# Kubernetes Rebar Integration “Cloud-Native Bare Metal”

Zero to K8s: simple-to-use, fast-to-start and completely self-contained

# Objective: Zero Touch Install & Upgrade

To encourage broader adoption of Kubernetes and support an ecosystem, we need highly streamlined on-premises and colo bare metal implementations that:

1. Deliver a self-contained Zero to Kubernetes experience
2. Align with cloud-native architecture
3. Fit a wide range of infrastructure models starting with bare metal
4. Minimize complexity for operators
5. Highly-distributed, zero-touch Telco/Edge use cases.

# Mission 1: Share Community Practices

**Installations should not be snowflakes!**

We built Digital Rebar for platform communities such as Kubernetes to evolve and share best practice automation.

We felt that simple, API-driven and complete underlay automation could provide the missing physical reference layer so that operational scripts could be portable.

*We never expect operators to use identical hardware or configurations.*

# Mission 2: Smallest Possible Footprint

**Complexity is the enemy of scale!**

We built Digital Rebar Provision for platform to minimize the footprint of provisioning because bootstrapping needs to work and get out of the way.

To accomplish our collaboration mission, operations must be able to easily understanding and use the system in a wide range of environments.  Simple extension points enable mapping to diverse systems of record.

*We improve usability by reducing complexity for operators*

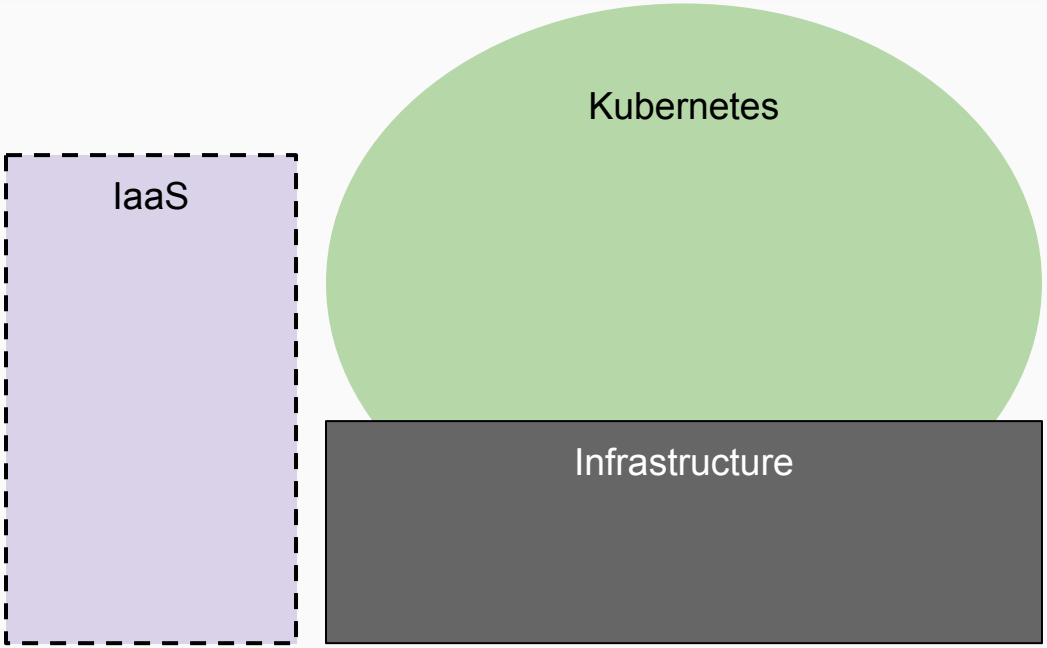
# Mission 3: Self-Contained System

**Integrated Provisioning Workflow avoids a “domino chain” of tools**

Provisioning hardware is different than VMs because there are multiple protocols, APIs, control layers and vendor specifications that must be coordinated to provide full life-cycle control.

