

5 июня 2024 📍 Москва, LOFT HALL#2

БЕКОН²⁴

Конференция по БЕзопасности
КОНтейнеров и контейнерных сред

Мультитенантность в K8s, есть ли серебряная пуля?

Аксёнов Константин

Руководитель разработки Deckhouse,
Флант



Константин Аксёнов

Руководитель разработки
Deckhouse

✉ konstantin.aksenov@flant.ru
🐙 github.com/konstantin-axenov

Чем занимаюсь

Больше 5 лет засыпаю и просыпаюсь с мыслями о Kubernetes.
Выступал на всех конференциях Бекон :)

Опыт

С 2011 занимаюсь разработкой.

С 2017 работаю в компании «**Флант**».

С 2020 руковожу разработкой **Deckhouse**.

С чем работаю больше всего



Проблематика

- Приложения могут влиять на доступность друг друга
- Например, под с более высоким приоритетом вытесняет другие поды с общего узла

```
bekon-2024 $kubectl get pods -o wide -A | grep prod
```

```
bekon-prod      nginx-78c4d8cdcb-78h8f      1/1      Running      0          68s      10.222.1.47      capsule-node-t3wq6
```

```
bekon-2024 $kubectl create -f - << EOF
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: my-greedy-pod
```

```
  namespace: my-best-prod
```

```
spec:
```

```
  containers:
```

```
  - name: nginx
```

```
    image: nginx
```

```
    resources:
```

```
      requests:
```

```
        memory: "1.5Gi"
```

```
  priorityClassName: system-cluster-critical
```

```
EOF
```

```
pod/my-greedy-pod created
```

```
bekon-2024 $kubectl get pods -o wide -A | grep prod
```

```
bekon-prod      nginx-78c4d8cdcb-xff9s      0/1      Pending      0          6s      <none>          <none>
```

```
my-best-prod_  my-greedy-pod              1/1      Running      0          6s      10.10.131.91    capsule-node-t3wq6
```


Проблематика

- Приложения могут выходить за пределы контейнера
- Например, из-за нарушения политик безопасности можно запустить под, через который злоумышленник сможет получить доступ к другим подам кластера

```

bekon-2024 $kubectl get pods -o wide -A | grep prod
bekon-prod          nginx-78c4d8cdcb-xff9s          1/1      Running    0          8m35s    10.222.1.47    capsule-node-t3wq6
my-innocent-prod    target-6c858d7666-nqkch          1/1      Running    0          4m19s    10.10.131.92   capsule-node-t3wq6
bekon-2024 $kubectl -n bekon-prod get pod nginx-78c4d8cdcb-xff9s -o yaml | grep "mountPath: /host\|volumes" -A 2
  - mountPath: /host
    name: noderooot
  - mountPath: /var/run/secrets/kubernetes.io/serviceaccount
--
volumes:
- hostPath:
  path: /
bekon-2024 $kubectl -n bekon-prod exec -ti nginx-78c4d8cdcb-xff9s -- chroot /host
# crictl exec -ti 39df873463f82 ip -4 a | grep "inet 10"
DEBU[0000] get runtime connection
DEBU[0000] ExecRequest: &ExecRequest{ContainerId:39df873463f82,Cmd:[ip -4 a],Tty:true,Stdin:true,Stdout:true,Stderr:false,}
DEBU[0000] ExecResponse: &ExecResponse{Url:http://127.0.0.1:36247/exec/LnW7buby,}
DEBU[0000] Exec URL: http://127.0.0.1:36247/exec/LnW7buby
DEBU[0000] StreamOptions: {0xc00011e000 0xc00011e008 0xc00011e010 true <nil>}
_ inet 10.10.131.92/32 scope global eth0

```

План выступления

- Определение мультитенантности
- Модели тенантности
- Доступные реализации
- Критерии выбора
- Выводы о существовании серебряной пули

Определение

Google

Мультитенантность — возможность изолированно обслуживать пользователей из разных организаций (т. е. независимых подписчиков) в рамках одного сервиса (одной инсталляции или развёртывания).

GIGA CHAT

Мультитенантная архитектура в контексте Kubernetes означает, что один кластер Kubernetes может использоваться несколькими организациями или клиентами без нарушения конфиденциальности и безопасности данных каждого клиента. **Это достигается путём разделения ресурсов кластера** между различными тенантами (организациями или клиентами), каждый из которых имеет свои собственные пространства имён, сетевые политики, ресурсы и другие настройки безопасности.

Разделение ресурсов кластера

- **Вычислительные ресурсы**

- CPU, GPU, RAM
- Хранилище (iops, throughput, максимальное количество дисков, подключаемых к одному узлу)
- Сеть (DNS, bandwidth узла кластера и внешнего балансировщика)

- **Kubernetes Control Plane**

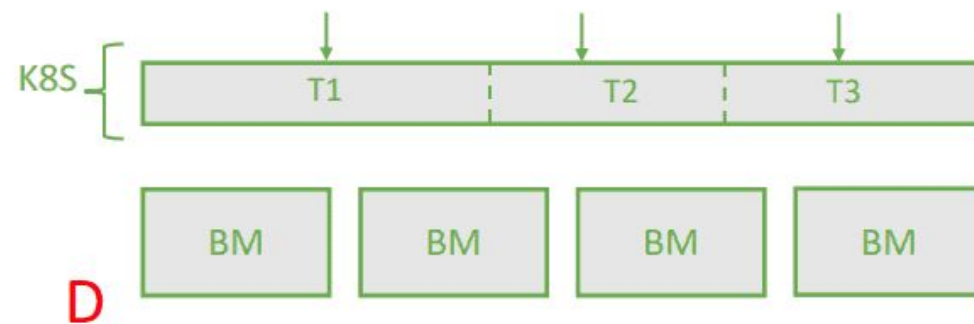
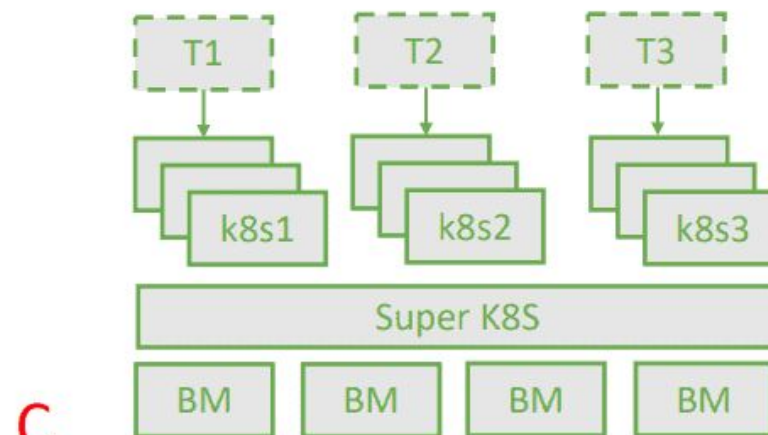
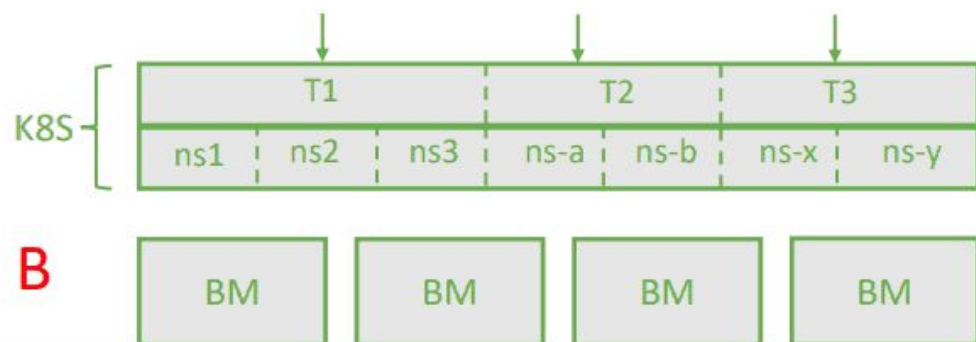
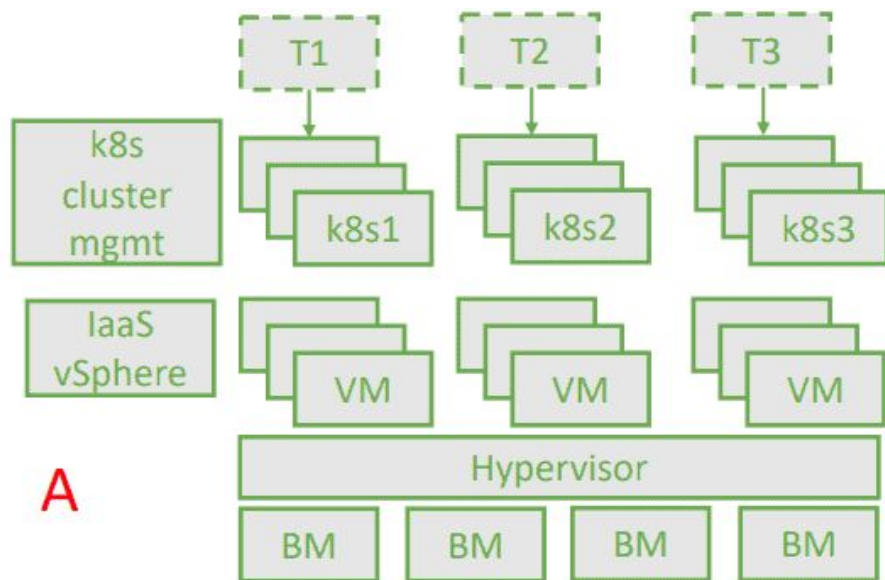
- **Внутрикластерные сервисы**

