MicroK8s vs RKE2 vs K3s

Сравнение для закрытого on-premise-кластера: Ubuntu 22.04/24.04, Oracle Linux, локальные репозитории и OCI Registry в Nexus

Состояние документации: июль 2026 года

Сравнительная таблица

Критерий	MicroK8s	RKE2	K3s
Основное назначение	Небольшие production-кластеры, edge, лаборатории, Ubuntu-инфраструктура	Production Kubernetes для дата-центров, enterprise и security-sensitive сред	Edge, небольшие кластеры, филиалы, CI, homelab, embedded
Позиционирование	«Low-ops Kubernetes» от Canonical	Enterprise-ready Kubernetes от SUSE/Rancher	Lightweight Kubernetes от SUSE/Rancher
Архитектура control plane	Kubernetes-компоненты работают как snap-managed systemd-сервисы	Один управляющий бинарник RKE2, но стандартные control-plane-компоненты запускаются как static pods через kubelet	Почти весь control plane инкапсулирован в одном процессе k3s server
Формат поставки	Snap-пакет	RPM или tarball; один основной бинарник плюс runtime/images	Один бинарник, install script или tarball
Container runtime	Встроенный containerd	Встроенный containerd	Встроенный containerd; также поддерживается cri-dockerd
Поддержка Ubuntu	Наиболее естественная и удобная платформа	Хорошая	Хорошая
Поддержка Oracle Linux	Технически возможна через snapd, но это нетипичный вариант	Хорошо подходит как systemd/RHEL-compatible ОС; при SELinux используется отдельный rke2-selinux RPM	Работает на современных RHEL-compatible системах, но может потребовать настройки NetworkManager, firewalld и SELinux
Зависимость от snapd	Обязательна	Нет	Нет
Удобство в смешанном Ubuntu/Oracle-кластере	Низкое или среднее	Высокое	Высокое
Air-gap установка	Поддерживается: snap-файлы, assertions, образы и registry mirror подготавливаются отдельно	Один из наиболее проработанных вариантов: private registry, image tarballs, Hauler, embedded mirror	Хорошо проработана: binary, image tarball, private registry или embedded mirror
Интеграция с Nexus OCI Registry	Образы можно получать через Nexus, но сам MicroK8s остаётся snap-артефактом	Очень удобная через /etc/rancher/rke2/registries.yaml	Очень удобная через /etc/rancher/k3s/registries.yaml
Хранение установочных пакетов в Nexus	Snap и assertion-файлы придётся хранить как raw artifacts либо использовать Snap Store Proxy	RPM - в Yum repository, tarball - в Raw repository, образы - в Docker repository	Binary/tarball - в Raw repository, образы - в Docker repository
Fallback во внешние registry	Настраивается в containerd вручную	Можно явно отключить default registry endpoint	Можно явно отключить default registry endpoint
Локальное P2P-распространение образов	Нет аналогичной встроенной схемы по умолчанию	Встроенный Spegel registry mirror, опционально	Встроенный Spegel registry mirror, опционально
Datastore по умолчанию	dqlite в HA-режиме; в отдельных конфигурациях etcd	Embedded etcd для типового HA-кластера; возможны внешние SQL datastore	SQLite для single-server; embedded etcd для HA; возможны MySQL, MariaDB и PostgreSQL
HA control plane	Автоматически включается при наличии трёх и более узлов	Три и более server-узла с embedded etcd	Три и более server-узла с embedded etcd либо несколько серверов с внешней БД
Разделение control-plane и worker	Возможно, но модель менее явно ориентирована на строгое разделение ролей	Полноценное и естественное разделение server/agent и control-plane/etcd ролей	Возможно разделение server/agent и отдельных server roles

