KUBERNETES IN A GROWN ENVIRONMENT AND INTEGRATION INTO CONTINUOUS DELIVERY

Stephan Fudeus, Expert Continuous Delivery
Dr. Sascha Mühlbach, Expert Infrastructure Architect

1&1

United Internet / 1&1 Mail & Media

United Internet

- Is a leading European internet specialist
- > 9000 employees
- 90k servers in 10 data centers
- Access
  - DSL and Mobile
- Applications
  - Business (Server, Hosting)
  - Consumer (WEB.DE etc.)

1&1 Mail & Media

- Main brands GMX, WEB.DE and MAIL.COM
- Various services around a free or paid mail account (calendar, news portal, cloud storage)
- 33 million active users / month

Speakers

- Stephan Fudeus
  - Joined 1&1 in 2005
  - Long-term experience in building highly scalable multi-tenant applications
  - Product Owner and TechLead for our Kubernetes Clusters
  - Twitter: @der_sfu

- Dr. Sascha Mühlbach
  - Expert Infrastructure Architect
  - 15 years professional experience
  - Responsible for the global operations strategy of the applications and systems infrastructure

Agenda

- Motivation / Environment

- Cluster-Design
- Network-Setup / Ingress

- Git-driven cluster operations
- Multi-Tenancy

- Build processes
- Continuous delivery environment
- Onboarding / Training

Motivation

- Why Container?
  - Strong coupling between code and application runtime environment
  - One build responsibility
  - Hide core infrastructure from application
  - Reproducibility in development
  - Follow new standards in software development

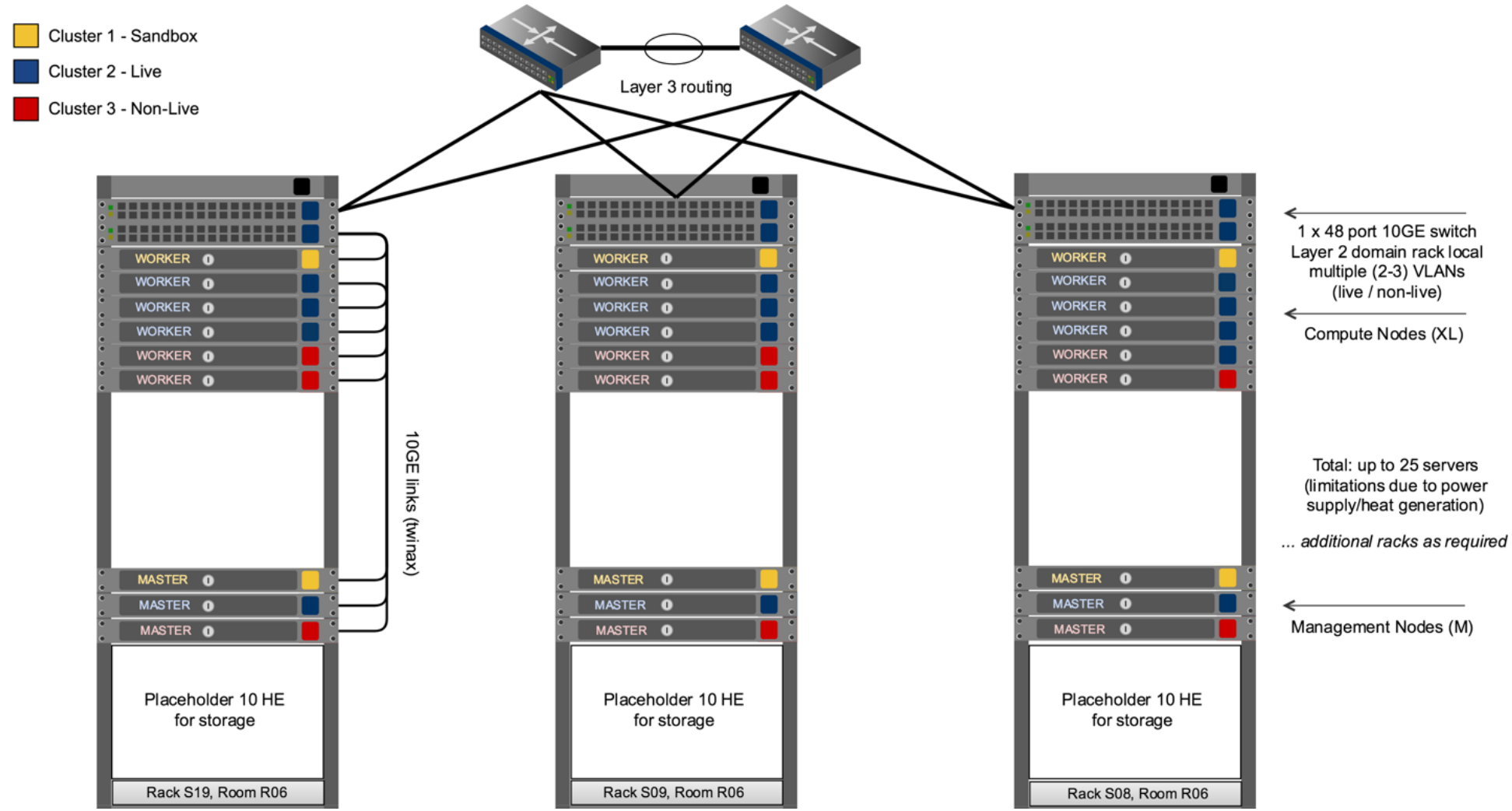
- Requirements
  - Reliable platform that provides the same level of availability that our existing environment is delivering
  - Efficient deployment for geo-redundant services in multiple data centers
  - Self-service for the development and application teams
  - Multi-tenancy with strong separation for security reasons
  - Must fit into the existing network environment
  - Mostly automated operation of the base Kubernetes platform

