
CERN

From Hire to Retire!

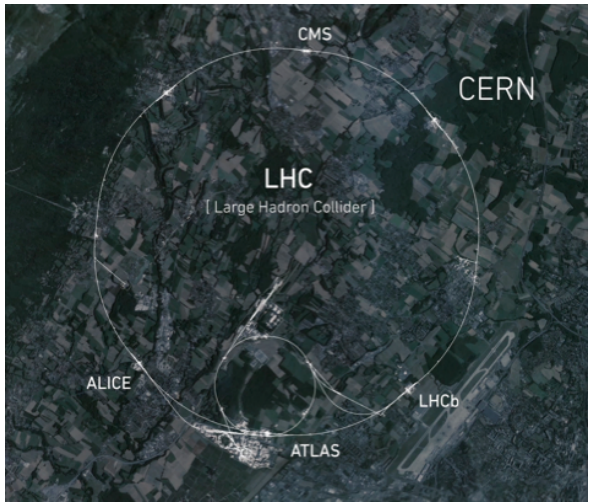
Server Life-cycle Management with Ironic at CERN

Arne Wiebalck & Surya Seetharaman
CERN Cloud Infrastructure Team

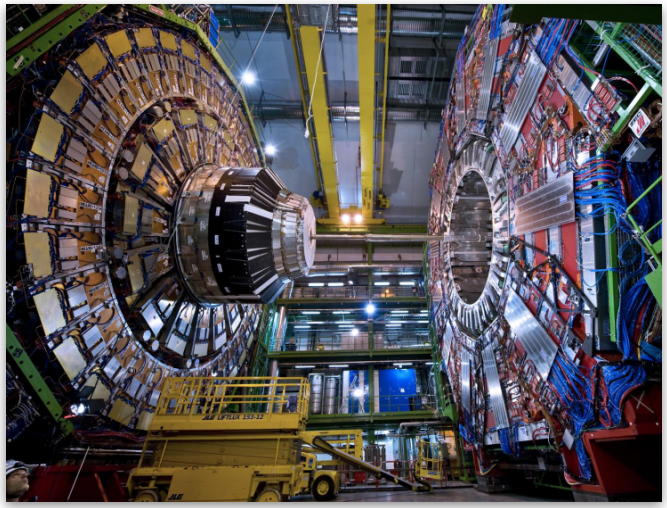
OPEN INFRASTRUCTURE SUMMIT Shanghai | November 4-6, 2019

CERN and CERN IT in 1 Minute ...

➔ Understand the mysteries of the universe!



- Large Hadron Collider
- 100 m underground
- 27 km circumference



- 4 main detectors
- Cathedral-sized
- O(10) GB/s

- Initial reconstruction
- Permanent storage
- World-wide distribution

The 2½ CERN IT Data Centers

Meyrin (CH)

~12800 servers

Budapest (HU)

~2200 servers

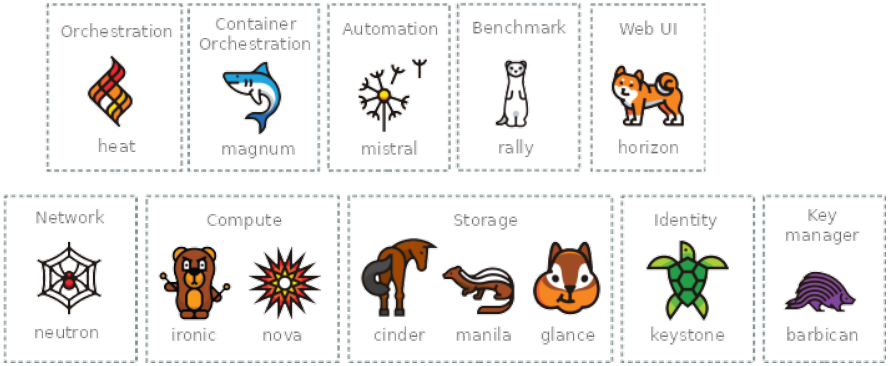
LHCb Point-8 (FR)

~800 servers

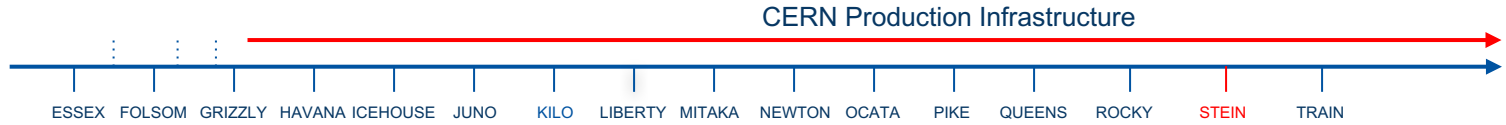
OpenStack Deployment in CERN IT

In production since 2013!