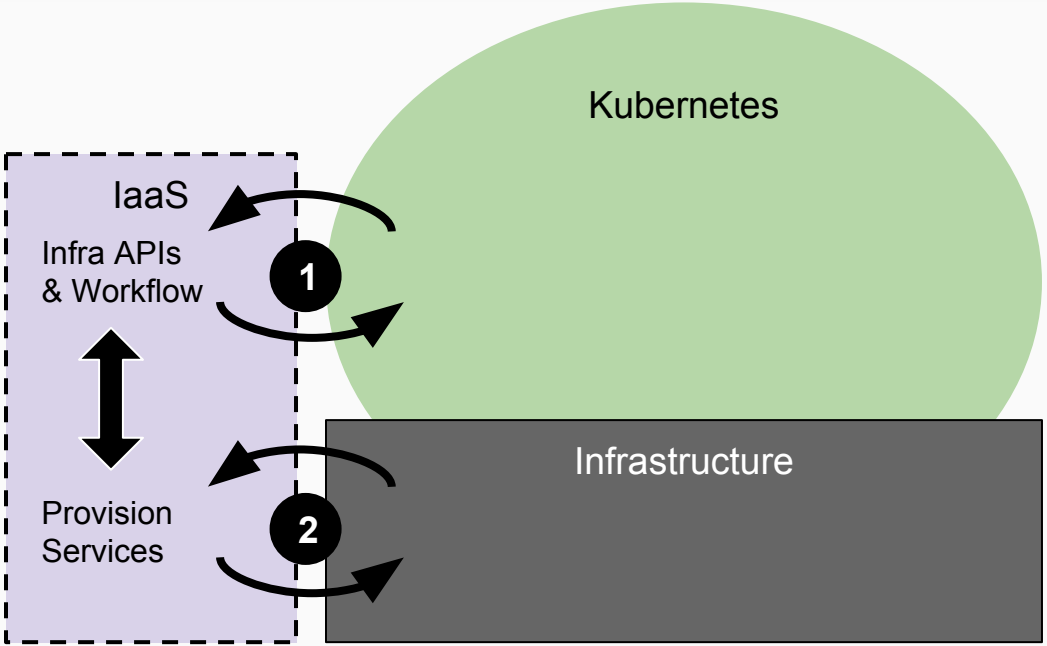
We built Digital Rebar Provision to smoothly coordinate activities across all these control surfaces and over multiple system reboots.  This unique workflow capability allows the platform to be completely self-contained for bootstrapping.

# Simple Kubernetes Reference Architecture




Starting from the most basic components, we use API driven infrastructure (IaaS) to build *and maintain* a Kubernetes cluster on general purpose infrastructure.

