РАЗРАБОТКА ТЕСТИРУЮЩЕЙ СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ СОВРЕМЕННЫХ ТЕХНОЛОГИЙ ИЗОЛЯЦИИ ПРОЦЕССОВ

Удовин И. А., Воронова В. В.
Кафедра информатики, Белорусский государственный университет информатики и радиоэлектороники
Минск, Республика Беларусь
E-mail: wilcot@ya.ru, veronika.voronova31@gmail.com

В данной статье рассматривается разработка тестирующей системы с использованием современных технологий изоляции процессов. Проанализирована предметная область применения программного комплекса, выделены функциональные требования. Описаны проблемы и их возможные решения, связанные с изолированным запуском кода на операционной системе Linux.

Введение

С развитием информационных технологий, которые требуют глубоких знаний для их применения, появляется все большая потребность в высококвалифицированных специалистах в сфере IT.

Как правило, знакомство с информационными технологиями начинается уже со школы. Кроме теоретической составляющей, значительную роль играет практическая часть. Школьникам предлагают писать программы, запускать их на компьютере и наблюдать за полученными результатами. С этого момента важно заложить понимание, что с помощью программ можно и нужно решать большее количество практических задач.

Олимпиады по спортивному программированию, в большинстве случаев, являются практическими и требуют написания кода, решающего поставленные задачи. При проверке решений возникает потребность в автоматизированных тестирующих системах. Хотя корректность программы можно проверить и вручную, использование автоматического способа крайне необходимо для избежания ошибок, возникающих в следствии человеческого фактора, а так же для ускорения процесса. Олимпиады по информатике как раз позволяют проверять решения в автоматическом режиме, ведь программы участников можно компилировать и запускать без непосредственного участия учителя.

В тоже время, тестирующие системы могут применяться не только для проведения школьных олимпиад, но и для проверки лабораторных работ в университетах, при собеседованиях в компании или при проведении соревнований различного рода.

I. Описание требований системы

Основной задачей тестирующей системы можно определить возможность проведения соревнований и собеседований. Можно выделить две основные группы пользователей системы: участники и организаторы соревнований, а так же администратор.

Минимально-требуемыми функциями с точки зрения участника либо организатора любой тестирующей системы можно выделить следующие:

- регистрация и авторизация;
- создание и редактирование соревнований;
- создание и редактирование пакетов задач;
- разделение пользователей по правам и ролям, управление доступом пользователей;
- просмотр условий задач;
- отправка решений по задачам;
- получение результатов тестирования решения.

Администратор в свою очередь может управлять списком доступных языков программирования, а также настраивать компиляцию и запуск решений, написанных на языке, которого еще нет в системе.

II. Основные компоненты системы

Для того, чтобы система могла безотказно выполнять необходимые функции, а также имела возможность горизонтально масштабироваться, она может быть разделена на следующие два компонента:

- web-интерфейс тестирующей системы – интерфейс, с которым взаимодействует пользователь;
- демон асинхронных операций (далее Invoker) – программа, предназначенная для тестирования посылок пользователей в изолированной среде.

Для работы тестирующей системы необходима база данных для хранения следующей информации:

- информация о пользователях, такая как логины, хеши паролей;
- информация о соревнованиях (название, тип, описание, список задач);
- информация по посылках пользователей (вердикт, использованное время и память);
- текущее состояние очереди посылок.

К примеру, может использоваться PostgreSQL - объектно-реляционная система управления базами данных [1].

Для работы Invoker-а необходима файловая система для хранения следующей информации:

- пакетов задач, состоящих из ручных тестов, чекеров, валидаторов и генераторов;
- результатов тестирования каждой посылки: вывод на этапе компиляции, выходные данные программы.

В этих целях удобно использовать распределенную файловую систему CephFS.

III. Пакет задачи и проверка решения

Пакет задачи представляет из себя архив данных со следующим содержимым:

1. чекер – программа, проверяющая, что выходные данные программы соответствуют выходным данным теста;
2. валидатор – программа, проверяющая, что входные данные тестов соответствуют условию задачи;
3. генератор – программа, генерирующая входные данные для тестов;
4. набор тестов – входные и ожидаемые выходные данные.