- Мониторинг (логи, метрики, трейсы)
- Ingress controller
- Операторы и контроллеры (cert-manager, service mesh и другие)

Kubernetes SIGs и документация

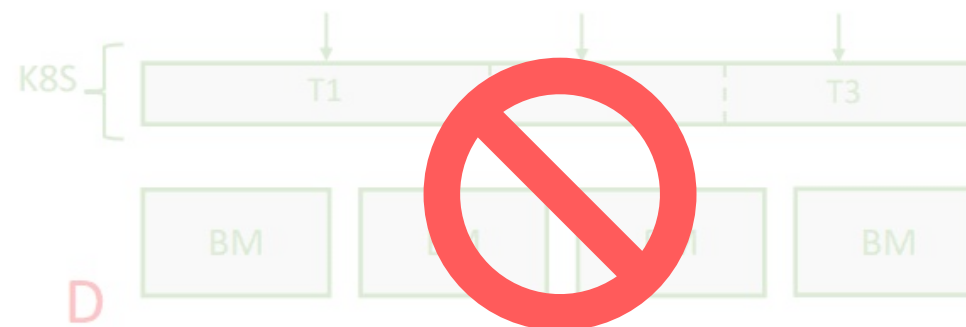
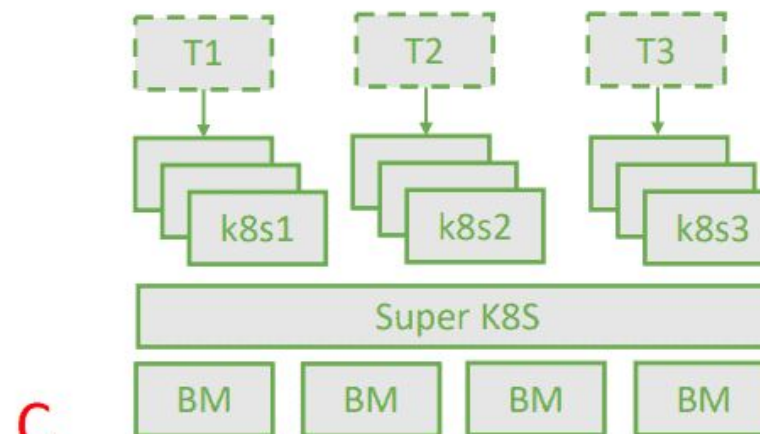
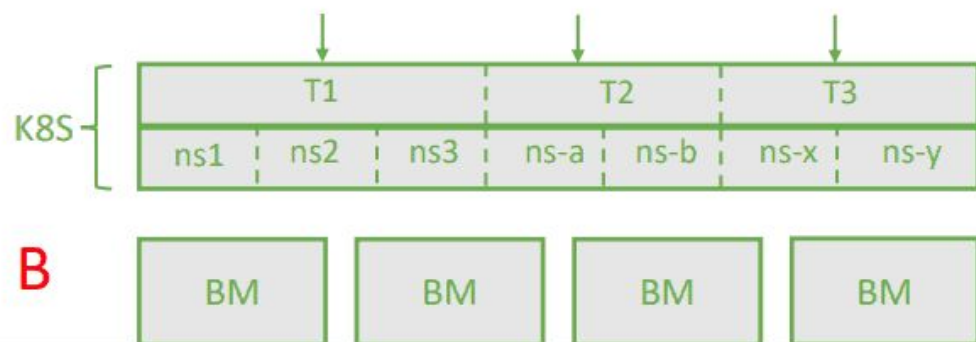
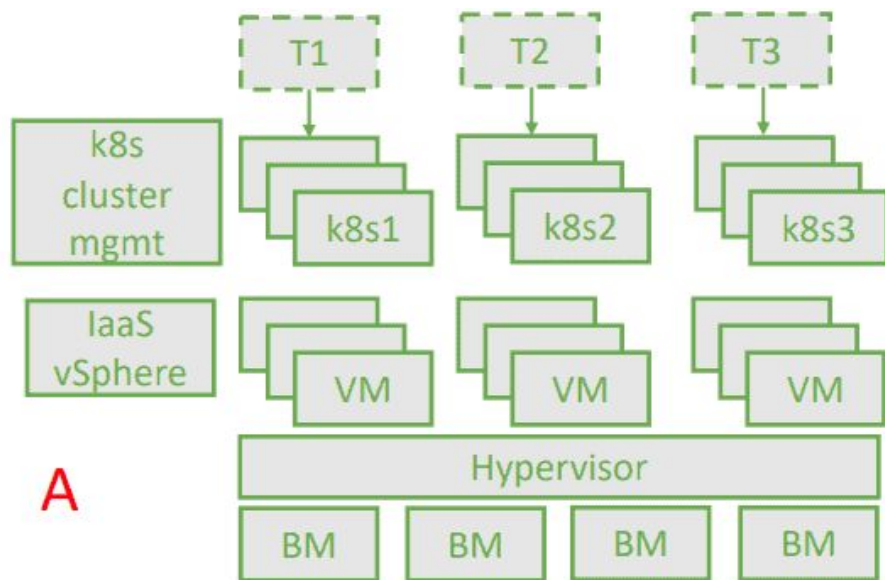
- Kubernetes Working Group for Multi-Tenancy (retired)
 - Три модели тенантности для Kubernetes
 - Hierarchical namespaces (HNC)
 - Multi-Tenancy Benchmarks
 - VirtualCluster (текущее название Cluster API Provider Nested)
- Мультитенантность — обзор вариантов и лучших практик
- Multicluster SIG
 - Kubefed
 - Cluster Registry

Multi-tenancy Architecture Options



Источник: K8s Multi-tenancy WG – Deep Dive KubeCon Europe 2019

Multi-tenancy Architecture Options



Источник: K8s Multi-tenancy WG – Deep Dive KubeCon Europe 2019

Три модели тенантности для Kubernetes

Множество кластеров

- Clusters as a Service (вариант А)
 - Cluster API
 - Managed Kubernetes-сервисы от облачных провайдеров
 - Kubernetes-дистрибутивы (OpenShift, Rancher, Deckhouse и другие)

Три модели тенантности для Kubernetes

Множество кластеров

- Clusters as a Service (вариант А)
 - Cluster API
 - Managed Kubernetes-сервисы от облачных провайдеров
 - Kubernetes-дистрибутивы (OpenShift, Rancher, Deckhouse и другие)

Всегда найдётся такое небольшое приложение или команда, делать для которых отдельный кластер будет просто избыточно

