

5 Essential K8s Tasks to Automate



Maria · [Follow](#)

Published in Kubeshop

7 min read · Jan 18, 2024



Listen



Share

Kubernetes has greatly changed the DevOps space and how we deploy applications. While it offers many powerful features, it also introduces many complexities around managing Kubernetes clusters. Keeping these clusters running smoothly, securely, and reliably requires constant attention and expertise. To address these challenges and ensure efficient management of clusters, Botkube offers a solution. Botkube is a collaborative Kubernetes troubleshooting and monitoring tool designed for both DevOps experts and developers who may not be Kubernetes experts. Botkube helps teams quickly respond to issues by sending timely alerts about what's happening in their Kubernetes environments. It's not just about alerts though; Botkube also lets teams automate responses, run Kubernetes commands, and follow Kubernetes best practices. Plus, it integrates with popular communication platforms like Slack, Microsoft Teams, Discord, and Mattermost, making it a valuable asset for any team working with Kubernetes.

In this blog post, we will explore how Botkube improves Kubernetes management. We will dive into five key Kubernetes tasks that Botkube can optimize and automate.

Task 1: Monitoring and Alerting Kubernetes Clusters

❌ v1/pods error

Error occurred for Pod **default/kube-prometheus-stack-grafana-5b4c5bf6c-qz57** in **botkube-lab** cluster

Messages:

- Readiness probe failed: Get "<http://172.0.90.17:3000/api/health>": context deadline exceeded (Client.Timeout exceeded while awaiting headers)

2023-02-20 12:20:13 PM

Run command...

`kubectl logs pod/kube-prometheus-stack-grafana-5b4c5bf6c-qz57 -n default` on `botkube-lab` by Automation "Show logs on error"

```
Defaulted container "grafana-sc-dashboard" out of: grafana-sc-dashboard, grafana-sc-datasources, grafana
{"time": "2023-02-20T11:19:43.889872+00:00", "level": "INFO", "msg": "Starting collector"}
{"time": "2023-02-20T11:19:43.891141+00:00", "level": "WARNING", "msg": "No
```

Monitoring is the foundation of Kubernetes management. In a Kubernetes environment, real-time insights into resource utilization, application behavior, and system health are crucial.