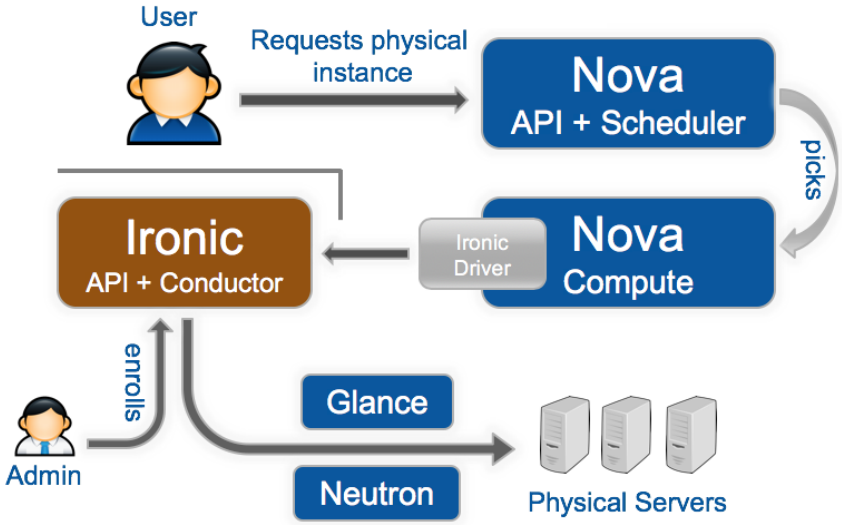
- 8’500 hosts with ~300k cores
- ~35k instances in ~80 cells
- 3 main regions (+ test regions)
- Wide use case spectrum
- Control plane a use case as well
  Ironic controllers are VMs on compute nodes which are physical instances Nova created in Ironic ...

