Automation for Setup and Configuration of OpenShift

Andy Wirtz

15th October 2019

Andy Wirtz

Andy:

- IT Consultant at ATIX AG, Germany
- Automation of data centers
- Deployment of cloud native services
- Expertise in Docker, Kubernetes, Istio

Contact:

- Phone: +49 (0)89 452 35 38-248
- Mail: wirtz@atix.de
- www.xing.com/profile/Andy_Wirtz2
- www.linkedin.com/in/andy–wirtz

Agenda

Agenda

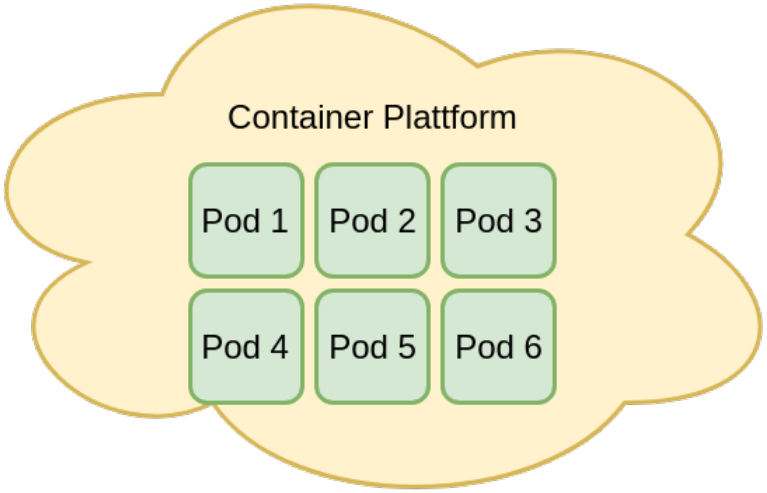
Container Platforms

Container platforms:

- Hardware abstraction
- Cloud for containers
- Efficient resource management

Automation:

- Scaling dependent on the requests
- Some ready to use pods in stock
- Self-healing
- Updates without downtime



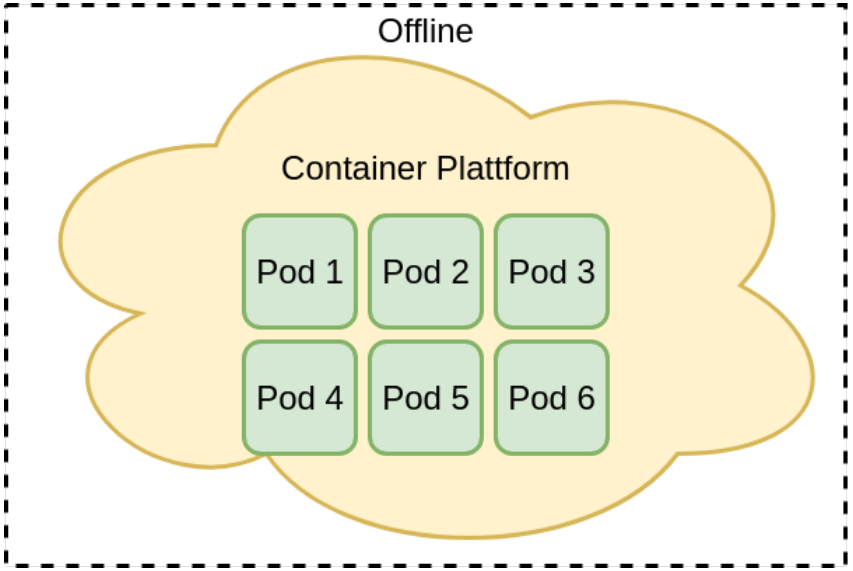
On-Premises

On-Premises:

- Cloud-native technologies for the own data centers
- Private cloud for containers
- Defining own security and protection of data privacy

Offline:

- Disconnected installation and operation
- No direct connection to the internet



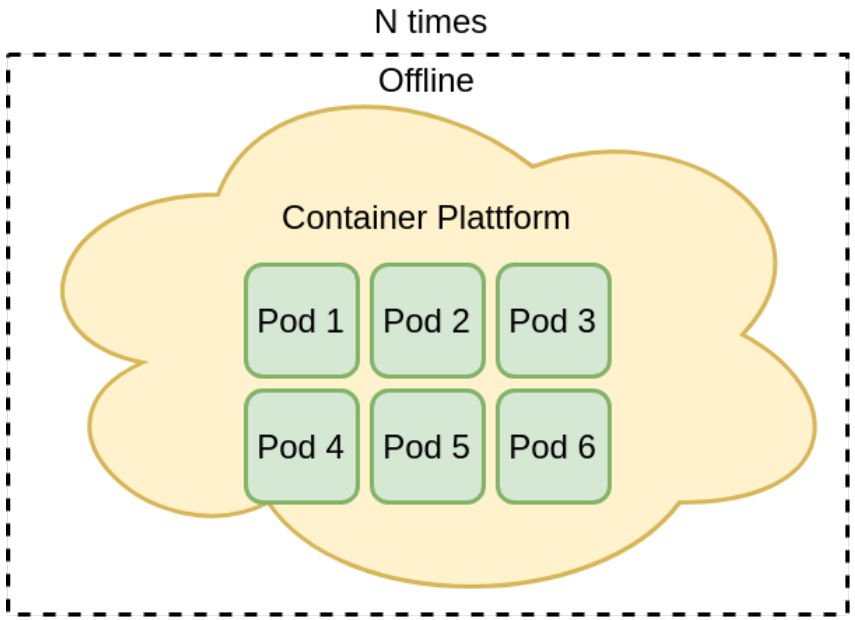
Automation

Automation:

- Standardization
- Reproducibility
- Necessary in a highly dynamic IT world

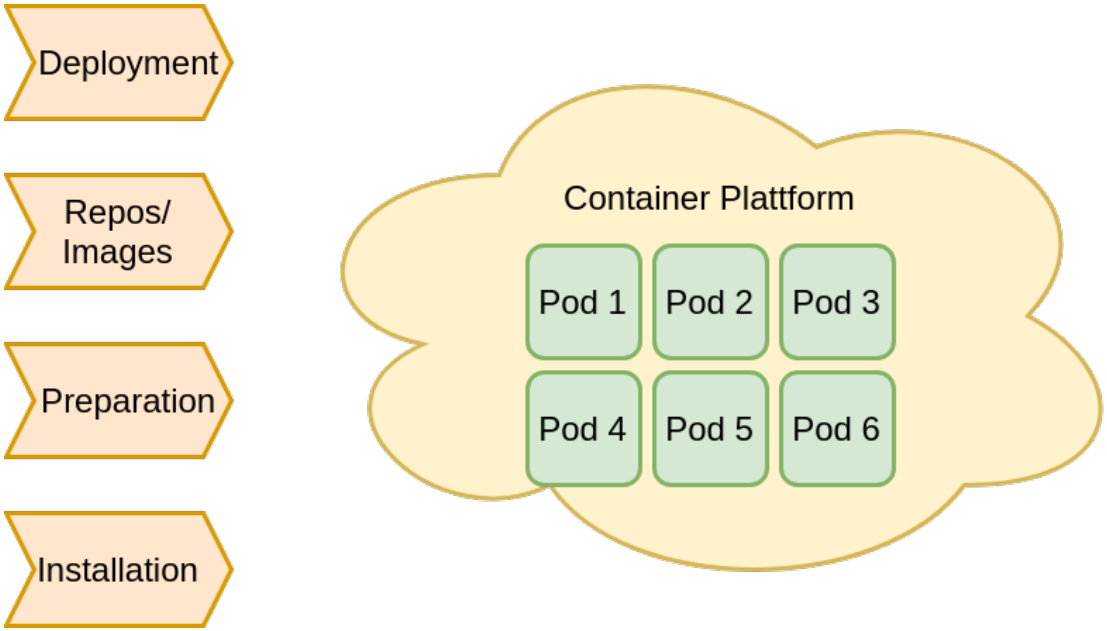
Container platforms:

- Automation of the setup
- Automation of the cluster
- Automation of the configuration



Agenda

Setup overview



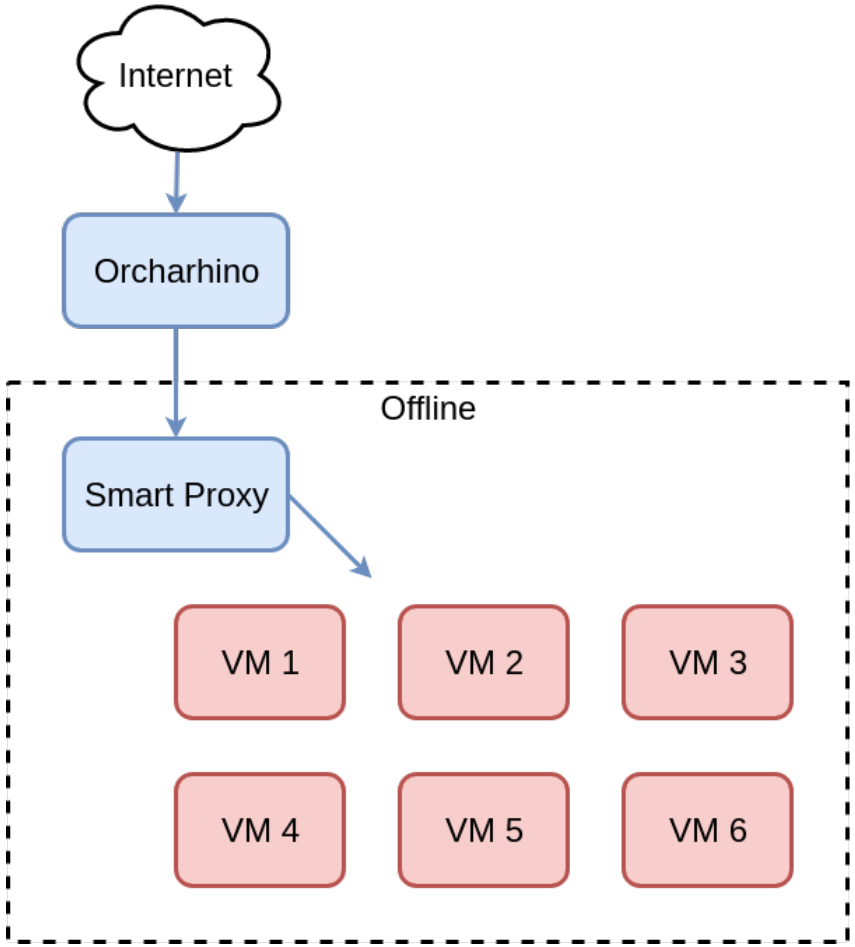
Offline installation

Offline installation:

- Hosts are disconnected from internet
- They need repositories/images for installation

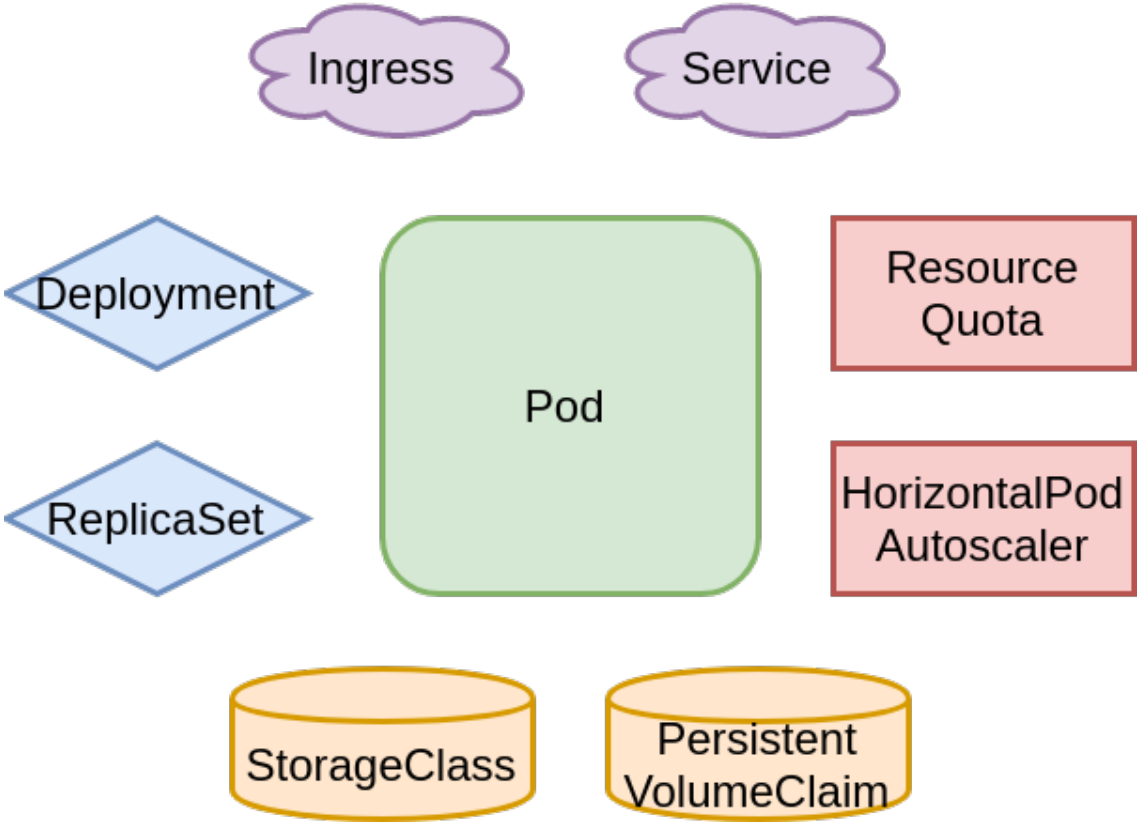
Orcharhino:

