University of Warsaw
Faculty of Mathematics, Informatics and Mechanics

Krzysztof Małysa
Student no. 394442

Multi-process sandbox for unprivileged users on Linux

Master's thesis
in COMPUTER SCIENCE

Supervisor:
dr Janina Mincer-Daszkiewicz
Institute of Informatics

Warsaw, December 2022

Abstract

TODO

Keywords

sandboxing, security, Linux, secure execution, arbitrary code execution

Thesis domain (Socrates-Erasmus subject area codes)

11.3 Informatics, Computer Science

Subject classification

Security and privacy – Systems security – Operating systems security

Tytuł pracy w języku polskim

Sandbox wielu procesów dla nieuprzywilejowanych użytkowników systemu Linux

Contents

Chapter 1

Introduction

1.1. Background

Secure execution environments are commonplace these days, from containers and virtual machines on servers to sandboxes on laptop and smartphones — most of which run on Linux. They are used to securely execute untrusted code, as well as trusted programs to prevent damage escalation in the event of unknown bugs. Isolation, limiting of and accounting use of the resources is their key features.

The features of Linux allow the creation of simple yet effective and efficient secure environments. They work at application runtime, so in most cases existing software does not need to be adapted to use them. This makes them easily applicable, and explains why they adoption is growing.

1.2. Related work

Approaches to form a secure execution environments differ. One is a virtualization or emulation e.g. QEMU [4] and KVM [3], VirtualBox [5], VMWare Workstation [9]. Although powerful and effective, they come with an enormous overhead i.e. booting up an entire operating system. Moreover, emulation noticeably slows down the runtime of an emulated application.

Containers provide much lower overhead: setup of an order of milliseconds and negligible runtime overhead. Docker [2], LXC [1] require root privileges to create a container. systemd-nspawn [8] requires root privileges to run.

Rootless are containers [6] that can be created and by an unprivileged user. Honestly, I discovered them lately in the process of doing this thesis. They provide almost all of the functionality of the normal containers but without the need to engage a privileged user. However, they often use setuid binaries and that is undesirable [7] and are not optimized to run sequences of short-running programs.

1.3. Goal of this thesis

Is to create a sandbox optimized for an online judge i.e. optimized for running the same program multiple times with different input. Also the sandbox has to be capable enough to allow running a compiler. However the below assumptions must be met:

- The system is running with cgroup v2

- The system is running with systemd
- Unprivileged user namespaces are enabled

The sandbox has to provide execution statistics:

- Memory used
- Execution CPU time
- Execution real time

with the following limits and isolation:

- Real time limit
- CPU time limit for single process programs
- Memory limit
- Stack memory limit
- Process limit
- Filesystem isolation with allowing bindings
- Visible processes isolation
- Visible users isolation
- Visible IPC isolation
- Network isolation
- Hostname isolation

The another goal is to compare the sandbox and its efficiency to rootless containers that are a related solution.

While the sandbox needs to isolate network, the goal is not to provide outer network (e.g. internet) access to the sandboxed programs.

1.4. Structure of the thesis

Chapter 2 contains brief summary of the Linux kernel features employed by the sandbox. Chapter 3 describes the sandbox and its design. In chapter 4 comparison with the rootless containers is described.

Chapter 2

Kernel features

Chapter 3

Sandbox design

Chapter 4

Comparison with rootless containers

Bibliography

[1] David Beserra et al. "Performance Analysis of LXC for HPC Environments." In: *CISIS*. IEEE Computer Society, 2015, pp. 358–363. ISBN: 978-1-4799-8870-9. URL: http://dblp.uni-trier.de/db/conf/cisis/cisis2015.html#BeserraMEBSF15.

[2] Dirk Merkel. "Docker: Lightweight Linux Containers for Consistent Development and Deployment". In: *Linux J.* 2014.239 (Mar. 2014). ISSN: 1075-3583. URL: http://dl.acm.org/citation.cfm?id=2600239.2600241.

[3] *Official website of Kernel Virtual Machine*. URL: https://www.linux-kvm.org/ (visited on 11/23/2022).

[4] *Official website of QEMU — A generic and open source machine emulator and virtualizer*. URL: https://www.qemu.org/ (visited on 11/23/2022).

[5] Oracle. *Official website of VirtualBox*. URL: https://www.virtualbox.org/ (visited on 11/23/2022).

[6] rootlesscontaine.rs. *Rootless Containers*. URL: https://rootlesscontaine.rs (visited on 11/28/2022).

[7] Giuseppe Scrivano. *Rootless containers with Podman and fuse-overlayfs*. June 4, 2019. URL: https://indico.cern.ch/event/757415/contributions/3421994/attachments/1855302/3047064/Podman_Rootless_Containers.pdf (visited on 11/28/2022).

[8] systemd. *systemd-nspawn — Spawn a command or OS in a light-weight container*. URL: https://www.freedesktop.org/software/systemd/man/systemd-nspawn.html (visited on 11/28/2022).

[9] VMWare. *Official website of VMWare Workstation*. URL: https://www.vmware.com/products/workstation/ (visited on 11/23/2022).