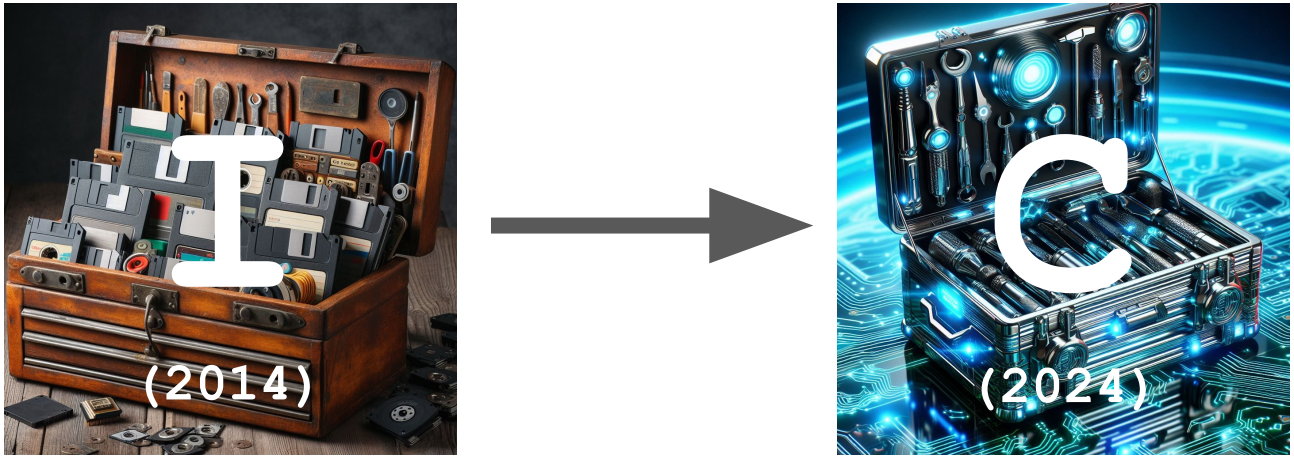
Digging through the toolbox

Ten years of Ironic features

Introduction

Is your toolbox rusty?

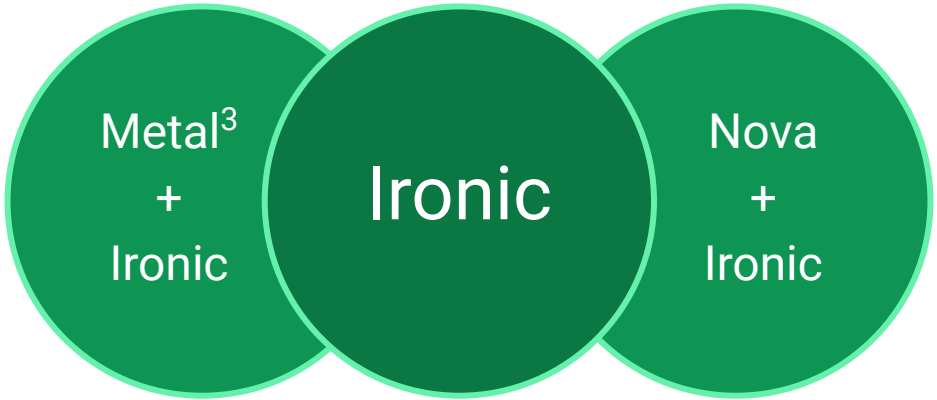
Feature availability depends on the version of Ironic you run



Using a modern version of Ironic is the best path to succeed

What style of toolbox are you using?

