Managing Kubernetes without losing your cool

Giant Swarm Webinar

May 3rd 2022

Hi 👋,

I'm **Marcus Noble,** a *platform engineer* at **Giant Swarm** ← *We're hiring*

I'm found around the web as ✨***AverageMarcus***✨ in most places and **@Marcus_Noble_** on Twitter 🐦

~5 years experience running Kubernetes in production environments.

💖

Summary

My 10 tips for working with Kubernetes

#1 → #5

Anyone can start using these today

#6 → #7

Good to know a little old-skool ops first

#8 → #10

Good have some programming knowledge

#0 - Pay someone else to deal with it

#1 - Love your terminal

#1 - Love your terminal

- ★ Bash? ZSH? Fish? - Doesn't matter as long as you're comfortable with it.
- ★ "rc" files - e.g. `.bashrc`, `.zshrc`
  These set configuration for each terminal session you open.
- ★ `alias` - easily create your own terminal commands
- ★ Look for "dotfiles" on GitHub - e.g. https://github.com/averagemarcus/dotfiles

#2 - Learn to love `kubectl`

#2 - Learn to love `kubectl`

Tip #1 in action

★ Add `alias k='kubectl'` to your .bashrc / .zshrc / .whateverrc

```
k get pods -A
```

★ The official docs offer a single page view of all built in commands: kubernetes.io/docs/reference/generated/kubectl/kubectl-commands

★ `kubectl explain` is your friend! Find out what any property of any Kubernetes resource is for. ➡

```
k explain pods.spec.containers
KIND:     Pod
VERSION:  v1

RESOURCE: containers <[]Object>

DESCRIPTION:
     List of containers belonging to the pod. Containers cannot currently be
     added or removed. There must be at least one container in a Pod. Cannot be
     updated.

     A single application container that you want to run within a pod.

FIELDS:
   args <[]string>
     Arguments to the entrypoint. The docker image's CMD is used if this is not
     provided. Variable references $(VAR_NAME) are expanded using the
     container's environment. If a variable cannot be resolved, the reference in
     the input string will be unchanged. Double $$ are reduced to a single $,
     which allows for escaping the $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)" will
     produce the string literal "$(VAR_NAME)". Escaped references will never be
     expanded, regardless of whether the variable exists or not. Cannot be
     updated.

   command      <[]string>
     Entrypoint array. Not executed within a shell. The docker image's
     ENTRYPOINT is used if this is not provided. Variable references $(VAR_NAME)
```

#3 - Multiple kubeconfigs

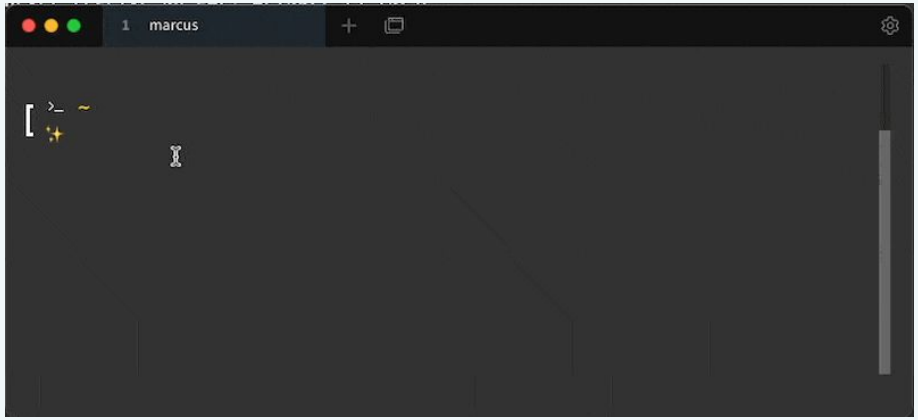
#3 - Multiple kubeconfigs

- ★ Quick switch between different Kubernetes contexts (clusters) and between different namespaces.

