Менеджер контейнерів islander

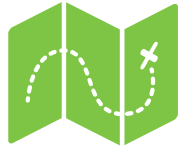
Денис Герасимук, Ярослав Морозевич, Дмитро Лопушанський

Суть проекту

Розробити аналог docker'а, який зможе запускати процеси в повністю ізольованих середовищах. **islander матиме такі функції:**

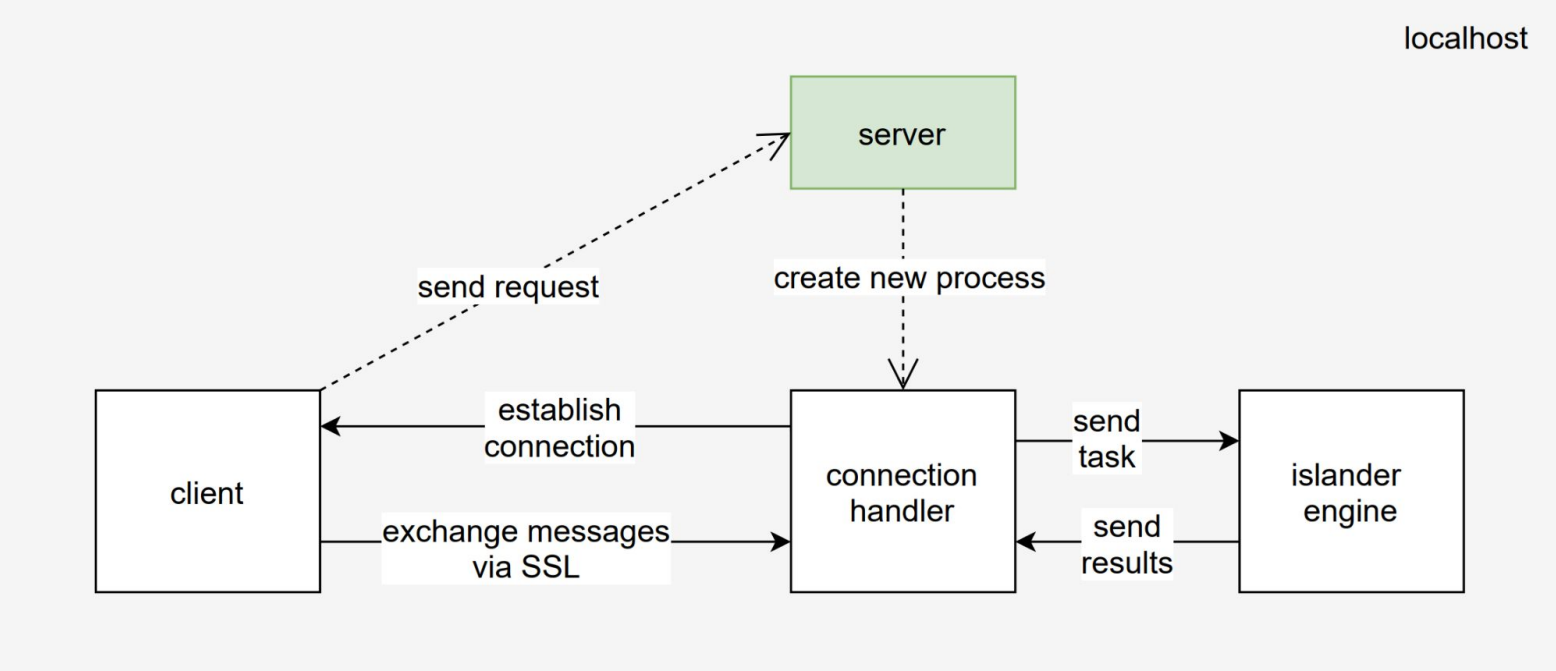
- Обмеження використання файлової системи, процесорної завантаженості, пам'яті, мережі
- Налаштування cgroups і створення namespace'ів
- client-server архітектура. CLI парсер та демон-процес

