Stop the (password) insanity !

Juan Peredo

@JuanPeredoTech

https://www.linkedin.com/in/juanperedotech

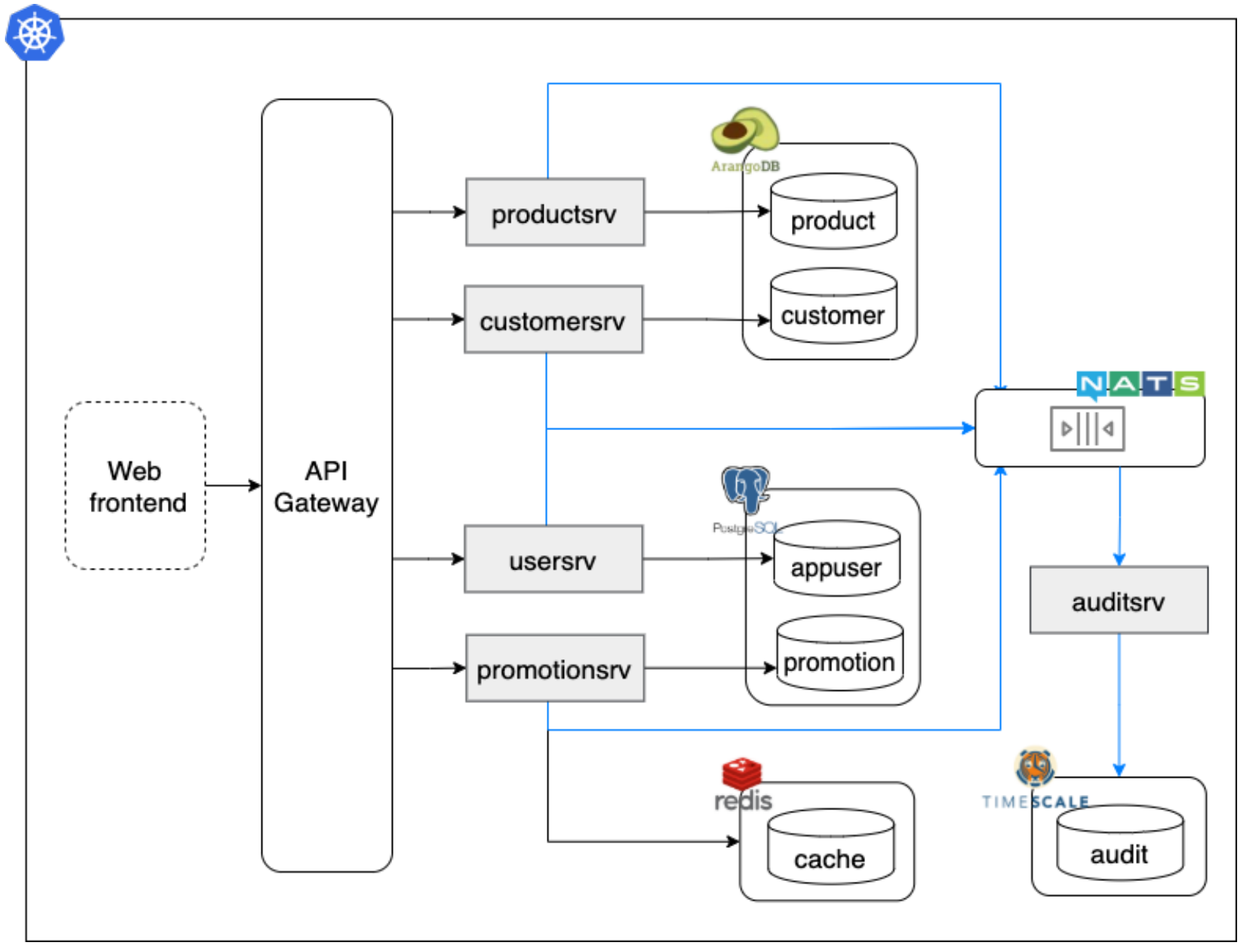
Vault

HashiCorp

A simple assignment

- Jane is the newest member of the team
- Just assigned to rotating the application credentials
- Must be done on a regular basis

A bump on the road



- App uses microservices:
  - 5 micro services
  - 5 databases
  - All services connect to broker (NATS)
  - 1 service uses caching
- **22** passwords & usernames

Doing a little more digging

- Services are managed by different teams
  - User service: User management team
  - Promotion service: Sales team
  - Customer and Product services: Master data team
  - Audit service: Traceability team
- Credentials need to be updated regularly

How Jane is feeling now

Our mission,

should we choose to accept it:

- Stop the (password) insanity by
  - Centralizing credentials management
  - Not changing any of the existing code
  - Bringing back Jane's work-life balance

