ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Обзорная статья
УДК 004.4'6
EDN YYPHHD
https://doi.org/10.34216/2587-6147-2024-3-65-67-71
Илья Романович Марков [1]
Вадим Андреевич Кулипанов [2]
Александр Валерьевич Орлов [3]
[1,2,3] Костромской государственный университет, г. Кострома, Россия
[1]ilya.markov212001@gmail.com, https://orcid.org/0000-0002-2522-6116
[2]vadimkulipanov555@mail.ru, https://orcid.org/0000-0002-0456-02973
[3]aorlov@list.ru, https://orcid.org/0000-0002-4995-3393

СОРЕВНОВАТЕЛЬНАЯ ПЛАТФОРМА ДЛЯ ПРОГРАММИРОВАНИЯ И МЕТОДЫ БЕЗОПАСНОГО ВЫПОЛНЕНИЯ НЕДОВЕРЕННОГО КОДА

Аннотация. *В статье рассматривается использование информационных систем для проведения олимпиад по программированию, в частности, в контексте исполнения кода из недоверенных источников. Проведен анализ существующих систем предназначенных для обеспечения безопасного выполнения задач участниками олимпиады, а также обсуждаются проблемы, с которыми сталкиваются организаторы и разработчики таких систем. Показаны их недостатки, такие как сложность использования, ограниченная функциональность, коммерческая направленность. Предложена система, способная функционировать во внутренней локальной сети учреждения, обеспечивая возможность проведения как онлайн-, так и офлайн-олимпиад. В статье уделено внимание мерам безопасности, которые могут быть приняты для минимизации рисков при выполнении кода из ненадежных источников. Основное внимание в работе авторы акцентируют на анализе существующих методов изоляции и виртуализации, выявляя их преимущества и недостатки.*
Ключевые слова: *современное образование, программирование, олимпиады, информационная система, проверка результатов, права доступа процесса, файловая система, виртуализация*

Для цитирования: Марков И. Р., Кулипанов В. А., Орлов А. В. Соревновательная платформа для программирования и методы безопасного выполнения недоверенного кода // Технологии и качество. 2024. № 3(65). С. 67–71. https://doi.org/10.34216/2587-6147-2024-3-65-67-71.

Review article
Ilya R. Markov [1]
Vadim A. Kulipanov [2]
Alexander V. Orlov [3]
[1,2,3] Kostroma State University, Kostroma, Russia

COMPETITIVE PROGRAMMING PLATFORM AND METHODS OF SAFELY EXECUTING UNTRUSTED CODE

Abstract. *The article discusses the use of information systems for holding programming Olympiads, especially in the context of executing code from untrusted sources. An analysis of existing systems designed to ensure the safe execution of tasks by Olympiad participants is conducted, and the problems faced by organisers and developers of such systems are discussed. Their shortcomings are shown, such as complexity of use, limited functionality, commercial focus. A system is proposed that can function in the internal local network of an institution, providing the ability to hold both online and offline Olympiads. The article also pays attention to security measures that can be taken to minimise risks when executing code from untrusted sources. The au-*

thors focus on the analysis of existing isolation and virtualisation methods, identifying their advantages and disadvantages.

Keywords: *modern education, programming, Olympiads, information system, checking results, process access rights, file system, virtualisation*

For citation: Markov I. R., Kulipanov V. A., Orlov A. V. Competitive programming platform and methods of safely executing untrusted code. Technologies & Quality. 2024. No 3(65). P. 67–71. (In Russ.). https://doi.org/10.34216/2587-6147-2024-3-65-67-71.

Введение. Олимпиады по программированию являются важной частью системы образования в области информатики. Они позволяют выявить одаренных детей, стимулировать интерес к программированию, а также повысить уровень подготовки будущих специалистов.

Традиционно олимпиады по программированию проводятся в очном формате. Однако это имеет ряд ограничений:

- высокая стоимость: организация очных олимпиад требует значительных финансовых затрат;
- ограниченная доступность: не все школьники и студенты имеют возможность участвовать в очных олимпиадах, которые проводятся в крупных городах.

В связи с этим актуальной становится задача создания платформы для проведения олимпиад по программированию в дистанционном формате.

