

Debugging Go in Kubernetes

by Mike Schinkel on *May 4th, 2023*

github.com/mikeschinkel

Or more accurately:

Debugging Go programs
running in a Kubernetes Pod
using Delve with GoLand

About Me

25+ years involved with dev as

- A developer,
- A trainer of devs, and
- A startup CEO selling to devs.

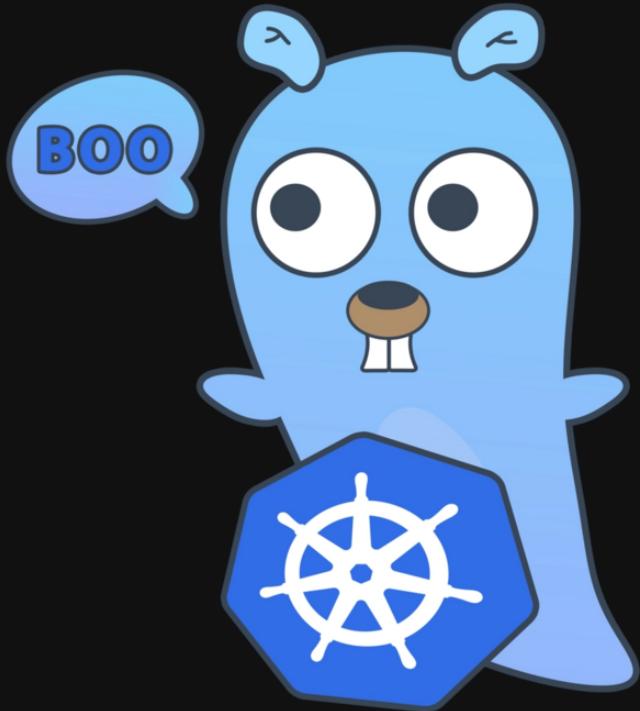
Contractor with these teams:

- SingleStore K8s Operator
- HPE Greenlake On-Prem
- Numerous lesser-known others

Currently:

- Free-agent dev: Go/K8s/CI-CD
- New YouTuber —
"Homelabbing for Developers"
youtube.com/@gearboxworks





Prerequisites & Assumptions

- Go programming experience
- Familiar with K8s YAML
- Can read Dockerfiles
- Can read Bash scripts
- Knowledge of Makefiles

Find the code
on Github

github.com/mikeschinkel/go-debuggable-k3d-pod

Open-source tools we'll use

1.

K3s — k3s.io

A lightweight distro of
K8s

2.

k3d — k3d.io

Runs a K3s cluster in a
Docker container

3.

Delve — github.com/go-delve

The defacto-standard
debugger for Go

Source for our Go "App"

```
● ● ●  
1 package main  
2  
3 import (  
4     "fmt"  
5     "time"  
6 )  
7  
8 const (  
9     DateFormat = "2006-01-02 3:04:05PM"  
10    DelayTime = 5  
11 )  
12  
13 // Debug with Go Remote = localhost:8765  
14 func main() {  
15     fmt.Println("Starting Debuggable Pod")  
16     for {  
17         dt := time.Now().Format(DateFormat)  
18         fmt.Printf("\n[%s] Hello World!", dt)  
19         time.Sleep(DelayTime * time.Second)  
20     }  
21 }
```

The go.mod



```
1 module debuggable-pod
2
3 go 1.19
```

The k3d.yaml



```
1  ---
2  apiVersion: k3d.io/v1alpha4
3  kind: Simple
4  servers: 1
5  kubeAPI:
6    hostPort: "6443"
7  image: rancher/k3s:v1.23.8-k3s1
8  registries:
9    use:
10      - k3d-registry.localhost:5000
11  options:
12    k3d:
13      wait: true
14      timeout: "60s"
15  ports:
16    - port: 8765:30800
17      nodeFilters:
18        - server:0
```

The pod.yaml – 1st part



```
1  ---
2  apiVersion: v1
3  kind: Pod
4  metadata:
5    name: debuggable-pod
6    labels:
7      debugger: dlv
8    annotations:
9      container.apparmor.security.beta.kubernetes.io/debuggable-app-container: unconfined
10   spec:
11     restartPolicy: Always
12     containers:
13       - name: debuggable-app-container
14         image: k3d-registry.localhost:5000/debuggable-app-image
15         imagePullPolicy: Always
16         securityContext:
17           capabilities:
18             add:
19               - SYS_PTRACE
20         ports:
21           - containerPort: 32345
22             hostPort: 32345
23             protocol: TCP
```