- ★ `kubectx` and `kubens`
  https://github.com/ahmetb/kubectx

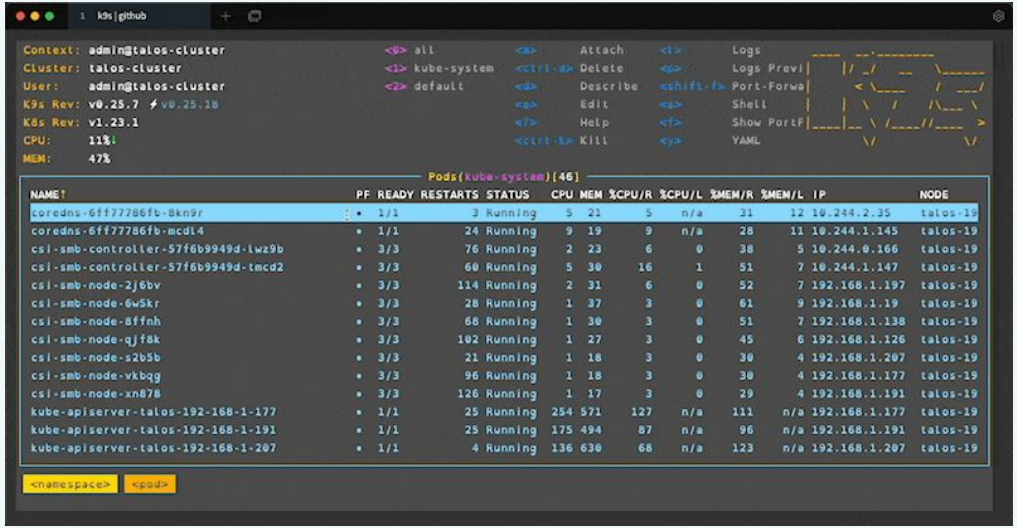
- ★ `kubie`
  https://github.com/sbstp/kubie

- ★ `kubeswitch`
  https://github.com/danielfoehrKn/kubeswitch



#4 - k9s

#4 - k9s



github.com/derailed/k9s

#5 - kubectl plugins

#5 - kubectl plugins

- ★ Any command in your `$PATH` that is prefixed with `kubectl-` becomes a kubectl plugin

- ★ Krew - package manager for kubectl plugins
  github.com/kubernetes-sigs/krew
- ★ Install plugins with:
  `kubectl krew install <PLUGIN NAME>`

```
$ cat kubectl-hello
#!/bin/bash
echo "Hello, Kube"


$ kubectl hello
Hello, Kube
```

- ★ Some of my fave plugins:
  - **stern** - Multi-pod/container log tailing
  - **tree** - Show hierarchy of resources based on ownerReferences
  - **outdated** - Find containers with outdated images
  - **gs** - Giant Swarm's plugin for working with our managed clusters

Summary

My 10 tips for working with Kubernetes

✅

~~#1 → #5~~

~~Anyone can start using these today~~

#6 → #7

Good to know a little old-skool ops first

#8 → #10

Good have some programming knowledge

#6 - kshell / kubectl debug

#6 - kshell / kubectl debug

Launch a temporary pod running a bash shell for cluster debugging

Tip #1 in action, again

```
alias kshell='kubectl run \
    -it                   \
    --image bash          \
    --restart Never       \
    --rm                  \
    shell'
```

Need more tools? Replace this with ubuntu

#6 - kshell / kubectl debug

Launch a temporary pod running a bash shell for cluster debugging

```
# kshell
If you don't see a command prompt, try pressing enter.
bash-5.1# nslookup google.com
Server:         1.1.1.1
Address:        1.1.1.1:53

Non-authoritative answer:
Name:   google.com
Address: 142.250.187.206
```

#6 - kshell / kubectl debug

Debugging a running pod - `kubectl exec`

```
# kubectl exec my-broken-pod -it -- sh

/app #
```

Note:

- ★ Needs a shell environment within the container
- ★ Limited to what's available in the container (or what you can pull in from the 'net)
- ★ Container needs to be running

#6 - kshell / kubectl debug

Debugging a running pod - `kubectl exec`

```
# kubectl exec my-broken-pod -it -- sh
error: Internal error occurred: error executing command in
container: failed to exec in container: failed to start exec……
```

Debugging a running pod - `kubectl debug` Requires Kubernetes v1.23

```
# kubectl debug -it --image bash my-broken-pod
Defaulting debug container name to debugger-gprmk.
If you don't see a command prompt, try pressing enter.
bash-5.1#
```

#6 - kshell / kubectl debug

Example - investigate a CrashLooping pod

```
# kubectl run debug-demo --image=bash -- exit 1

# kubectl get pods debug-demo
NAME          READY   STATUS             RESTARTS      AGE
debug-demo    0/1     CrashLoopBackOff   2 (20s ago)   44s
```

This will prevent us from `kubectl exec` into the pod

```
# kubectl debug -it --image bash debug-demo
Defaulting debug container name to debugger-5mkjj.
If you don't see a command prompt, try pressing enter.
bash-5.1#
```

#6 - kshell / kubectl debug

When to use what:

	`kshell`	`kubectl exec`	`kubectl debug`
Multiple workloads experiencing network issues	✅		
Workload not running as expected but not CrashLooping and isn't a stripped down image (e.g. not Scratch / Distroless)		✅	
Workload not running as expected but not CrashLooping and has an image based on Scratch / Distroless or similar			✅
Workload is CrashLooping			✅

#7 - kube-ssh

#7 - kube-ssh

★ github.com/AverageMarcus/kube-ssh (or github.com/giantswarm/kubectl-enter)

★ Give ssh-*like* access to a node's underlying host, great for instances where nodes are provisioned without SSH or access is blocked.

```
sh -c "$(curl -sSL https://raw.githubusercontent.com/AverageMarcus/kube-ssh/master/ssh.sh)"
[0] - ip-10-18-21-146.eu-west-1.compute.internal
[1] - ip-10-18-21-234.eu-west-1.compute.internal
[2] - ip-10-18-21-96.eu-west-1.compute.internal
Which node would you like to connect to? 1

If you don't see a command prompt, try pressing enter.
[root@ip-10-18-21-234 ~]#
```

Please verify any script before you execute it!

Why? - I prefer to use ephemeral instances with the minimal needed to run Kubernetes, no sshd, no port 22 open etc. but there are times when you just need to check what's actually going on with the underlying host machine.

#7 - kube-ssh

How it works

ssh.sh

```
kubectl run kube-ssh --restart=Never -it --rm --image overridden
--overrides '
{
  "spec": {
    "hostPID": true,
    "hostNetwork": true,
    '"${NODE_SELECTOR}"'
    "tolerations": [{ "operator": "Exists" }],
    "containers": [
      {
        "name": "kube-ssh",
        "image": "averagemarcus/kube-ssh:latest",
        "stdin": true,
        "tty": true,
        "securityContext": { "privileged": true }
      }
    ]
  }
}' --attach "$@"
```

Ensure we can run on any node

Container image containing `nsenter`

Allows us to switch to a host PID

Dockerfile

```
FROM debian:buster as builder

WORKDIR /tmp
RUN apt-get update && \
    apt-get install -yq \
    make gcc gettext autopoint \
    bison libtool automake pkg-config

ADD https://github.com/karelzak/util-linux/archive/v2.34.tar.gz .
RUN tar -xf v2.34.tar.gz && \
    mv util-linux-2.34 util-linux \
    cd util-linux && \
    ./autogen.sh && \
    ./configure && \
    make LDFLAGS="--static" nsenter

FROM scratch
COPY --from=builder /tmp/util-linux/nsenter /
ENTRYPOINT [
  "/nsenter", "--all", "--target=1", "--", "su", "-"
]
```

#7 - kube-ssh

This won't work with Talos, for example

- ★ Some caveats - underlying host needs a shell
- ★ You require enough permissions to launch pods with privileged securityContext - RBAC, PSPs and Admission Controllers could all potentially block this. (This could also be considered a benefit to this approach over traditional SSH)
- ★ Not a real SSH session - so no key authentication, file transfer, port forwarding
- ★ `nsenter` - "The nsenter command executes program in the namespace(s) that are specified in the command-line options." (Man page)

Summary

My 10 tips for working with Kubernetes

✅ ~~#1 → #5~~

~~Anyone can start using these today~~

✅ ~~#6 → #7~~

~~Good to know a little old-skool ops first~~

#8 → #10

Good have some programming knowledge

#8 - Webhooks

#8 - Webhooks

↩ OK, actually 3 but we're ignoring CRD conversion webhooks

★ Two types of webhooks:

`ValidatingWebhook`	Ability to block actions against the API server if fails to meet given criteria.
`MutatingWebhook`	Modify requests before passing them on to the API server.

★ Implement more advanced access control than is possible with RBAC. [Restricting cluster-admin permissions]

★ Add default labels to resources as they're created.

★ Enforce policies such as not using `latest` as an image tag or ensuring all workloads have resource requests/limits specified.