- Lifecycle management tool
- Based on TheForeman & Katello
- Provides repositories/images
- Smart proxy synchronizes
- Smart proxy is local mirror for offline hosts



Agenda

Cluster overview



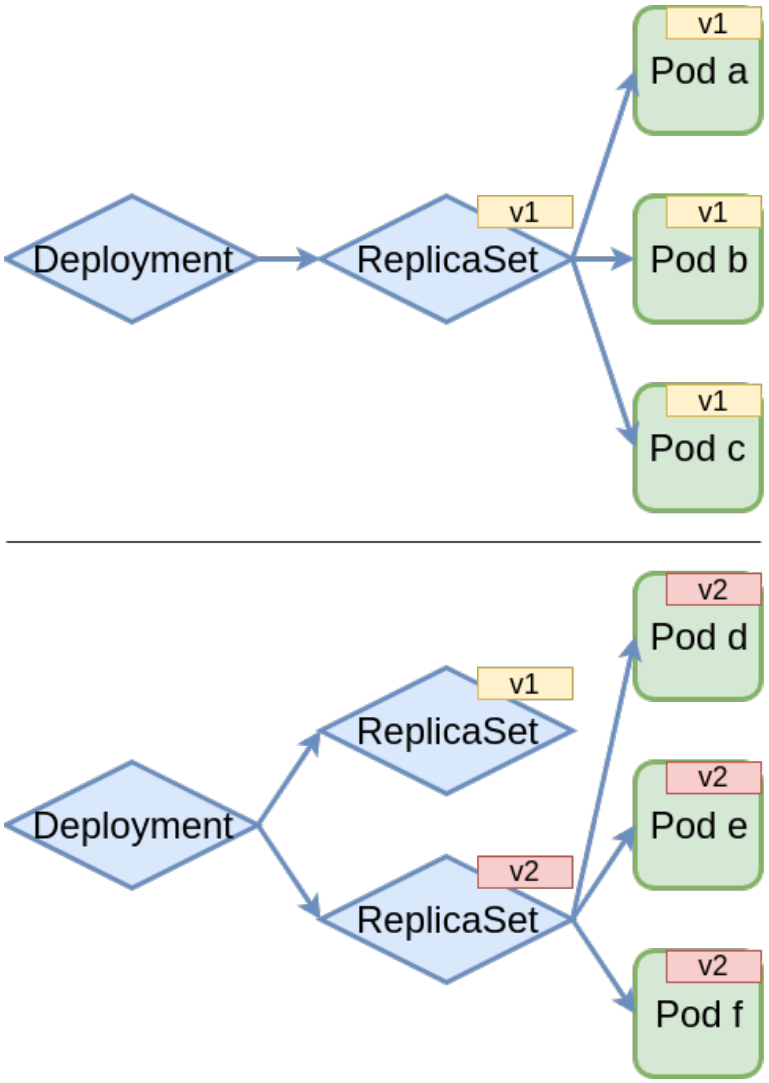
Deployment Object

Deployment object:

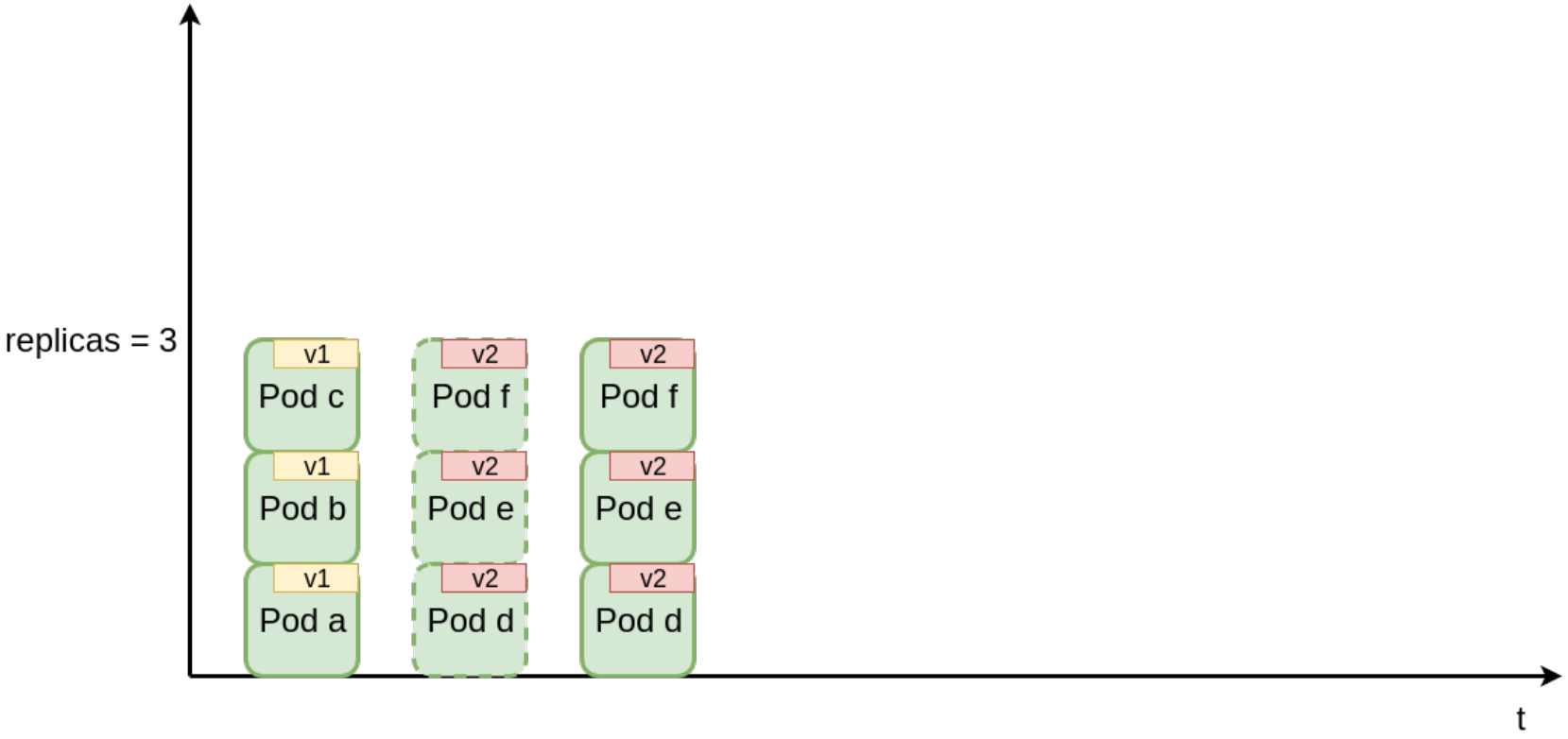
- Update apps declaratively
- Trigger rollout by applying a change
- Tool reacts