Environment

- Organizational Environment
  - Approx. 25 Dev Teams with 10 Ops Teams
  - Strong organizational separation between PM / Dev / Ops
  - 24/7
  - Central Ops Team to build and run the Kubernetes platform

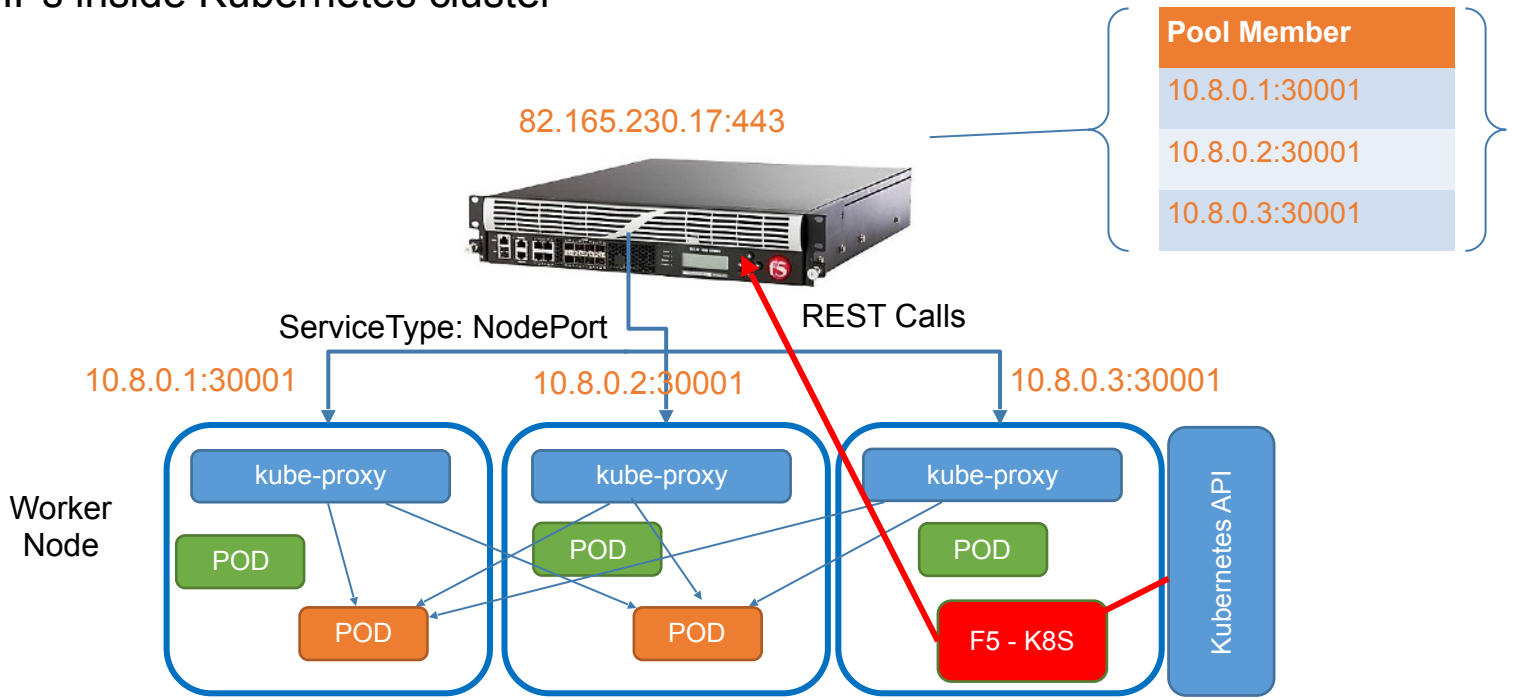
- Technical Environment
  - 3 datacenters (2 in DE, 1 in US) that are owned by us
  - bare metal and virtual machines (KVM, ESX)
  - All servers are Puppet managed
    - Our infrastructure has ~15.000 Puppet clients
  - Majority of services are written in Java
  - Ongoing transition to CD and microservices