Существующие решения. В настоящее время существует ряд платформ для проведения олимпиад по программированию. Наиболее популярными из них являются:

- CodeForces: https://codeforces.com;
- TopCoder: https://www.topcoder.com;
- AtCoder: https://atcoder.jp;
- HackerRank: https://www.hackerrank.com;
- LeetCode: https://leetcode.com.

Эти платформы предоставляют широкий спектр возможностей для организации и проведения олимпиад, включая:

- создание и редактирование задач: платформы позволяют создавать задачи различной сложности, используя различные языки программирования;
- систему оценки: платформы автоматически оценивают решения участников олимпиады, что обеспечивает честность и объективность оценки результатов;
- статистику и рейтинги: платформы позволяют отслеживать результаты участников олимпиады, а также сравнивать их достижения с достижениями других участников.

Недостатки существующих решений. Несмотря на свои преимущества, существующие платформы имеют ряд недостатков:

- сложность использования: многие платформы имеют сложный интерфейс и требуют от пользователей специальных знаний в области программирования;
- ограниченная функциональность: некоторые платформы не предоставляют всех необходимых функций для организации и проведения олимпиад, например возможности создания командных олимпиад или проведения олимпиад с несколькими турами;
- коммерческая направленность: многие платформы являются платными, что делает их недоступными для широкого круга пользователей.

Предлагаемая архитектура. Разработанная нами система способна функционировать во внутренней локальной сети учреждения, обеспечивая возможность проведения как онлайн-, так и офлайн-олимпиад. Также предлагаемая платформа имеет следующие преимущества:

1. Персонализированный подход. Мы стремимся создать простую и удобную платформу, которая будет отвечать потребностям олимпиадных участников. Мы предоставляем возможность участникам выбирать уровень сложности задач, участвовать в турнирах и соревнованиях, а также учиться на практических примерах.

2. Разнообразные форматы соревнований. Мы предлагаем различные форматы соревнований, начиная от классических олимпиадных задач до групповых проектов, где участники должны работать в команде, чтобы успешно решить поставленные задачи.

3. Обратная связь и поддержка. Мы предоставляем участникам обратную связь по их решениям, помогаем им развивать свои навыки программирования и стимулируем к саморазвитию. Мы также готовы помогать участникам в случае возникновения вопросов или проблем.

4. Организаторы. Организация олимпиад будет предоставляться различным организаторам, таким как школы, вузы, а также IT-компании, которые впоследствии смогут выделить для себя потенциальных сотрудников.

В целом разработанная нами информационная система для проведения олимпиад по

программированию стремится создать стимулирующую и интересную среду для развития навыков программирования у участников, отличаясь своим уникальным подходом к проведению соревнований и обучению.

Предлагается следующая архитектура платформы для проведения олимпиад по программированию:

- Ядро платформы: ядро платформы должно обеспечивать работу основных функций платформы, таких как создание и редактирование задач, система оценки, статистика и рейтинги.
- Модуль управления пользователями: этот модуль должен обеспечивать регистрацию и авторизацию пользователей, а также управление их профилями.
- Модуль управления соревнованиями: этот модуль должен обеспечивать создание, настройку и проведение олимпиад.
- Модуль системы оценки: этот модуль должен обеспечивать автоматическую оценку решений участников олимпиады.
- Модуль статистики и рейтингов: этот модуль должен обеспечивать отображение статистики и рейтингов участников олимпиады.

Преимущества предлагаемой архитектуры. Предлагаемая архитектура имеет ряд преимуществ:

- модульность: модульная архитектура платформы позволяет легко добавлять новые функции и возможности;
- масштабируемость: платформа может быть легко масштабирована для поддержки большого количества пользователей и соревнований;
- простота использования: платформа имеет простой и понятный интерфейс, который доступен для широкого круга пользователей;
- открытость: платформа открыта для модификации, что позволяет другим разработчикам создавать новые функции и возможности.

Вопросы безопасности. Создание платформы для проведения олимпиад по программированию в дистанционном формате позволит решить ряд проблем, связанных с проведением очных олимпиад. Предлагаемая архитектура платформы является модульной, масштабируемой, простой в использовании и открытой для модификации. Но чтобы создать свою платформу, нужно учесть безопасность исполнения недоверенного кода.

Существующие аспекты изоляции кода. В настоящее время существует ряд аспектов для изоляции кода, которые позволили бы обеспечить безопасность при исполнении недоверенного кода:

- права доступа процесса;
- файловая система;
- изоляция сети;
- виртуализация.