Update strategies:

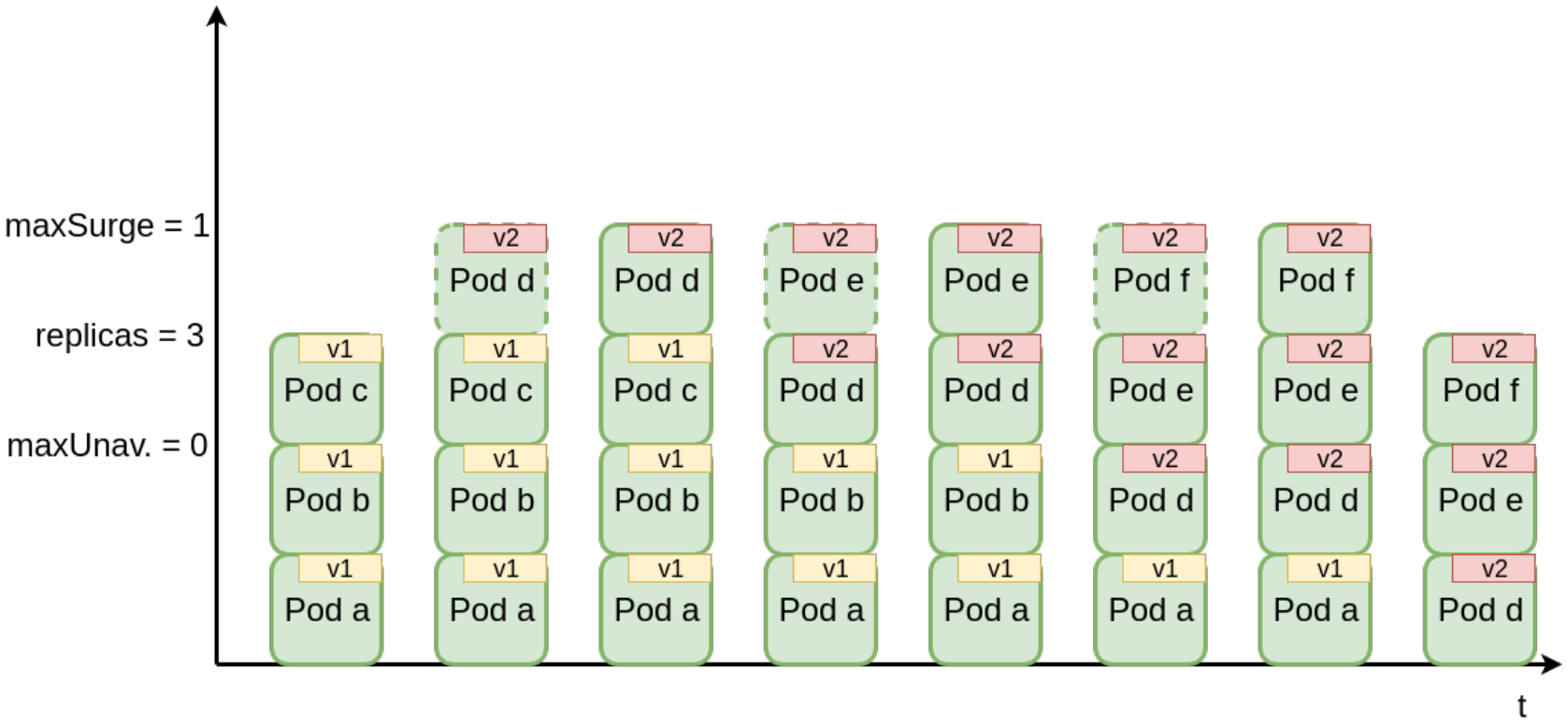
- Recreate strategy
- Rolling update
- Blue green deployment
- Canary release