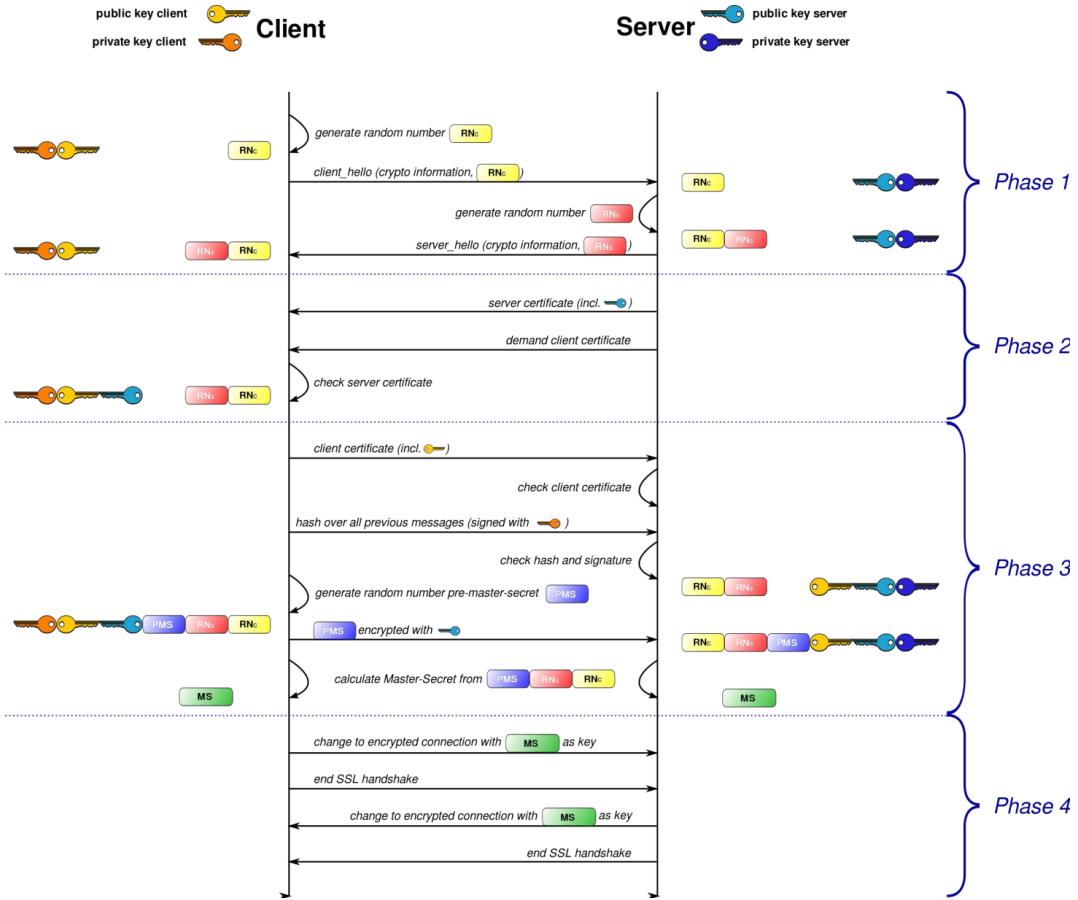
Етапи розробки



1. Етап дослідження: Технології, функціонал, деталі реалізації.
2. Розробка скриптів, які зможуть ізолювати процеси за певними параметрами
3. Написання парсера і сервера, які будуть спілкуватися через сокети
4. Поєднання всіх частин проекту
5. Підтримка Volumes, Bind Mount, TmpFS Менеджмент даних контейнера
6. Додавання network namespace, менеджмент контейнерів
7. Покращення взаємодії клієнтів та сервера; робота з новими типами програм
8. Покриття коду тестами, додавання нових фіч, загальне покращення