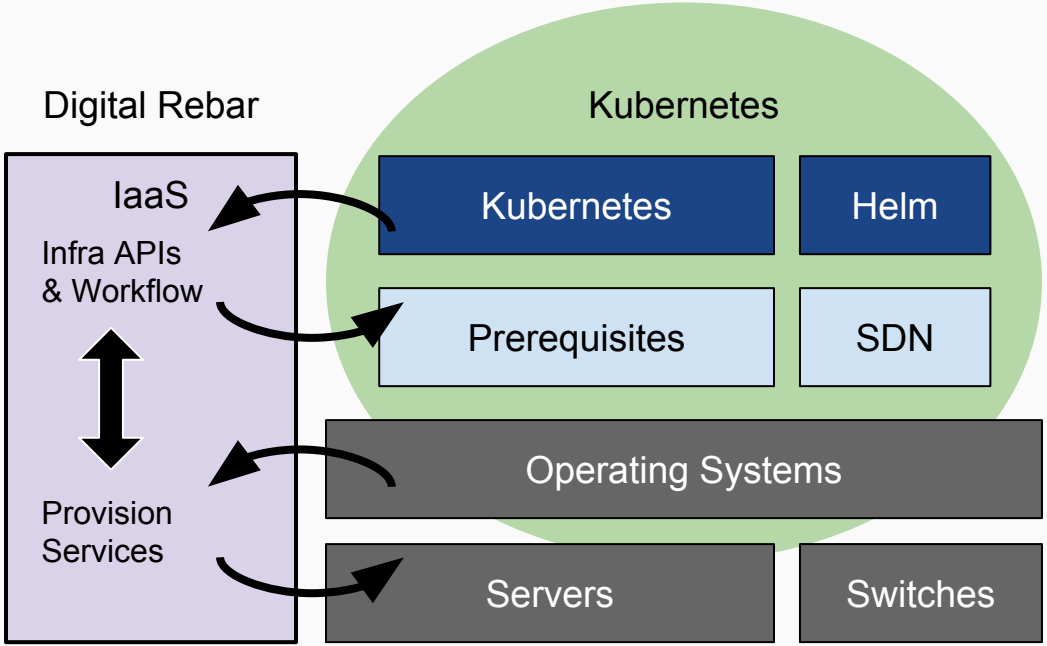
# Simple Stack - Kubernetes and IaaS




1) The ongoing state of the Kubernetes cluster should be incorporated into the IaaS control functions.

2) The IaaS should have sufficient internal state to manage and report all infrastructure operations.

# Simple Stack - Digital Rebar as the IaaS




Digital Rebar provides all the infrastructure controls and workflow required to build and integrate with a running Kubernetes cluster.

Unlike a Cloud IaaS, the system also has to build required support services.

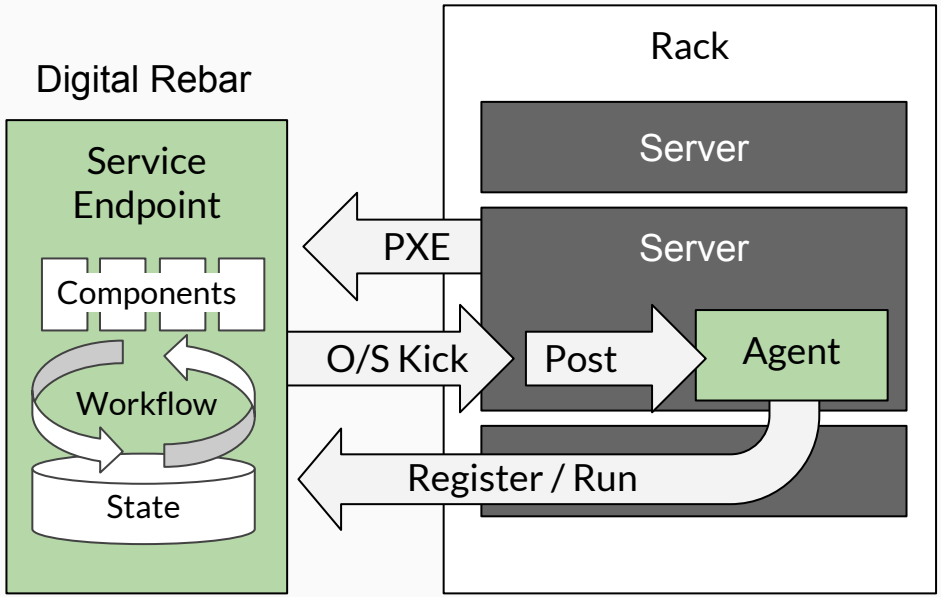
# What is Digital Rebar (DR)?

APLv2 Digital Rebar (DR), https://rebar.digital, provides a Golang API-driven infrastructure service for building and maintaining the Kubernetes bare metal environment.

DR includes a complete netboot (DHCP/PXE/iPXE) platform without any prerequisites that is able to take systems from initial boot discovery, inventory and validation, o/s installation and post-boot configuration.