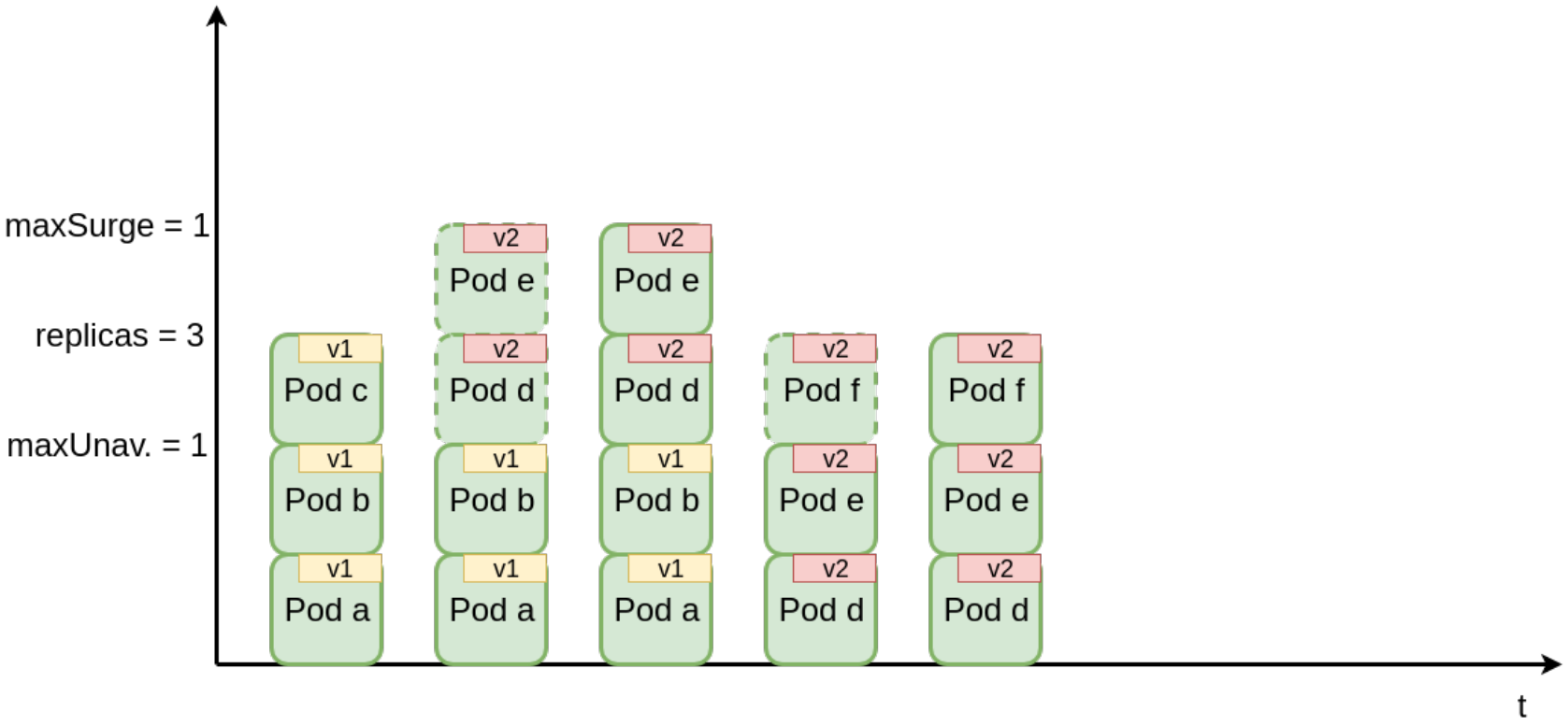
Recreate Strategy



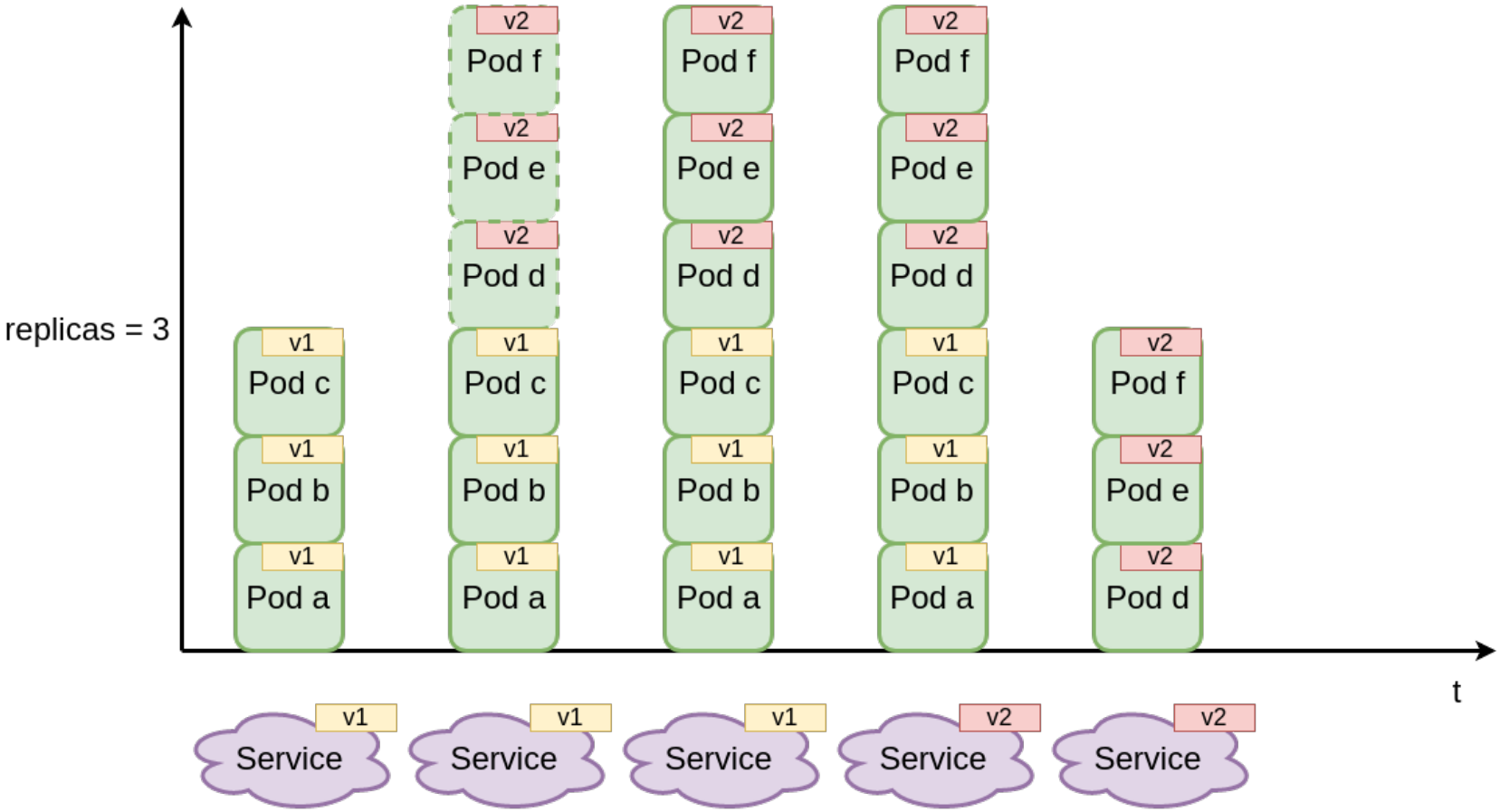
Rolling Update 1



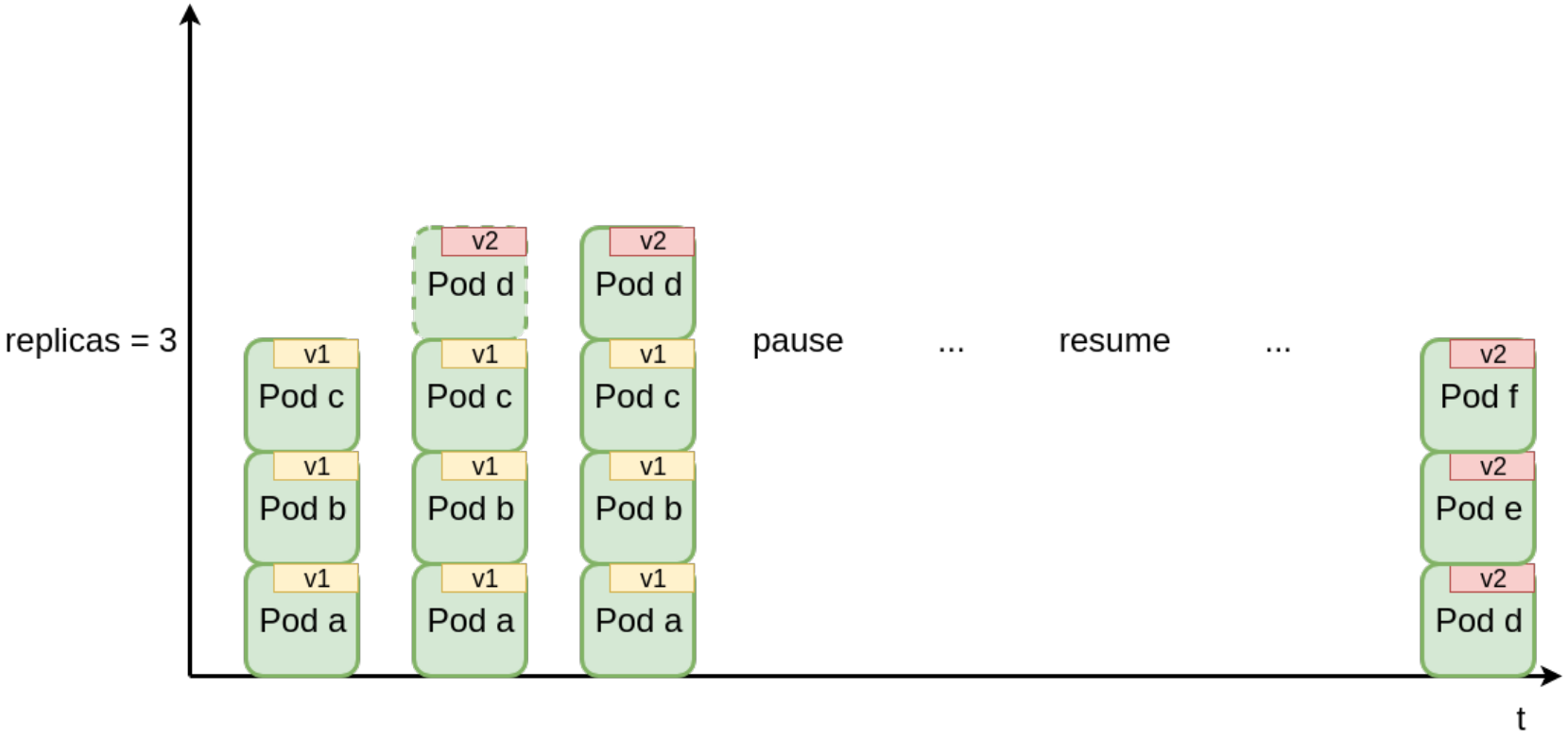
Rolling Update 2



Blue Green Deployment



Canary Release