Архітектура проекту



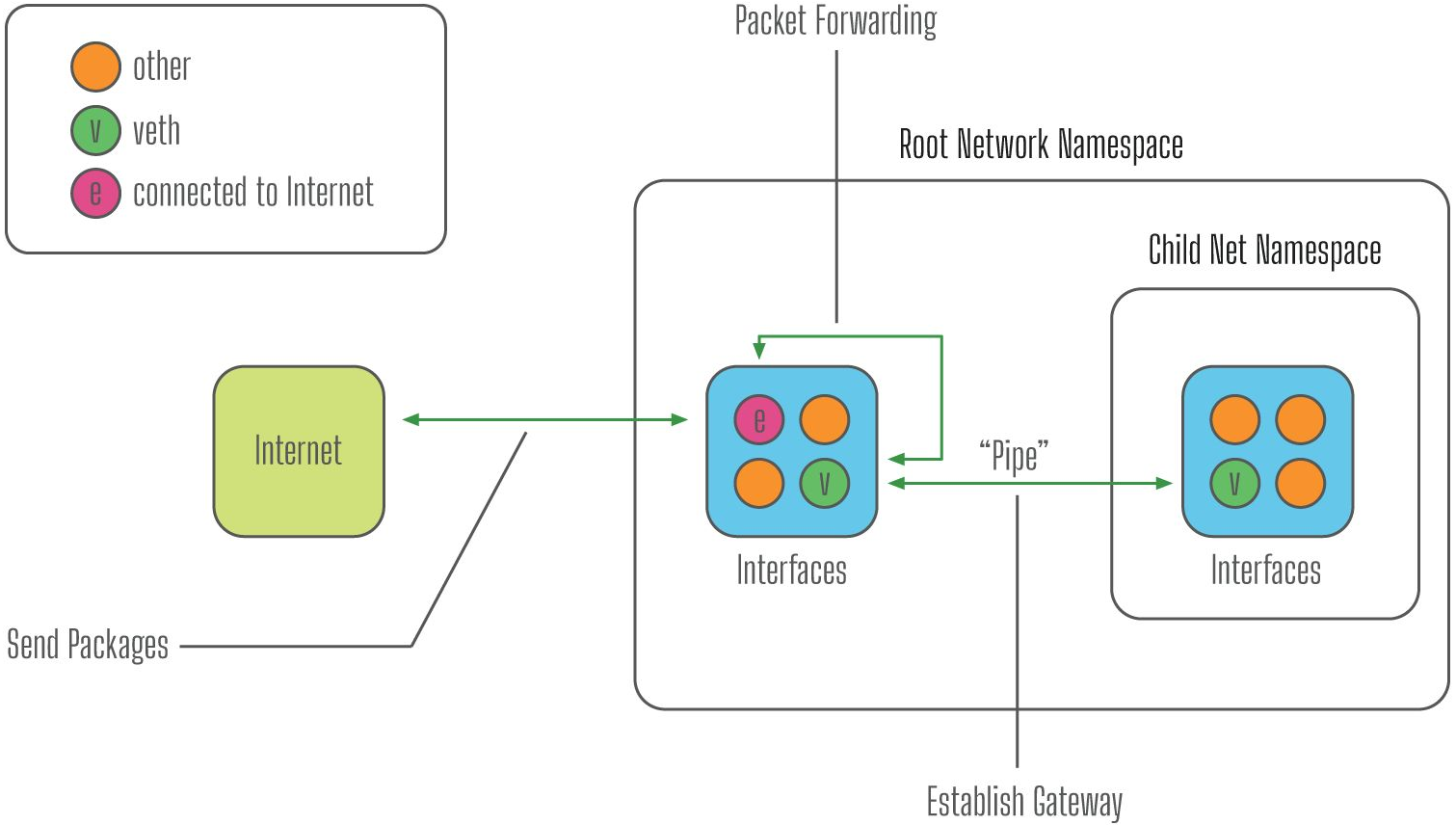

public key client
private key client
Client
Server
public key server
private key server
generate random number
client_hello (crypto information,
generate random number
server_hello (crypto information,
server certificate (incl.
demand client certificate
check server certificate
client certificate (incl.
check client certificate
hash over all previous messages (signed with
check hash and signature
generate random number pre-master-secret
encrypted with
calculate Master-Secret from
change to encrypted connection with
as key
end SSL handshake
change to encrypted connection with
as key
end SSL handshake
Phase 1
Phase 2
Phase 3
Phase 4