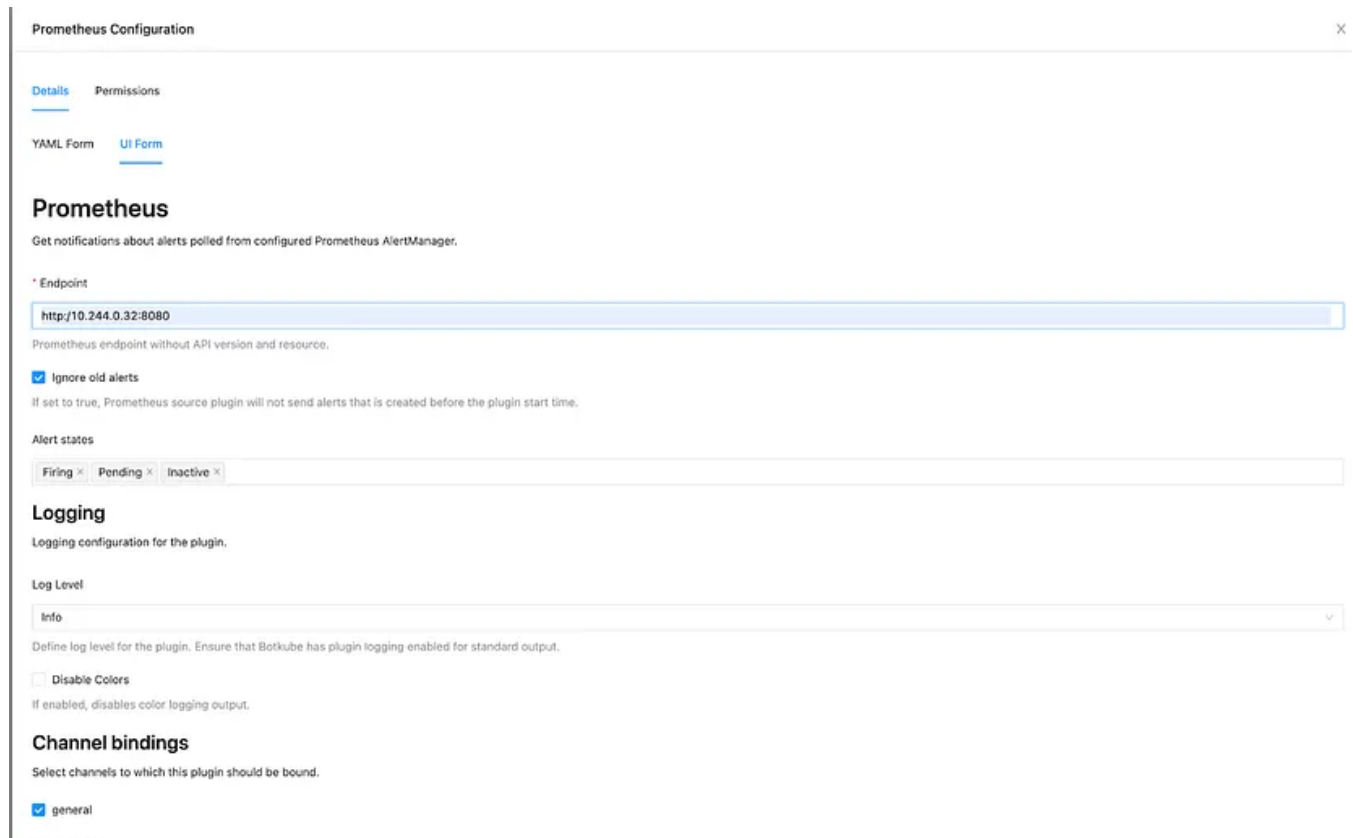
Botkube can automate your monitoring setup, transforming how you manage incoming queries and requests. It intelligently categorizes and responds to various types of queries based on their specific channel and frequency. This feature ensures a more efficient and streamlined handling of all incoming queries, enhancing your team's responsiveness and productivity.

Additionally, Botkube incorporates proactive monitoring capabilities into your chat platform by seamlessly integrating with Prometheus, a popular metrics tool for Kubernetes.

This integration means you're not just responding to issues as they arise but are also able to actively monitor and preemptively address potential concerns in your Kubernetes environment. The automation of these processes significantly reduces manual oversight, allowing your team to focus on more strategic tasks.

Manually setting up Prometheus alerts in Slack requires creating a PrometheusRule resource to define alert conditions in a YAML file. Following this, you must configure Alertmanager to route alerts to Slack channels based on routing rules, grouping, and timing settings, and establish a connection to Slack using a webhook.

Botkube simplifies this process significantly. It allows you to bypass these steps by entering the Prometheus endpoint and selecting your preferred alert states and logging configuration. Additionally, while manual setup limits integration options to Slack, PagerDuty, and email, excluding MS Teams and Discord users, Botkube offers a comprehensive solution.



The screenshot shows the 'Prometheus Configuration' dialog box in Botkube. It has tabs for 'Details' and 'Permissions', with 'Details' being the active tab. Inside 'Details', there are sub-tabs for 'YAML Form' and 'UI Form', with 'UI Form' being active. The main heading is 'Prometheus', followed by the description 'Get notifications about alerts polled from configured Prometheus AlertManager.' Below this, there is a section for '* Endpoint' with a text input field containing 'http://10.244.0.32:8080'. A note below the input says 'Prometheus endpoint without API version and resource.' There is a checked checkbox for 'Ignore old alerts' with a note: 'If set to true, Prometheus source plugin will not send alerts that is created before the plugin start time.' Below that is a section for 'Alert states' with three buttons: 'Firing', 'Pending', and 'Inactive'. The 'Firing' button is selected. There is a section for 'Logging' with the description 'Logging configuration for the plugin.' Below this is a 'Log Level' dropdown menu set to 'Info'. A note below says 'Define log level for the plugin. Ensure that Botkube has plugin logging enabled for standard output.' There is a 'Disable Colors' checkbox which is unchecked, with a note: 'If enabled, disables color logging output.' Finally, there is a section for 'Channel bindings' with the description 'Select channels to which this plugin should be bound.' Below this is a checked checkbox for 'general'.