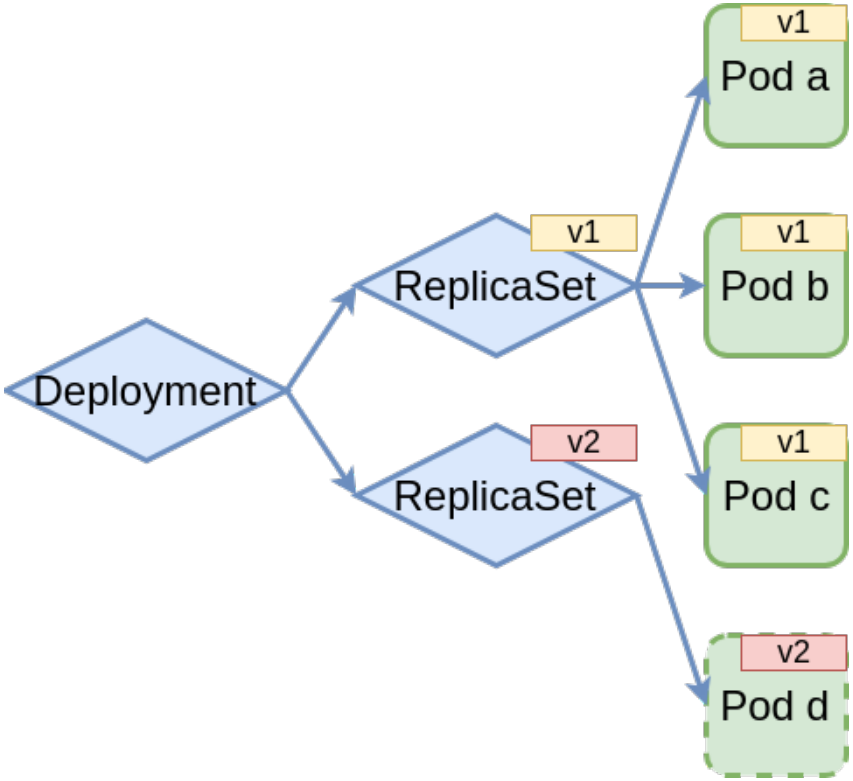
Automatic Stoppage

Automatic stoppage:

- Block rollouts of bad versions
- Define readiness probe
- Use paramenter "minReadySeconds"

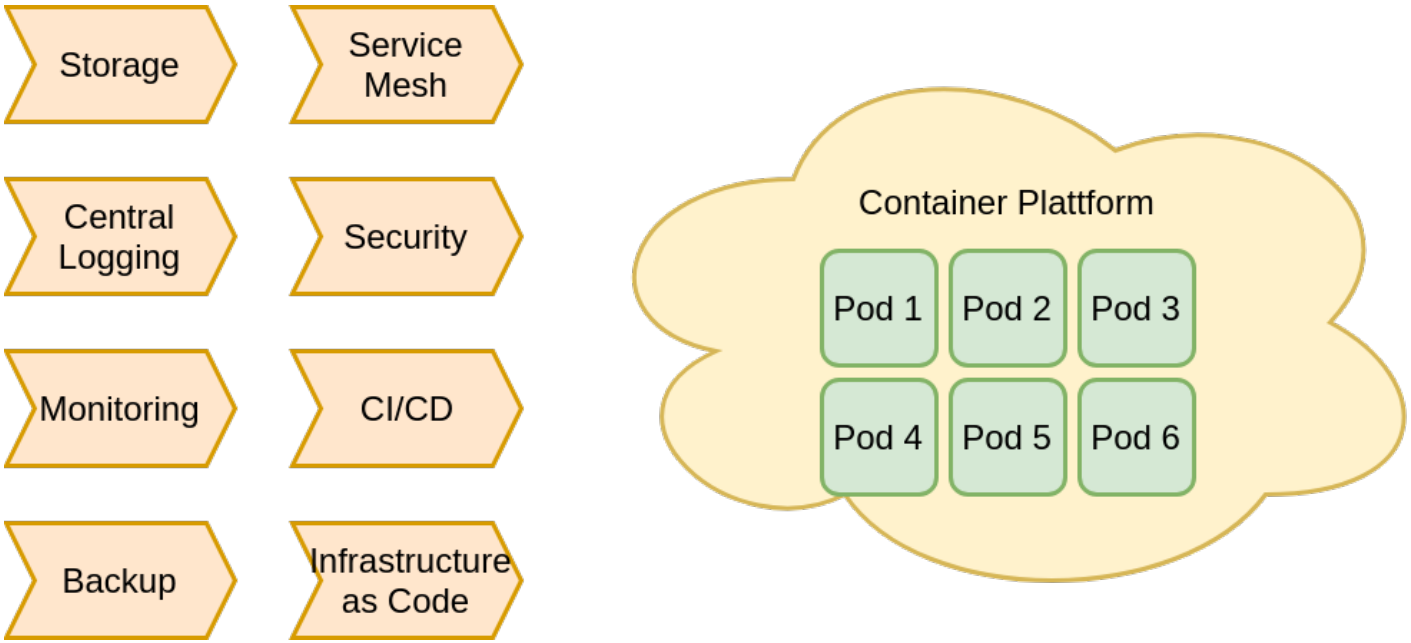
Readiness probes:

- HTTP GET probe
- Exec probe
- TCP socket probe



Agenda

Configuration overview



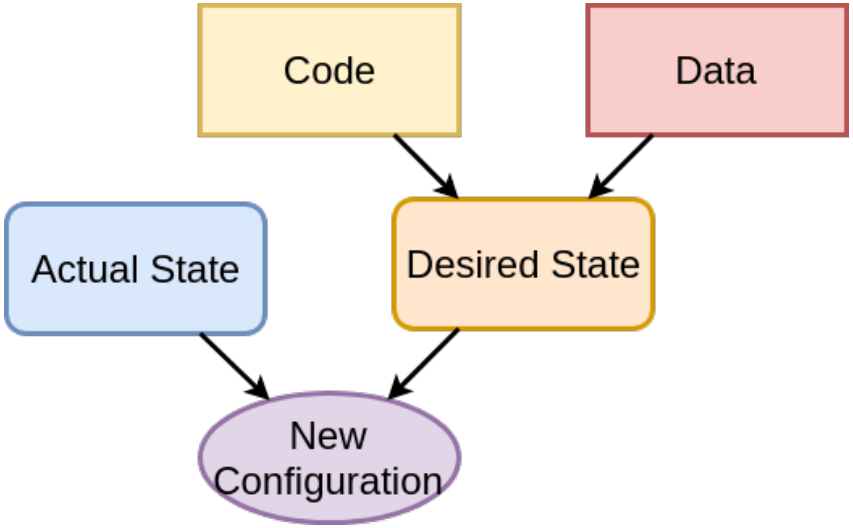
Infrastructure as Code

Infrastructure as Code:

- Declarative model
- Define your desired state
- Tool compares desired and actual state
- Tool acts if need be

Cluster administration via code:

- Use version control repositories
- Seperate code and data
- Make use of idempotence



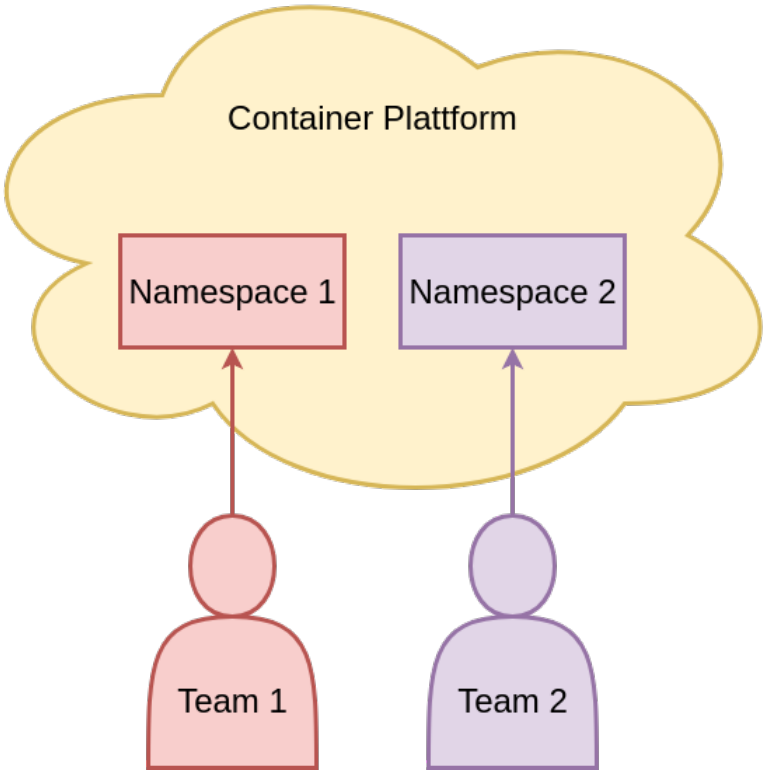
Namespaces

Namespaces:

- Seperate teams and applications

Projects:

- OpenShift object
- For multitenancy



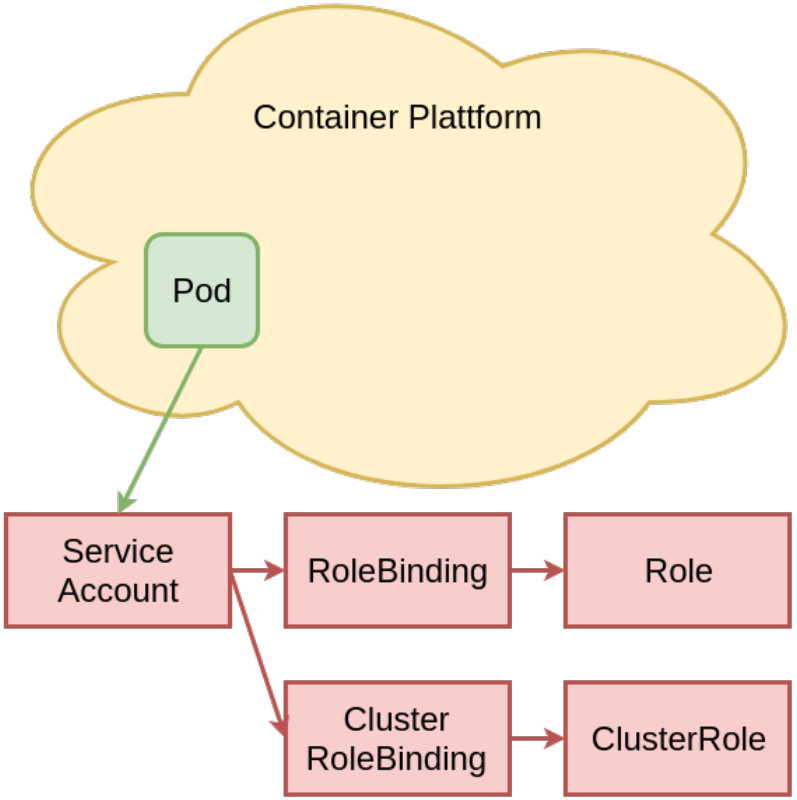
Role-Based-Access-Control

Role-Based-Access-Control:

- Service accounts: "who"
- (Cluster) role bindings: "is allowed to"
- (Cluster) roles: "perform what to whom"