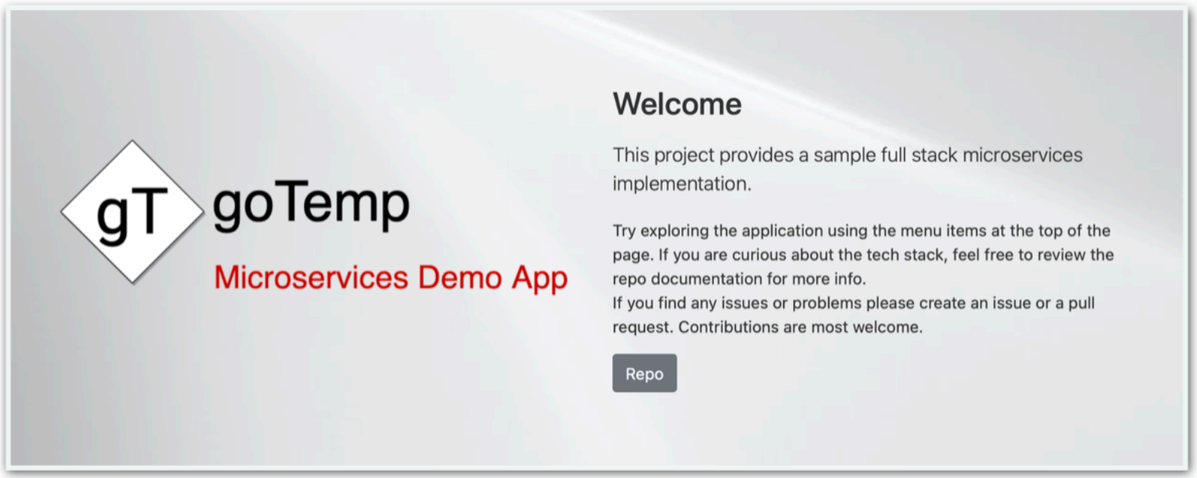
Your guide in this mission

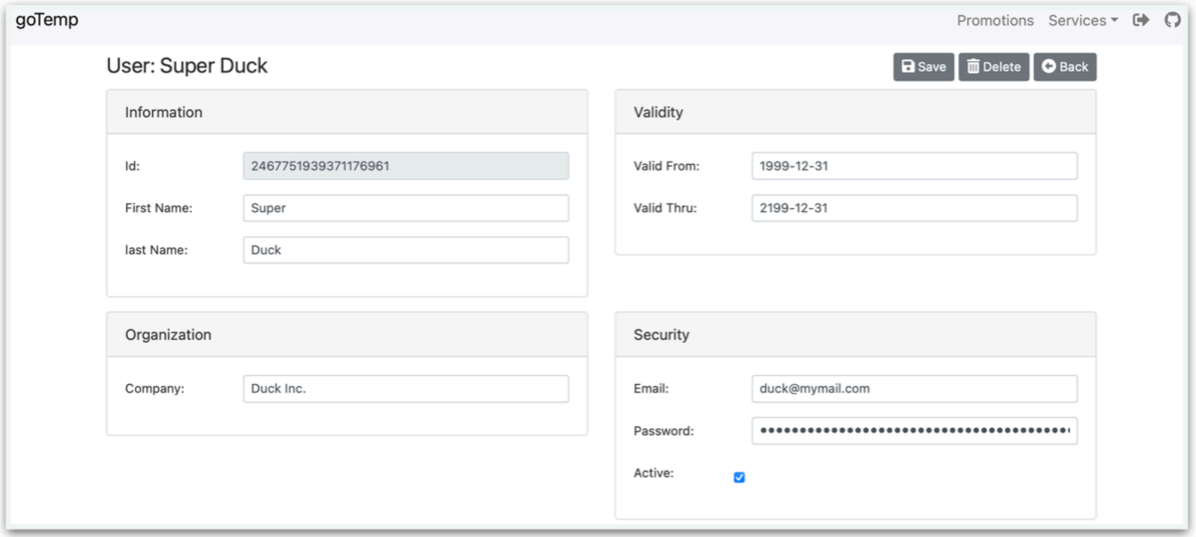
Independent tech architect / developer and everything in between.

Spent his extended 2020 pandemic induced "vacation":

