2 ИЮНЯ 2026, МОСКВА, ЛОФТ ГОЭЛРО

БЕКОН LUNTRY '26

ЕДИНСТВЕННАЯ КОНФЕРЕНЦИЯ ПО БЕЗОПАСНОСТИ КОНТЕЙНЕРОВ И КОНТЕЙНЕРНЫХ СРЕД

Контейнеры под защитой PARSEC: пополнение в рядах AppArmor, SELinux

Антон Баранов | ГК АСТРА

БЕКОН'26

Контейнеры под защитой PARSEC: пополнение в рядах AppArmor и SELinux

Антон Баранов

Платформа Боцман

Обо мне

БЕКОН'26

- Ведущий разработчик платформы Боцман в ГК «Астра»
- 10 лет коммерческой разработки на C++ и Golang
- Чем занимаюсь сейчас: Kubernetes, сетевые политики и безопасность контейнерных рантаймов.
- Выступал на конференции БеКон 2025 с докладом “Расширение политик фильтрации трафика в Cilium”
- Люблю глубоко разбираться в технологиях и делиться знаниями.



План

БЕКОН'26

01 Безопасность контейнеров

02 PARSEC: МКЦ, МРД и ЗПС

03 МКЦ и МРД в Kubernetes

04 Containerd

05 Runc

06 На практике

07 Итоги

Безопасность контейнеров

01

Безопасность контейнеров

БЕКОН'26

Уровни

- Уровень образа, реестра
- Уровень среды выполнения
- Уровень оркестратора
- Уровень хостовой ОС
- Уровень сети
- Уровень приложения

Безопасность контейнеров

БЕКОН'26

Уровни

- Уровень образа, реестра
- Уровень среды выполнения
- Уровень оркестратора
- Уровень хостовой ОС
- Уровень сети
- Уровень приложения

Защита

- Сканирование образов
- Runtime-защита и обнаружение аномалий
- Admission-контроллеры
- Сетевые политики
- Управление секретами
- Ограничение возможностей самого контейнера: системы мандатного контроля доступа: **AppArmor** и **SELinux**.

MAC для контейнеров в K8S: AppArmor

БЕКОН'26

AppArmor

- Принцип работы — path-based
- Профили могут работать в двух режимах: enforce, complain
- В Kubernetes профиль назначается Pod через securityContext

```
spec:
 securityContext:
   appArmorProfile:
     type: Localhost
     localhostProfile: k8s-nginx
```

Пример профиля

```
#include <tunables/global>

profile k8s-nginx flags=(attach_disconnected) {
     #include <abstractions/base>

     network inet tcp,
     file,

     # запрещаем запись в исполняемые пути
     deny /bin/** w,
     deny /sbin/** w,
     deny /usr/bin/** w,
}
```

MAC для контейнеров в K8S: SELinux

БЕКОН'26

SELinux

- Работает с метками (на файл, процесс, сокет)
- Политика описывает взаимодействие типов (Type Enforcement)
- Поддерживает многоуровневую безопасность и категории

```
spec:
 securityContext:
   seLinuxOptions:
     type: my_container_t
```

Пример модуля

```
policy_module(my_container, 1.0)

require {
    type container_t;
    type http_port_t;
}

# свой домен для контейнера
type my_container_t;

# разрешаем слушать только HTTP-порт
allow my_container_t http_port_t:tcp_socket
name_bind;
```

PARSEC

02

PARSEC

БЕКОН'26

СЗИ в Astra Linux

Средство Защиты Информации, интегрированное в ядро операционной системы для обеспечения максимальной безопасности.

Сертификация ФСТЭК

Подтвержденный 1-й уровень доверия, позволяющий использовать систему в самых критически важных инфраструктурах.

Научная база

В основе лежит научно обоснованная подсистема безопасности, гарантирующая математическую корректность моделей защиты.

МРД

Мандатное управление доступом.

Разграничение прав на основе меток конфиденциальности субъектов и объектов.

МКЦ

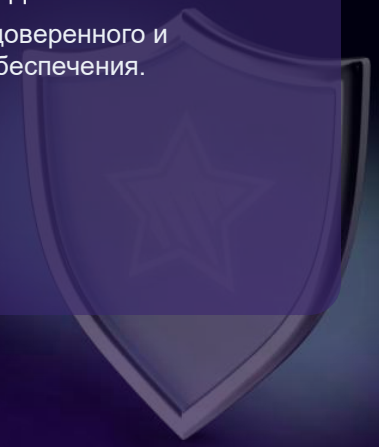
Мандатный контроль целостности.

Предотвращение модификации критических данных низкоцелостными процессами.

ЗПС

Замкнутая программная среда.

Контроль исполнения только доверенного и подписанного программного обеспечения.



Метка безопасности

```
<КОНФИДЕНЦИАЛЬНОСТЬ> : <ЦЕЛОСТНОСТЬ> : <КАТЕГОРИИ> : <ФЛАГИ> : <ЛИН. ЦЕЛОСТНОСТЬ>
```

Метка безопасности

```
<КОНФИДЕНЦИАЛЬНОСТЬ> : <ЦЕЛОСТНОСТЬ> : <КАТЕГОРИИ> : <ФЛАГИ> : <ЛИН. ЦЕЛОСТНОСТЬ>
```

Мандатный Контроль Целостности (МКЦ)

- Процесс пишет только в файлы со своим уровнем целостности или ниже
- Записать в файл с более высокой целостностью — запрещено
- Высокоцелостный процесс не сможет запустить низкоцелостный исполняемый файл
- Не участвует в сетевом взаимодействии

Метка безопасности

```
<КОНФИДЕНЦИАЛЬНОСТЬ> : <ЦЕЛОСТНОСТЬ> : <КАТЕГОРИИ> : <ФЛАГИ> : <ЛИН. ЦЕЛОСТНОСТЬ>
```

Мандатный Контроль Целостности (МКЦ)

- Процесс пишет только в файлы со своим уровнем целостности или ниже
- Записать в файл с более высокой целостностью — запрещено
- Высокоцелостный процесс не сможет запустить низкоцелостный исполняемый файл
- Не участвует в сетевом взаимодействии

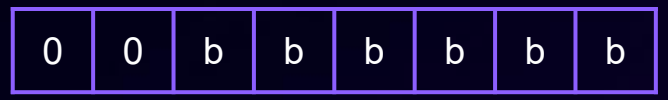
Мандатное Разграничение Доступа (МРД)

- Процесс читает файл, если его уровень больше или равен уровню файла
- Процесс пишет в файл, если его уровень равен уровню файла
- Категории — 64-битная маска: доступ только при наличии категорий
- Участвует в сетевом взаимодействии

PARSEC: Целостность

БЕКОН'26

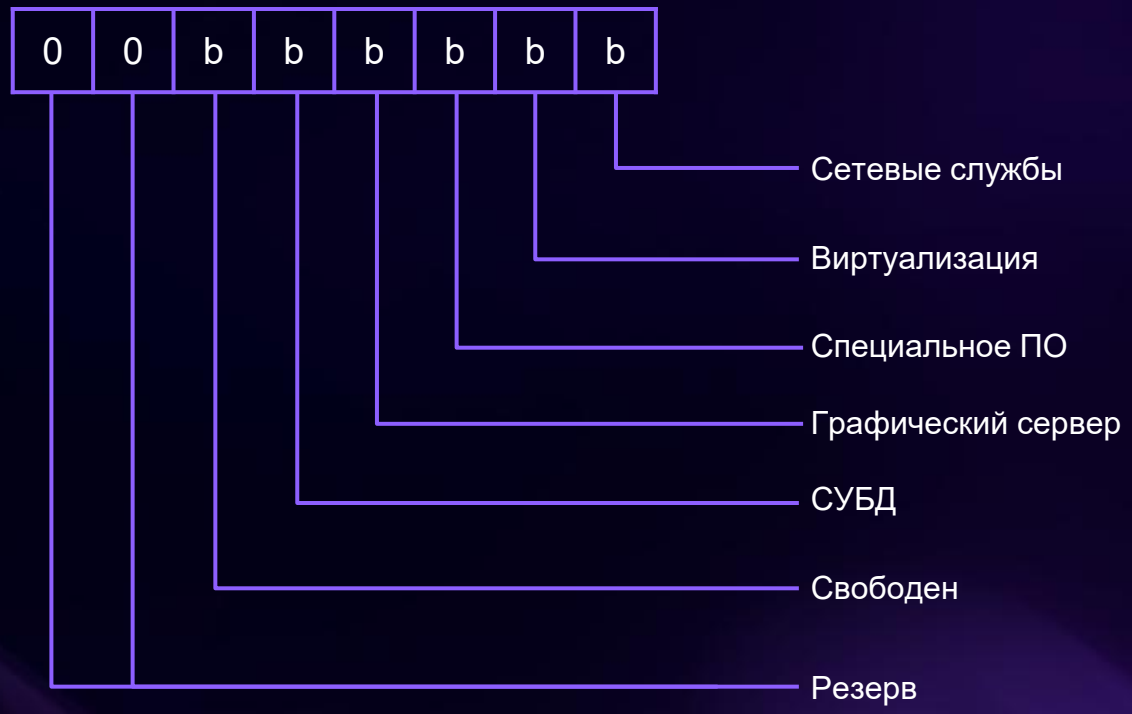
Уровень целостности представляет собой маску, состоящую из набора единичных значений уровней целостности



PARSEC: Целостность

БЕКОН'26

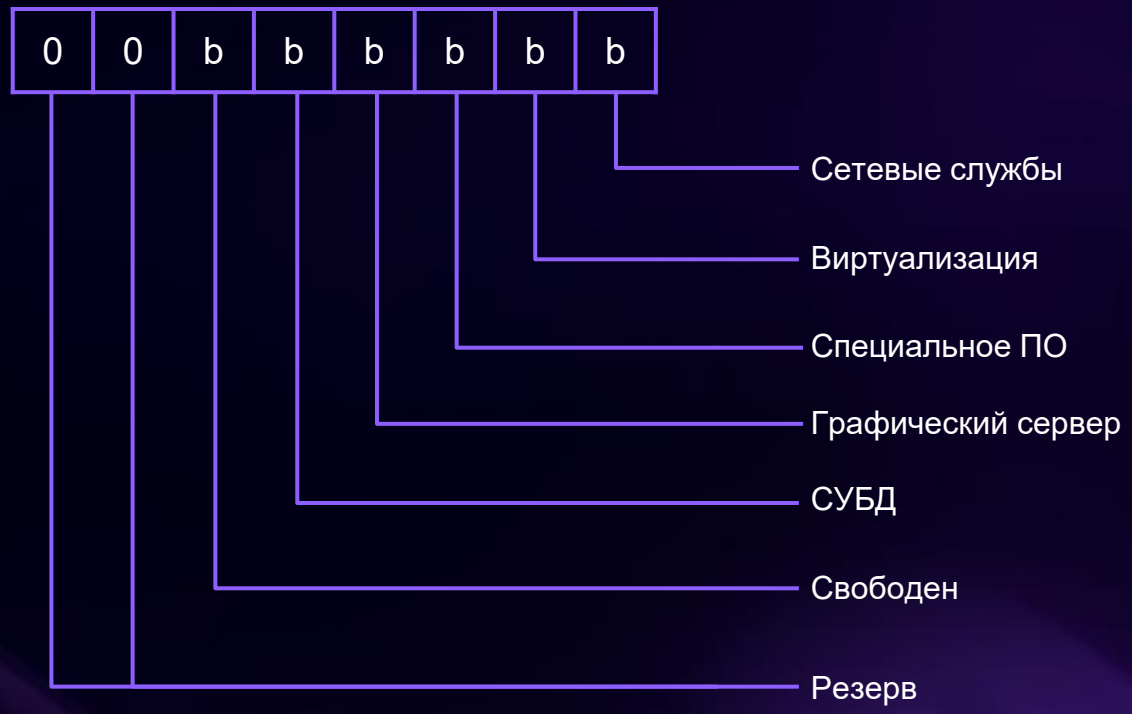
Уровень целостности представляет собой маску, состоящую из набора единичных значений уровней целостности



PARSEC: Целостность

БЕКОН'26

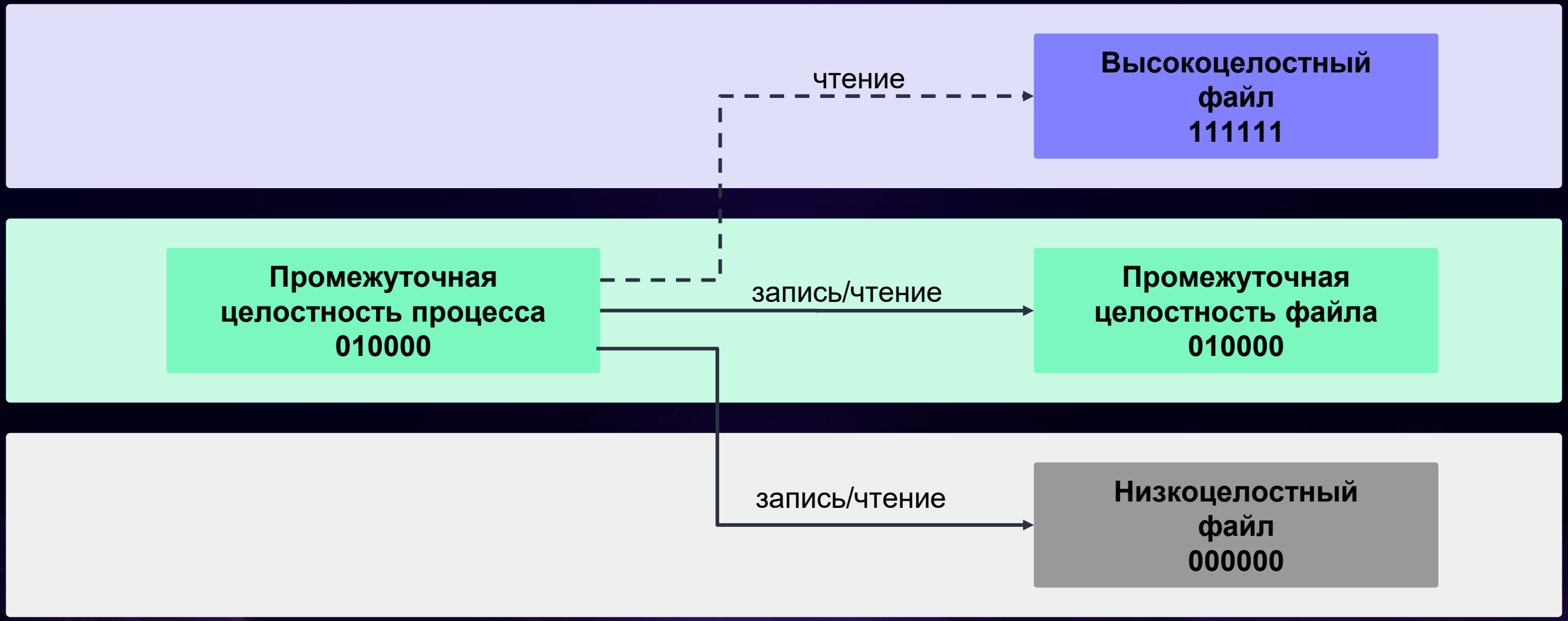
Уровень целостности представляет собой маску, состоящую из набора единичных значений уровней целостности





PARSEC: Целостность

БЕКОН'26

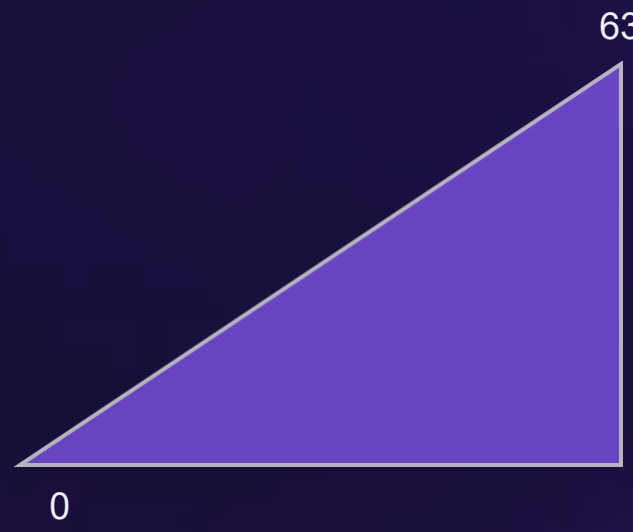


PARSEC: Конфиденциальность

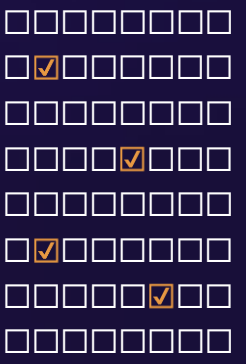
БЕКОН'26

Классификационная метка

Level (Уровень)



