kubernetes

Kubernetes Vagrant CI / CD

GitHub Project Repo: kubernetesVagrant

Sample of Work of fully automate build, deploy, validate containers locally. Deploy on a k8s cluster and validations.

September 30, 2021

Outline I

Outline II

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker.
  - Podman.
  - CRI-O.
- Docker Vs Podman.
- What is actually k8s?

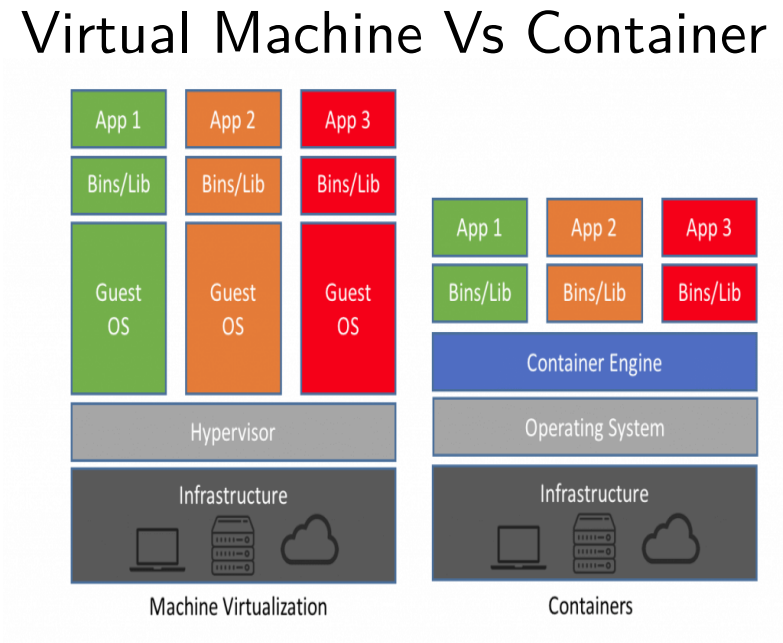


Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

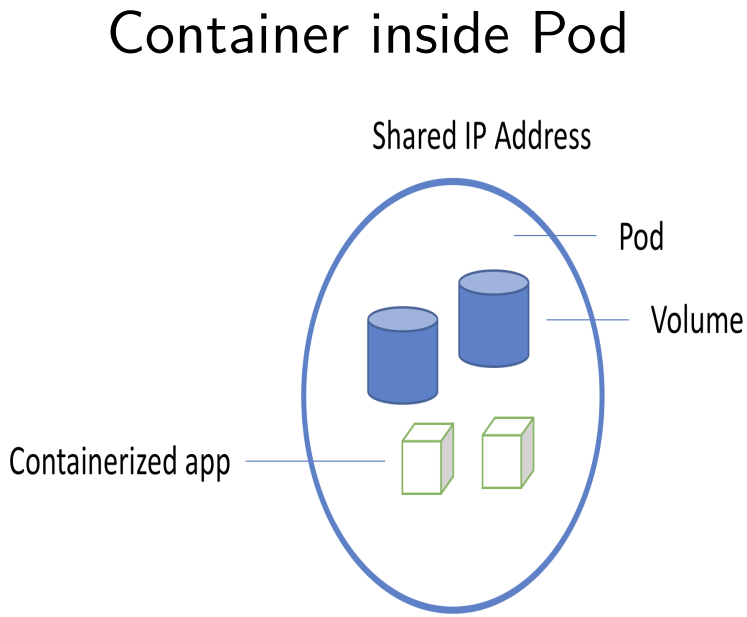


Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

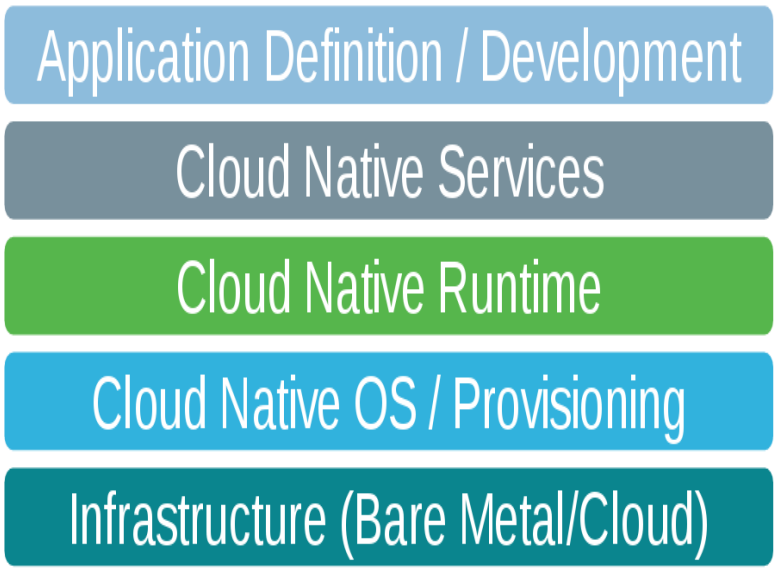


Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

Most known (insecure) socket

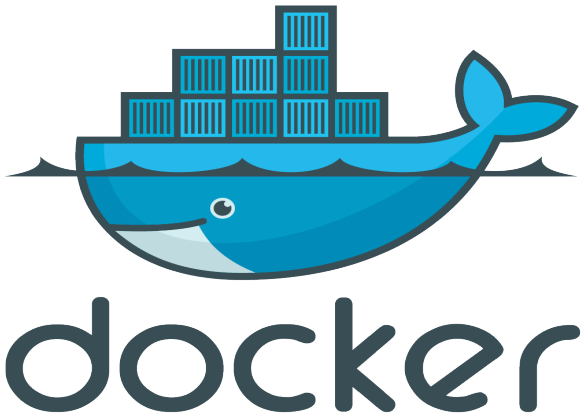


Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

Most unknown (secure) socket

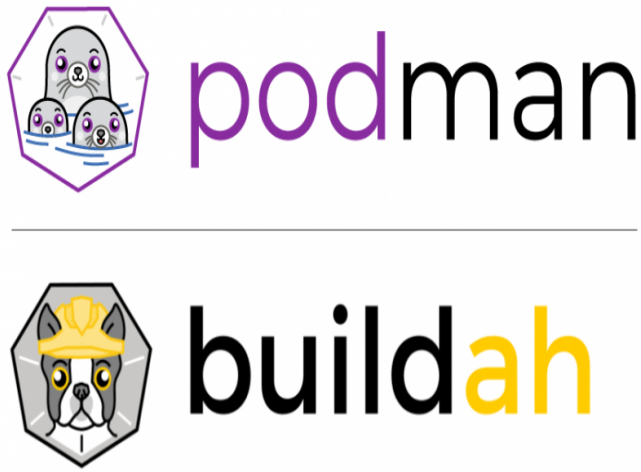


Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

Lightest fastest socket

Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

Read about it

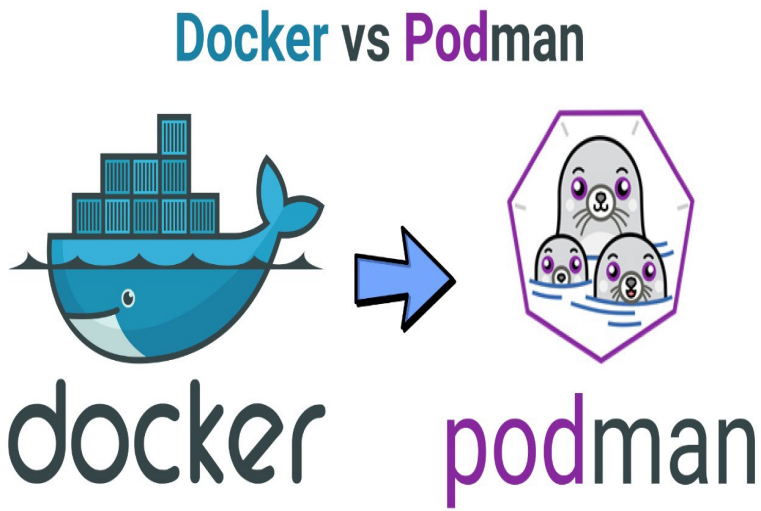


Figure 1: k8s Overview

High level description

- VM Vs Container.
- What is a Pod (pea pod)?
- Container Runtime Interface(s) (CRI).
  - Docker
  - Podman
  - CRI-O
- Docker Vs Podman.
- What is actually k8s?

Is a puzzle of elements

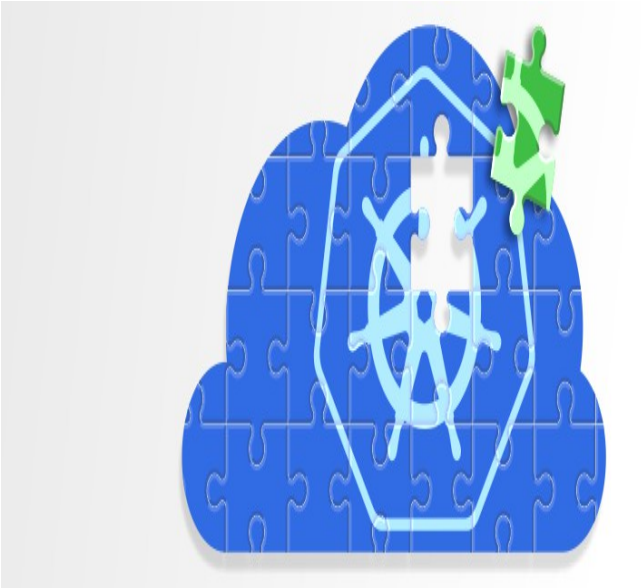


Figure 1: k8s Overview

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Solutions to Requirements

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Solutions to Requirements

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Solutions to Requirements

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Problems / Desires / Solutions on CI / CD

Assessment Requirements

- CI / CD (Locally / Remotely).
- Everything As a Code
- Validation Locally!!!
- OS / Infra. dependencies.

Justification of Requirements

- Works on my PC, why not on Cloud?
- Human error (manual).
- Test (automatically).
- GitHub Actions, Jenkins

Solutions to Requirements

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Solution to problem

Problems

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Solutions

- Containers. Build / deploy / validate locally (controlled env)!
- Fully automated procedure on every step!
- Launch a k8s cluster locally and run all tests locally!
- High Level Programming Language with error handling!

Solution to problem

Problems

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Solutions

- Containers. Build / deploy / validate locally (controlled env)!
- Fully automated procedure on every step!
- Launch a k8s cluster locally and run all tests locally!
- High Level Programming Language with error handling!

Solution to problem

Problems

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Solutions

- Containers. Build / deploy / validate locally (controlled env)!
- Fully automated procedure on every step!
- Launch a k8s cluster locally and run all tests locally!
- High Level Programming Language with error handling!

Solution to problem

Problems

- Solution has to be reproducible locally. Exactly as cloud.
- Minimal human interaction. Auto error handling (Rollback)!
- Powerful PCs! (8 CPUs / 35 GB RAM). Browsing (tabs)?
- No Vendor binding (Azzure DevOps, Jenkins, Bamboo etc).

Solutions

- Containers. Build / deploy / validate locally (controlled env)!
- Fully automated procedure on every step!
- Launch a k8s cluster locally and run all tests locally!
- High Level Programming Language with error handling!

Ansible

Possible Questions

- Why Ansible?
- Are there any benefits of this tool?
- Ansible works on ssh how it will work locally?
- How it can interact with Containers, k8s, Cloud, tests?

Possible Answers

- Written in Python 2/3. Developed and maintained by RedHat.
- Woks perfectly without extra configurations on all OS.
- It can be configured to run on localhost without ssh session.
- It has infinite amount of packages for OS, Containers, Cloud.

Ansible

Possible Questions

- Why Ansible?
- Are there any benefits of this tool?
- Ansible works on ssh how it will work locally?
- How it can interact with Containers, k8s, Cloud, tests?

Possible Answers

- Written in Python 2/3. Developed and maintained by RedHat.
- Woks perfectly without extra configurations on all OS.
- It can be configured to run on localhost without ssh session.
- It has infinite amount of packages for OS, Containers, Cloud.

Ansible

Possible Questions

- Why Ansible?
- Are there any benefits of this tool?
- Ansible works on ssh how it will work locally?
- How it can interact with Containers, k8s, Cloud, tests?

Possible Answers

- Written in Python 2/3. Developed and maintained by RedHat.
- Woks perfectly without extra configurations on all OS.
- It can be configured to run on localhost without ssh session.
- It has infinite amount of packages for OS, Containers, Cloud.