- No going to the movies
- Not eating out
- Writing the app used our demo

Our Application



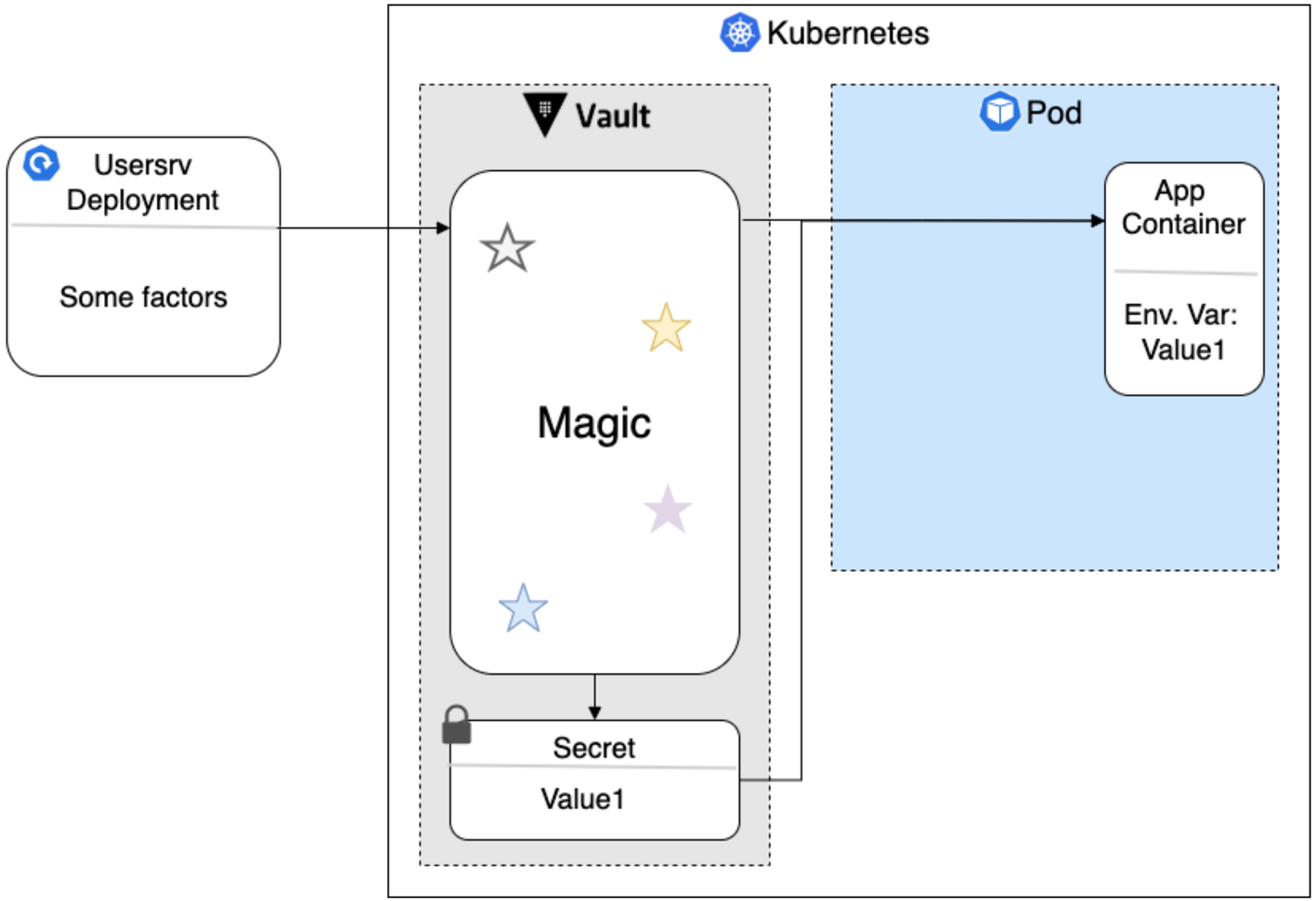


- 5 microservices: user, customer, product, promotion & audit
- Runs on Kubernetes
- Credentials read from environment variables
- We will focus initially on the user service (Usersrv)

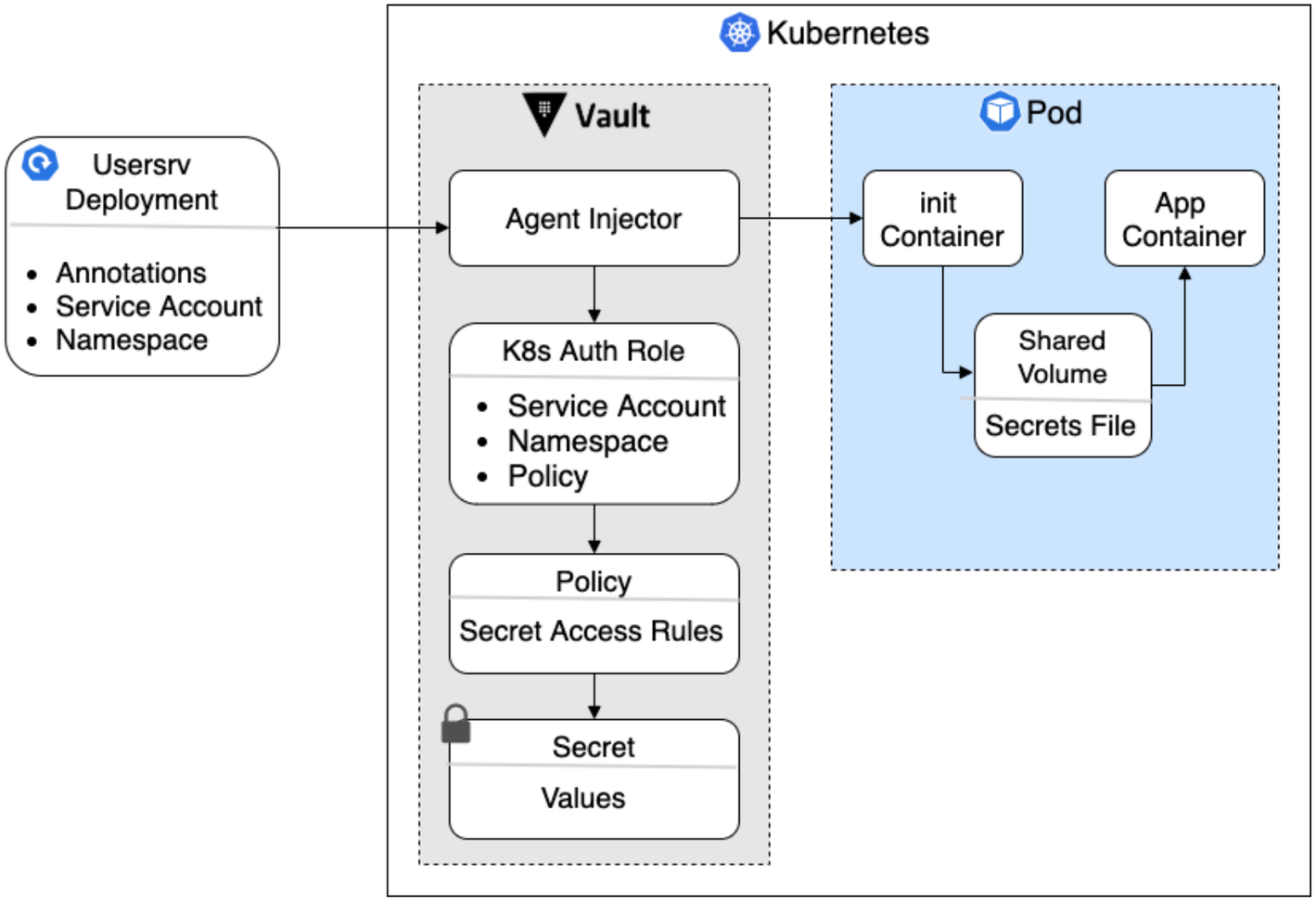
Usersrv Environment Variables

```
MICRO_SERVER_ADDRESS=:50053
POSTGRES_CONNECT=
    postgresql://postgres:Passwd@pgdb/
        appuser?application_name=userSrv
MICRO_BROKER=nats
MICRO_BROKER_ADDRESS=natsUser:Passwd@nats
DISABLE_AUDIT_RECORDS=false
```

The Goal


Kubernetes
Vault
Pod
Usersrv
Deployment
Some factors
Magic
App
Container
Env. Var:
Value1
Secret
Value1

The plan


Kubernetes
Vault
Pod
Usersrv
Deployment
Annotations
Service Account
Namespace
Agent Injector
K8s Auth Role
Service Account
Namespace
Policy
Policy
Secret Access Rules
Secret
Values
init
Container
App
Container
Shared
Volume
Secrets File