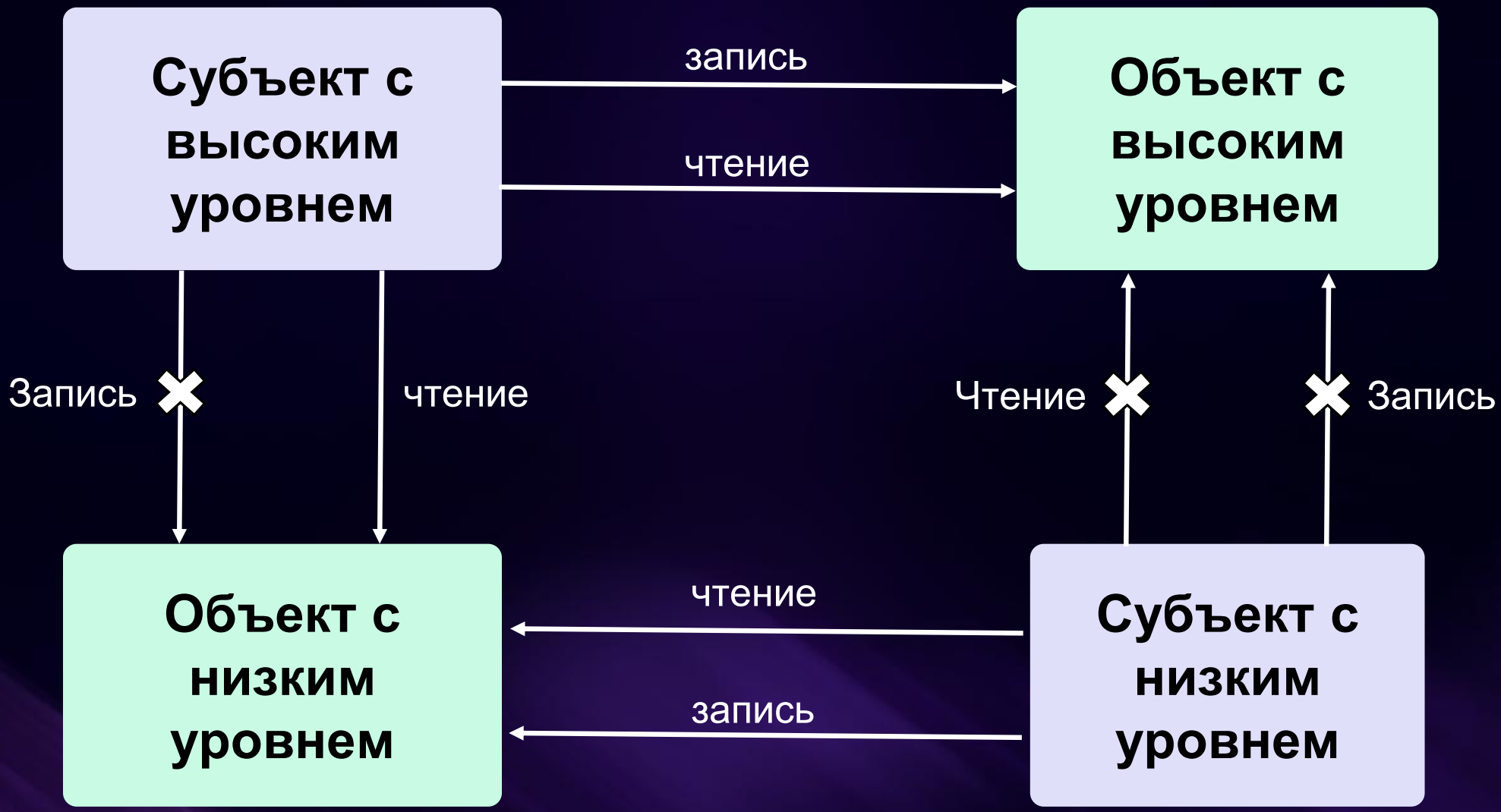
Categories (Категории)



Всего 64 категории

PARSEC: Конфиденциальность

БЕКОН'26



PARSEC: привилегии и флаги

БЕКОН'26

Привилегия	Что разрешает
`PARSEC_CAP_CHMAC`	Изменять метку конфиденциальности и целостности объектов
`PARSEC_CAP_IGNMACCAT`	Игнорировать категории конфиденциальности
`PARSEC_CAP_IGNMACINT`	Игнорировать уровень целостности (МКЦ)
`PARSEC_CAP_SUMAC`	Запускать процессы с **другой** классификационной меткой

Флаг	Что означает
`ccnr`	Каталог может содержать файлы с **разными** метками, но не выше метки каталога
`ehole`	Игнорировать мандатные правила при **записи** в объект (например, `/dev/null`)
`whole`	Разрешить запись в файл **снизу вверх** (чтение — по обычным правилам МРД)
`ssi`	Разрешает чтение и выполнение только субъектам с меткой целостности не ниже собственной

PARSEC

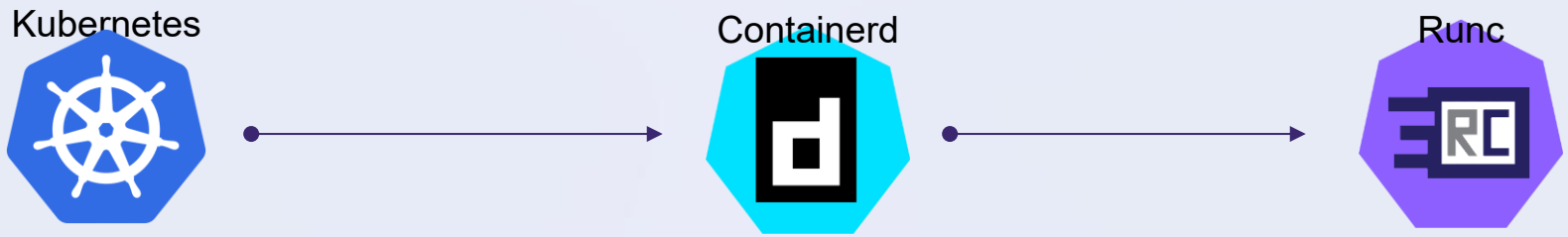
БЕКОН'26

Критерий	AppArmor	**Parsec**	SELinux
Режим работы	Доп. опция	**Включено по умолчанию**	Доп. опция
Принцип работы	на основе путей	**на основе меток**	на основе меток + Type Enforcement
Контроль пользователей	Нет	**Да**	Да
Защита от root	в рамках профиля	**Да, через МКЦ**	в рамках политики
Контроль сети	Per-process фильтр	**Метка в пакетах**	Ограничение портов
Криптографическая целостность бинарников	нет	**Да, ЗПС**	нет
Основная модель угроз	Скомпрометированное приложение	**Недоверенный пользователь**	Приложение + привилегированный пользователь
Сложность	Низкая	**Средняя**	Высокая
Открытость	Open source	**Полностью своя разработка**	Open source

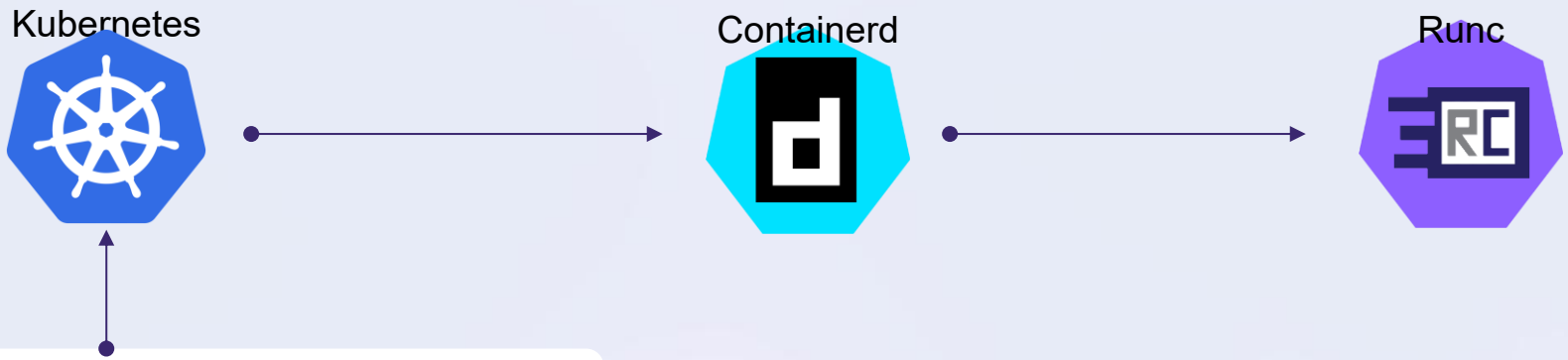
МКЦ и МРД в Kubernetes

03

МКЦ и МРД в Kubernetes



МКЦ и МРД в Kubernetes



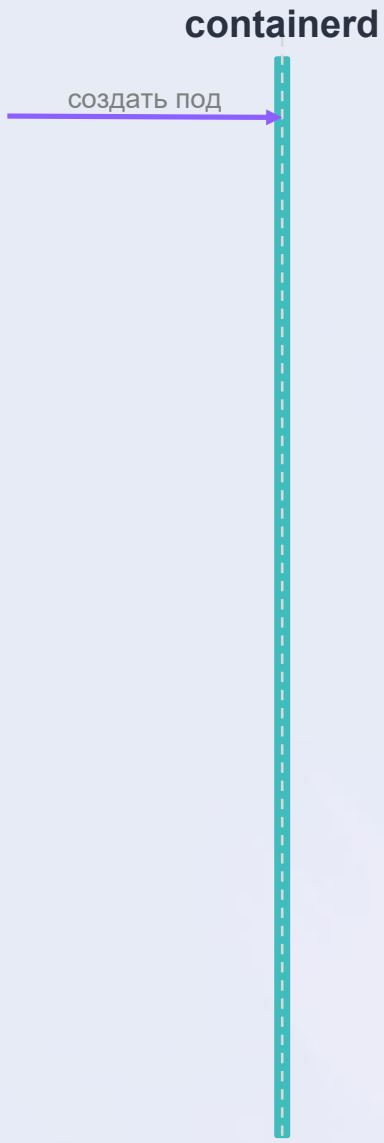
Аннотации в манифесте пода

```
annotations:
  io.astralinux.parsec.ilev: "0"
  io.astralinux.parsec.clev: "1"
  io.astralinux.parsec.cap:  "PARSEC_CAP_CHMAC"
```