Ansible

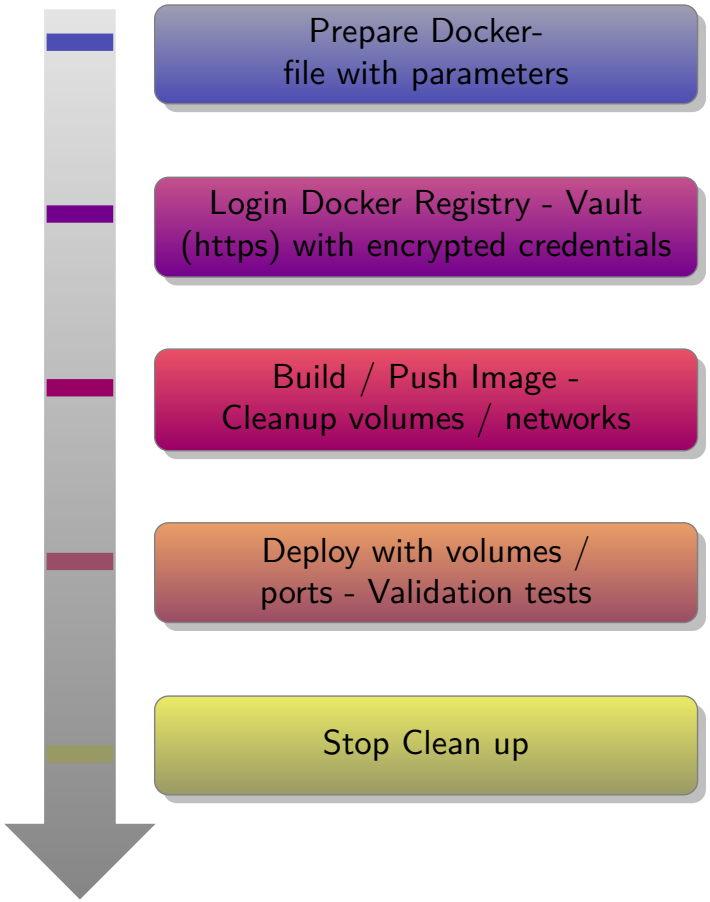
Possible Questions

- Why Ansible?
- Are there any benefits of this tool?
- Ansible works on ssh how it will work locally?
- How it can interact with Containers, k8s, Cloud, tests?

Possible Answers

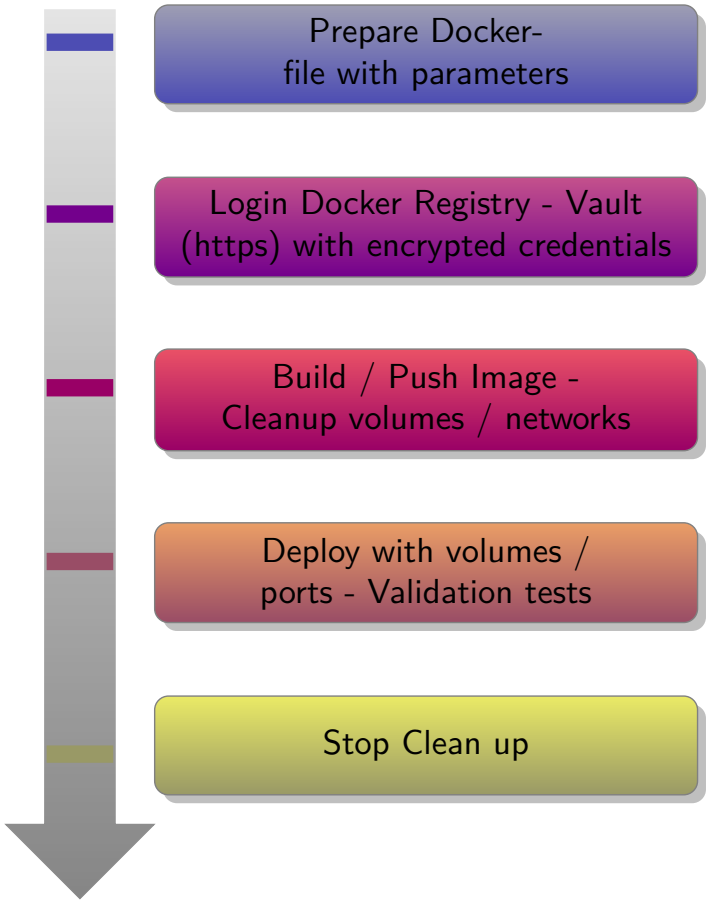
- Written in Python 2/3. Developed and maintained by RedHat.
- Woks perfectly without extra configurations on all OS.
- It can be configured to run on localhost without ssh session.
- It has infinite amount of packages for OS, Containers, Cloud.

Local Procedure



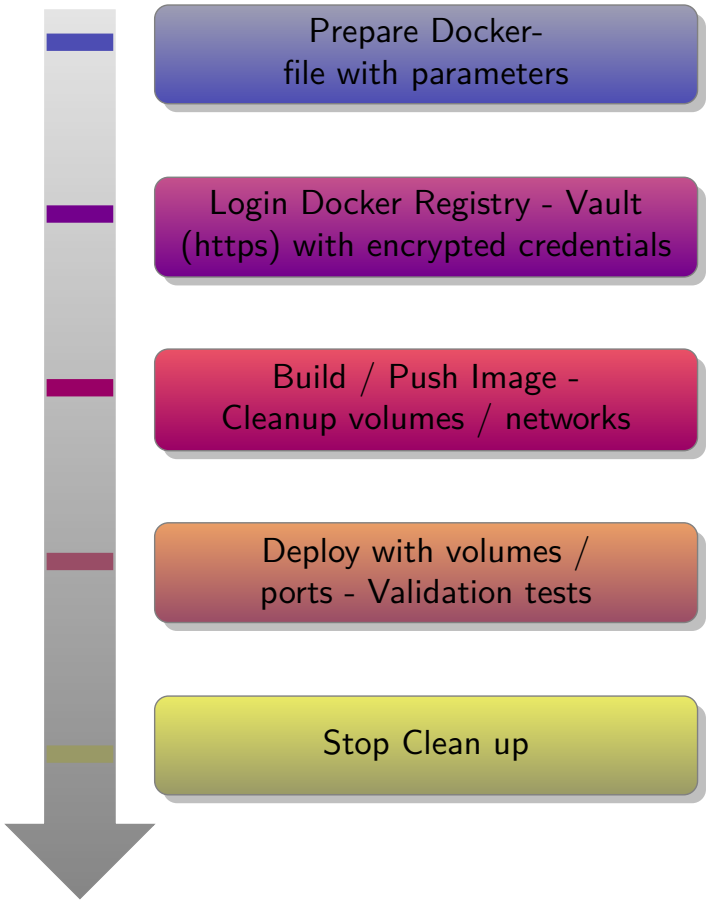
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azure Registry.
  - Build Dockerfile.
  - Push Image.
  - Login Azure.
  - Pull Image.
  - Run container (id).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



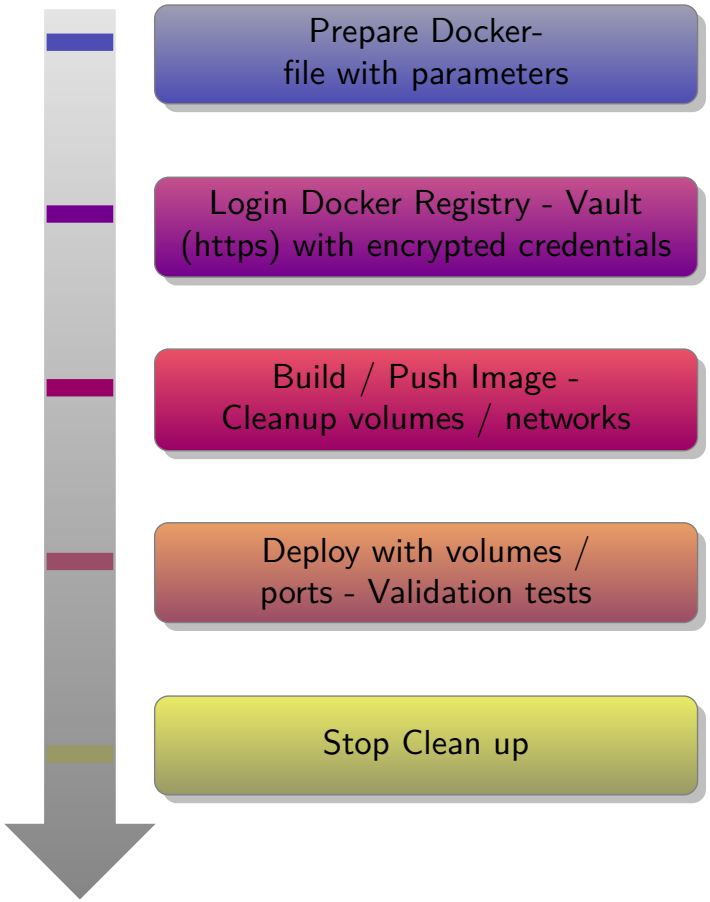
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Login Azzure.
  - Pull Image.
  - Run container.
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



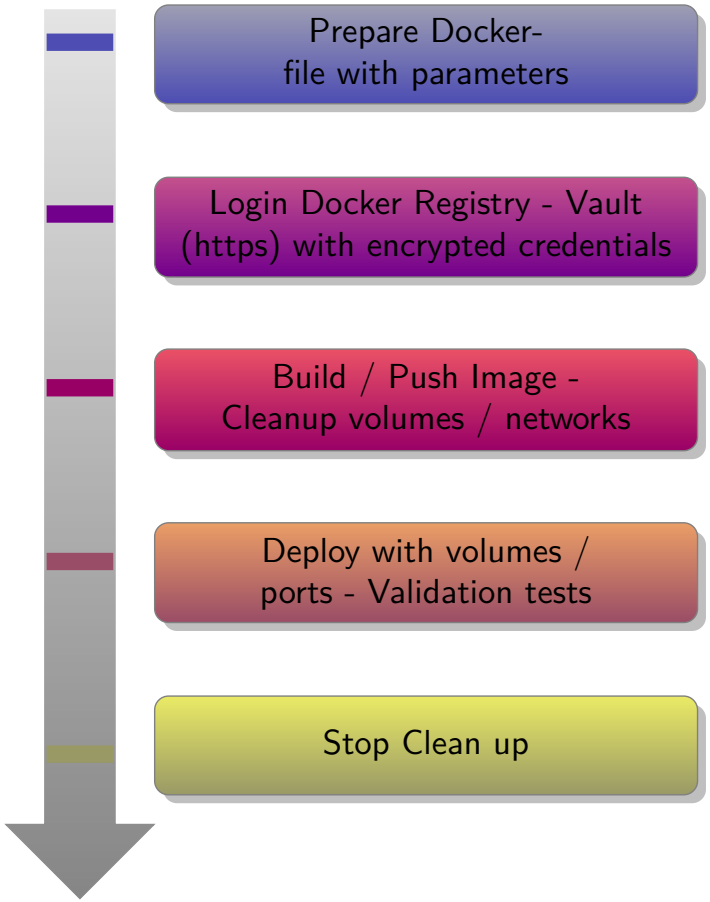
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



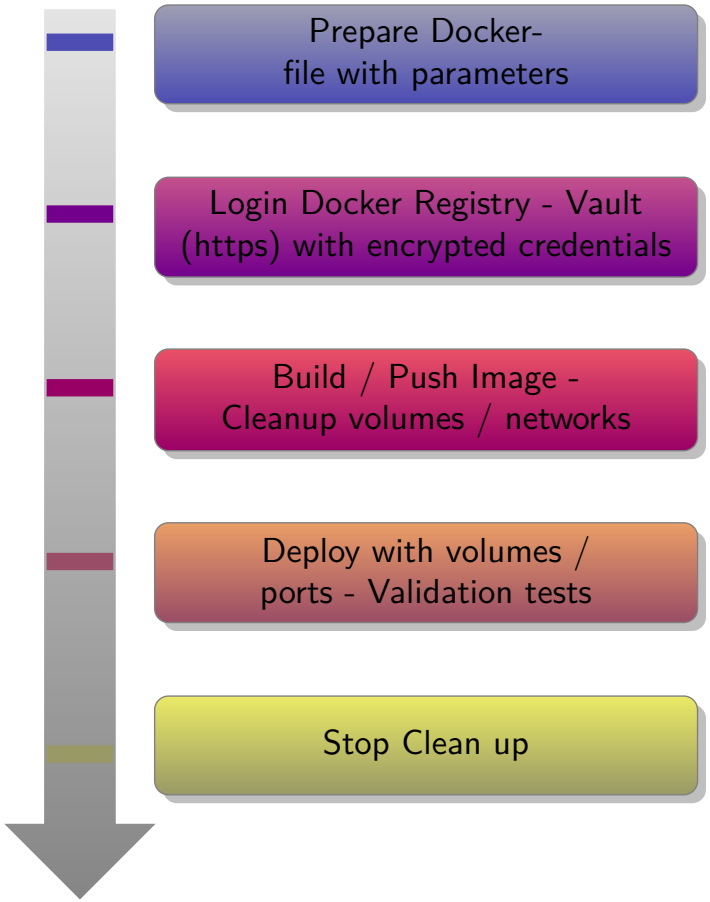
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
    - Build Dockerfile.
    - Push Image.
    - Logout Azzure.
    - Prune everything.
    - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