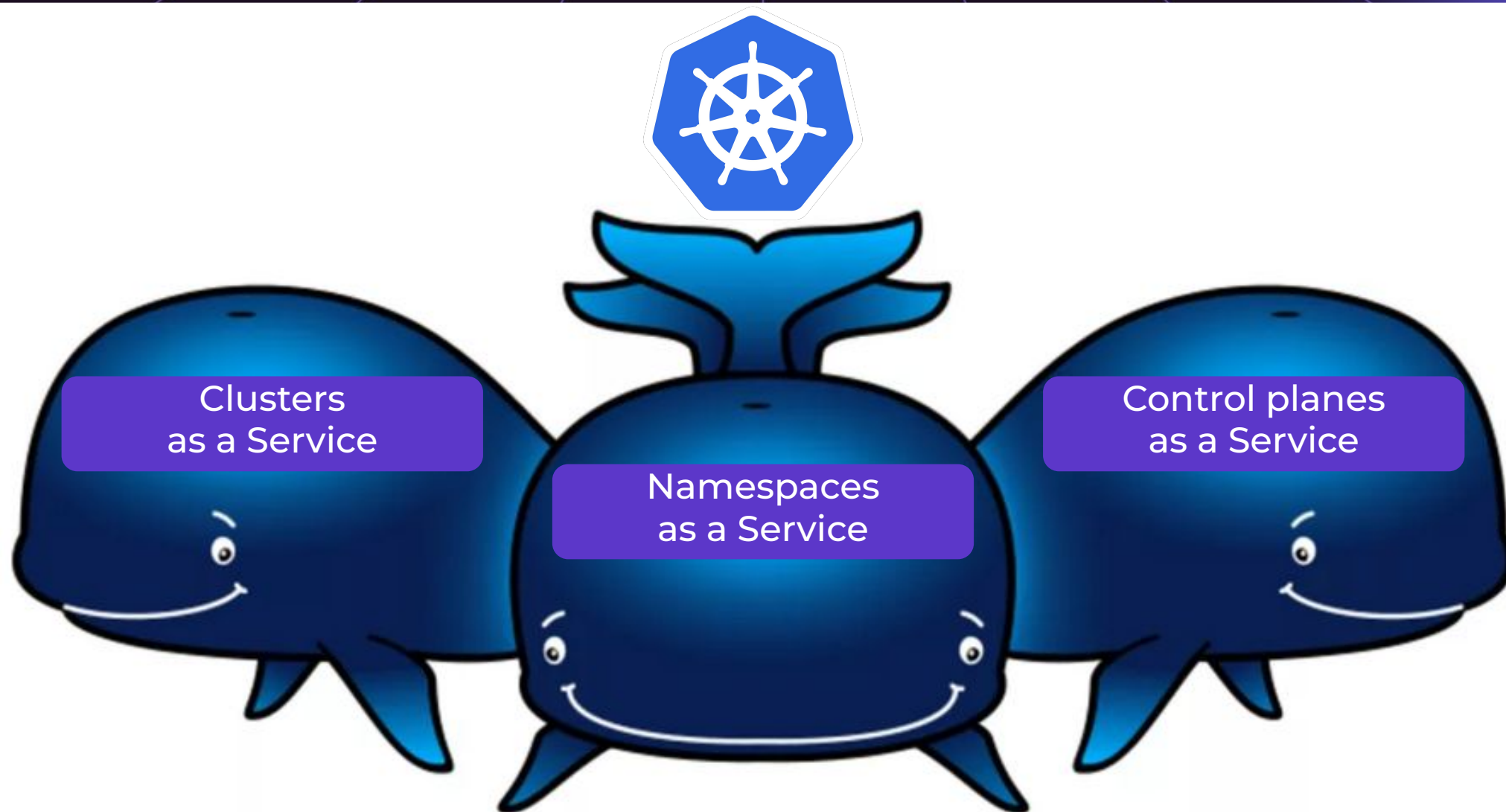
Три модели тенантности для Kubernetes

Множество кластеров

- Clusters as a Service (вариант А)
 - Cluster API
 - Managed Kubernetes-сервисы от облачных провайдеров
 - Kubernetes-дистрибутивы (OpenShift, Rancher, Deckhouse и другие)

Shared-кластер

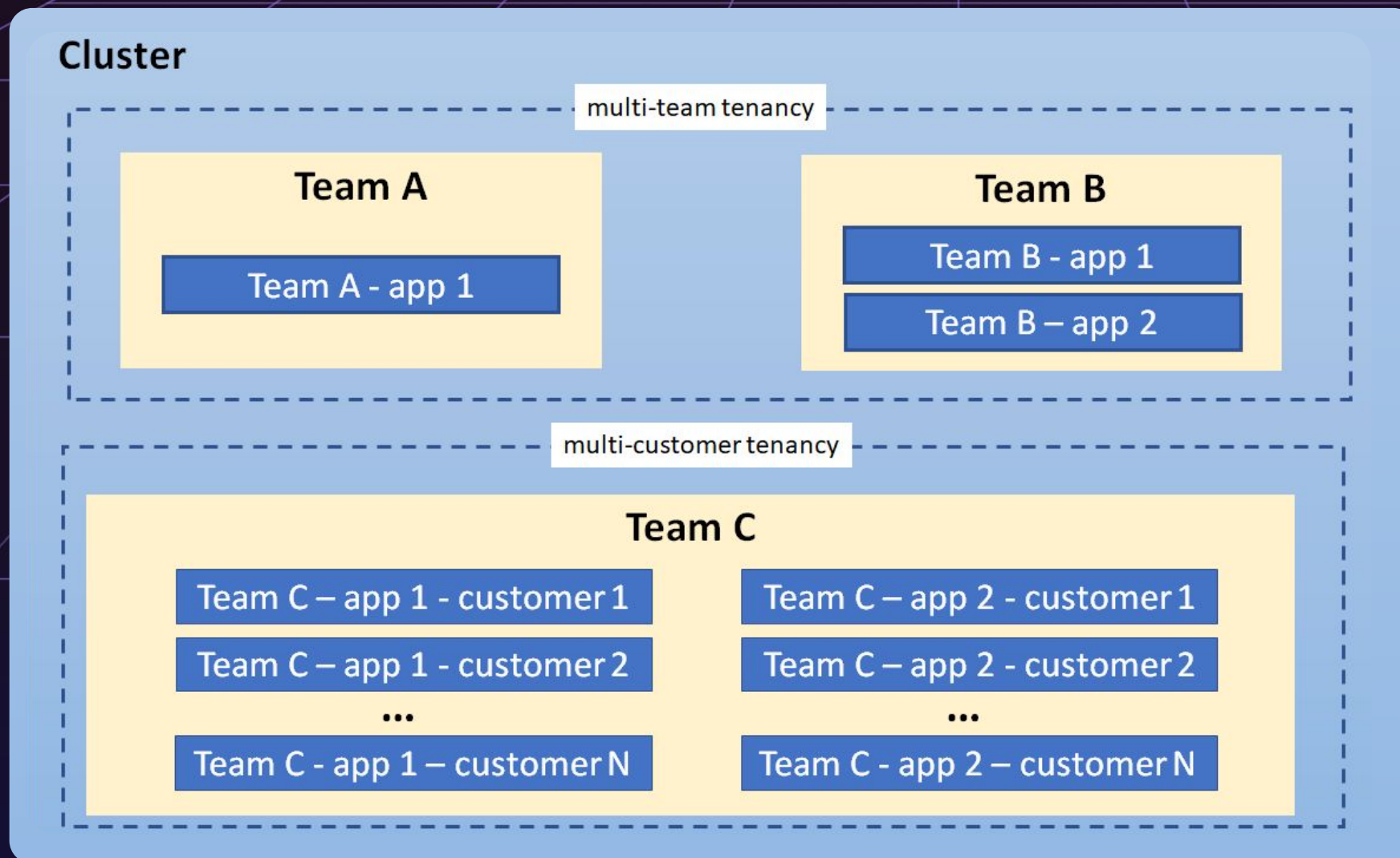
- Namespaces as a Service (вариант В)
 - Hierarchical Namespace Controller (HNC)
 - Capsule
- Control planes as a Service (вариант С)
 - Cluster API Provider Nested (бывший VirtualCluster от SIG Multi-Tenancy)
 - vCluster от Loft Labs
 - [Kamaji](#)



Мультитенантность: когда требуется?

- Множество команд
- Множество клиентов (например, SaaS)
- Другие примеры с разделением ресурсов:
 - Предоставление Kubernetes как сервиса для команды разработки
 - Один кластер под dev-, stage- и prod-окружения
 - Множество приложений
- Понимание реализации изоляции в рамках shared-кластера может потребоваться, даже если вы выбрали стратегию Clusters as a Service

Варианты мультитенантности в shared-кластере



Источник: kubernetes.io/docs/concepts/security/multi-tenancy/#tenants

Виды изоляции в shared-кластере

«soft» multi-tenancy

Namespace с настроенными сетевыми политиками — это всё ещё «soft»?

«hard» multi-tenancy

Выделенные worker-узлы — это уже достаточно «hard»?

