Ссылка на статью:
// Ученые записки УлГУ. Сер. Математика и информационные технологии. УлГУ. Электрон. журн. 2021, № 2, с. 22-30.

Поступила: 25.11.2021
Окончательный вариант: 30.11.2021

УДК 004.658.2

Особенности настройки клиент-серверного взаимодействия системы 1С

Корниенко Д. В.*[*], *Мишина С. В.*, *Мельников М. О.

[*] dmkornienko@mail.ru
ЕГУ им. И.А. Бунина, Елец, Россия

В статье проанализированы преимущества использования контейнеров при развертывании приложений в сравнении с ручной настройкой каждой рабочей станции по отдельности, которую производит администратор сервера. Рассмотрен инструмент контейнеризации Docker. Приведён процесс создания и запуска docker-контейнера, содержащего сервер 1С. Проанализированы полученные результаты, сделаны соответствующие выводы.

***Ключевые слова**: 1С, docker, контейнеризация, виртуализация.*

Введение

Высокие темпы экономического роста практически всех успешных компаний обусловлены в том числе оперативным внедрением и грамотным использованием информационных технологий. Эффективность работы современного предприятия во многом зависит от уровня автоматизации её бизнес-процессов. Высокая конкуренция на рынке предъявляет бизнесу ряд априорных требований в виде скорости получения, обработки и надёжного хранения актуальных данных. Это послужило причиной полного перехода к компьютеризированному ведению налогового, бухгалтерского, производственного учёта, а также автоматизации хозяйственной деятельности фирм. Внедрение уже зарекомендовавших себя на рынке программных продуктов компании 1С позволяет оптимизировать работу, улучшить эффективность учётной деятельности, а также повысить качество работы предприятия в целом [1].

Платформа 1С предоставляет возможность выбора между двумя вариантами работы системы: клиент-серверный и файловый.

Файловый вариант предполагает работу единственного пользователя или небольшой группы пользователей локальной сети предприятия. При этом все используемые данные хранятся в одной файловой системе управления базой данных (далее СУБД).

Клиент-серверный вариант рассчитан на обслуживание больших рабочих групп. Данный подход базируется на более надёжной трёхуровневой архитектуре [2]. Такая архитектура подразумевает использование трёх отдельных, взаимодействующих между собой компонентов (рис. 1):

– клиентского приложения;
– сервера или кластера серверов 1С: Предприятие;
– сервера базы данных (далее БД).

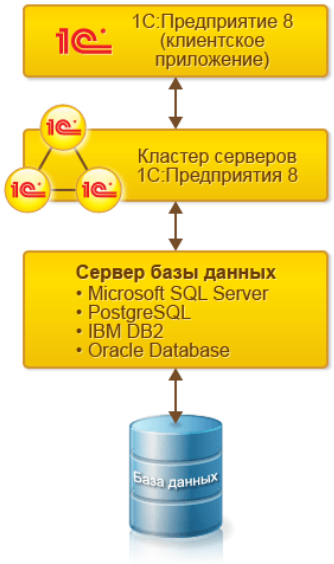

Рис. 1. Архитектура клиент-серверной версии 1С [2]

Сервер 1С представляет собой программное средство, работающее в клиент-серверном режиме с базой данных 1С [3]. Непосредственно взаимодействие происходит через обращение к одной из используемых СУБД (PostgreSQL, Microsoft SQL Server, Oracle DB, IBM Db2). Как правило, сервер 1С применяют крупные и средние организации, в которых доступ к БД требуется сразу большому количеству сотрудников.

Кроме того, использование сервера имеет ряд значительных преимуществ:

– все объёмные и ресурсоёмкие операции происходят на сервере, что даёт значительный прирост быстродействия клиентским приложениям;
– повышение безопасности обусловлено отсутствием у пользователей непосредственного доступа к серверу;
– более гибкое администрирование.

В физическом представлении кластер серверов 1С и сервер БД, как правило, располагаются на разных компьютерах, что позволяет балансировать и распределять входящую

нагрузку. В классическом представлении кластер состоит из нескольких персональных компьютеров (рабочих серверов), на каждом из которых функционирует несколько обслуживающих процессов (рис. 2).