Рассмотрим различные подходы для обеспечения изоляции кода по всем вышеупомянутым аспектам.

Права доступа процесса. User: nobody. Во многих Unix-подобных операционных системах, nobody [1] (англ. nobody – никто) – имя пользователя, не являющегося владельцем ни одного файла, не состоящего ни в одной привилегированной группе и не имеющего никаких полномочий, кроме стандартных для обычных пользователей. Идентификатор пользователя nobody обычно или наибольший из возможных (в противоположность суперпользователю), или один из системных идентификаторов пользователя (во многих системах – между 1 и 100).

В общем случае запуск демонов от имени nobody, часто применяющийся на серверах, призван уменьшить размер повреждений, которые может нанести злоумышленник, получивший над этим процессом контроль. Тем не менее полезность этого способа снижается, если подобным образом можно запустить более одного демона, так как получение контроля над одним из них позволит получить контроль и над всеми остальными. Причина этого в том, что процессы, запущенные от имени nobody, имеют возможность посылать сигналы друг другу, а также могут использовать системные вызовы отладки друг над другом (например, ptrace в Linux), что означает возможность чтения и записи одним процессом памяти другого процесса. Создание отдельного аккаунта для каждого демона, как это рекомендует Linux Standard Base, обеспечивает более надежную политику безопасности.

Файловая система chroot. При использовании chroot [2] создается новая среда оболочки с отдельным корневым каталогом, не влияющим на основную систему. Это удобно для тестирования программ с другими библиотеками без изменения основной системы. Но chroot требует опыта, так как неправильное использование может нарушить стабильность системы.

PIVOT_ROOT

Имея наш новый mount namespace и копию системных файлов, мы хотели бы смонтировать эти файлы в корневом каталоге нового mount namespace. Linux предлагает системный вызов pivot_root [3] (есть соответствующая

команда), который позволяет контролировать то, что именно процессы видят как корневую файловую систему.

Изоляция сети. Iptables. Iptables [4] – это мощный инструмент управления сетью в Linux, который позволяет администраторам управлять входящими и исходящими пакетами данных. Это основной инструмент для настройки межсетевых экранов в системах Linux.

Iptables работает путем проверки пакетов данных на соответствие определенным критериям и выполнения заданных действий, если пакеты соответствуют этим критериям. Эти критерии и действия определяются в таблицах, которые состоят из набора правил.

Виртуализация. Docker. Докер [5] – платформа для разработки, доставки и эксплуатации приложений, ускоряющая процесс разработки и выкладывания. С помощью докера приложение можно отделить от инфраструктуры и управлять им как управляемым приложением. Docker помогает быстрее выкладывать код, тестировать и запускать приложения, сокращая время от написания до запуска. Платформа контейнерной виртуализации позволяет запускать практически любое приложение, безопасно изолированное в контейнере, обеспечивая легковесную работу без гипервизора. Docker полезен для упаковывания, раздачи и выкладывания приложений для разработки, тестирования и продакшена в дата-центры и облака.

Комплексные решения. Отдельно стоит упомянуть комплексные решения, которые охватывают сразу несколько аспектов изоляции кода.

БЕЛЫЙ СПИСОК МОДУЛЕЙ

В списке рассылки ядра Linux опубликован набор патчей с реализаций LSM-модуля White Egret [6], представляющего средства для обеспечения защиты системы через применение белого списка исполняемых компонентов. White Egret допускает исполнение только кода приложений и библиотек, которые явно разрешены и занесены в заранее определенный белый список. Исполнение всех не включенных в список приложений блокируется, что не позволяет выполнить в системе недозволенные программы и вредоносное ПО. White Egret хорошо подходит для статичных окружений, состав которых не меняется длительное время, например для типовых серверов и промышленных управляющих систем.

Обработка белого списка дозволенных программ выполняется в пользовательском окружении при помощи процесса WEUA (White Egret User Application). В процессе обработки системных вызовов execve и mmap_file ядро отправляет в WEUA запрос, передавая полный путь к исполняемому файлу. WEUA на основании белого списка принимает решение о возможности исполнения данного файла. Вызов mmap_file применяется для перехвата загрузки разделяемых библиотек в область памяти, допускающую выполнение. Кроме файлового пути, проверяется хэш от содержимого исполняемого файла, что позволяет блокировать файлы, измененные после занесения в белый список. Взаимодействие WEUA с ядром осуществляется при помощи интерфейса netlink.

