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

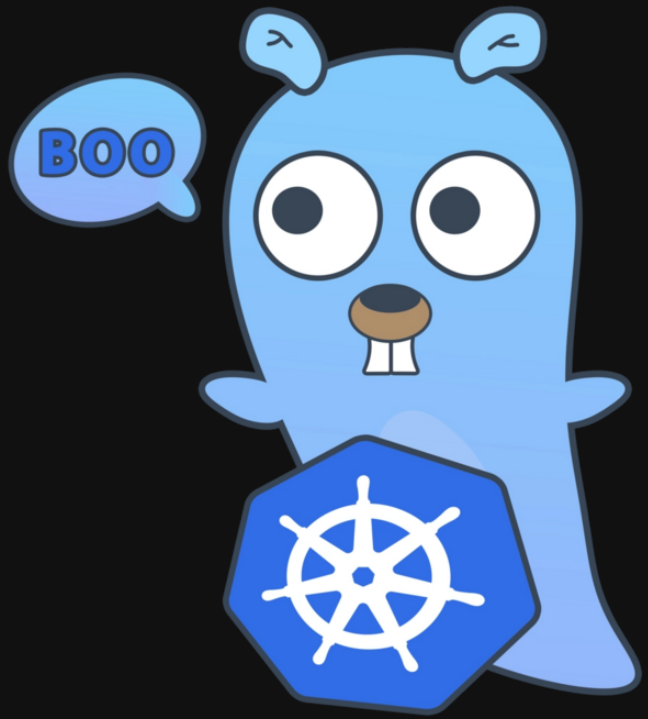
- A developer,
- A trainer of devs, and
- A startup CEO selling to devs.

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"

```go
package main

import (
        "fmt"
        "time"
)

const (
        DateFormat = "2006-01-02 3:04:05PM"
        DelayTime  = 5
)

// Debug with Go Remote = localhost:8765
func main() {
        fmt.Println("Starting Debuggable Pod")
        for {
                dt := time.Now().Format(DateFormat)
                fmt.Printf("\n[%s] Hello World!", dt)
                time.Sleep(DelayTime * time.Second)
        }
}
```

The go.mod

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

The k3d.yaml

```yaml
---
apiVersion: k3d.io/v1alpha4
kind: Simple
servers: 1
kubeAPI:
  hostPort: "6443"
image: rancher/k3s:v1.23.8-k3s1
registries:
  use:
    - k3d-registry.localhost:5000
options:
  k3d:
    wait: true
    timeout: "60s"
ports:
  - port: 8765:30800
    nodeFilters:
      - server:0
```

The pod.yaml – 1st part

```yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: debuggable-pod
  labels:
    debugger: dlv
  annotations:
    container.apparmor.security.beta.kubernetes.io/debuggable-app-container: unconfined
spec:
  restartPolicy: Always
  containers:
  - name: debuggable-app-container
    image: k3d-registry.localhost:5000/debuggable-app-image
    imagePullPolicy: Always
    securityContext:
      capabilities:
        add:
        - SYS_PTRACE
    ports:
    - containerPort: 32345
      hostPort: 32345
      protocol: TCP
```

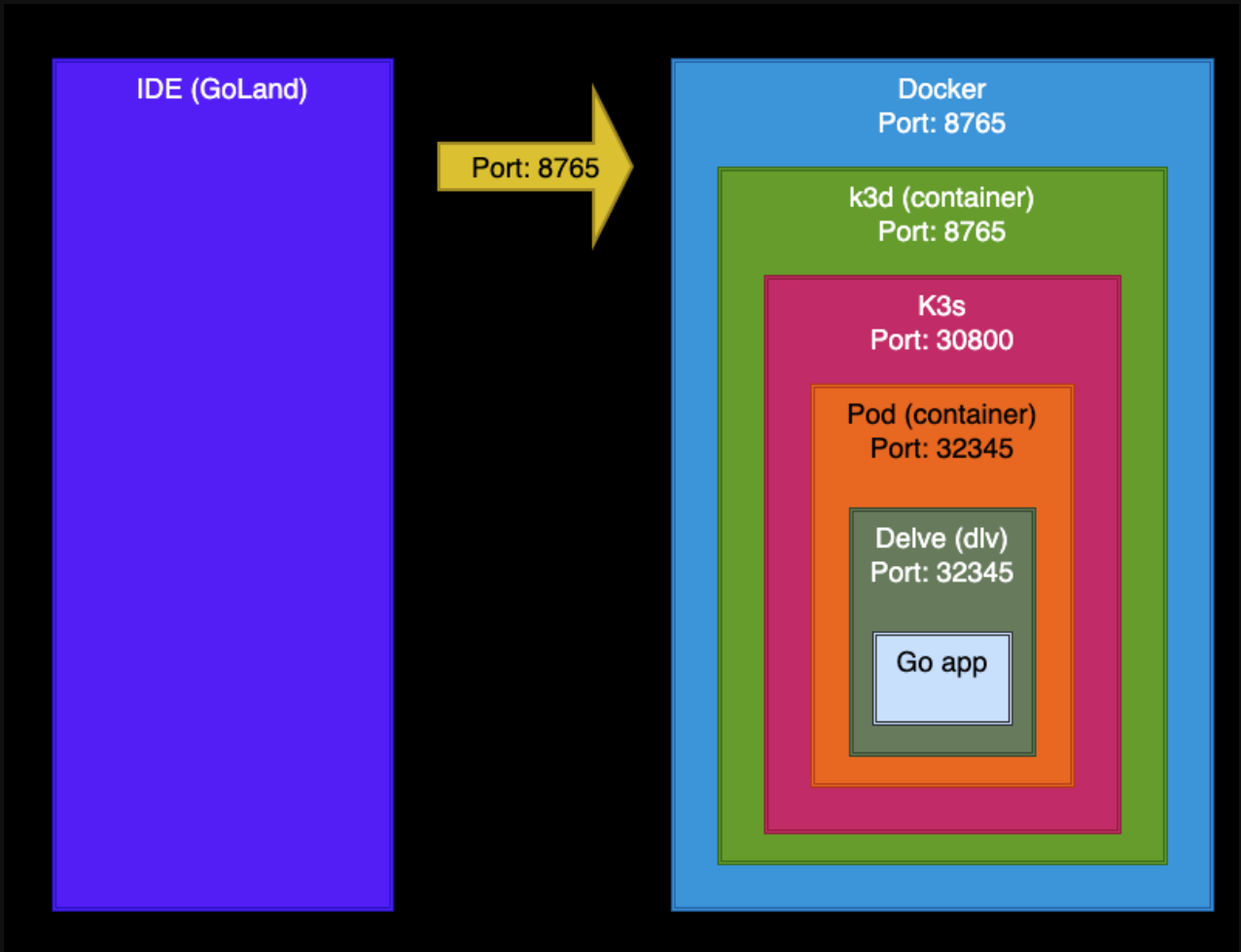
The pod.yaml – 2nd part

```yaml
---
apiVersion: v1
kind: Service
metadata:
  name: dlv-port-service
spec:
  type: NodePort
  selector:
    debugger: dlv
  ports:
  - name: dlv-port
    protocol: TCP
    port: 32345
    nodePort: 30800
```

