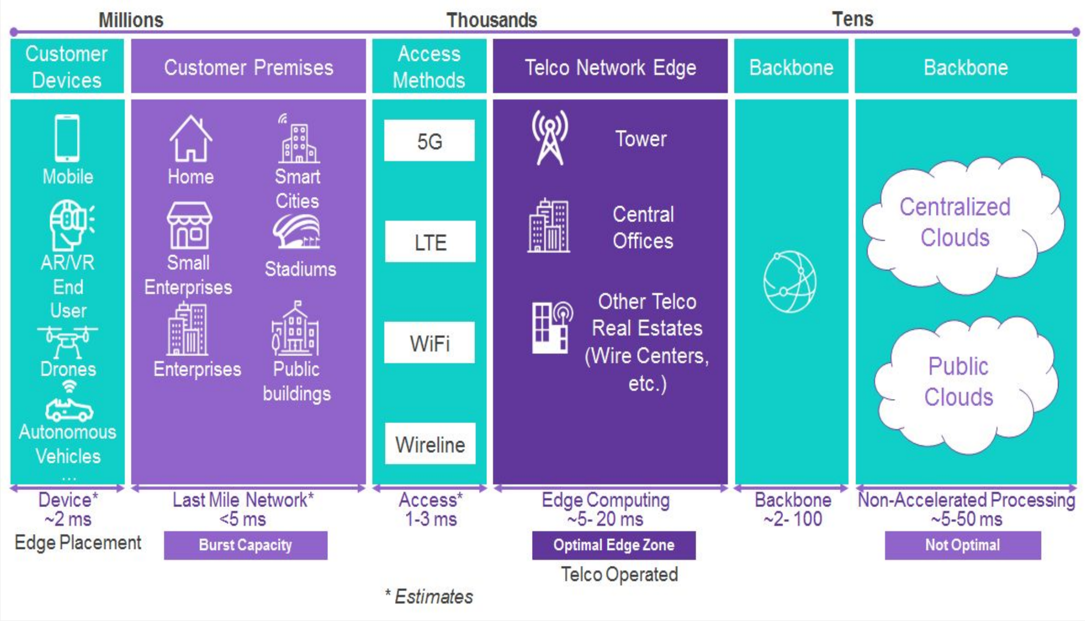
# Akraino Kubernetes-Native Infrastructure

A blueprint family for Edge

Yolanda Robla & Ricardo Noriega
Principal SW Engineer
CTO Networking team

# Edge Computing and Akraino

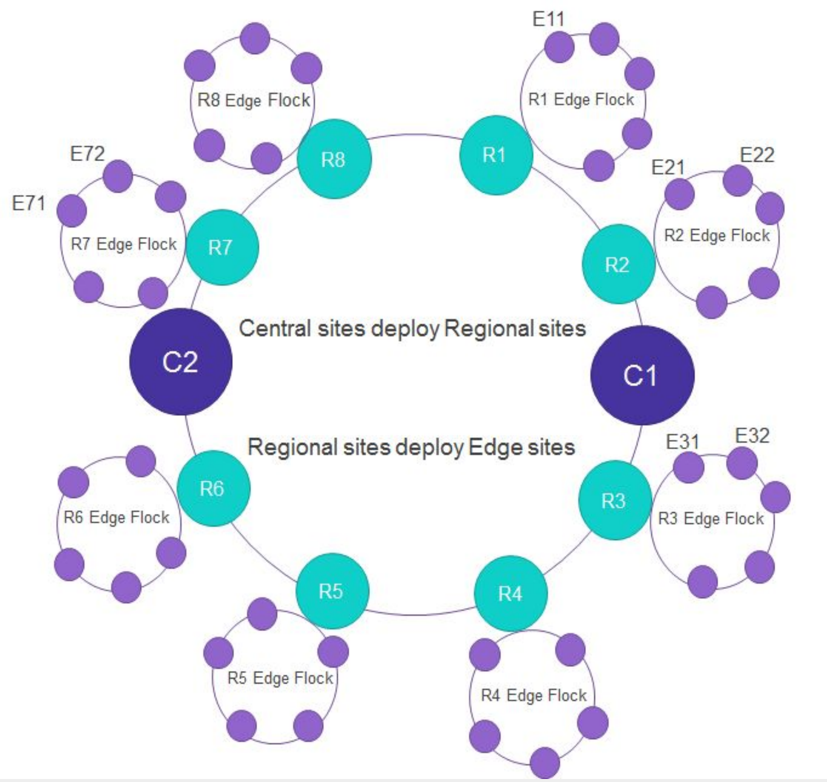
- Why **Edge computing**? The demand for real-time processing capabilities raises a need to place computing at the edge instead of relying on centralized processing. It brings processing and storage capabilities closer to the user endpoint, using the cloud. It reduces cost of ownership, enables faster processing, and meets specific data privacy rules.
- Akraino Edge Stack (https://wiki.akraino.org) is an open source software stack that improves the state of edge cloud infrastructure for carrier, provider and IoT networks. It belongs to LF edge org (https://www.lfedge.org/) , and is composed by more than 11 blueprint families, to support a variety of edge use cases (5G, AI/ML, Video Processing…)
- Community goals: faster edge innovation - end-to-end ecosystem (from hardware, to config, to apps) - improve user experience - provide seamless edge cloud interoperability - usage and improvement of open source