Botkube excels at automating monitoring tasks by delivering real-time alerts and notifications, which empowers DevOps teams to respond quickly and reduce downtime. This streamlined approach enhances operational efficiency and flexibility in managing alerts and notifications.


Task 2: Resource Scaling


Resource scaling in Kubernetes can be a challenging endeavor, especially in response to fluctuating workloads. Manually adjusting resources can be time-consuming and error-prone, leading to inefficiencies. Botkube offers a valuable solution for automating resource scaling within a team. It simplifies access to Kubernetes clusters by providing actionable notifications and the ability to execute `kubectl`, `helm`, and `GitOps` commands directly from a shared team channel.

hakuna-matata ▾

+ Add a bookmark

Today ▾

 **Mateusz Szostok** 6:01 PM
joined #hakuna-matata along with Botkube.


 **Botkube** APP 6:02 PM
My watch begins for cluster 'labs'! ✂
✖ batch/v1/jobs error
Error occurred for Job **default/oops** in **labs** cluster
Messages:

- Job has reached the specified backoff limit

2023-01-23 6:04:20 PM

@Botkube gh create issue |

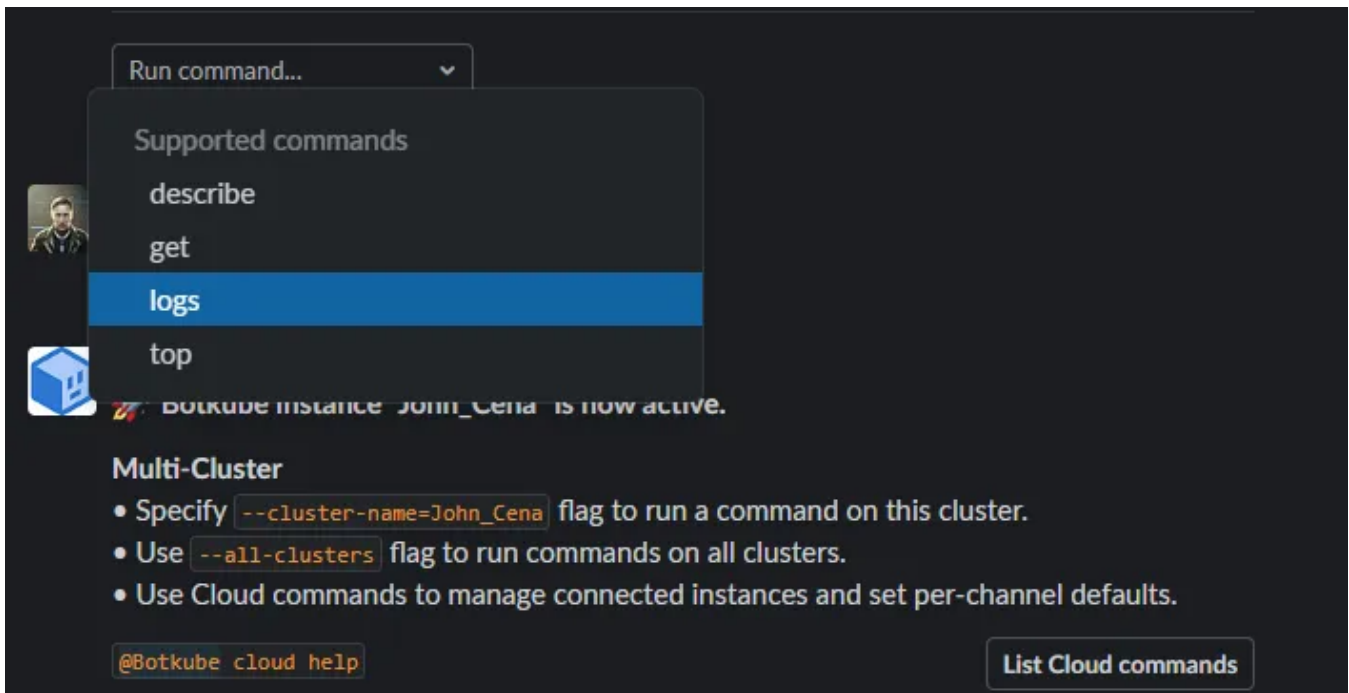
+ | 📎 | 🎤 | 😊 | @ | Aa



Shift + Return to add a new line

One example of this is the ability to automate the process of creating GitHub issues for failing Kubernetes resources such as jobs, deployments, statefulsets, and pods. This plugin can create GitHub issues that include Kubernetes-specific information, making debugging more efficient. The key advantage is that it eliminates the need to connect to your cluster via a terminal, install and run kubectl commands, or manually copy and paste information into a browser. Botkube offers an efficient and user-friendly solution to the challenges of resource scaling in Kubernetes.

Task 3: Kubernetes Log Management



Effective log management is essential to maintaining and troubleshooting Kubernetes clusters. By automating the collection and analysis of logs, Botkube centralizes these critical data points, making them easily accessible for quick analysis. This centralization offers a comprehensive view of all events and errors within the Kubernetes environment, significantly easing the troubleshooting process. With