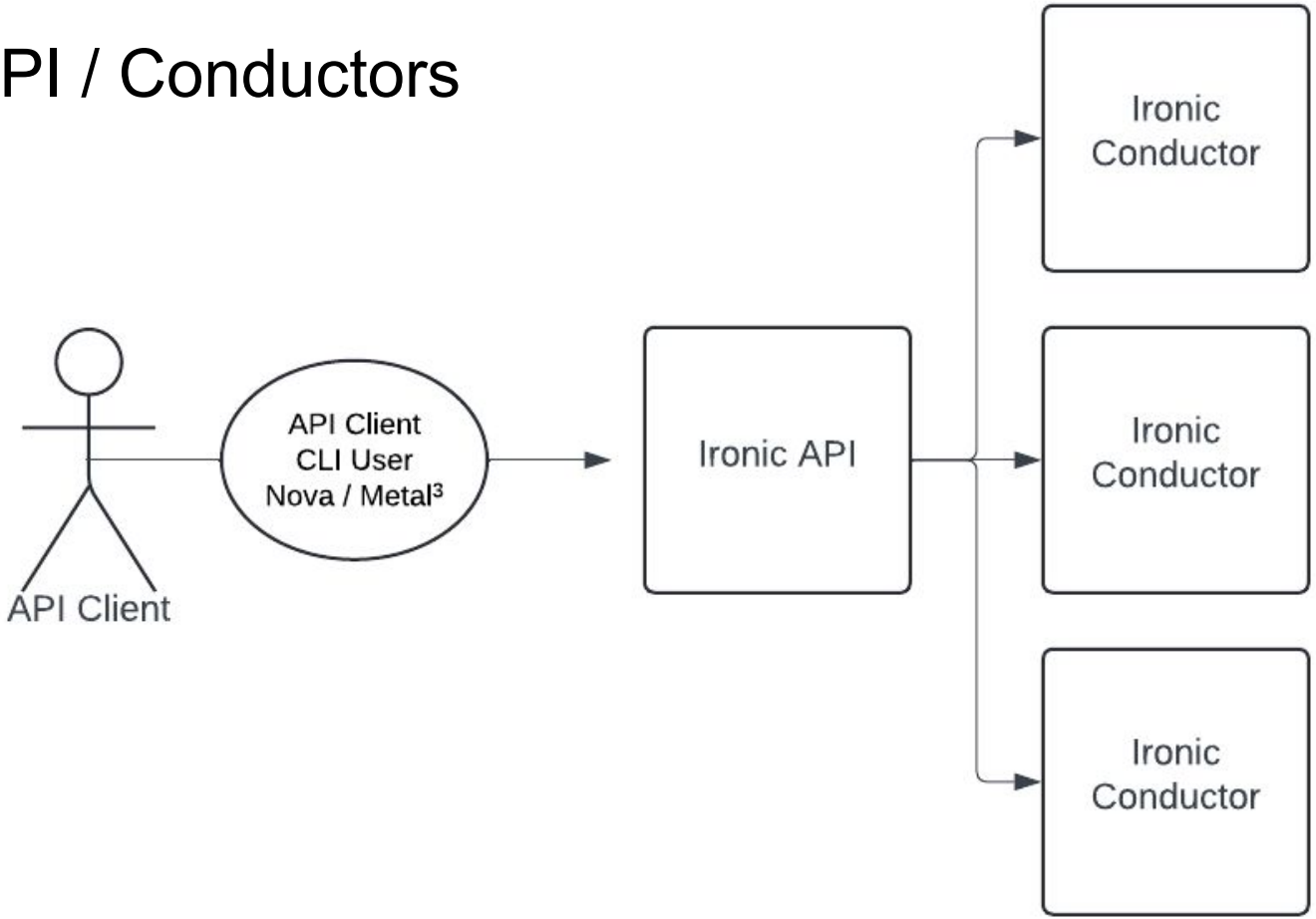
Not all Ironic features can be used in all contexts