AKRAINO EDGE STACK

# Akraino Edge Stack and blueprints

- A typical service provider will have thousands of Edge sites: deployed at a cell tower, central offices… so end-to-end edge automation and zero-touch provisioning are required to minimize OPEX and meet requirements for agility
- For resiliency, the deployment follows a hierarchy of deployments: collection of central sites, regional sites and edge sites. This deployment is achieved using blueprints.
- Akraino is composed of multiple blueprints. A blueprint is a declarative configuration of an entire stack addressed for specific use cases (5G, AI/ML…), using a reference architecture developed by the community.
- A declarative configuration is used to define all components in the ref architecture: hardware, software, tools, method of deployment, etc..
- KNI (Kubernetes Native Infrastructure) is a family of blueprints inside Akraino, that leverages best-practices and tools of Kubernetes to declare edge stacks.



Red Hat

# Why “Kubernetes-Native Infrastructure”?

Kubernetes-managed infrastructure

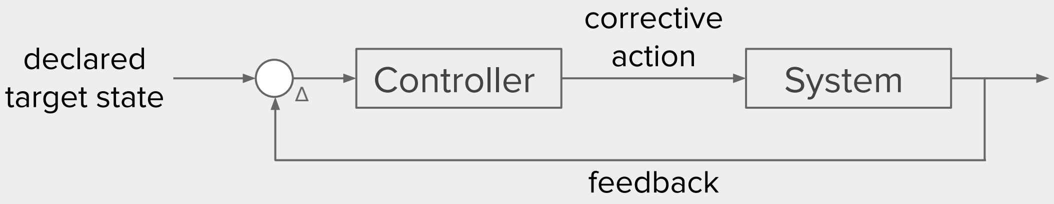
built on the rich tooling & best-practices of Kubernetes community & ecosystem

rich orchestration and lifecycle management

optimized for Kubernetes-native workloads, allows hybrid deploys with Kubevirt

# Declarative Edge Stacks with KNI

- Kubernetes is built around the **operator pattern**:
  - Operators capture the logic of a human operator managing services. A human that knows the logic of an app and knows how to take care of it. An operator is composed by a CRD and its controller
  - Controllers monitor a system for deviations between the *user-declared target state* and the reality and take corrective actions to reconciles reality with the declared target state.
- KNI-Edge applies this pattern **across the whole Edge Stack**:



declared target state

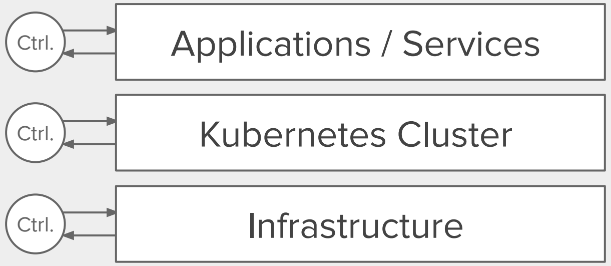
Δ

Controller

corrective action

System

feedback



Ctrl.

