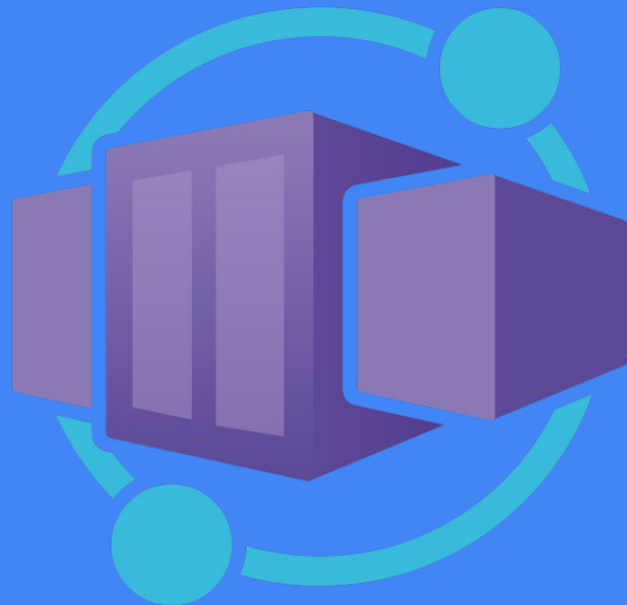


Механизмы контрольных групп

Урок 2
Linux: cgroups и «запуск контейнера» без
использования Docker





Основы контейнеризации

1

Лекция 1: Механизмы пространства имен

2

Семинар 1: Механизмы пространства имен

3

Лекция 2: Механизмы контрольных групп

4

Семинар 2: Механизмы контрольных групп

5

Лекция 3: Введение в Docker

6

Семинар 3: Введение в Docker

7

Лекция 4: Dockerfiles и слои

8

Семинар 4: Dockerfiles и слои

9

Лекция 5: Docker Compose и Docker Swarm

10

Семинар 5: Docker Compose и Docker Swarm

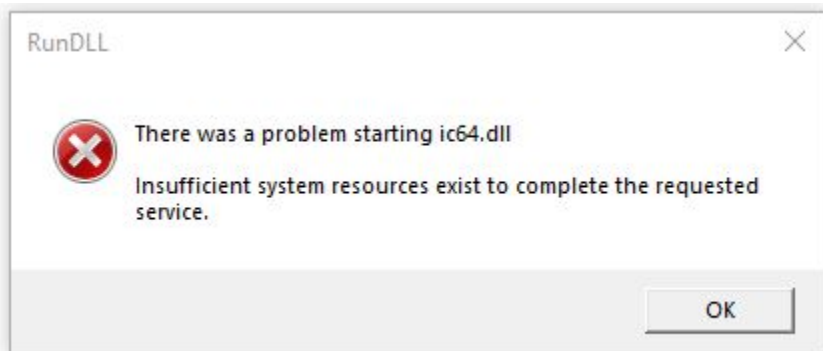


Что будет на уроке сегодня

- 📌 cgroups – появление механизма процесс модификации
- 📌 Архитектура и составляющие механизма
- 📌 Примеры управления группами
- 📌 Недостатки cgroups

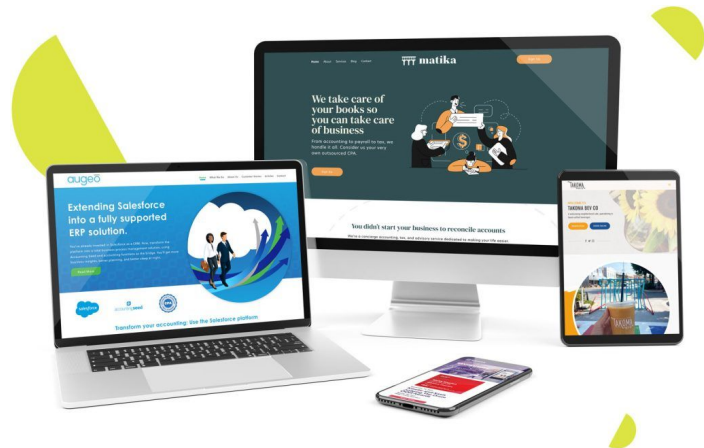


cgroup — это механизм для иерархической организации процессов и распределения системных ресурсов