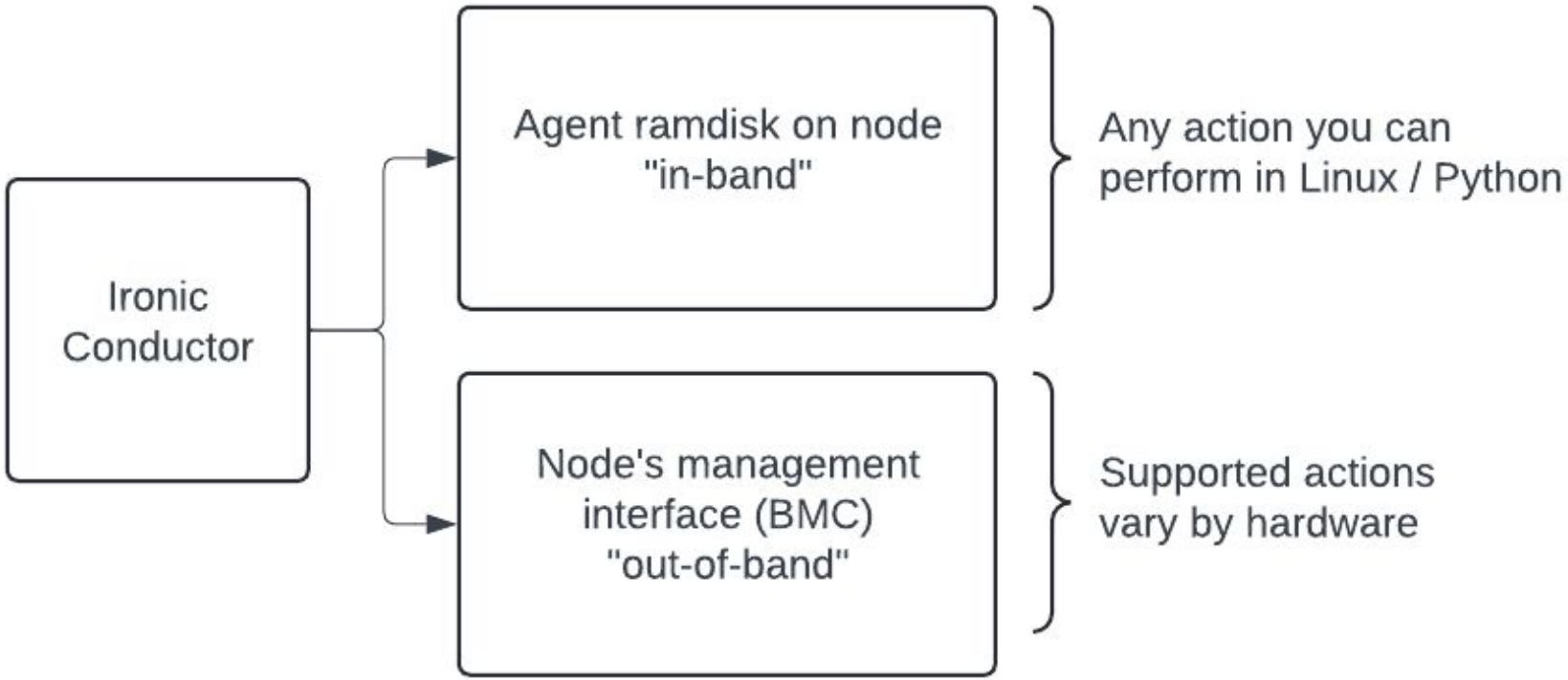
You'll need direct access to an Ironic API to utilize all these tools.

Basic Architecture

Ironic API / Conductors



Node Control



Preferred Hardware

- Best used with servers with a fully featured BMC
- Redfish is an HTTP API standard for BMCs

Ironic can do the most with redfish-supporting hardware

Hardware Types & Drivers

Hardware Types

- Ironic adapts to different server configurations via **hardware type**.
- A hardware type is a collection of interfaces for specific hardware
- Interfaces are pluggable implementations of Ironic functionality, often dependent on hardware capabilities