★ "Hotfix" for security issues (e.g. mutating all pods to include a `LOG4J_FORMAT_MSG_NO_LOOKUPS` env var to prevent Log4Shell exploit). [Log4Shell Mitigation]

#8 - Webhooks

- ★ Build your own operator to implement custom logic
- ★ Kyverno - Kubernetes native policy management. Create `Policy` and `ClusterPolicy` resources to define rules in YAML
- ★ OPA Gatekeeper - Policy management built on top of Open Policy Agent

Kyverno Policy

```yaml
apiVersion: kyverno.io/v1
kind: ClusterPolicy
metadata:
  name: block-bulk-certconfigs-delete
  annotations:
    policies.kyverno.io/description: Block delete all bug in CLI
spec:
  rules:
  - name: block-bulk-certconfigs-delete
    match:
      any:
      - resources:
          kinds: [CertConfig]
    preconditions:
      any:
      - key: ""
        operator: Equals
        value: ""
    validate:
      message: |
        Your current kubectl-gs version contains a critical bug
      deny:
        conditions:
        - key: ""
          operator: In
          value: [DELETE]
```

Taken from our Restricting cluster-admin permissions blog post

#8 - Webhooks

Notes:

This is one of the main causes we see of clusters being down

- Where possible always avoid applying webhooks to resources in `kube-system`. This can cause a deadlock if those pods try to come up before the webhook service is available.
- Be aware of the `failurePolicy` property - it defaults to “fail” which can cause troubles if your service handling the webhook goes down.
- The `reinvocationPolicy` property can be set if changes made by a `MutatingWebhook` may need to go through other defined webhooks again.
- Ordering - first `MutatingWebhooks` then `ValidatingWebhooks`. No guaranteed control of order within these two phases.

#9 - Kubernetes API

#9 - Kubernetes API

All Kubernetes operations are done via the API - kubectl uses it, in-cluster controllers use it, the scheduler uses it and ***you can use it too!*** ✨

The API can also be extended by either:

- the creation of Custom Resource Definitions (CRDs)
- implementing an Aggregation Layer (such as what metrics-server implements).

We're not going to cover this today

#9 - Kubernetes API

You can easily try out the API using `kubectl` with the `--raw` argument.

This is the equivalent to `kubectl get pods -n default`

```
# kubectl get --raw /api/v1/namespaces/default/pods
{"kind":"PodList","apiVersion":"v1","metadata":{"selfLink":...
```

If no host is provided kubectl will use the API of the current context.

HTTP Method	Kubectl command
GET	`kubectl get --raw`
POST	`kubectl create --raw`
DELETE	`kubectl delete --raw`
PUT	`kubectl replace --raw`

#9 - Kubernetes API

Not sure what APIs are available?

```
# kubectl api-resources
NAME                              SHORTNAMES   APIVERSION                     NAMESPACED   KIND
bindings                                       v1                             true         Binding
componentstatuses                 cs           v1                             false        ComponentStatus
configmaps                        cm           v1                             true         ConfigMap
endpoints                         ep           v1                             true         Endpoints
deployments                       deploy       apps/v1                        true         Deployment
```

API endpoint format:

From the "Name" column above

```
/{API_VERSION}/namespace/{NAMESPACE_NAME}/{RESOURCE_KIND}/{NAME}
```

Only if namespaced column is 'true'

Optional name of the specific resource

#9 - Kubernetes API

Not sure what APIs are available?

```
# kubectl api-resources
NAME                              SHORTNAMES   APIVERSION                     NAMESPACED   KIND
bindings                                       v1                             true         Binding
componentstatuses                 cs           v1                             false        ComponentStatus
configmaps                        cm           v1                             true         ConfigMap
endpoints                         ep           v1                             true         Endpoints
deployments                       deploy       apps/v1                        true         Deployment
```

This is the "core" API

If `APIVERSION` is just v1 the endpoint starts with `/api/v1/`

E.g. `/api/v1/componentstatuses`

#9 - Kubernetes API

Not sure what APIs are available?

```
# kubectl api-resources
NAME                              SHORTNAMES   APIVERSION                     NAMESPACED   KIND
bindings                                       v1                             true         Binding
componentstatuses                 cs           v1                             false        ComponentStatus
configmaps                        cm           v1                             true         ConfigMap
endpoints                         ep           v1                             true         Endpoints
deployments                       deploy       apps/v1                        true         Deployment
```

Otherwise, the endpoint starts with `/apis/{APIVERSION}/`

Note the extra 's'

E.g. `/apis/apps/v1/`

#9 - Kubernetes API

Not sure what APIs are available?

```
# kubectl api-resources
NAME                  SHORTNAMES   APIVERSION   NAMESPACED   KIND
bindings                           v1           true         Binding
componentstatuses     cs           v1           false        ComponentStatus
configmaps            cm           v1           true         ConfigMap
endpoints             ep           v1           true         Endpoints
deployments           deploy       apps/v1      true         Deployment
```

The `NAMESPACED` column indicates if the resource is bound to a namespace.

