УДК 004.4 **Информационные технологии**

Убеев Валерий Геннадьевич, *специалист, Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича, Россия, г. Санкт-Петербург*

АНАЛИЗ СУЩЕСТВУЮЩИХ МЕХАНИЗМОВ КОНТЕЙНЕРИЗАЦИИ В ОПЕРАЦИОННЫХ СИСТЕМАХ

Аннотация: В статье рассматриваются существующие механизмы контейнеризации в операционных системах. Как подходить к вопросу её защиты, а также рассмотрим, что нам предлагают основные игроки рынка защиты сред контейнеризации. Рассмотрим конкретные технологии. Чем отличается контейнеризация от виртуализации, преимущества контейнеризации.

Ключевые слова: механизм контейнеризации, операционная система, технология, приложение, виртуализация.

Annotation: The article discusses the existing containerization mechanisms in operating systems. How to approach the issue of its protection, and also consider what the main players in the market for protecting containerization environments offer us. Consider specific technologies. What is the difference between containerization and virtualization, advantages of containerization.

Keywords: containerization mechanism, operating system, technology, application, virtualization.

В 1979 году впервые зародилась идея изоляции пользовательских пространств. Именно в этот период в ядре UNIX появился такой системный вызов, как chroot. Рассматриваемый системный вызов предоставлял возможность поменять путь каталога корня, предназначавшийся для целой группы процессов, на абсолютно новую локацию в файловой системе. Иными

словами, он позволял формировать новый корневой каталог, который в свою очередь полностью изолирован от первого. В качестве дальнейшего развития системного вызова chroot стало формирование FreeBSD jails, которое произошло в новом столетии, а именно в 2000 году. В рамках FreeBSD jails изначально возникла частичная изоляция сетевых интерфейсов. В течение первых годов нового столетия стали активно развиваться технологии виртуализации на уровне ОС. Так, в 2001 году был создан Linux VServer, в 2004 году был создан Solaris Containers, а в 2005 году – OpenVZ. Стоит отметить, что в 2008 году впервые была презентована такая система, как LXC (LinuX Containers). Упомянутая система предоставляла возможность одновременно запускать сразу несколько изолированных Linux-систем на одном сервере. Система LXC применяла для работы ряда механизмов изоляции ядра. К данным механизмам относятся: namespaces и cgroups. В 2013 году разработчиками создана и представлена платформа Docker, которая популяризирована контейнерные технологии за счет крайне простого использования и довольно обширного перечня функциональных возможностей. Подчеркивается, что изначально Docker применял LXC с целью запуска контейнеров. Но со временем Docker перешел на использование собственной библиотеки libcontainer, которая аналогично была завязана на функционале ядра Linux.

Технология контейнеризации представляет собой одну из форм виртуализации ОС. Рассматриваемая технология предлагает изоляцию приложений в контейнерах. Стоит отметить, что абсолютно все контейнеры применяют одинаковую операционную систему. В связи с этим, посредством использования данной технологии представляется возможным осуществлять запуск приложения с необходимыми библиотеками в типовом контейнере. Данный контейнер, в свою очередь, соединяется с хвостом или же любой иной внешней компонентной посредством использования просто интерфейса. В качестве главного достоинства исследуемой технологии выделяют снижение непроизводительных накладных расходов, проблема которых является актуальной при использовании технологии виртуализации. Чтобы грамотно и

четко организовать работу полноценных виртуальных машин, аппаратное обеспечение компьютерной техники должно быть эмулировано одним из гипервизоров, в число которых можно включить Xen, Hypev-V и KVM – каждая виртуальная машина имеет в своем составе полноценную копию гостевой ОС. Тогда же, когда применяется технология контейнеров, компьютер имеет всего одно общее ядро операционной системы, внутри легковесных контейнеров поверх которого работают процессы запущенных программ. В таком случае обязанность эмулировать аппаратное обеспечение пропадает, так как система использует физические средства сервера. К контейнерным системам можно отнести Linux-VServer, LXC, Docker и систему OpenVZ.

Разберем каждую технологию. Так, говоря о Linux-VServer можно отметить, что система позволяет неоднократно создавать виртуальные сервера, именуемые VPS, работающие независимо под контролем одного ядра ОС. Каждый виртуальный сервер своей структурой связан с неким контекстом, под которым понимается упорядоченная совокупность определенных свойств, включающая в себя имена конкретных пользователей и групп с идентификаторами, принадлежащими им, и объединяющая процессы данной системы, не позволяя при этом появляться процессам, не входящим в контекст. Процессы корневого сервера внедряются в контекст под нулевым номером и имеют множество возможностей, что в основном реализовано за счет доступности для них процессов и данных виртуальных серверов. Главный сервер, в своей файловой системе располагает корни систем виртуальных серверов в форме некоторого каталога или директории. Основным преимуществом этого является возможность масштабирования на случай, когда в узле хранения данных обнаруживается довольно большое количество контейнеров. Главный набор недостатков же заключается в сложности реализации механизмов сохранения и восстановления состояния и живой миграции, что происходит из-за того, что процессы, после перезапуска, не могут получить те же самые PID (идентификаторы процессов).

Система виртуализации OpenVZ базируется на ядре Linux. В качестве

основного свойства рассматриваемой системы выступает довольно динамичное и активное распределение имеющихся ресурсов физического сервера. Под физическим сервером понимается его процессор и память. Стоит отметить, что распределение осуществляется одновременно между всеми запущенными машинами. При этом каждая из упомянутых машин получает такое количество ресурсов, которое необходимо для реализации полноценной и эффективной работы. Исследуемый подход предоставляет возможность использовать ее с наивысшей эффективностью, поскольку одномоментно производится загрузка довольно большого числа виртуальных серверов. Благодаря указанному свойству, решения на основе контейнерной виртуализации, зачастую распространяются среди действующих хостинговых организаций.