Interfaces

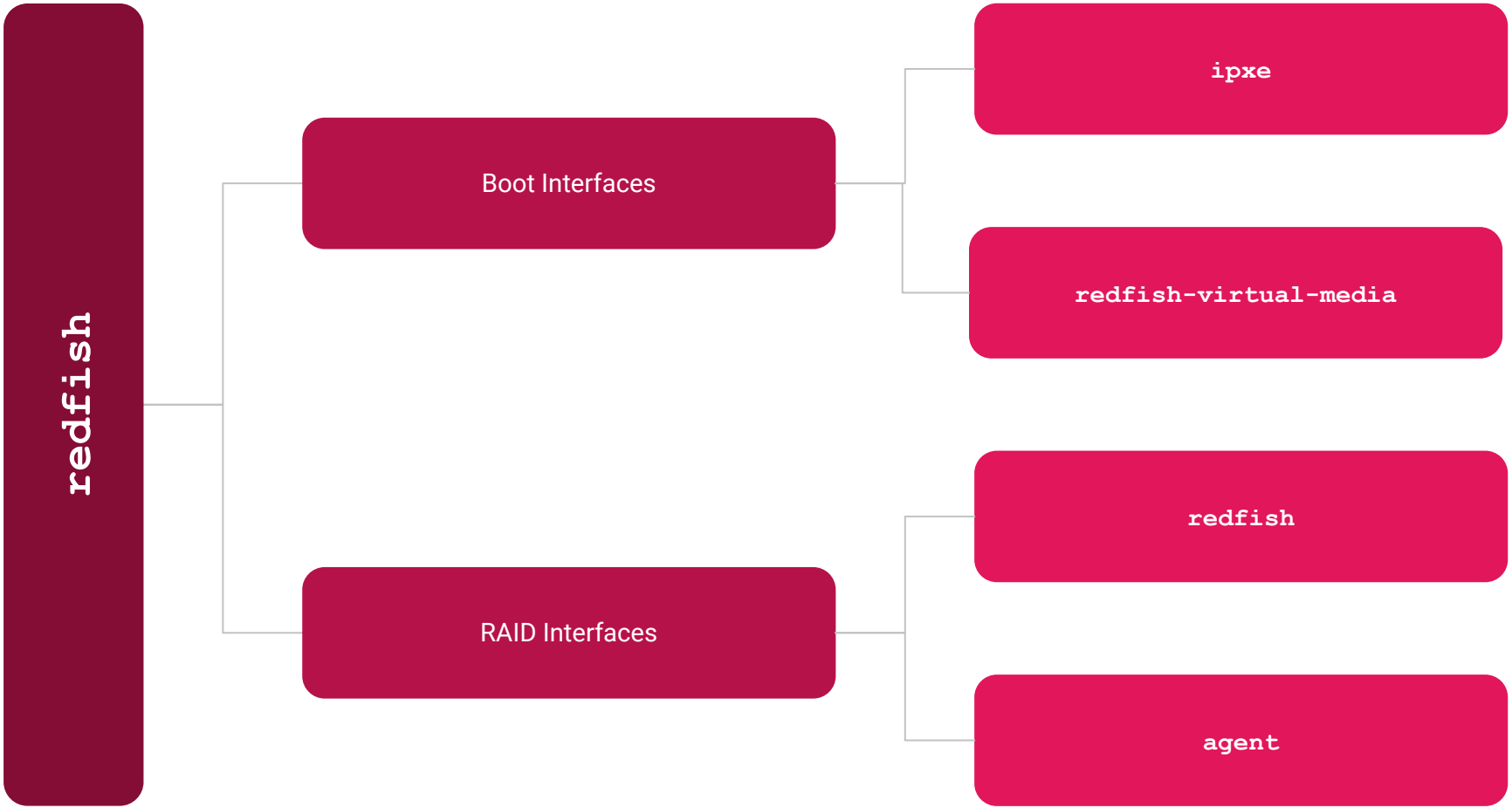
Interface	Use Case	Example(s)
bios	Create/read/update BIOS settings	`redfish`
boot	Methods for booting workloads	`pxe, ipxe, http, http-ipxe, redfish-virtual-media`
console	Provides console access	`ilo, ipmitool-socat`
deploy	Methods for deploying onto a server	`direct, anaconda, ramdisk`
firmware	Report version and update firmware	`redfish`
inspect	Inspect and report hardware details	`agent, inspector, redfish`

More interfaces…

Interface	Use Case	Example(s)
management	Configure next boot device	`redfish, ipmitool`
network	Orchestrate network configuration	`neutron, flat, noop`
power	Server power control	`redfish, ipmitool, agent, snmp`
raid	Configure RAID arrays	`agent, redfish`
rescue	Provides rescue environment	`agent`
storage	Support for attaching volumes	`cinder, external`