Botkube, you gain enhanced visibility into your cluster's operations, simplifying debugging and accelerating issue resolution. Users can execute `kubectl logs` or `kubectl describe` commands directly from their chat window using simple drop-down menus. This integration not only makes the process more efficient but also halves the time typically spent on log analysis.

This scenario illustrates Botkube's log management functionality: when a response message exceeds the messaging platform's size limit, Botkube efficiently handles this by uploading the response as a text file. This is a typical workflow within Botkube, showcasing its adaptability and ease of use in managing extensive data.

1. Start by listing the pods using the command `@Botkube kubectl get pods -n kube-system``get_pods`

```
get pods -n kube-system on gke-playground
```

NAME	READY	STATUS	RESTARTS	AGE
helm-install-traefik-crd-f9d76	0/1	Completed	0	57d
helm-install-traefik-2nwlv	0/1	Completed	2	57d
coredns-d76bd69b-bxsh8	1/1	Running	1 (27h ago)	6d3h
local-path-provisioner-6c79684f77-nrg8l	1/1	Running	7 (27h ago)	57d
svclb-traefik-bbeed849-bp7fc	2/2	Running	14 (27h ago)	57d
traefik-df4ff85d6-f2mkz	1/1	Running	7 (27h ago)	57d
metrics-server-7cd5fcb6b7-hzvq8	1/1	Running	1 (27h ago)	6d3h

1. Next, to access the logs for a specific pod, such as Traefik, use the command `@Botkube kubectl logs -f traefik-df4ff85d6-f2mkz -n kube-system`.

```
get pods -n kube-system on gke-playground
```

NAME	READY	STATUS	RESTARTS	AGE
helm-install-traefik-crd-f9d76	0/1	Completed	0	57d
helm-install-traefik-2nwlv	0/1	Completed	2	57d
coredns-d76bd69b-bxsh8	1/1	Running	1 (27h ago)	6d3h
local-path-provisioner-6c79684f77-nrg8l	1/1	Running	7 (27h ago)	57d
svclb-traefik-bbeed849-bp7fc	2/2	Running	14 (27h ago)	57d
traefik-df4ff85d6-f2mkz	1/1	Running	7 (27h ago)	57d
metrics-server-7cd5fcb6b7-hzvq8	1/1	Running	1 (27h ago)	6d3h

1. If the output is too large, Botkube will upload the response as a file in a reply thread. You can view the complete output by expanding this thread.

1. For more specific log details, such as filtering for log lines containing the word “Configuration,” enter “Configuration” into the Filter Output input box and hit enter. Botkube will then present you with the filtered log results.

