

HKOSCon 2024

Metrics, Logs, Traces and Profiles with Grafana LGTM

Wong Hoi Sing, Edison

2024-07-06

Contents

Introduction	2
Get the Code	2
About Me	3
Contact Me	10
References	10
What is Grafana LGTM Stack?	18
Prometheus + LGTM Stack + Pyroscope	18
Prometheus	19
Grafana	24
Loki	28
Tempo	32
Mimir	36
Pyroscope	40
Demo with AlviStack Vagrant Box	44
System Requirements	44
Install VirtualBox	44
Install Vagrant	46
Prepare Vagrantfile	48

vagrant up	50
vagrant provision	52
\$ vagrant ssh	54
kubectl get node	56
kubectl get pod	56
Verify with Browser	56
Create a Grafana Dashboard	56
Connect with Data Sources	56
Q&A	56

Introduction

Get the Code

- GitHub Repo
 - GitHub Page
 - Vagrantfile
-



About Me

- Wong Hoi Sing, Edison (hswong3i)

- 2005: Drupal Developer & Contributor
 - 2008: HKDUG, Founder
 - 2010: PantaRei Design, Founder
 - 2020: HKOSCON 2020, Speaker
 - 2021: AlviStack, Founder
 - 2022: HKOSCON 2022, Speaker
 - 2024: Most Active GitHub user in Hong Kong
-





DMC-FZ18 F2.8 1/30s ISO100



CLOUD NATIVE LANDSCAPE EXPLORE GUIDE STATS Search items

AlviStack Vagrant Box Packaging for Kubernetes

PantaRei Design Limited

Platform Certified Kubernetes - Installer

This Vagrant Box provides Libvirt and VirtualBox image for running Kubernetes in single node all-in-one mode.

Repositories

alvistack/vagrant-kubernetes (primary)

<https://github.com/alvistack/vagrant-kubernetes> PRIMARY Apache License 2.0 good first issues 0 open

24 Stars 1 Contributors Oct '20 First commit Jul '24 Latest commit Jul '21 Latest release

Participation stats

Languages

AlviStack – Hong Kong Based Kubernetes Distribution



Time: 10:15 – 11:15
Language: English

Search

SEARCH



Schedule



Speakers

Mr. Edison Wong Hoi Sing

Mr. Eason Lai

Ms. Amanda Lam

Mr. Cyrus Wong

Mr. Davide Benvegnu

Mr. Ivan Ma

github-actions[bot] · Update Hong Kong · 36fb409 · 3 Weeks ago · History

[Preview](#) [Code](#) [Blame](#) 11368 lines (11354 loc) · 320 KB Code 55% faster with GitHub Copilot

Top GitHub Users By Total Contributions in Hong Kong 🇭🇰

Top GitHub Users passing Views 827278

The `public contributions` and `private contributions` by users in Hong Kong on `2024/6/11 12:56 AM UTC`. This list contains users from `Hong Kong` and cities `Hong-kong` `Kowloon`.

There are `138 countries` and `674 cities` can be found [here](#).

There are `930 users` in Hong Kong. You need at least `30 followers` to be on this list.

Don't forget to star ★ this repository

[Top Users By Public Contributions](#) [Top Users By Total Contributions](#) [Top Users By Followers](#)

Share on

#	Name	Company	Twitter Username	Location	Public Contributions	Total Contributions
1	hswong3i Wong Hoi Sing Edison	Http://pantarei-desi	hswong3i	Hong Kong	42611	42611
2	BattlefieldDuck TatLead	No Company	BattlefieldDuck	Hong Kong	26207	30200
3	dirkarnez Dirk Arnez	Freelance	No Twitter Username	Hong Kong	7944	15359
	HTCAzuroY1212	@teamhartex.l	No Twitter			

Contact Me

- Address: Unit 326, 3/F, Building 16W, No.16 Science Park West Avenue, Hong Kong Science Park, Shatin, N.T.
- Phone: +852 3576 3812
- Fax: +852 3753 3663
- Email: sales@pantarei-design.com
- Web: <http://pantarei-design.com>

References

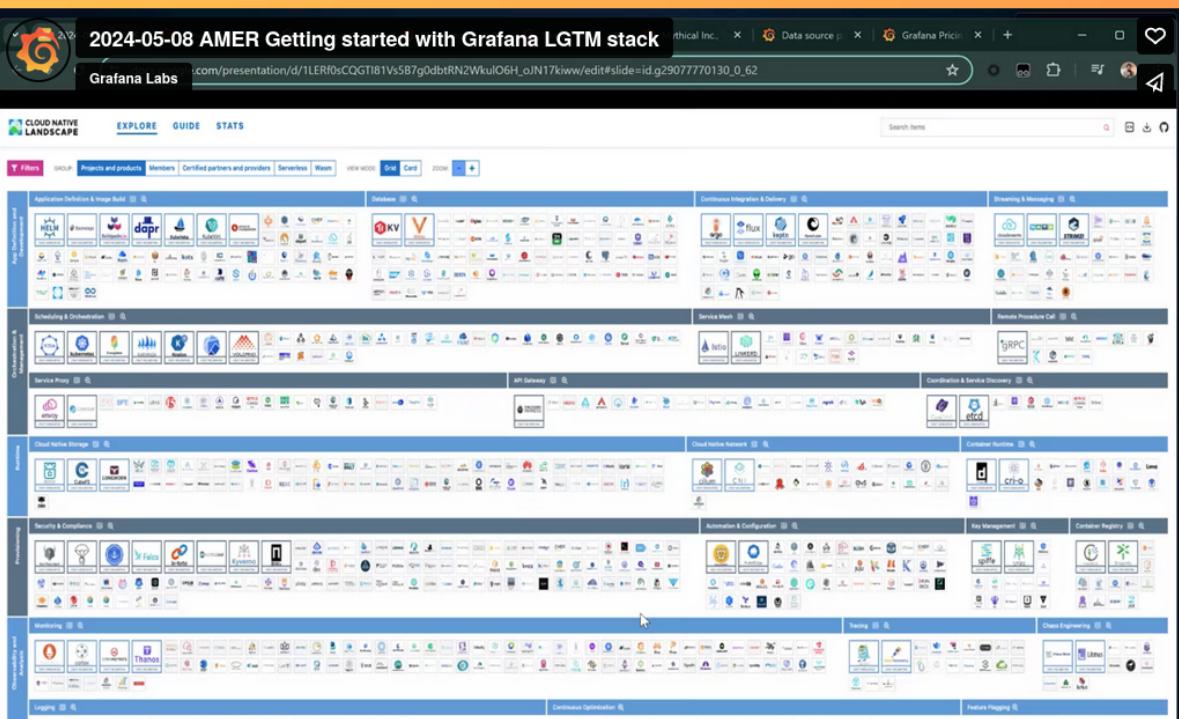
- Getting started with managing your metrics, logs, and traces using Grafana
 - Official webinar from Grafana Labs
 - For beginner with simple show case
 - Around 60 mintues
-