Рассматриваются следующие элементы, поддерживаемые системой LinuX Containers: пространства имен ядра Linux, для изоляции сетевых процессов используется network namespaces, cgroups. Для того, чтобы система могла планировать ввод и вывод она использует алгоритм Completely Fair Queuing, точно также как и в OpenVZ, упомянутом ранее.

На данный момент наиболее популярный инструмент для контроля контейнеров - Docker, который базируется на LXC-системе и является своеобразным расширителем ее возможностей. К главным достоинствам Docker относится следующее:

- ускорение процесса разработок. Иными словами, теперь не существует потребности в установке различных дополнительных инструментов, поскольку их запуск осуществляется сразу в контейнере;
- комфортное программирование приложений, утилит;
- понятное выполнение мониторинга;
- простота процесса масштабирования.

Докер может осуществлять свою функциональную деятельность, как на базовой операционной системе Linux, так и на других популярных площадках ОС, к которым относятся Windows и macOS. Основное отличие в использование докера на базе Windows состоит в том, что на данной площадке допускается

программирование платформы в небольшую виртуальную машину. В настоящее время, Docker считается относительной молодой технологией, активно развивающейся на рынке. Именно по этой причине в ней с некоторой периодичностью выявляются различные недочеты, которые оперативно устраняются. Подобные недочеты в работе и их устранение влечет за собой утрату синхронизации, а также наблюдается нарушение в процессе совместимости и трудности в переходе на новую версию. Следовательно, можно отметить, что пользователи при решении выявленных проблем, впоследствии сталкиваются с еще большим перечнем недочетов, а именно: различные изменения в формате конфигурации файлов; полная несовместимость с рядом иных современных ПО.

Следует подчеркнуть, что в условиях изменения среды приложений, как показывает практика, зачастую появляются такие проблемы в работе, возникновение которых изначально даже не предполагалось. В том числе это происходит в условиях запуска кода в процесс тестирования соответствующего специального оборудования. Подчеркивается, что изменения могут затрагивать, как вычислительные ресурсы, так и сети в целом. Различия могут наблюдаться в типологиях сети, политики безопасности отдельно взятой сети и иных аспектах. В результате все это отражается на эффективности применения виртуальных контейнеров.

Сравнительный анализ таких механизмов контейнеризации, как OpenVZ, LXC, Docker и Linux-VServer сведен и представлен в таблице 1.

При изучении данных таблицы, представленной ниже, становится ясно, что на данный момент лучшие системы контейнеризации – LinuX Container (LXC) и Docker, первая из которых довольно сложна в настройке, вследствие чего более предпочтительной становится система Docker, основанная на первой и максимально упрощающая создание и настройку контейнеров.

Таблица 1 – Сравнительный анализ механизмов контейнеризации.

Критерии \ Система управления контейнерами	Linux-VServer	OpenVZ	LXC	Docker
Изоляция процессов, IPC	+	+	+	+
Изоляция файловой системы	+	+	+	+
Изоляция сети	+/- (сеть физического сервера)	+	+	+
Планирование процессора	+	+	+	+
Ограничение использования системных ресурсов	+	+	+	+
Возможность сохранения /восстановления	-	+	+	+
Возможность живой миграции	-	+	+	+

Библиографический список:

1. Гельфанд А. М. СПОСОБЫ ВЫБОРА СТЕГОКОНТЕЙНЕРОВ ДЛЯ ПЕРЕДАЧИ ДАННЫХ //Региональная информатика и информационная безопасность. – 2020. – С. 260-262.

2. Гельфанд А. М. и др. Разработка модели распространения самомодифицирующегося кода в защищаемой информационной системе //Современная наука: актуальные проблемы теории и практики. Серия: Естественные и технические науки. – 2018. – №. 8. – С. 91-97.

3. Волкогонов В. Н., Гельфанд А. М., Деревянко В. С. Актуальность автоматизированных систем управления //Актуальные проблемы

инфотелекоммуникаций в науке и образовании (АПИНО 2019). – 2019. – С. 262-266.

4. Цветков А. Ю. АНАЛИЗ СУЩЕСТВУЮЩИХ МЕТОДОВ АТАК ТИПА ПЕРЕПОЛНЕНИЯ БУФЕРА НА ОПЕРАЦИОННЫЕ СИСТЕМЫ СЕМЕЙСТВА MICROSOFT //Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2019). – 2019. – С. 751-756.

5. Катасонов А. И., Красов А. В., Цветков А. Ю. РАЗРАБОТКА УНИВЕРСАЛЬНОГО АЛГОРИТМА ПО СОЗДАНИЮ ПРОСТЕЙШИХ МОДУЛЕЙ ЯДРА ДЛЯ РАЗЛИЧНЫХ ВЕРСИЙ ЯДРА LINUX //Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2021). – 2021. – С. 438-442.

6. Сахаров Д. В. и др. Использование математических методов прогнозирования для оценки нагрузки на вычислительную мощность IOT-сети //Научно-аналитический журнал «Вестник Санкт-Петербургского университета Государственной противопожарной службы МЧС России». – 2020. – №. 2. – С. 86-94.

7. Волкогонов В. и др. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ МОНИТОРИНГА СЕТИ ОРГАНИЗАЦИИ НА ОСНОВЕ СИСТЕМЫ ZABBIX.