В случае, если для генерации теста используется генератор, то вместо входных данных указываются параметры запуска генератора.

Так как решение некоторой задачи представляет из себя исходный код программы, то для проверки его корректности необходимо запустить программу на наборе тестов из пакета задачи. Проверка решения состоит из следующих этапов:

1. компиляция исходного кода генераторов, чекеров и валидаторов;
2. компиляция исходного кода программы решения задачи;
3. генерация тестов;
4. запуск исполняемого файла на наборе тестов;
5. анализ результатов выполнения программы.

Каждый из этих этапов подразумевает запуск исполняемого кода. Во время компиляции это запуск компилятора, во время тестирования – запуск решения, во время анализа результатов – запуск чекера. Выполнение этих программ не должно влиять друг на друга, поэтому возникает необходимость в их изоляции.

IV. Изолированный запуск программ

Для того, чтобы запускать программы в изолированной среде, можно использовать технологии виртуализации, что позволит безопасно запускать программы. Однако данный подход имеет большие накладные расходы из-за необходимости запуска операционной системы в этой среде. Это сильно замедляет тестирование решений и увеличивает потребление ресурсов.

Начиная с версии 3.8 ядра Linux, операционная система позволяет запускать программы в изолированной среде без использования технологии виртуализации. Это стало возможным благодаря Linux namespaces – пространств имен, в которых запускаются программы [2].

При запуске любой программы, Invoker создает отдельные пространства имен: `IPC`, `MOUNT`, `NET`, `PID`, `USER` и `UTS`. Для большей безопасности, Invoker запускается от обычного пользователя (не от суперпользователя `root`). В таком случае нам становятся недоступны некоторые необходимые нам команды, например `mount`. Поэтому в начале производится инициализация пространства имен `USER`. Это позволяет получить нам `root`-права внутри изолированной среды. После этого можно спокойно инициализировать оставшиеся пространства имен [3].

Для того, чтобы не нужно было для каждого теста заново инициализировать все пространства имен, они инициализируются один раз перед запуском всех тестов. Теперь программа решения задачи помещается в заранее инициализированные пространства имен и запускается на каждом тесте. Такой подход позволяет значительно ускорить тестирование, однако возникает риск, что запуск одного теста может повлиять на запуск других, так как изменения в изолированной файловой системе остаются. Чтобы этого избежать и не создавать заново пространство имен `MOUNT` для каждого теста, используется виртуальная файловая система `OverlayFS`. Она позволяет сохранить исходные версии файлов, а все изменения помещает в отдельный каталог, который перед каждым тестом можно отчистить [4].

Заключение

Разработанное программное средство позволяет проводить собеседования и олимпиады, где в качестве решения предлагается написание программ. Для компиляции и запуска программ используются современные технологии изоляции процессов, что позволяет избежать последствий запуска вредоносного кода, а также ускорить тестирование программ за счет уменьшения накладных расходов благодаря использованию технологий виртуализации.

Список литературы

1. Новиков, Б. А. Основы технологий баз данных: учебное пособие / Б. А. Новиков, Е. А. Горшкова, Н. Г. Графеева; под ред. Е. В. Рогова. – 2-е издание – М.: ДМК Пресс, 2020. – С. 33–34. – ISBN 978-5-97060-841-8.
2. Namespaces in operation, part 1: namespaces overview [Электронный ресурс] / LWN.net – Режим доступа: https://lwn.net/Articles/531114/. – Дата доступа: 26.09.2020.
3. User namespaces progress [Электронный ресурс] / LWN.net – Режим доступа: https://lwn.net/Articles/528078/. – Дата доступа: 26.09.2020.
4. Wang, K. C. Systems Programming in Unix/Linux – Springer, 2018. – С. 119–120. – ISBN 978-3-319-92428-1