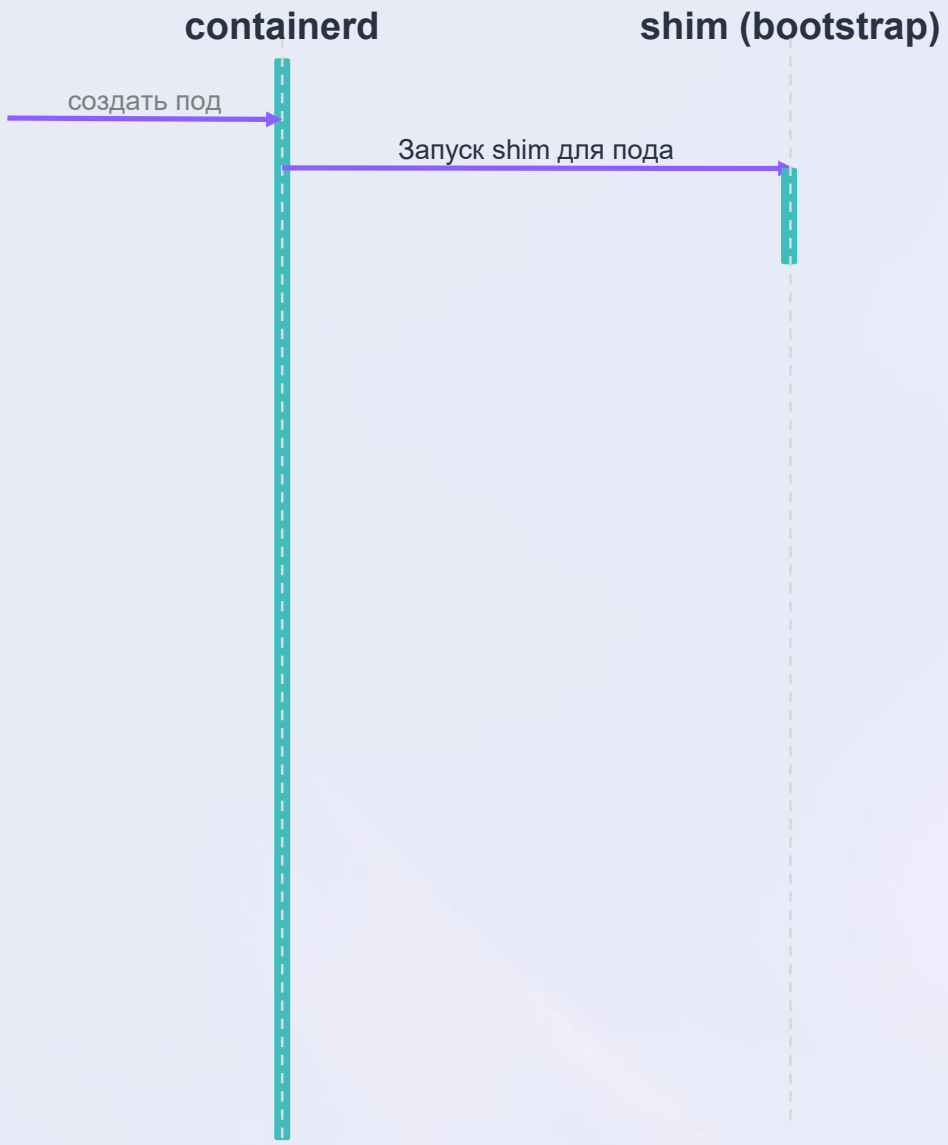
Containerd

04

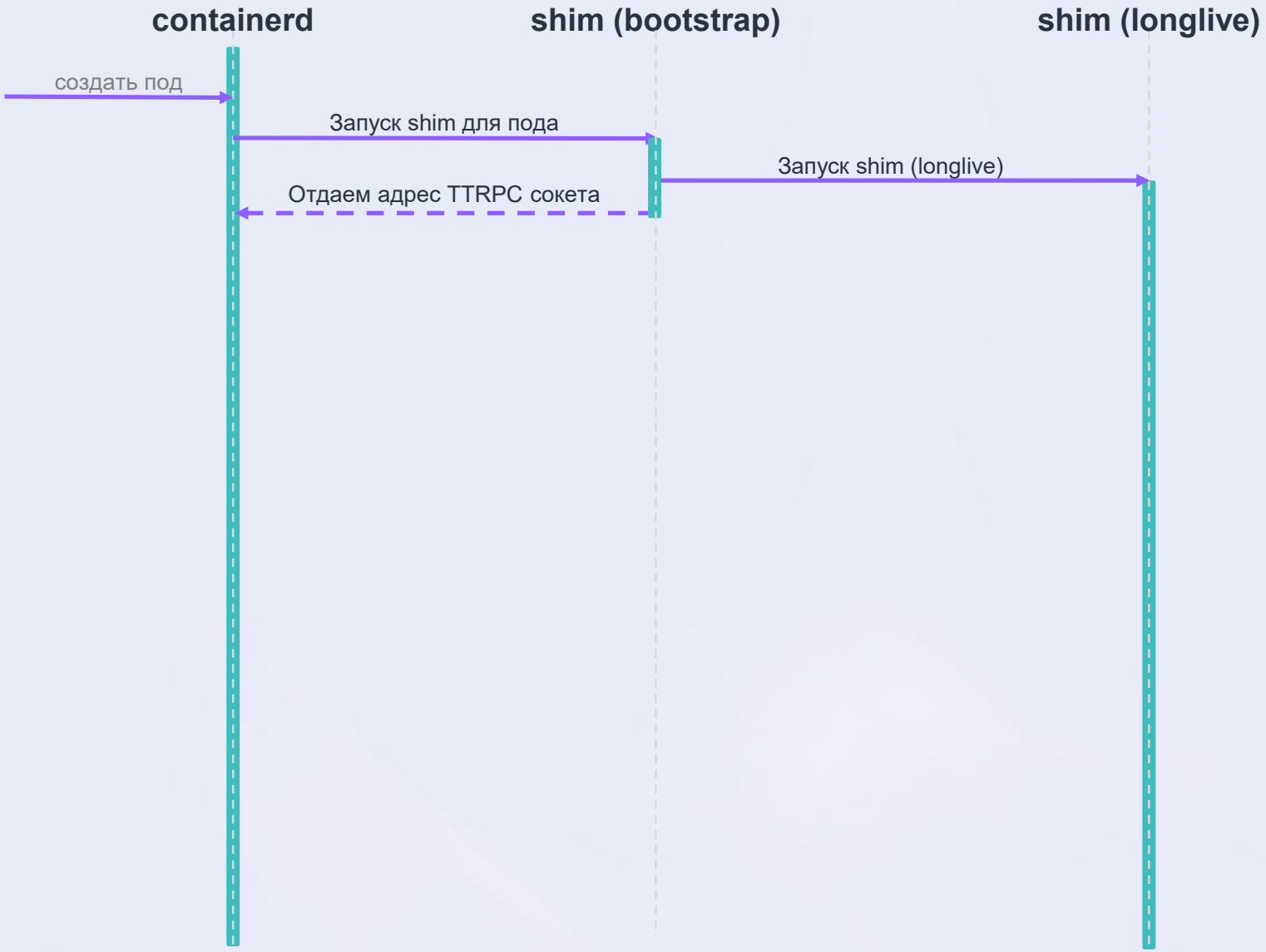
Containerd



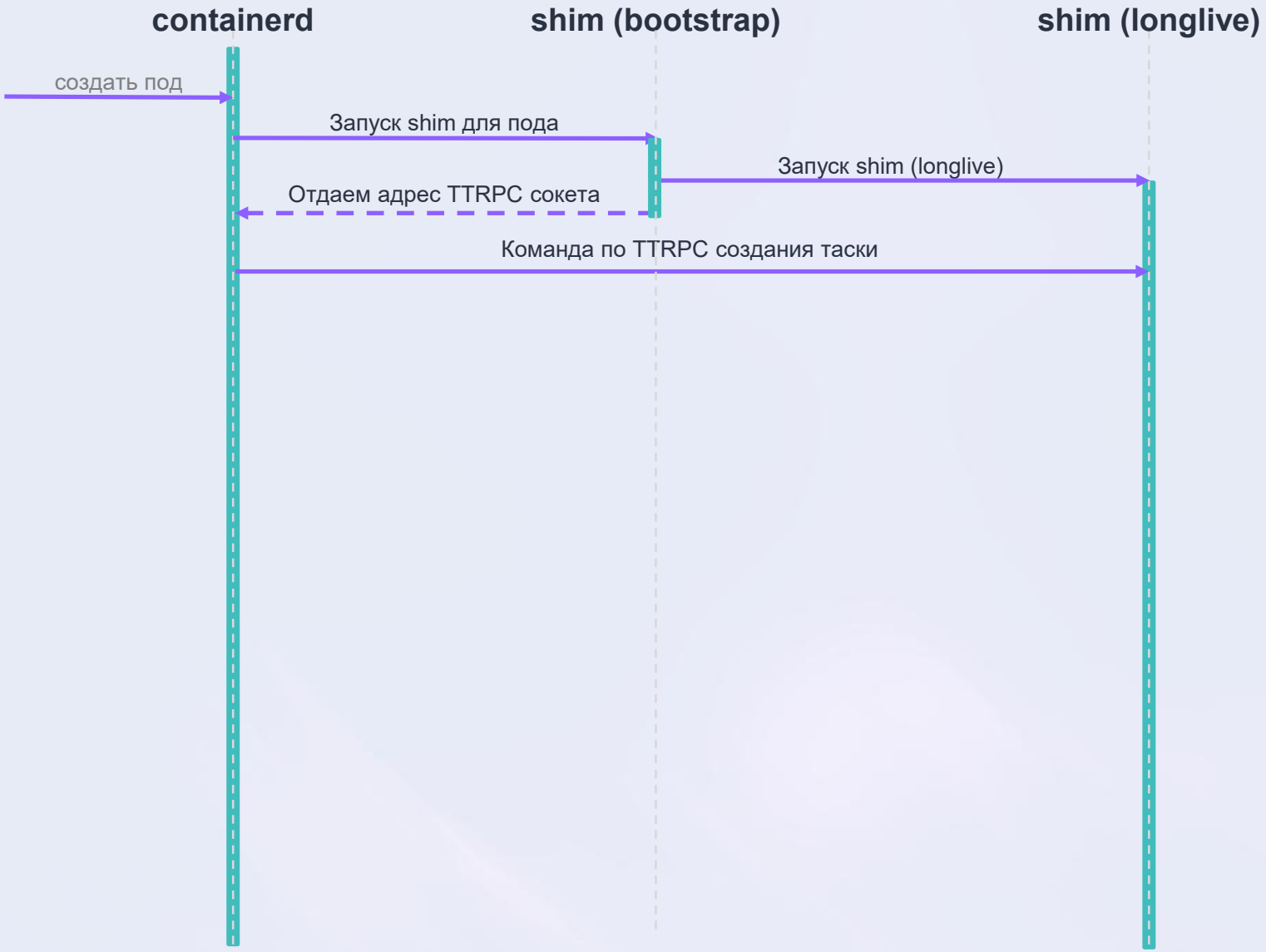
Containerd


containerd
shim (bootstrap)
создать под
Запуск shim для пода

Containerd


containerd
shim (bootstrap)
shim (longlive)
создать под
Запуск shim для пода
Запуск shim (longlive)
Отдаем адрес TTRPC сокета

Containerd


containerd
shim (bootstrap)
shim (longlive)
создать под
Запуск shim для пода
Запуск shim (longlive)
Отдаем адрес TTRPC сокета
Команда по TTRPC создания таски

Containerd


containerd
shim (bootstrap)
shim (longlive)
runc
создать под
Запуск shim для пода
Запуск shim (longlive)
Отдаем адрес TTRPC сокета
Команда по TTRPC создания таски
Запуск контейнерного процесса
runc init

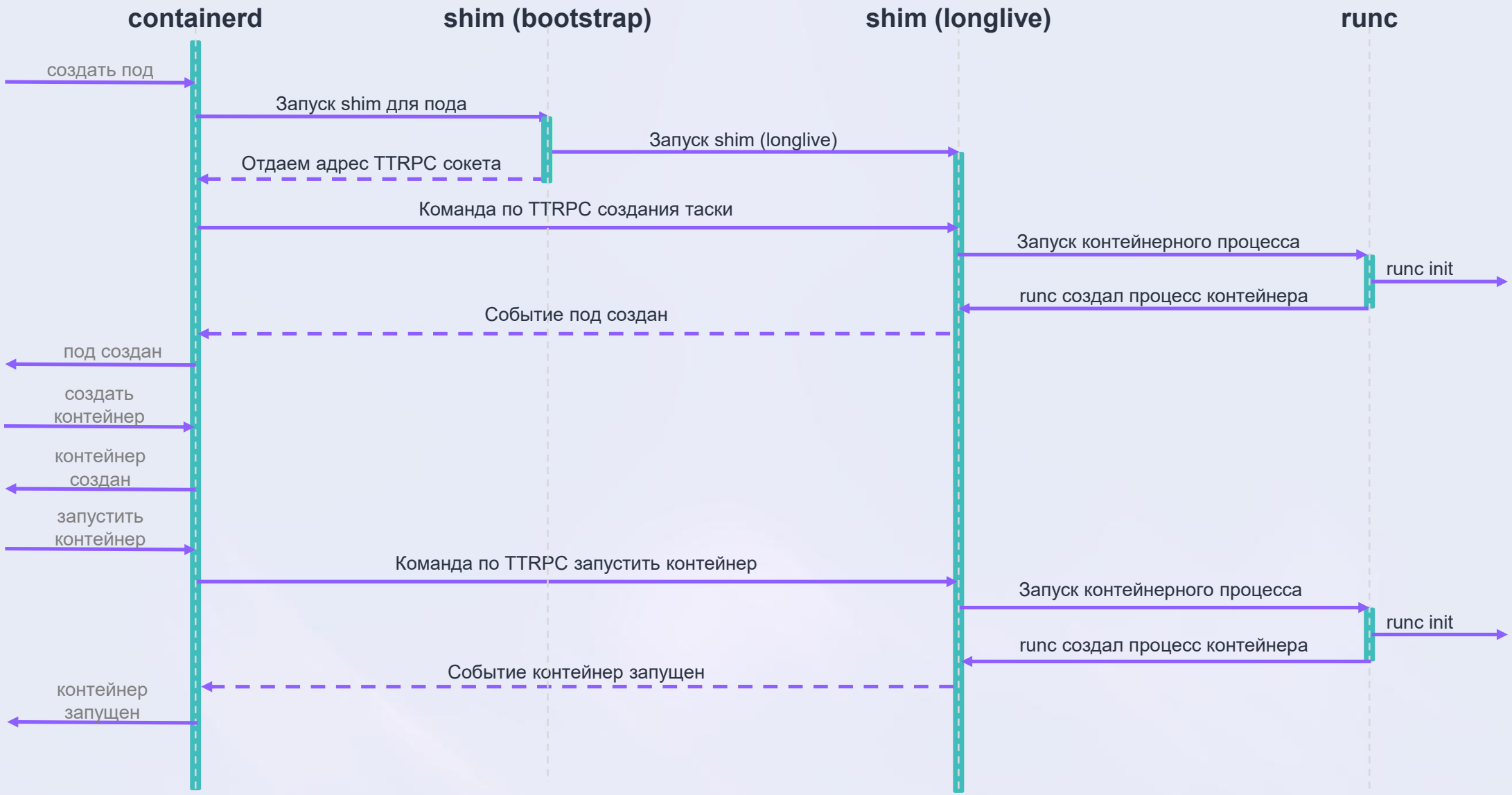
Containerd


containerd
shim (bootstrap)
shim (longlive)
runc
создать под
Запуск shim для пода
Запуск shim (longlive)
Отдаем адрес TTRPC сокета
Команда по TTRPC создания таски
Запуск контейнерного процесса
runc init
runc создал процесс контейнера
Событие под создан
под создан

Containerd

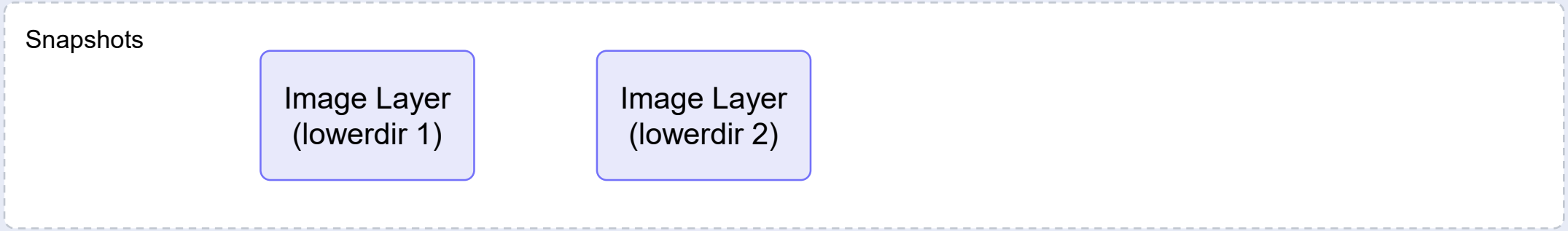

containerd
shim (bootstrap)
shim (longlive)
runc
создать под
Запуск shim для пода
Запуск shim (longlive)
Отдаем адрес TTRPC сокета
Команда по TTRPC создания таски
Запуск контейнерного процесса
runc init
runc создал процесс контейнера
Событие под создан
под создан
создать контейнер
контейнер создан

Containerd


containerd
shim (bootstrap)
shim (longlive)
runc
создать под
Запуск shim для пода
Запуск shim (longlive)
Отдаем адрес TTRPC сокета
Команда по TTRPC создания таски
Запуск контейнерного процесса
runc init
runc создал процесс контейнера
Событие под создан
под создан
создать контейнер
контейнер создан
запустить контейнер
Команда по TTRPC запустить контейнер
Запуск контейнерного процесса
runc init
runc создал процесс контейнера
Событие контейнер запущен
контейнер запущен