Cluster Design



Network Setup for Frontend Zone

- Integration of existing F5 BigIP load balancing platform with their features
- Service IPs are BGP-routed to Balancer and then forwarded with SNAT to NodePorts
- BGP enables global redundancy
- No public IPs inside Kubernetes cluster

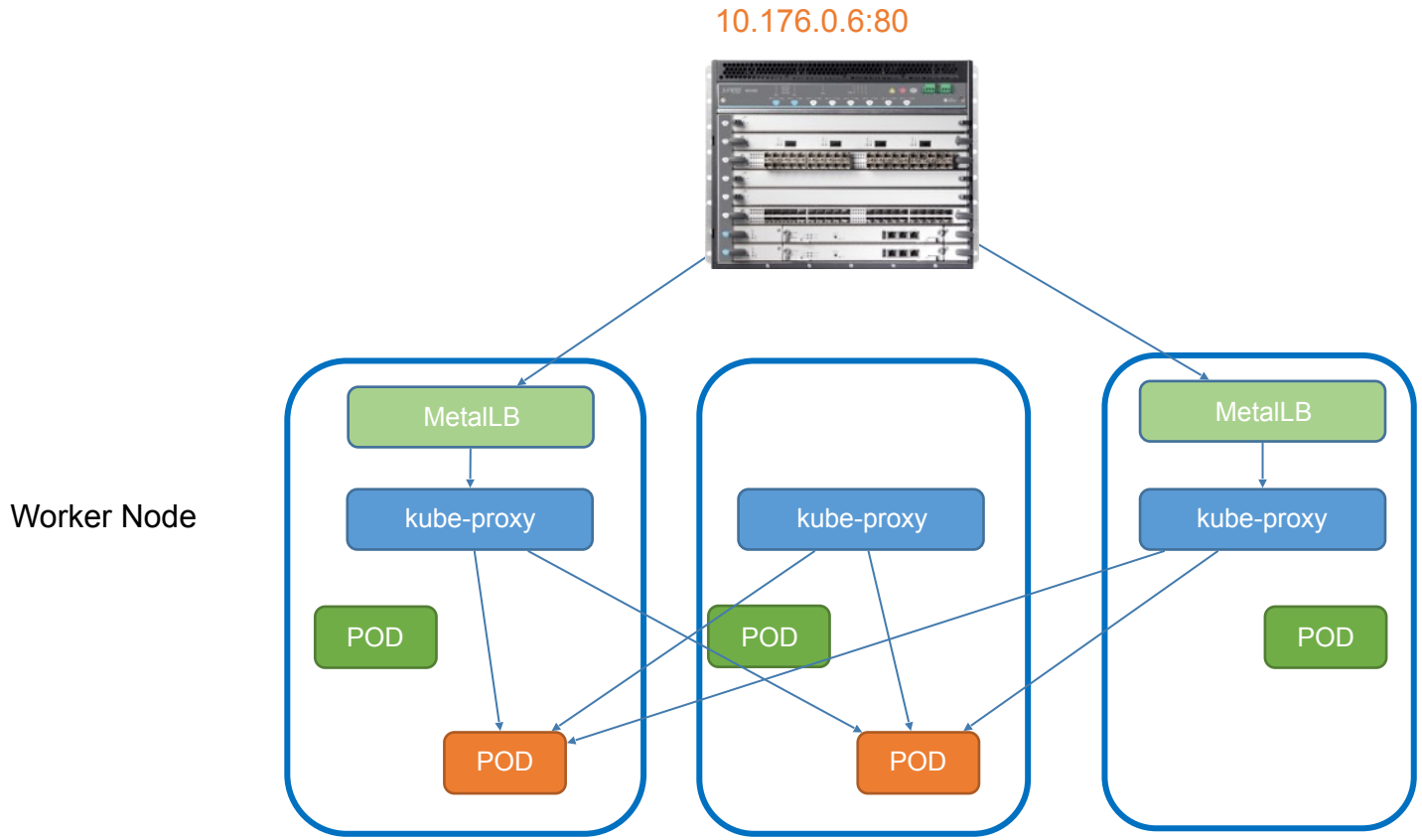


Network configuration via ConfigMap for F5

```
apiVersion: v1
kind: ConfigMap
metadata:
  labels:
    app: prometheus
    f5type: virtual-server
    pipeline-managed: "true"
  name: prometheus-ingress
  namespace: monitoring
data:
  schema: "f5schemadb://bigip-virtual-server_v0.1.7.json"
  data: |
    {
      "virtualServer": {
        "backend": {
        "serviceName": "prometheus",
        "servicePort": 9090
        },
        "frontend": {
          "partition": "k8s-be-qa-iz1-bs",
          "iapp": "/Common/K8S_iApp_010",
          "iappPoolMemberTable": {
            "name": "Pool_Definition__Pool_Members",
            "columns": [
              {"name": "Pool_Member_IP", "kind": "IPAddress"},
              {"name": "Pool_Member_Port", "kind": "Port"}
            ]
          },
          "iappOptions": {
            "description": "Prometheus monitoring for Kubernetes"
          },
          "iappVariables": {
            "Virtual_Definition__VS_Type": "{{ .Values.global.ingress.type }}",