The pod.yaml – 2nd part



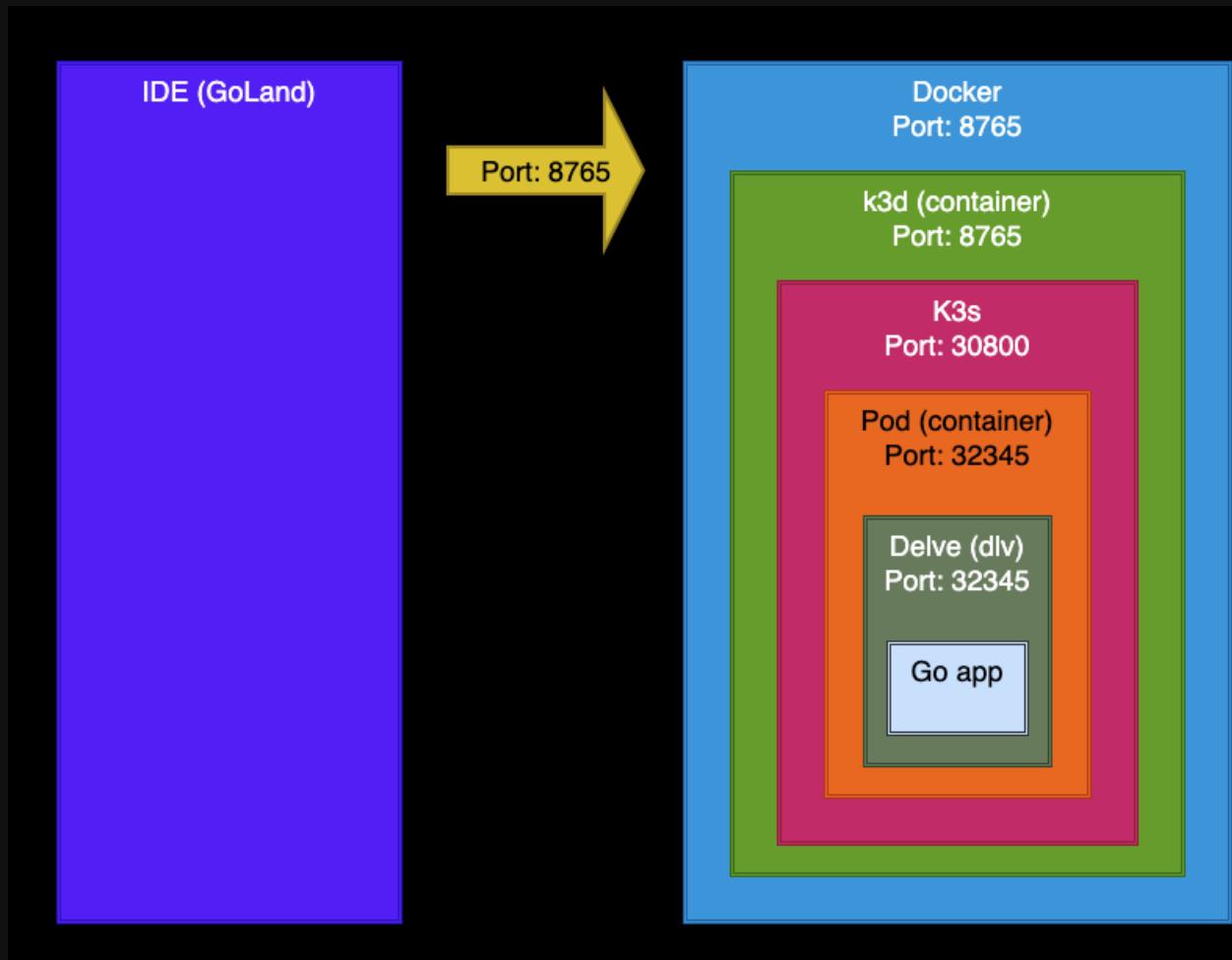
```
1  ---
2  apiVersion: v1
3  kind: Service
4  metadata:
5    name: dlv-port-service
6  spec:
7    type: NodePort
8    selector:
9      debugger: dlv
10   ports:
11     - name: dlv-port
12       protocol: TCP
13       port: 32345
14       nodePort: 30800
```