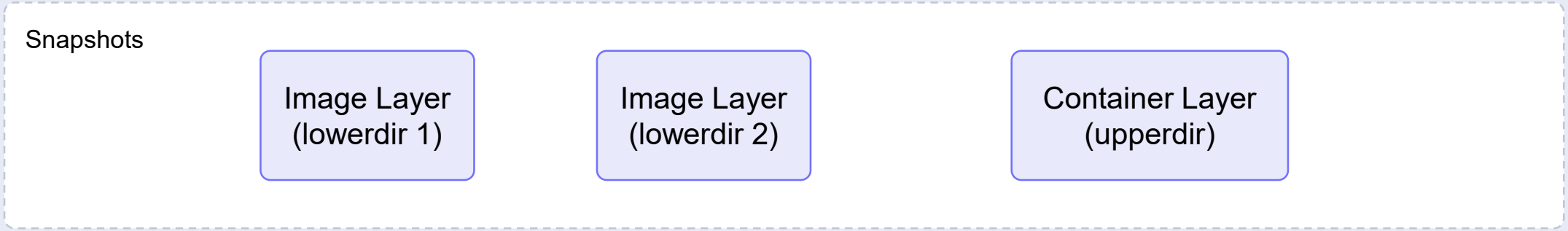
Containerd

Overlayfs snapshotter



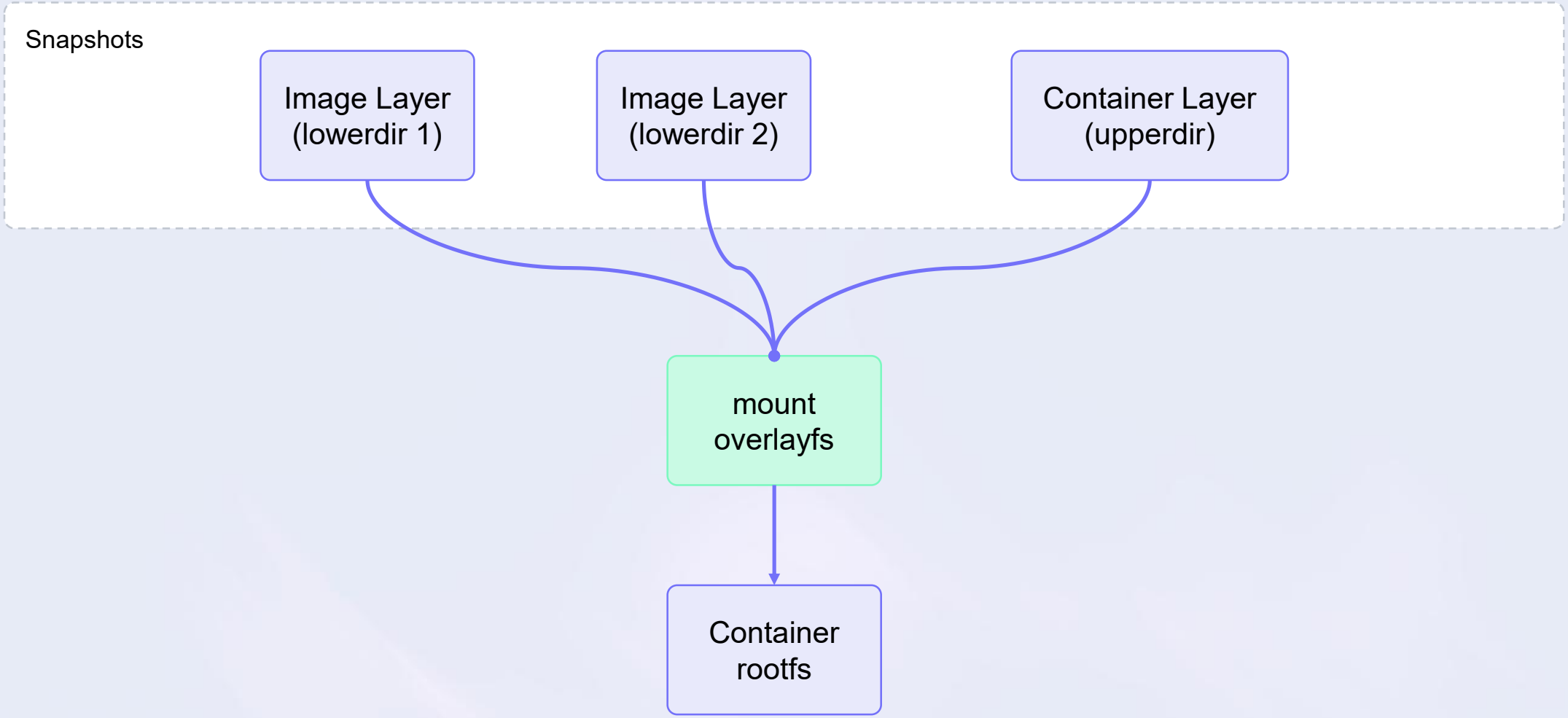
Containerd

Overlayfs snapshotter



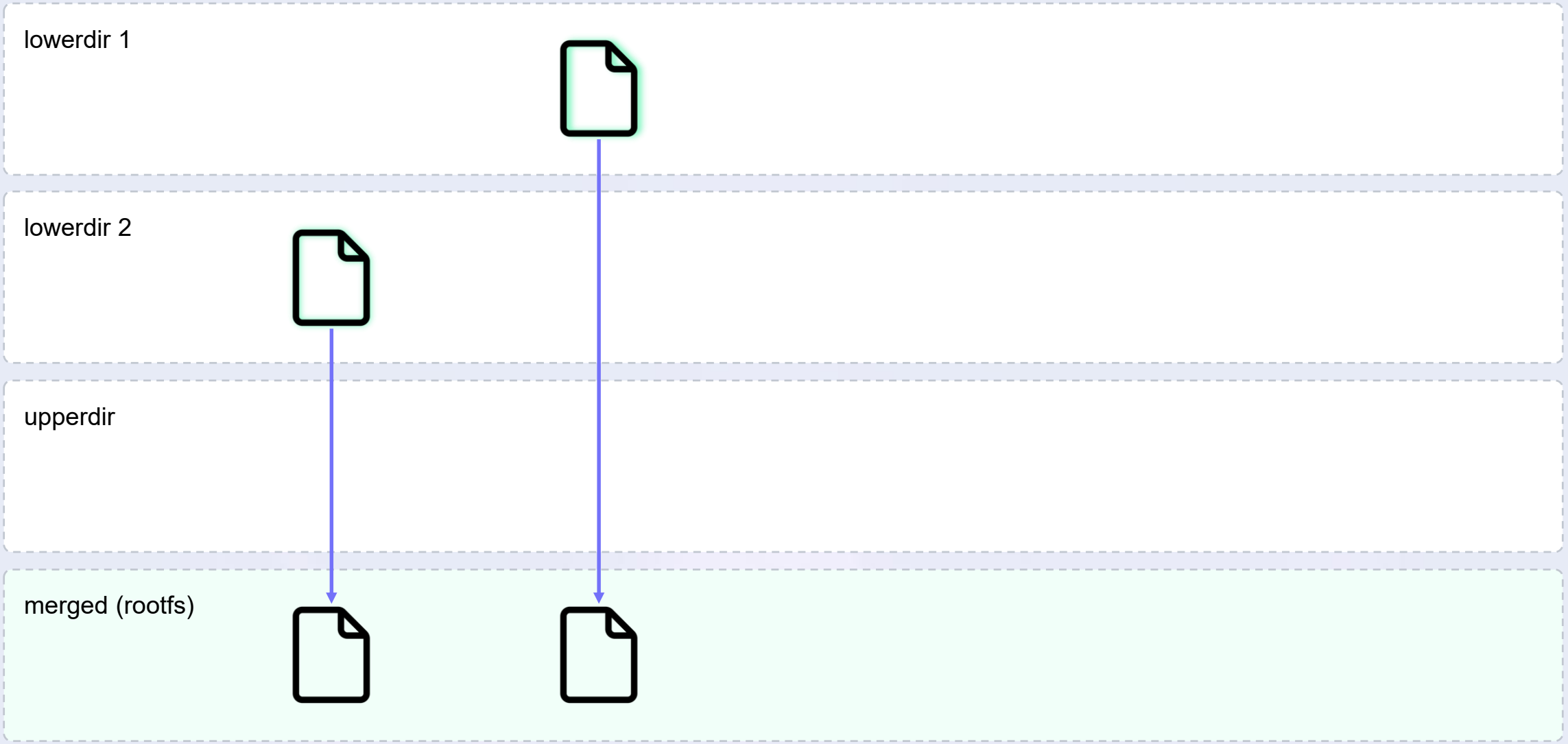
Containerd

Overlayfs snapshotter



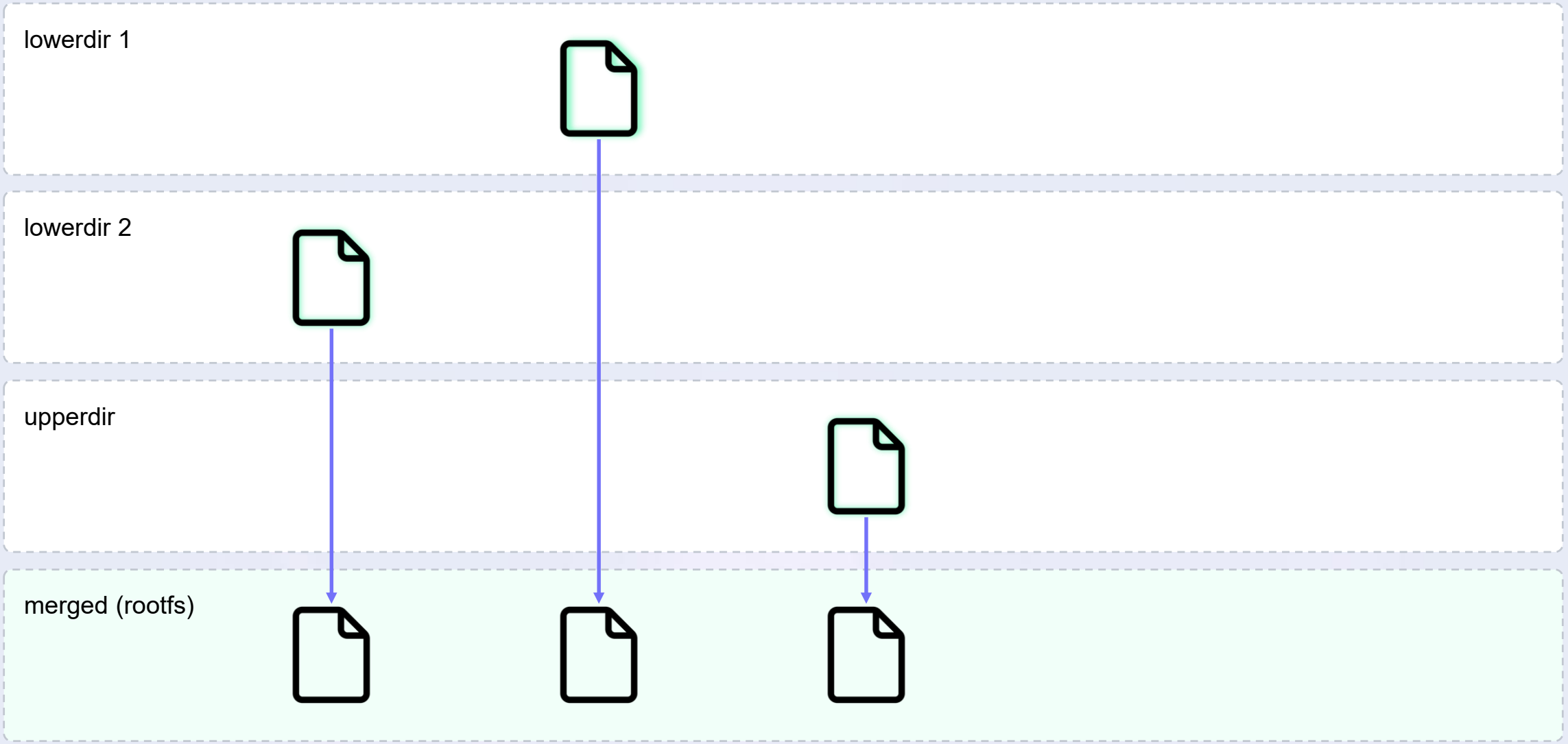
Containerd

Overlayfs snapshotter



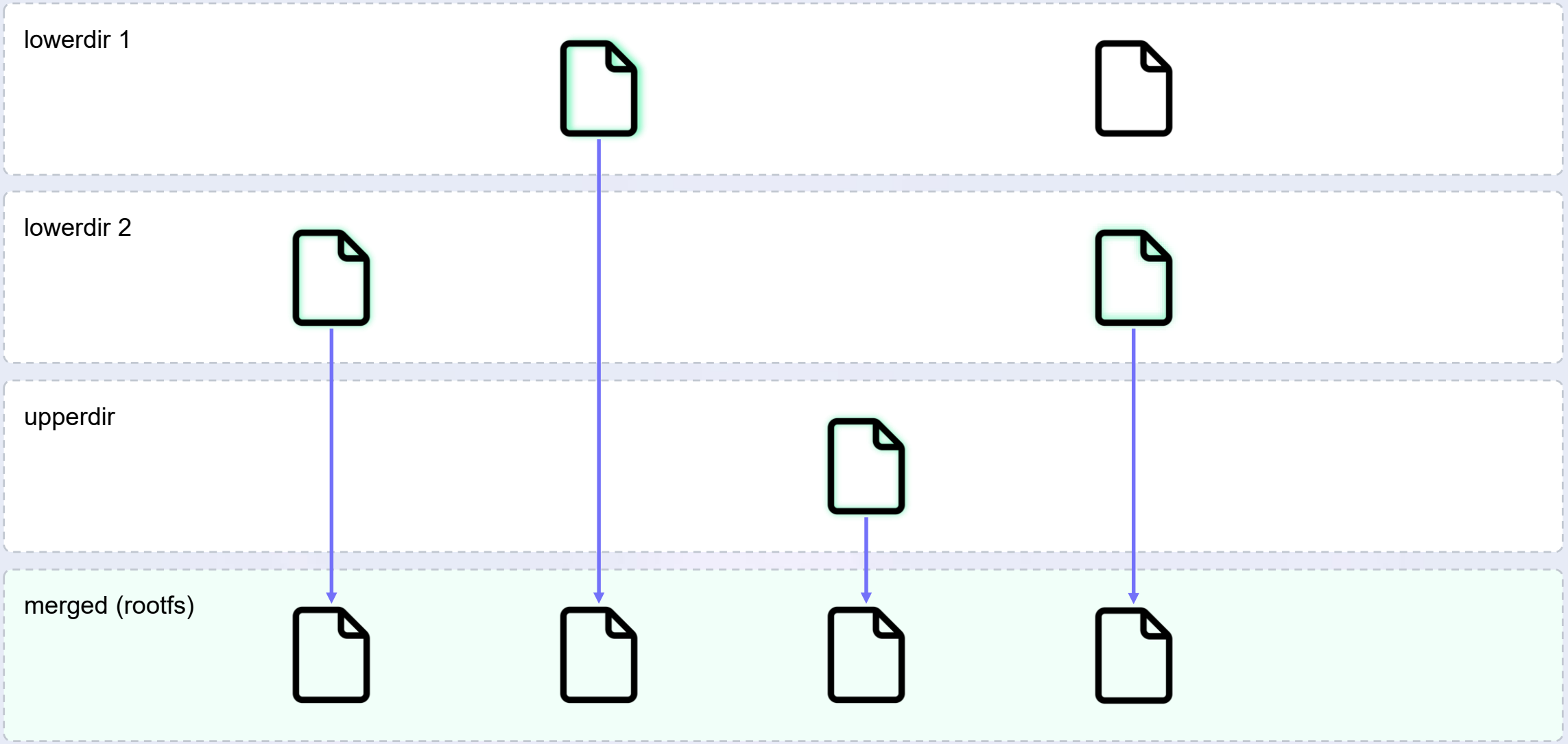
Containerd

Overlayfs snapshotter



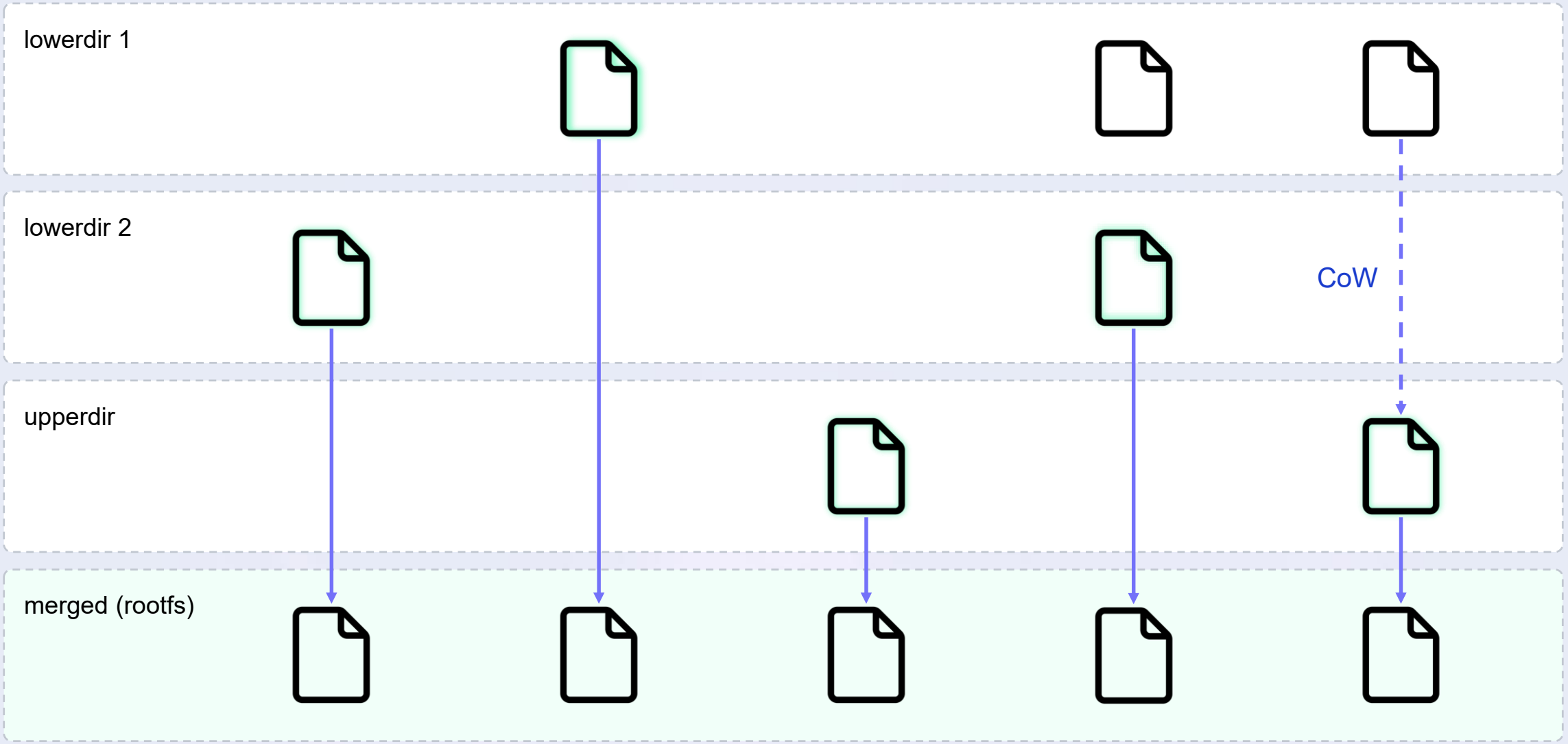
Containerd

Overlayfs snapshotter



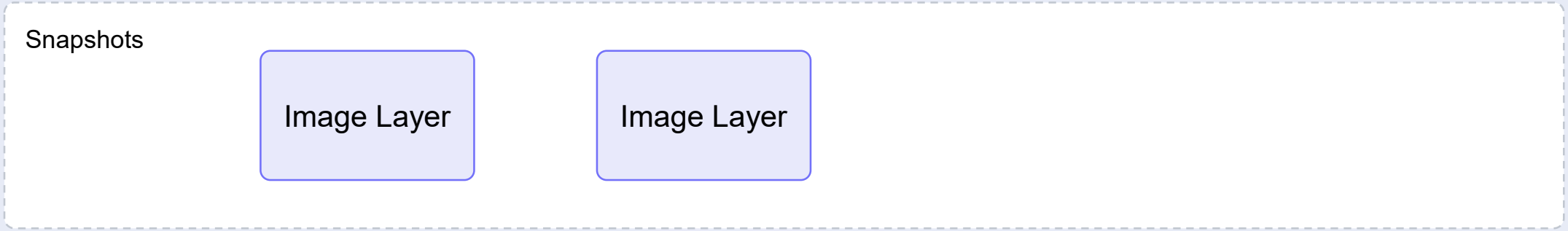
Containerd

Overlayfs snapshotter



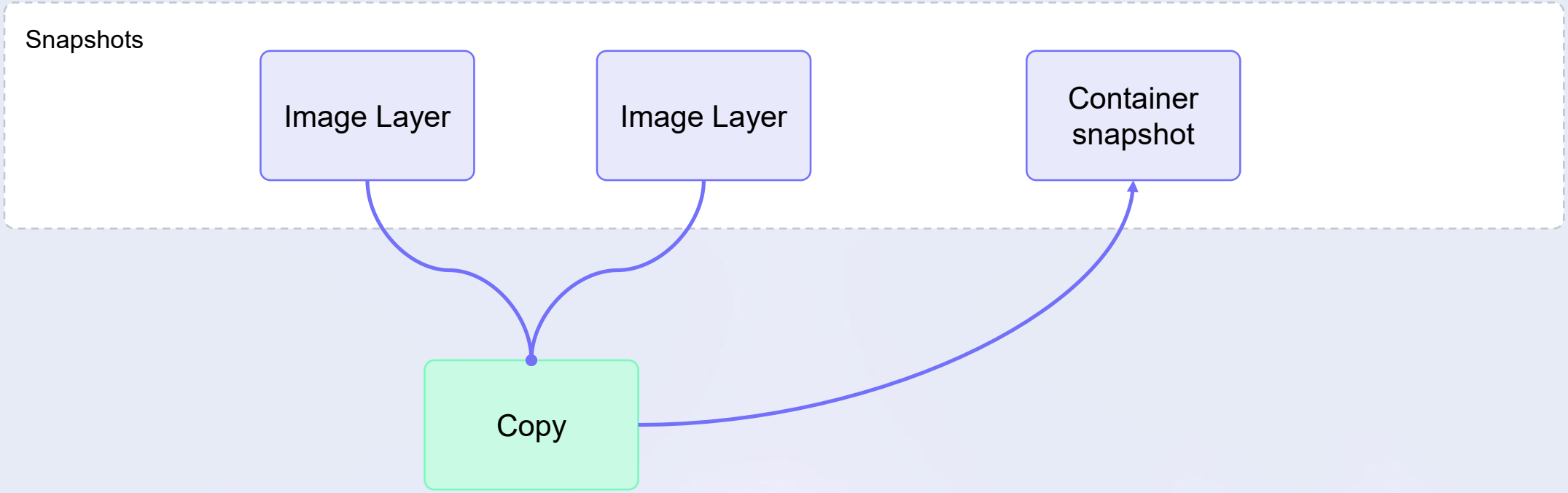
Containerd

Native snapshotter



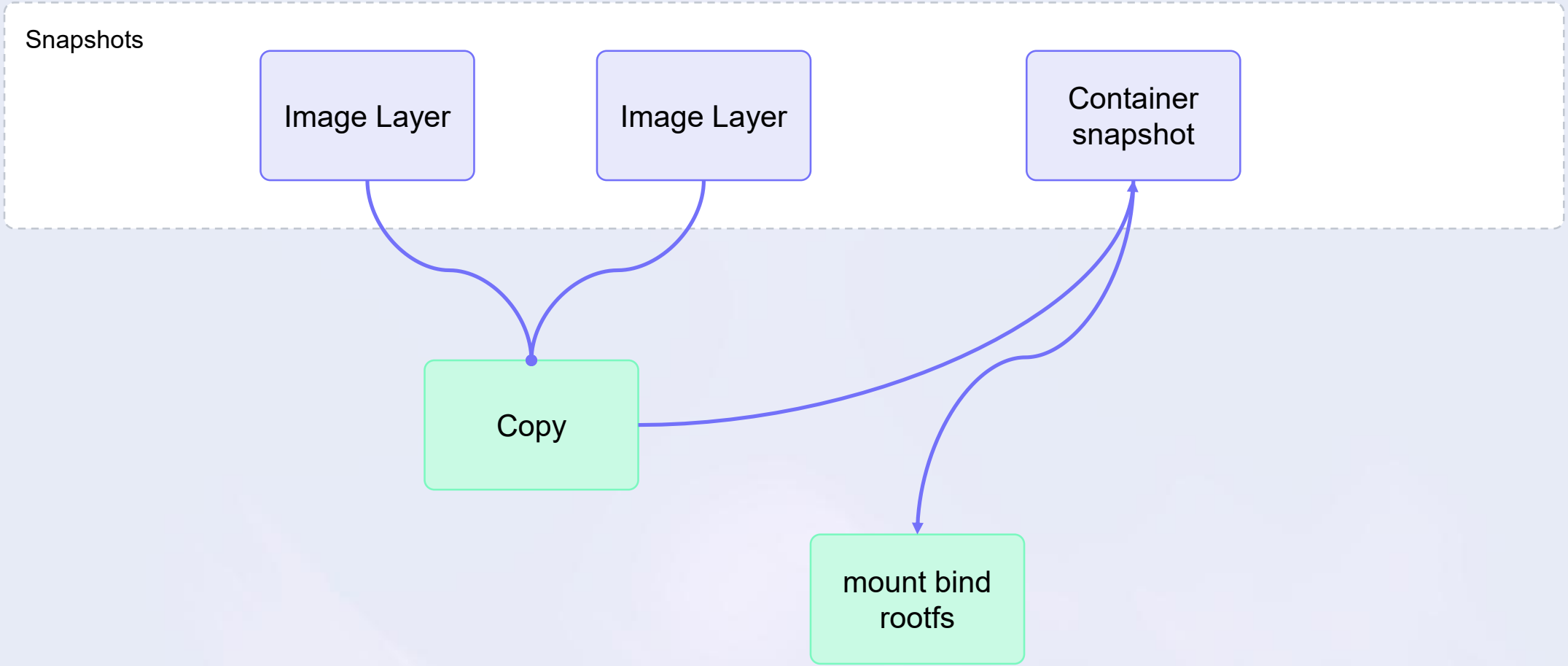
Containerd

Native snapshotter



Containerd

Native snapshotter



Containerd
Внедряем PARSEC

При создании контейнера

```
opts := []containerd.NewContainerOpts{
    containerd.WithSnapshotter(...),
    customopts.WithNewSnapshot(...),
    containerd.WithSpec(...),
    containerd.WithContainerLabels(...),
    containerd.WithContainerExtension(...),
    containerd.WithRuntime(...),
    containerd.WithParsecLabels(clev, ilev),
}
```

`WithParsecLabels(clev, ilev)`

Назначает файловой системе контейнера заданные в аннотациях метки