            "Virtual_Definition__VS_VLANs_Enabled": "{{ .Values.global.ingress.vlan }}",
            "Virtual_Definition__VS_TCP_Client_Profile": "{{ .Values.global.ingress.tcp_client_profile }}",
            "Virtual_Definition__VS_HTTP_Profile": "{{ .Values.global.ingress.http_profile }}",
            "Virtual_Definition__VS_SSL_Client_Profiles": "{{ .Values.global.ingress.ssl_client_profiles }}",
            "Virtual_Definition__VS_Persistence": "{{ .Values.global.ingress.stickyness }}",
            "Virtual_Definition__VS_SSL_Server_Profiles": "{{ .Values.global.ingress.ssl_server_profiles }}",
            "Virtual_Definition__VS_TCP_Server_Profile": "{{ .Values.global.ingress.tcp_server_profile }}",
            "Virtual_Definition__VS_iRules": "{{ .Values.global.ingress.irules }}",
            "Pool_Definition__Pool_Health_Monitors": "{{ .Values.global.ingress.healthmonitor }}",
            "Virtual_Definition__VS_IP": "{{ .Values.global.ingress.vip }}",
            "Virtual_Definition__VS_Port": "{{ .Values.global.ingress.port }}"
          }
        }
      }
    }
```

Network Setup for Backend Zone

- In backend networks, we use MetalLB (no specific Layer 7 requirements)
- Service IPs are BGP-announced with ECMP distribution (easy scaling)
- LoadBalancing only with K8S base algorithms or ingress controller features