Demo

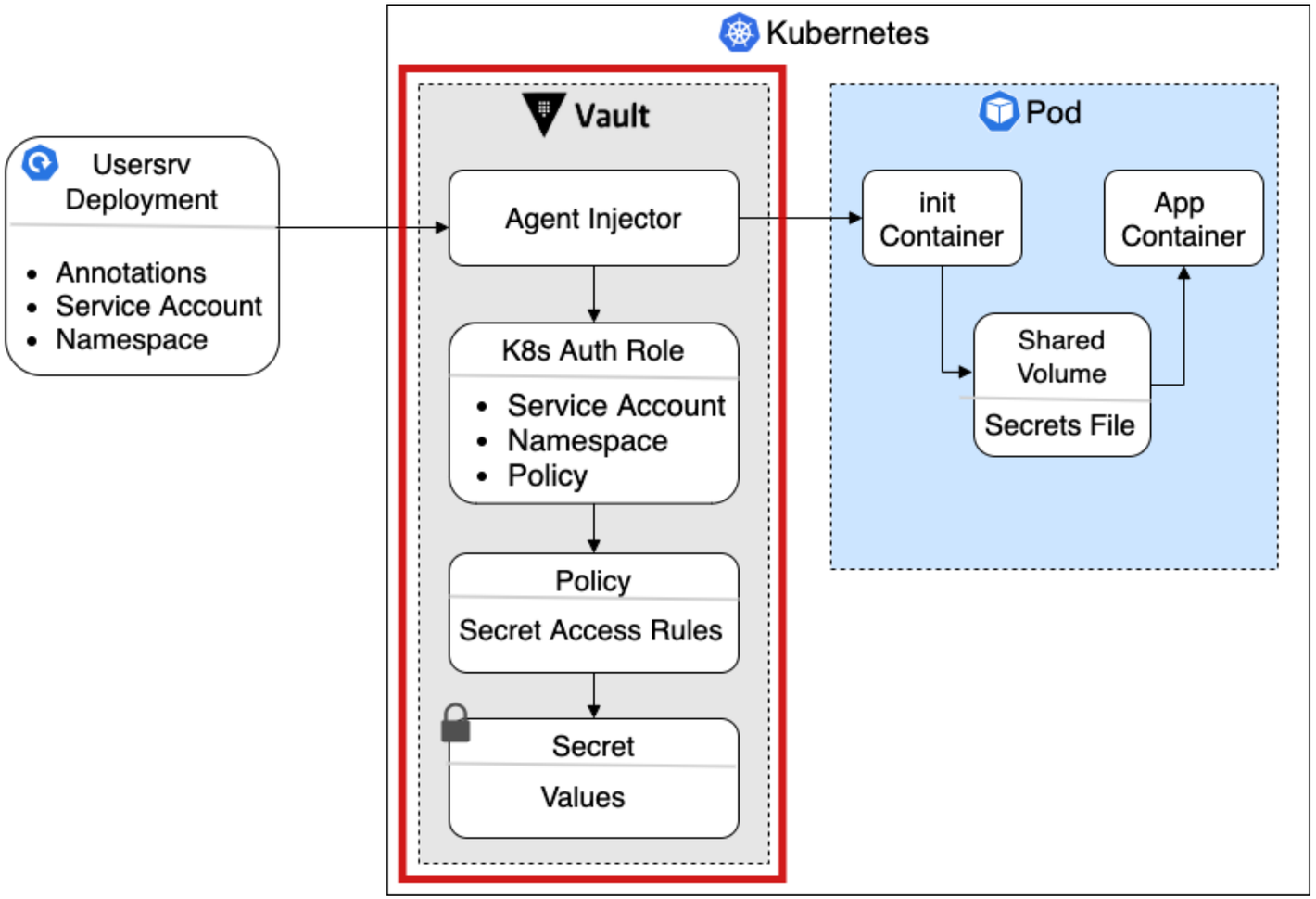
Scripts can be found at:

https://github.com/camba1/gotemp/tree/master/vault

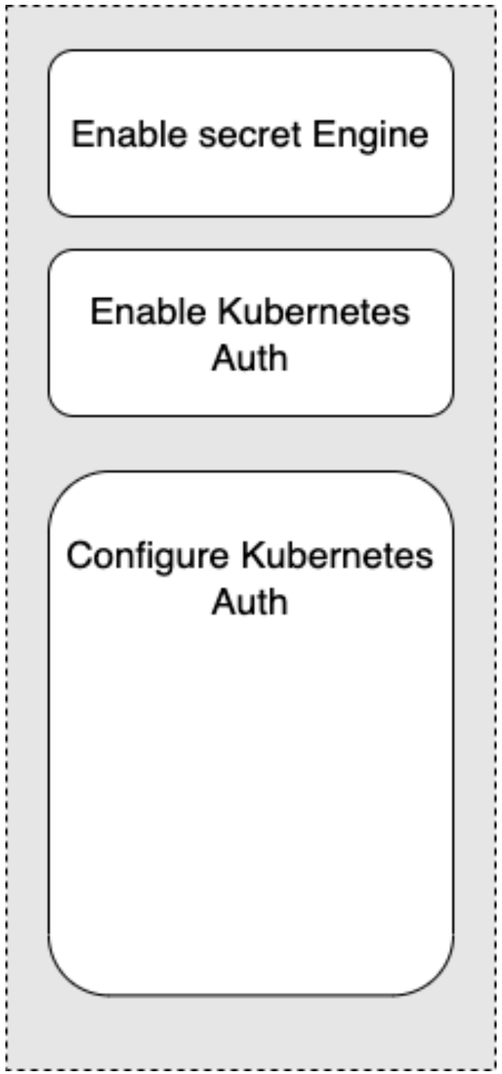
Application found at:

https://github.com/camba1/gotemp

Vault Configuration

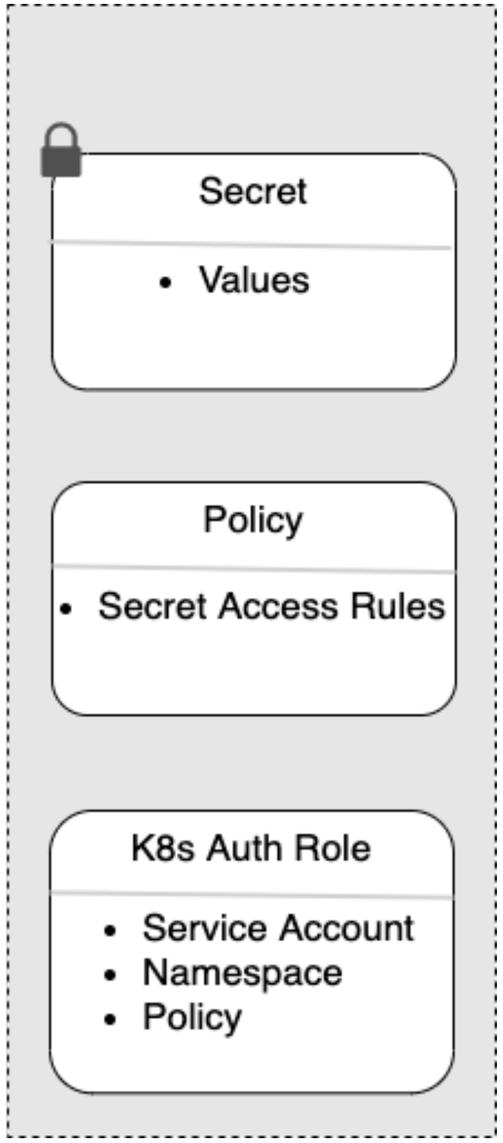


Configure Vault



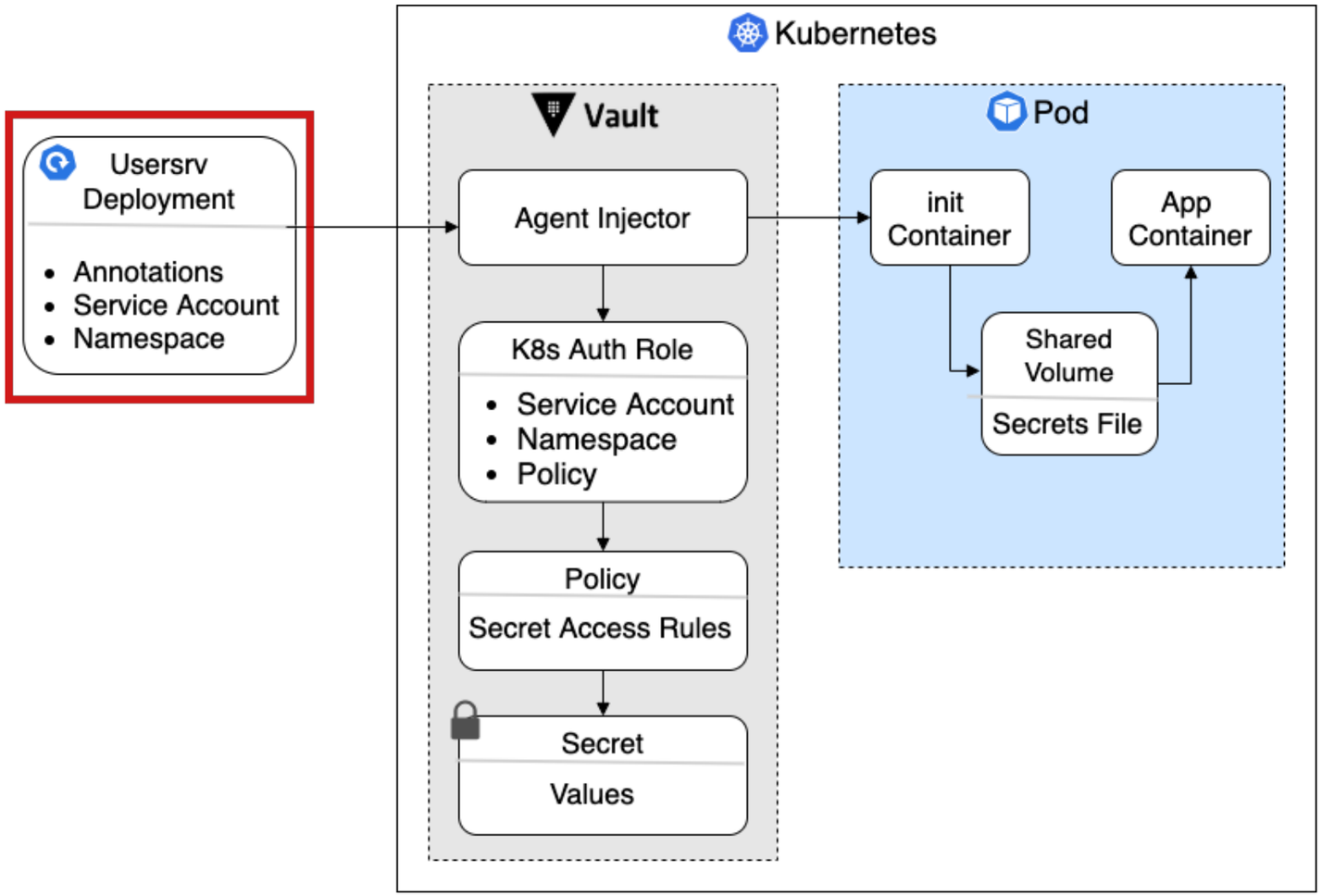

```
vault secrets enable -path=gotempkv kv-v2


vault auth enable kubernetes


vault write auth/kubernetes/config \
  token_reviewer_jwt=
   "$(cat /var/run/secrets/
        kubernetes.io/serviceaccount/token)" \
  kubernetes_host=
   "https://
        $KUBERNETES_PORT_443_TCP_ADDR:443" \
  kubernetes_ca_cert=
    @/var/run/secrets/kubernetes.io/
        serviceaccount/ca.crt
```