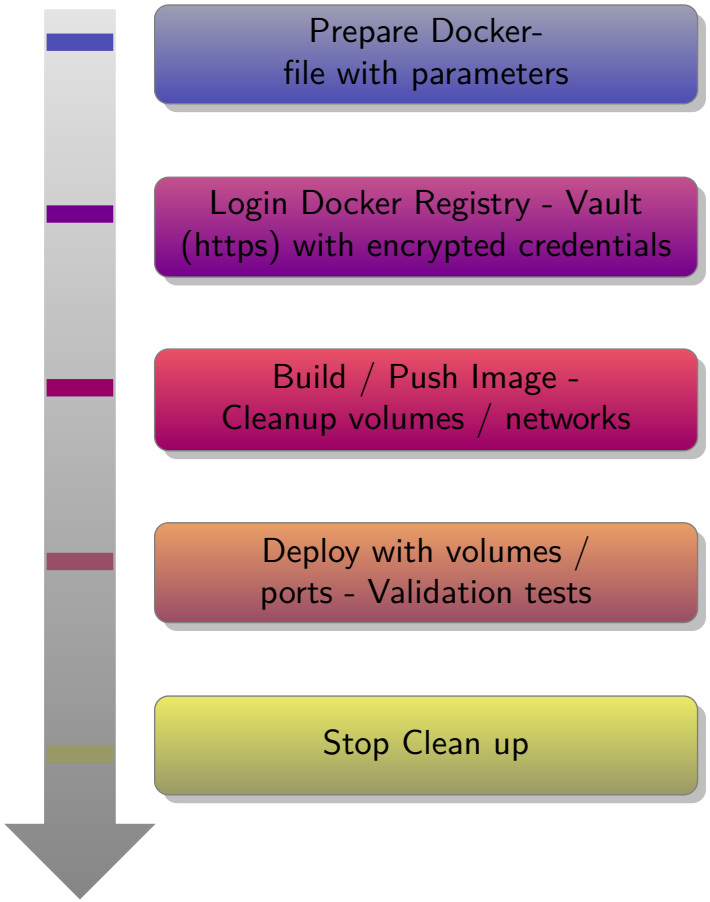
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



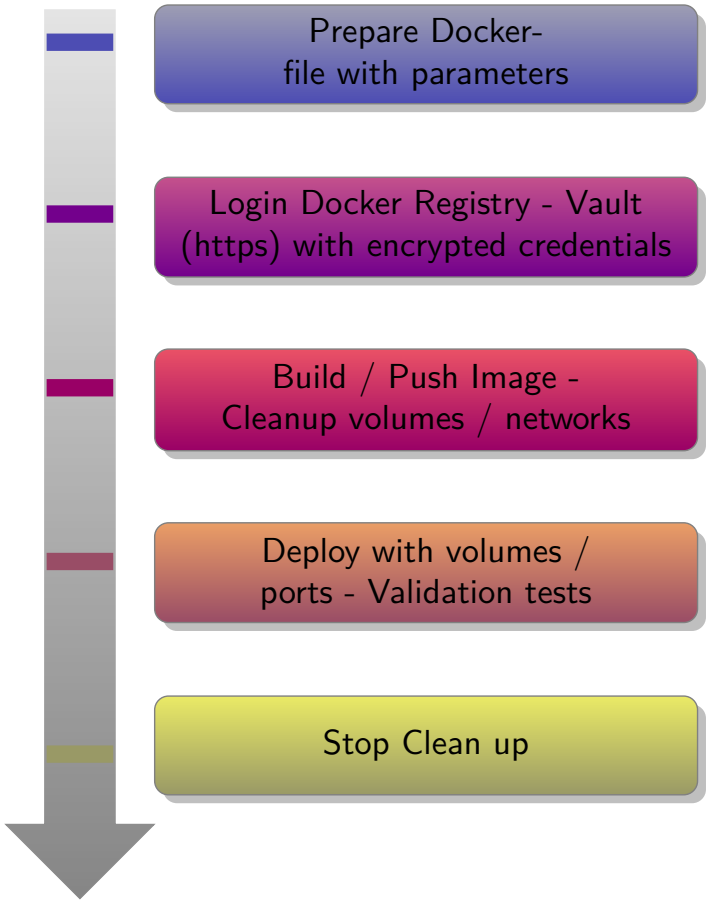
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



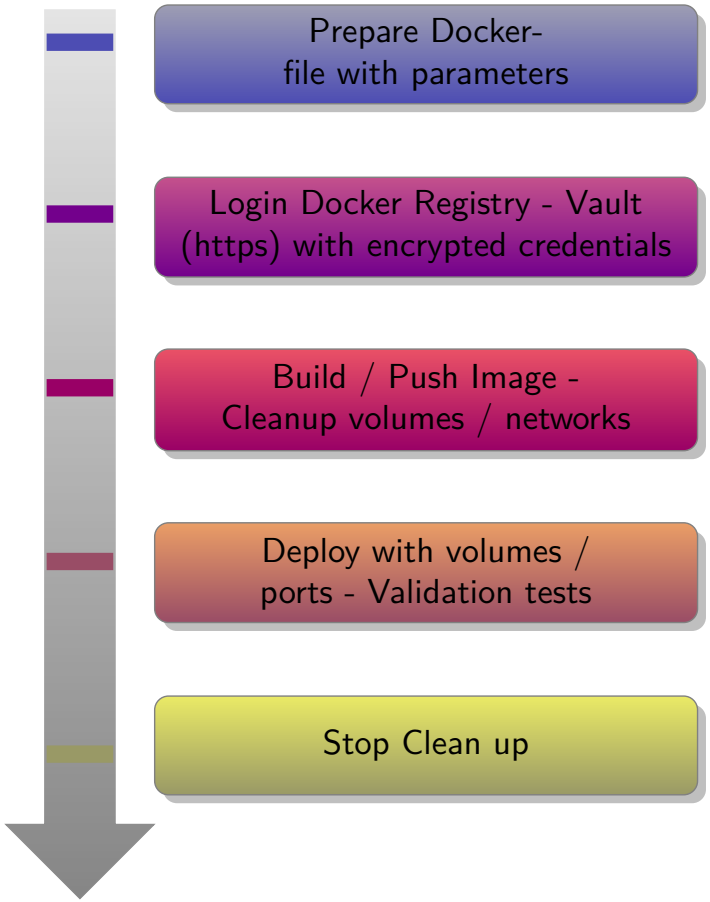
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



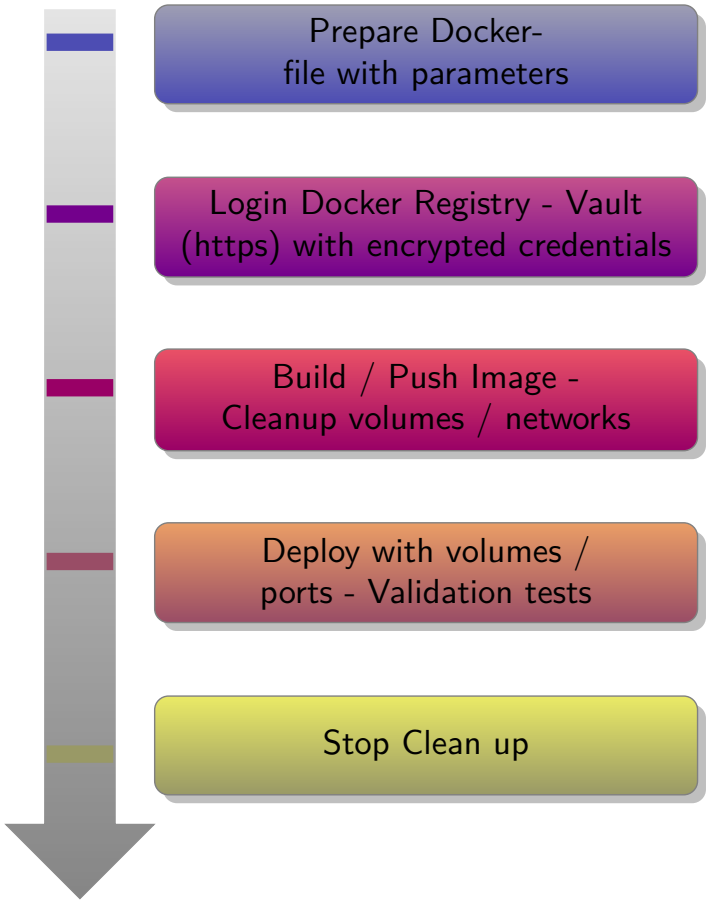
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



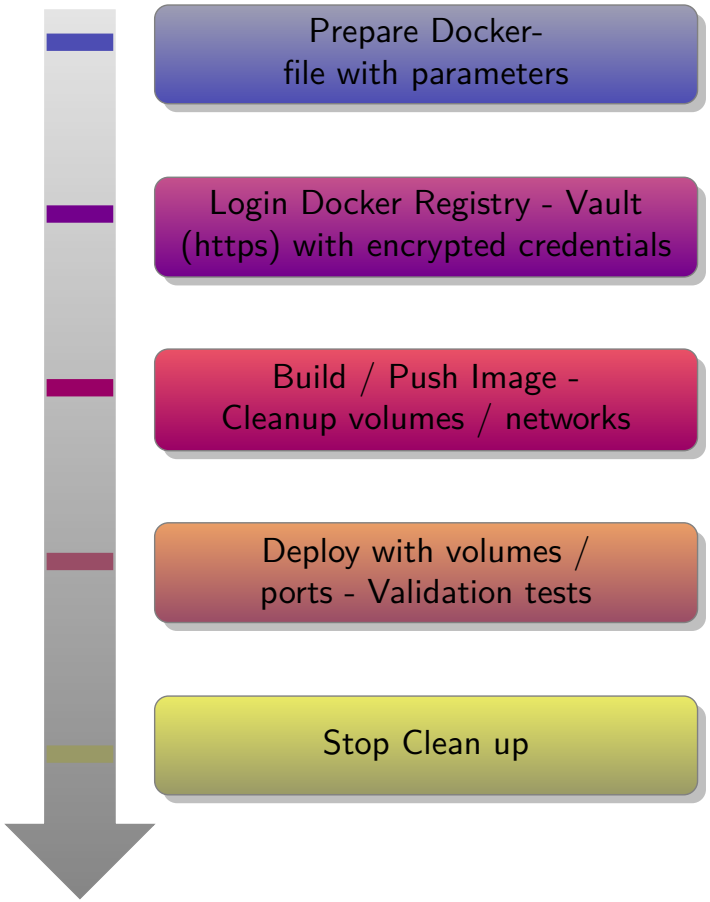
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



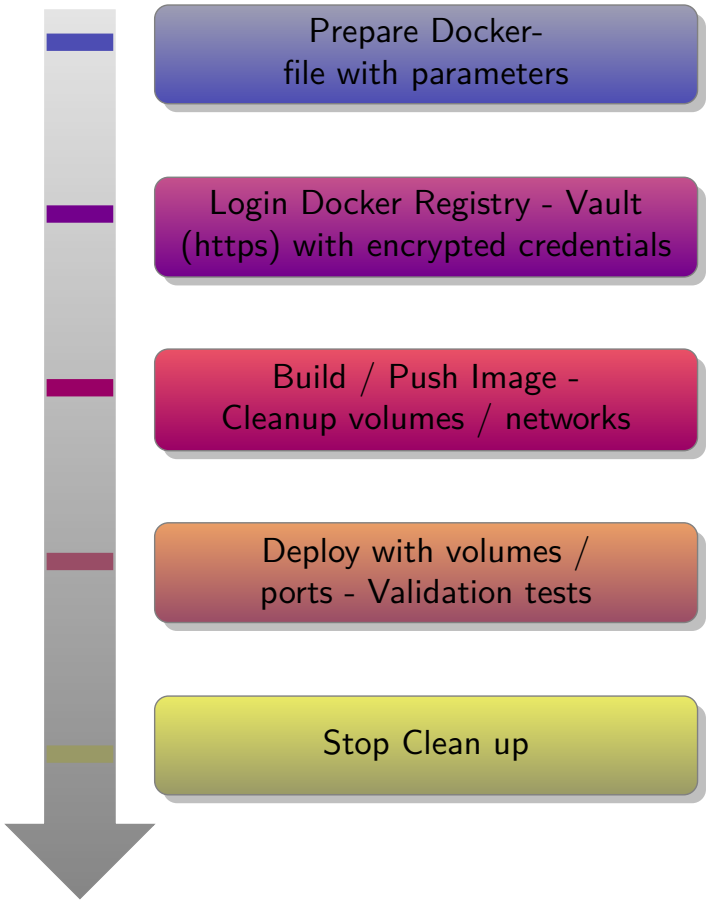
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



