Trickery with LD_PRELOAD

Project Report
Operating Systems Lecture Spring Semester 2025

University of Basel
Faculty of Science
Department of Mathematics and Computer Science

Carina Fehr
Dehlen Thavrajah
Eda Kaynar
Simon Hammer

June 11, 2025

Contents

1 Introduction

LD_PRELOAD is an environment variable in linux. It can be used to load a customized shared library at runtime and before all other, standard objects. LD_PRELOAD can therefore be used to change the behavior of system calls. This does not manipulate the syscalls directly, but the glibc wrappers for them. The modified functions are only called if they have been linked dynamically. With static linking, LD_PRELOAD has no effect. In this project, the following seven functions are modified: getchar(), read(), open(), write(), execve(), connect() and malloc().

1.1 Subsection Figures

1.2 Subsection Tables

2 Methology

The implementation consists of seven hijacked functions that have been combined into a library at the end. When activated in a terminal session with "export LD_PRELOAD=./hacked.so", they give the user the impression that his computer has been seriously hacked.

2.1 Open()

The hijacked open() function has three main elements: when a user tries to open the file "secrets.txt", the function creates a new txt file with the content "you should not be so curious". If the user now writes something, it will be written into this new file, so the user has no way of reading or modifying the secrets. It is also not possible to copy the file to a new one. If the user tries to open the file openThis.txt, he will get a message that he has been hacked and should not do everything he is told. The last manipulation concerns the preloadLib file. This is where the code from the hijacks is implemented, so the user should not be able to manipulate it.

2.2 Write()

The Write function manipulates the terminal output for an invalid command. If the user types a command such as "invalidcommand", it is normally displayed that this command was not found. With the new function, the user is misled by replacing the output with "finished execution: no errors", so that he does not realize that his command has done nothing.

2.3 Read()

The manipulated read function affects the commands cat, bash and less. The commands are not executed, instead the terminal displays in red that unauthorized access has been detected. Then various directories are listed as deleted to give the impression that the user is losing all his files. The control + C command is deactivated in the meantime, simulating total loss of control.

2.4 Lessons learned

We all learned a lot from this project. Not only in terms of programming in C in the Linux environment, which was still quite foreign to us, but also in terms of teamwork. Our project was divided up well among the team members according to the individual functions. This was particularly helpful at the beginning, so that we could all work at our own pace, and showed us how important it is in projects to divide the work evenly and clearly from the start.
We looked in detail at the behavior of the individual functions, which gave us a valuable insight into the commands used on a daily basis.

References

Appendix A: Plots

Appendix B: Declaration of Independent Authorship

Copy text from https://dmi.unibas.ch/fileadmin/user_upload/dmi/Studium/Computer_Science/Diverses/Verwendung_AI/2025-02-17_Eigenstaendigkeitserklaerung_Declaration-of-Independent-Authorship_DE_EN_neu.pdf

And sign it (all authors!)

Signature of all authors as PDF