**Botkube** APP Today at 5:05 PM`logs -f traefik-df4ff85d6-f2mkz -n kube-system` on `gke-playground`

Response.txt ▾

```
1 time="2022-10-17T09:26:20Z" level=info msg="Configuration loaded
  from flags."
2 time="2022-10-17T09:26:20Z" level=error msg="Cannot create
  service: service not found" ingress=ingress namespace=botkube
  serviceName=foo
  servicePort="&ServiceBackendPort{Name:,Number:3001,}"
  providerName=kubernetes
3 time="2022-10-17T09:26:20Z" level=error msg="Cannot create
  service: service not found" serviceName=foo
  providerName=kubernetes ingress=ingress namespace=botkube
```

2 replies

**Botkube** APP < 1 minute ago**Filter output**

↵ Press 'enter' to submit

**Botkube** APP < 1 minute ago`logs -f traefik-df4ff85d6-f2mkz -n kube-system --filter=Configuration` on `gke-playground` by `@huseyin`

```
time="2022-10-17T09:26:20Z" level=info msg="Configuration loaded from
flags."
```

Task 4: GitOps Workflows

hakuna-matata

+ Add a bookmark

```
► checking prerequisites
✓ Kubernetes 1.25.0+k3s1 >=1.24.0-0
✓ prerequisites checks passed
```

Today

```
Flux create source git webapp-latest --url=https://github.com/stefanprodan/podinfo --branch=master --interval=3m on 1abs by @Mateusz Szostok
```

```
+ generating GitRepository source
► applying GitRepository source
✓ GitRepository source created
⊙ waiting for GitRepository source reconciliation
✓ GitRepository source reconciliation completed
✓ fetched revision: master@sha1:dd3869b1a177432b60ea1e3ba99c10fc9db850fa
```

Only visible to you

podinfo

```
Name:      podinfo
Revision:   master@sha1:dd3869b1
Suspended: False
Ready:      True
Message:    stored artifact for revision 'master@sha1:dd3869b1'
```

Actions

Raw output

**Mateusz Szostok** 11:06 AM

@Botkube flux diff ks podinfo --path ./kustomize --github-ref 5

1

Message #hakuna-matata

+ Aa 😊 @ 📎 🔊 🗑

▶

Implementing GitOps practices in Kubernetes is crucial for engineering teams as it automates the management of Kubernetes resources through synchronization with Git repositories. This approach fosters collaboration, facilitates version control, and simplifies issue tracking within the team. Botkube plays a significant role in enhancing GitOps workflows by reducing manual intervention and providing real-time monitoring and collaboration capabilities.

A notable feature is the Botkube Flux plugin, which streamlines the integration of Kubernetes clusters, GitHub repositories, and the Flux CLI. This plugin empowers users to execute Flux CLI commands directly from their preferred communication platforms, ensuring mobile-friendly interactivity. It automates tasks like monitoring GitHub pull requests, notifying users about new pull requests, and rendering event-aware buttons for running diff reports. Without the plugin, these tasks would involve multiple manual steps, elevating the risk of errors and hindering productivity.

In summary, Botkube's GitOps plugins bridge the gap between GitOps tools and communication platforms, offering an efficient, interactive, and user-friendly

approach to managing Kubernetes deployments.

Task 5: K8s Configuration Management



Botkube APP 12:15 AM

`helm install --repo https://charts.bitnami.com/bitnami psql postgresql` on `labs`

```
NAME: psql
LAST DEPLOYED: Thu Dec 22 00:15:38 2022
NAMESPACE: botkube
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: postgresql
CHART VERSION: 12.1.6
APP VERSION: 15.1.0
```