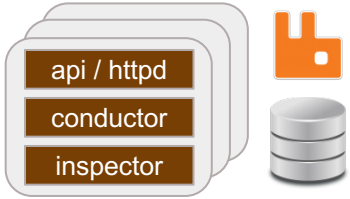


ceph RDO puppet



Ironic & CERN's Ironic Deployment

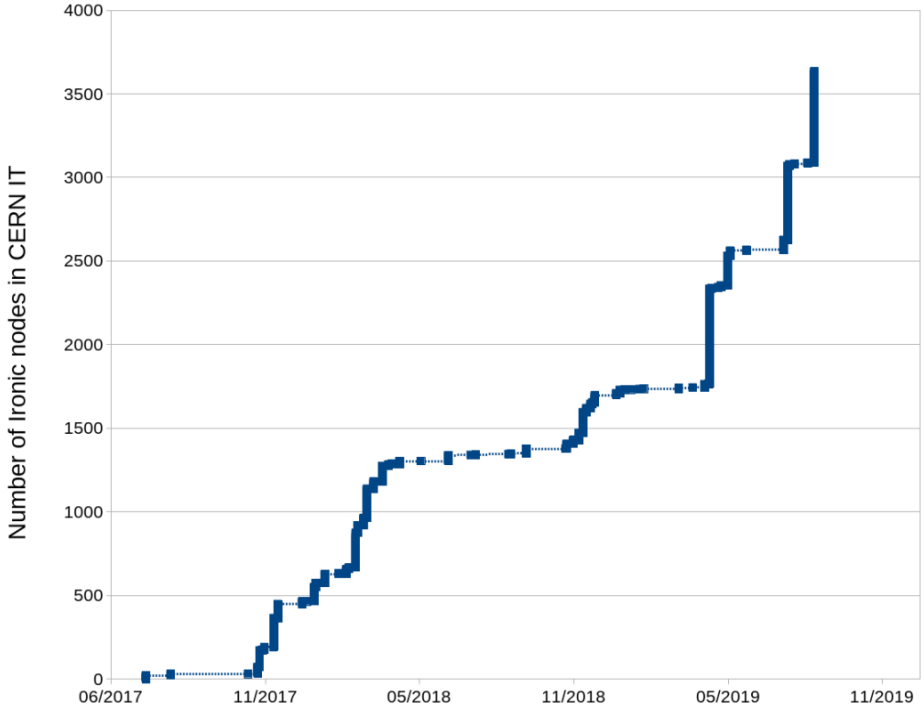




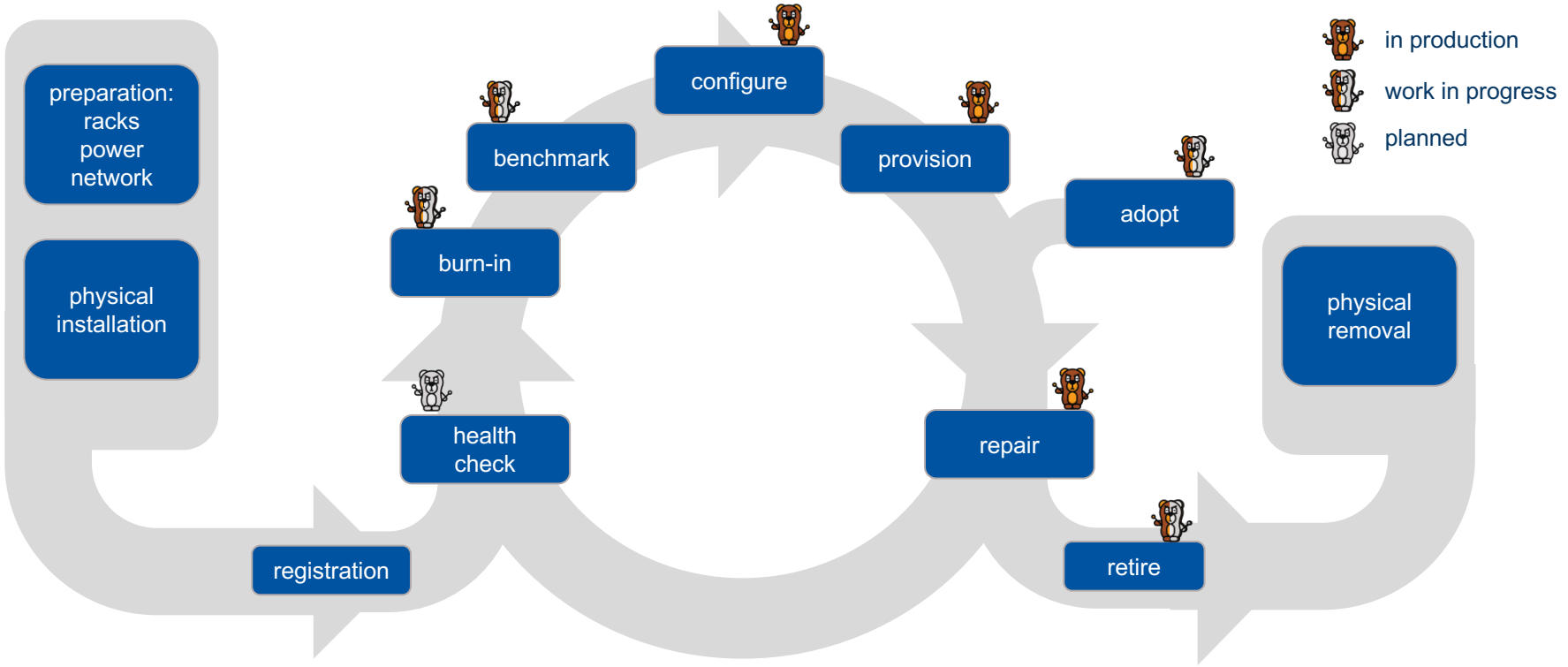
- 3x Ironic controllers
- in a bare metal cell (1 CN for 3'500 nodes!)
- currently on Stein++