Interface Options

- A hardware type can support multiple options for the same interface
- Overriding the default interface for a type can be done in config or per node
- Many interfaces may require special configurations or hardware to function


redfish
Boot Interfaces
ipxe
redfish-virtual-media
RAID Interfaces
redfish
agent

Server Lifecycle

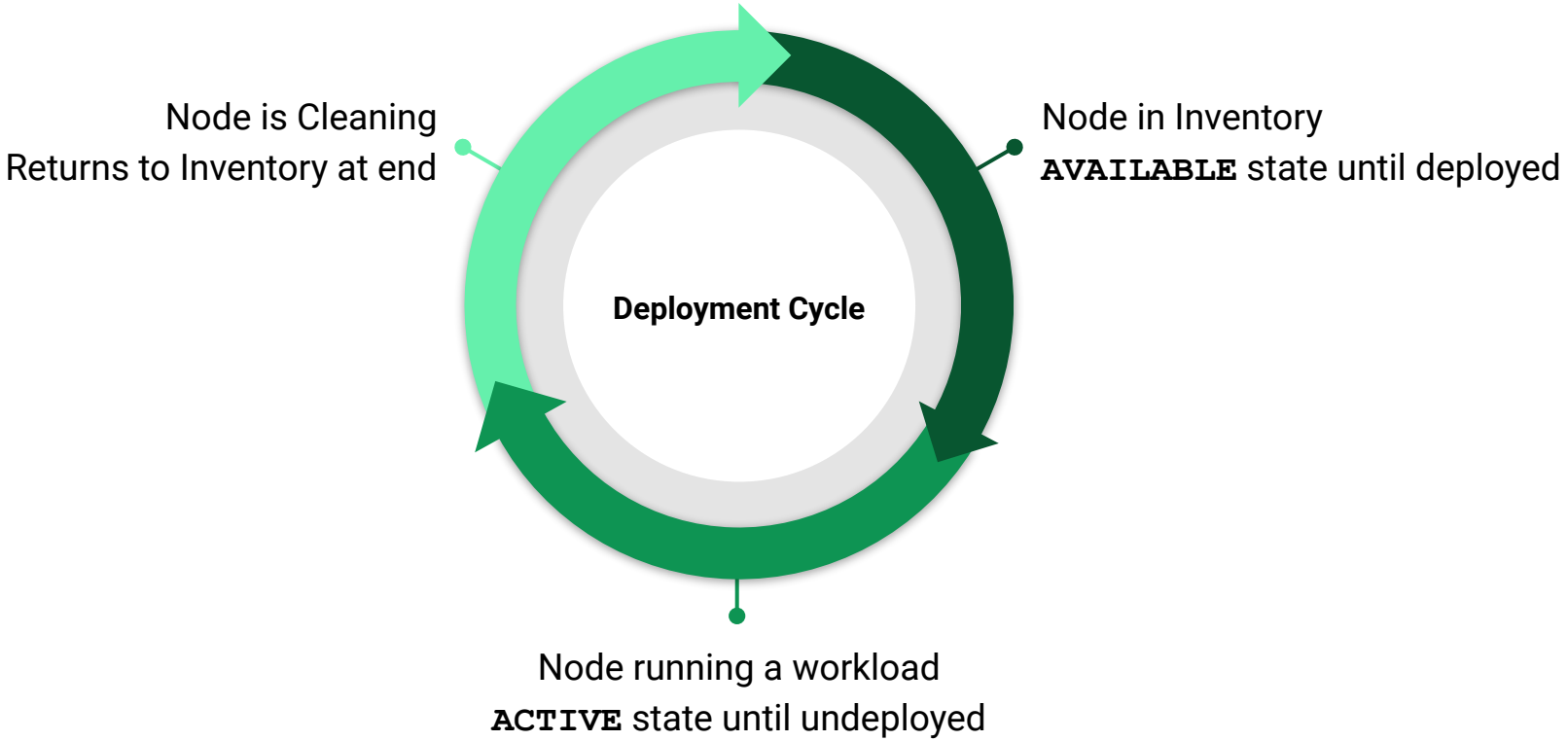
Node Onboarding

- Enroll (2015)
  - Add basic node information to Ironic
- Validate (2015)
  - Ensures required keys are set
  - Verify steps (2021) were added to ensure the server is ready for Ironic to provision
- Inspect (2016)
  - Inventory hardware on server

Node Inspection

- Uses interface-dictated method to get information about hardware
- Information can be saved to DB or a separate data store
- Extremely customizable