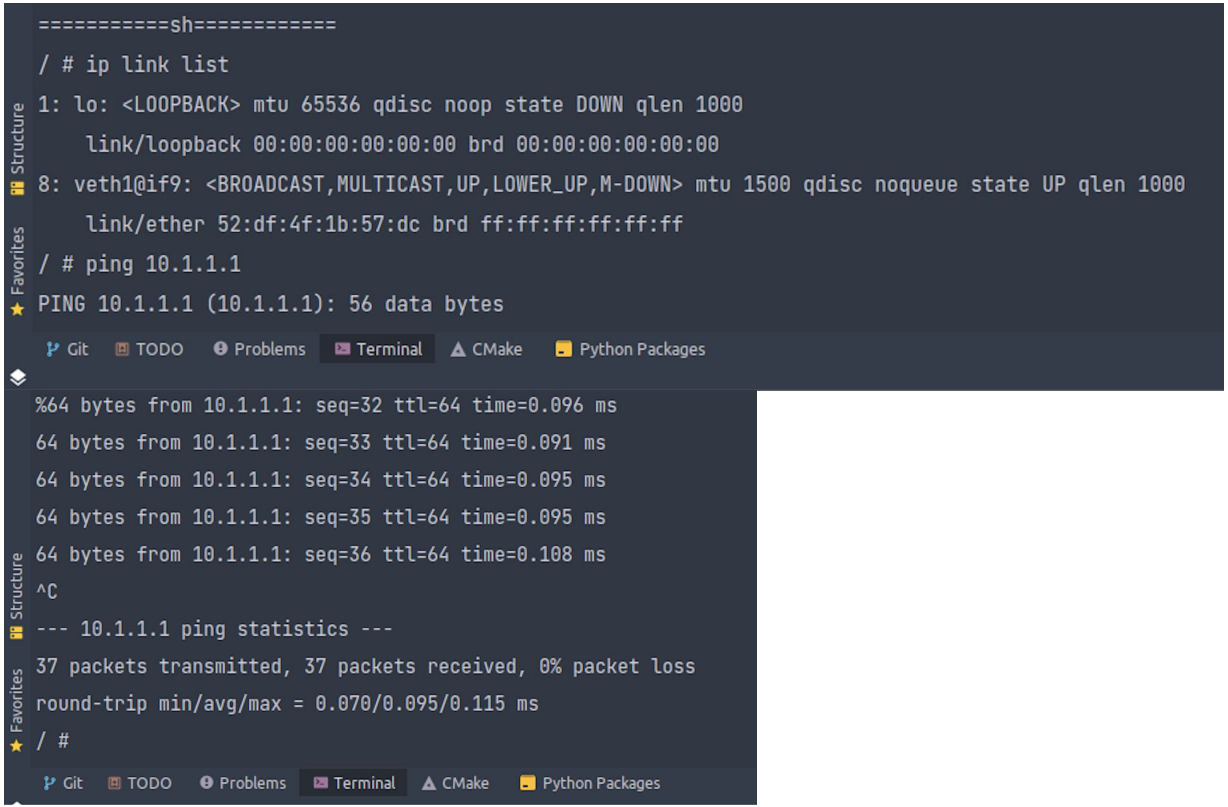
Що таке namespace?

- Ізоляційний механізм для ресурсів
- Забезпечує відображення ресурсів зі змінами дозволів
- Зміни до процесів, які знаходяться в певному просторі імен, є невидимі поза його межами

Види namespace'ів

- **Mount** - керує точками монтування
- **Network** - керує мережевим стеком
- **PID** - надає процесам незалежний набір id
- **UTS** - дозволяє одній системі мати різні імена хостів/доменів
- **User Namespace** - забезпечує ізоляцію привілеїв користувача
- **IPC** - забезпечує комунікацію між процесами


other
v
veth
e
connected to Internet
Packet Forwarding
Root Network Namespace
Child Net Namespace
Internet
e
v
Interfaces
"Pipe"
v
Interfaces
Send Packages
Establish Gateway


==========sh==========
/ # ip link list
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
8: veth1@if9: <BROADCAST,MULTICAST,UP,LOWER_UP,M-DOWN> mtu 1500 qdisc noqueue state UP qlen 1000
link/ether 52:df:4f:1b:57:dc brd ff:ff:ff:ff:ff:ff
/ # ping 10.1.1.1
PING 10.1.1.1 (10.1.1.1): 56 data bytes
Git
TODO
Problems
Terminal
CMake
Python Packages
Structure
Favorites
%64 bytes from 10.1.1.1: seq=32 ttl=64 time=0.096 ms
64 bytes from 10.1.1.1: seq=33 ttl=64 time=0.091 ms
64 bytes from 10.1.1.1: seq=34 ttl=64 time=0.095 ms
64 bytes from 10.1.1.1: seq=35 ttl=64 time=0.095 ms
64 bytes from 10.1.1.1: seq=36 ttl=64 time=0.108 ms
^C
--- 10.1.1.1 ping statistics ---
37 packets transmitted, 37 packets received, 0% packet loss
round-trip min/avg/max = 0.070/0.095/0.115 ms
/ #
Git
TODO
Problems
Terminal
CMake
Python Packages
Structure
Favorites

Що таке Cgroup?