Виды изоляции в shared-кластере

«soft» multi-tenancy

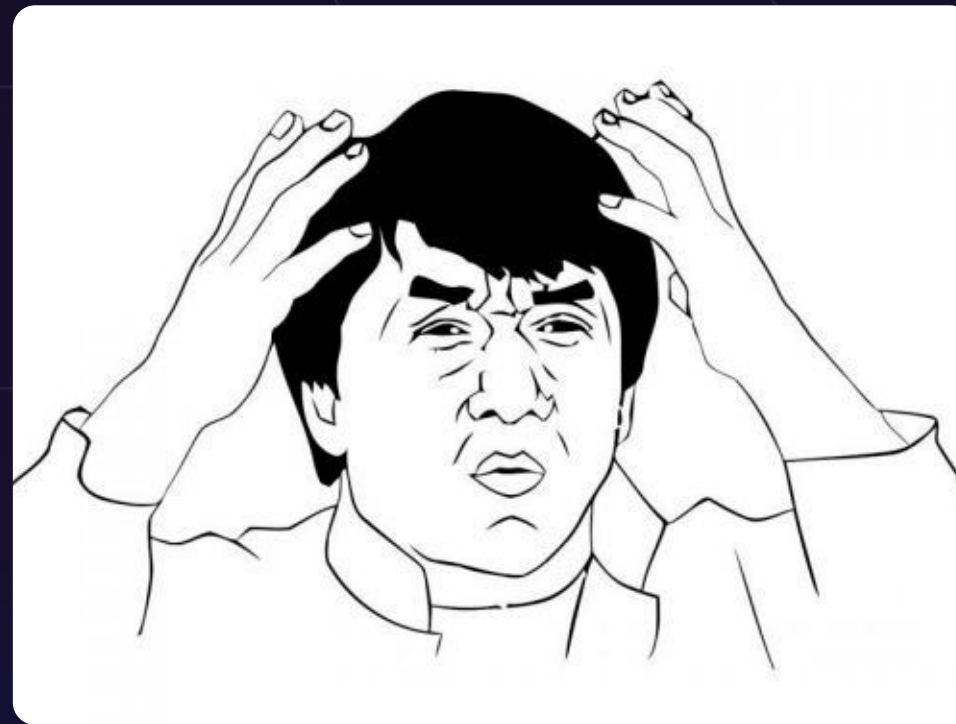
Namespace с настроенными сетевыми политиками — это всё ещё «soft»?



«hard» multi-tenancy

Выделенные worker-узлы — это уже достаточно «hard»?

However, the terms "hard" and "soft" can often be confusing, as there is no single definition that will apply to all users.



Виды изоляции в shared-кластере

- Изоляция Control plane

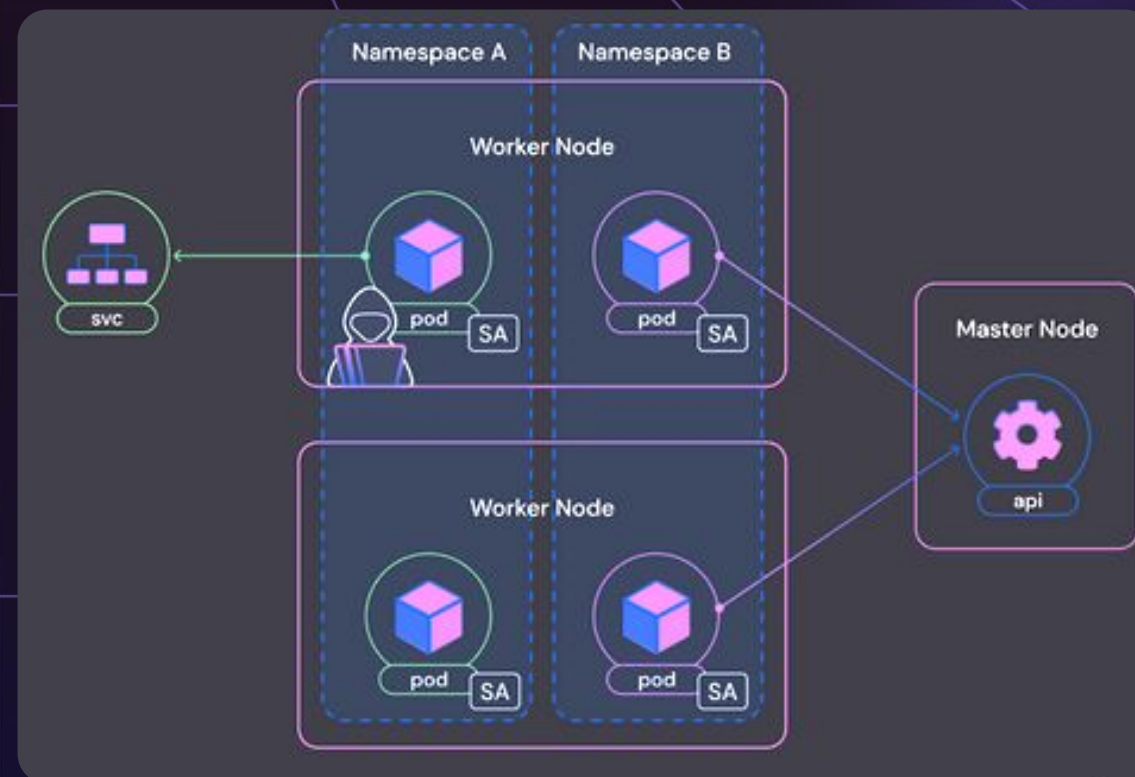
- Namespaces
- Access control
- Quotas

- Изоляция Data plane

- Network
- Storage
- Sandboxing containers
- Node

- Дополнительно

- API Priority and Fairness
- DNS
- Quality-of-Service
- Operators



Источник: www.peach.wiz.io/kubernetes

Изоляция Control plane

- **Namespaces**

- Группировка namespaced-объектов API, можно не волноваться, что имена пересекутся
- Индивидуальные настройки безопасности для namespace
- Логическое разделение запускаемых нагрузок

- **Access control**

- Role-based access control (RBAC) для пользователей и приложений
- Принцип наименьших привилегий

- **Quotas**

- Resource Quotas (Compute, Storage, Object Count) на namespace
- Квотирование по скоупам (например, по PriorityClass)

Изоляция Data plane

- **Network**

- Сетевые политики (Network Policies)
- Необходимо контролировать как ingress-, так и egress-трафик
- Разрешать только необходимые соединения как между подами в разных namespaces, так и между подами в одном namespace

- **Storage**

- Отдельный StorageClass для каждого тенанта. Если общий StorageClass, то обязательно persistentVolumeReclaimPolicy = Delete

- **Node**

- Выделенные под тенант узлы решают проблему noisy neighbors и другие тенанты будут в большей безопасности в случае компрометации узла

- **Sandboxing containers**

- Контейнеры используют общее ядро узла, поэтому для большей изоляции можно рассмотреть gVisor или Kata Containers

Дополнительно

- API Priority and Fairness

- Позволяет управлять приоритетом при доступе к Kubernetes API из подов



Дополнительно

- **API Priority and Fairness**

- Позволяет управлять приоритетом при доступе к Kubernetes API из подов

- **Quality-of-Service**

- Ограничение bandwidth сети с помощью возможностей CNI
 - Разные StorageClasses для контроля использования Storage
 - Управление приоритетами подов и их вытеснением с узлов через PriorityClass

- **DNS**

- Возможно использование отдельного DNS для каждого тенанта с целью дополнительной изоляции

- **Operators**

- Должны поддерживать создание объектов в namespace тенанта и на выделенных узлах

5 hardening factors (P.E.A.C.H.) для Kubernetes

- Privilege Hardening
 - Нельзя давать доступ к ClusterRole
 - Ограниченный список прав RBAC
 - Установка admission controller и настройка Pod Security Standards, хотя бы Baseline
- Encryption hardening
 - Заккрытие доступа к cluster-wide-ресурсам (например, веб-хукам)
 - Отдельный StorageClass на тенант или persistentVolumeReclaimPolicy = Delete
- Authentication Hardening
 - automountServiceAccountToken = false
- Connectivity Hardening
 - Настройка Network Policies
- Hygiene
 - Пароли в выводе логов, забытые секреты и т. д.