Applications / Services

Ctrl.

Kubernetes Cluster

Ctrl.

Infrastructure

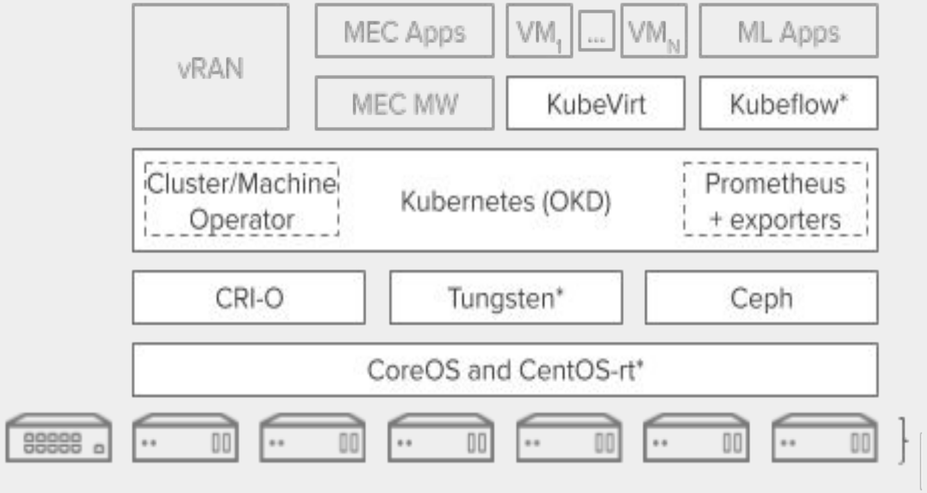
“I want Knative and Kubeflow deployed. Make it so!”

“I want Kubernetes upgraded to version 1.14. Make it so!”

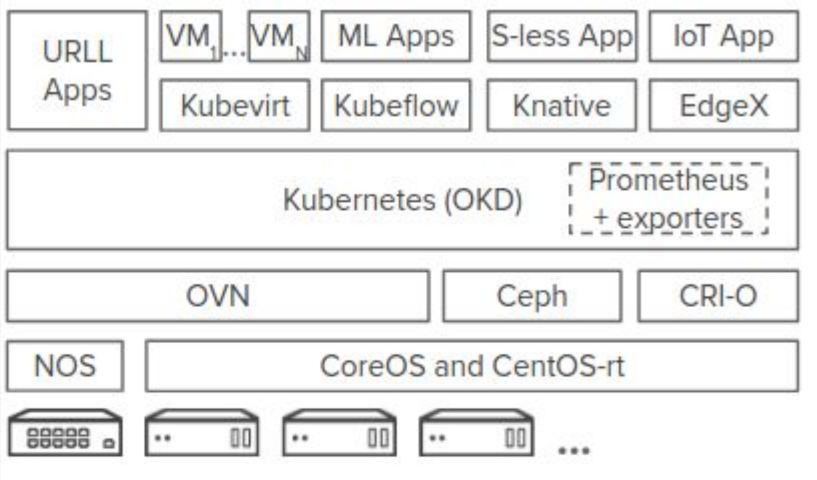
“I want 6 machines provisioned with CoreOS. Make it so!”