CERN's Ironic Deployment: Node Growth

- **New deliveries**
  - Ironic-only
- **Data center repatriation**
  - adoption imminent
- **Scaling issues**
  - power sync
  - resource tracker
  - image conversion



Server Life-cycle Management with Ironic



The Early Life-cycle Steps ...

registration

➔ **Currently done with a CERN-only auto-registration image**
- ➢ could move to Ironic, unclear if we want to do this

health check

➔ **Currently done by "manually" checking the inventory**
- ➢ should move to Ironic's introspection rules (S3)

burn-in

➔ **Will become a set of cleaning steps "burnin-{cpu,disk,memory,network}"**
- ➢ rely on standard tools like badblocks
- ➢ stops upon failure

benchmark

➔ **Will become a cleaning step "benchmark"**
- ➢ launches a container which will know what to do

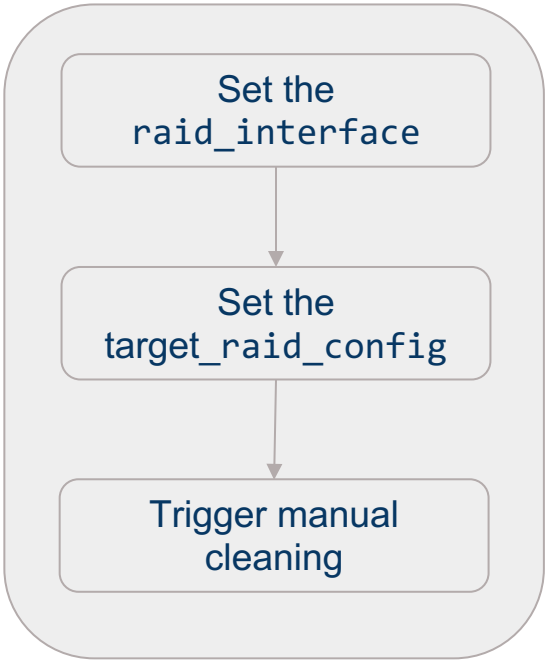

in production


work in progress


planned

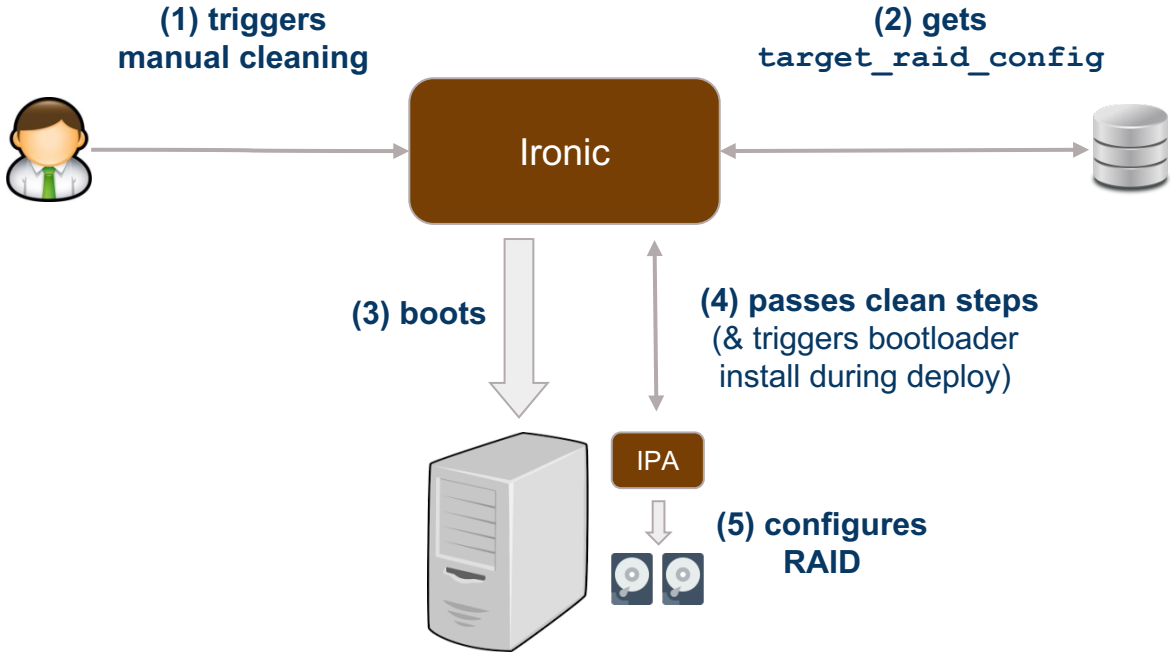
Configure Clean-time Software RAID

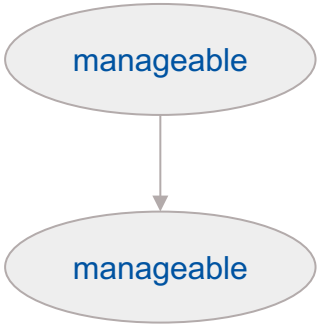
- **Vast majority of our 15'000 nodes rely on Software RAID**
  - Redundancy & space
- **The lack of support in Ironic required re-installations**
  - Additional installation based on user-provided kickstart file
  - Other deployments do have similar constructs for such setups
- **With the upstream team, we added Software RAID support**
  - Available in Train
  - *Initial* support
  - In analogy to hardware RAID implemented as part of 'manual' cleaning
  - In-band via the Ironic Python Agent