www.peach.wiz.io/kubernetes

Findings Table								
Namespace	Severity	Confidence	Principals	Container	Pod	Type	Description	Neighbours
bekon-prod	LOW	MEDIUM	None	None	None	DOS_NO_QUOTA	There is no resource limits on this namespace - if attacker controls resource creation it can cause DoS in other namespaces.	None
bekon-prod	HIGH	HIGH	None	nginx	nginx-6bcff9c566-nrnrxq	CONTAINER_PRIVILEGED_ACCESS_TO_HOST	Container is privileged and thus can escape to worker node and access shared secrets.	{'capsule-system', 'kube-system', 'default'}
bekon-prod	HIGH	HIGH	None	nginx	nginx-6bcff9c566-nrnrxq	CONTAINER_BPF_CAPABILITY	Container has SYS_BPF capability allowing kernel-level access to other process resources, (f.e. packet capture and secret stealing).	{'capsule-system', 'kube-system', 'default'}
bekon-prod	HIGH	HIGH	None	nginx	nginx-6bcff9c566-nrnrxq	CONTAINER_PTRACE_CAPABILITY	Container has SYS_PTRACE capability allowing control of other namespace processes running on the same worker node.	{'capsule-system', 'kube-system', 'default'}
bekon-prod	HIGH	HIGH	None	nginx	nginx-6bcff9c566-nrnrxq	CONTAINER_IPC_CAPABILITY	Container has IPC_OWNER capability allowing control of other namespace processes running on the same worker node.	{'capsule-system', 'kube-system', 'default'}
bekon-prod	HIGH	HIGH	None	None	nginx-6bcff9c566-nrnrxq	POD_ESCAPE_CORE_PATTERN	Pod has access to host through sensitive volume mount /.	{'capsule-system', 'kube-system', 'default'}
bekon-prod	HIGH	MEDIUM	None	None	nginx-6bcff9c566-nrnrxq	POD_ACCESS_TO_HOST	Pod has access to host through sensitive volume mount /.	{'capsule-system', 'kube-system', 'default'}

github.com/wiz-sec-public/namespacehound

Multi-Tenancy Benchmarks

5 Passed | 12 Failed | 0 Skipped | 1 Errors |
Completed in 27.09046809s

=====				
NO	ID	TEST	RESULT	
1	MTB-PL1-CC-CPI-1	Block access to cluster resources	Passed	
2	MTB-PL1-BC-CPI-2	Block Multitenant Resources	Error	
3	MTB-PL1-BC-CPI-3	Block add capabilities	Failed	
4	MTB-PL1-BC-CPI-4	Require run as non-root user	Failed	
5	MTB-PL1-BC-CPI-5	Block privileged containers	Failed	
6	MTB-PL1-BC-CPI-6	Block privilege escalation	Failed	
7	MTB-PL1-CC-DI-1	Require always imagePullPolicy	Failed	
8	MTB-PL1-CC-FNS-1	Configure namespace resource quotas	Failed	
9	MTB-PL1-CC-FNS-2	Configure namespace object limits	Failed	
10	MTB-PL1-BC-HI-1	Block use of host path volumes	Failed	
11	MTB-PL1-BC-HI-1	Block use of NodePort services	Failed	
12	MTB-PL1-BC-HI-3	Block use of host networking and ports	Failed	
13	MTB-PL1-BC-HI-4	Block use of host PID	Failed	
14	MTB-PL1-BC-HI-5	Block use of host IPC	Failed	
15	MTB-PL1-CC-TI-1	Block modification of resource quotas	Passed	
16	MTB-PL2-BC-OPS-3	Create Role Bindings	Passed	
17	MTB-PL2-BC-OPS-4	Create Network Policies	Passed	
18	MTB-PL2-CC-TI-1	Block role privilege escalation	Passed	
19	MTB-PL1-CC-TI-2	Block access to other tenant resources	Skipped	

github.com/kubernetes-retired/multi-tenancy/tree/master/benchmarks

Имплементации для shared-кластера

- **Namespaces as a Service**

- Hierarchical Namespace Controller (HNC)
- Capsule

- **Control planes as a Service**

- Cluster API Provider Nested (бывший VirtualCluster от SIG Multi-Tenancy)
- vCluster от Loft Labs
- Kamaji

Hierarchical Namespace Controller (HNC)

• Реализация

- CRD — SubnamespaceAnchor, HierarchyConfiguration
- Mutating, Validating веб-хуки и controller

```

bekon-2024 $kubectl get ns -l hnc.x-k8s.io/included-namespace=true
NAME      STATUS   AGE
app-1     Active   54m
bekon     Active   62m
default   Active   178m
tenant-a  Active   55m
tenant-b  Active   55m
bekon-2024 $kubectl hns tree bekon
bekon
├── [s] tenant-a
│   └── [s] app-1
└── [s] tenant-b

[s] indicates subnamespaces
bekon-2024 $kubectl get subnamespaceanchors.hnc.x-k8s.io -A
NAMESPACE  NAME      AGE
bekon       tenant-a  56m
bekon       tenant-b  56m
tenant-a    app-1     55m

```

```

apiVersion: hnc.x-k8s.io/v1alpha2
kind: HierarchyConfiguration
metadata:
  creationTimestamp: "2024-05-26T15:16:14Z"
  finalizers:
    - hnc.x-k8s.io/hasSubnamespace
  generation: 2
  name: hierarchy
  namespace: bekon
  resourceVersion: "22744"
  uid: fdd79518-3ec1-43ed-b72e-465ccacd22af
spec: {}
status:
  children:
    - tenant-a
    - tenant-b_

```

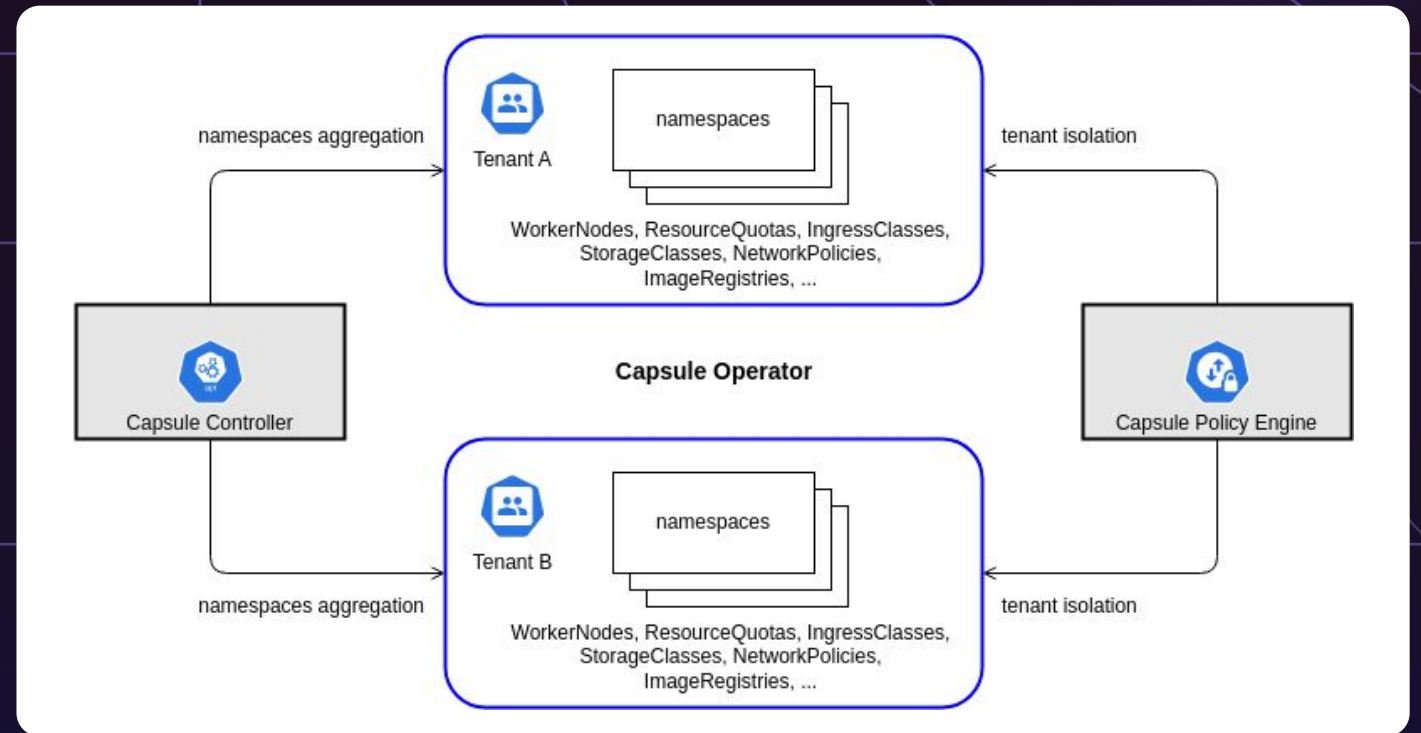
Hierarchical Namespace Controller (HNC)

- «Из коробки» не защищает ни от каких угроз
- Синхронизация ресурсов в дочерних namespaces
 - По умолчанию только RBAC, но можно настроить любые.
Есть возможность управлять режимами синхронизации
- Проект не развивается, hierarchical resource quotas так и остались в бете
- Выдавать доступы придётся через RBAC Kubernetes
- Бесит, что плагин для kubectl называется hns, а сокращение проекта hnc :)

```
bekon-2024 $kubectl get role -A | grep team-a-sre
bekon-2024 $kubectl -n tenant-a create role team-a-sre --verb=update --resource=deployments
role.rbac.authorization.k8s.io/team-a-sre created
bekon-2024 $kubectl get role -A | grep team-a-sre
app-1          team-a-sre      2024-05-26T16:14:17Z
tenant-a       team-a-sre      2024-05-26T16:14:17Z
```