# KNI Blueprints in progress

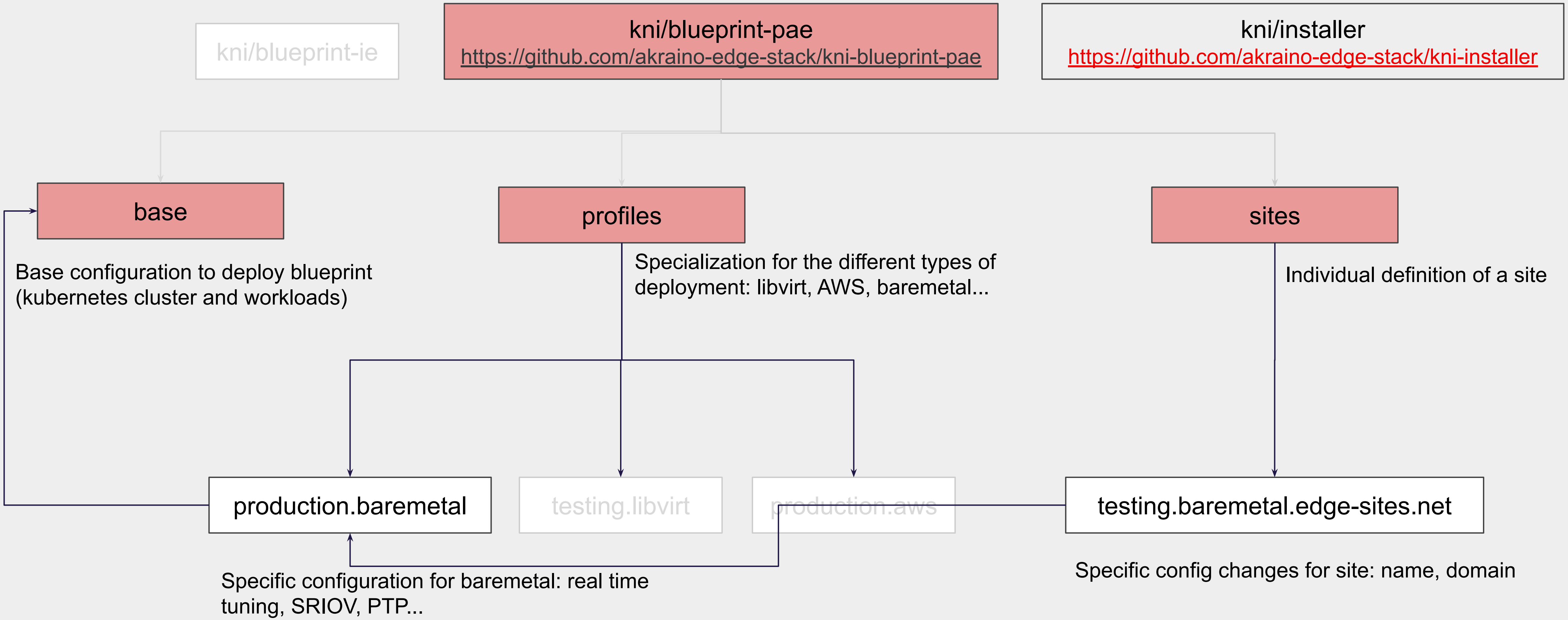
Provider Access Edge (PAE)
Optimized for real-time and networking performance for vRAN and MEC workloads.



Industrial Edge (IE)
Optimized for small footprint and low-latency for IoT, serverless, and machine learning workloads.



# KNI blueprint components



kni/blueprint-ie

kni/blueprint-pae
https://github.com/akraino-edge-stack/kni-blueprint-pae

kni/installer
https://github.com/akraino-edge-stack/kni-installer

base

Base configuration to deploy blueprint (kubernetes cluster and workloads)

profiles

Specialization for the different types of deployment: libvirt, AWS, baremetal...