Grafana Labs Products Open source Solutions Learn Docs Company

English

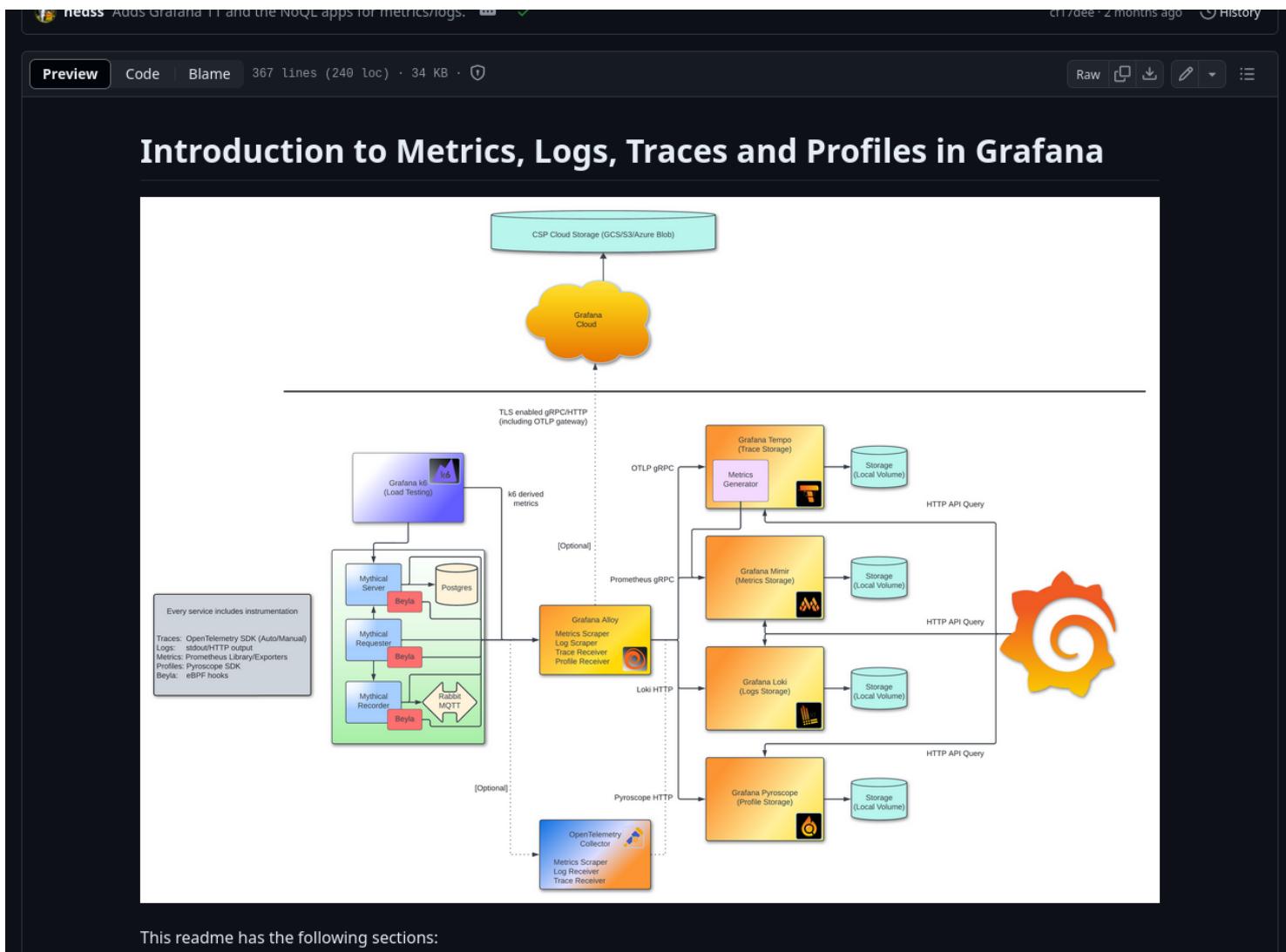
Getting started with managing your metrics, logs, and traces using Grafana

Share this: [in](#) [Twitter](#)



The screenshot shows the Grafana Cloud Native Landscape dashboard. The top navigation bar includes links for Products, Open source, Solutions, Learn, Docs, Company, a search icon, Downloads, Contact us, and Sign in. Below the navigation is a language dropdown set to English. The main title is "Getting started with managing your metrics, logs, and traces using Grafana". A "Share this" button with links for LinkedIn and Twitter is present. The dashboard itself is titled "2024-05-08 AMER Getting started with Grafana LGTM stack" and displays a grid of icons representing various open-source projects and tools, categorized into sections like Application Definition & Image Build, Container Integration & Delivery, Streaming & Messaging, Scheduling & Orchestration, Service Mesh, Remote Procedure Call, Service Proxy, API Gateway, Coordination & Service Discovery, Cloud Native Storage, Cloud Native Networks, Container Runtime, Security & Compliance, Automation & Configuration, Key Management, Container Registry, Monitoring, Testing, DevOps Engineering, Logging, Continuous Optimization, and Feature Flipping.

- Introduction to Metrics, Logs, Traces and Profiles in Grafana
 - Official complete demo for Grafana LGTM stack
 - For advance user with Docker + Docker Compose experience
 - Around 2 hours
-



This readme has the following sections:

- 可觀測性宇宙的第一天 - Grafana LGTM 全家桶的起點
 - Detail explanation for Grafana LGTM + Kubernetes
 - For advance user with daily Kubernetes experience
 - AT LEAST 6 HOURS (reading doc only…)
-



可觀測性宇宙的第一天 - Grafana LGTM 全家桶的起點

15th 鐵人賽

kubernetes

grafana

observability

可觀測性



mikehsu0618

團隊 所以隊名要叫什麼

2023-09-16 00:36:13

6984 瀏覽

分享至



前言

這是一個從異世界歸來後，一個不小心踏入 Kubernetes 可觀測性宇宙的故事，一切都要從一個非本科小白在愚昧山丘的頂端遇見 Grafana 開始說起。

- AlviStack Vagrant Box Packaging for Kubernetes
 - CNCF Certified Kubernetes on Virtual Box or Libvirt
 - Simply `vagrant up` and ready for use
 - May customize for Dev/CI/CD
 - Around 10 minutes
-



alvistack/kubernetes-1.30 Vagrant box

How to use this box with [Vagrant](#):

Vagrantfile [New](#)

```
Vagrant.configure("2") do |config|
  config.vm.box = "alvistack/kubernetes-1.30"
end
```

[v20240629.1.1](#) currently released version

This version was created 6 days ago.

There isn't a description.