Users and groups:

- OpenShift objects
- For humans instead of processes in pods
- Sync with corperate database



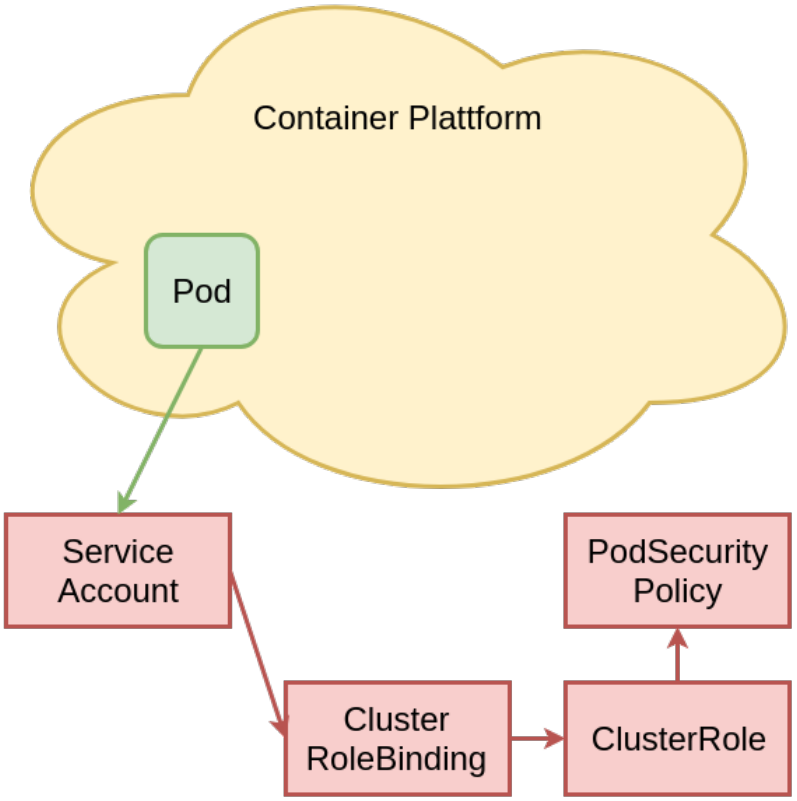
Pod Security Policies

Pod security policies:

- Restrict security-related features
- Ensure isolation to hosts
- Prevent priviled escalation

Security context constraints:

- OpenShift object
- Same objective



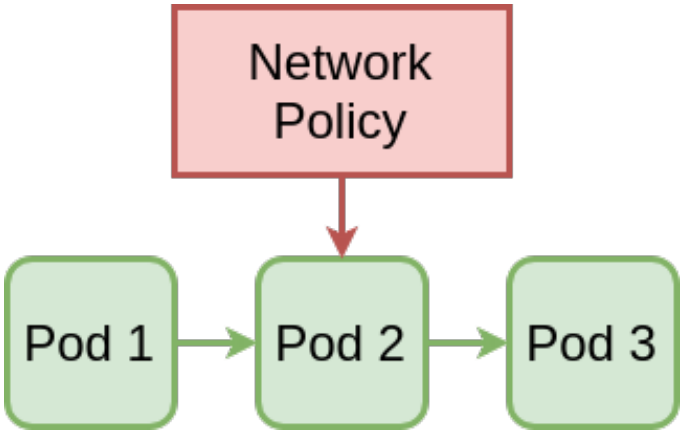
Network Policies

Network policies:

- Isolates the pod network
- Limit inbound and/or outbound traffic
- Cluster admin can define default deny

Isolation via:

- Namespaces
- Labels
- Pod IPs



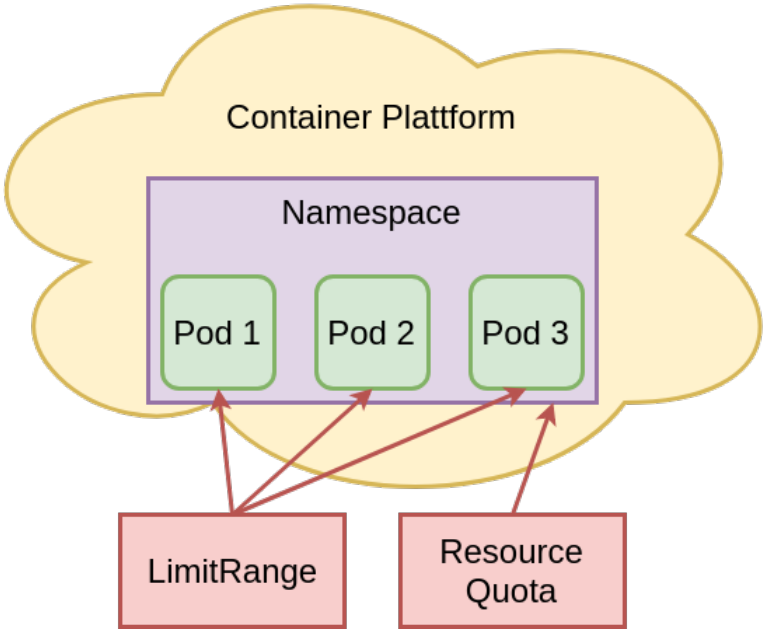
Resource Management

Resource management:

- Limit range: defines min, max, default limits and requests for pods
- Resource quota: defines amount of resources available for pods in namespace

Resources:

- CPU and memory
- Storage
- Number of objects

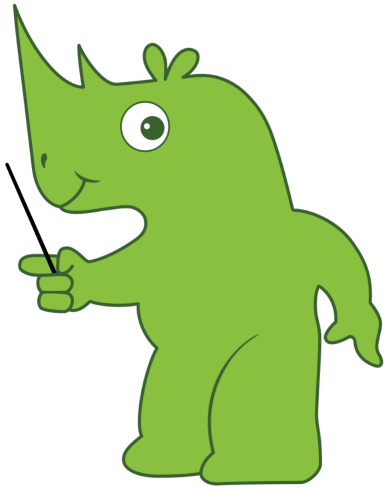


Agenda

Automation of the Setup

Orcharhino:

- Deployment of hosts (VMs)
- Provisioning of repositories/images
- Preparation of the hosts
- Installation of the container platform



FOREMAN

ANSIBLE

Automation in the Cluster

Kubernetes/OpenShift:

- Reconciliation loop
- Rolling updates
- Service discovery
- Load balancing
- Storage provisioning
- Storage binding
- Quota enforcement
- Horizontal scaling

Automation of the Configuration

Git:

- Distributed version control system
- Infrastructure as Code
- Seperate code and data
- Use CI/CD

Kubectl apply:

- Apply changes declaratively
- Rollout new configurations
- Make use of idempotence