sites

Individual definition of a site

production.baremetal

testing.libvirt

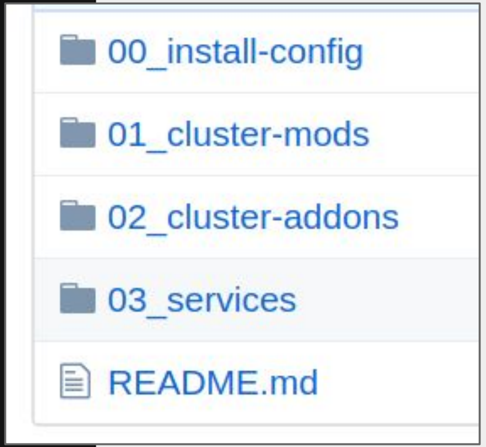
production.aws

testing.baremetal.edge-sites.net

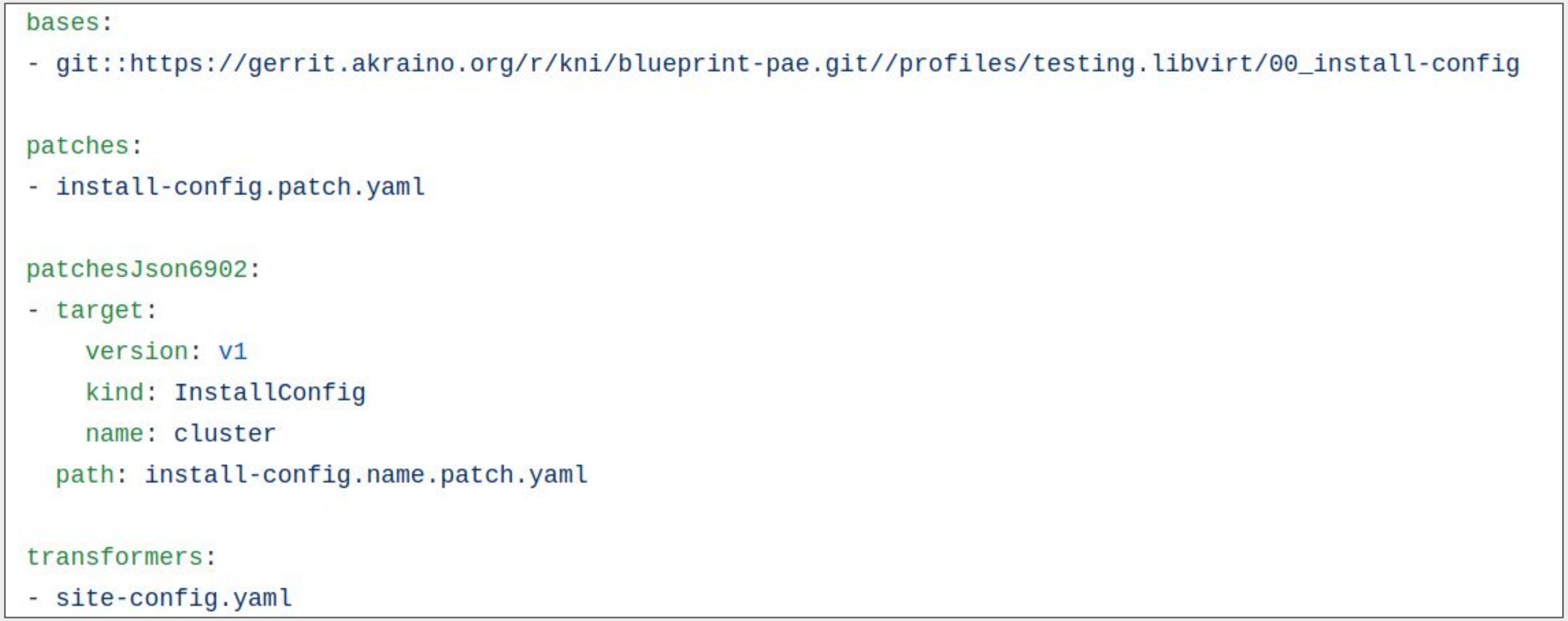
Specific configuration for baremetal: real time tuning, SRIOV, PTP...