2 providers for this version.

libvirt

[unknown *](#)

Hosted by Vagrant Cloud (1.75 GB)



virtualbox

[unknown *](#)

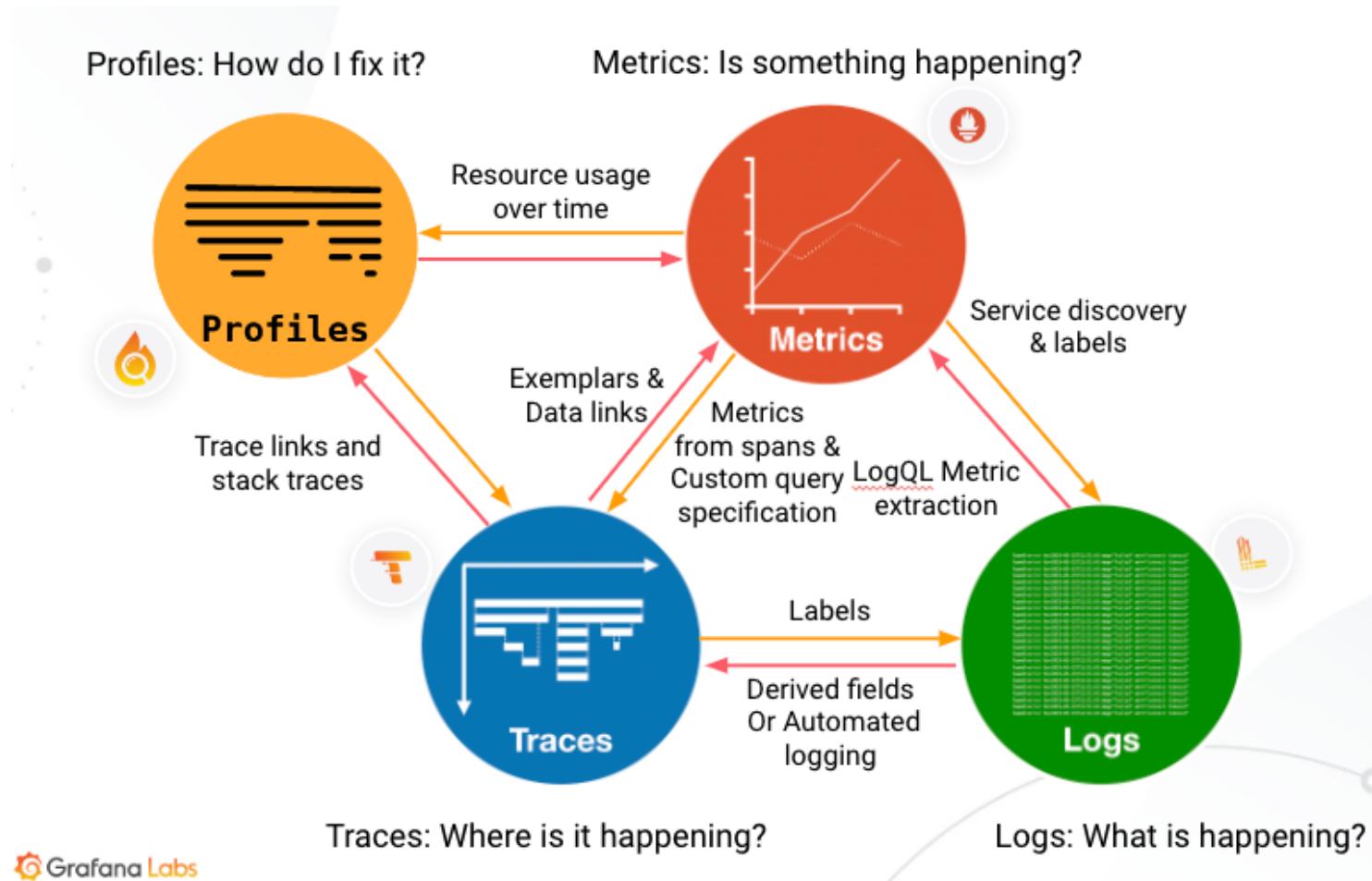
Hosted by Vagrant Cloud (1.68 GB)



What is Grafana LGTM Stack?

Prometheus + LGTM Stack + Pyroscope

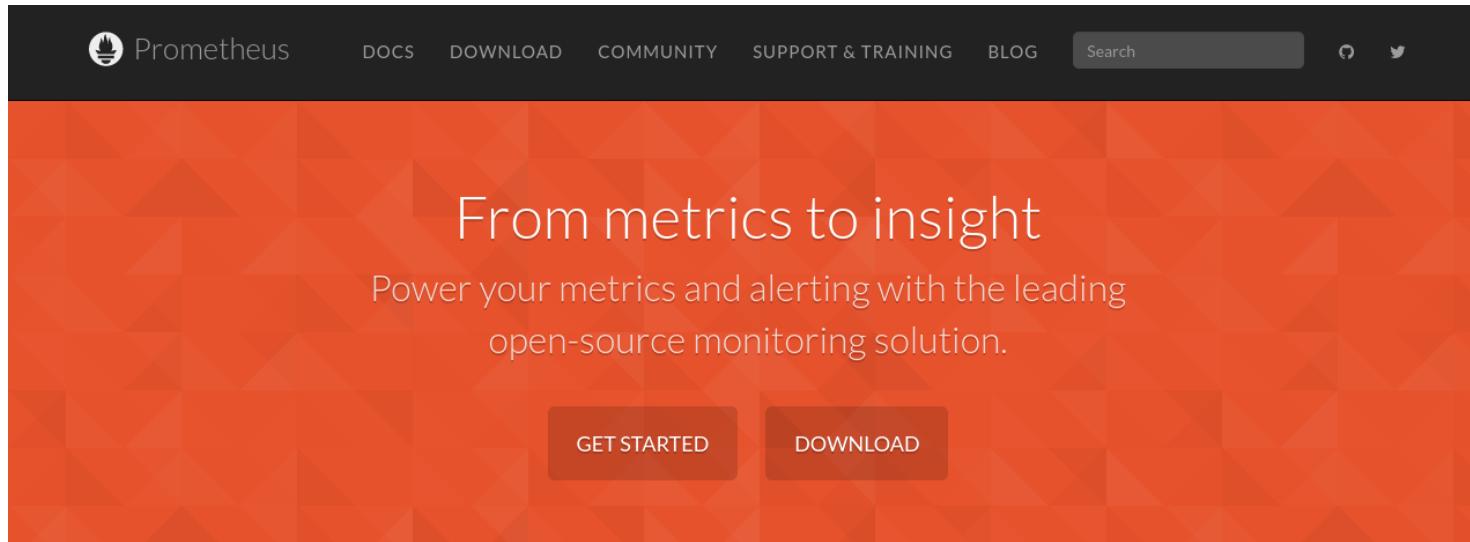
- Monitoring: Collects and stores metrics as time series data
 - Logs: What is happening?
 - Metrics: Is something happening?
 - Traces: Where is it happening?
 - Profiles: How do I fix it?
 - Dashboard: Data visualization and monitoring
-



Prometheus

- Prometheus is a systems monitoring and alerting toolkit
- Multi-dimensional data model

- PromQL, a flexible query language to leverage this dimensionality
 - 2nd CNCF graduated project after Kubernetes
 - Apache License 2.0
-



The banner features the Prometheus logo at the top left. A search bar and social media links (GitHub and Twitter) are at the top right. The main heading "From metrics to insight" is centered above the subtext "Power your metrics and alerting with the leading open-source monitoring solution." Below the subtext are two buttons: "GET STARTED" and "DOWNLOAD".