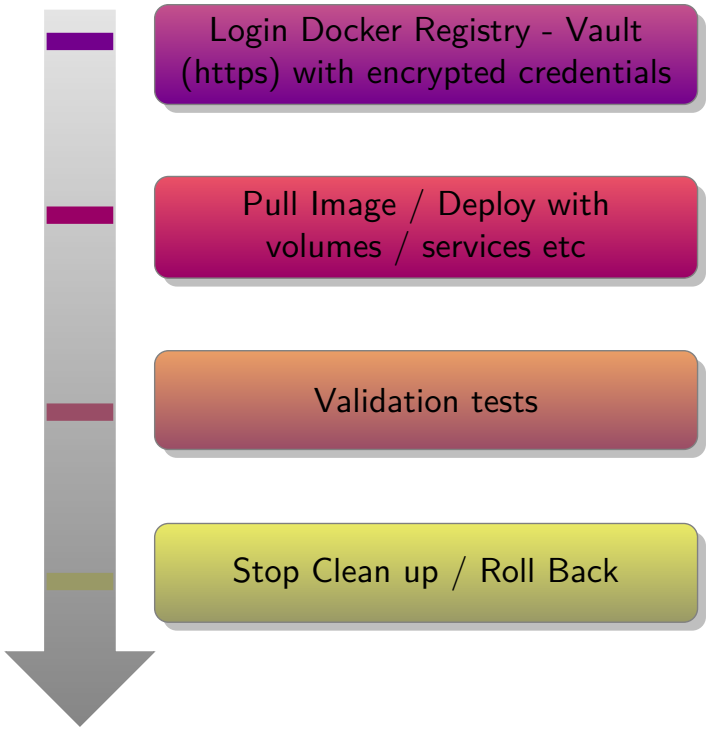
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local Procedure



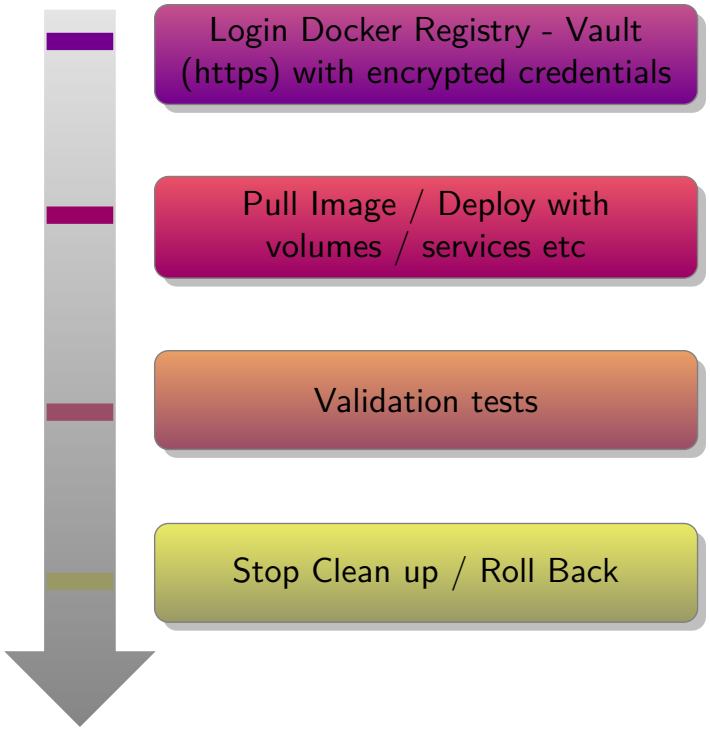
- Dockerfile (template).
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Build Dockerfile.
  - Push Image.
  - Logout Azzure.
  - Prune everything.
  - Raise error (if).
- Deployment (volume).
- Validation (tests).
- Stop, Cleanup.

Local k8s Deployment Procedure



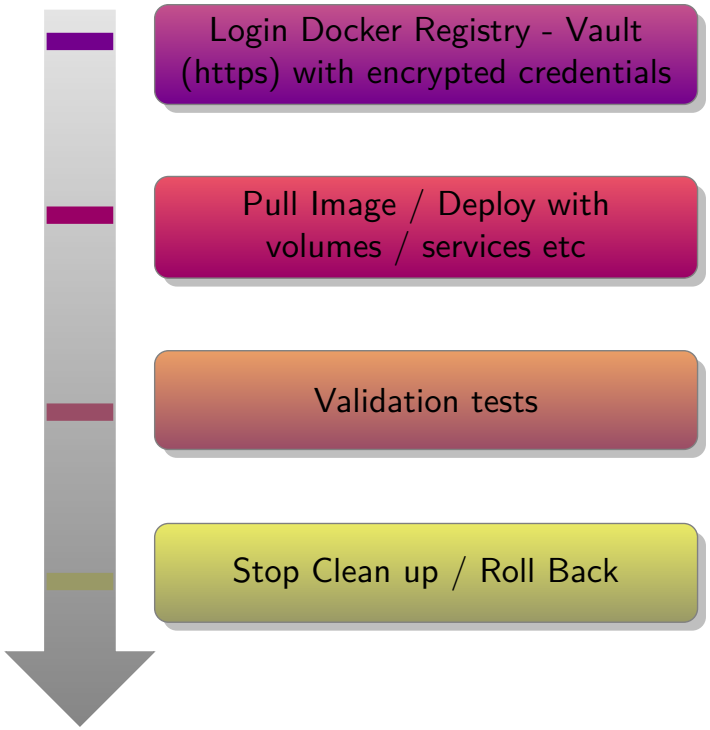
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Bitnami.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
  - Selenium (ui).
  - Sonar (code).
  - Kerberos.

Local k8s Deployment Procedure



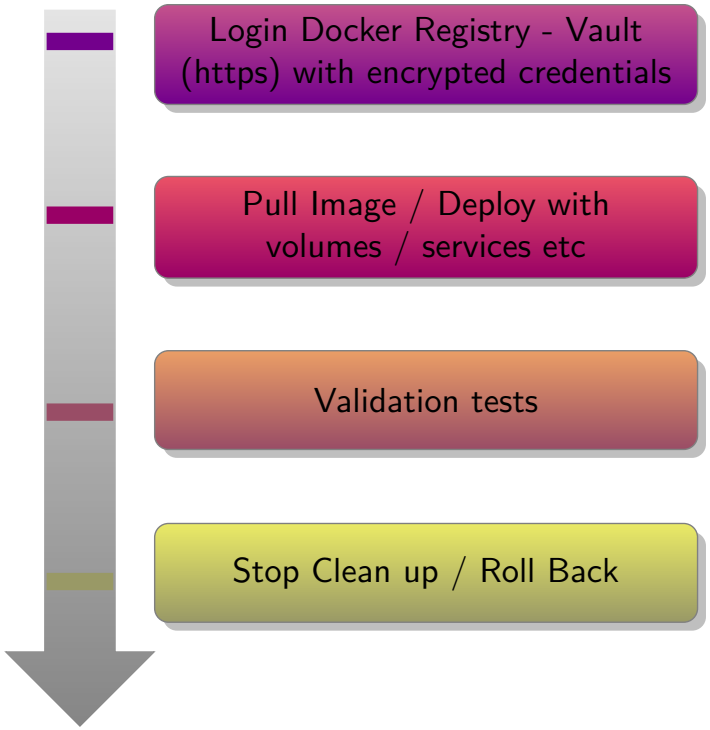
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
- Rollback (if).
- Stop / Remove.
- Notifications.

Local k8s Deployment Procedure



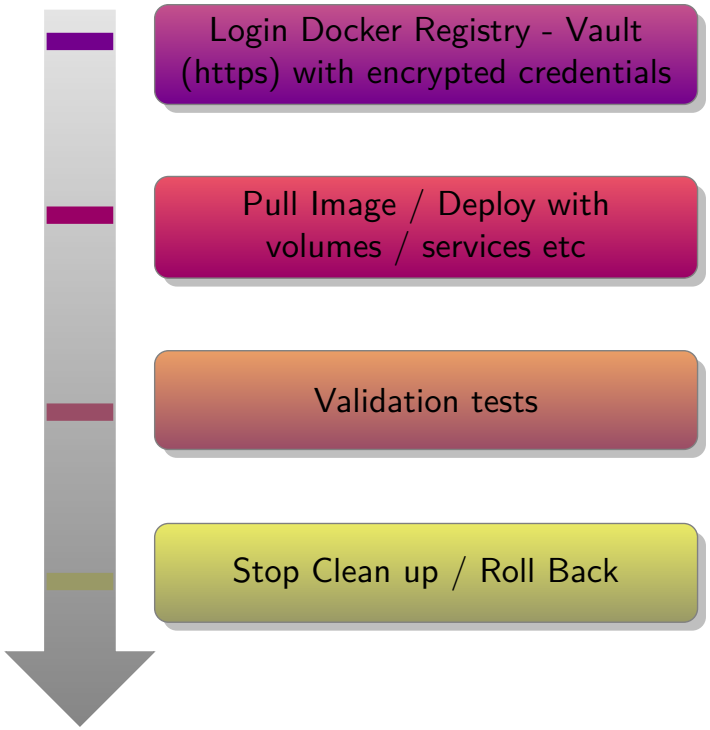
- Vault (https).
- Any socket.
  - Azzure Registry.
    - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
- Rollback (if).
- Stop / Remove.
- Roll back.

Local k8s Deployment Procedure



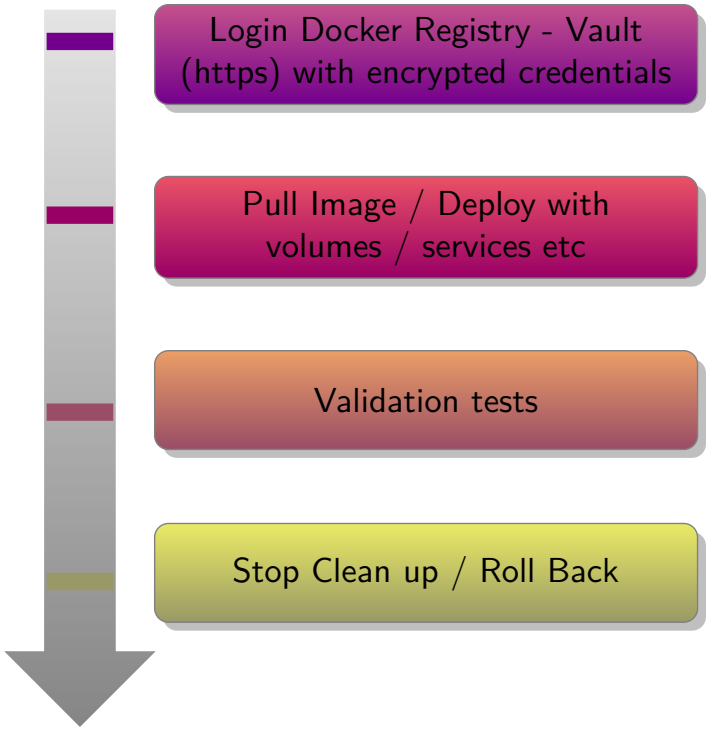
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
- Rollback (if).
- Stop Clean up.
- Roll back.

Local k8s Deployment Procedure



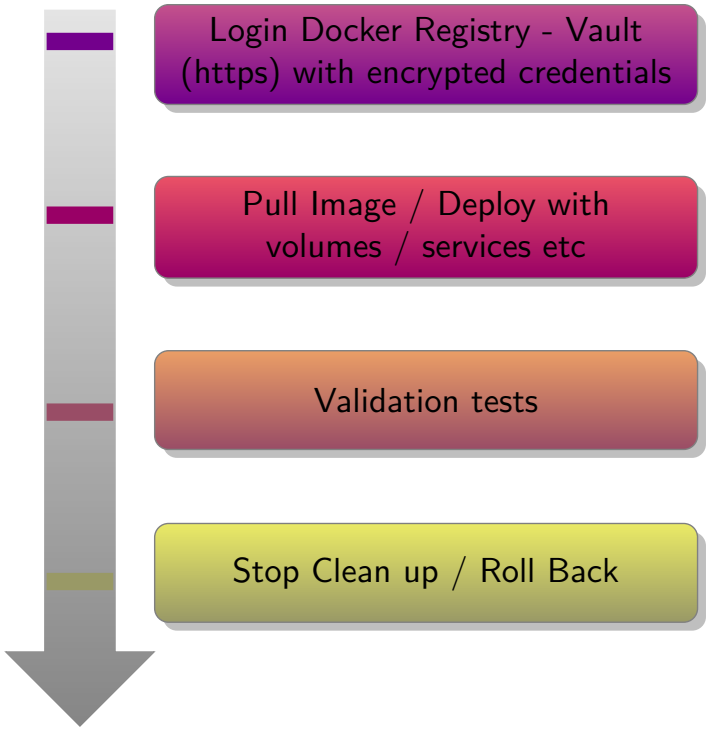
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
- Rollback.