- Для `overlayfs` снапшота влияет только на upperdir
- Для `native` снапшота полностью на всю итоговую ФС



Runc

05

Runc

Первым делом запускается C код

до golang runtime

```
#cgo CFLAGS: -Wall
extern void nsexec();
void __attribute__((constructor)) init(void) {
        nsexec();
}
```

Runc

Первым делом запускается C код

до golang runtime

```
#cgo CFLAGS: -Wall
extern void nsexec();
void __attribute__((constructor)) init(void) {
        nsexec();
}
```

Но действительно что то делает только при runc init

```
pipenum = getenv_int("_LIBCONTAINER_INITPIPE");
if (pipenum < 0) {
        /* We are not a runc init. Just return to go runtime. */
        return;
}
```

Runc

БЕКОН'26

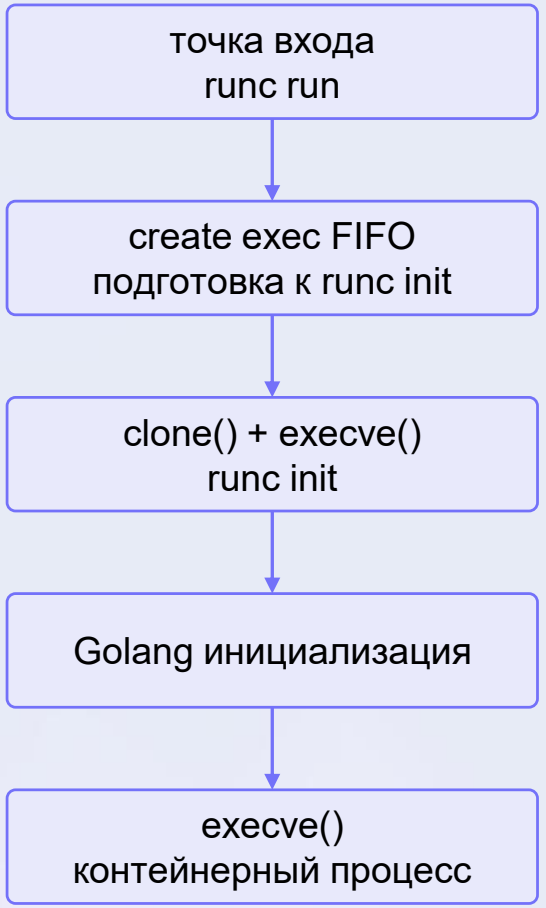
Первым делом запускается C код

до golang runtime

```
#cgo CFLAGS: -Wall
extern void nsexec();
void __attribute__((constructor)) init(void) {
        nsexec();
}
```

Но действительно что то делает только при runc init

```
pipenum = getenv_int("_LIBCONTAINER_INITPIPE");
if (pipenum < 0) {
        /* We are not a runc init. Just return to go runtime. */
        return;
}
```



Runc

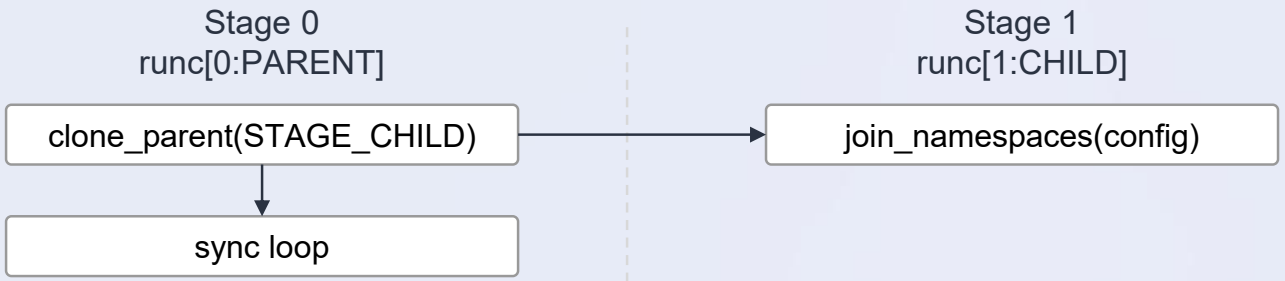
процесс инициализации контейнера

Stage 0
runc[0:PARENT]

clone_parent(STAGE_CHILD)