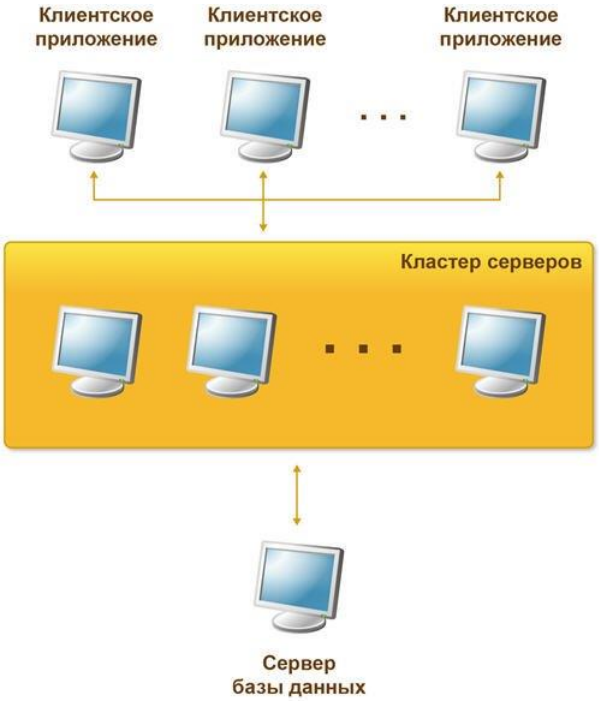

Рис. 2. Клиент-серверная схема работы 1С [2]

Количество рабочих станций в кластере может достигать нескольких десятков. При этом на каждом таком компьютере должен быть установлен и определённым образом сконфигурирован продукт 1С. Для этого администратору системы придётся вручную устанавливать и идентичным образом настраивать программное обеспечение (далее ПО) на каждом компьютере. Это заметно увеличит время развёртывания среды, трудозатраты оператора и повысит шанс появления ошибок. Поэтому для автоматизации развертывания 1С сервера сразу на нескольких станциях или запуска нескольких версий сервера на одном физическом компьютере применяется ряд DevOps-практик, например, контейнерная виртуализация [5]. Поиску способу повышения эффективности конфигурирования рабочих станций и автоматизации ручного развёртывания с помощью контейнерных технологий посвящена данная статья.

1. Виртуализация и контейнеры

Контейнерная виртуализация или контейнеризация – это технология программной (на уровне операционной системы) виртуализации, которая позволяет создавать ряд изолированных единиц пользовательского пространства [4]. В сообществе разработчиков такие единицы принято называть зонами или контейнерами.

Контейнер представляет собой некоторый виртуальный диск-файл, который содержит заранее сконфигурированный определённым образом программный продукт и набор необходимых для его работы зависимостей. В качестве зависимостей могут выступать:

- исходные файлы с программным кодом;
- среда запуска приложения;
- системный инструментарий;
- статические, динамические библиотеки;
- конфигурационные файлы, скрипты и прочее.

Со стороны администратора или клиента контейнер представляется отдельной независимой операционной системой, которая использует ядра и ресурсы основной (хост) операционной системы. Работа приложений в контейнере подобна прямому взаимодействию с хост-системой, при этом обеспечивается полная изолированность содержимого зоны (рис. 3). Это даёт возможность гарантировать отсутствие воздействия друг на друга приложений из разных контейнеров.

Рис. 3. Концепция программной виртуализации

Таким образом, контейнеризация организует собственное виртуальное окружение со своей областью процессов, потоков и стеком вызовов, что делает контейнер аналогом виртуальной машины. При этом в отличие от аппаратной виртуализации создаваемый образ избегает необходимости в эмуляции виртуального оборудования.

Кроме того, подход с использованием контейнеризации обеспечивает ряд преимуществ:

- Легковесность: образы контейнеров занимаю немного пространства на диске.
- Высокая скорость работы за счёт отсутствия накладных расходов на эмуляцию аппаратного окружения.
- Изолированность: отделение от общей инфраструктуры системы позволяет безопасно вносить изменения и совершать обновление продукта.
- Независимость от аппаратного обеспечения.
- Инкапсуляция: всё необходимое для корректной работы приложения уже упаковано в образ.