Local k8s Deployment Procedure



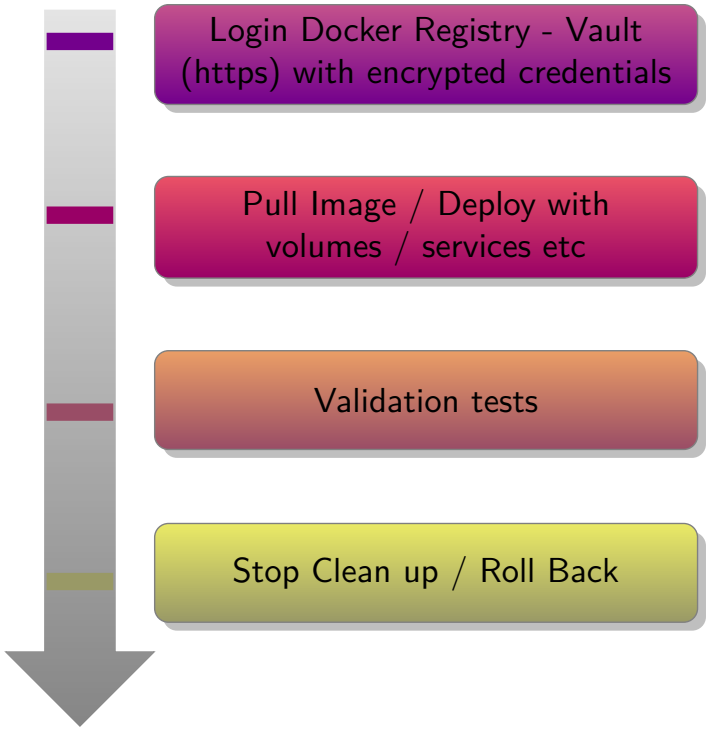
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
  - Raise error (if).
  - Stop / Clean up.
  - Roll back.

Local k8s Deployment Procedure



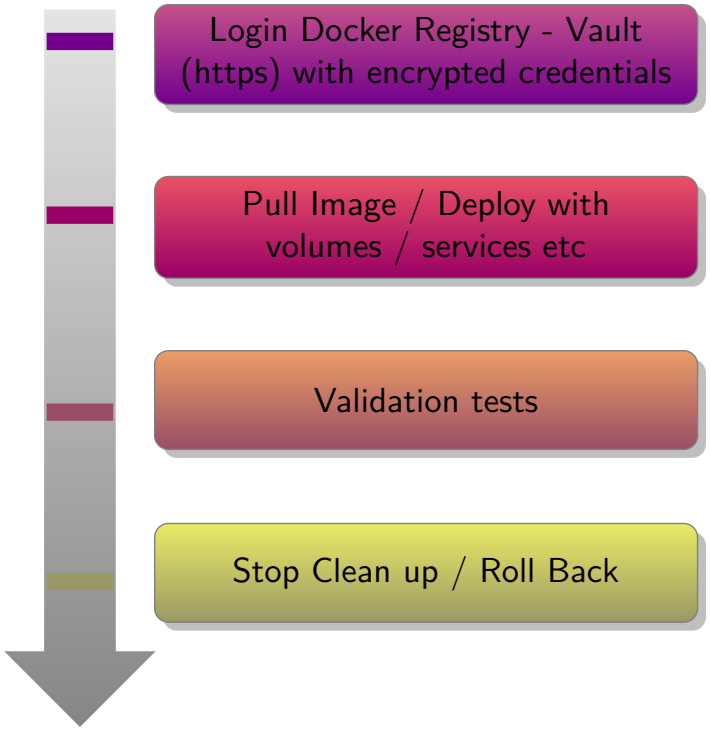
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
  - Raise error (if).
  - Stop, Cleanup.
  - Roll back.

Local k8s Deployment Procedure



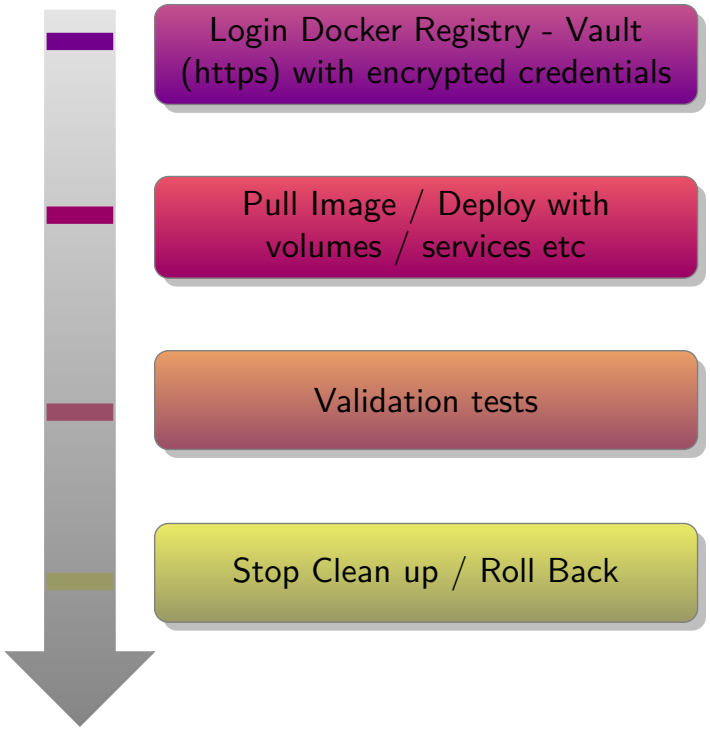
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
  - Raise error (if).
  - Stop, Cleanup.
  - Roll back.

Local k8s Deployment Procedure



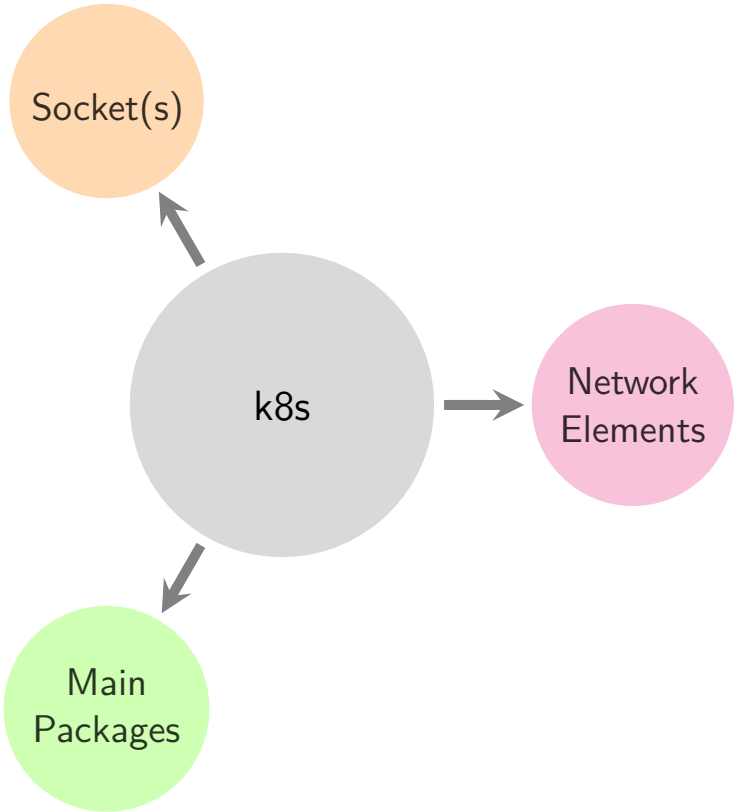
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
  - Raise error (if).
  - Stop, Cleanup.
  - Roll back.

Local k8s Deployment Procedure



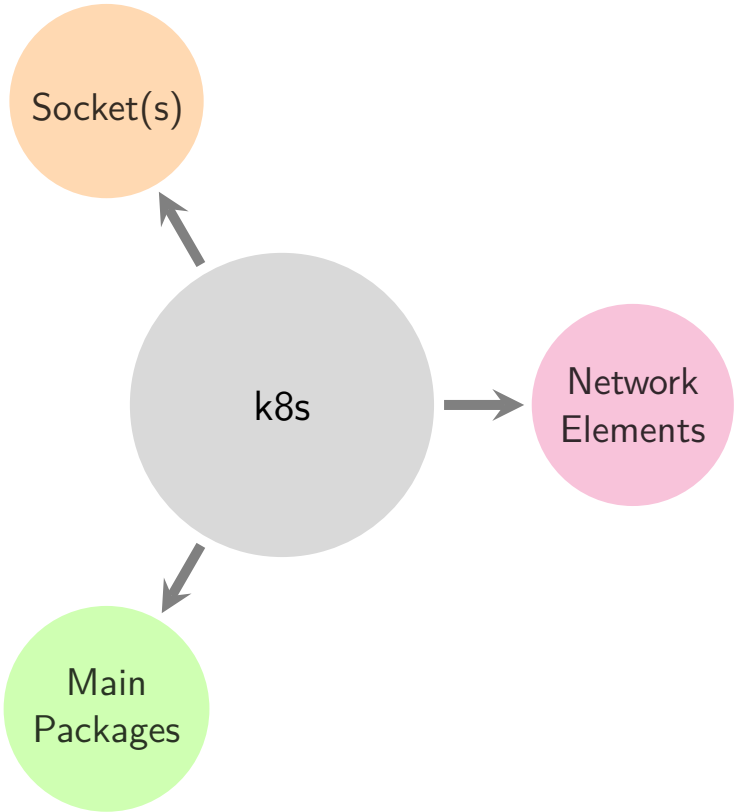
- Vault (https).
- Any socket.
  - Azzure Registry.
  - Pull Image.
- Deployment (volume).
- Logout Azzure.
- Validation (tests).
  - Raise error (if).
  - Stop, Cleanup.
  - Roll back.

Kubernetes Minimal Core Components



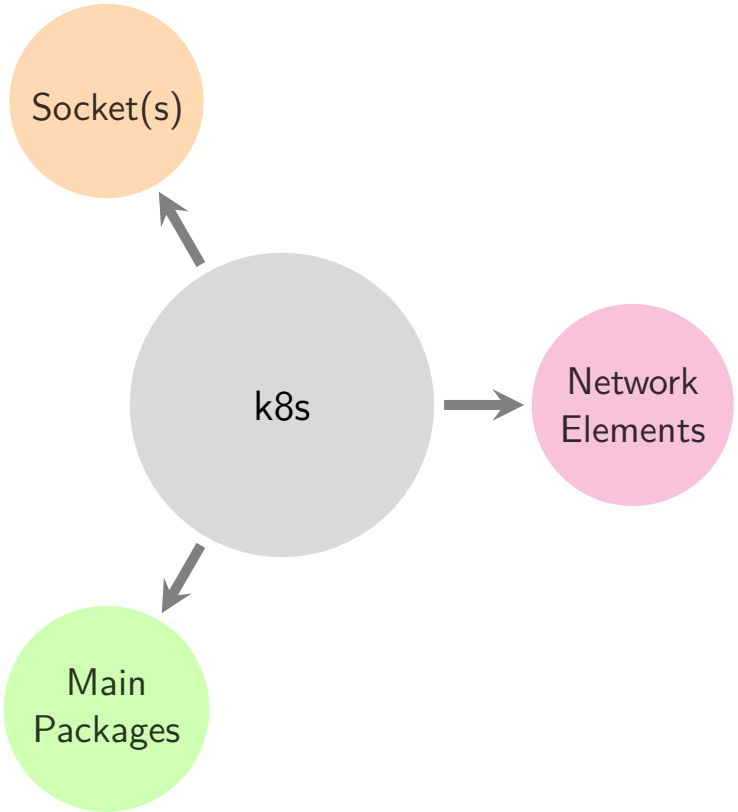
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - Weave Net [2]
  - Canal [3]