Network configuration via Service for MetalLB

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    name: prometheus
  name: prometheus
  namespace: monitoring
spec:
  clusterIP: 100.72.112.153
  externalTrafficPolicy: Cluster
  type: LoadBalancer
  loadBalancerIP: 10.176.0.6
  ports:
  - name: prometheus
    port: 80
    protocol: TCP
    targetPort: 9090
  selector:
    app: prometheus
  sessionAffinity: ClientIP
  sessionAffinityConfig:
    clientIP:
      timeoutSeconds: 10800
```

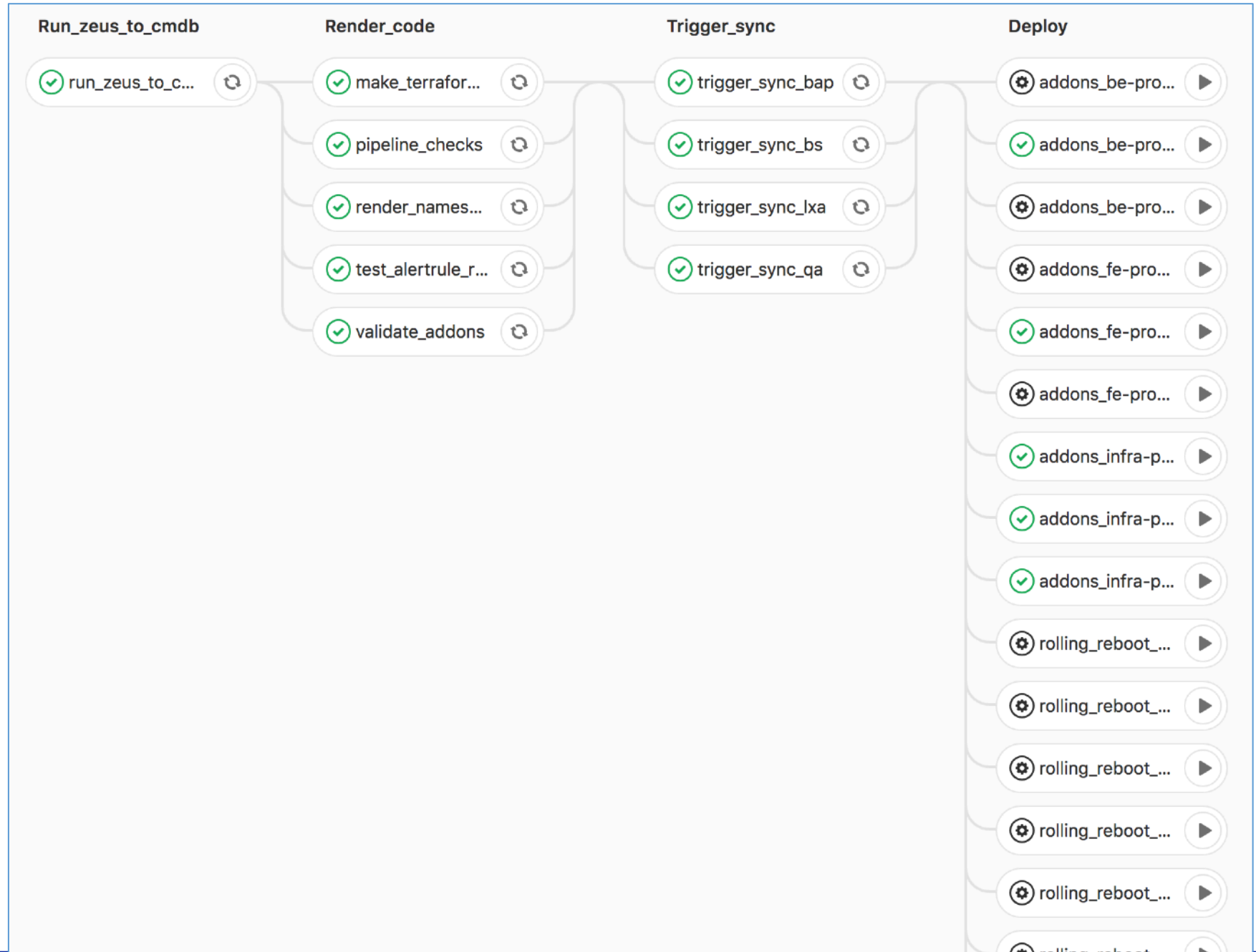
Git-driven cluster operations

- Maturity level via 3 branches (master, integration, production)

- All cluster operations are triggered based on Gitlab-CI pipelines
  - automatically on git-pushes to relevant branch
  - manually triggered jobs for cluster changes
  - scheduled jobs for periodic changes (namespace updates / purges)

Git-driven operations use cases

- Full redeployment of clusters
  - Only if cluster is broken, will wipe everything
  - Will redeploy all nodes in parallel
- Rolling upgrade of clusters
  - Usually done on a weekly basis
  - Will wipe and reset nodes one by one
- Namespaces update
  - Nightly updates for production
  - on-push for integration
- Addon update
  - Addons as helm charts, rendered via helm template and injected via kubectl apply
  - Done ad-hoc for addon changes without redeployment


Run_zeus_to_cmdb
Render_code
Trigger_sync