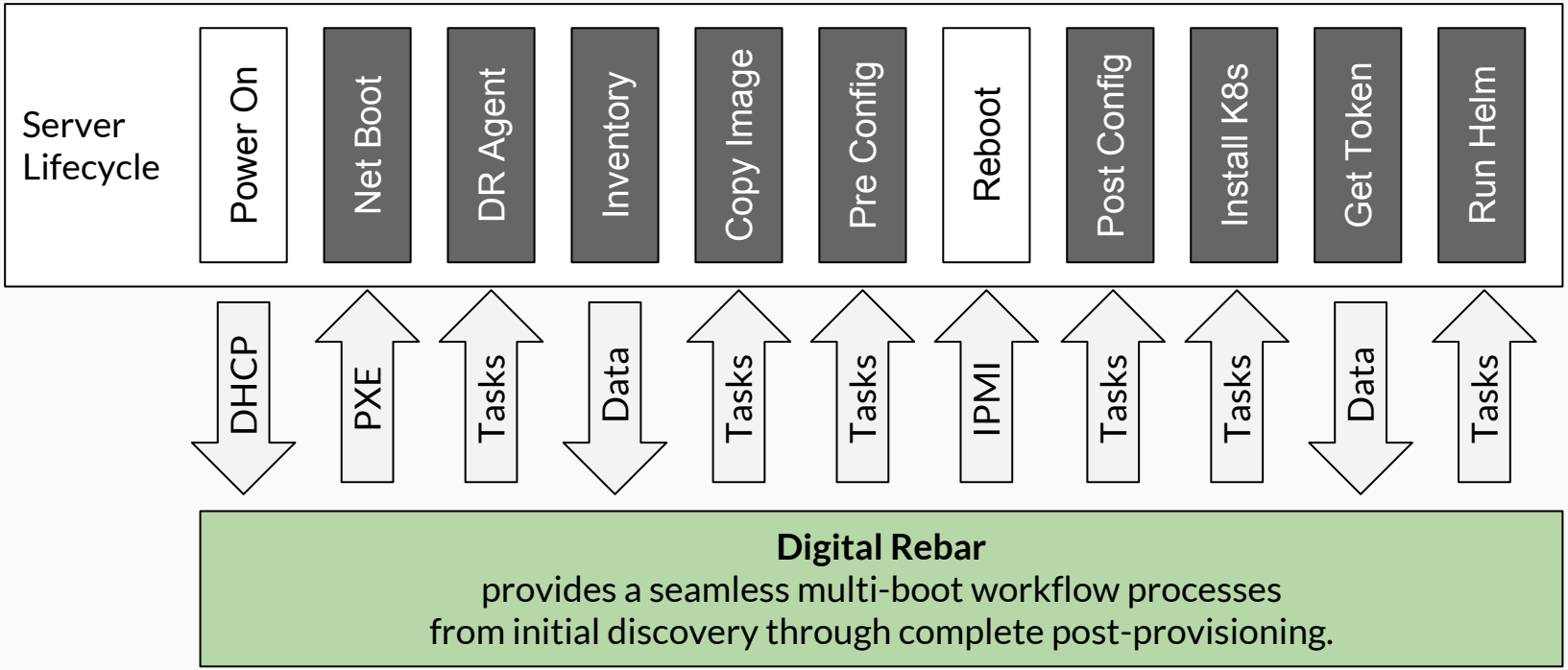
DR also offers an extensible, intent-based workflow for external systems to act on the infrastructure in a simple, controlled way.

# Digital Rebar Architecture




Digital Rebar offers an intent-based REST API that allows external systems and Kubernetes to continue to act on the infrastructure in a controlled way over multiple provisioning cycles. That control allows integrated on-going cluster maintenance.