- **Namespace** — обмежує привілеї процесу

  Cgroup — ставить ліміти та обмежує типи ресурсів

- Дозволяють розподіляти ресурси серед визначених груп процесів

Cgroup subsystems

- **blkio** - читання та запис блочних девайсів
- **cpu** - доступ до процесора
- **devices** - доступ до девайсів
- **net_cls** - ліміти network io
- **memory** - RAM ліміти для cgroup

```
$ ls /sys/fs/cgroup/

blkio     cpu,cpuacct  freezer  net_cls          perf_event
cpu       cpuset       hugetlb  net_cls,net_prio pids
cpuacct   devices      memory   net_prio         systemd
```

Приклад використання

 # Create a group

$ cd /sys/fs/cgroup

$ mkdir -p memory/group1

 # Set a memory limit of 150M

$ echo 150M > memory/group1/memory.limit_in_bytes

 # Add shell to group

$ echo $$ > memory/group1/tasks

Islander Data Management

Проблеми:

- Дані не зберігаються, коли цього контейнера більше не існує
- Спільний доступ до даних
- high-performance I/O тощо

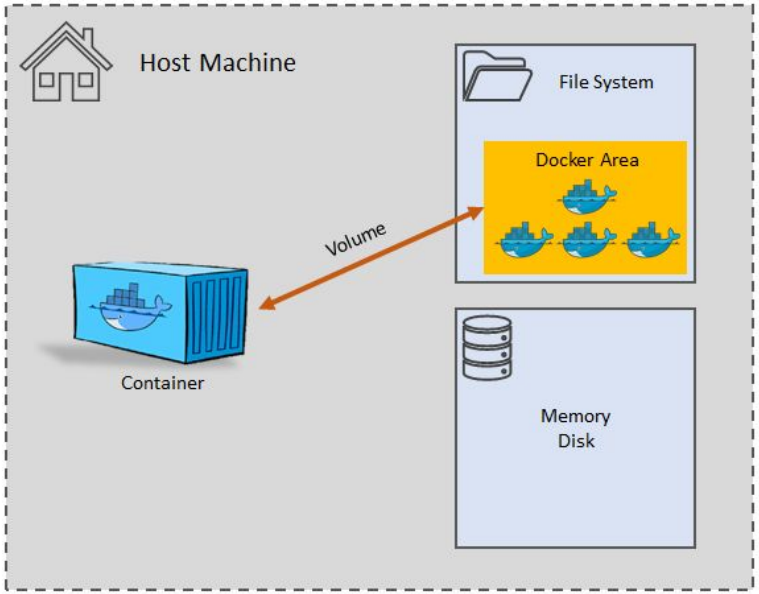
Рішення:

- Islander volumes
- Bind mounts
- Tmpfs mounts

Islander Volumes

Особливості:

- Більша ефективність, ніж у mount
- Легше створити backup або перемістити
- Взаємодія з volumes на віддалених хостах або хмарних провайдерах