APPARMOR

AppArmor [7] является реализацией системы Mandatory Access Control (MAC), основанной на архитектуре Linux Security Modules (LSM). Модель безопасности AppArmor заключается в привязке атрибутов контроля доступа не к пользователям, а к программам. AppArmor обеспечивает изоляцию с помощью профилей, загружаемых в ядро, как правило, при загрузке.

AppArmor отличается от остальных реализаций MAC в Linux принципом действия на основе путей, также он позволяет смешивать профили принудительного исполнения и режима предупреждений. Кроме того, AppArmor использует вложенные файлы для облегчения разработки.

Система безопасности Mandatory Access Control (MAC) предполагает централизованный контроль над правилами политики доступа, при котором рядовые пользователи не имеют возможности вносить в них какие-либо изменения. Разработчик политики определяет, какие программы или процессы могут выполнять определенные действия с системными ресурсами. MAC фокусируется в большей степени на программах, нежели на пользователях и решает задачу разграничения доступа процессов к ресурсам ОС.

Предлагаемый подход. На основе приведенных ранее подходов для обеспечения безопасности исполнения недоверенного кода нами сделан выбор в пользу AppArmor и Docker. AppArmor обеспечивает безопасность в аспектах прав доступа процесса, файловой системы и изоляции сети, а Docker – виртуализации. Такой выбор был сделан в силу объемной документации и большой популярности данных подходов.

Заключение. Использование технологии песочницы позволяет обеспечить безопасное исполнение недоверенного кода на платформе для проведения олимпиад по программированию. Предлагаемый подход является простым, экономичным и гибким.

СПИСОК ИСТОЧНИКОВ
1. Nobody (пользователь) // Wikipedia. URL: https://ru.wikipedia.org/wiki/Nobody_ (дата обращения: 25.05.2024).
2. Антонов К. Как использовать утилиту Chroot в Linux? // Server Space. URL: https:// server-space.by/support/help/kak-ispolzovat-utilitu-chroot-v-linux/?utm_source=google.com&utm_medium= organic&utm_campaign=google.com&utm_referrer=google.com. (дата обращения: 25.05.2024).
3. Убах И. П. Глубокое погружение в Linux namespaces. Ч. 3 // Хабр. URL: https://habr.com/ ru/articles/541304 (дата обращения: 25.05.2024).
4. Введение в Iptables // Хабр. URL: https://habr.com/ru/articles/747616 (дата обращения: 25.05.2024).
5. Понимая Docker // Хабр. URL: https://habr.com/ru/articles/253877 (дата обращения: 25.05.2024).
6. Для ядра Linux предложена реализация белого списка исполняемых приложений // OpenNET. URL: https://opennet.ru/46626-whiteegret (дата обращения: 25.05.2024).
7. Безопасный Linux вместе с AppArmor // Хабр. URL: https://habr.com/ru/companies/ruvds/ articles/532988 (дата обращения: 25.05.2024).

REFERENCES
1. Nobody (user): Wikipedia. URL: https://ru.wikipedia.org/wiki/Nobody_ (accessed 25.05.2024).
2. Antonov K. How do I use the Chroot utility on Linux? Server Space. URL: https://serverspace.by/support/help/kak-ispolzovat-utilitu-chroot-v-linux/?utm_source=google.com& utm_ medium= organic&utm_campaign=google.com&utm_referrer=google.com. (accessed 25.05.2024).
3. Ubah I. P. Deep dive into Linux namespaces. P. 3. The Habr. URL: https://habr.com/ru/articles/541304 (accessed 25.05.2024).
4. Introduction to Iptables. The Habr. URL: https://habr.com/ru/articles/747616 (accessed 25.05.2024).
5. Understanding Docker. The Habr. URL: https://habr.com/ru/articles/253877 (accessed 25.05.2024).
6. For the Linux kernel, an implementation of the white list of executable applications is proposed. Open-Net. URL: https://opennet.ru/46626-whiteegret (accessed 25.05.2024).
7. Secure Linux with AppArmor. The Habr. URL: https://habr.com/ru/companies/ruvds/articles/532988 (accessed 25.05.2024).

Статья поступила в редакцию 6.06.2024
Принята к публикации 23.09.2024