# Digital Rebar
# Seamless, Multi-Boot Workflow



# Advanced Features

**Open Source Digital Rebar**

- Netboot Open Operating Systems
- Multiple Kubernetes Options
- HA Kubernetes (KRIB)
- Ansible Integration (Kubespray)
- Terraform Bare Metal Plug-In
- TLS Certificate Plugin
- DHCP / UEFI & Legacy Bios

**RackN Additions:**

- Image-Based Deployments
- Ansible Tower Integration
- RAID/BIOS Configuration
- ESXi & Windows OS Provision
- Multi-tenant and RBAC
- Highly Available Configurations

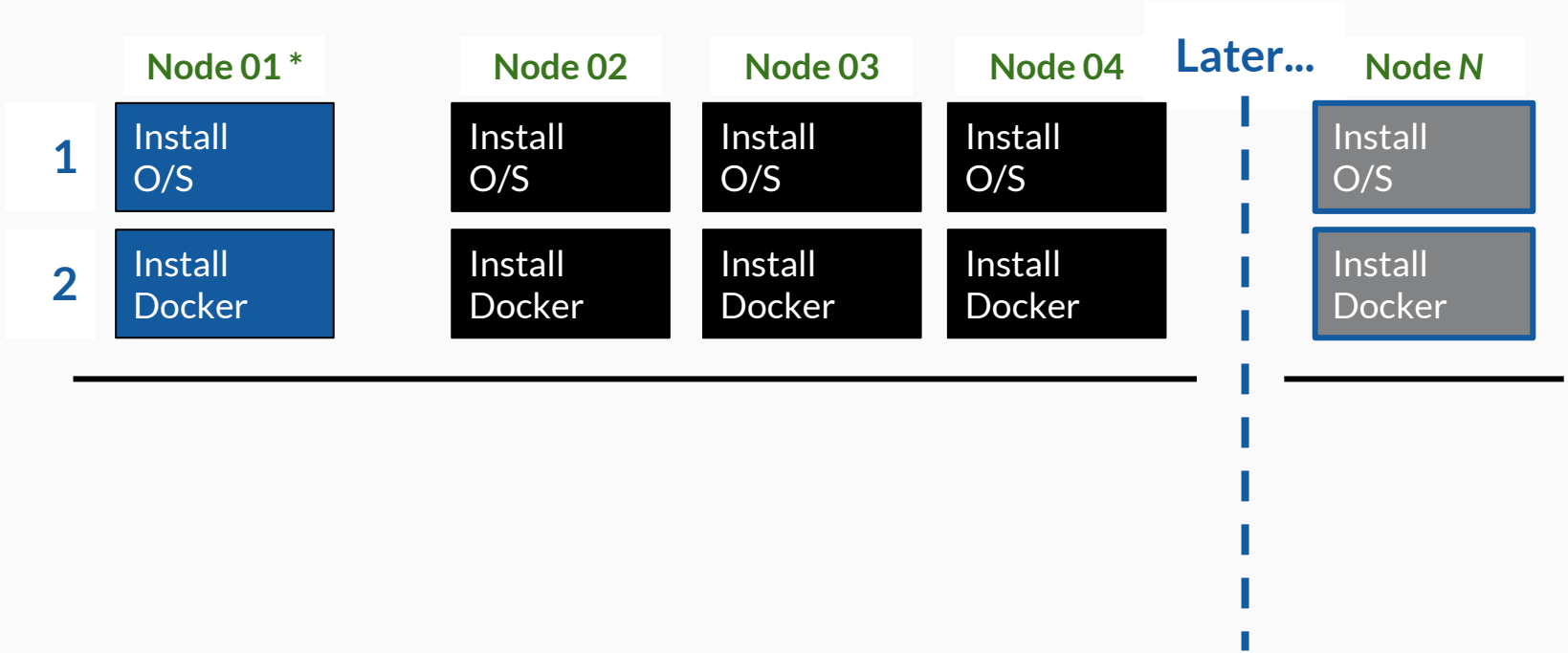
# Self-Bootstrapping Kubernetes with DR

Using open source tooling allow bare metal machines to go from initial boot to a fully functioning Kubernetes cluster *without human guidance*.