– Переиспользование: подготовленные образы можно использовать множество раз на различном оборудовании.
– К недостаткам контейнерной виртуализации относят:
– Трудности при менеджменте: каждый образ управляется отдельно, из-за чего при работе с большим количеством зон приходится использовать специализированные инструменты – оркестраторы.
– Привязка к архитектуре управляющей операционной системы: при создании контейнера требуется указать конкретное семейство операционных систем, на которых будет запускаться образ.

Собранный контейнер управляется посредством специализированных контейнерных движков.

Контейнерный движок – это отдельное программное средство, отвечающее за получение и запуск контейнеров, а также организующее взаимодействие с пользователем посредством API-запросов. Наиболее популярными являются движки LXC и OpenVZ, реже используется Solaris Containers, RKT, Railcar.

OpenVZ представляет виртуализацию уровня ядра ОС Linux. Данная среда позволяет запускать несколько копий ОС на одном сервере за счёт организации так называемых «виртуальных частных сред выполнения».

LXC – система изолированного запуска нескольких экземпляров ОС Linux на одном внутреннем узле. Технология основана на использовании Linux cgroups и, как правило, применяется при разработке PaaS-хостингов и создании изолированных динамических образов (dynos).

OpenVZ и LXC используются при необходимости создания практически полноценных экземпляров ОС, однако на их основе разработаны платформы, специализирующиеся на изоляции отдельных сервисов с минимальным операционным окружением. Одной из таких платформ является Docker.

2. Docker

Docker представляет собой контейнеризатор приложений, который позволяет автоматизировать модерирование и развёртывание виртуальных образ-контейнеров [4]. С помощью Docker можно скачивать, создавать, настраивать, запускать контейнеры, управлять их жизненным циклом и прочее. Ранее Docker был доступен только на операционных системах семейства Linux, но с появлением технологии WSL стал полноценно использоваться и в Windows.

Docker-система включает в себя ряд компонентов:

– Docker-host. Устройство, на котором запускается Docker. Хостом может выступать как физическая рабочая станция, так и вириальная машина.
– Docker-daemon. Процесс, ожидающий команды системы и работающий в фоновом режиме. Этот демон отвечает за все действия, связанные с управлением

контейнером. Кроме того, в нём хранится вся информация о контейнерах, загруженных в систему.

- Docker-client. Клиентское приложение, через которое администратор системы взаимодействует с docker-daemon. Поддерживается несколько вариантов интерфейса: API, консольный ввод или полноценное приложение с графическим интерфейсом.
- Docker-image. Сформированный образ, в который помещается набор файлов-зависимостей, необходимых для работы приложения. Из него впоследствии разворачивается контейнер.
- Docker-container. Развёрнутое на основе docker-image и запущенное в изолированной среде приложение.
- Docker-registery. Репозиторий (например, DockerHub), на который загружаются готовые образы (docker-image).
- Dockerfile. Файл-конфигуратор, который содержит «инструкцию» по сборке образа, записанную в декларативном стиле. В файле построчно перечислены все программы, файлы, зависимости и выполняемые команды, которые нужны при построении образа.

Работа с Docker осуществляется с помощью нескольких команд, которые обрабатывает docker-daemon (рис. 4):

- Docker pull – инициирует скачивание docker-image из dcoker-registry.
- Docker build – запускает чтение dockerfile и сборку образа.
- Docker run – превращает собранный docker-image в контейнер и запускает его.