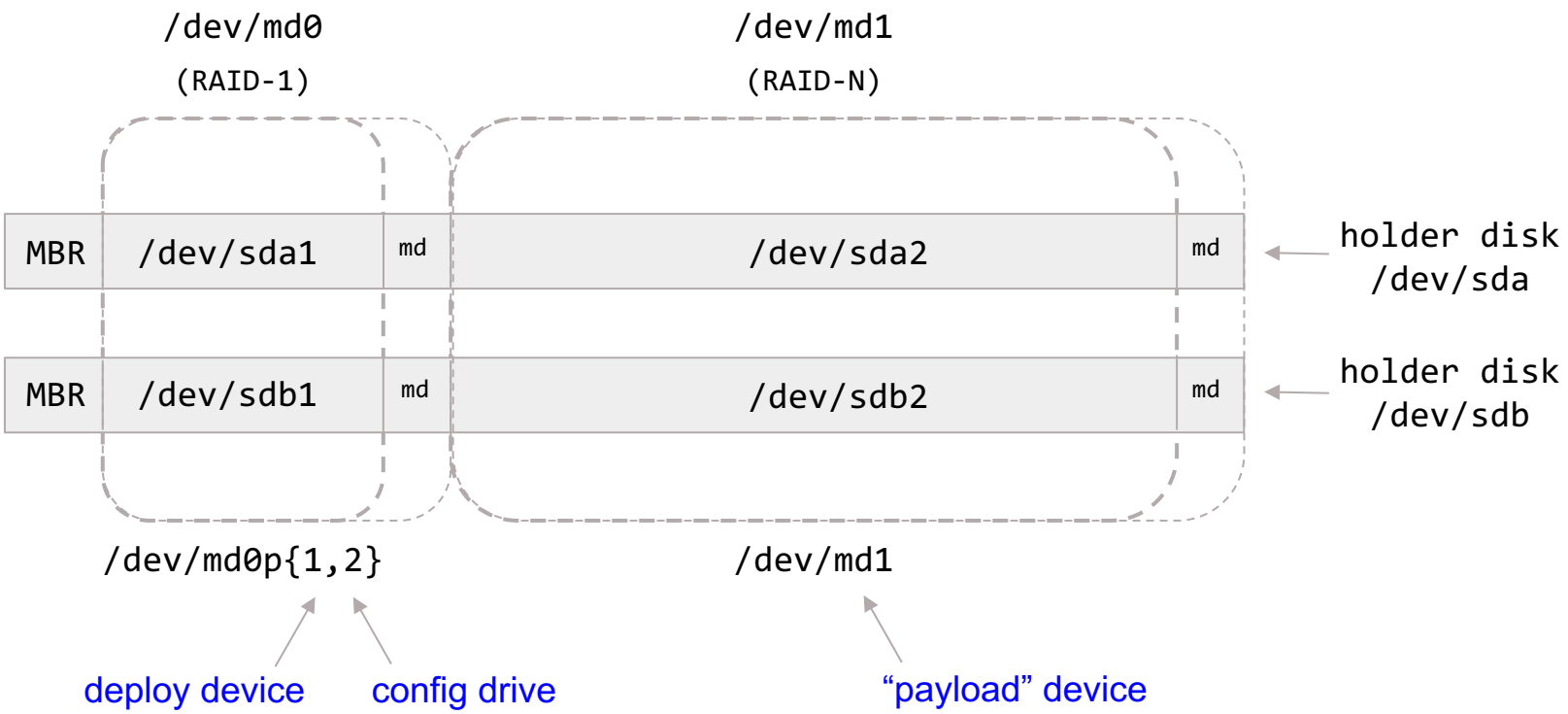
Configure Clean-time Software RAID

➔ How does Ironic configure the RAID?





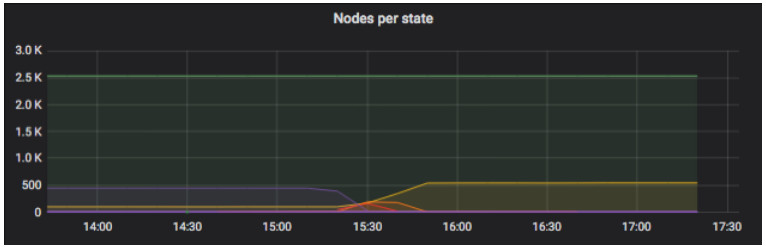
Configure Clean-time Software RAID



Configure Clean-time Software RAID

➔ What about GPT/UEFI and disk selection?

- ➢ Initial implementation uses MBR, BIOS, and fixed partition for the root file system ...
- ➢ GPT works (needed mechanism to find root fs)
- ➢ UEFI will require additional work … ongoing!