Requires:

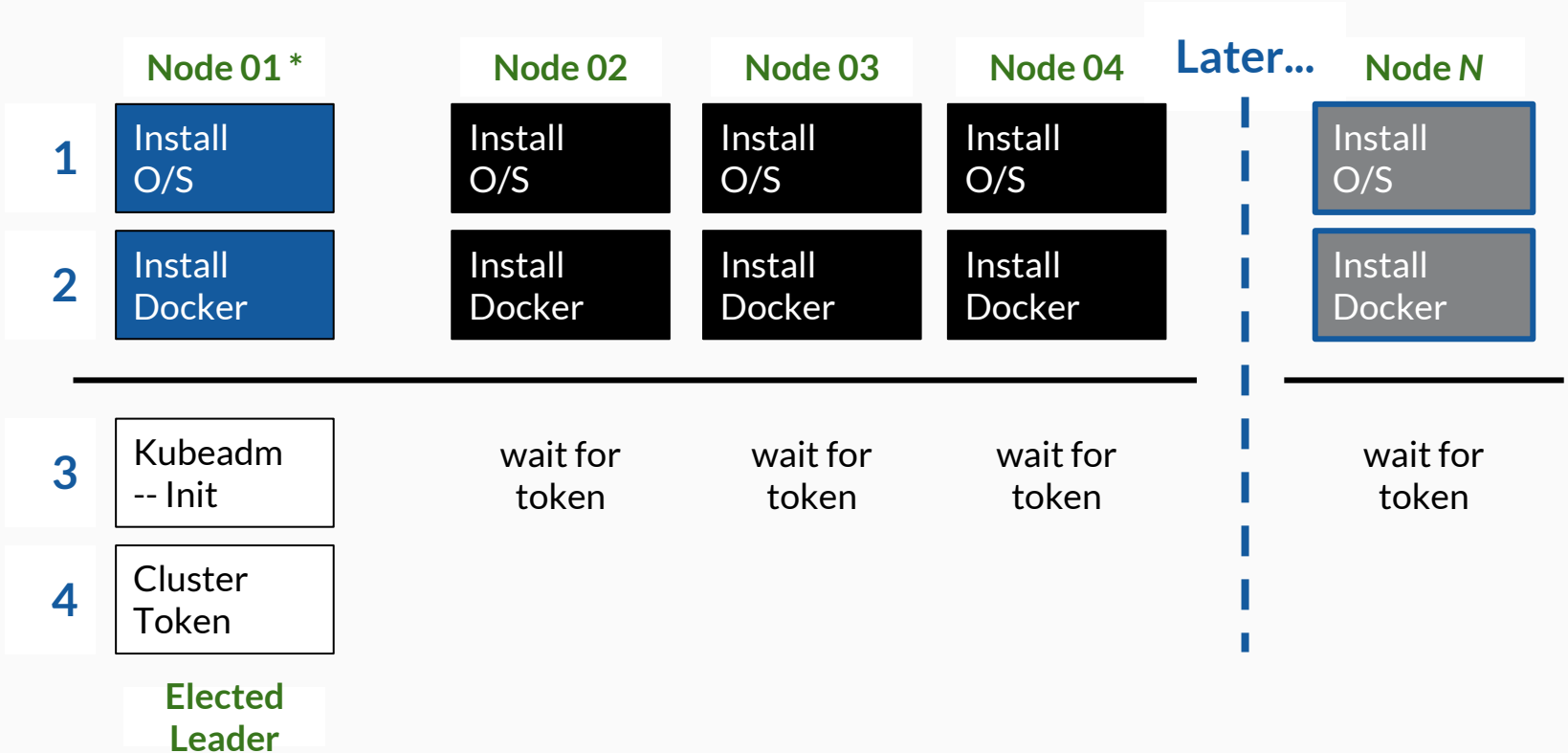
- Automatic server discovery, inventory, validation and provisioning
- Elect leader(s) in the installation process to generate tokens
- Sequence work (aka workflow) operations across the cluster
- Store and manage installation artifacts as created
- Provide secure external status about cluster status

