УДК 004.056

ИЗОЛЯЦИЯ ВЫПОЛНЯЮЩИХСЯ ПРОЦЕССОВ В LINUX: ПОДХОД К БЕСШОВНОЙ ИНТЕГРАЦИИ В ИЗОЛИРОВАННУЮ СРЕДУ

Бирюков Д.Н. (ВКА), **Компаниец Р.И.** (ВКА), **Яроцкий Г.Д.** (ВКА)
Научный руководитель – доктор технических наук, профессор Бирюков Д.Н.
(Военно-космическая академия имени А.Ф.Можайского)

Введение. В настоящее время изоляция и контейнеризация играют огромную роль в процессах разработки и эксплуатации программного обеспечения. Готовые решения, такие как Docker [1], предлагают заранее описать изолированную среду для последующего развёртывания ПО. В существующих открытых решениях отсутствует возможность интеграции уже запущенных процессов в изолированное окружение.

Основная часть. С целью устранения этого ограничения предлагается подход к бесшовной интеграции процессов в Linux в изолированную среду, сохраняющую непрерывность их выполнения, с последующей разработкой программного обеспечения, позволяющего переносить процессы в контролируемую среду без их остановки. Данное решение использует только инструменты изоляции, предоставляемые самим Linux, что делает его легковесным и легко портируемым под различные сборки и позволяет минимизировать риски, связанные с анализом вредоносного программного обеспечения, тестированием уязвимых приложений и обеспечением безопасности систем, не нарушая их работоспособности. В условиях растущих киберугроз и необходимости оперативного реагирования на инциденты, предлагаемый подход представляет собой перспективное решение, способное добавлять процесс в отдельную среду, если системы безопасности посчитают его подозрительным.

Для реализации программного решения используются ключевые механизмы Linux, такие как *Linux Namespaces* [2], обеспечивающие изоляцию различных аспектов системы:

- *Mount Namespace* – изолирует файловую систему;
- *PID Namespace* – ограничивает видимость идентификаторов процессов;
- *Network Namespace* – изолирует сетевые ресурсы;
- *User Namespace* – управляет правами пользователей.

Дополнительно применяется *Seccomp-bpf* для ограничения системных вызовов, доступных процессу [3], и *Cgroups* контроля над ресурсами, которые могут быть использованы процессом. *Cgroups* также позволяют отслеживать попытки доступа к определённым компонентам, что делает их эффективным инструментом для выявления и анализа вредоносной активности в системе [4]. Кроме того, *Linux capabilities* обеспечивают гибкое управление привилегиями процесса и контроль над наследованием прав дочерними процессами [5].

Выводы. Таким образом, предлагаемый подход формирует новый класс решений по предотвращению вторжений в систему и позволяет изолировать уязвимые или подозрительные процессы, не отключая их, что в свою очередь не нарушает работу важных приложений, не приводит к сбоям в работе остальных компонентов системы, взаимодействующих с этим процессом, а также позволяет не только контролировать и анализировать подозрительные процессы, исключая необходимость их остановки. Это делает подход актуальным и востребованным в таких областях, как кибербезопасность, разработка и тестирование программного обеспечения.

Список использованных источников:

1. Документация по Docker [Электронный ресурс]. – URL: https://docs.docker.com/ – (дата обращения 01.10.2024).

2. Linux Namespaces [Электронный ресурс]. – URL: https://man7.org/linux/man-pages/man7/namespaces.7.html – (дата обращения 06.09.2024).

3. Защита контейнеров с помощью фильтров Seccomp [Электронный ресурс]. – URL: https://habr.com/ru/companies/ruvds/articles/689184/ – (дата обращения 10.09.2024).

4. Linux Kernel Documentation [Электронный ресурс]. – URL: https://www.kernel.org/doc/Documentation/cgroup-v1/cgroups.txt – (дата обращения 16.09.2024).

5. В двух словах о привилегиях Linux (capabilities) [Электронный ресурс]. – URL: https://habr.com/ru/companies/otus/articles/471802/ – (дата обращения 20.09.2024).

Яроцкий Г.Д. (автор) Подпись

Бирюков Д.Н. (автор) Подпись

Компаниец Р.И.(автор) Подпись