Setup Secrets

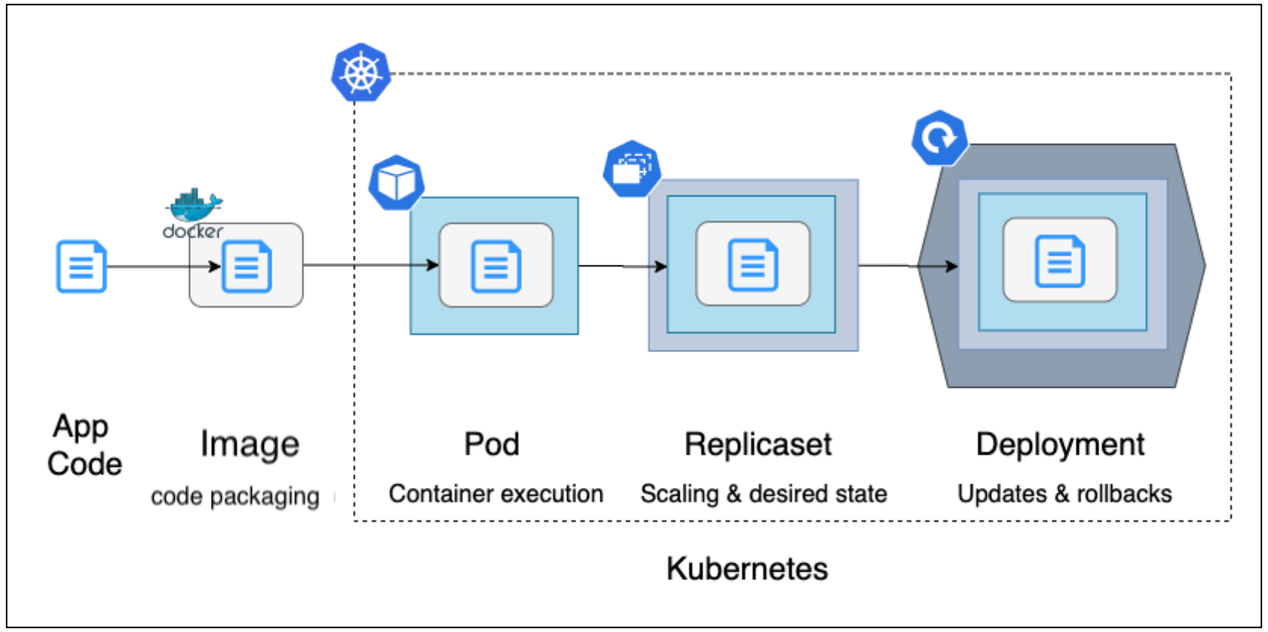



```
vault kv put \
    gotempkv/broker/nats/usersrv \
    username="natsUser" \
    password="Passwd" \
    server="nats"

vault policy write gotemp-usersrv - <<EOF
path "gotempkv/data/broker/nats/usersrv" {
  capabilities = ["read"]
}
EOF

vault write \
    auth/kubernetes/role/gotemp-usersrv \
  bound_service_account_names=gotemp-usersrv \
  bound_service_account_namespaces=default \
  policies=gotemp-usersrv \
  ttl=24h
```

Usersrv Configuration

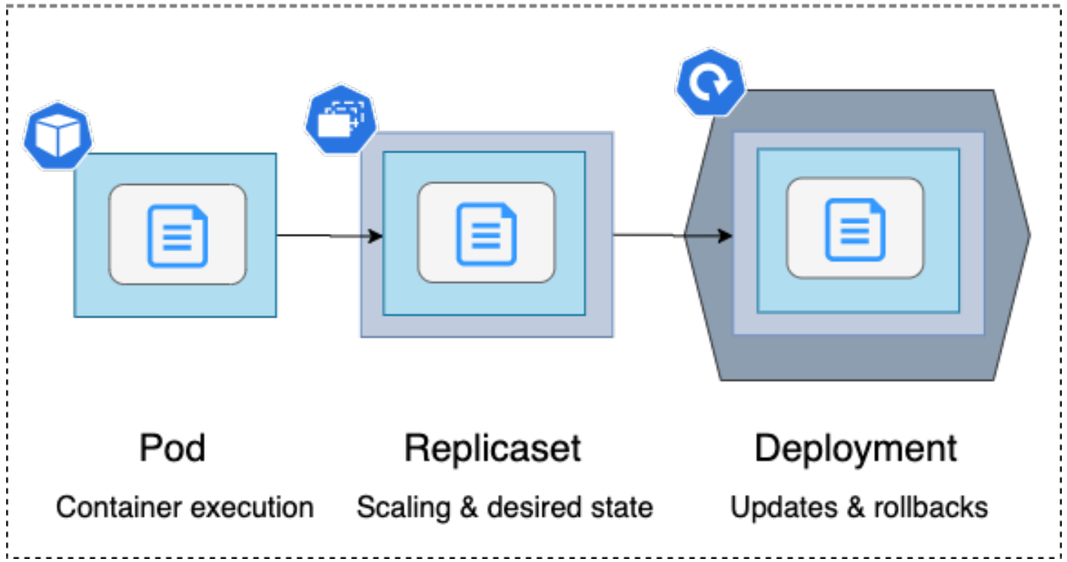


Apps in Kubernetes



- Pods: Smallest deployment unit
- Replicaset: Controls number of pod instances
- Deployment: Where all the magic happens

Usersrv Deployment YAML




```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: usersrv
spec:
  replicas: 1
  template:
    metadata:
      annotations:
        kompose.version: 1.21.0 ()
    spec:
      containers:
      - env:
        - name: MICRO_BROKER_ADDRESS
          valueFrom: ...
        - name: POSTGRES_CONNECT
          valueFrom: ....
        image: bolbeck/gotemp_usersrv
        name: usersrvcont
      serviceAccountName: ""  ...
```

Service Account

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: gotemp-usersrv
```

Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
   {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
```

Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
   {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
```

Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
   {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
```

Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
   {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
```

Pull secret from Vault:

- Using auth role: "gotemp-usersrv"
- From end point: "gotempkv/data/broker/nats/usersrv"
- Place it in the file: /vault/secret/secret-nats.txt

Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
   {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
```