Kubernetes Minimal Core Components



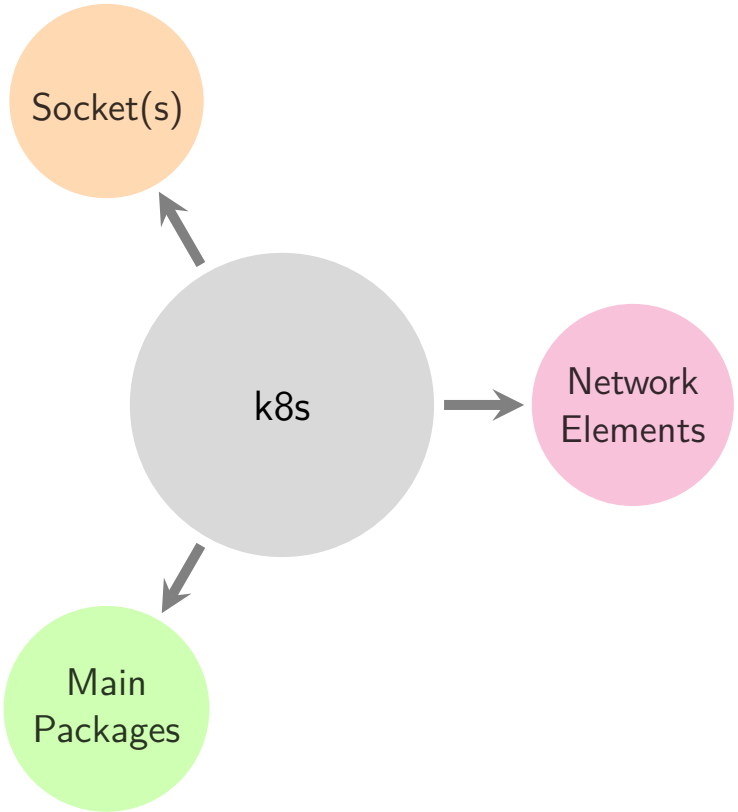
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - Weave Net [2]
  - Canal [3]

Kubernetes Minimal Core Components



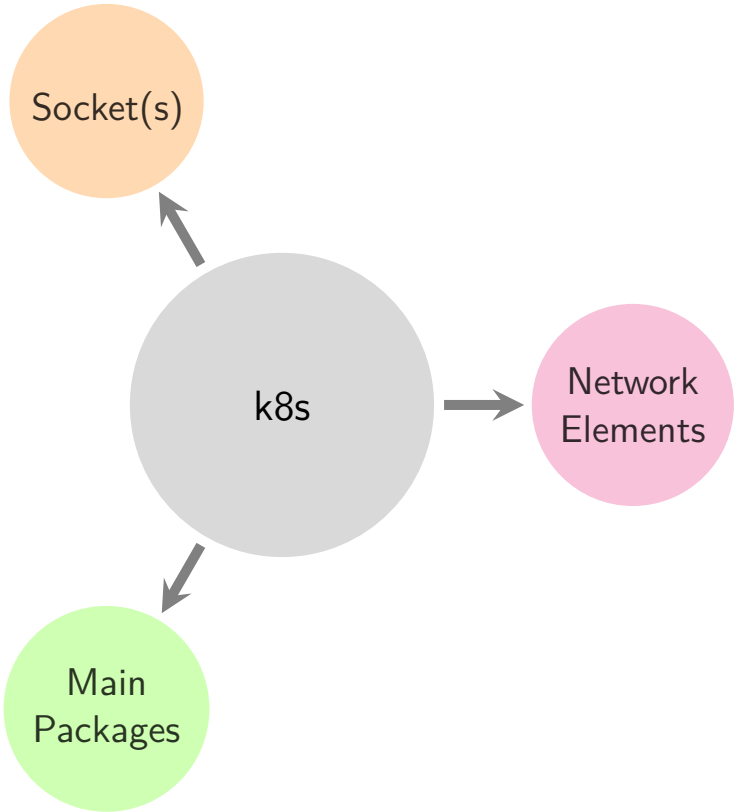
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - Weave Net [2]
  - Canal [3]

Kubernetes Minimal Core Components



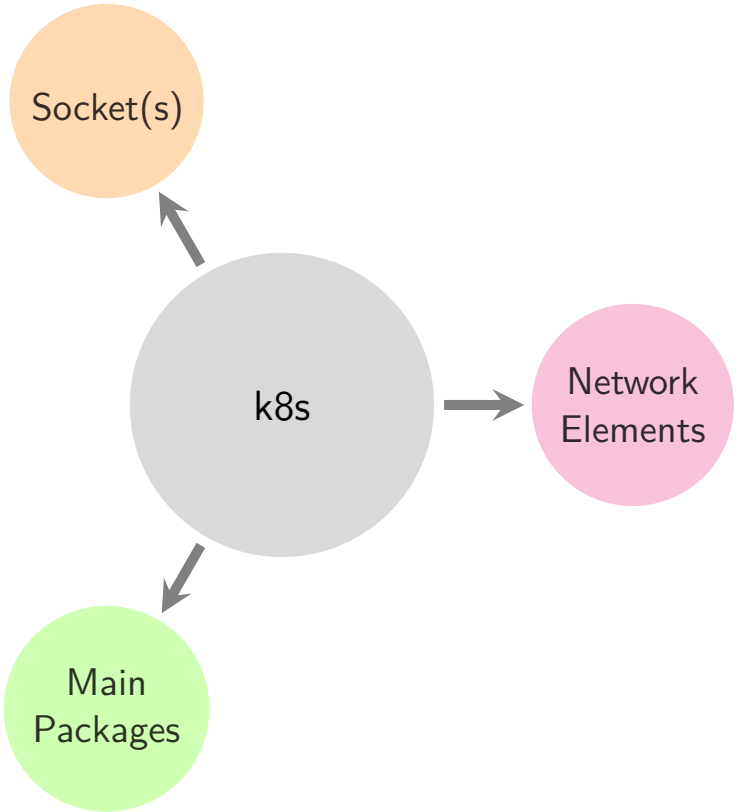
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - Weave Net [1]
  - Canal [1]

Kubernetes Minimal Core Components



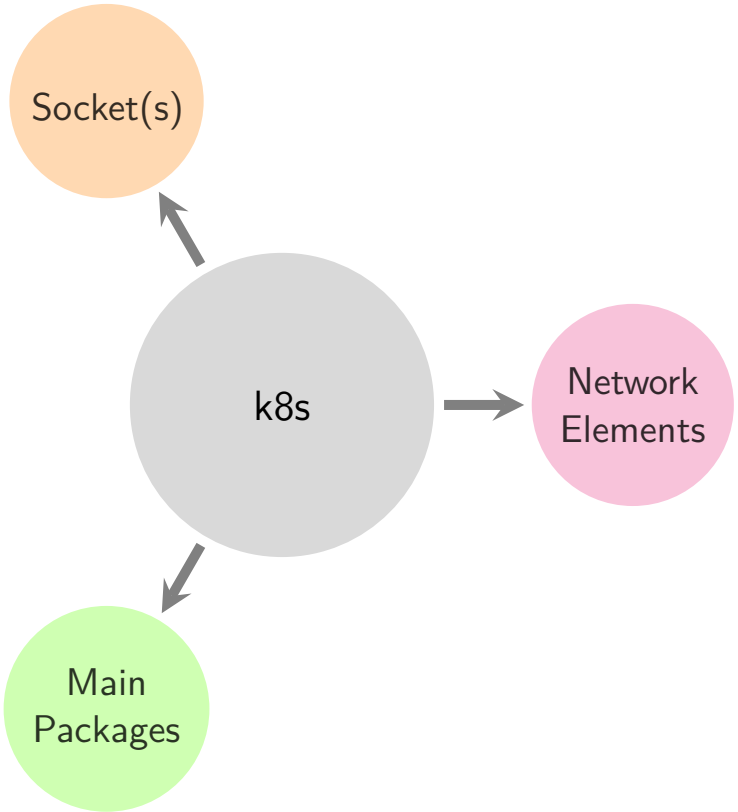
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - Weave Net [2]
  - Canal [3]

Kubernetes Minimal Core Components



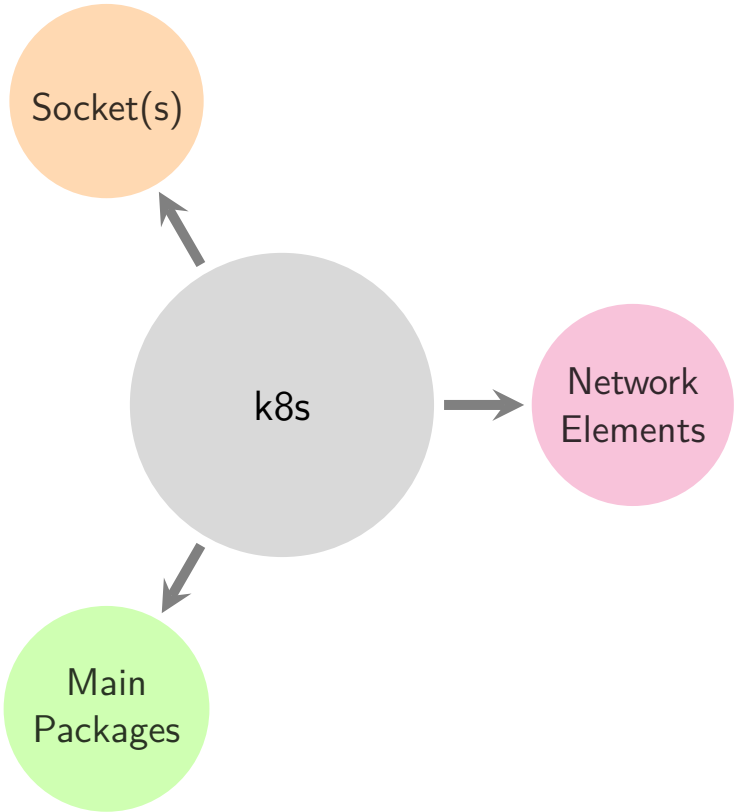
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements

Kubernetes Minimal Core Components



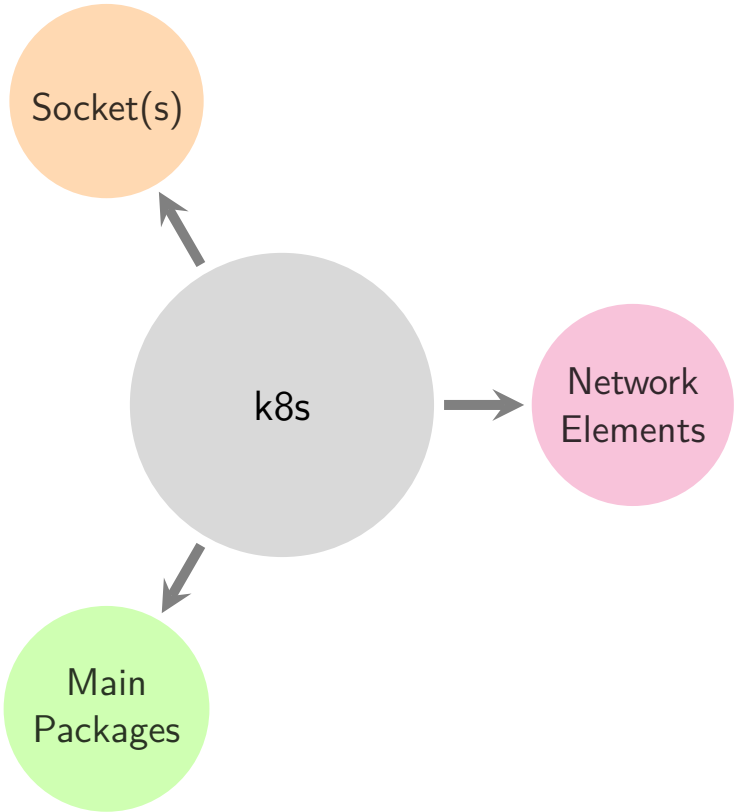
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - Weave Net [1]
  - Cilium [1]