From metrics to insight

Power your metrics and alerting with the leading open-source monitoring solution.

[GET STARTED](#) [DOWNLOAD](#)

➊ Dimensional data

Prometheus implements a highly dimensional data model. Time series are identified by a metric name and a set of key-value pairs.

➋ Powerful queries

PromQL allows slicing and dicing of collected time series data in order to generate ad-hoc graphs, tables, and alerts.

➌ Great visualization

Prometheus has multiple modes for visualizing data: a built-in expression browser, Grafana integration, and a console template language.

➍ Efficient storage

Prometheus stores time series in memory and on local disk in an efficient custom format. Scaling is achieved by functional sharding and federation.

➎ Simple operation

Each server is independent for reliability, relying only on local storage. Written in Go, all binaries are statically linked and easy to deploy.

➏ Precise alerting

Alerts are defined based on Prometheus's flexible PromQL and maintain dimensional information. An alertmanager handles notifications and silencing.

➐ Many client libraries

Client libraries allow easy instrumentation of services. Over ten languages are supported already and custom libraries are easy to implement.

➑ Many integrations

Existing exporters allow bridging of third-party data into Prometheus. Examples: system statistics, as well as Docker, HAProxy, StatsD, and JMX metrics.

- Prometheus Operator deploy with Kubernetes CRDs
 - kube-prometheus provides example configurations for a complete cluster monitoring stack
 - Here we are going to demo with kube-prometheus-stack with Helm chart style
-

helm-charts / charts / kube-prometheus-stack / [↑ Top](#)

kube-prometheus-stack

Installs the [kube-prometheus stack](#), a collection of Kubernetes manifests, [Grafana](#) dashboards, and [Prometheus rules](#) combined with documentation and scripts to provide easy to operate end-to-end Kubernetes cluster monitoring with [Prometheus](#) using the [Prometheus Operator](#).

See the [kube-prometheus](#) README for details about components, dashboards, and alerts.

Note: This chart was formerly named `prometheus-operator` chart, now renamed to more clearly reflect that it installs the `kube-prometheus` project stack, within which Prometheus Operator is only one component.

Prerequisites

- Kubernetes 1.19+
- Helm 3+

Get Helm Repository Info

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm repo update
```

See [helm repo](#) for command documentation.

Install Helm Chart