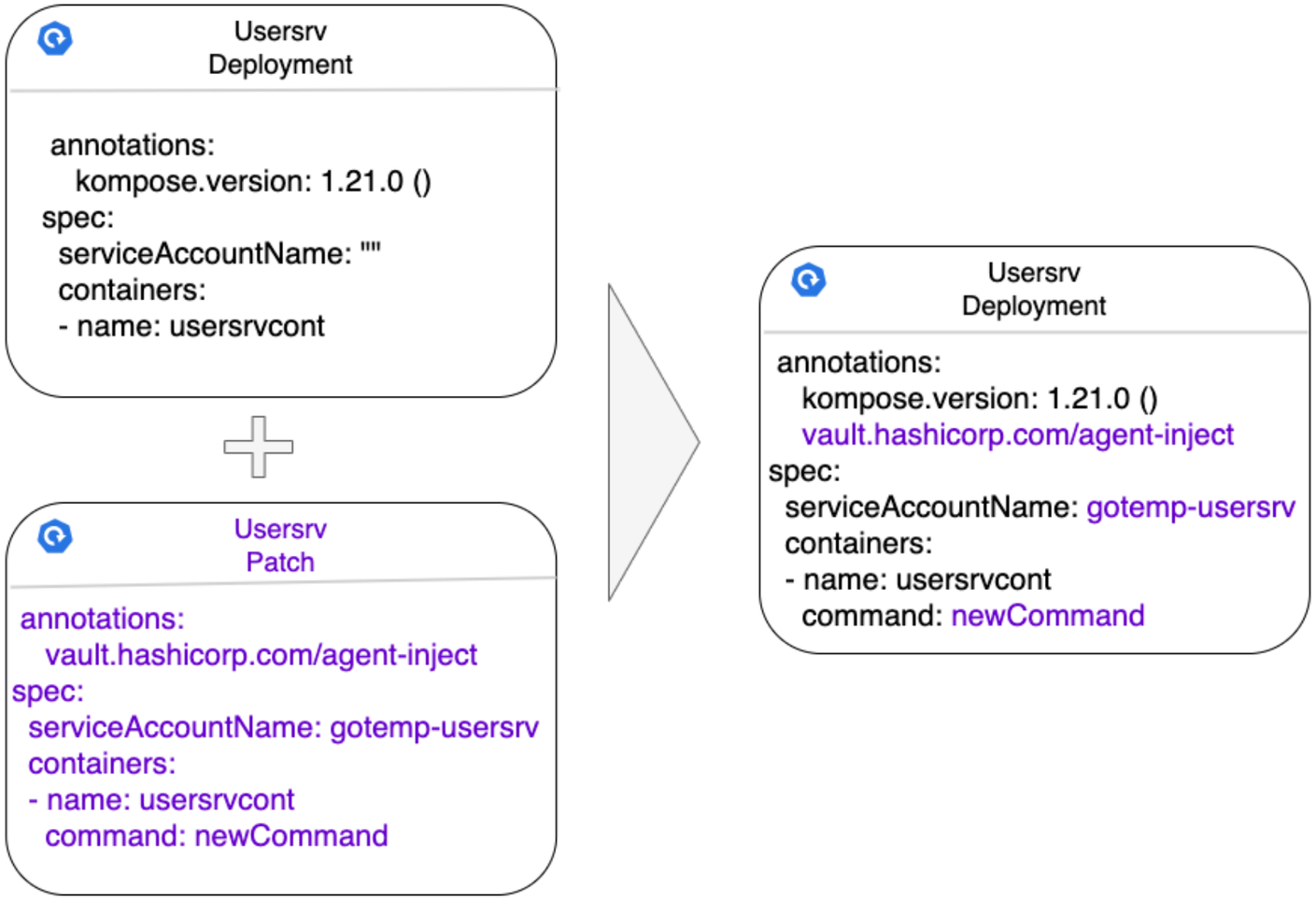
Format secret as :

export MICRO_BROKER_ADDRESS = "username:password@server"

Annotations

```
vault.hashicorp.com/agent-inject: "true"
vault.hashicorp.com/agent-pre-populate-only: "true"
vault.hashicorp.com/role: "gotemp-usersrv"
vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
vault.hashicorp.com/agent-inject-template-nats.txt: |
   {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
        export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
```