Deploy
run_zeus_to_c...
make_terrafor...
pipeline_checks
render_names...
test_alertrule_r...
validate_addons
trigger_sync_bap
trigger_sync_bs
trigger_sync_lxa
trigger_sync_qa
addons_be-pro...
addons_be-pro...
addons_be-pro...
addons_fe-pro...
addons_fe-pro...
addons_fe-pro...
addons_infra-p...
addons_infra-p...
addons_infra-p...
rolling_reboot_...
rolling_reboot_...
rolling_reboot_...
rolling_reboot_...
rolling_reboot_...

Multi Tenancy

- Common platform for several teams
  - PodSecurityPolicies (no-root, no host-net, r/o layers)
  - Dedicated resources for teams
    - Dedicated in-cluster prometheus for scraping
    - Configurable log-sink (Elasticsearch, Kafka)
  - Authentication via OIDC <-> Dex <-> LDAP

- Maximum separation between teams targeted
  - Namespaces are a „managed“ resource
  - Resource constraints defined centrally per namespace
  - Users are restricted to their namespaces via RBAC
  - Network policies
  - Team-centric „helper“ namespace
    - e.g. $team-helper
    - Used for managed resources, e.g. team-prometheus
  - Individual namespaces per (group of) application and stage
    - $team-$app-live, $team-$app-prelive

Multi Tenancy

- Dedicated namespaces for individuals
  - Purpose: Training, PoC, Experiments
  - Daily process to read users from LDAP and generate and flush namespaces
  - Service exposure via central ingress controller (traefik)

Namespace-Config via yaml