```
helm install [RELEASE_NAME] prometheus-community/kube-prometheus-stack
```

See [configuration](#) below.

See [helm install](#) for command documentation.

Dependencies

Grafana

- Grafana is a multi-platform open source analytics and interactive visualization web application
 - It can produce charts, graphs, and alerts for the web when connected to supported data sources
 - AGPLv3
-

Grafana Labs Products Open source Solutions Learn Docs Company Downloads Contact us Sign in

Star us on GitHub | ★ 61,456

Grafana OSS

Visualize your data, optimize your performance

Easily collect, correlate, and visualize data with beautiful dashboards using Grafana — the open source data visualization and monitoring solution that drives informed decisions, enhances system performance, and streamlines troubleshooting.

The easiest way to get started with the latest version of Grafana is with [Grafana Cloud](#), our fully managed observability stack.

[Download](#) [Docs](#) [GitHub](#)



 Transform data into real-time insights
Grafana's growing suite of visualizations, ranging from time series graphs to heatmaps to cutting-edge 3D charts, help you decode complex datasets.

 Seamlessly build a single pane of glass
With 150+ Grafana plugins, you can unify all your data sources into a single dashboard to streamline data monitoring and troubleshooting.

 Resolve issues faster with Grafana Alerting
Alert on data from a wide variety of data sources to identify problems and fix them quickly.

The evolution of Grafana

Inspired by the open source community and our growing number of users, Grafana introduces new innovations in every release to improve how you visualize, correlate, and share your data.

- Support multiple installation style, e.g. monolithic, microservice, or even manually
 - Here we are going to demo with distributed Helm chart style
-

Grafana Labs Products Open source Solutions Learn Docs Company Downloads Contact us Sign in

Grafana Labs documentation

Search docs

Product: Grafana

Viewing: v11.1 (latest)

Find another version

Grafana documentation

- > What's new
- > Breaking changes
- > Upgrade Grafana
- > About Grafana
- > Introduction
- > Get started with Grafana Open Source

Set up

- Install Grafana
 - Debian or Ubuntu
 - RHEL or Fedora
 - SUSE or openSUSE
 - Grafana Docker image
 - Grafana on Kubernetes
 - Grafana on Helm Charts
 - macOS
 - Windows
- > Configure Grafana
- Start Grafana
- Sign in to Grafana

Documentation > Grafana documentation > Set up > Install Grafana > Grafana on Helm Charts

Open source

Deploy Grafana using Helm Charts

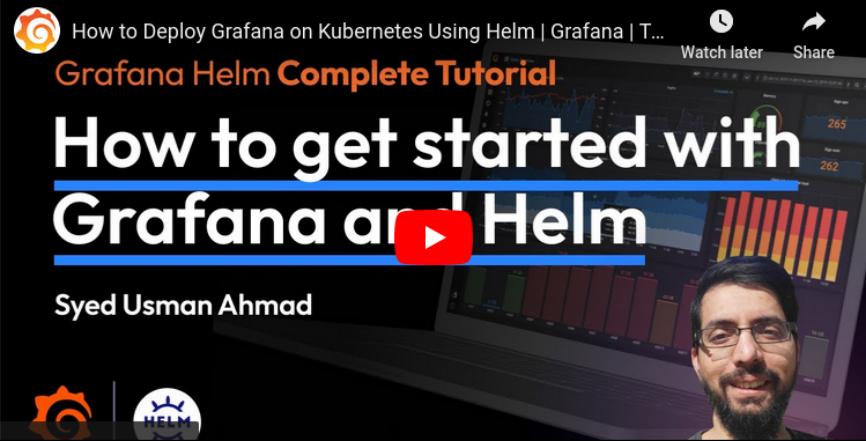
This topic includes instructions for installing and running Grafana on Kubernetes using Helm Charts.

Helm is an open-source command line tool used for managing Kubernetes applications. It is a graduate project in the [CNCF Landscape](#).

NOTE

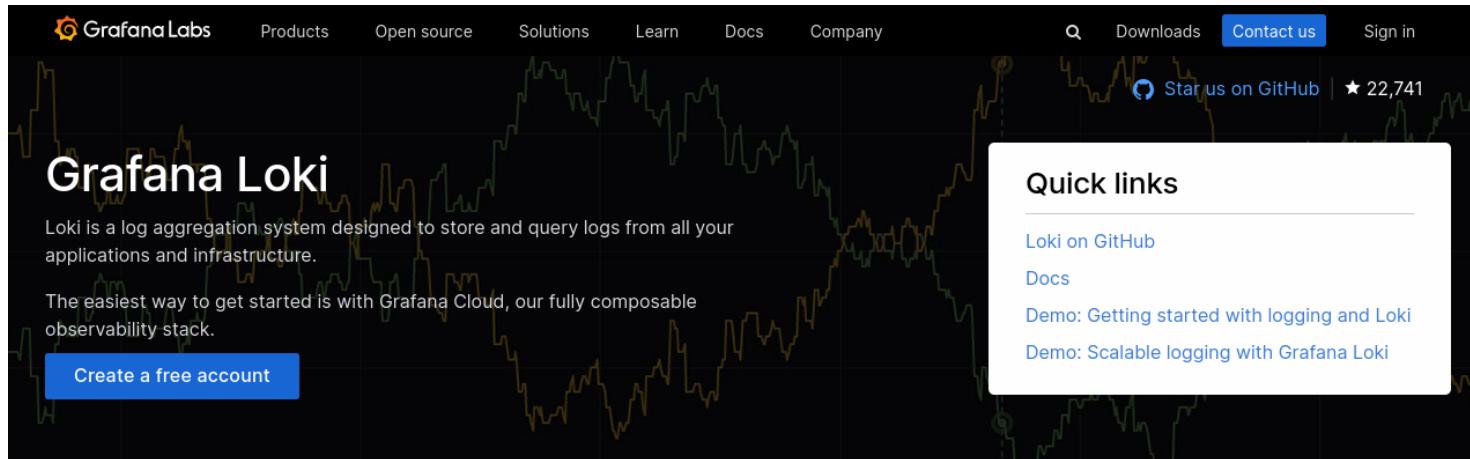
The Grafana open-source community offers Helm Charts for running it on Kubernetes. Please be aware that the code is provided without any warranties. If you encounter any problems, you can report them to the [Official GitHub repository](#).

Watch this video to learn more about installing Grafana using Helm Charts:



Loki

- Loki is a backend store for long-term log retention
 - Designed to be very cost effective and highly scalable
 - AGPLv3
-



Grafana Loki

Loki is a horizontally scalable, highly available, multi-tenant log aggregation system inspired by Prometheus. It is designed to be very cost effective and easy to operate. It does not index the contents of the logs, but rather a set of labels for each log stream.

The Loki project was started at Grafana Labs in 2018, and announced at [KubeCon Seattle](#).
Loki is released under the AGPLv3 license.

Grafana Labs is proud to lead the development of the Loki project, building first-class support for Loki into Grafana, and ensuring Grafana Labs customers receive Loki support and features they need.

Why use Grafana Loki?

- Support multiple installation style, e.g. monolithic, microservice, or even manually
 - Here we are going to demo with distributed Helm chart style
-

The screenshot shows the Grafana Labs documentation website for Grafana Loki. The left sidebar is dark-themed and lists various installation methods: Release notes, Get started, Set up (with Install selected), Monitoring, Install using Tanka, Install using Docker, Install locally, Install on Istio, Install from source, Migrate, and Upgrade. Under Set up > Install, the 'Install microservice Loki' option is highlighted with an orange border. The main content area has a light background and displays the following information:

Documentation > Grafana Loki > Set up > Install > Install using Helm > Install microservice Loki

Open source RSS

Install the microservice Helm chart

This Helm Chart deploys Grafana Loki on Kubernetes.

This chart configures Loki to run Loki in [microservice / distributed mode](#). The microservices deployment mode runs components of Loki as distinct processes.

The default Helm chart deploys the following components:

- **Compactor component** (1 replica): Compacts and processes stored data.
- **Distributor component** (3 replicas, maxUnavailable: 2): Distributes incoming requests. Up to 2 replicas can be unavailable during updates.
- **IndexGateway component** (2 replicas, maxUnavailable: 1): Handles indexing. Up to 1 replica can be unavailable during updates.
- **Ingestor component** (3 replicas): Handles ingestion of data.
- **Querier component** (3 replicas, maxUnavailable: 2): Processes queries. Up to 2 replicas can be unavailable during updates.
- **QueryFrontend component** (2 replicas, maxUnavailable: 1): Manages frontend queries. Up to 1 replica can be unavailable during updates.
- **QueryScheduler component** (2 replicas): Schedules queries.

It is not recommended to run scalable mode with [filesystem](#) storage. For the purpose of this guide, we will use MinIO as the object storage to provide a complete example.

Prerequisites

- Helm 3 or above. See [Installing Helm](#).
- A running Kubernetes cluster.
- (Optional) A Memcached deployment for better query performance. For information on configuring Memcached, refer to the [caching section](#).

To deploy Loki in microservice mode (with MinIO):

Tempo

- Tempo is a distributed tracing backend
 - Designed to be cost-efficient, requiring only object storage to operate
 - Deeply integrated with Grafana, Prometheus, and Loki
 - AGPLv3
-

The screenshot shows the Grafana Tempo landing page. At the top, there's a navigation bar with links for Products, Open source, Solutions, Learn, Docs, Company, Downloads, Contact us, and Sign in. A GitHub star count of 3,758 is also displayed. The main content area features a tracing visualization with metrics like 0ms, 16.16ms, and 53.36ms. A prominent blue button says "Create free account". To the right, a white box titled "Quick links" contains links to Demo, GitHub Project, Docs, and a specific demo link: "Demo: Get started with Grafana Tempo".



Introduction to Grafana Tempo

Grafana Tempo is an open source, easy-to-use, and high-scale distributed tracing backend. Tempo is cost-efficient, requiring only object storage to operate, and is deeply integrated with Grafana, Prometheus, and Loki. Tempo can ingest common open source tracing protocols, including Jaeger, Zipkin, and OpenTelemetry.

The Tempo project was started at Grafana Labs and announced at Grafana ObservabilityCON in October 2020. It became generally available with the 1.0 release in June 2021. Tempo is released under the AGPLv3 license.

Grafana Labs is proud to lead the development of the Tempo project, building first-class support for Tempo into Grafana, and ensuring Grafana Labs customers receive Tempo support and features they need.

The Goal

Sample 100% of our read path

- Support multiple installation style, e.g. monolithic, microservice, or even manually
 - Here we are going to demo with distributed Helm chart style
-

The screenshot shows the Grafana Labs documentation website for Grafana Tempo. The left sidebar is dark-themed and contains a search bar, a product dropdown set to 'Grafana Tempo', and a navigation menu. The main content area has a light background and displays the 'Deploy Tempo with Helm' page. It includes a breadcrumb trail, a 'Documentation' button, and sections for 'Deploy Tempo with Helm' and 'Example Helm chart'. The 'Deploy with Helm' section is highlighted with an orange border. The footer features social media icons and links for 'Suggest an edit', 'Contribute to docs', 'Report a problem', 'Community', and 'Support'.

Grafana Labs documentation

Products Open source Solutions Learn Docs Company

Documentation > Grafana Tempo > Set up > Deploy with Helm

Open source

Product

Grafana Tempo

Viewing: v2.5.x (latest)

Find another version

Grafana Tempo

› Release notes

› Introduction

› Get started

Set up

› Plan your deployment

Upgrade

Deploy with Helm

› Deploy with operator

Deploy on Linux

Deploy on Kubernetes with Tanka

Set up a test application for a Tempo cluster

› Configure

› Manage

› Metrics-generator

› Query with TraceQL

› Troubleshoot

› API

Community

Was this page helpful?

Yes No

Suggest an edit Contribute to docs Report a problem

Community Support

Mimir

- Mimir is a TSDB for long-term storage for Prometheus
 - Designed to be horizontally scalable, highly available, multi-tenant
 - 100% Prometheus compatible
 - AGPLv3
-

Grafana Labs Products Open source Solutions Learn Docs Company Downloads Contact us Sign in

Star us on GitHub | ★ 3,858

What is Grafana Mimir?

Mimir is an open source, horizontally scalable, highly available, multi-tenant TSDB for long-term storage for Prometheus.

[Read the announcement blog post →](#)

Get Mimir up and running in 10 minutes:

[Tutorial](#) [Intro to metrics webinar](#)

Introducing Grafana Mimir
Grafana Mimir: Open Source time series database

Prometheus or Grafana Agent
Scrape metrics from your applications and remote write these metrics to Mimir

Grafana
Query metrics from Mimir

05:00



Mimir overview

Mimir lets you scale metrics to 1 billion active series and beyond, with high availability, multi-tenancy, durable storage, and blazing fast query performance over long periods of time.