Ограничения и ресурсы

Пример



Пример

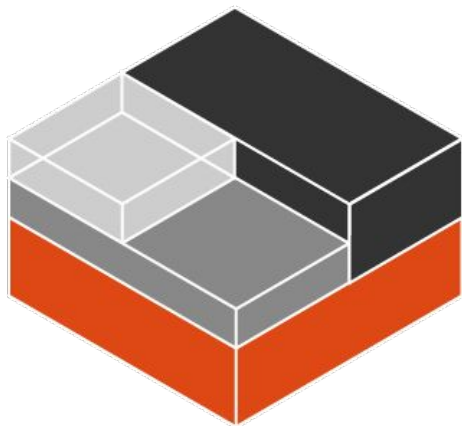




Историческая справка



LXC подсистема контейнеризации



LXC подсистема контейнеризации

Возможности:

- Ограничение ресурсов
- Приоритизация
- Регистрация затрат тех или иных ресурсов приложением либо группой приложений
- Изоляция
- Управление

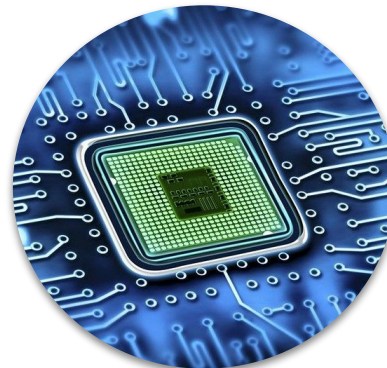
Состав компонента

Модули:

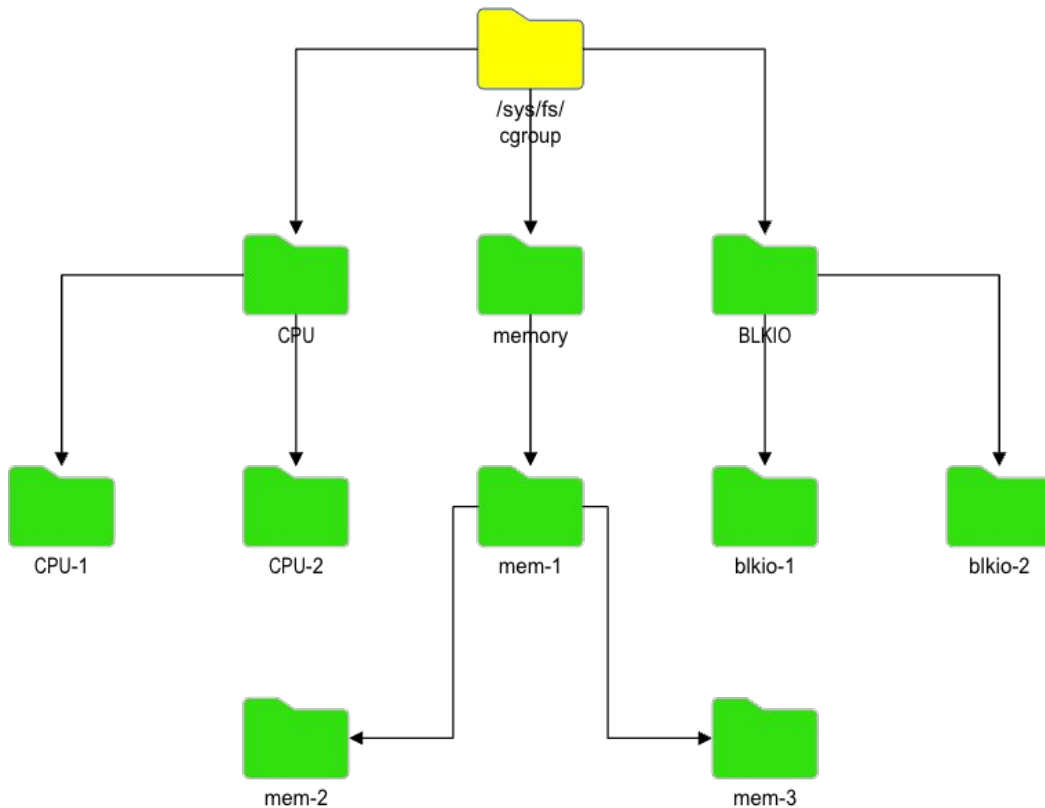
- blkio
- cpuacct
- cpu
- cpuset
- devices
- freezer

