Stop the (password) insanity!

Juan Peredo
@JuanPeredoTech
https://www.linkedin.com/in/juanperedotech
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 plan
Scripts can be found at:

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

Application found at:

https://github.com/camba1/gotemp
Vault Configuration

- Usersrv Deployment
  - Annotations
  - Service Account
  - Namespace

- Vault
  - Agent Injector
  - K8s Auth Role
    - Service Account
    - Namespace
    - Policy
  - Policy
  - Secret Access Rules
  - Secret
  - Values

- Kubernetes

- Pod
  - init Container
  - App Container
  - Shared Volume
  - Secrets File
vault secrets enable -path=gotempkv kv-v2

vault auth enable kubernetes

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

- Usersrv Deployment
  - Annotations
  - Service Account
  - Namespace

Vault
- Agent Injector
- K8s Auth Role
  - Service Account
  - Namespace
  - Policy
- Policy
- Secret Access Rules
- Secret
- Values

Pod
- init Container
- Shared Volume
- Secrets File
- App Container
Apps in Kubernetes

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

**Usersrv Deployment YAML**

Pod  
- Container execution

Replicaset  
- Scaling & desired state

Deployment  
- Updates & rollbacks
apiVersion: v1
kind: ServiceAccount
metadata:
  name: gotemp-usersrv
Annotatons

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 --}
Annota&ons

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 -}}
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
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"
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

Before:
```
- annotations:
  - kompose.version: 1.21.0
- spec:
  - serviceAccountName: ""
  - containers:
    - name: usersrvcont
```

After:
```
- annotations:
  - kompose.version: 1.21.0
  - vault.hashicorp.com/agent-inject
- spec:
  - serviceAccountName: gotemp-usersrv
  - containers:
    - name: usersrvcont
      command: newCommand
```
The Patch

```yaml
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

<table>
<thead>
<tr>
<th>Action</th>
<th>Result</th>
<th>Usersrv</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application Startup</strong></td>
<td>All services and DBs started up</td>
<td>Environment Variables: Username: postgres</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password: XXXXXXX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB Credentials:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Username: postgres</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password: XXXXXXX</td>
</tr>
<tr>
<td><strong>Change DB credentials</strong></td>
<td>DB credentials changed</td>
<td>DB Credentials:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Username: postgres</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password: YYYYYYY</td>
</tr>
<tr>
<td><strong>Patch usersrv deployment</strong></td>
<td>Updated usersrv deployed</td>
<td>Environment Variables: Username: postgres</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password: YYYYYYY</td>
</tr>
<tr>
<td></td>
<td>Agent Injector triggered</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Init Container created in usersrv pod</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Secrets files created</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Init container ends</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Service sources secret files &amp; updates</td>
<td></td>
</tr>
<tr>
<td></td>
<td>environment variables</td>
<td></td>
</tr>
</tbody>
</table>
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

- Centralized Credentials Management
- Easy setup
- Less busy, More awesome
- New or Existing Applications
- Scalable solution
- Low barrier to entry
- Scriptable & Automatable
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