Реализация

- CRD — Tenant, TenantResource, GlobalTenantResource
- Mutating и Validating веб-хуки
- Controller, Policy Engine и Proxy



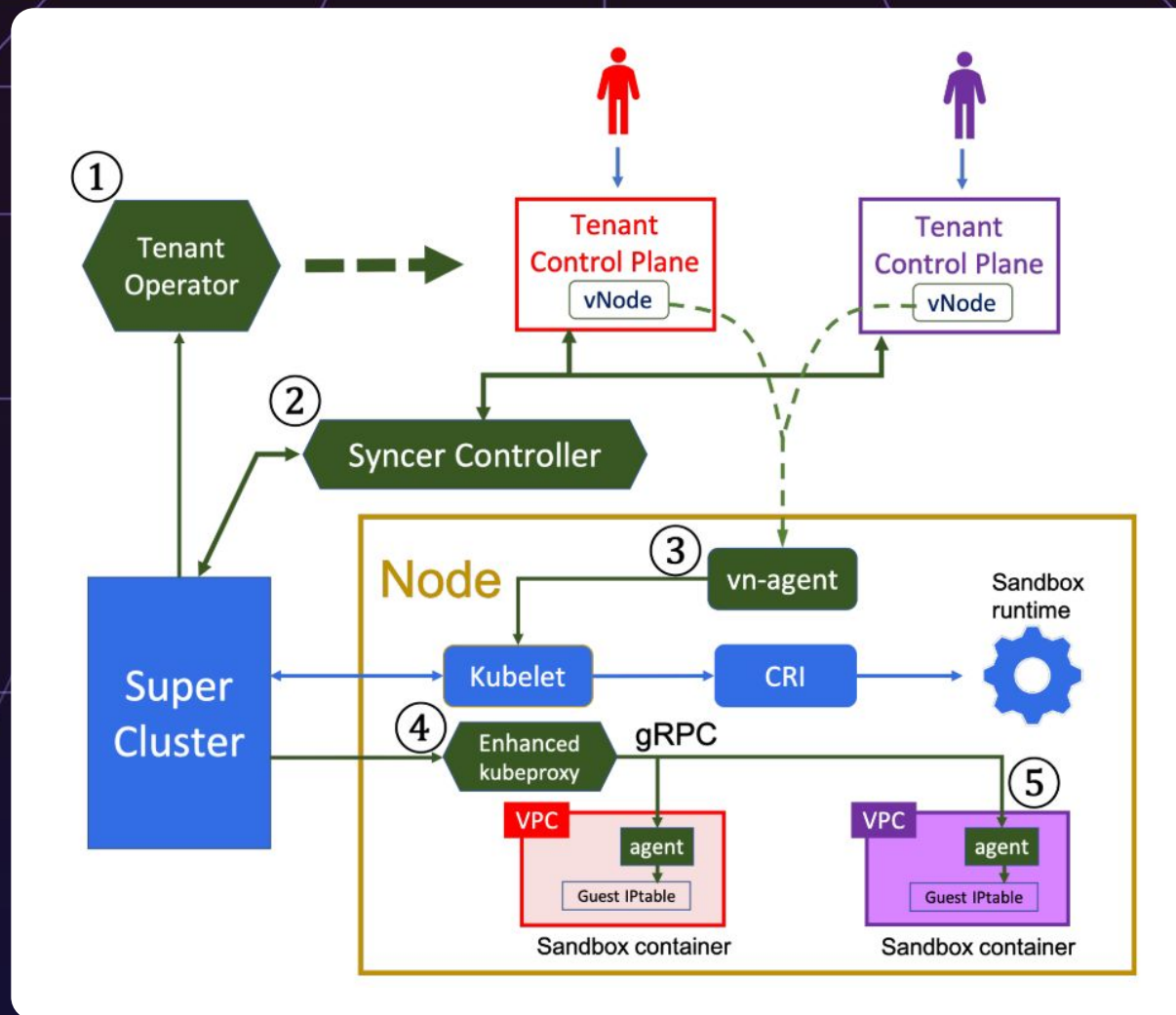
Источник: capsule.clastix.io/docs/#kubernetes-multi-tenancy-made-easy

Capsule

- Скрипт для управления пользователями тенанта (доступ к API по сертификатам)
- Очень удобная документация с понятными кейсами использования, например, есть инструкция, как настроить OIDC-аутентификацию
- Все необходимые настройки для изоляции тенантов и прохождения МТВ «из коробки» (IngressClass, NetworkPolicies, Resource Quotas, Node Isolation и др.)
- Синхронизация глобальных и внутренних ресурсов между тенантами
- Capsule Proxy, которая позволяет дать доступ к cluster-wide-ресурсам

```
bekon-2024 $kubectl --kubeconfig capsule.yaml get tenant bekon -o json | jq .status
{
  "namespaces": [
    "bekon-prod",
    "bekon-test"
  ],
  "size": 2,
  "state": "Active"
}
bekon-2024 $kubectl --kubeconfig capsule.yaml get tenant bekon
NAME      STATE    NAMESPACE QUOTA    NAMESPACE COUNT    NODE SELECTOR    AGE
bekon     Active              2                      54s
bekon-2024 $kubectl --kubeconfig bob-bekon.kubeconfig get ns
Error from server (Forbidden): namespaces is forbidden: User "bob" cannot list resource "namespaces" in API group "" at the cluster scope
bekon-2024 $kubectl --kubeconfig bob-bekon-proxy.kubeconfig get ns --insecure-skip-tls-verify
NAME          STATUS    AGE
bekon-prod    Active    64s
bekon-test    Active    66s
```

Cluster API Provider Nested

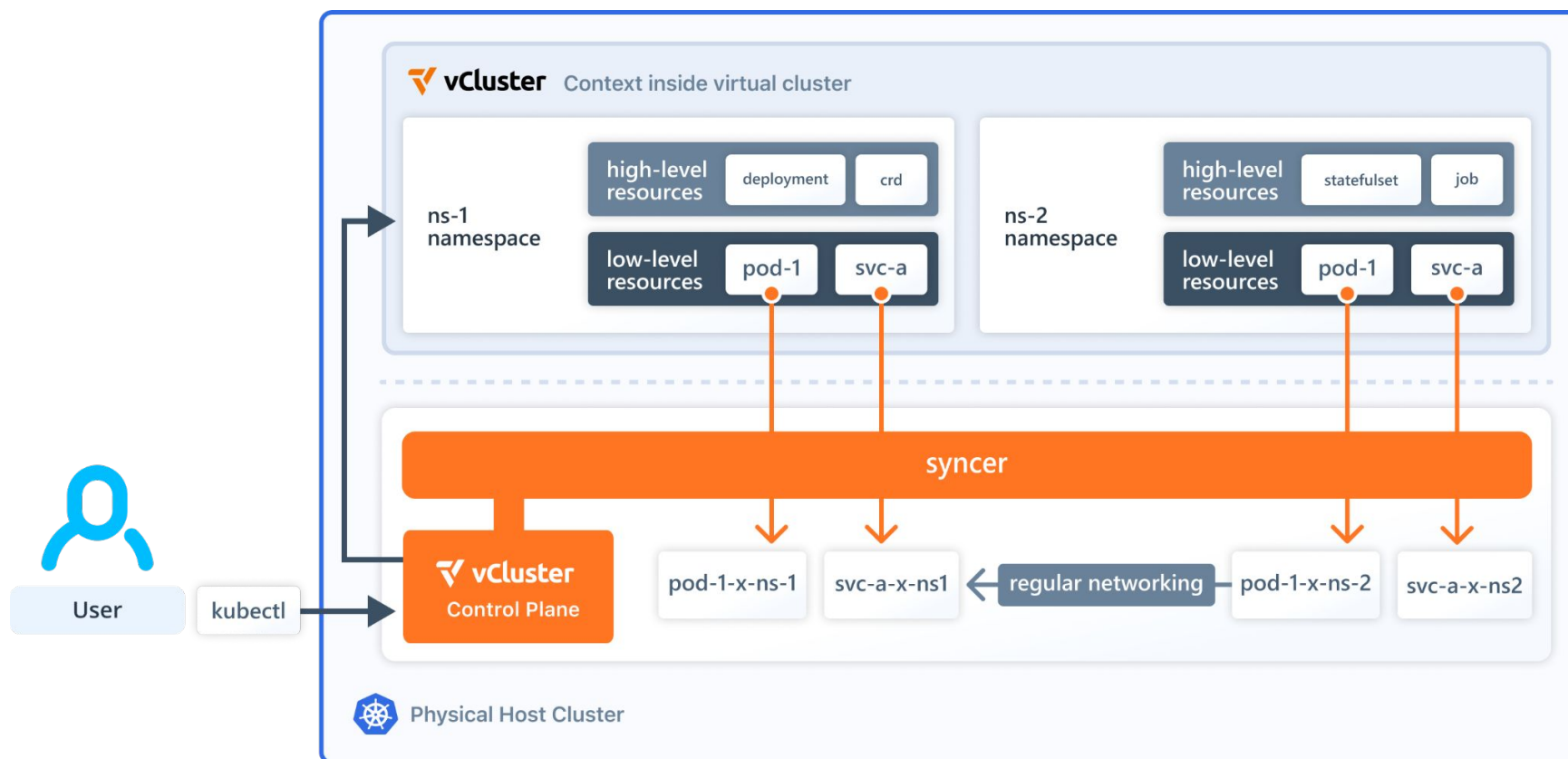


Источник: github.com/kubernetes-sigs/cluster-api-provider-nested/blob/main/virtualcluster/doc/vc-icdcs.pdf