Specific config changes for site: name, domain

# How a site looks like?

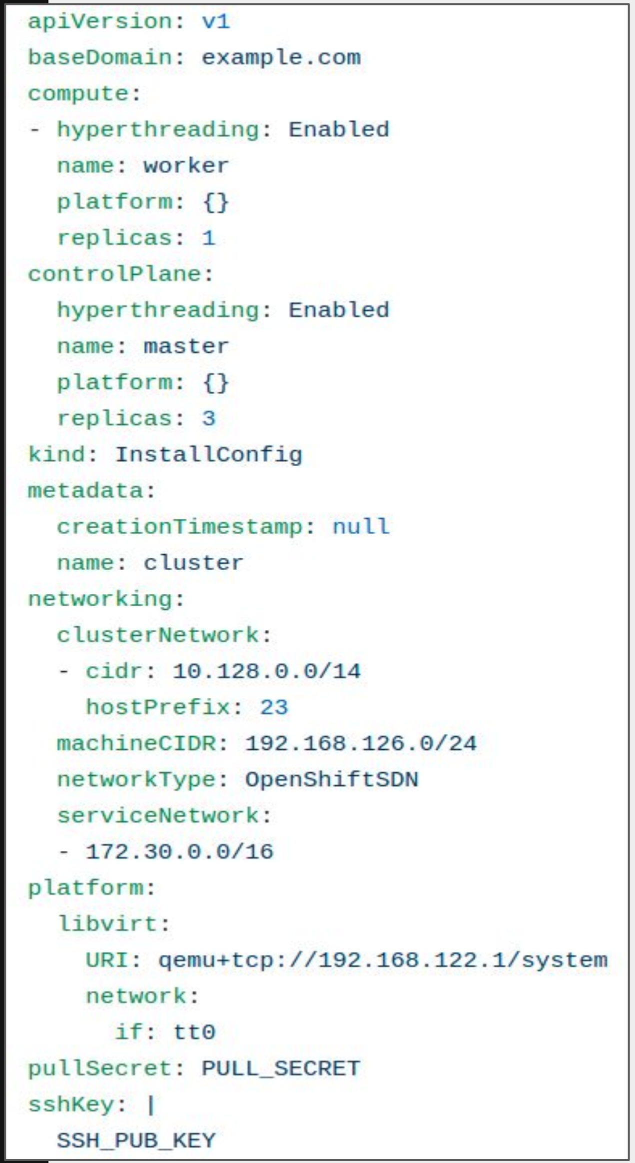


```
00_install-config
01_cluster-mods
02_cluster-addons
03_services
README.md
```



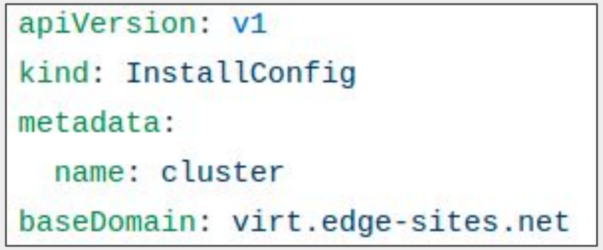
```
bases:
- git::https://gerrit.akraino.org/r/kni/blueprint-pae.git//profiles/testing.libvirt/00_install-config

patches:
- install-config.patch.yaml

patchesJson6902:
- target:
    version: v1
    kind: InstallConfig
    name: cluster
  path: install-config.name.patch.yaml

transformers:
- site-config.yaml
```

# How a site looks like?

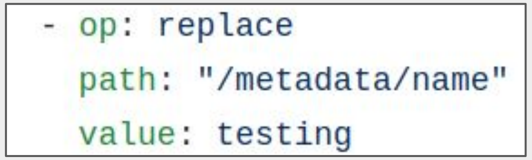


```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 1
controlPlane:
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
kind: InstallConfig
metadata:
  creationTimestamp: null
  name: cluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineCIDR: 192.168.126.0/24
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  libvirt:
    URI: qemu+tcp://192.168.122.1/system
    network:
      if: tt0
pullSecret: PULL_SECRET
sshKey: |
  SSH_PUB_KEY
```

install-config.patch.yaml



```
apiVersion: v1
kind: InstallConfig
metadata:
  name: cluster
baseDomain: virt.edge-sites.net
```



```
- op: replace
  path: "/metadata/name"
  value: testing
```

install-config.name.patch.yaml

testing.virt.edge-sites.net

# Demo of Declarative Infrastructure Management

Edgy: edge network operator

Download knictl

Create site

Fetch requirements

Prepare manifests

Deploy site

Apply workloads

Happy Edgy

# Demo time!