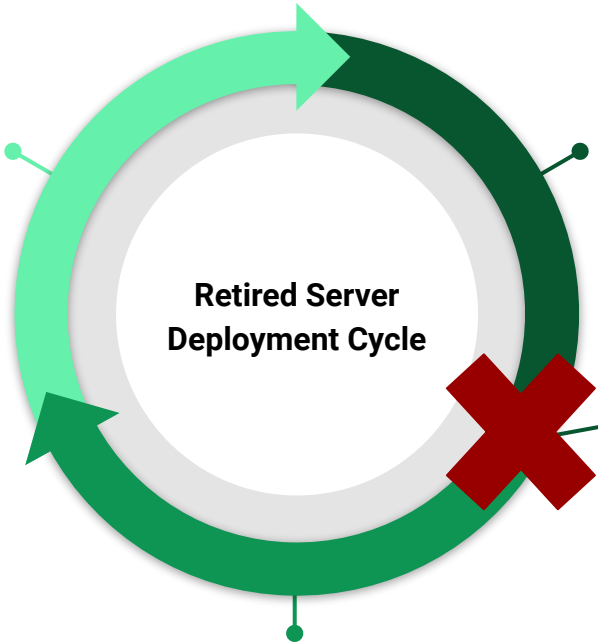
Node Deployment & Automated Cleaning (2015)



Node Retirement (2022)

Use node retirement to remove EOL servers from your cloud without disruption

Node Retirement (2022)



Node is Cleaning
Not returned to inventory,
left in **`MANAGEABLE`** state

Node ready for deletion
In **`MANAGEABLE`** state

Node cannot be deployed
While marked as retired

Node running a workload
(while set as retired)
`ACTIVE` state until undeployed

Steps

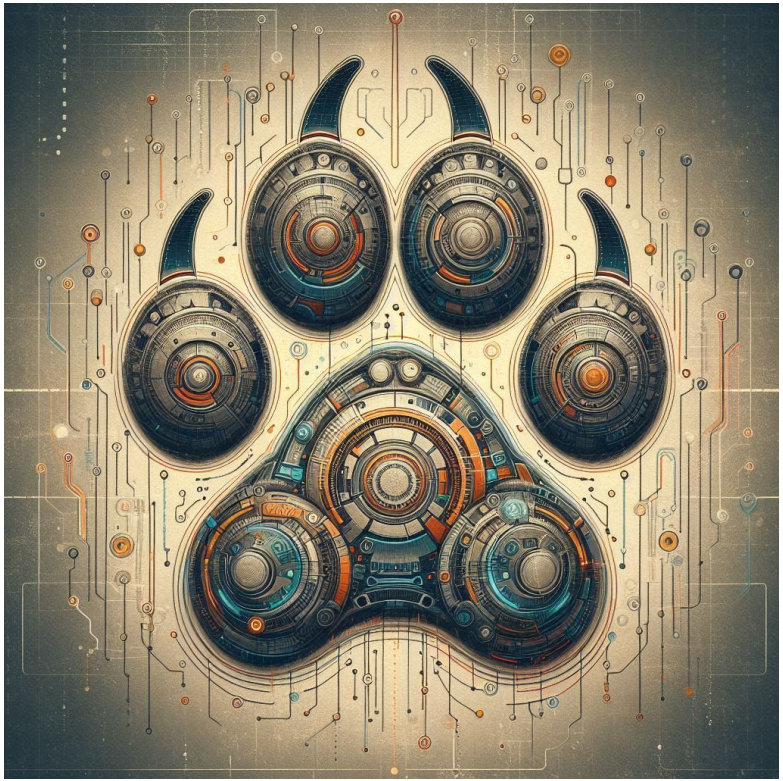
What are steps?

A step is a basic unit of work in Ironic.

A step consists of:

- Name
- Interface
- Priority (0 means disabled)
- Arguments (optional)

Some steps are only usable in some workflows.



Types of Steps

In band (2015)

- Utilizes Ironic-Python-Agent ramdisk to perform tasks
- Easy to add / customize steps
- Can embed vendor utilities into ramdisks

Types of Steps

Out of Band (2016)

- Performs task via interactions with BMCs
- Steps can come from many different interfaces
- Not designed for significant customization

Types of Steps

Control Flow (2023)

- Power control
- Wait for a specified period of time
- Hold until API called to unhold