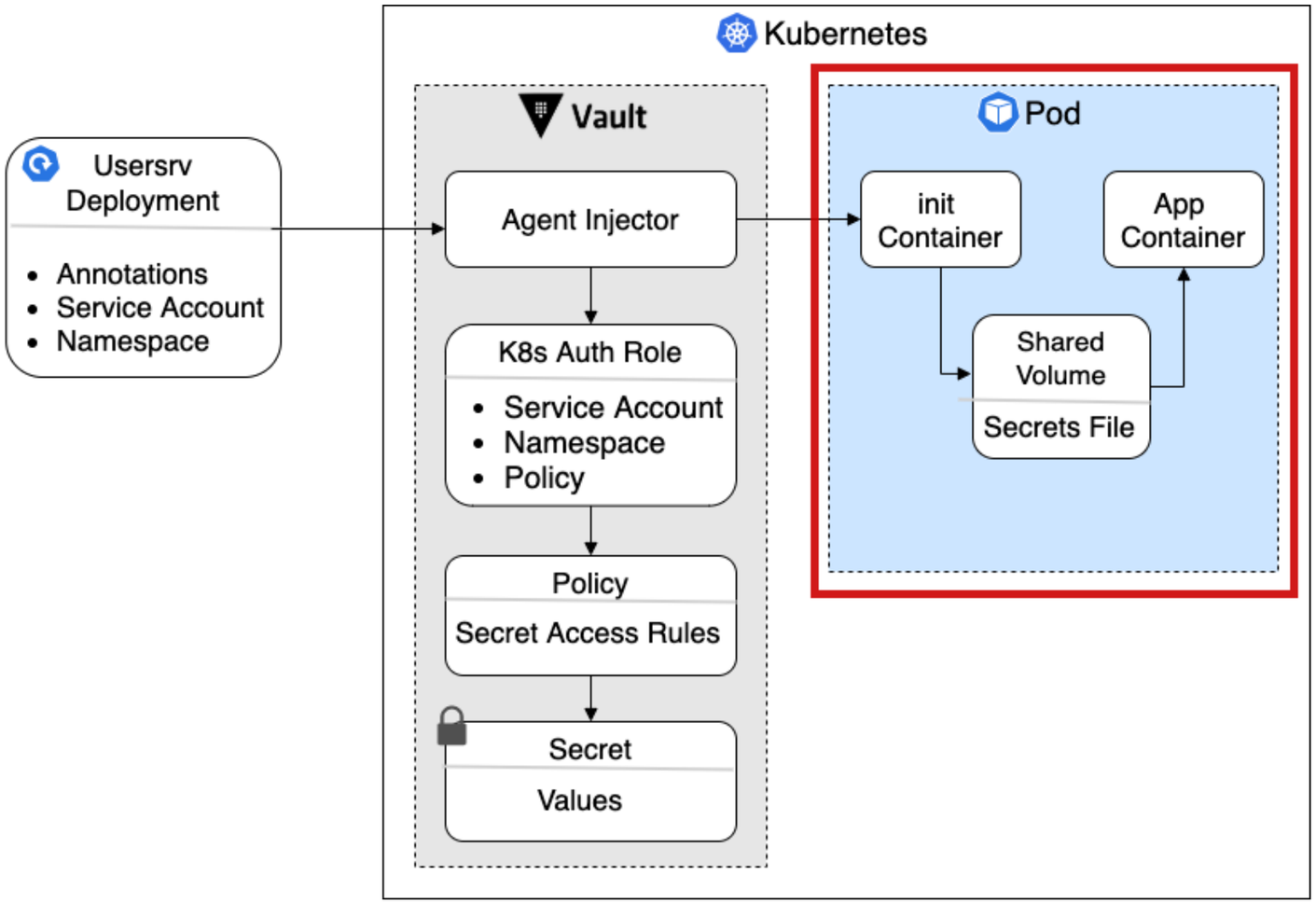
Patch the deployment



The Patch

```
spec:
  template:
    metadata:
      annotations:
        vault.hashicorp.com/agent-inject: "true"
        vault.hashicorp.com/role: "gotemp-usersrv"
        vault.hashicorp.com/agent-pre-populate-only: "true"
        vault.hashicorp.com/agent-inject-secret-nats.txt: "gotempkv/data/broker/nats/usersrv"
        vault.hashicorp.com/agent-inject-template-nats.txt: |
          {{- with secret "gotempkv/data/broker/nats/usersrv" -}}
                export MICRO_BROKER_ADDRESS="{{ .Data.data.username }}:
                        {{ .Data.data.password }}@{{ .Data.data.server }}"
          {{- end -}}
    spec:
      serviceAccountName: gotemp-usersrv
      containers:
      - name: usersrvcont
        command: ['sh', '-c', 'source /vault/secrets/nats.txt && ./userServerAlp' ]
```

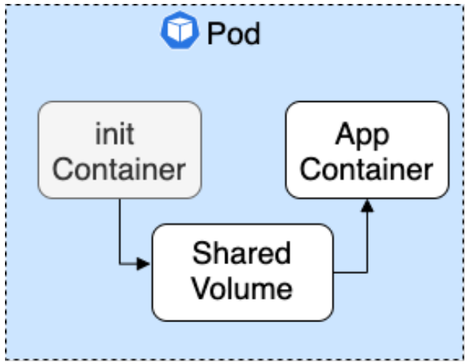
Deploy Changes

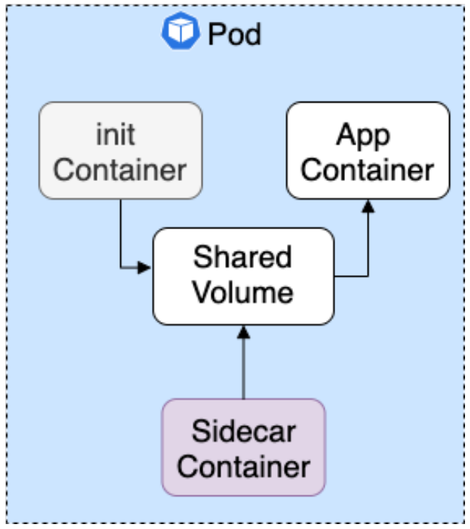


Putting it all together

Action	Result	Usersrv	PostgreSQL
Application Startup	All services and DBs started up	Environment Variables: Username: postgres Pasword: XXXXXXX	DB Credentials: Username: postgres Pasword: XXXXXXX
Change DB credentials	DB credentials changed		DB Credentials: Username: postgres Pasword: YYYYYYY
Patch usersrv deployment	Updated usersrv deployed	Environment Variables: Username: postgres Pasword: YYYYYYY	
	Agent Injector triggered		
	Init Container created in usersrv pod		
	Secrets files created		
	Init container ends		
	Service sources secret files & updates environment variables		

Keeping application in sync





Init container great for set up secrets on startup

```
agent-pre-populate-only: "true"
```

Sidecar container used to update secrets overtime

```
agent-pre-populate-only: "false"
```

Jane after the enhancements

What else could Jane implement ?

- Automating with the CI/CD pipeline
- Integrate patch code into application deployments
- Database credentials management

Key take aways

Questions ?

Thanks to the Hashicorp for organizing this conference!

Appendix

Photos

 Image by Alberto Galvis from Pixabay

 Image by Robin Higgins from Pixabay

 Image by Robin Higgins from Pixabay

 Image by Robin Higgins from Pixabay

 Photo by Campaign Creators on Unsplash

 Image by Mohamed Hassan from Pixabay

Photos

 Photo by AbsolutVision on Unsplash

 Photo by Markus Spiske on Unsplash