k3s-io / **k3s** Public

Code | Issues 53 | Pull requests 18 | Discussions | Actions | ...

43 Branches | 1084 Tags | Go to file | Go to file | Code | ...

github-actions[bot] and **brandond** — chore: Update rancher/local-path-provisioner i...

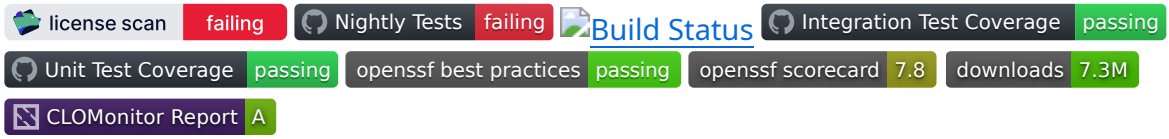
6ba341e · 19 hours ago

.agents	Agentic workflows (#14129)	2 months ago
.github	Bump docker version in CI to 29	3 days ago
cmd	Replace PreparingExecutor wit...	4 months ago
contrib	Add firewall section to check-c...	8 months ago
docs	Fix typos in documentation (#...	7 months ago
manifests	chore: Update rancher/local-p...	19 hours ago
package	Build k3s with xx to avoid QEM...	2 months ago
pkg	Add advertise-address to SANs...	yesterday
scripts	chore: Update rancher/local-p...	19 hours ago
tests	Add advertise-address to SANs...	yesterday
updatecli	updatecli: add missing require...	last month
.clomonitor.yml	Add .clomonitor.yml file so we ...	last year
.dockerignore	Faster K3s Binary Build Option...	3 years ago
.gitignore	Add sequential split-server test	11 months ago
.golangci.yml	Address new linting requireme...	2 months ago
ADOPTERS.md	Add Sophotech to the list of ad...	2 months ago
BUILDING.md	Build k3s with xx to avoid QEM...	2 months ago
CODEOWNERS	Update CODEOWNERS for k3s-...	6 years ago
CODE_OF_CONDUCT.md	Update CODE_OF_CONDUCT....	last year

CONTRIBUTING.md	Filter large untemplated issue...	last week
DCO	Simplify CONTRIBUTING.md	6 years ago
Dockerfile	Tidy dockerfile and build stages	yesterday
Dockerfile.manifest	[main] Bump k3s-io/kubernete...	3 weeks ago
Dockerfile.test	[main] Bump k3s-io/kubernete...	3 weeks ago
GOVERNANCE.md	Inclusive naming proposal (iss...	11 months ago
LICENSE	Initial Commit	8 years ago
MAINTAINERS	Update Orlin's mail	3 months ago
Makefile	Tidy dockerfile and build stages	yesterday
README.md	Fix a couple of small nits in RE...	last year
ROADMAP.md	Inclusive naming proposal (iss...	11 months ago
channel.yaml	Update stable channel to v1.3...	last week
docker-compose.yml	fix: agent volume in example d...	2 years ago
go.mod	Drop wrangler import from co...	yesterday
go.sum	Drop wrangler import from co...	yesterday
install.sh	fix: change sles to use slemicr...	2 weeks ago
install.sh.sha256sum	fix: change sles to use slemicr...	2 weeks ago
k3s-rootless.service	k3s-rootless.service: use fuse-...	5 years ago
k3s.service	Add 2>dev/null when checking...	3 years ago
main.go	Replace PreparingExecutor wit...	4 months ago

README | Code of conduct | More

K3s - Lightweight Kubernetes



Lightweight Kubernetes. Production ready, easy to install, half the memory, all in a binary less than 100 MB.

Great for:

- Edge
- IoT
- CI
- Development
- ARM
- Embedding k8s
- Situations where a PhD in k8s clusterology is infeasible

What is this?

K3s is a fully conformant production-ready Kubernetes distribution with the following changes:

1. It is packaged as a single binary.
2. It adds support for sqlite3 as the default storage backend. Etcd3, MariaDB, MySQL, and Postgres are also supported.
3. It wraps Kubernetes and other components in a single, simple launcher.
4. It is secure by default with reasonable defaults for lightweight environments.
5. It has minimal to no OS dependencies (just a sane kernel and cgroup mounts needed).
6. It eliminates the need to expose a port on Kubernetes worker nodes for the kubelet API by exposing this API to the Kubernetes control plane nodes over a websocket tunnel.

K3s bundles the following technologies together into a single cohesive distribution:

- Containerd & runc
- Flannel for CNI
- CoreDNS
- Metrics Server
- Traefik for ingress
- Klipper-lb as an embedded service load balancer provider
- Kube-router netpol controller for network policy
- Helm-controller to allow for CRD-driven deployment of helm manifests
- Kine as a datastore shim that allows etcd to be replaced with other databases
- Local-path-provisioner for provisioning volumes using local storage
- Host utilities such as iptables/nftables, ebtables, ethtool, & socat

These technologies can be disabled or swapped out for technologies of your choice.

Additionally, K3s simplifies Kubernetes operations by maintaining functionality for:

- Managing the TLS certificates of Kubernetes components
- Managing the connection between worker and server nodes
- Auto-deploying Kubernetes resources from local manifests in realtime as they are changed.
- Managing an embedded etcd cluster

What's with the name?

