Virus

Bonuses should only be assessed if and only if the mandatory game is PERFECT. By PERFECT, we obviously hear that it is fully realized, it is not possible to put his behavior in default, even in case of error, as vicious as it is, misuse, etc. Concretely, this means that if the mandatory part did not obtain ALL the points during this defense, the bonuses must be entirely IGNORED.

Bonus made for this project

You can count up to 5 different bonuses. Bonuses must be a useful minimum, and well implemented.

Here are examples of eligible bonuses

- Infection on 32-bit binary.
- Utility functions.
- Use packing.
- Optimization infection.
- Infection on non-binary file.
- ...

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



ratings

Do not forget to check the flag



Okay



Outstanding project



Empty work



Incomplete work



No author file



Invalid compilation



Standard



cheat



Crash



Incomplete group



Forbidden function

Conclusion

Leave a comment on this evaluation

Preview !!!