Mimir was started at Grafana Labs and [announced in 2022](#). The mission for the project is to make it the most scalable, most performant open source time series database for metrics, by incorporating what Grafana Labs engineers have learned running Grafana Enterprise Metrics and Grafana Cloud Metrics at massive scale. Mimir is released under the AGPLv3 license.

Grafana Labs is proud to lead the development of the Mimir project, building first-class support for Mimir into Grafana, and ensuring Grafana Labs customers receive Mimir support and features they need.

- Support multiple installation style, e.g. monolithic, microservice, or even manually
 - Here we are going to demo with distributed Helm chart style
-

The screenshot shows a dark-themed documentation page for Grafana Mimir. The left sidebar contains navigation links for 'Grafana Labs documentation', 'Search docs', 'Product' (set to 'Grafana Mimir'), 'Viewing: v2.12.x (latest)', and a detailed 'Set up' section with 'Deploy with Helm' highlighted. The main content area shows the 'Deploy Mimir with Helm' page, which includes a brief description of the Helm chart, deployment instructions, and a 'Was this page helpful?' poll with 'Yes' and 'No' buttons. Below this are 'Related documentation' cards for 'Deploying the Grafana Mimir monitoring mixin', 'Grafana mimir-distributed Helm chart documentation', and 'Monitor Grafana Mimir'.

Grafana Labs documentation

Products Open source Solutions Learn Docs Company

Documentation > Grafana Mimir > Set up > Deploy with Helm

Open source

Product

Grafana Mimir

Viewing: v2.12.x (latest)

Find another version

Grafana Mimir

Release notes

Get started

Set up

Deploy with Helm

Deploy with Puppet

Deploy with Jsonnet and Tanka

Migrate

Configure

Send

Manage

Query

Visualize

References

Copyright notice

Deploy Mimir with Helm

The [mimir-distributed](#) Helm chart for Grafana Mimir and Grafana Enterprise Metrics allows you to configure, install, and upgrade Grafana Mimir or Grafana Enterprise Metrics within a Kubernetes cluster.

To deploy Mimir using the [mimir-distributed](#) Helm chart, see [Get started with Grafana Mimir using the Helm chart](#).

Was this page helpful?

Yes No

Suggest an edit Contribute to docs Report a problem

Community Support

Related documentation

Docs Deploying the Grafana Mimir monitoring mixin

Docs Grafana mimir-distributed Helm chart documentation

Docs Monitor Grafana Mimir

Pyroscope

- Pyroscope is a continuous profiling database
 - Designed to be fast, scalable, highly available, and efficient storage and querying
 - AGPLv3
-

The screenshot shows the Grafana Pyroscope homepage. At the top, there's a navigation bar with links for Products, Open source, Solutions, Learn, Docs, Company, Downloads, Contact us, and Sign in. Below the navigation is an announcement banner: "ANNOUNCEMENT" (with a megaphone icon) and "Grafana Cloud Profiles, powered by Pyroscope, is now GA". A "Learn more →" button is next to the announcement. To the right of the announcement is a GitHub star count: "Star us on GitHub | ★ 9,643". Below the announcement is a section titled "Continuous profiling" with a link. The main title is "What is Grafana Pyroscope?". Below the title is a description: "Grafana Pyroscope is an open source continuous profiling database that provides fast, scalable, highly available, and efficient storage and querying. This helps you get a better understanding of resource usage in your applications down to the line number." Below the description are links to "GitHub project", "Blog announcement", and "Documentation". On the right side of the page is a screenshot of the Grafana interface showing a timeline and a heatmap of resource usage.



Grafana Pyroscope overview

Grafana Pyroscope allows you to aggregate continuous profiling data with high availability, multi-tenancy, and durable storage.

Grafana Pyroscope was formed by the merger of two open source continuous profiling projects: Phlare, which was launched by Grafana Labs in 2022, and Pyroscope, which was founded by Ryan Perry and Dmitry Filimonov in 2021. The mission for the Grafana Pyroscope project is to enable continuous profiling at scale for the open source community, giving developers a better understanding of the resource usage of their code. By doing so, Pyroscope allows users to understand their application performance and optimize their infrastructure spend.

Grafana Labs is proud to lead the development of the Grafana Pyroscope project. The project

-
- Suport multiple installation style, e.g. monolithic, microservice, or even manually

- Here we are going to demo with distributed Helm chart style
-

The screenshot shows a dark-themed documentation page for Grafana Pyroscope. At the top, there's a navigation bar with links for Products, Open source, Solutions, Learn, Docs, Company, Downloads, Contact us (which is highlighted in blue), and Sign in. On the left, a sidebar titled "Grafana Labs documentation" includes a search bar ("Search docs") and a dropdown menu for "Product" set to "Grafana Pyroscope". It also shows the current version as "v1.6.x (latest)" and a link to "Find another version". The main content area has a breadcrumb trail: Documentation > Grafana Pyroscope > Deploy on Kubernetes > Deploy with Helm. A pink button labeled "Open source" is visible. The main title is "Deploy Pyroscope using the Helm chart". Below it, a paragraph explains that the Helm chart allows configuration, installation, and upgrade of Pyroscope within a Kubernetes cluster. A section titled "Before you begin" contains instructions common across Kubernetes flavors, mentioning knowledge of Kubernetes cluster installation, configuration, and operation, as well as the use of `kubectl`. It lists hardware requirements (a single node with 4 cores and 16GiB RAM) and software requirements (Kubernetes 1.20 or higher, `kubectl` command, Helm 3 or higher). A "Verify that you have:" section lists access to the cluster, persistent storage enabled with a default StorageClass, and functional DNS service. The "Install the Helm chart in a custom namespace" section is described as using a custom namespace to avoid overwriting the default namespace later.

Grafana Labs documentation

Products Open source Solutions Learn Docs Company

Downloads Contact us Sign in

Documentation > Grafana Pyroscope > Deploy on Kubernetes > Deploy with Helm

Open source

Product

Grafana Pyroscope

Viewing: v1.6.x (latest)

Find another version

Grafana Pyroscope

- > Release notes
- > Introduction
- Get started
- > Configure the client to send profiles
- Upgrade
- Deploy on Kubernetes
 - Deploy with Helm
 - Deploy with Jsonnet and Tanka
- > Configure the server
- > View and analyze profile data
- > Reference: Pyroscope Architecture
- Copyright notice

Deploy Pyroscope using the Helm chart

The Helm chart allows you to configure, install, and upgrade Pyroscope within a Kubernetes cluster.

Before you begin

These instructions are common across any flavor of Kubernetes and assume that you know how to install, configure, and operate a Kubernetes cluster as well as use `kubectl`.

Hardware requirements:

- A single Kubernetes node with a minimum of 4 cores and 16GiB RAM

Software requirements:

- Kubernetes 1.20 or higher
- The `kubectl` command for your version of Kubernetes
- Helm 3 or higher

Verify that you have:

- Access to the Kubernetes cluster
- Persistent storage is enabled in the Kubernetes cluster, which has a default storage class set up. You can [change the default StorageClass](#).
- DNS service works in the Kubernetes cluster

Install the Helm chart in a custom namespace

Use a custom namespace so that you don't have to overwrite the default namespace later in the procedure.

Demo with AlviStack Vagrant Box

System Requirements

- Windows 10+
- 4 Core CPU (for running a 2 Core CPU VM)
- 16GB Memory (for running a 8GB VM)
- 50GB SSD

Install VirtualBox

- Download and install VirtualBox
 - Download and install Extension Pack
 - Verify result
 - `$ vboxmanage --version`
 - <https://www.virtualbox.org/wiki/Downloads>
-



VirtualBox

search...
Login Preferences
Start Page Index History

Download VirtualBox

Here you will find links to VirtualBox binaries and its source code.

VirtualBox binaries

By downloading, you agree to the terms and conditions of the respective license.

VirtualBox 7.0.18 platform packages

- ⇒ Windows hosts
- ⇒ macOS / Intel hosts
- Linux distributions
- ⇒ Solaris hosts
- ⇒ Solaris 11 IPS hosts

The binaries are released under the terms of the GPL version 3.

See the [changelog](#) for what has changed.

You might want to compare the checksums to verify the integrity of downloaded packages. *The SHA256 checksums should be favored as the MD5 algorithm must be treated as insecure!*

- [SHA256 checksums](#), [MD5 checksums](#)

Note: After upgrading VirtualBox it is recommended to upgrade the guest additions as well.

VirtualBox 7.0.18 Oracle VM VirtualBox Extension Pack

- ⇒ All supported platforms

Support VirtualBox RDP, disk encryption, NVMe and PXE boot for Intel cards. See [this chapter from the User Manual](#) for an introduction to this Extension Pack. The Extension Pack binaries are released under the [VirtualBox Personal Use and Evaluation License \(PUEL\)](#). Please install the same version extension pack as your installed version of VirtualBox.

VirtualBox 7.0.18 Software Developer Kit (SDK)

- ⇒ All platforms

User Manual

The VirtualBox User Manual is included in the VirtualBox packages above. If, however, you would like to take a look at it without having to install the whole thing, you also access it here:

Install Vagrant

- Download and install Vagrant
 - Verify Result
 - `$ vagrant --version`
 - <https://developer.hashicorp.com/vagrant/install>
-

The screenshot shows the Vagrant website's 'Install' page. At the top, there's a navigation bar with links for 'Vagrant', 'Install', 'Intro', 'Tutorials', 'Documentation', 'Vagrant Cloud', and 'Try Cloud'. A search bar and user account options are also at the top right.

The main content area has a breadcrumb trail: 'Developer / Vagrant / Install'. It features a large 'Install Vagrant' button with a blue 'V' icon. To its right is a dropdown menu showing '2.4.1 (latest)'. Below this, there are sections for 'macOS', 'Windows', and 'Linux'.

macOS

- Package manager**