```yaml
.qa-ns: &qa-default-resources
  cpu_total: 20
  memory_total: 130Gi
  max_pod_size: L
.dev-ns: &dev-default-resources
  cpu_total: 10
  memory_total: 65Gi
  max_pod_size: S
team:
  ams:
    log_sink:
      address: "kafka1,kafka2,kafka3"
      type: "kafka"
    prometheus_vip: "10.176.0.7"
    app_namespaces:
      ams-tooling-qa: *qa-default-resources
      ams-testing-qa: *qa-default-resources
      ams-testing-dev: *dev-default-resources
    admins:
    - "ams-admins"
    - "ams-users"
personal_namespaces:
- cn=ams-users
- cn=ams-admin
- cn=other-users
cluster_name: "sandbox-intg-iz2-bap"
mam_dc: "bap"
activate_network_policies: true
```

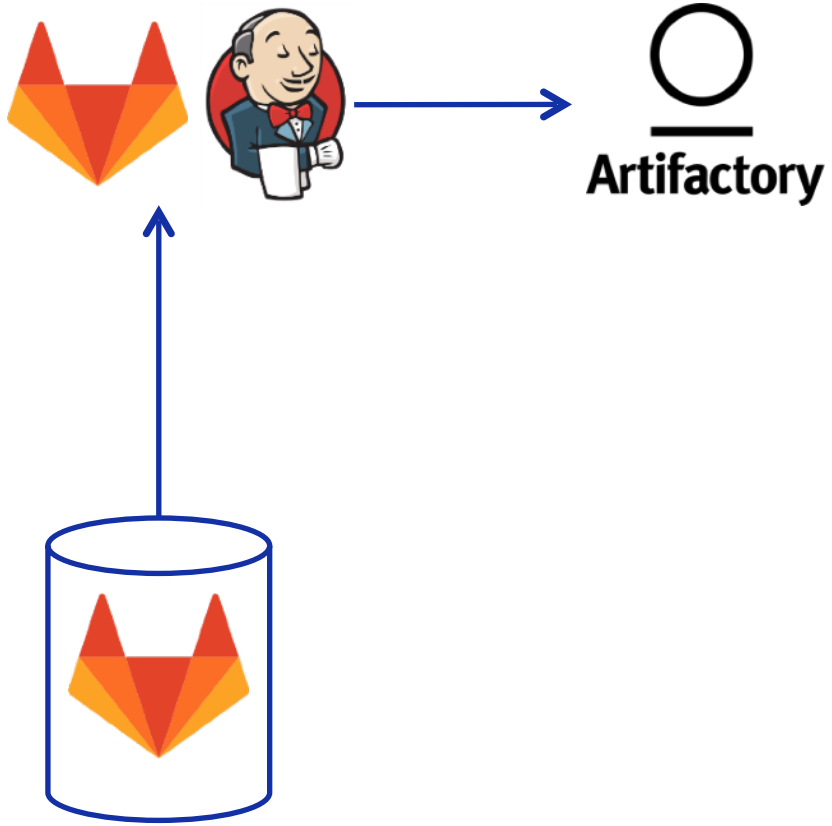
Rendered via helm

36 resulting manifests:

- 1 kind: Deployment
- 1 kind: Ingress
- 1 kind: Service
- 2 kind: ConfigMap
- 2 kind: ServiceAccount
- 4 kind: LimitRange
- 4 kind: Namespace
- 4 kind: ResourceQuota
- 8 kind: NetworkPolicy
- 9 kind: RoleBinding

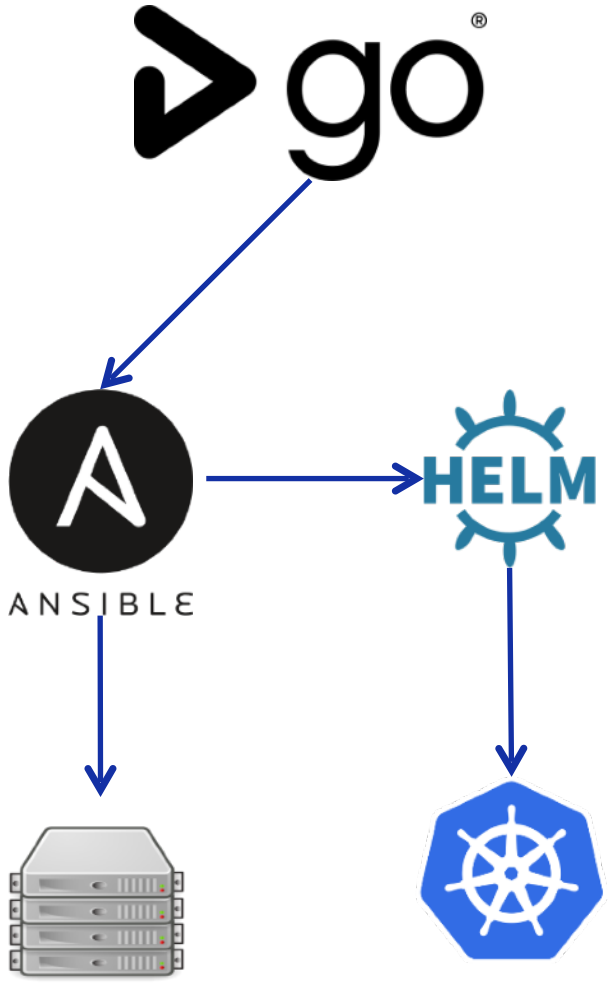
Build Processes

- Fully automated builds
- High degree of standardization
  - e.g. central maven POM
- Parallel builds for classical and container deployments
  - Containers use a centrally provided base image
- Build processes are triggered upon base image changes