The Dockerfile for the Pod

```
FROM golang:1.19-alpine AS builder

ENV CGO_ENABLED=0

RUN apk update \
    && apk add --no-cache git \
    && go install github.com/go-delve/delve/cmd/dlv@v1.9.1

WORKDIR /app

COPY . /app

RUN go build -o debuggable-go-app -gcflags="all=-N -l" /app/main.go

EXPOSE 32345 32345

CMD [ "dlv", \
    "--listen=:32345", \
    "--headless=true", \
    "--api-version=2", \
    "--accept-multiclient", \
    "exec", "/app/debuggable-go-app"]
```

Components/Ports



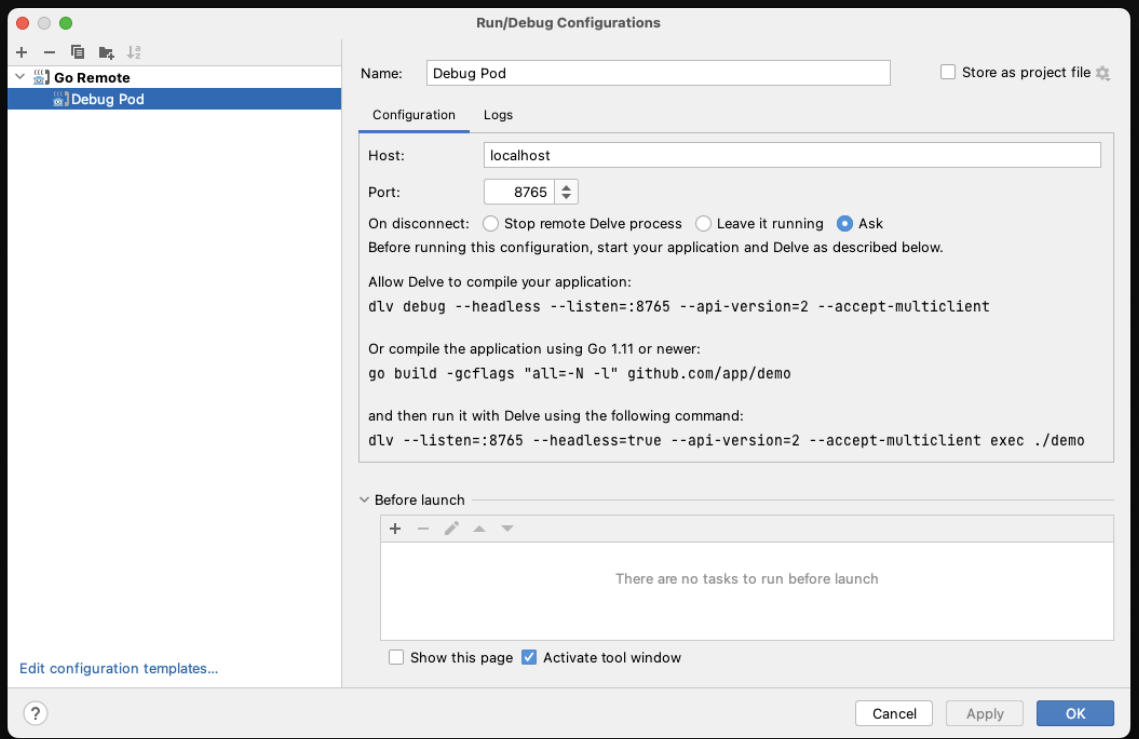
Preparing the Example

```
git clone https://github.com/mikeschinkel/go-debuggable-k3d-pod

make install  # Installs required software

make init     # Initializes k3d registry and cluster

make build    # Builds the Go App and Docker container

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