

Виртуалка и Docker

Что такое VPS мы уже знаем. Давайте расставим все точки над ё:

1. Обычная программа на CPU.

Программа обычно исполняется на центральном процессоре. Программа, как мы знаем из урока по машине Тьюринга, — это последовательность команд ЦП.

Вот таких:

```
SECTION .text
org 0x100
mov ah, 0x9
mov dx, hello
int 0x21
mov ax, 0x4c00
int 0x21
SECTION .data hello: db "Hello, world!", 0xD, 0xA, '$'
```

Это всем программам программа. Выводит «Hello, world» на экран.

Куча действий, не правда ли?

Это ассемблер. Мы такое даже трогать не будем. Жуть жуткая почти для всех.

И так работали все программы на компьютере, да и сейчас работают. Делят машинное время. Сначала одна выполнится немного, потом другая и так далее. И кажется, что идет параллельная обработка и одновременно куча всего делается.

Правда, одновременно исполняется столько потоков, сколько ядер у вашего процессора.

2. Виртуалка.

Когда мы делаем виртуалку, то специальная программа транслирует код программы, которая как бы должна выполняться на виртуальном процессоре в код, который будет выполняться на настоящем. Т.е., есть программная прослойка, которая делает виртуализацию.

Например, вам нужно сделать расчеты на калькуляторе, но с собой его нет. Зато есть Василий, который в 10 метрах от вас. У него есть заветный вычислитель. И Вы ему кричите «27:3», он такой «27 на 3?», вы «Да», он через минуту дает ответ, взяв паузу на набор примера багровыми пальцами, «9».

Сам бы давно посчитал, но калькулятора нет, мы создали себе виртуальный калькулятор и он работает. Круто, что если Василий сгинет, то Пётр подоспеет и вычисления не остановятся. Это круче, чем иметь свой калькулятор в кармане, хоть и медленнее.

Но для виртуализации калькулятора нам нужно иметь абсолютно целого Василия или Петра. Не только ту его часть, которая кричит и считает, типа «орала», ушей и пальцев, но и спины, ног и прочего, потому что Василий по отдельности не работает.

Накладно как-то. Кормить надо.

С виртуалками такая же история. Круто, но ресурсов жрется много, во время простоя виртуалки (когда нет заданий) всё равно она молотит, нужно дисковое пространство в пару десятков Гб, чтобы гостевая ОС работала нормально. Ну и ставить нужно всё с нуля, слава Богу, хоть ОС ставится сама у хостеров. А всё остальное от олдскула ничем не отличается.

Был у меня на старой работе коллега-админ, лет 50-55, Лёша, по прозвищу «Самоделкин». Вот он мог скурить сигарету за 5 тяг. Это единственный крутой навык, который он демонстрировал. Сервер новый делал 2-3 дня, перенос СУБД с одного кластера на другой с переездом баз, это вообще неделя. Ну нельзя же так жить, говорили ему. А он потягивал дым в усы и говорил «разбираемся». И многие не хотели быть зависимыми от самоделкиных и придумали технологию Docker.

3. Docker.

Это тоже вариант виртуализации, однако более умный. Единица виртуализации у нас называется контейнер. В чем отличия от простой виртуалки?

- Процессы исполняются на центральном процессоре, т.е., не нужна вычислительная прокладка. Т.е., работают на 10-15% быстрее виртуалок при тех же параметрах. Если рабочая нагрузка 0, то ресурс не жретя.
- Не нужно много дискового пространства, т.к. в контейнер можно/нужно подмонтировать папку сервера-хоста. Заодно она может служить местом обмена файлами с внешним миром.
- Порт контейнера может быть автоматически (без бубна и настроек файрвола) проброшен на любой другой порт хоста. Т.е., скажем, есть контейнер с веб-сервером, который отвечает по порту 8080, можно легко настроить так, чтобы при обращении к серверу-хосту по порту 80, он перенаправлял запрос на порт 8080 нужного контейнера.
- Каждый контейнер может быть легко унаследован от другого. Бабка за дедку. Дедка за репку. Хоп, глядишь, а внучка-то у нас частично репка. То, что она частично дед и бабка, надеюсь, сомнений нет.
- Ведь круто создать свой контейнер парой команд, не тратя время на установку и настройку всего окружения. Если словами, то так: пусть мой контейнер будет наследником от Ubuntu 20.04, я ставлю туда python3 и пару других пакетов. Заливаю свою программу и делаю скрипт, который ее запускает. Теперь для того чтобы запустить мой контейнер где-то далеко, достаточно выполнить команду `docker run` и имя моего контейнера. Помните, как долго надо было возиться с новым сервером, чтобы наконец перенести на него свой любимый старый сайт...
- Есть центральный репозиторий контейнеров типа github, только называется dockerhub. Можно одной командой запустить у себя, скажем, сервер postgresql. Образ скачается и запустится. Вот она, админская кнопка «бабло».
- Контейнеры можно клонировать, можно запускать и останавливать, переносить и пр.
- Из контейнеров можно собирать «рой» и управлять ими централизованно.

Ах да. Эти **все штуки придумали в Google** и выложили для нас в открытый доступ.

Задание

Прочитать про докер.

Найти в документации, как поставить докер на свою машину.



Поставить докер.

Найти что такое Portainer

Поставить Portainer на свою машину, войти, посмотреть что она может.