# Kubernetes Rebar Integrated Bootstrapping Workflow (using Kubeadm)

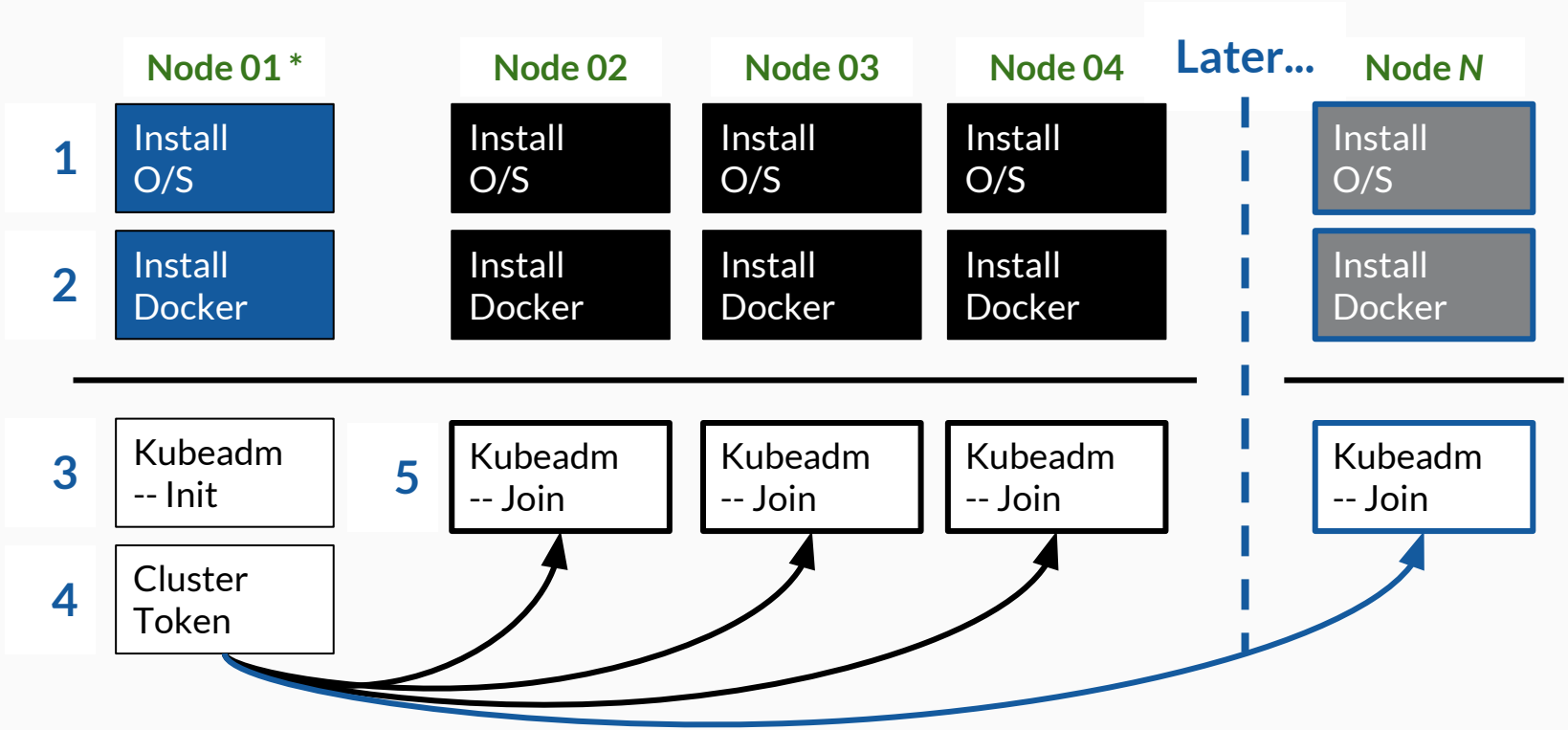



* Node 01 is wins Kubeadm init election (can also be preassigned)

# Kubernetes Rebar Integrated Bootstrapping Workflow (using Kubeadm)



# Kubernetes Rebar Integrated Bootstrapping Workflow (using Kubeadm)




* Node 01 wins Kubeadm init election (can also be preassigned) - multiple HA Masters are supported