Федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики им. Тихонова

Департамент электронной инженерии

ОТЧЕТ

О ПРАКТИЧЕСКОЙ РАБОТЕ №10

по дисциплине «Программные и аппаратные средства защиты информации»

«Программные межсетевые экраны»

Вариант 4

Студент гр. БИБ201

Шадрунов Алексей

Дата выполнения: 16 июня 2023 г.

Преподаватель:

Перов А. А.

«___» _____ 2023 г.

Москва, 2023

Contents

1 Introduction

The concept of containerization is relatively mature. Process isolation traces its origins to the introduction of the chroot syscall in 1979 [A Brief History of Containers: From the 1970s Till Now]. Containerization significantly increased in popularity since 2013, when Docker emerged, and now containers play a leading role in modern software development and distribution practices. As the technology was widely adopting, the security concerns became more evident. As a result, while numerous vulnerabilities particularly related to containerized applications were discovered within the popular platforms, various defensive mechanisms were developed.

2 Terminology

In order to understand the following defensive technologies, we should revisit the basic definitions.

Container is an isolated process which uses a shared kernel [A Brief History of Containers: From the 1970s Till Now]. It may seem similar to Virtual Machine, especially when the process inside the container is shell. However, the containers and virtual machines represent opposite approaches to virtualisation. While each virtual machine usually has a separate guest kernel, which works on top of the host kernel, containers usually share the kernel among each other and the host operating system. Nevertheless, containerized applications are still relatively isolated: they may have its own network stack, separate filesystem and limited set of resources at the disposal. This isolation relies on several Linux kernel features: Linux Namespaces, chroot, cgroups and capabilities. [Liz chapter 4]

Kernel namespaces restrict which resources the process may see. [chapter 4]. After being assigned ? to a particular namespace, the process cannot modify the resources like mount points and networking, and thus is unable to affect the processes outside the given namespace [https://docs.docker.com/engine/security/]. Current Linux kernel supports isolation for eight types of resources:

- Cgroup — isolates cgroup root directory
- IPC — isolates IPC, POSIX message queues
- Network — isolates Network devices, stacks, ports
- Mount — isolates mount points
- PID — isolates process IDs
- Time — isolates boot and monotonic clocks
- User — isolates user and group IDs

- UTS — isolates hostname and NIS domain name

[https://man7.org/linux/man-pages/man7/namespaces.7.html] rewrite

The six namespaces provided by the Linux kernel are IPC, UTS, PID, Network, Mount, and User [5]. The IPC namespace is responsible for the isolation of message queues, semaphores and shared memory; the UTS namespace is used to isolate host names and domain names; the PID namespace is responsible for isolating process PID numbers; the Network namespace is responsible for the isolation of network resources; the Mount namespace is used to isolate files The mount point for the system; the User namespace is used to isolate security-related identifiers and properties. [https://ieeexplore.ieee.org/abstract/document/10090121]

Chroot is another important isolation technology. This command and underlining system call change the root directory of the process. Once executed, the process cannot access anything that was outside of the current root directory [https://man7.org/linux/man-pages/man7/namespaces.7.html, https://man7.org/linux/man-pages/man2/chroot.2.html]. However, it was not designed for properly sandboxing a process, so it should be complimented with other isolation techniques. [https://man7.org/linux/man-pages/man2/chroot.2.html].

There is another system call for changing root directory — pivot_root. In practice, it is preferred, because it uses the mount namespace. Maybe it is more secure from jail escapes [liz 6 41].