Kubernetes Minimal Core Components



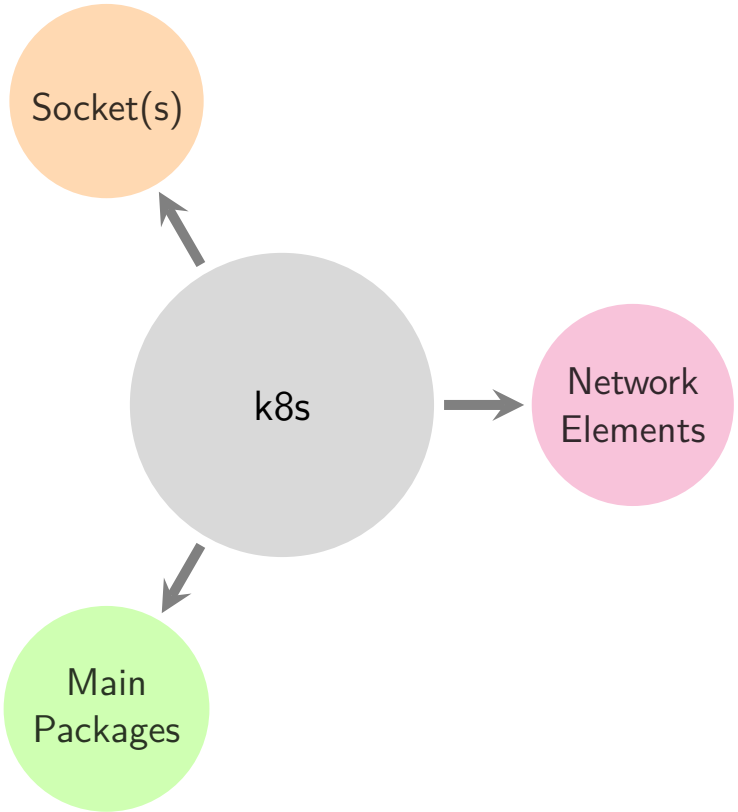
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - WeaveNet [1]
  - Cilium [1]

Kubernetes Minimal Core Components



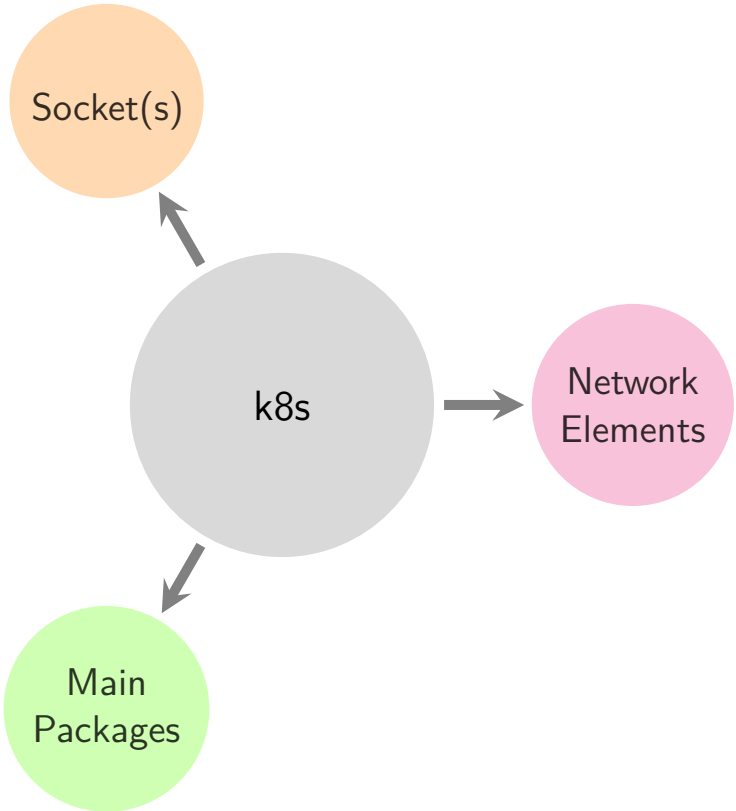
- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - WeaveNet [1]
  - Cilium [1]

Kubernetes Minimal Core Components



- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - WeaveNet [1]
  - Cilium [1]

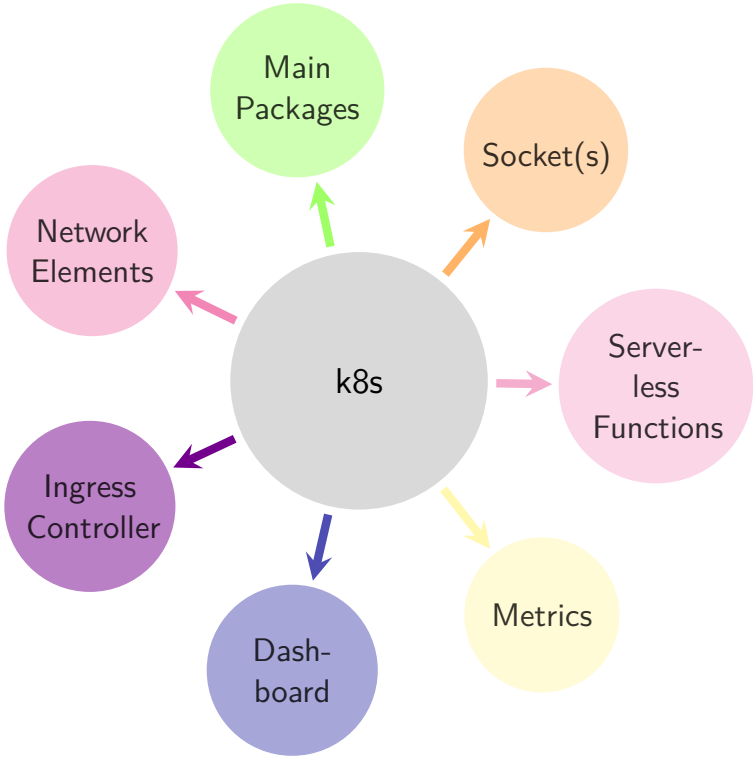
Kubernetes Minimal Core Components



- Socket:
  - Containerd (Docker)
  - CRI-O
- Main Packages:
  - kubeadm
  - kubelet
  - kubectl
- Network Elements
  - Calico [1]
  - WeaveNet [1]
  - Cilium [1]

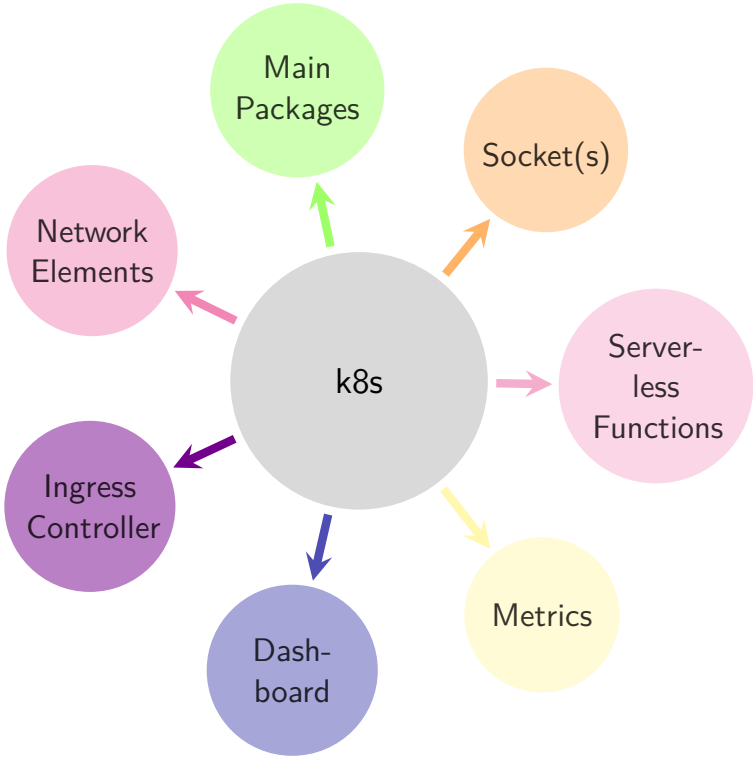
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



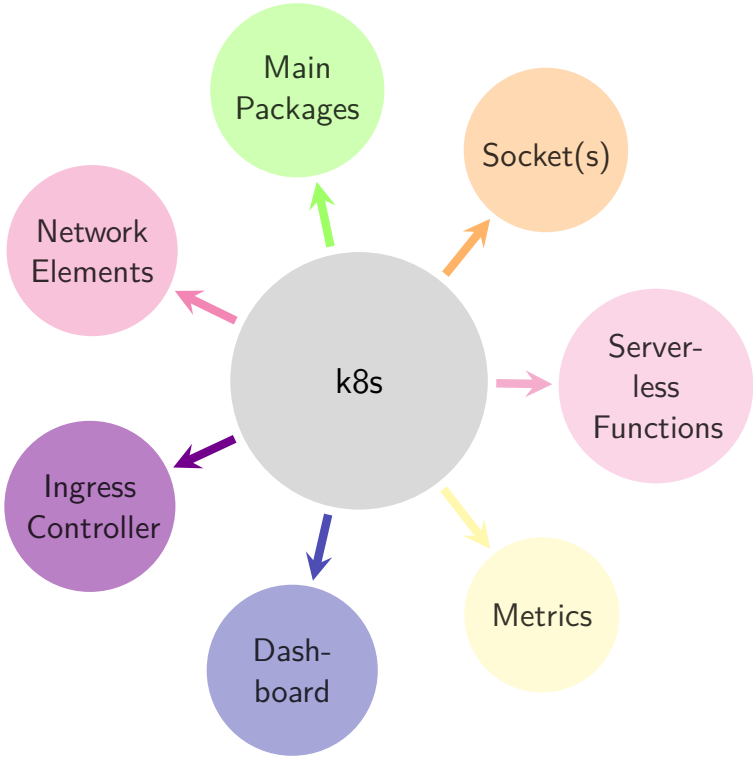
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



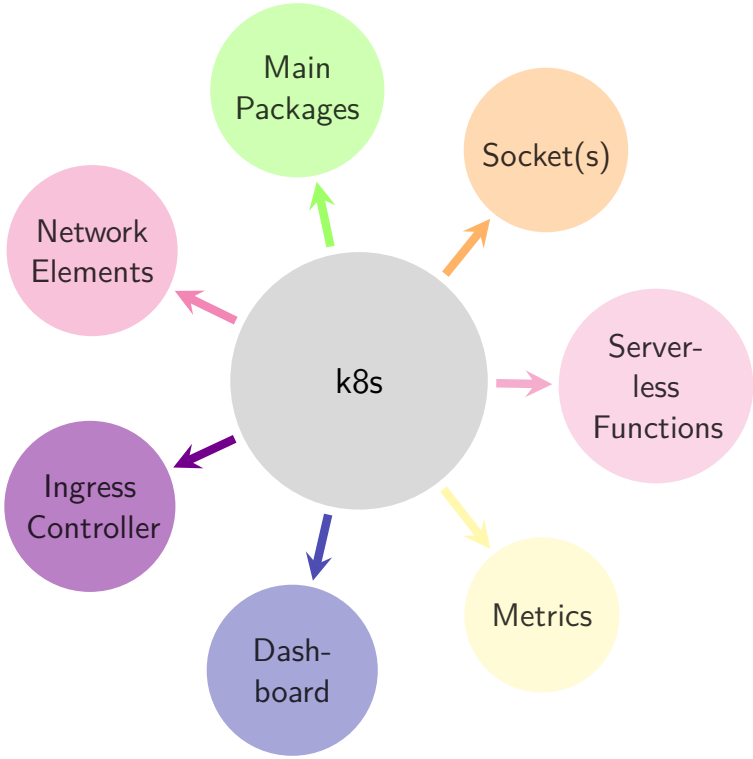
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



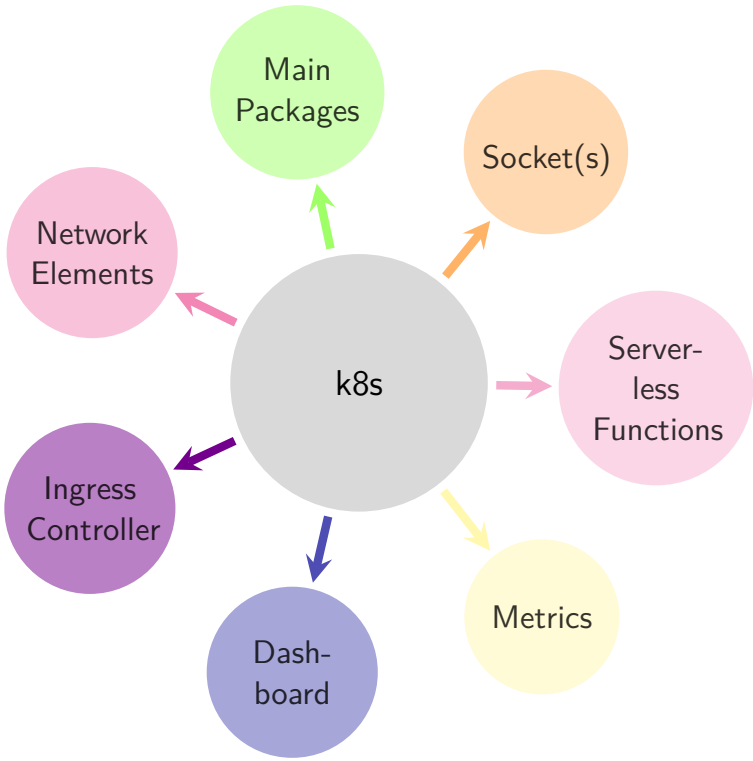
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