```
brew tap hashicorp/tap  
brew install hashicorp/tap/hashicorp-vagrant
```
- Binary download**
 - AMD64** Version: 2.4.1 [Download](#)
 - ARM64** Version: 2.4.1 [Download](#)

Windows

- Binary download**
 - AMD64** Version: 2.4.1 [Download](#)
 - I686** Version: 2.4.1 [Download](#)

Linux

About Vagrant
Vagrant is the command line utility for managing the lifecycle of virtual machines.

Featured docs

- Install
- Command-Line Interface
- Boxes
- Networking
- Synced Folders
- Providers

Vagrant Cloud
Virtual boxes for Linux, Laravel and any development environment
[Try Vagrant Cloud for free →](#)

Prepare Vagrantfile

- Describe the type of machine
 - How to configure these machines
 - How to provision these machines
 - <https://developer.hashicorp.com/vagrant/docs/vagrantfile>
-

The screenshot shows the Vagrant documentation website with a dark theme. The top navigation bar includes links for Home, Vagrant, Install, Intro, Tutorials, Documentation, Vagrant Cloud, Try Cloud, a search bar, and user account options.

The left sidebar contains a navigation tree under the Documentation section, with 'Vagrantfile' currently selected and highlighted in blue. Other sections include Installation, Commands (CLI), Vagrant Share, Overview (which is also highlighted), Configuration Version, Minimum Vagrant Version, Tips & Tricks, config.vm, config.ssh, config.winrm, config.winssh, config.vagrant, Boxes, Provisioning, Networking, Synced Folders, Cloud-Init, and Disks.

The main content area shows the 'Vagrantfile' page for version v2.4.1 (latest). The page title is 'Vagrantfile'. The text explains that the primary function of the Vagrantfile is to describe the type of machine required for a project, and how to configure and provision these machines. It notes that Vagrantfiles are called Vagrantfiles because the actual literal filename for the file is `Vagrantfile` (casing does not matter unless your file system is running in a strict case sensitive mode).

It states that Vagrant is meant to run with one Vagrantfile per project, and the Vagrantfile is supposed to be committed to version control. This allows other developers involved in the project to check out the code, run `vagrant up`, and be on their way. Vagrantfiles are portable across every platform Vagrant supports.

The syntax of Vagrantfiles is [Ruby](#), but knowledge of the Ruby programming language is not necessary to make modifications to the Vagrantfile, since it is mostly simple variable assignment. In fact, Ruby is not even the most popular community Vagrant is used within, which should help show you that despite not having Ruby knowledge, people are very successful with Vagrant.

Lookup Path

When you run any `vagrant` command, Vagrant climbs up the directory tree looking for the first Vagrantfile it can find, starting first in the current directory. So if you run `vagrant` in `/home/mitchellh/projects/foo`, it will search the following paths in order for a Vagrantfile, until it finds one:

On this page:

- [Vagrantfile](#)
- [Lookup Path](#)
- [Load Order and Merging](#)
- [Available Configuration Options](#)

vagrant up

- \$ vagrant up --provider virtualbox
 - Running VM with VirtualBox
 - Configure CPU/RAM/Disk
 - Configure network port mapping
 - Mount \$PWD into VM as /vagrant
 - Provision after VM is up and running
 - <https://developer.hashicorp.com/vagrant/docs/cli/up>
-

The screenshot shows the Vagrant documentation website. The top navigation bar includes links for Home, Vagrant, Install, Intro, Tutorials, Documentation, Vagrant Cloud, Try Cloud, a search bar, and user account options.

The left sidebar contains a 'Filter sidebar' button and a list of Vagrant commands under 'Commands (CLI)'. The listed commands are: Overview, box, cloud, connect, destroy, global-status, halt, init, login, package, plugin, port, powershell, provision, rdp, reload, and resume.

The main content area shows the 'Developer / Vagrant / Documentation / Commands (CLI) / up' path. A dropdown menu indicates 'v2.4.1 (latest)'. The page title is 'Up' and the command is 'vagrant up [name|id]'. It describes the command as creating and configuring guest machines according to the [Vagrantfile](#). It notes that this is the single most important command in Vagrant. Below this, a section titled 'Options' lists several command-line options:

- `[name]` - Name of machine defined in [Vagrantfile](#). Using `[name]` to specify the Vagrant machine to act on must be done from within a Vagrant project (directory where the `Vagrantfile` exists).
- `[id]` - Machine Id found with `vagrant global-status`. Using `[id]` allows you to call `vagrant up id` from any directory.
- `-[no-]destroy-on-error` - Destroy the newly created machine if a fatal, unexpected error occurs. This will only happen on the first `vagrant up`. By default this is set.
- `-[no-]install-provider` - If the requested provider is not installed, Vagrant will attempt to automatically install it if it can. By default this is enabled.
- `-[no-]parallel` - Bring multiple machines up in parallel if the provider supports it. Please consult the provider documentation to see if

vagrant provision

- \$ vagrant provision
 - Sometime you hope to R&D and debug the provision session
 - After VM up and running you could re-run the provision steps
 - <https://developer.hashicorp.com/vagrant/docs/cli/provision>
-

The screenshot shows the Vagrant documentation website with a dark theme. The top navigation bar includes links for Home, Vagrant, Install, Intro, Tutorials, Documentation, Vagrant Cloud, Try Cloud, a search bar, and user account options.

The left sidebar contains a navigation tree under 'Commands (CLI)'. The 'provision' command is highlighted with a dark grey background. Other commands listed include box, cloud, connect, destroy, global-status, halt, init, login, package, plugin, port, powershell, rdp, reload, and resume.

The main content area displays the 'Provision' page. The breadcrumb navigation shows Developer / Vagrant / Documentation / Commands (CLI) / provision. A dropdown menu indicates the version is v2.4.1 (latest). The page title is 'Provision'. The command syntax is shown as `vagrant provision [vm-name]`. The text explains that this command runs any configured [provisioners](#) against the running Vagrant managed machine. It highlights its usefulness for incremental development of shell scripts, Chef cookbooks, or Puppet modules, mentioning how simple modifications to provisioning scripts can be tested by running `vagrant provision`.

The 'Options' section lists a bullet point: `--provision-with x,y,z` - This will only run the given provisioners. An example is given: If you have a `:shell` and `:chef_solo` provisioner and run `vagrant provision --provision-with shell`, only the shell provisioner will be run.

A link to 'Edit this page on GitHub' is present at the bottom of the content area.