Example built-in steps

Step Name	Interface	What it does
`erase_devices` (2015)	Agent (IB)	Erases block devices
`update_firmware` (2023)	Firmware (OOB)	Updates firmware
`apply_configuration` (2019)	Bios (OOB)	Applies a given bios configuration
`burnin_disk`, **`burnin_cpu`**, **`burnin_memory`**, (2021)	Agent (IB)	Performs a burn-in test on the indicated component
`create_configuration` (2015)	Raid (OOB *and* IB)	Creates a RAID configuration

Custom steps in IPA

Copy an Example	Add your step	Embed into ramdisk	Use New Ramdisk
Copy the most appropriate Hardware Manager example out of IPA's examples/ folder and use it to create a new python repo/package.	Following instruction and comments in the example, implement your step in python. It can be as simple as shelling out and checking an exit code.	Now, when building IPA ramdisk, install your new Hardware Manager package alongside IPA. Also ensure any added dependencies – such as vendor tools are built into the ramdisk.	Using configuration or API calls, configure Ironic to use your new ramdisk.

Step-based Workflows

Automated Cleaning (2015)

- Automated cleaning occurs by default when a node is deleted.
- By default, only contains in-band steps to clean disks.
- Additional steps can be added to automated cleaning by:
  - Using a custom hardware manager
  - Using [conductor]clean_step_priority_override to increase priority of disabled steps
- Only steps with no required arguments can be used

Node Servicing (2024) & Manual Cleaning (2016)

- Pass a list of steps to execute on a node
- Manual cleaning is designed for undeployed nodes
  - `MANAGEABLE -> CLEANING -> MANAGEABLE`
- Node servicing is designed for deployed nodes
  - `ACTIVE -> SERVICING -> ACTIVE`

Deploying with Steps (2018)

- When deploying a node, you can also specify additional steps to run!
- Ironic always includes required deployment steps, at documented priorities
- Provided steps are merged into these required steps for customization.

You cannot specify arbitrary deploy steps with any non-standalone Ironic deployment method.

Deploy Templates (2019)

- Pre-curated lists of additional steps to run during deployments
- Merged with required steps, same as with explicit deploy steps

Nova flavors can be set up
to use a deploy template via traits!

Deployment Interfaces

Deployment via Image aka “Direct” (2015)

- Formerly known as “agent”
- Boots a ramdisk agent to write an image to disk
- Default deploy interface, used by most Ironic operators

Strengths:

- Easy to customize via IB steps
- Image deploys standard in OpenStack

Weaknesses:

- Requires working image
- No support for custom partitioning

Deployment via Ramdisk (2020)

- Boots a node into a provided ramdisk (usually an ISO)
- Not to be confused with **deploy ramdisks** used by other interfaces.

Strengths:

- Works with ISO based workflows
- Doesn’t require working local storage

Weaknesses:

- Requires working image
- Limitations around config drive

Deployment via Anaconda (2021)

- Orchestrates deployment of RHEL-based distributions via anaconda
- Full access to kickstart file syntax, including partitioning tools

Strengths:

- Full access to kickstart
- Custom partitioning

Weaknesses:

- Only tested on RHEL 8/9
- Only supports RHEL-like OSes

Deployment via Homegrown tooling (2017, 2021)

- **`ansible`** (2017) uses custom ansible playbooks to deploy
- **`custom-agent`** (2021) uses custom steps to deploy for full customization

Strengths:
- Fully customizable

Weaknesses:
- Nothing works out of the box

Deployment in Standalone

- Setup metadata in instance_info
  - Image information
  - Capabilities
  - Configdrive
- Make Virtual Interface (VIF) attach calls to Neutron, if using networking
- Call Ironic set provision state API to deploy

Allocations (2019)

- Use allocations API to find a node for deployment
- Can match on resource class and/or traits
- Allocated nodes get instance_uuid set
- Allocations removed explicitly, or after an undeploy

Metalsmith (2019)

- Deployment and scheduling tool for Ironic & Neutron
  - Turn the multiple API calls into a single CLI command using Allocation API
- Optionally coordinates with Glance
- Functionality to be merged into Ironic API / official clients over time

Bifrost (2015)

- Install & deployment tool for standalone Ironic
- Can be used to oneshot deploy servers in bulk or for a long-running Ironic
- Based on ansible; playbooks can be customized to local needs

Deployment in Metal^3

CNCF API design dictates a declarative way. Metal3 uses CAPI with K8S to provision bare metal servers.