Cluster API Provider Nested

- Последний релиз провайдера был в 2021 году
- К сожалению, не заработало с ходу

```
Error: current version of clusterctl is only compatible with v1beta1 providers, detected v1alpha4 for provider cluster-api
```

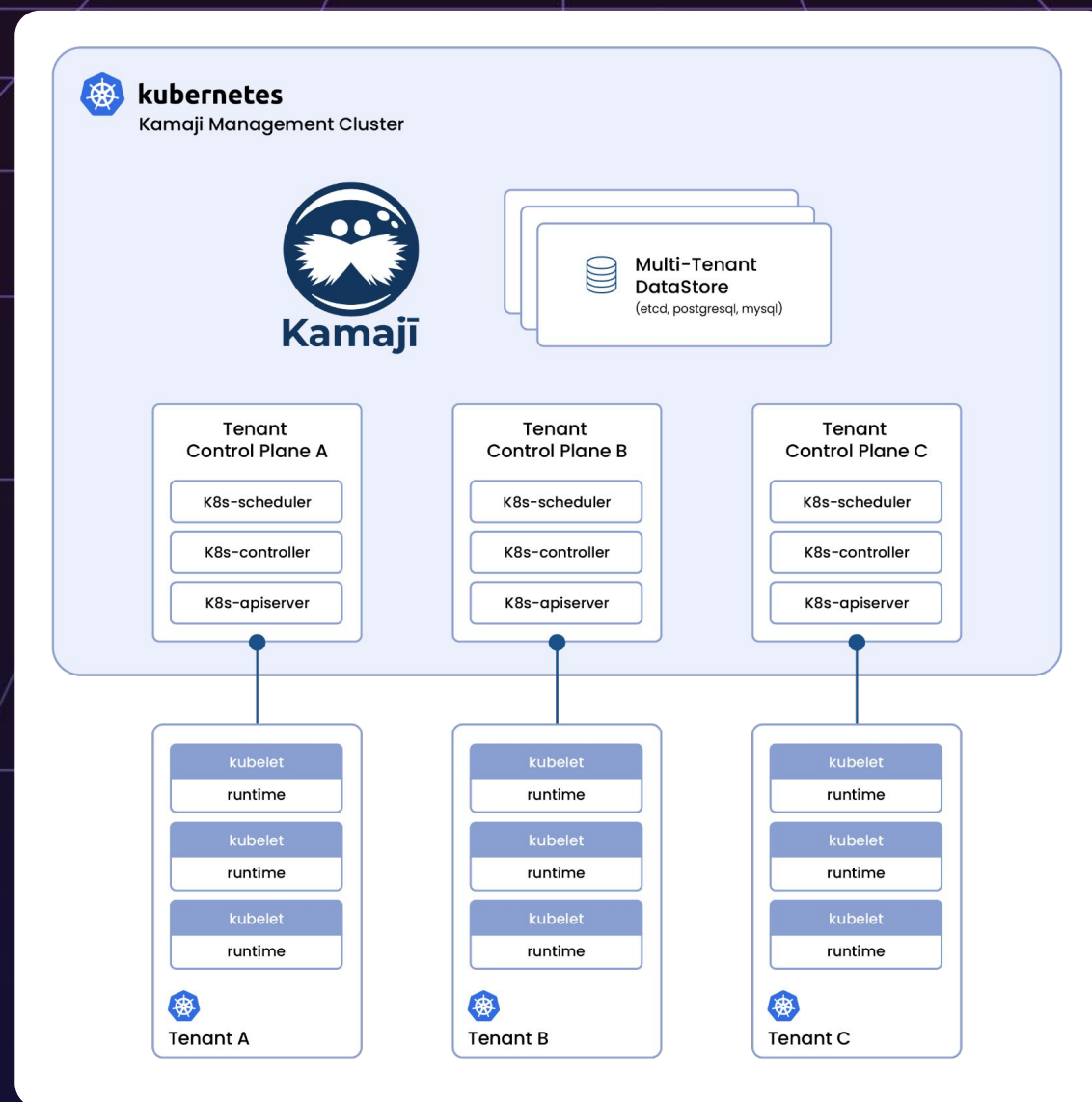



Источник: www.vcluster.com/docs

vCluster

- Отдельный coredns на виртуальный кластер
- Доступные виды изоляции «из коробки»:
 - NetworkPolicies
 - PodSecurityStandards
 - ResourceQouta, LimitRange
 - Node isolation
- SSO-, HA-режим и managed etcd только в PRO-версии

```
bekon-2024 $kubectl get pods -o wide -A | grep nginx
demo-nginx    nginx-deployment-757fc5bb5c-b52m8    1/1    Running    0        13m    10.10.241.198    vcluster-node-ninld    <I
demo-nginx    nginx-deployment-757fc5bb5c-6tmrc    1/1    Running    0        13m    10.10.241.199    vcluster-node-ninld    <I
bekon-2024 $kubectl get pods -o wide -A | grep nginx | grep prod
bekon        nginx-deployment-757fc5bb5c-6tmrc-x-demo-nginx-x-prod    1/1    Running    0        14m    10.10.241.199
bekon        nginx-deployment-757fc5bb5c-b52m8-x-demo-nginx-x-prod    1/1    Running    0        14m    10.10.241.198
```