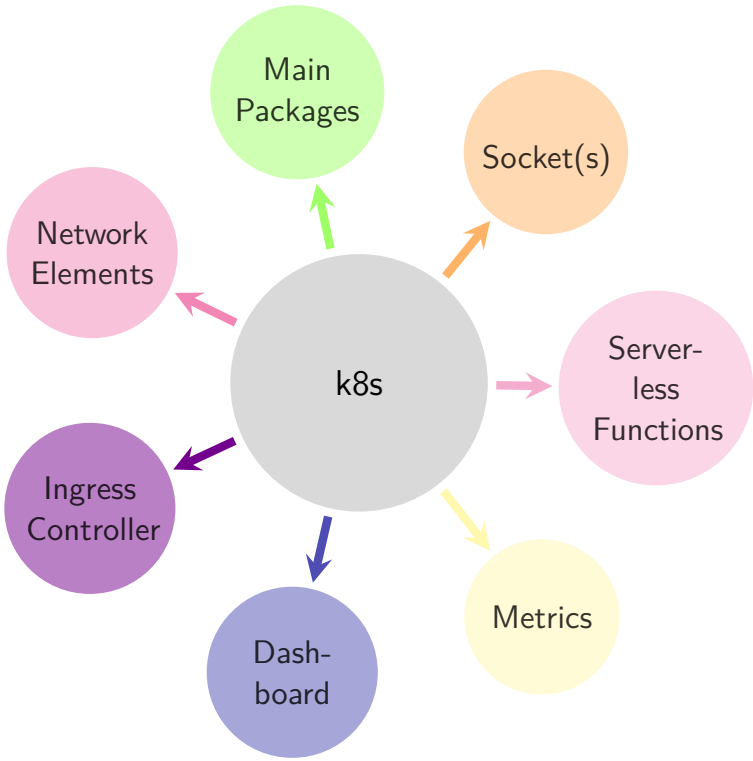
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



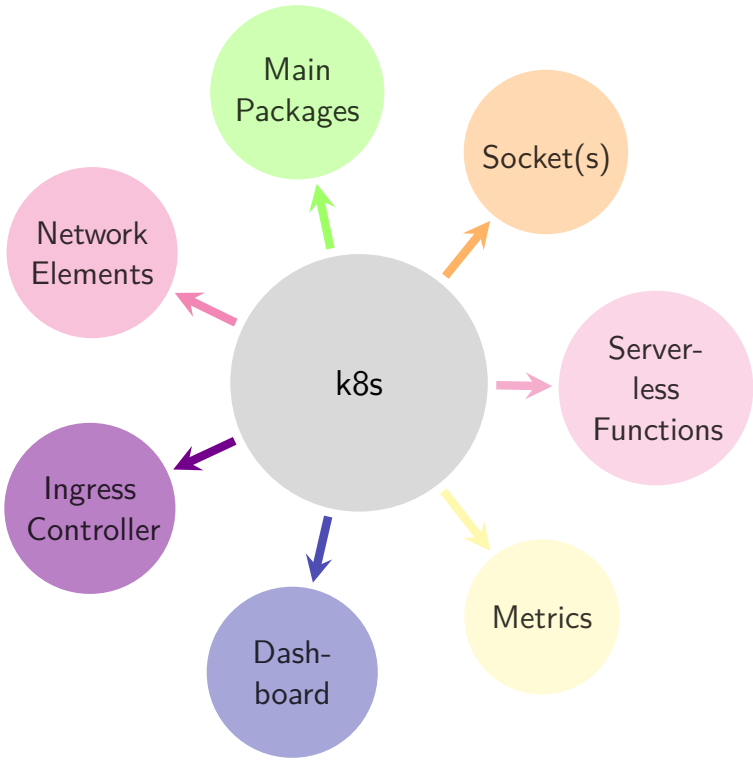
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



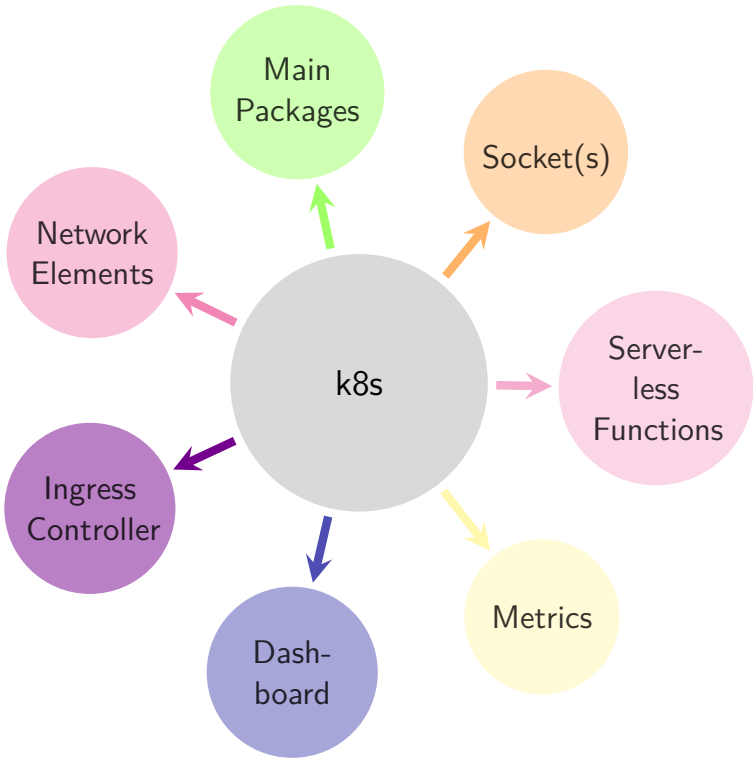
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



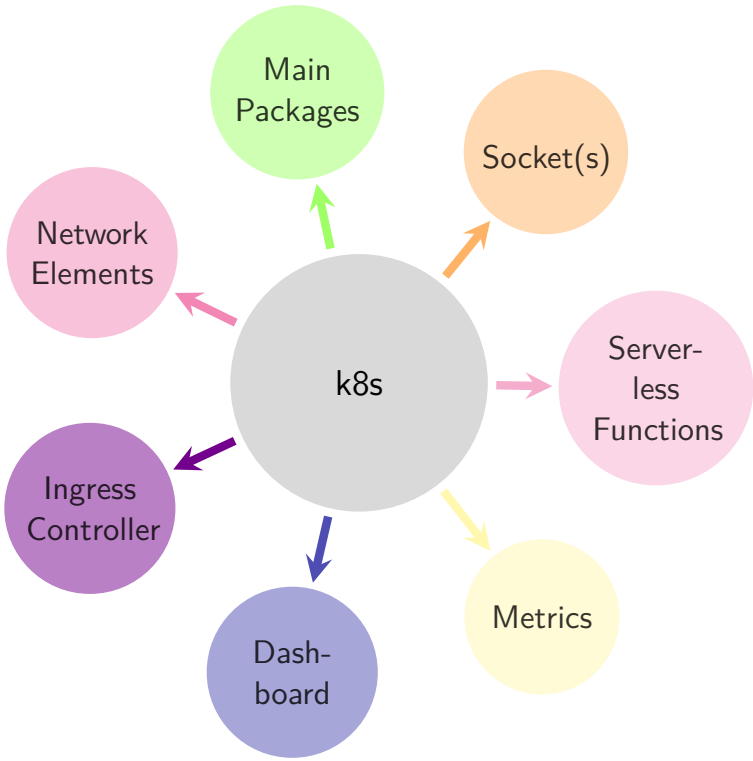
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



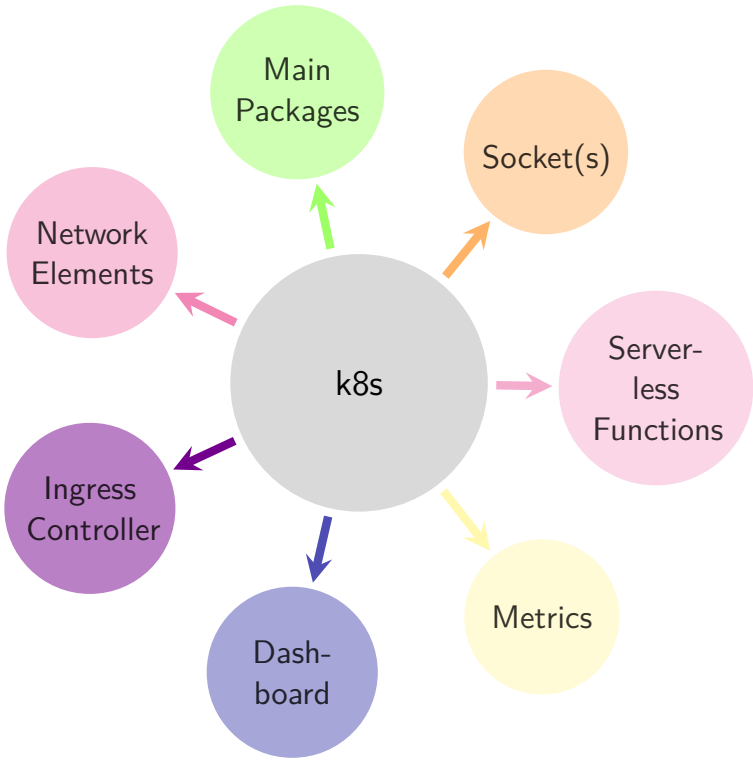
Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



Local Auto Deployment

- Previous Elements
- Ingress Controller:
  - NGINX [3]
  - HaProxy [3]
- Metrics (HPA)
- Dashboard
  - Self Signed CA
- Serveless Functions, Functions as a Service (FaaS)
- Operatin System:
  - Ubuntu



Coming up: Demo

Summary

- In section “Introduction” page “2” Container Run Time Interfaces.
- In section “CI / CD” page “3” Problems / Desires / Solutions on CI / CD.
- In section “Implementation” page “4” CI / CD Flow Build / Deploy / Test.

Extra Notes

- Demo on CI / CD build / deploy / validation and error handling cases.
- Both the CI / CD and Kubernetes project are provided as open source contribution. The Presentation was written in LaTeX

Summary

- In section “Introduction” page “2” Container Run Time Interfaces.
- In section “CI / CD” page “3” Problems / Desires / Solutions on CI / CD.
- In section “Implementation” page “4” CI / CD Flow Build / Deploy / Test.

Extra Notes

- Demo on CI / CD build / deploy / validation and error handling cases.
- Both the CI / CD and Kubernetes project are provided as open source contribution. The Presentation was written in LaTeX

Summary

- In section “Introduction” page “2” Container Run Time Interfaces.
- In section “CI / CD” page “3” Problems / Desires / Solutions on CI / CD.
- In section “Implementation” page “4” CI / CD Flow Build / Deploy / Test.

Extra Notes

- Demo on CI / CD build / deploy / validation and error handling cases.
- Both the CI / CD and Kubernetes project are provided as open source contribution. The Presentation was written in LaTeX

Summary

- In section “Introduction” page “2” Container Run Time Interfaces.
- In section “CI / CD” page “3” Problems / Desires / Solutions on CI / CD.
- In section “Implementation” page “4” CI / CD Flow Build / Deploy / Test.

Extra Notes

- Demo on CI / CD build / deploy / validation and error handling cases.
- Both the CI / CD and Kubernetes project are provided as open source contribution. The Presentation was written in LaTeX

Summary

- In section “Introduction” page “2” Container Run Time Interfaces.
- In section “CI / CD” page “3” Problems / Desires / Solutions on CI / CD.
- In section “Implementation” page “4” CI / CD Flow Build / Deploy / Test.

Extra Notes

- Demo on CI / CD build / deploy / validation and error handling cases.
- Both the CI / CD and Kubernetes project are provided as open source contribution. The Presentation was written in LaTeX

Summary

- In section “Introduction” page “2” Container Run Time Interfaces.
- In section “CI / CD” page “3” Problems / Desires / Solutions on CI / CD.
- In section “Implementation” page “4” CI / CD Flow Build / Deploy / Test.

Extra Notes

- Demo on CI / CD build / deploy / validation and error handling cases.
- Both the CI / CD and Kubernetes project are provided as open source contribution. The Presentation was written in LaTeX

Coming up: Q & A

References I

GNU LESSER GENERAL PUBLIC LICENSE
GNU Operating System
available at https://www.gnu.org/licenses/lgpl.html.

Author: K. Community
Container Runtimes
available at https://kubernetes.io/docs/setup/production-environment/container-runtimes/.

Author: K. Community
Cluster Networking
available at https://kubernetes.io/docs/concepts/cluster-administration/networking/.