- Policy: updates / rebuilds are enforced every 4 weeks

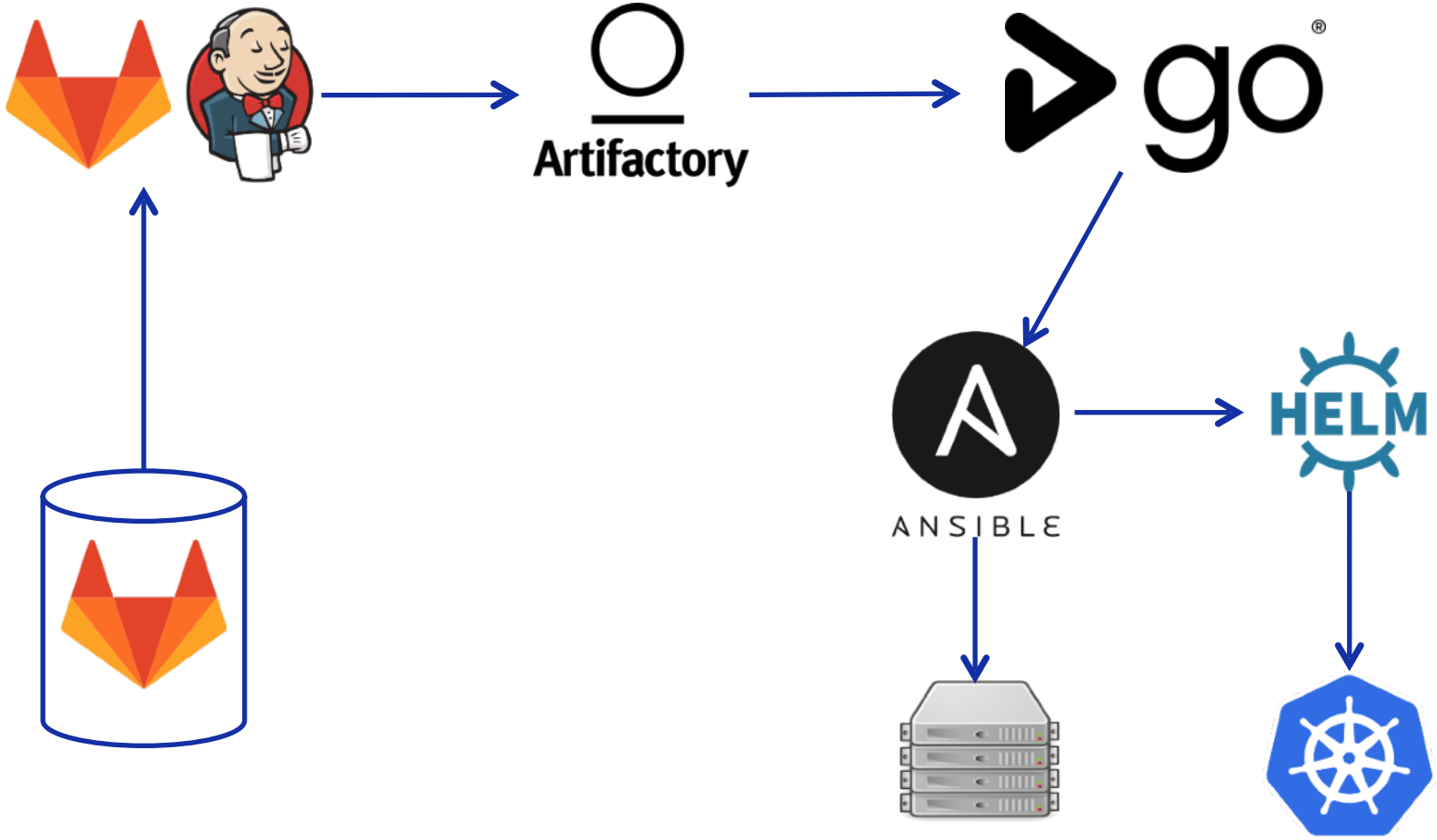


Continuous Delivery Environment

- GoCD maps business processes
  - Dedicated instance per team
  - Standardized pipeline templates
- Technical processes are mapped separately
  - Ansible for host based deployments
  - Helm/Kubectl for k8s deloyments
- Supports hybrid deployments
  - Container and Hosts in parallel
  - Hybrid usage via loadbalancer
  - Assists during transition phase



Fully automated deployment chain



Onboarding & Training

- 4 training blocks for system administrators (1-2 days each)
  - Docker & Kubernetes
  - GoCD & Helm
    - Pipeline Design
    - Helm Templating
  - Development Techniques for Ops
    - Repositories and versioning
    - Secure Software Development Lifecycle
  - Operating Container Applications
    - Monitoring, Logging and Failure Handling
    - Operations Lifecycle

Links

- F5-Ctrl (https://github.com/F5Networks/k8s-bigip-ctlr)
- MetalLB (https://metallb.universe.tf/)
- Dex (https://github.com/coreos/dex)
- GoCD (https://www.gocd.org)

- https://**jobs**.1und1.de/

- https://web.de
- https://www.gmx.net
- https://www.mail.com
- https://www.united-internet.de/