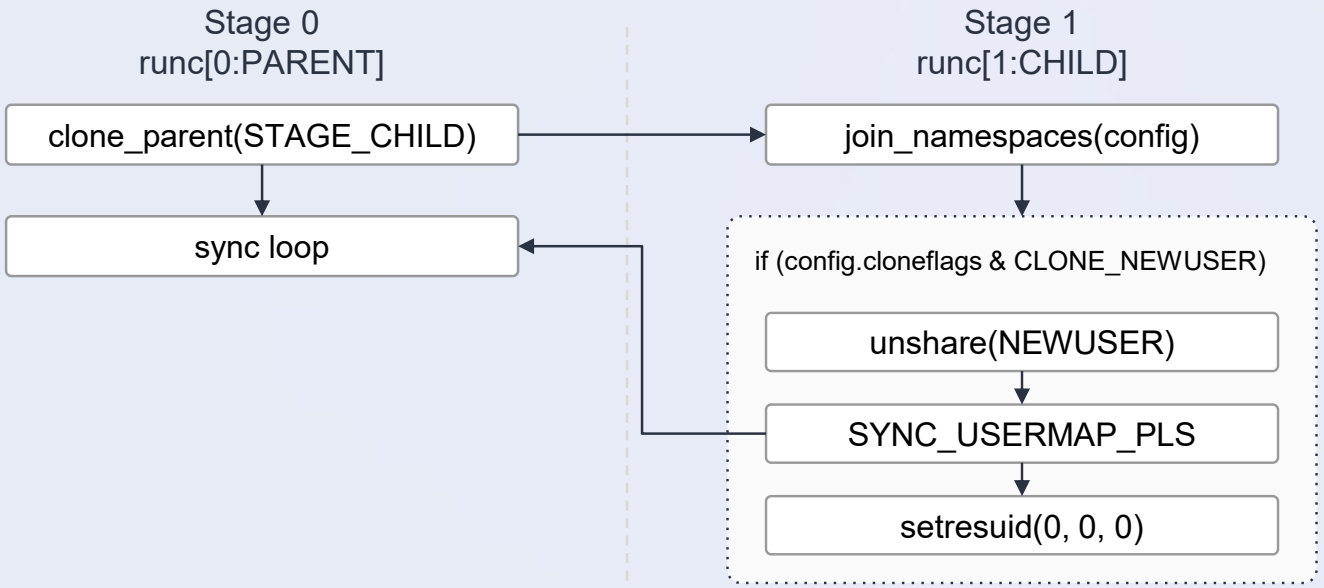
Runc

процесс инициализации контейнера



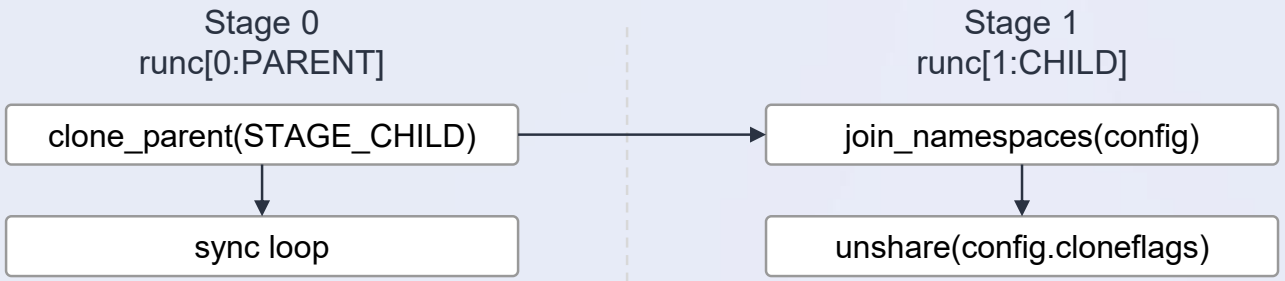
Runc

процесс инициализации контейнера



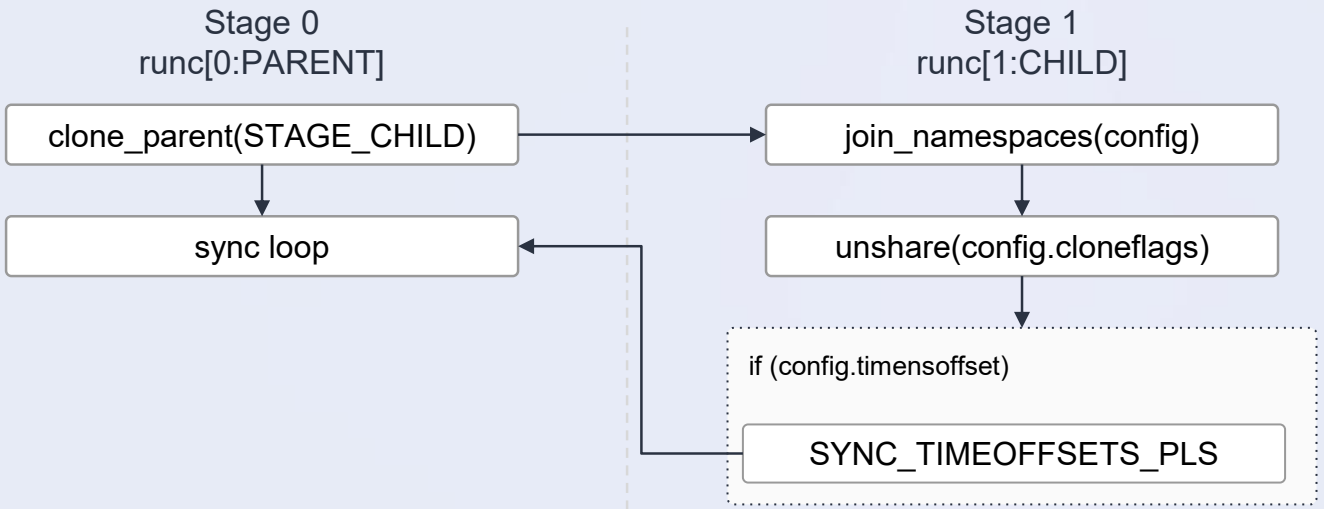
Runc

процесс инициализации контейнера



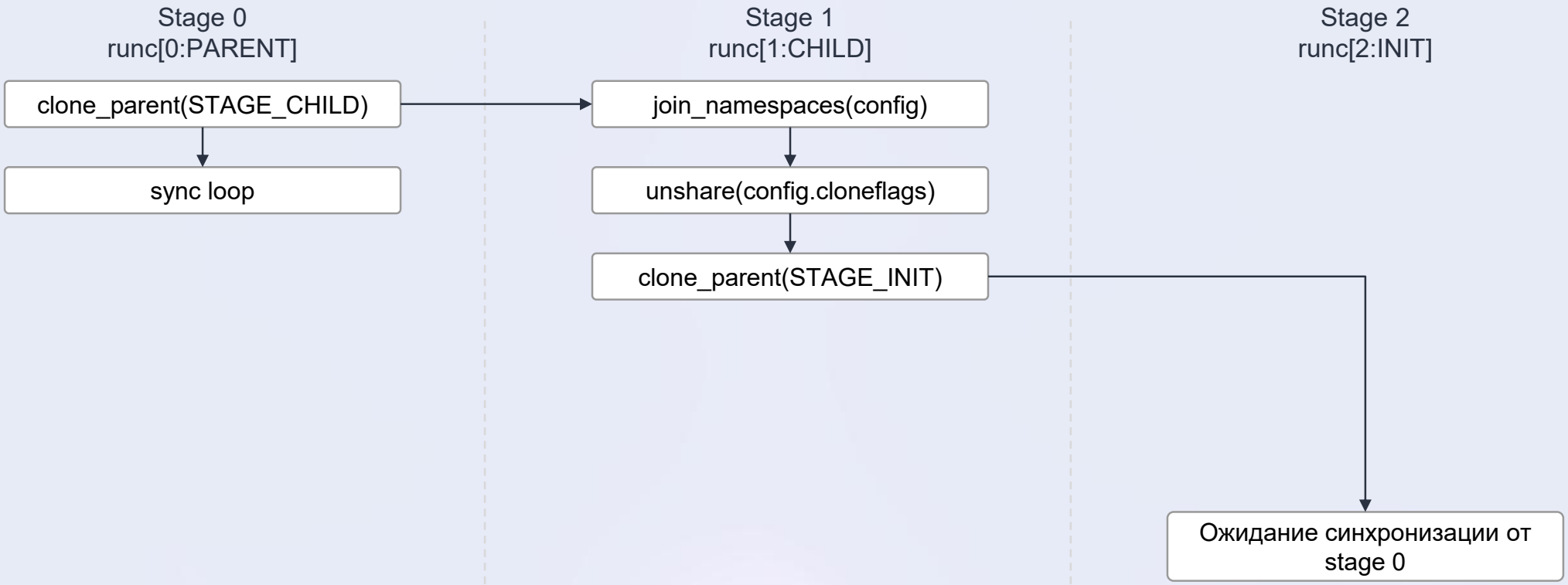
Runc

процесс инициализации контейнера



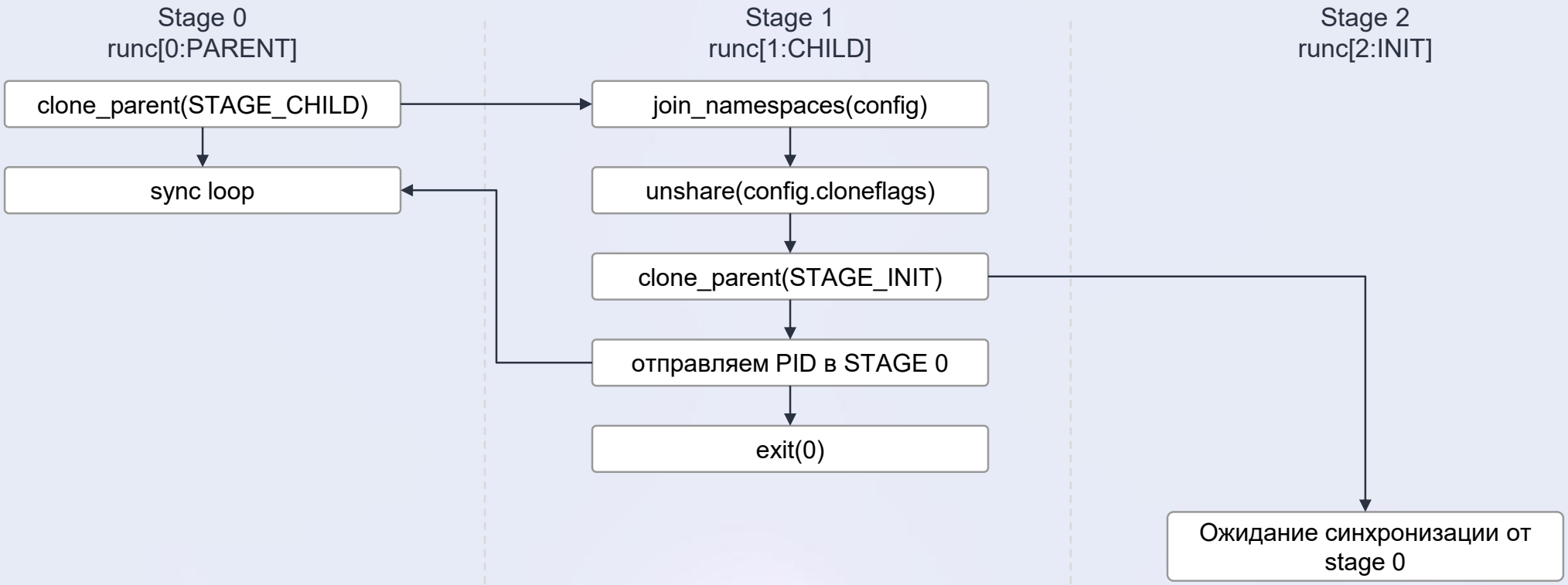
Runc

процесс инициализации контейнера



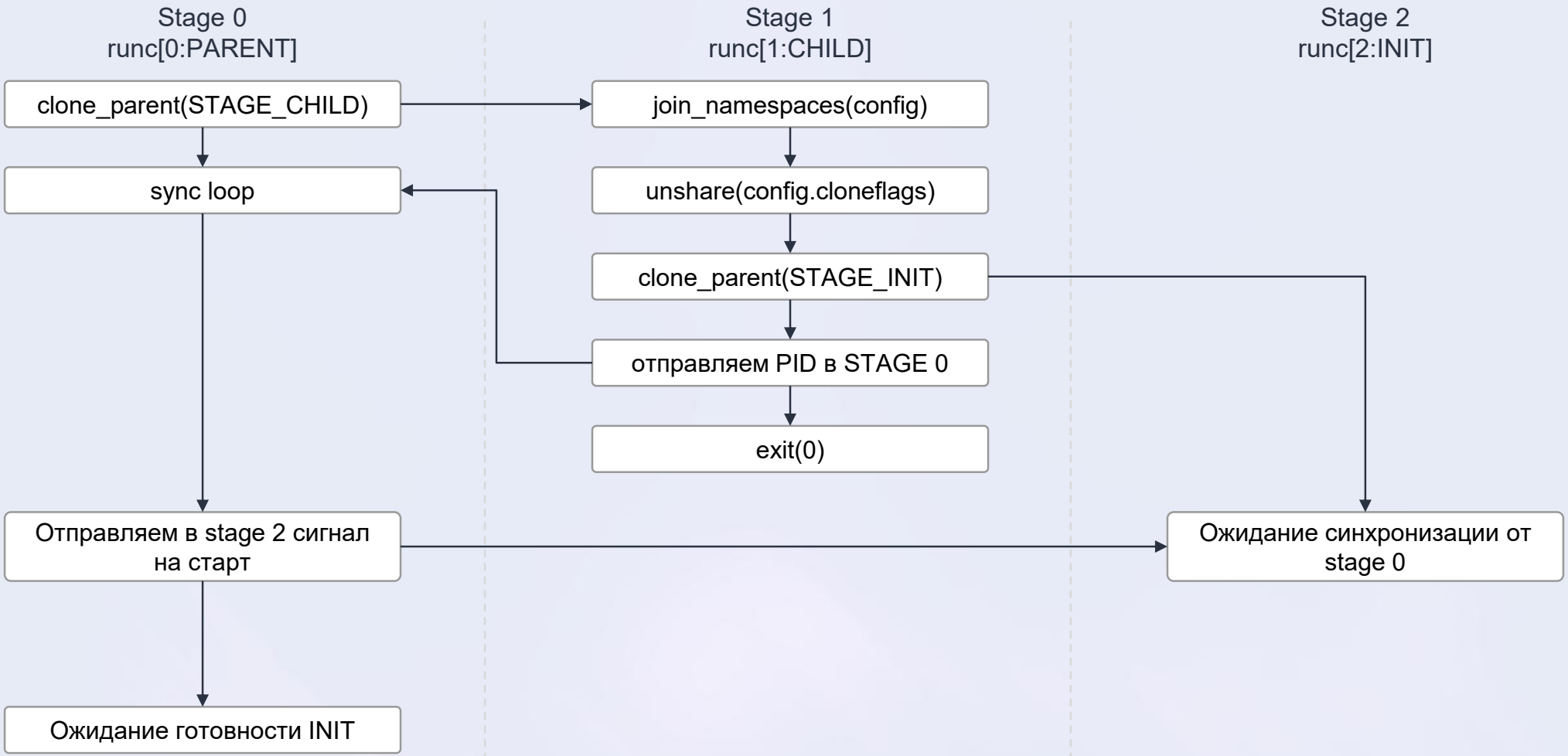
Runc

процесс инициализации контейнера



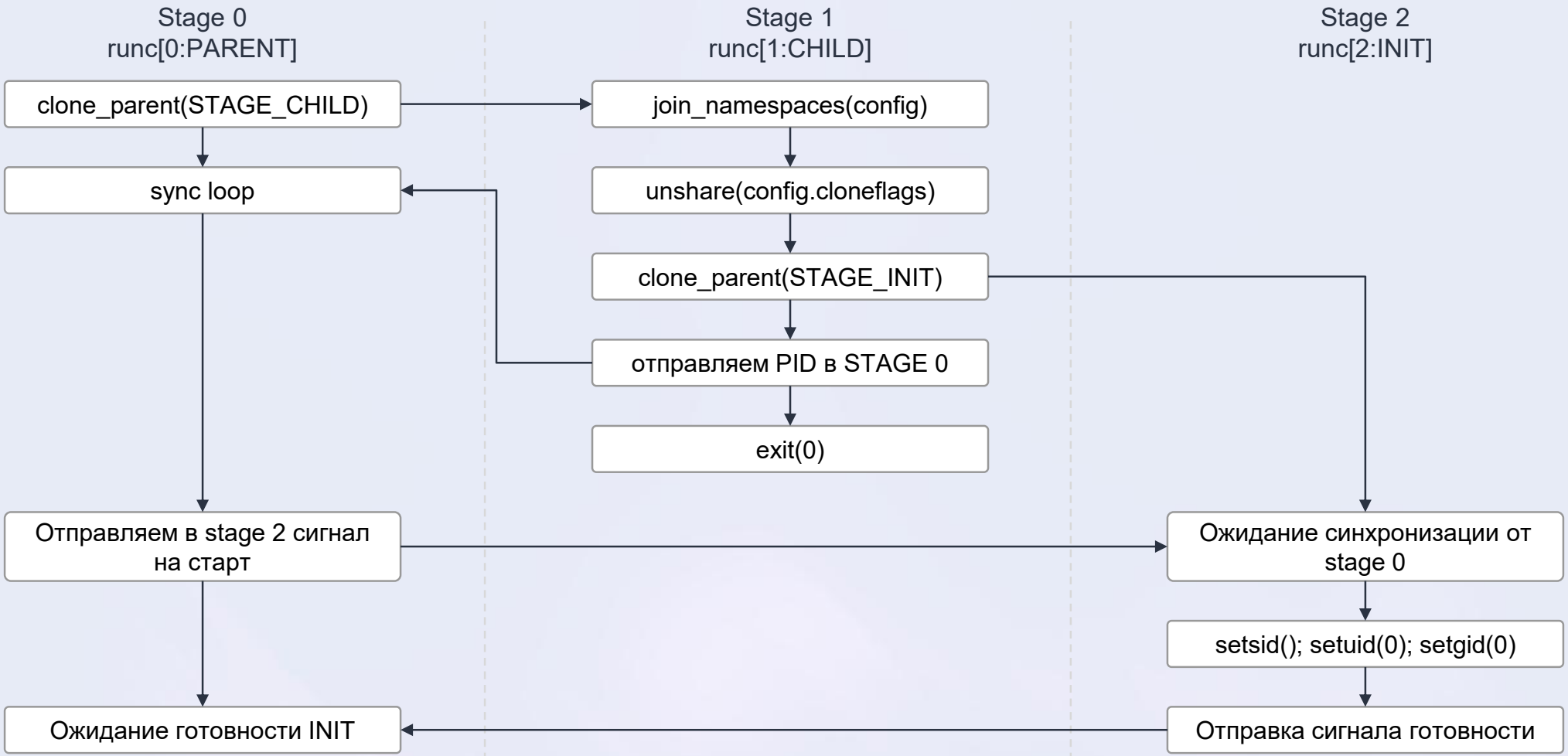
Runc

процесс инициализации контейнера



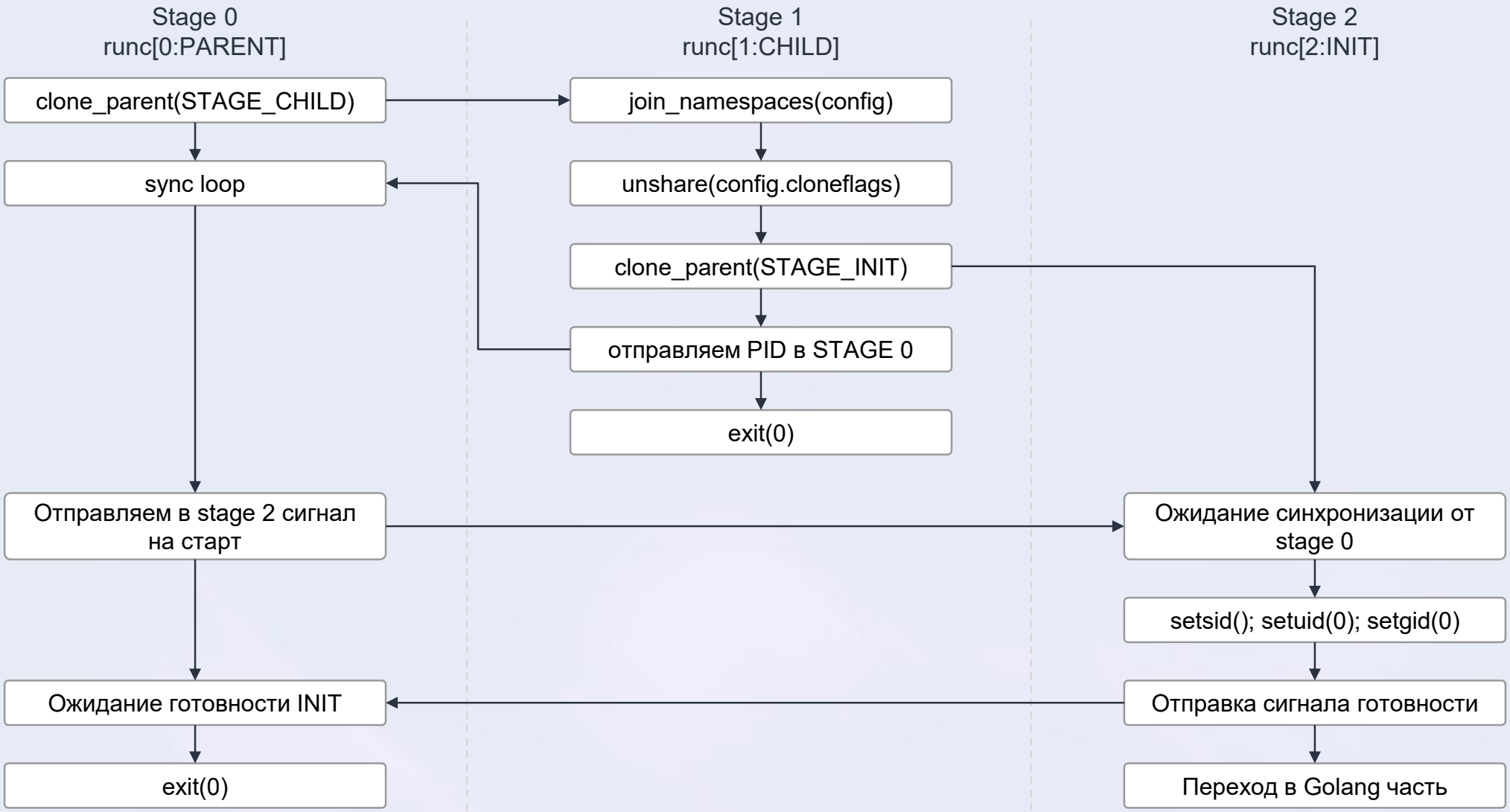
Runc

процесс инициализации контейнера



Runc

процесс инициализации контейнера



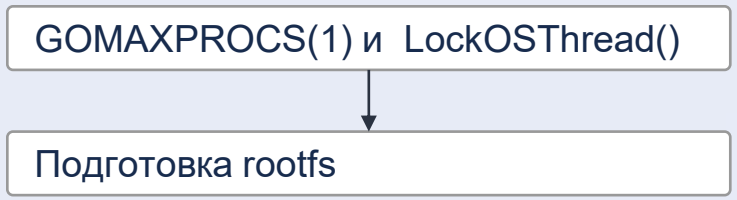
Runc

процесс инициализации контейнера

GOMAXPROCS(1) и LockOSThread()