- ➢ Disk selection not yet possible … ongoing!



Cleaning 400 nodes triggered the creation of Software RAIDs

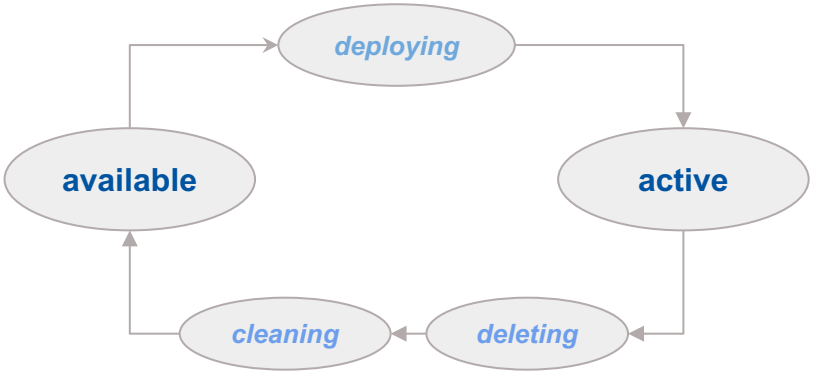
➔ How to repair a broken RAID?

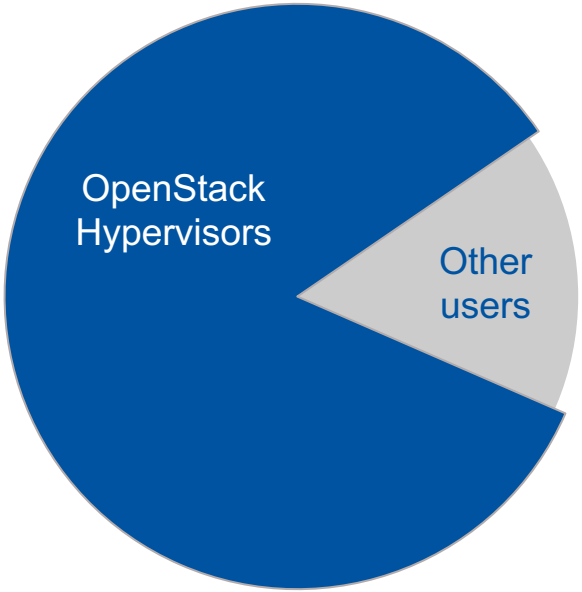
- ➢ “broken” == “broken beyond repair” (!= degraded)
- ➢ Do it the cloudy way: delete the instance!
- ➢ At CERN: {delete,create}_configuration steps part of our custom hardware manager
- ➢ What about ‘`nova rebuild`’?