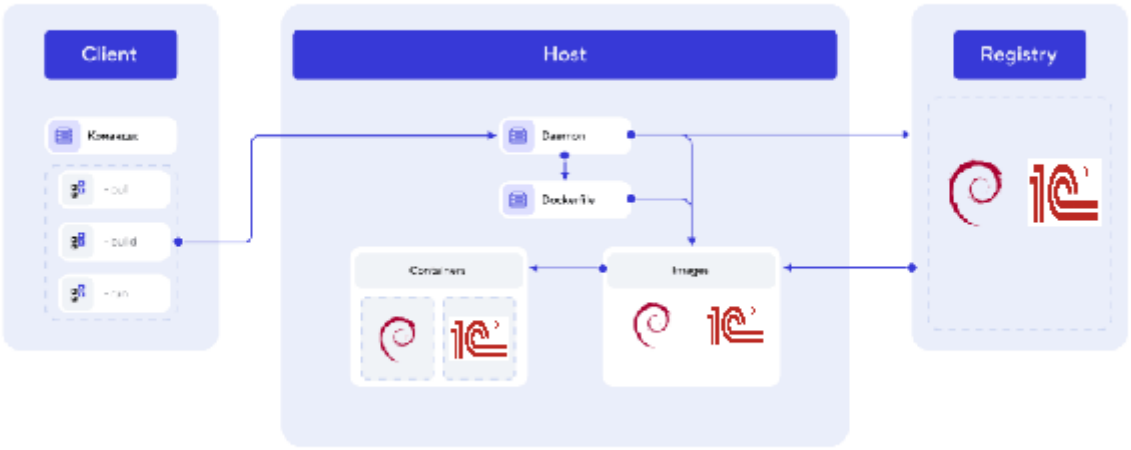

Рис.4. Взаимодействие компонентов Docker

Если возникает необходимость развернуть целый технологический стек, где каждый компонент представлен отдельным контейнером, принято использовать Docker compose.

Docker compose – это инструмент, позволяющий автоматизировать одновременное управление несколькими контейнерами, которые должны входить в состав единого приложения [6]. Определение, настройка и описание взаимодействия сервисов-компонентов происходит в отдельном файле docker-compose.yml.

Команда docker-compose up собирается все контейнеры-компоненты вместе в изолированной среде и запускает развёрнутое составное приложение. При разработке более сложных, масштабируемых систем, как правило, возможностей docker compose уже не хватает [7]. В таком случае приходится использовать более мощный инструмент оркестровки контейнеров – Kubernetes.

3. Создание и запуск Docker-контейнера с сервером 1С

При создании контейнера для развёртывания сервера 1С необходимо иметь:

- msi установщик платформы 1С: Предприятие 8 – 1CEnterprise 83.msi;
- установщик СУБД Microsoft SQL Server – mssqlcli.msi;
- dockerfile с описанием конфигурации – dockerfile;
- powershell скрипт для запуска служб 1С – Wait-Service.ps1;
- powershell скрипт для установки и настройки системы в целом – starup.ps1.

Для начала необходимо расположить все файлы в одной директории. Там же создаётся dockerfile со следующим содержимым:

```
FROM microsoft/windowsservercore
SHELL ["powershell", "-Command", "$ErrorActionPreference = 'Stop'; $ProgressPreference = 'SilentlyContinue';"]
WORKDIR /
COPY startup.ps1 Wait-Service.ps1 1cEnt.zip mssqlncli.msi ./
RUN .\startup.ps1; powershell.exe -Command Remove-Item startup.ps1 -Force
CMD .\Wait-Service.ps1 -ServiceName '1C:Enterprise 8.3 Server Agent' -AllowServiceRestart
```

Строка *FROM microsoft/windowsservercore* указывает основной дистрибутив, на основе которого будет функционировать внутреннее содержимое контейнера. В данном случае это Windows Server Core.

Команда WORKDIR / отвечает за установку рабочего каталога. После этого происходит копирование установочных файлов в рабочую директорию: *COPY startup.ps1 Wait-Service.ps1 1cEnt.zip mssqlncli.msi./*. После чего командой RUN запускается выполнение установочного скрипта statup.ps1. Этот файл содержит следующий powershell-код, отвечающий за установку 1С и настройку соответствующих служб:

```
msiexec /i "1CEnterprise 83.msi" /qr TRANSFORMS=adminstallrelogon.mst;1049.mst DESIGNERALLCLIENTS=0
THICKCLIENT=0 THINCLIENTFILE=0 THINCLIENT=1 WEBSERVEREXT=0 SERVER=1
CON FREPOSSERVER=0 CONVERTER77=0
SERVERCLIENT=0 LANGUAGES=RU
Remove-Item c:\mssqlncli.msi -Force
```

sc.exe config "1C:Enterprise 8.3 Server Agent" depend= "/"