Filter output

String pattern to filter by

Open in app ↗

Sign up

Sign in

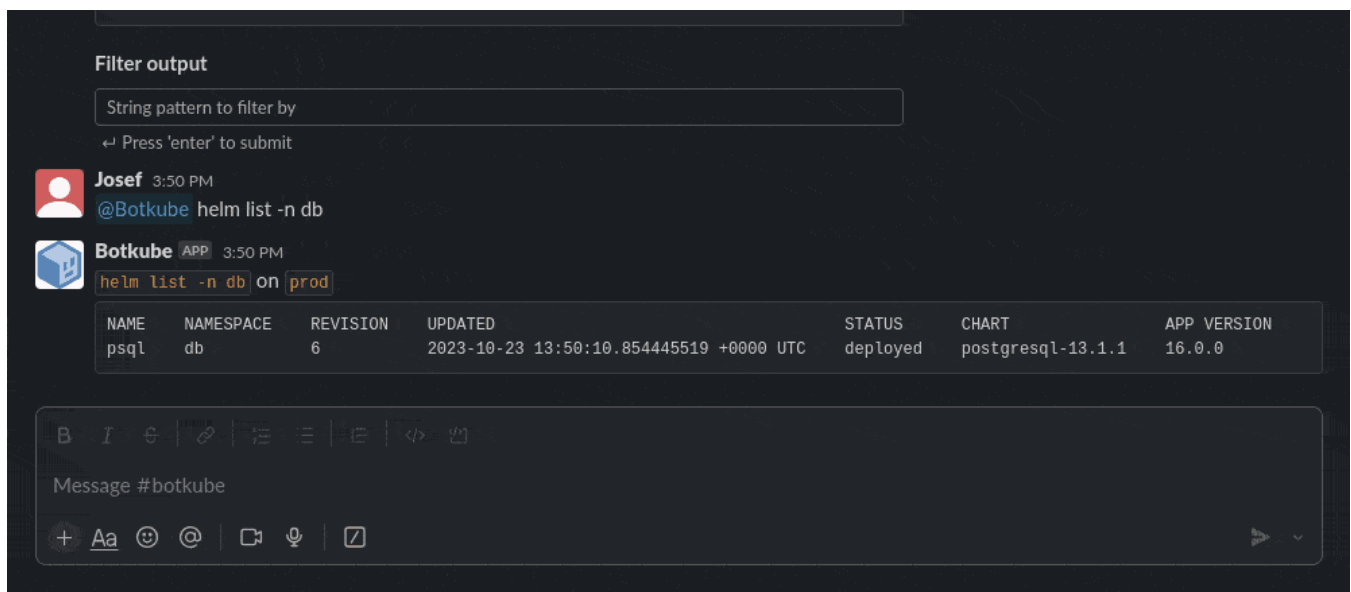


Search



Managing Kubernetes configurations can be challenging, especially when done manually. Traditional methods involve navigating through multiple interfaces and complex command lines, which can be time-consuming and prone to errors. This complexity often poses a significant hurdle in efficiently managing Helm releases.

Botkube's Helm executor plugin enhances this process. This plugin allows users to run Helm commands directly from their chat interface. This integration streamlines Kubernetes configuration management, making it more accessible and less error-prone compared to manual methods.



With the Botkube Helm plugin, users can easily view, manage, and roll back Helm releases. This is helpful for application developers, who can quickly check the status of their releases without switching between different tools. By integrating Helm command functionalities into common communication platforms, Botkube not only simplifies Kubernetes management but also aligns it with contemporary workflows, making the entire process more efficient and user-friendly.

Conclusion

Botkube can automate a variety of Kubernetes tasks and can enable your DevOps team to build a more productive workflow. Each automation demonstrates Botkube's ability to simplify complex processes, from automating monitoring setups and streamlining log management to enhancing GitOps workflows and improving configuration management with its Helm plugin. Botkube not only addresses the manual challenges inherent in these tasks but also integrates these functionalities into familiar communication platforms, making Kubernetes management more accessible, efficient, and aligned with modern collaborative practices.

Getting Started with Botkube Today

Try it for yourself! Follow our step-by-step [tutorial](#) to set up Botkube using our web app. We're excited to hear how you use Botkube. Create a support ticket directly in the dashboard, share your stories with us in the [Botkube Slack community](#). We'll even send you some cool swag as a thank you.

Kubernetes

Automation

DevOps

Chatops

Monitoring