We wanted an installation of Kubernetes that was half the size in terms of memory footprint. Kubernetes is a 10 letter word stylized as k8s. So something half as big as Kubernetes would be a 5 letter word stylized as K3s. A '3' is also an '8' cut in half vertically. There is neither a long-form of K3s nor official pronunciation.

Is this a fork?

No, it's a distribution. A fork implies continued divergence from the original. This is not K3s's goal or practice. K3s explicitly intends not to change any core Kubernetes functionality. We seek to remain as close to upstream Kubernetes as possible. However, we maintain a small set of patches (well under 1000 lines) important to K3s's use case and deployment model. We maintain patches for other components as well. When possible, we contribute these changes back to the upstream projects, for example, with SELinux support in containerd. This is a common practice amongst software distributions.

K3s is a distribution because it packages additional components and services necessary for a fully functional cluster that go beyond vanilla Kubernetes. These are opinionated choices on technologies for components like ingress, storage class, network policy, service load balancer, and even container runtime. These choices and technologies are touched on in more detail in the What is this? section.

How is this lightweight or smaller than upstream Kubernetes?

There are two major ways that K3s is lighter weight than upstream Kubernetes:

1. The memory footprint to run it is smaller
2. The binary, which contains all the non-containerized components needed to run a cluster, is smaller

The memory footprint is reduced primarily by running many components inside of a single process. This eliminates significant overhead that would otherwise be duplicated for each component.

The binary is smaller by removing third-party storage drivers and cloud providers, explained in more detail below.

What have you removed from upstream Kubernetes?

This is a common point of confusion because it has changed over time. Early versions of K3s had much more removed than the current version. K3s currently removes two things:

1. In-tree storage drivers
2. In-tree cloud provider

Both of these have out-of-tree alternatives in the form of CSI and CCM, which work in K3s and which upstream is moving towards.

We remove these to achieve a smaller binary size. They can be removed while remaining conformant because neither affects core Kubernetes functionality. They are also dependent on third-party cloud or data center technologies/services, which may not be available in many K3s' use cases.

Getting Started

- Quick Install
- Architecture
- FAQ
- Contribute

Community

- **Slack**

Join Slack to chat with K3s developers and other K3s users. Great place to learn and ask questions: #k3s and #k3s-contributor and #k3s channel in CNCF Slack

- **Getting involved**

GitHub Issues - Submit your issues and feature requests via GitHub.

- **Community Meetings and Office hours**

The K3s developer community hangs out on Zoom to chat. Everybody is welcome.

Add the Linux Foundation iCal to your calendar:

- AMS/EMEA TZ 10:00 am PST - every *second* Tuesday of the month
- EMEA/APAC TimeZone friendly - every *third* Tuesday of the month

Meeting notes and agenda: https://hackmd.io/@k3s/meet-notes/

Meeting recordings: K3s Channel

You can check also the full details on the website: https://k3s.io/community

What's next?

Check out our roadmap to see what we have planned moving forward.

Release cadence

K3s maintains pace with upstream Kubernetes releases. Our goal is to release patch releases within one week, and new minors within 30 days.

Our release versioning reflects the version of upstream Kubernetes that is being released. For example, the K3s release v1.27.4+k3s1 maps to the `v1.27.4` Kubernetes release. We add a postfix in the form of `+k3s<number>` to allow us to make additional releases using the same version of upstream Kubernetes while remaining semver compliant. For example, if we discovered a high severity bug in `v1.27.4+k3s1` and needed to release an immediate fix for it, we would release `v1.27.4+k3s2`.

Documentation

Please see the official docs site for complete documentation.

Quick-Start - Install Script

The `install.sh` script provides a convenient way to download K3s and add a service to systemd or openrc.

To install k3s as a service, run:

```
curl -sfL https://get.k3s.io | sh -
```

A kubeconfig file is written to `/etc/rancher/k3s/k3s.yaml` and the service is automatically started or restarted. The install script will install K3s and additional utilities, such as `kubectl`, `crictl`, `k3s-killall.sh`, and `k3s-uninstall.sh`, for example:

```
sudo kubectl get nodes
```

`K3S_TOKEN` is created at `/var/lib/rancher/k3s/server/node-token` on your server. To install on worker nodes, pass `K3S_URL` along with `K3S_TOKEN` environment variables, for example:

```
curl -sfL https://get.k3s.io | K3S_URL=https://myserver:6443 K3S_TOKEN=XXX sh -
```

Manual Download

1. Download `k3s` from latest release, x86_64, armhf, arm64 and s390x are supported.
2. Run the server.

```
sudo k3s server &
# Kubeconfig is written to /etc/rancher/k3s/k3s.yaml
sudo k3s kubectl get nodes

# On a different node run the below. NODE_TOKEN comes from
# /var/lib/rancher/k3s/server/node-token on your server
sudo k3s agent --server https://myserver:6443 --token ${NODE_TOKEN}
```

Contributing

Please check out our contributing guide if you're interested in contributing to K3s.

Security

Security issues in K3s can be reported by sending an email to security@k3s.io. Please do not file issues about security issues.

Releases 1 023

v1.36.3+k3s1 Latest
2 weeks ago

+ 1 022 releases

Contributors 304

+ 290 contributors

Languages

Go 90.8% **Shell** 7.1% **HCL** 0.6% **Go Template** 0.6% **Ruby** 0.5% **Dockerfile** 0.3% **Makefile** 0.1%