```
$ vagrant ssh
```

- \$ vagrant ssh
 - SSH into VM with user vagrant
 - Could switch as root with sudo su -
 - Host \$PWD already mount into VM as /vagrant
 - <https://developer.hashicorp.com/vagrant/docs/cli/ssh>
-

The screenshot shows the Vagrant documentation website with the URL [https://www.vagrantup.com/docs/cli/ssh.html](#). The page title is "SSH". The sidebar on the left lists various Vagrant commands under "Commands (CLI)". The main content area shows the "v2.4.1 (latest)" version of the "SSH" documentation. It includes a command-line example for connecting to a default machine:

```
$ vagrant ssh
```

Welcome to your Vagrant-built virtual machine.
Last login: Fri Sep 14 06:23:18 2012 from 10.0.2.2
\$ logout
Connection to 127.0.0.1 closed.

Below this, another example shows connecting to the "default" machine:

```
$ vagrant ssh default
```

Welcome to your Vagrant-built virtual machine.
Last login: Fri Jul 20 15:09:52 2018 from 10.0.2.2
\$ logout
Connection to 127.0.0.1 closed.
\$

On multi-machine setups, you can login to each VM using the name as displayed on `vagrant status`.

```

kubectl get node
root@kubernetes-1:~# kubectl get node
NAME      STATUS   ROLES      AGE      VERSION
kubernetes-1   Ready    control-plane   9m18s   v1.29.2

kubectl get pod
root@kubernetes-1:~# kubectl get pod --all-namespaces
NAMESPACE      NAME                               READY   STATUS    RESTARTS   AGE
csi-hostpath   csi-hostpath-socat-0            1/1     Running   0          6m36s
csi-hostpath   csi-hostpathplugin-0           8/8     Running   0          6m36s
kube-system    coredns-76f75df574-2tdbp        1/1     Running   0          9m2s
kube-system    coredns-76f75df574-g687d        1/1     Running   0          9m2s
kube-system    kube-addon-manager-kubernetes-1  1/1     Running   0          6m48s
kube-system    kube-apiserver-kubernetes-1       1/1     Running   0          9m18s
kube-system    kube-controller-manager-kubernetes-1  1/1     Running   0          9m18s
kube-system    kube-flannel-ds-kmq79           1/1     Running   0          6m36s
kube-system    kube-proxy-vd4qj                1/1     Running   0          9m2s
kube-system    kube-scheduler-kubernetes-1       1/1     Running   0          9m18s
kube-system    snapshot-controller-7b6f9cf9b4-rj5v5  1/1     Running   0          6m36s

```

Verify with Browser

- Check the result (from host, with VirtualBox port mapping enabled in Vagrantfile):
 - <http://localhost:8080>

Create a Grafana Dashboard

Connect with Data Sources

Q&A