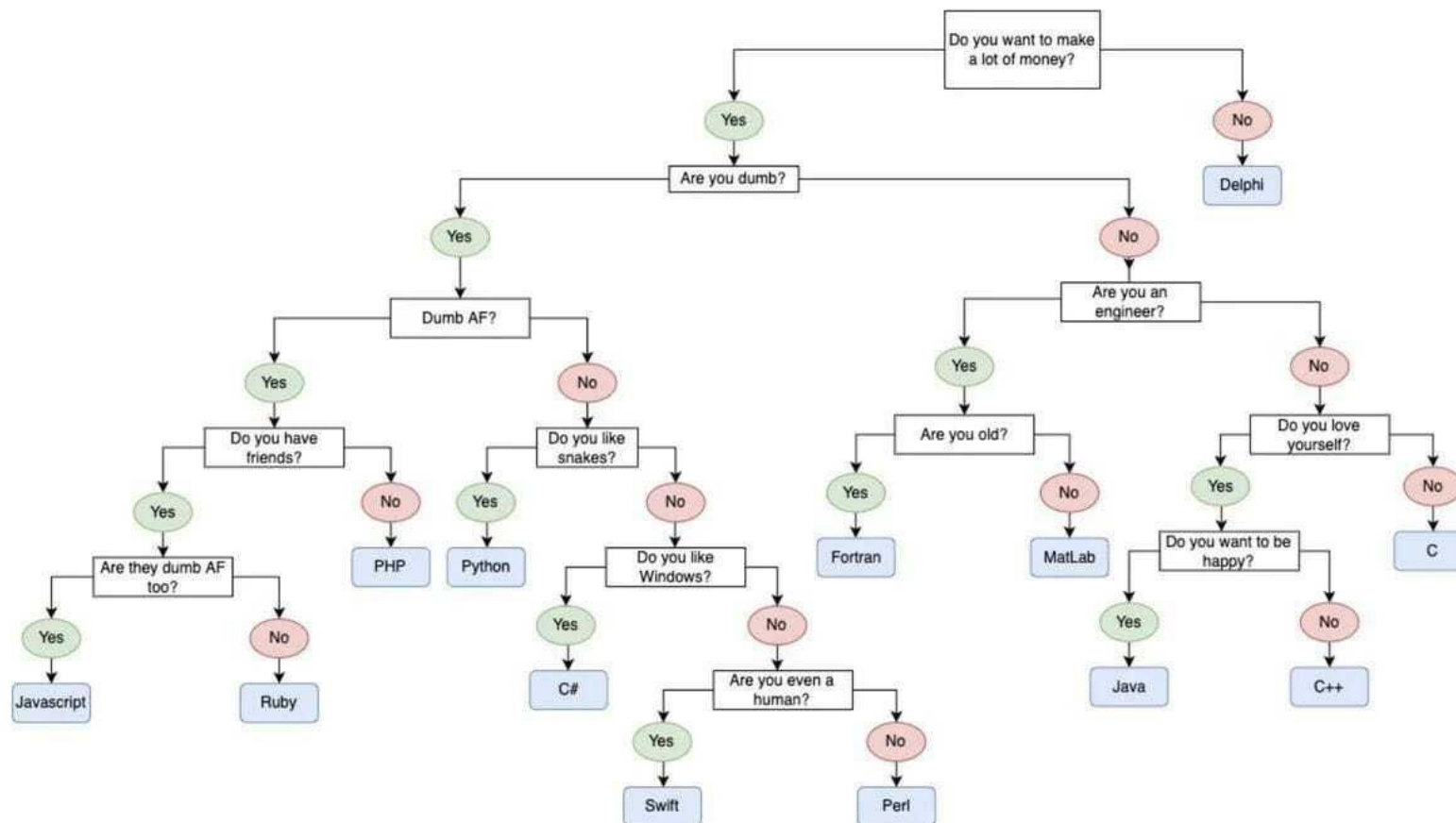


Источник: github.com/clastix/kamaji

- Тот же разработчик, что и у Capsule
- Capsule является правильным выбором благодаря балансу между функциями и простотой использования
- Отдельный Control Plane на каждый кластер
- Полная изоляция: worker-узлы только под арендатор
- Кластеры арендаторов не могут взаимодействовать друг с другом
- Необходимо разбираться с Cluster API, иначе придётся добавлять worker-узлы вручную
- Kamaji следует рассматривать как фреймворк, который позволяет вашей организации выступать в качестве поставщика Managed Kubernetes-услуг
 - <https://github.com/clastix/kamaji/discussions/402>

Какой подход к мультитенантности выбрать?


HOW TO CHOOSE YOUR PROGRAMMING LANGUAGE



Какой подход к мультитенантности выбрать?



Какой подход к мультитенантности выбрать?

	Namespace Per Tenant	 vCluster Per Tenant	Cluster Per Tenant
Isolation	very weak	strong	very strong
Access for Tenants	very restricted	vCluster admin	cluster admin
Cost	very cheap	cheap	expensive
Resource Sharing	easy	easy	very hard
Overhead	very low	very low	very high

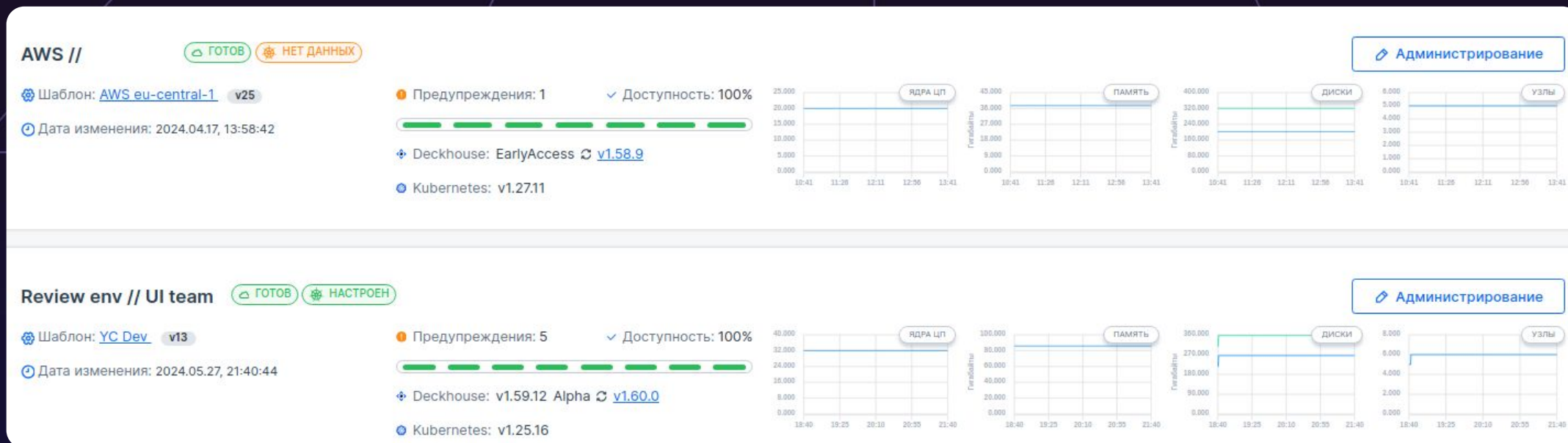
Источник: loft.sh/docs/virtual-clusters/what-are-virtual-clusters

Какой подход к мультитенантности выбрать?

Разделение ресурсов	Namespaces as a Service	Control planes as a Service	Clusters as a Service
CPU, GPU, RAM	Отдельные узлы для улучшения изоляции	Отдельные узлы для улучшения изоляции	Отличная изоляция
Хранилище	Отдельный StorageClass	Отдельный StorageClass	Отдельный StorageClass
Сеть	Сетевые политики	Сетевые политики / выделенные воркеры	Отличная изоляция
Kubernetes Control Plane	Ограничение доступа к cluster-wide-ресурсам	Отличная изоляция	Отличная изоляция
Мониторинг	Потребуется сложная конфигурация	Потребуется сложная конфигурация	Отличная изоляция
Ingress Controller	Общий для экономии, разные для изоляции	Общий для экономии, разные для изоляции	Всегда отдельный на кластер, поэтому отличная изоляция
Операторы и контроллеры	Зависит от реализации оператора/контроллера	Отличная изоляция	Отличная изоляция

Гибридный подход

- Clusters as a Service + Namespaces as a Service
- Реализация в Deckhouse Kubernetes Platform
 - CRD — ProjectTemplate, Project
 - Мультиотенантный интерфейс управления кластерами Deckhouse Commander (скоро)
 - Мультиотенантные метрики / логи (скоро)



Гибридный подход

Проекты позволяют получить следующие возможности:

- Ограничение ресурсов
- Сетевая изоляция
- Автоматические алерты и сбор логов
- Выбор профиля безопасности
- Настройка администраторов проекта
- Настройка допустимых для проекта UID/GID
- Правила аудита обращения Linux-пользователей проекта к ядру
- Сканирование запускаемых образов контейнеров на наличие известных уязвимостей (CVE)
- Назначение выделенных групп узлов для запуска подов

**+ Возможность создавать
собственные шаблоны проектов**

```
apiVersion: deckhouse.io/v1alpha2
kind: Project
metadata:
  name: my-project
spec:
  description: This is an example from
the Deckhouse documentation.
  projectTemplateName: default
  parameters:
    resourceQuota:
      requests:
        cpu: 5
        memory: 5Gi
        storage: 1Gi
      limits:
        cpu: 5
        memory: 5Gi
    networkPolicy: Isolated
    podSecurityProfile: Restricted
    extendedMonitoringEnabled: true
    administrators:
      - subject: Group
        name: k8s-admins
```


Гибридный подход

- Подход применяется в других Kubernetes-дистрибутивах, например OpenShift и Rancher
- Гибридный подход был предложен в рамках доклада Multi-tenancy WG

Sample conservative reference solution for today



KubeCon



CloudNativeCon

Europe 2019

- Hybrid solution (Option A + Option B)
 - Dedicated clusters (Option A) for tenants that need hard isolation, privileges
 - Shared clusters (Option B) for non-privileged tenants/ applications

Источник: K8s Multi-tenancy WG – Deep Dive KubeCon Europe 2019

Есть ли серебряная пуля?

- Как обычно, всё зависит от задачи
- Собрав требования, необходимо рассмотреть все варианты
- Придётся оценивать не только закрытие угроз и уровень изоляции, но и:
 - Стоимость ресурсов (человеческих, в том числе)
 - Сложность реализации

Есть ли серебряная пуля?

- Как обычно, всё зависит от задачи
- Собрав требования, необходимо рассмотреть все варианты
- Придётся оценивать не только закрытие угроз и уровень изоляции, но и:
 - Стоимость ресурсов (человеческих, в том числе)
 - Сложность реализации
 - Надёжность решения (эту тему сегодня почти не затрагивали)

5 июня 2024 📍 Москва, LOFT HALL#2
Конференция по БЕзопасности
КОНтейнеров и контейнерных сред

БЕIKOИH



konstantin.aksenov@flant.ru



[konstantin_aksenov](https://www.telegram.com/konstantin_aksenov)



deckhouse.ru



[YouTube](#)



[Telegram](#)