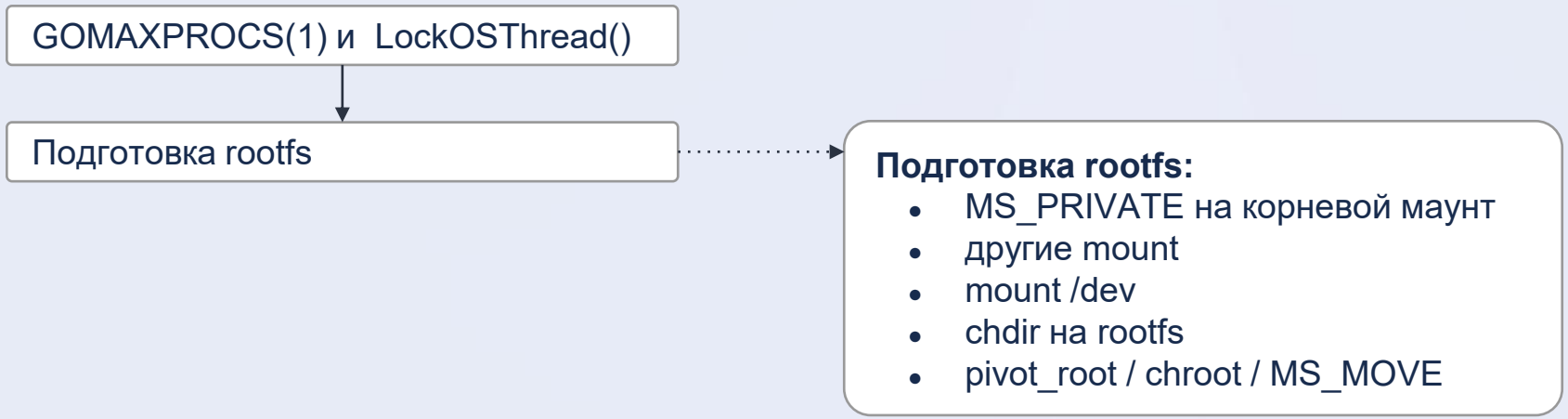
Runc

процесс инициализации контейнера



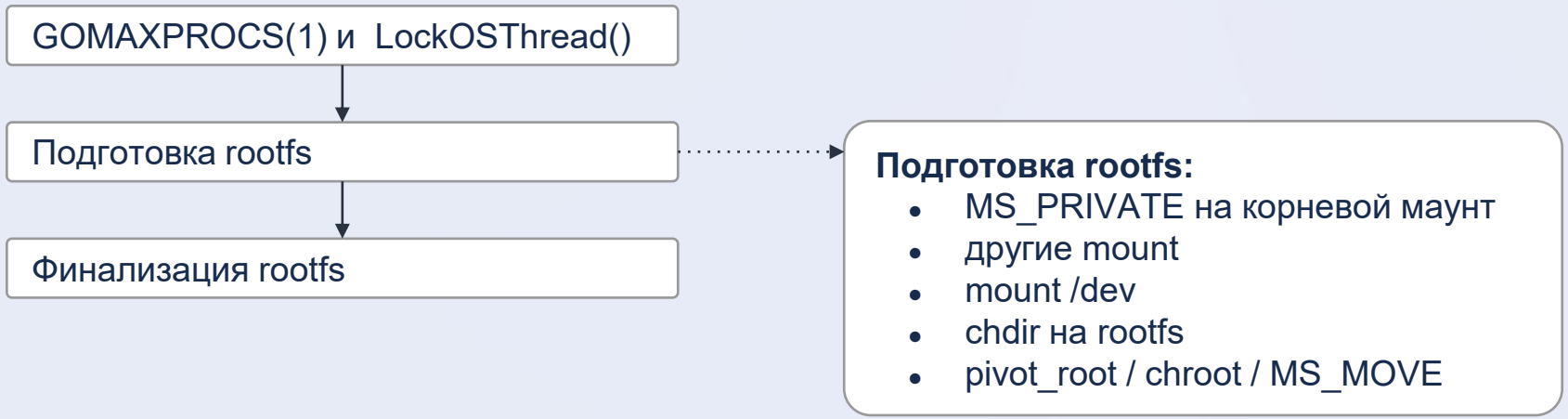
Runc

процесс инициализации контейнера



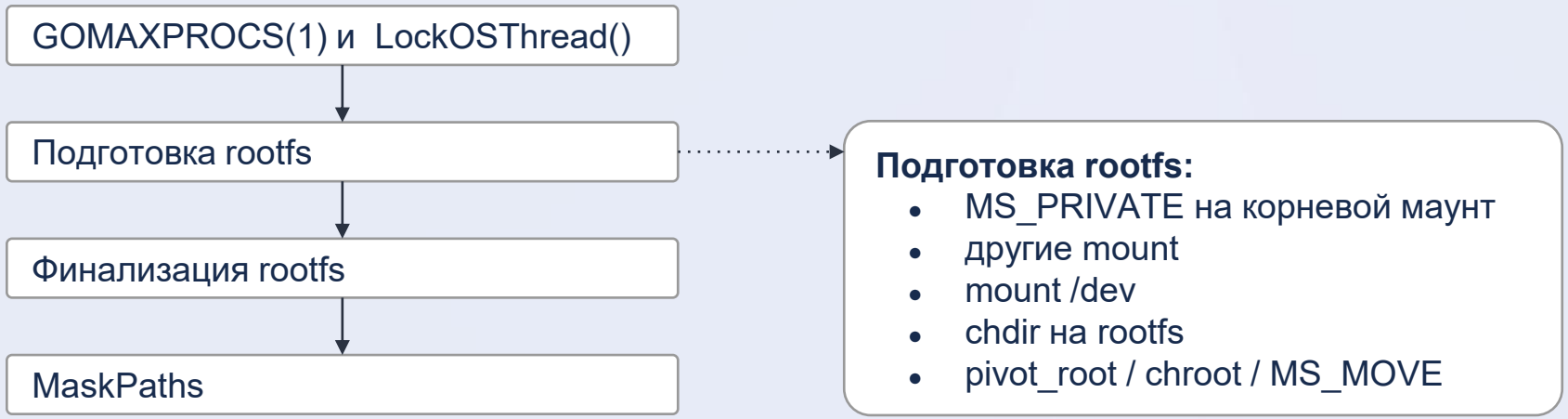
Runc

процесс инициализации контейнера



Runc

процесс инициализации контейнера



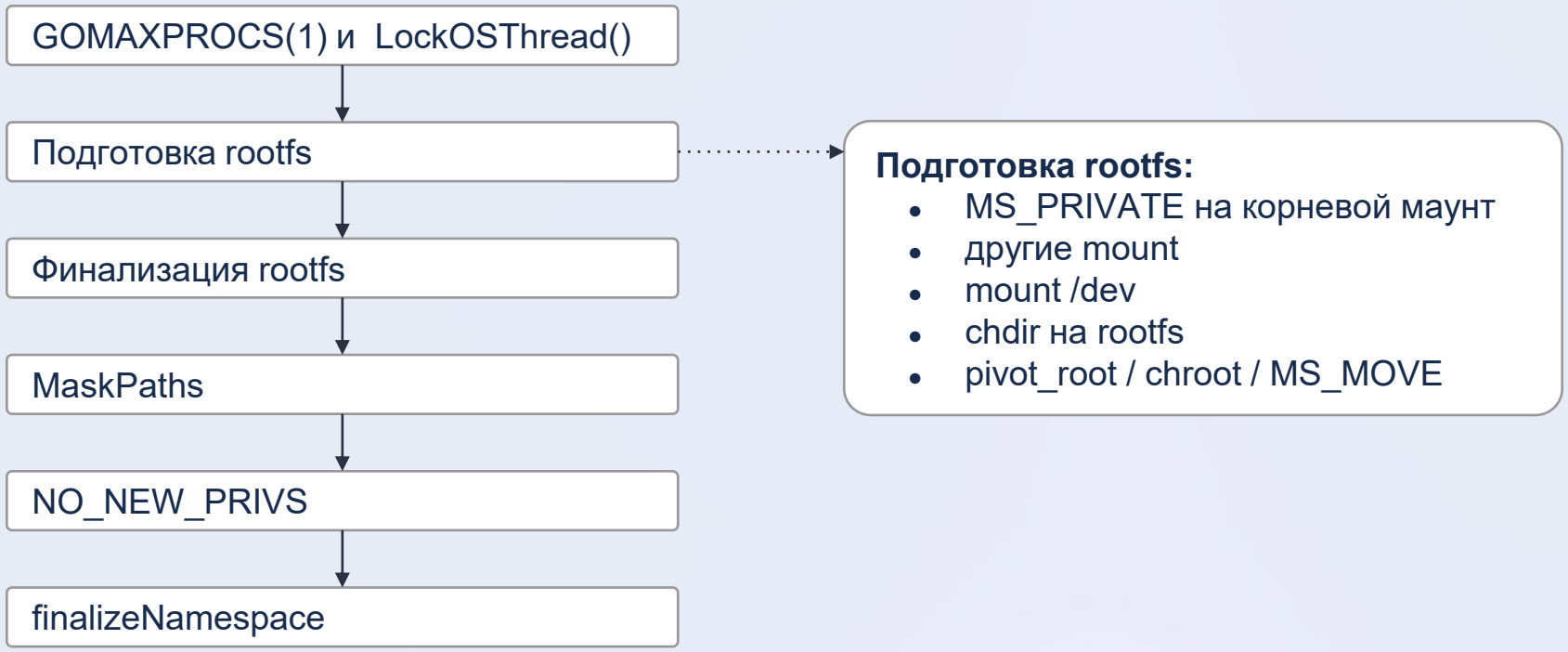
Runc

процесс инициализации контейнера



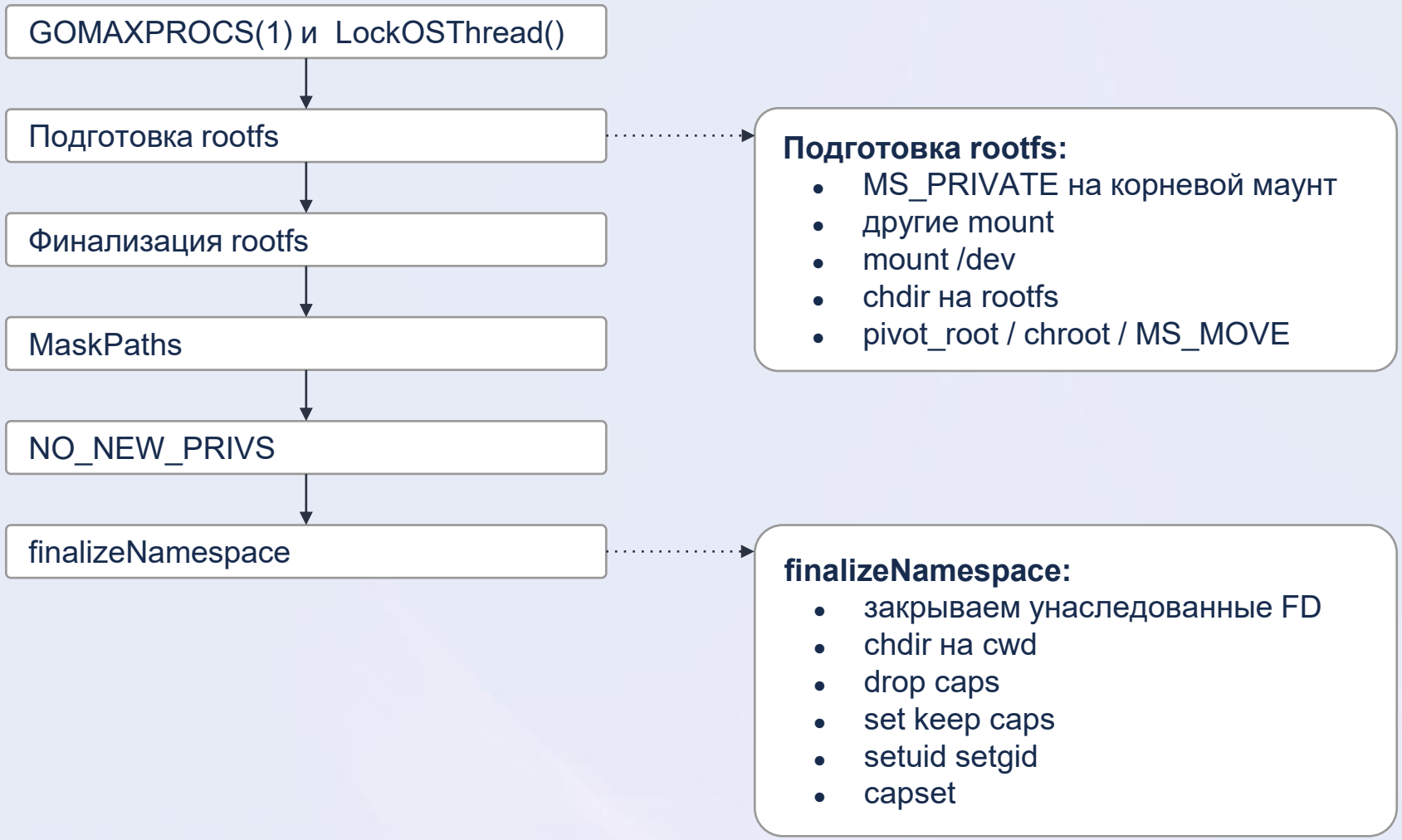
Runc

процесс инициализации контейнера



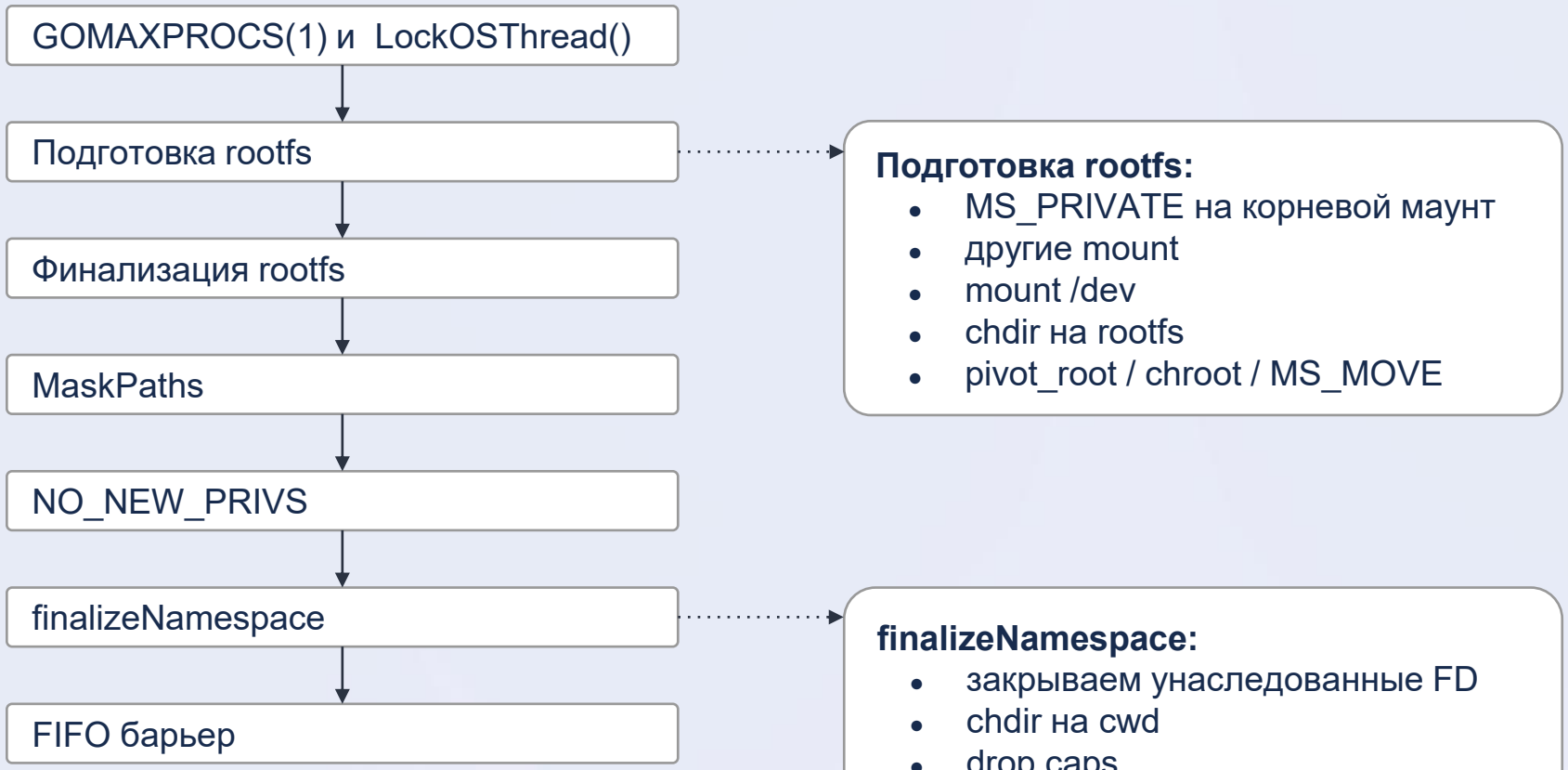
Runc

процесс инициализации контейнера



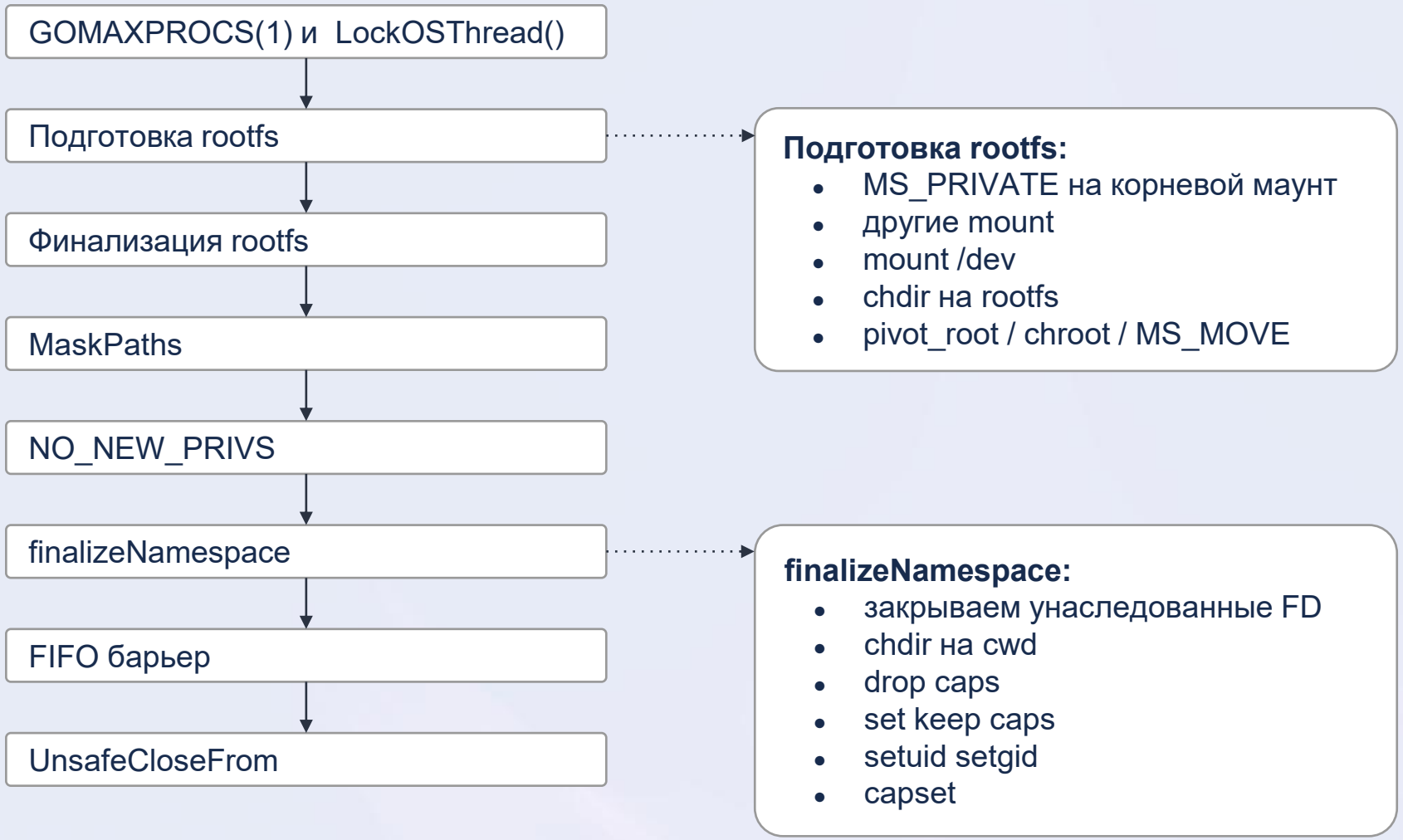
Runc

процесс инициализации контейнера



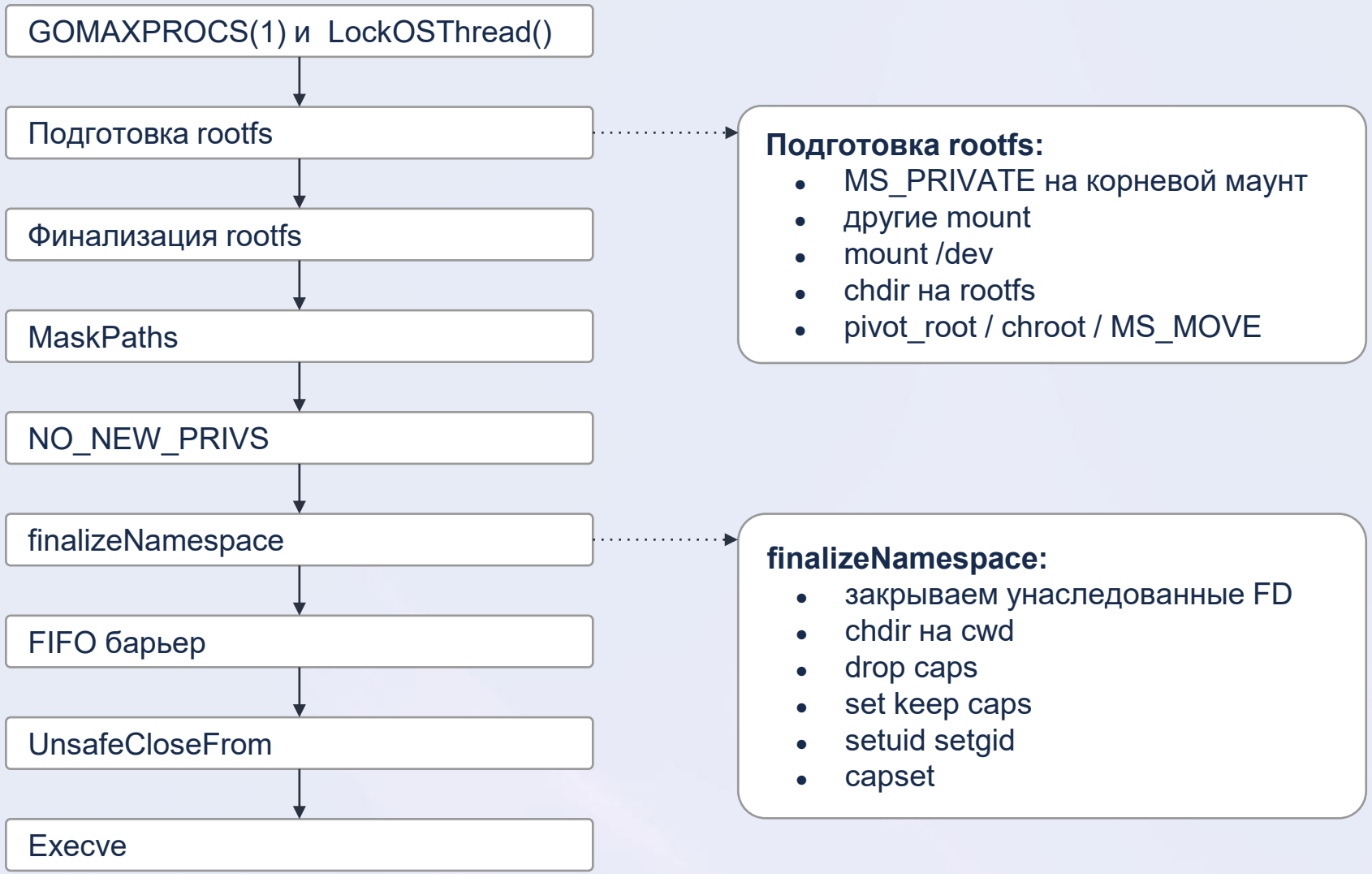
Runc

процесс инициализации контейнера



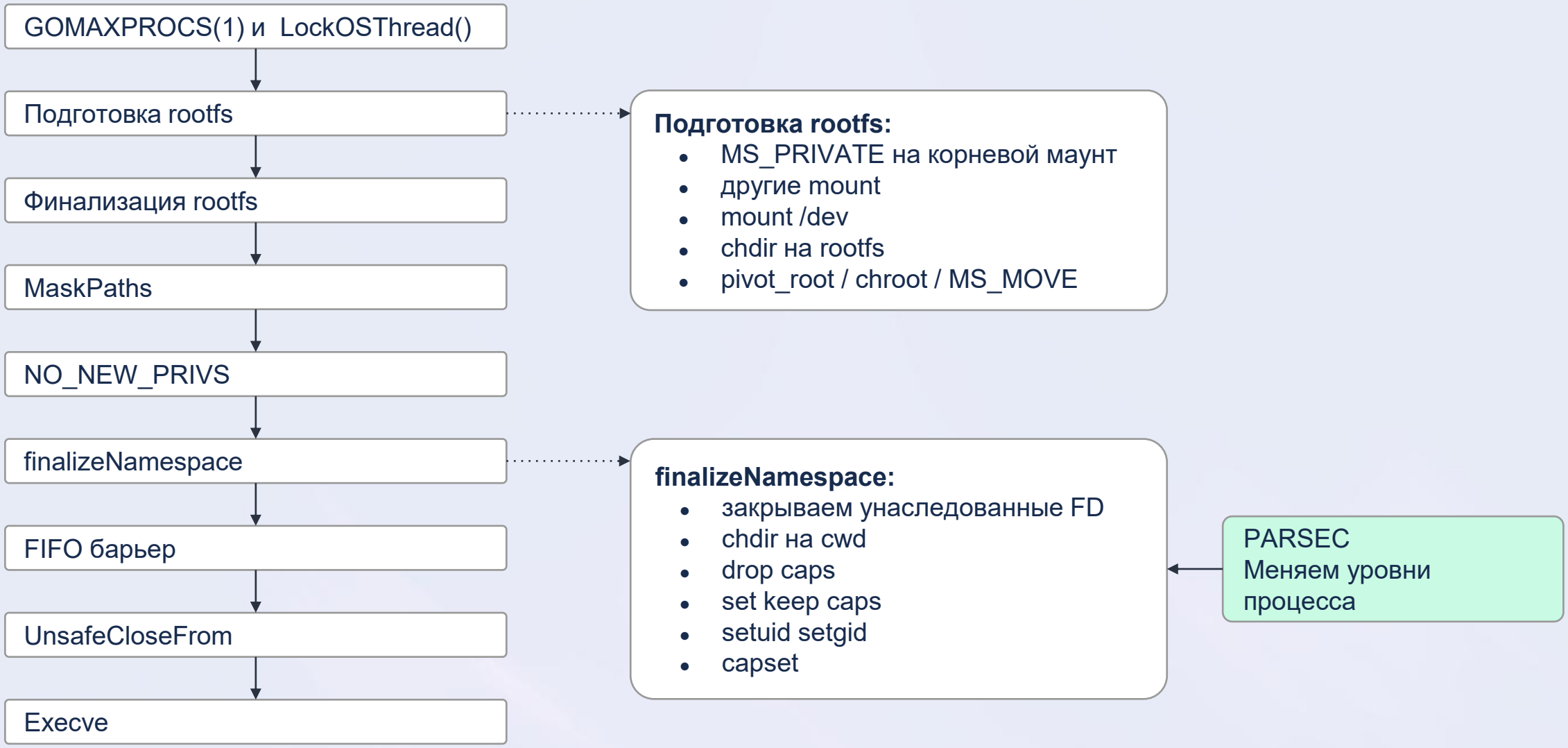
Runc

процесс инициализации контейнера



Runc

процесс инициализации контейнера



Практика

06

Bad pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: everything-allowed-exec-pod
  labels:
    app: pentest
spec:
  hostNetwork: true
  hostPID: true
  hostIPC: true
  tolerations:
  - key: node-role.kubernetes.io/control-plane
    operator: Exists
    effect: NoSchedule
  containers:
  - name: everything-allowed-pod
    image: ubuntu
    securityContext:
      privileged: true
    volumeMounts:
    - mountPath: /host
      name: noderoot
    command: [ "/bin/sh", "-c", "--" ]
    args: [ "while true; do sleep 30; done;" ]
  volumes:
  - name: noderoot
    hostPath:
      path: /
```

Bad pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: everything-allowed-exec-pod
  labels:
    app: pentest
spec:
  hostNetwork: true
  hostPID: true
  hostIPC: true
  tolerations:
  - key: node-role.kubernetes.io/control-plane
    operator: Exists
    effect: NoSchedule
  containers:
  - name: everything-allowed-pod
    image: ubuntu
    securityContext:
      privileged: true
    volumeMounts:
    - mountPath: /host
      name: noderoot
    command: [ "/bin/sh", "-c", "--" ]
    args: [ "while true; do sleep 30; done;" ]
  volumes:
  - name: noderoot
    hostPath:
      path: /
```

Контейнер может ВСЁ!

Bad pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: everything-allowed-exec-pod
  labels:
    app: pentest
spec:
  hostNetwork: true
  hostPID: true
  hostIPC: true
  tolerations:
  - key: node-role.kubernetes.io/control-plane
    operator: Exists
    effect: NoSchedule
  containers:
  - name: everything-allowed-pod
    image: ubuntu
    securityContext:
      privileged: true
    volumeMounts:
    - mountPath: /host
      name: noderoot
    command: [ "/bin/sh", "-c", "--" ]
    args: [ "while true; do sleep 30; done;" ]
  volumes:
  - name: noderoot
    hostPath:
      path: /
```

Контейнер может ВСЁ!

с PARSEC по умолчанию контейнер с нулевыми уровнями

Bad pod