Критерий	MicroK8s	RKE2	K3s
CNI по умолчанию	Calico	Обычно Canal; доступны Calico, Cilium, Flannel и Multus	Flannel плюс kube-router для NetworkPolicy
Выбор CNI	Возможен, но требует больше ручного вмешательства	Наиболее широкий штатный выбор	Flannel заменяется, но альтернативная конфигурация менее «первоклассная», чем в RKE2
Ingress	Устанавливается как addon	Управляется packaged Helm charts; в актуальной ветке выполняется переход к Traefik	Traefik включён по умолчанию
LoadBalancer для bare metal	MetalLB addon	Обычно устанавливают MetalLB, kube-vip или внешний LB	Встроенный ServiceLB; для production часто заменяется на MetalLB/kube-vip
Локальный StorageClass	hostpath-storage addon	Обычно local-path-provisioner либо внешний CSI	local-path-provisioner включён по умолчанию
Дополнительные компоненты	Большой каталог microk8s enable ...: DNS, registry, ingress, MetalLB, Ceph и другие	Меньше «демо-addons», больше ориентации на стандартные Helm/CSI/CNI-компоненты	Traefik, ServiceLB, CoreDNS, metrics-server и local-path входят в поставку
CIS hardening	Есть cis-hardening addon и документация	Ключевой приоритет проекта, профиль и подробные self-assessment guides	Есть официальный CIS Hardening Guide, но security/compliance не является главным отличием дистрибутива
FIPS	Возможности завязаны преимущественно на Ubuntu Pro и Canonical-стек	Штатная ориентация на FIPS 140-2 и hardened images	Не является основным преимуществом K3s
SELinux / RHEL-family	Менее естественный стек	Отдельный официальный пакет rke2-selinux	Поддерживается, но требует проверки host configuration
Потребление ресурсов	Низкое или среднее	Обычно выше K3s из-за более стандартной архитектуры control plane и security-компонентов	Самое низкое из трёх
Простота начальной установки	Очень высокая на Ubuntu с доступом к Snap Store	Высокая, но параметров и компонентов больше	Очень высокая
Простота air-gap эксплуатации	Средняя: нужно отдельно управлять snap supply chain	Высокая и предсказуемая	Высокая
Контроль обновлений	Через snap channels и snap refresh; в закрытой среде требуется отдельный процесс доставки snap	Полный контроль над RPM/tarball и image bundle	Полный контроль над бинарником и image bundle
Риск непреднамеренного обновления	Выше при обычном подключении к Snap Store, если не настроены refresh controls	Низкий при фиксированной версии RPM/tarball	Низкий при фиксированной версии бинарника
Совместимость с обычными Ansible/systemd-процессами	Средняя: пути и сервисы находятся внутри snap namespace	Высокая	Высокая
Интеграция с Rancher Manager	Возможна как импорт внешнего кластера	Нативная и наиболее полная	Нативная
Типичный размер кластера	Малый и средний	Средний и крупный production-кластер	Малый и средний; возможны крупные, но это не основное позиционирование
Главное преимущество	Максимально простой Kubernetes на Ubuntu	Production, security, air-gap, mixed OS, Rancher	Минимальная сложность и потребление ресурсов
Главный недостаток	Snap становится отдельным каналом поставки, плохо вписывающимся в RPM/APT/Nexus-only модель	Более тяжёлый и сложный, чем K3s	Меньше enterprise-oriented defaults и более сильные отклонения от классического upstream deployment model

Архитектурные различия

RKE2: использует модель установки одним бинарником, но запускает стандартные компоненты control plane как static pods. Это ближе к традиционной production-архитектуре Kubernetes.

K3s: инкапсулирует значительную часть control plane в единый процесс, поэтому легче по ресурсам и проще в эксплуатации.

MicroK8s: управляет Kubernetes-компонентами через snap и systemd. На Ubuntu это удобно, но в смешанной ОС-инфраструктуре появляется отдельный контур поставки snap-артефактов.

RKE2 и K3s имеют штатную схему private registry через registries.yaml, поддерживают air-gap image bundles и встроенный распределённый OCI mirror Spegel. В RKE2 security/compliance является одной из основных архитектурных целей: CIS, hardened images, SELinux и FIPS-oriented deployment.

Вывод для вашей инфраструктуры

1. RKE2 - основной рекомендуемый вариант

Для production air-gap окружения RKE2 лучше всего соответствует вашим требованиям:

- одинаковая модель установки на Ubuntu и Oracle Linux;
- RPM или tarball можно полностью контролировать через Nexus;
- OCI-образы можно зеркалировать в Nexus Docker Registry;
- нет зависимости от Snap Store;
- штатные SELinux policies для Oracle Linux;
- embedded etcd и нормальное разделение control-plane/worker;
- проще внедрять CIS hardening;
- предсказуемый процесс обновления и rollback;
- нативная интеграция с Rancher Manager.

Oracle Linux: при включённом SELinux необходимо разместить rke2-selinux и зависимости в локальном RPM-репозитории. Обычно также потребуются container-selinux, iptables-nft, libnftnl, policycoreutils и selinux-policy.

2. K3s - для небольших и менее критичных кластеров

K3s имеет смысл выбирать, когда:

- кластер небольшой;
- ресурсы узлов ограничены;
- нужна максимально простая установка;
- не требуется FIPS;
- допустима менее классическая архитектура control plane;
- приемлемы встроенные Traefik, ServiceLB и local-path-provisioner.

Для production HA следует сразу использовать embedded etcd, а не строить кластер вокруг SQLite. SQLite подходит для single-server deployment и не предназначен для нескольких server-узлов.

3. MicroK8s - только при Ubuntu-first стратегии

MicroK8s разумен, когда:

- все узлы работают на Ubuntu;
- компания уже использует snap или Snap Store Proxy;
- нужен небольшой кластер с минимальным количеством ручной настройки;
- важна интеграция с другими Canonical-продуктами.

Для смешанной среды Ubuntu + Oracle Linux и существующей Nexus-centric supply chain MicroK8s добавляет отдельный контур доставки:

```
APT/YUM repositories в Nexus
          +
OCI images в Nexus
          +
snap + assertions / Snap Store Proxy
```

У RKE2 схема проще:

```
RPM или tarball в Nexus
          +
OCI images в Nexus
          +
registries.yaml
```

Итоговая оценка

Продукт	Оценка для вашего случая
RKE2	9/10 - основной кандидат
K3s	7/10 - для небольших или edge-кластеров
MicroK8s	5/10 - приемлем при полном переходе на Ubuntu и snap

Практический выбор: RKE2 для основных production-кластеров; K3s - для небольших вспомогательных, тестовых или удалённых кластеров. MicroK8s в смешанной ОС-инфраструктуре существенных преимуществ перед ними не даёт.

Источники

- RKE2 Architecture
- RKE2 Air-Gap Install
- K3s Private Registry Configuration
- K3s Datastore Options
- Canonical MicroK8s Alternative Install Methods