Provision The Instance Life-cycle

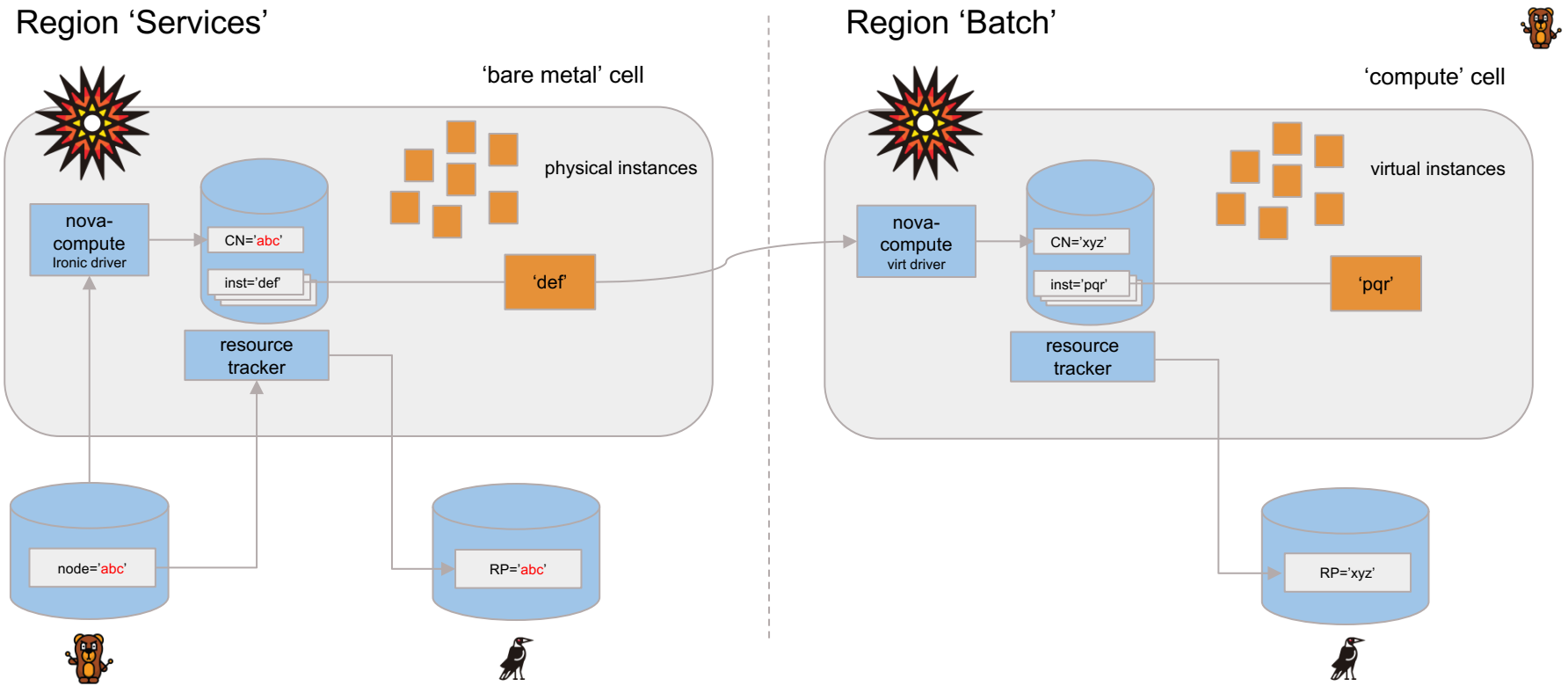
Why Ironic?

- ➔ Single interface for virtual and physical resources
- ➔ Same request approval workflow and tools
- ➔ Satisfies requests where VMs are not apt
- ➔ Consolidates the accounting





Provision Physical Instances as Hypervisors



Adopt “Take over the world!”