В последней строке dockerfile записывается команда, вызываемая после запуска контейнера: *.\Wait-Service.ps1 -ServiceName '1C:Enterprise 8.3 Server Agent' –AllowServiceRestart.*

После того, как dockerfile полностью сконфигурирован, из текущего каталога запускается команда для построения образа: docker build. Когда построение образа завершено, необходимо проверить его работоспособность. С помощью команды docker container ls возможно узнать id образа (рис. 5).

```
docker images
REPOSITORY                      TAG                 IMAGE ID            CREATED             SIZE
peoples/1c-windows              latest              aac100b82a18        1 days ago          10.2GB
```

Рис. 5. Список docker-образов в системе

Запустить docker-image можно следующим образом: *docker run -d -p 1541:1541 -p 1540:1540 -p 1560-1591:1560-1591 <ID_DOCKER_IMAGE>*.

Полученный docker-image доступен для распространения (через docker-registery или любым другим способом) на рабочие станции предприятия. С его помощью можно автоматически развёртывать систему командой docker run. Запуск, остановка или перезапуск контейнера с сервером 1С происходит через: docker container start/stop/restart <CONTAINER_ID>.

Заключение

Таким образом, автоматизация хозяйственной деятельности большинства предприятий России и стран СНГ уже не представляется возможной без использования ряда программных продуктов фирмы 1С. Клиент-серверный режим работы с платформой является наиболее предпочтительным, безопасным и эффективным вариантом для внедрения в средних и крупных организациях.

Благодаря созданию изолированных образов, инструмент контейнерной виртуализации Docker позволяет организовать автоматическое развёртывание сервера 1С на нескольких физических, виртуальных, облачных станциях или же настроить работу нескольких независимых версий платформы 1С: Предприятие 8 на одном локальном устройстве. Это позволит, единожды сконфигурировав образ на одном компьютере, развернуть всё необходимое, определённым образом настроенное программное окружение на любом количестве рабочих станций в этом кластере.

Таким образом, с помощью применения контейнерной виртуализации мы добиваемся линейного сокращения временных затрат на типовую установку и настройку ПО, уменьшению трудозатрат администратора кластера, и, как следствие, снижения вероятности возникновения ошибок в конфигурации, вызванных человеческим фактором.

Список литературы

1. Корниенко Д.В. *Реализация ведения управленческого учета в 1С:ERP Управление предприятием 2*: учебно-методическое пособие. Елец : ФГБОУ ВО «Елецкий государственный университет им. И.А. Бунина», 2020. 88 с.
2. *1С: Предприятие 8 Система Программ* [Электронный ресурс]. Режим доступа: https://v8.1c.ru/ (дата обращения: 28.10.2021).
3. Хрусталева Е. Ю. *Технологии интеграции 1С:Предприятия 8.3* / Е. Ю. Хрусталева. Москва : ООО «1С-Паблишинг», 2020. – 320 с.
4. Милл И. *Docker на практике* / И. Милл, Э. Хобсон Сейерс . Москва : ДМК Пресс, 2020. 516 с.
5. Мишина, С. В. Примеры реализации механизма криптографии в системе 1С:предприятие 8 / С. В. Мишина // *Continuum. Математика. Информатика. Образование*. 2020, № 4(20), с. 85-93.
6. Моуэт Э. *Использование Docker*. Москва : ДМК Пресс, 2017. 354 с.
7. Артамонов Ю. С. Инструментальное программное обеспечение для разработки и поддержки исполнения приложений научных вычислений в кластерных системах / Артамонов Ю. С., Востокин С. В. // *Вестник Самарского государственного технического университета*. 2015, № 4, с. 785-798.

Features of setting up client-server interaction of the 1C system

Kornienko, D. V.**, Mishina, S. V., Melnikov, M. O.

* dmkornienko@mail.ru

I. A. Bunin Elets State University, Elets, Russia

The paper analyzes the benefits of using containers in deployment Applications in comparison with the manual configuration of each workstation separately, which is performed by the server administrator. The Docker containerization tool is considered. The process of creating and launching a docker container including a 1C server is presented.

Keywords: *1C, docker, containerization, virtualization.*