The Dockerfile for the Pod



```
1 FROM golang:1.19-alpine AS builder
2
3 ENV CGO_ENABLED=0
4
5 RUN apk update \
6     && apk add --no-cache git \
7     && go install github.com/go-delve/delve/cmd/dlv@v1.9.1
8
9 WORKDIR /app
10
11 COPY . /app
12
13 RUN go build -o debuggable-go-app -gcflags="all=-N -l" /app/main.go
14
15 EXPOSE 32345 32345
16
17 CMD [ "dlv", \
18       "--listen=:32345", \
19       "--headless=true", \
20       "--api-version=2", \
21       "--accept-multipleclients", \
22       "exec", "/app/debuggable-go-app" ]
```

Components/Ports



Preparing the Example

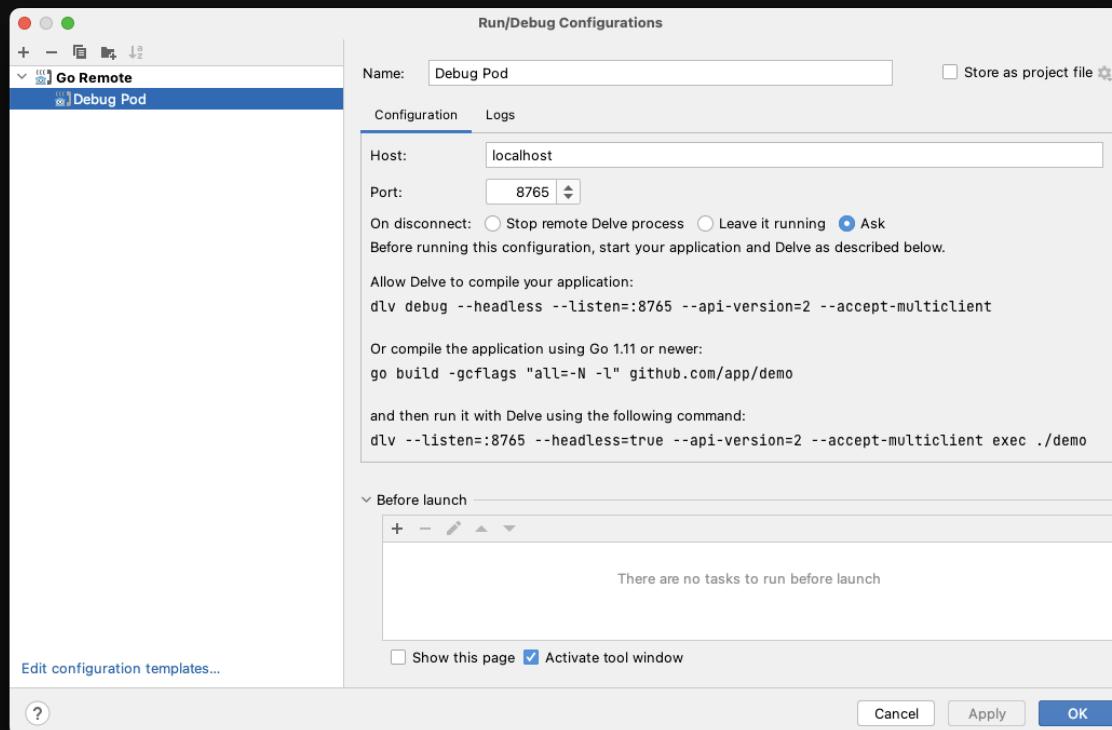


```
1 git clone https://github.com/mikeschinkel/go-debuggable-k3d-pod
2
3 make install    # Installs required software
4
5 make init       # Initializes k3d registry and cluster
6
7 make build      # Builds the Go App and Docker container
8
9 make deploy     # Push container to K8s registry, apply Pod YAML
```

Demo #1: Commands to Try

Commands	Notice
docker ps	Images, ports for proxy
kubectl get all	Names, service ports
kubectl exec -it pod/debuggable-pod -- sh	
top	PID 1's command
ls -al	files
kubectl logs pods/debuggable-pod	Output

Demo #2: Debugging in the IDE



What did we learn? Ports!

Component	Port
The K8s Cluster	8765
The K8s Pod	30800
The Go App	32345

You can use any available port numbers here,
they just need to be used in the right places.

Questions?

(Again, find the code here:)

github.com/mikeschinkel/go-debuggable-k3d-pod

Thank you!

Who am I, again?	Mike Schinkel
Code	github.com/mikeschinkel/go-debuggable-k3d-pod
These Slides	slides.com/mikeschinkel/debugging-go-apps-in-k8s
YouTube channel (Coming soon)	www.youtube.com/@gearboxworks
Available for contract work	mike@newclarity.net
My LinkedIn	www.linkedin.com/in/mikeschinkel