Защита хостовой файловой системы:

```
sh-5.2# echo 1 > /host/proc/sys/net/nf_conntrack_max
sh: /host/proc/sys/net/nf_conntrack_max: Permission denied
sh-5.2# echo BAD >  /etc/kubernetes/manifests/newpod.yaml
sh: /etc/kubernetes/manifests/newpod.yaml: Permission denied
```

Bad pod

Защита хостовой файловой системы:

```
sh-5.2# echo 1 > /host/proc/sys/net/nf_conntrack_max
sh: /host/proc/sys/net/nf_conntrack_max: Permission denied
sh-5.2# echo BAD >  /etc/kubernetes/manifests/newpod.yaml
sh: /etc/kubernetes/manifests/newpod.yaml: Permission denied
```

Защита systemd:

```
sh-5.2# systemctl status kubelet
● kubelet.service - kubelet: The kubelet Node Agent
     ...
sh-5.2# systemctl stop kubelet
Failed to stop kubelet.service: Access denied
See system logs and 'systemctl status kubelet.service' for details.
```

Bad pod

Защита хостовой файловой системы:

```
sh-5.2# echo 1 > /host/proc/sys/net/nf_conntrack_max
sh: /host/proc/sys/net/nf_conntrack_max: Permission denied
sh-5.2# echo BAD >  /etc/kubernetes/manifests/newpod.yaml
sh: /etc/kubernetes/manifests/newpod.yaml: Permission denied
```

Защита systemd:

```
sh-5.2# systemctl status kubelet
● kubelet.service - kubelet: The kubelet Node Agent
     ...
sh-5.2# systemctl stop kubelet
Failed to stop kubelet.service: Access denied
See system logs and 'systemctl status kubelet.service' for details.
```

Защита от запуска:

```
sh-5.2# whereis runc
runc: /usr/local/sbin/runc
sh-5.2# runc run spy
sh: runc: command not found
```

Bad pod

Защита хостовой файловой системы:

```
sh-5.2# echo 1 > /host/proc/sys/net/nf_conntrack_max
sh: /host/proc/sys/net/nf_conntrack_max: Permission denied
sh-5.2# echo BAD >  /etc/kubernetes/manifests/newpod.yaml
sh: /etc/kubernetes/manifests/newpod.yaml: Permission denied
```

Защита systemd:

```
sh-5.2# systemctl status kubelet
● kubelet.service - kubelet: The kubelet Node Agent
     ...
sh-5.2# systemctl stop kubelet
Failed to stop kubelet.service: Access denied
See system logs and 'systemctl status kubelet.service' for details.
```

Защита от запуска:

```
sh-5.2# whereis runc
runc: /usr/local/sbin/runc
sh-5.2# runc run spy
sh: runc: command not found
```

Защита от чтения (с флагом ssi):

```
sh-5.2# cat /var/lib/kubelet/config.yaml
cat: /var/lib/kubelet/config.yaml: permission denied
```

Итоги

07

Итоги

БЕКОН'26

Мандатные метки успешно передаются через OCI-аннотации без изменений в Kubernetes

Containerd пробрасывает метки в аннотации и ставит метки для снапшота

В Runc реализован патч для установки мандатных меток процесса

PARSEC защищает хостовую файловую систему и системные компоненты от привилегированного контейнера

БЕКОН '26

КОНФЕРЕНЦИЯ ПО БЕЗОПАСНОСТИ КОНТЕЙНЕРОВ И КОНТЕЙНЕРНЫХ СРЕД

bekon.luntry.ru

bootsman.tech

astragroup

gruppa_astra