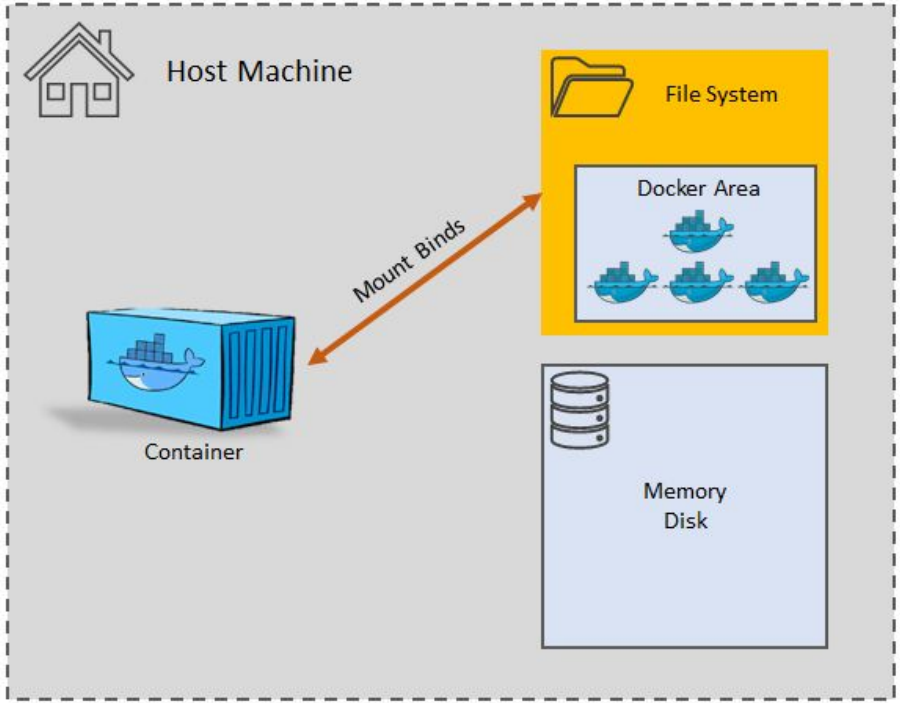


Btrfs or B-tree

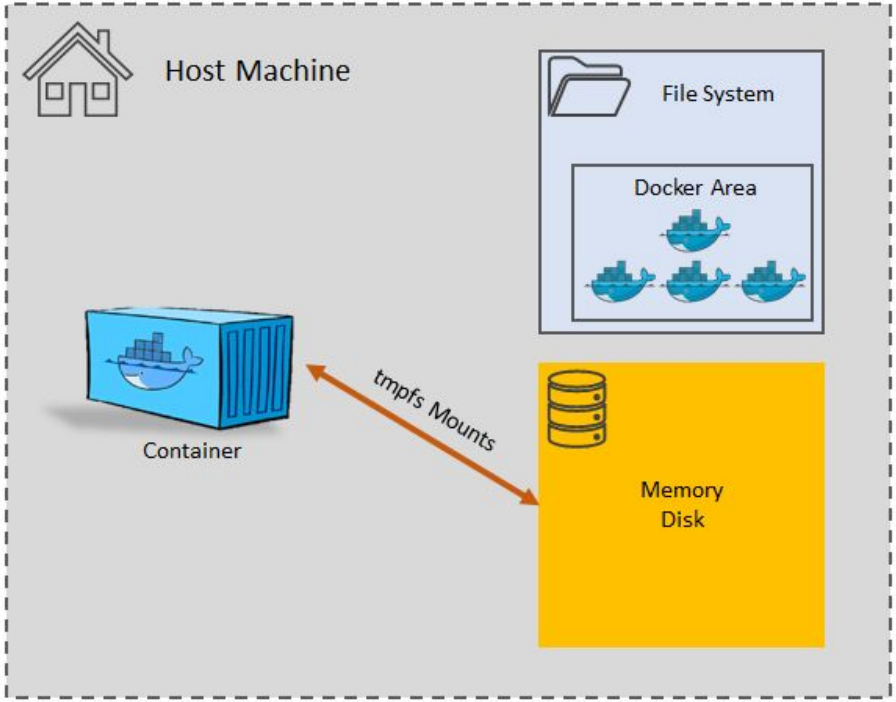
Особливості:

- Створення snapshots
- Використовує алгоритми компресії даних на рівні файлової системи
- Використовує контрольну суму CRC32C → цілісність даних і уникнути пошкодження даних
- Оптимізована підтримка SSD-дисків тощо

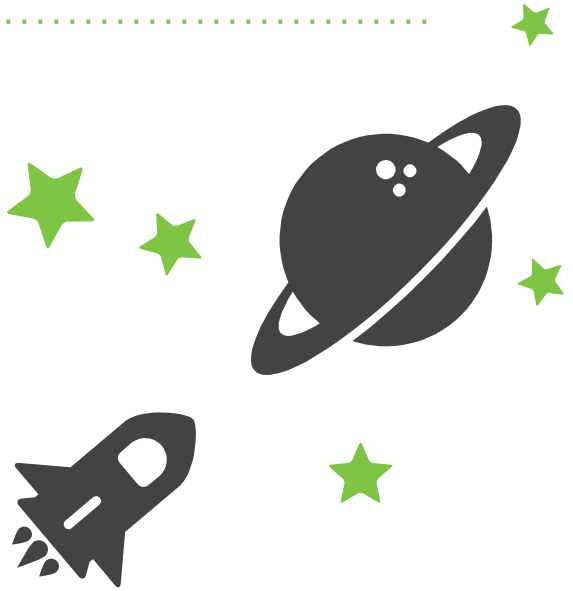
Bind Mounts


Host Machine
File System
Docker Area
Mount Binds
Container
Memory
Disk

Tmpfs Mounts



Демо



Дякуємо!

Час для запитань

https://github.com/denysgerasymuk799/UCU_OS_Course_Project

Модель Cgroup

- Подібні до процесів
  - ієрархічні
  - дочірні cgroups успадковують певні атрибути від батьківської cgroup

- Відмінне:
  - Linux є єдиним деревом процесів
  - модель cgroup — одне або кілька окремих, не пов'язаних між собою дерев процесів

Модель Cgroup