If false: `/api/v1/componentstatuses`

If true: `/apis/apps/v1/namespaces/default/deployment`

Name of the namespace to use

Omit this to search all namespaces

#9 - Kubernetes API

Resources:

- kubernetes/client-go - the official Golang module for interacting with the Kubernetes API
- Kubernetes Provider for Terraform (actually uses the above Go module under the hood)
- kubernetes-client org on GitHub has many official clients in different languages

Where is this useful?

★ Building our own CLI / desktop tooling (e.g. k9s, Lens).

★ Cluster automation - resources managed by CI, CronJobs, etc.

★ Building our own operators to extend Kubernetes.

#10 - CRDs & Operators

#10 - CRDs & Operators

Extend Kubernetes' built-in API and functionality with your own Custom Resource Definitions (CRDs) and business logic (operators).

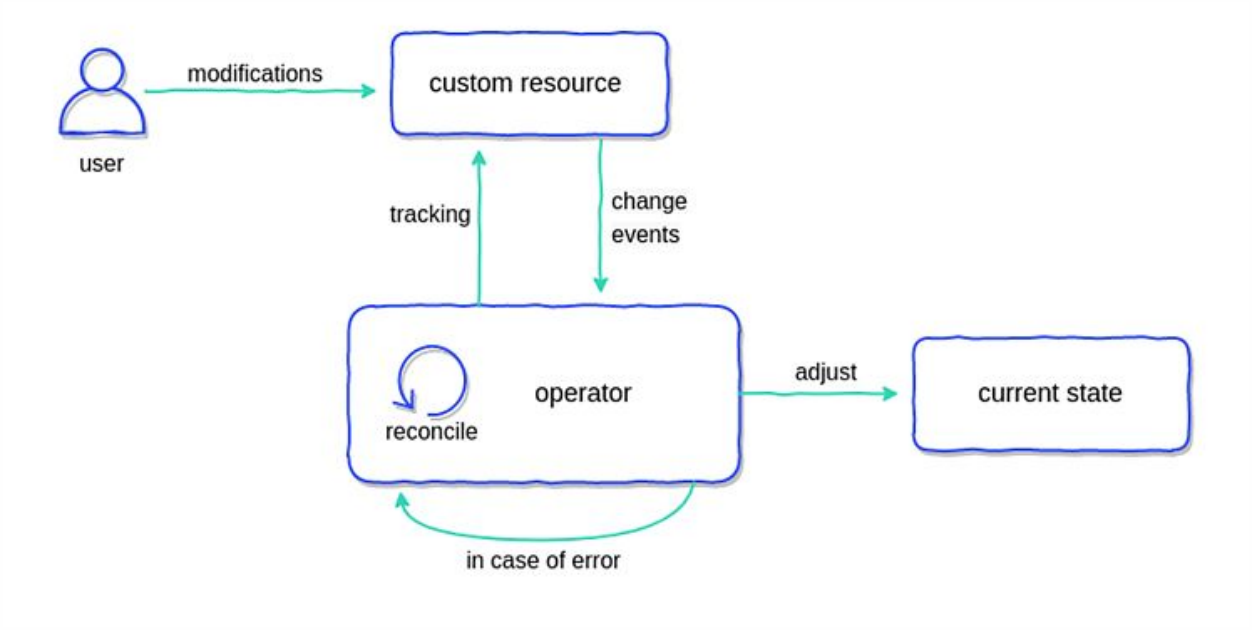
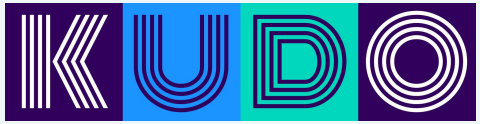


Image credits: Container Solutions
https://blog.container-solutions.com/kubernetes-operators-explained

#10 - CRDs & Operators

Frameworks



Metacontroller

References

- https://kubernetes.io/docs/concepts/extend-kubernetes/operator/
- https://blog.container-solutions.com/kubernetes-operators-explained
- https://operatorhub.io/ - Directory of existing operators

Videos

Summary

My 10 tips for working with Kubernetes

✅ ~~#1 → #5~~

~~Anyone can start using these today~~

✅ ~~#6 → #7~~

~~Good to know a little old-skool ops first~~

✅ ~~#8 → #10~~

~~Good have some programming knowledge~~

Recap

#1 - Love your terminal

#2 - Learn to love kubectl

#3 - Multiple kubeconfigs

#4 - k9s

#5 - Kubectl plugins

#6 - kshell / kubectl debug

#7 - kube-ssh

#8 - Webhooks

#9 - Kubernetes API

#10 - CRDs & Controllers

Thank You

🧡