Tools for Ironic Operation

Conductor Groups (2019)

- Allows grouping of conductors to manage nodes
- Useful for deployments…
  - …with severe security requirements
  - …with an Ironic API managing geographically distant servers
- Nova can be configured so a compute only sees nodes from a specific group
- Used for horizontally scaling before Node sharding

Node Sharding (2023)

- Horizontally scale services that interact with Ironic with node sharding
- Node’s have a shard value which dictates what shard they are in
- Any tool that queries for a full list of nodes can be sharded
  - Nova computes (2024)
  - Operational scripts

Conductor Graceful Shutdown

- Conductors, at shutdown, will wait for locks to clear (2023)
  - Default of 60 seconds
  - Activated with **SIGTERM**

- Use **SIGUSR2** to mark conductor for draining (2024)
  - Gives running actions (deploys/cleans/etc) time to finish (default is several minutes)
  - Prevents new work from being assigned to a conductor

Node History (2022)

- Records events in the history of the node
- Gives you a historical look at the node in addition to current status
- Useful for troubleshooting nodes or finding failure patterns

Notifications (2016)

- Optionally send a notification to a log file or message bus
- Notifications sent on many triggers:
  - State changes
  - Creation/Modification of API resources
  - …and almost any other action Ironic can perform
- Can be useful for billing, troubleshooting, Statistics-gathering
- Supported by most OpenStack projects

Adoption (2019)

- Add servers to Ironic that already have a workload installed
- Only officially supported for standalone Ironic
- Good choice for migrating environments

Tools for End Users

Not just an Admin API (2021)

- Nodes can be assigned to a project via owner or lessee
- node.owner is the “permanent” node-owner
- node.lessee is a “temporary” instance-user
- Policy allows customization of allowed rules

Useful RBAC tools

- Project managers can onboard nodes for their project
  - Also automatically sets node.owner to the project
- Automatically set lessee when a project member deploys a node
  - ..via Ironic directly (2022)
  - ..via Nova (in progress, est. 2024)
- Single Ironic cluster servicing multiple projects
  - “Bare Metal as a Service”

What’s next for Ironic?

Our best feature: Community

- Ironic has had over 600 unique contributors in ironic repo alone. **Thank you!**

- We want you to be a part of the community. Ask us anything!
  - Have a use case we don't handle?
  - Can't figure out how to implement one of the features you heard today?
  - Just wanting someone who understands what it's like when your hardware isn't behaving

- Find us!
  - #openstack-ironic on OFTC IRC (bridged to matrix)
  - openstack-discuss@openstack.org

Thank you…

Any questions?

…but wait, there’s more!

Retiring and Deprecated Features

Do Not Use

- Chassis API
  - Officially deprecated; if you need this functionality checkout parent/child node support (2023)
- ironic-staging-drivers
  - Unofficial driver repo; not a part of Ironic and unlikely to work in most cases
- Configuration molds (`import_configuration`/`export_configuration`)
  - Only ever supported by Dell hardware.
  - Was never adopted by the larger hardware community

Obsolete Drivers

Driver	Hardware/Vendor	Alternative
`ibmc`	Huawei	`redfish`
`xclarity`	Lenovo Cluster Manager	`redfish` (to individual systems)
`idrac-wsman`	Dell iDRAC 5 & 6	`idrac-redfish` (iDRAC 7+)
`ilo`	HPE iLO 5 or older	`redfish` (iLO 6 and newer)

Ironic will only fully remove these drivers when hardware they exclusively support is EOL.
When possible, use the **`redfish`** driver!

ironic-inspector (the separate service)

- Ironic Inspector service functionality has been merged into Ironic (2024)
  - except node discovery
  - except inspection rules

- New deployments should not use an independent ironic-inspector service

Inspection functionality still fully exists, now built into Ironic!*

Not recommended for use, but not deprecated

- Partition images
  - Still officially supported, but whole-disk images are preferred
  - May lead to different bootloader configuration than the usual distribution expectation
- Node discovery
  - Requires a DHCP server setup to boot anything in a subnet.
  - Users still have to supply BMC credentials to discovered nodes
  - Most deployers import hardware via packing lists and have MACs already
- Network Boot from Volume (without BMC assistance)
  - Cannot support secure boot without significant operator effort

Thank you!
(again)