- memory
- net_cls
- netprio
- ns
- pids
- unified

Ядро



Подсистемы





Управляющие файлы



cgroup.clone_children

Передача дочерним группам свойства родительских



tasks

Список PID всех процессов группы



cgroup.procs

Список TGID групп процессов



cgroup.event_control

Отправка уведомлений в случае изменения статуса



notify_on_release

Включение/отключение выполнения команды



Псевдофайлы - это файлы, которые автоматически создаются для представления какого-либо другого объекта в виде файла с целью возможности взаимодействия с ним.



blkio - это подсистема управления
процедурами ввода/вывода
блочных устройств



Параметры blkio



blkio.weight

Определяет относительный вес для операций ввода/вывода

```
1 $ echo 200 > blkio.weight
2 $ cat blkio.weight
3 200
4
```



blkio.weight_device

Определяет параметр для конкретного устройства

```
1 $ echo 8:0 200 > blkio.weight_device
2 $ cat blkio.weight_device
3 8:0 200
```




blkio.weight

```
1 $ echo 8:0 200 > blkio.weight_device
2 $ cat blkio.weight_device
3 8:0 200
```

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINTS
sda	8:0	0	80G	0	disk	
└─sda1	8:1	0	1M	0	part	
└─sda2	8:2	0	2G	0	part	/boot
└─sda3	8:3	0	78G	0	part	
└─└─ubuntu--vg-ubuntu--lv	253:0	0	39G	0	lvm	/



сри - отвечает за управление
доступом контрольных групп к
процессорам системы



Параметры cpu



cpu.shares

Определяет относительный вес для операций ввода/вывод



cpu_rt_runtime_us

Определяет параметр для конкретного устройства



cpu._rt_period_us

Определяет относительный вес для операций ввода/вывод



Пример использования:

`cpu.rt_runtime_us 4000000`

`cpu.rt_period_us 10000000`



criact - создает отчеты о занятости
ресурсов процессора



Параметры cpuact



cpuact.stat

Возвращает число циклов процессора, затраченных на обработку заданий



cpuact.usage

Возвращает суммарное время, затраченное на обработку заданий группы



cpuact.usage_percpu

Попроцессорный возврат затраченного времени на обработку заданий



cgroup отвечает за выделение
ресурсов процессора контрольным
группам



Параметры cpuset



cpuset.cpus

Определяет количество доступных процессоров



cpuset.cpu_exclusive

Определение возможности совместного использования процессоров



devices отвечает за управление
доступом к устройствам



Параметры devices



devices.allow

Описывает устройства, к которым разрешен доступ в рамках cgroup

Содержит в себе:

1. Типы устройств:

- a - применяется ко всем символьным и блочным устройствам
- b - блочное устройство
- c - символьное устройство (ссылка на устройство, на файл)

2. Старший и младшие номера

```
NAME                                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda                                8:0    0   80G  0 disk
├─sda1                             8:1    0    1M  0 part
├─sda2                             8:2    0    2G  0 part /boot
├─sda3                             8:3    0   78G  0 part
└─ubuntu--vg-ubuntu--lv 253:0    0   39G  0 lvm  /
```

3. Режим доступа

- r - доступ разрешен только на чтение
- w - доступ разрешен на чтение и запись
- m - разрешение доступа на создание файлов, если они не существуют



Параметры **devices**



devices.deny

Описывает устройства, к которым запрещен доступ в рамках cgroup



devices.list

Описывает устройства, к которым настроено управление доступом



freezer отвечает за остановку и
возобновление заданий
контрольной группы



Параметры freezer



freezer.state

Описывает статус подсистемы

Доступны следующие значения:

- frozen - задания приостановлены
- freezing - система в стадии приостановки задач
- thawed - возобновление работы заданий в группе



memory создает отчеты об
использовании ресурсов памяти



Параметры memory



memory.stat

Статистика использования памяти



memory.usage_in_bytes

Занятая память процессами из выбранной cgroup



memory.max_usage_in_bytes

Доступный объем памяти (без файла подкачки)



memory.limit_in_bytes

Доступный объем памяти (с файлом подкачки)



memory.failcnt

Счетчик достижения лимита памяти



С помощью `net_cls` можно
идентифицировать пакеты,
поступающие из контрольной
группы



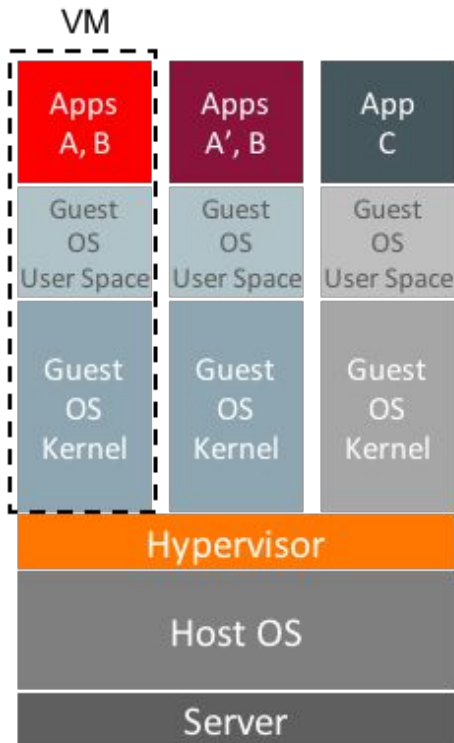
`net_cls.classid`



Контейнеры и LXC



Механизм LXC



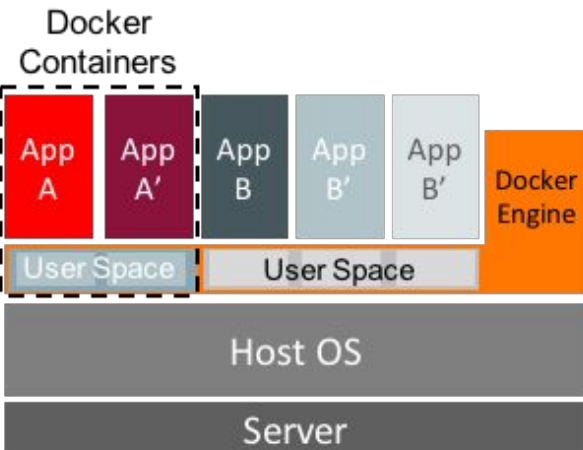
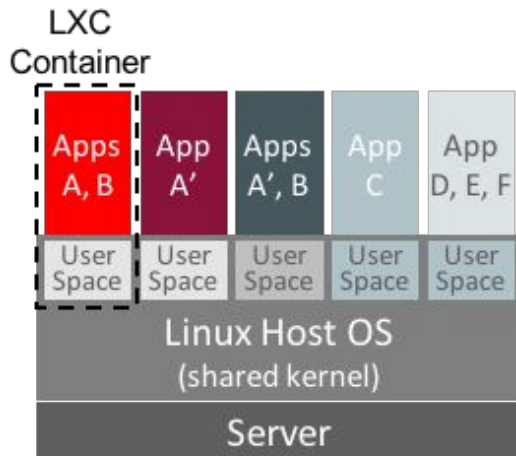
Простота и наследственность

Переиспользование сервиса, портативность



Схожесть с виртуальными машинами, но потребление, как у Docker

Легковесные образы и похожий механизм контейнеризации









Итоги занятия



Подведем итоги

-  Рассмотрели механизм контрольных групп (cgroups)
-  Изучили его архитектуру и составляющие
-  Рассмотрели примеры управления группами
-  Обозначили недостатки контрольных групп первого поколения