- ➔ **Ironic provides “adoption” of nodes, but this does not include instance creation!**
- ➔ **Our procedure to adopt production nodes into Ironic:**
  - ➢ Enroll the node, including its resource_class (now in ‘manageable’)
  - ➢ Set fake drivers for this node in Ironic
  - ➢ Provide the node (now in ‘available’)
  - ➢ Create the port in Ironic (usually created by inspection)
  - ➢ Let Nova discover the node
  - ➢ Add the node to the placement aggregate
  - ➢ Wait for the resource tracker
  - ➢ Create instance in Nova (with flavor matching the above resource_class)
  - ➢ Set real drivers and interfaces in Ironic

Repair Even with Ironic nodes do break!

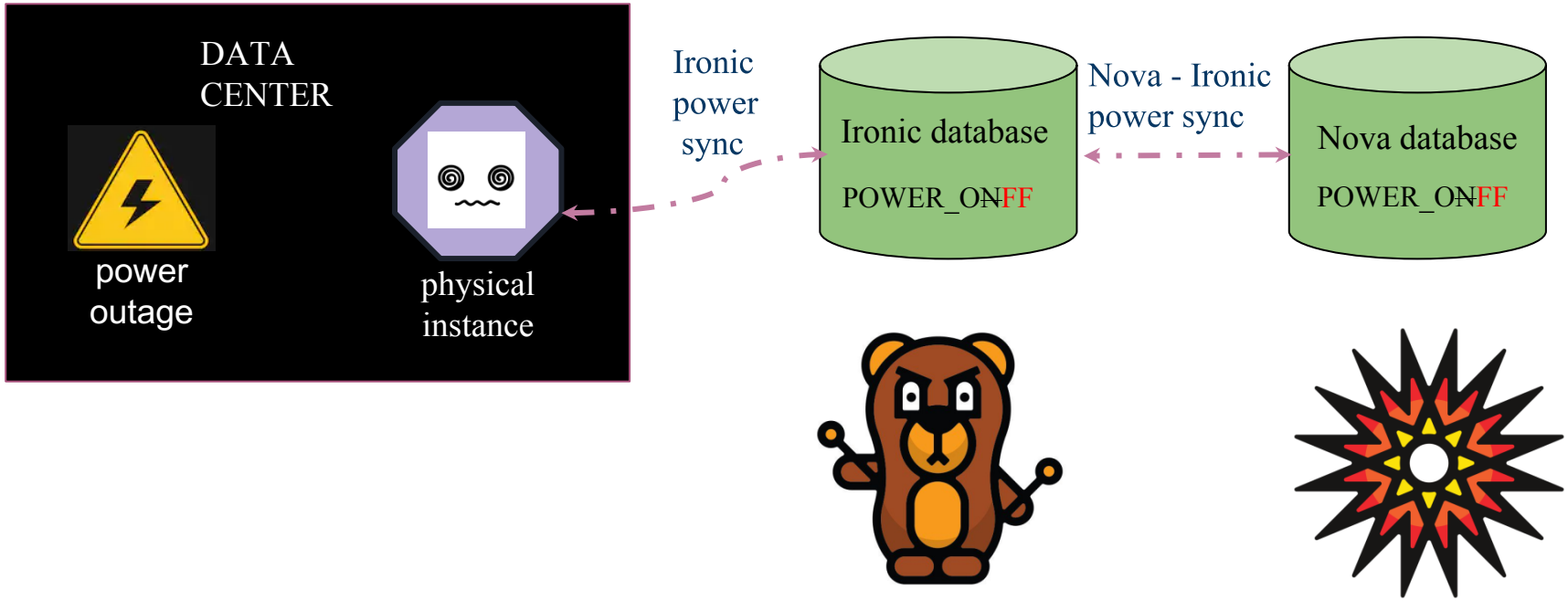
- **The OpenStack team does not directly intervene on nodes**
  - Dedicated repair team (plus workflow framework based on Rundeck & Mistral)
- **Incidents: scheduled vs. unplanned**
  - BMC firmware upgrade campaign vs. motherboard failure
- **Introduction of Ironic required workflow adaptations, training, and ...**
  - New concepts like “physical instance”
  - Reinstallations
- **… upstream code changes in Ironic and Nova**
  - power synchronisation (the “root of all evil”)

RUNDECK

MISTRAL
an OpenStack Community Project

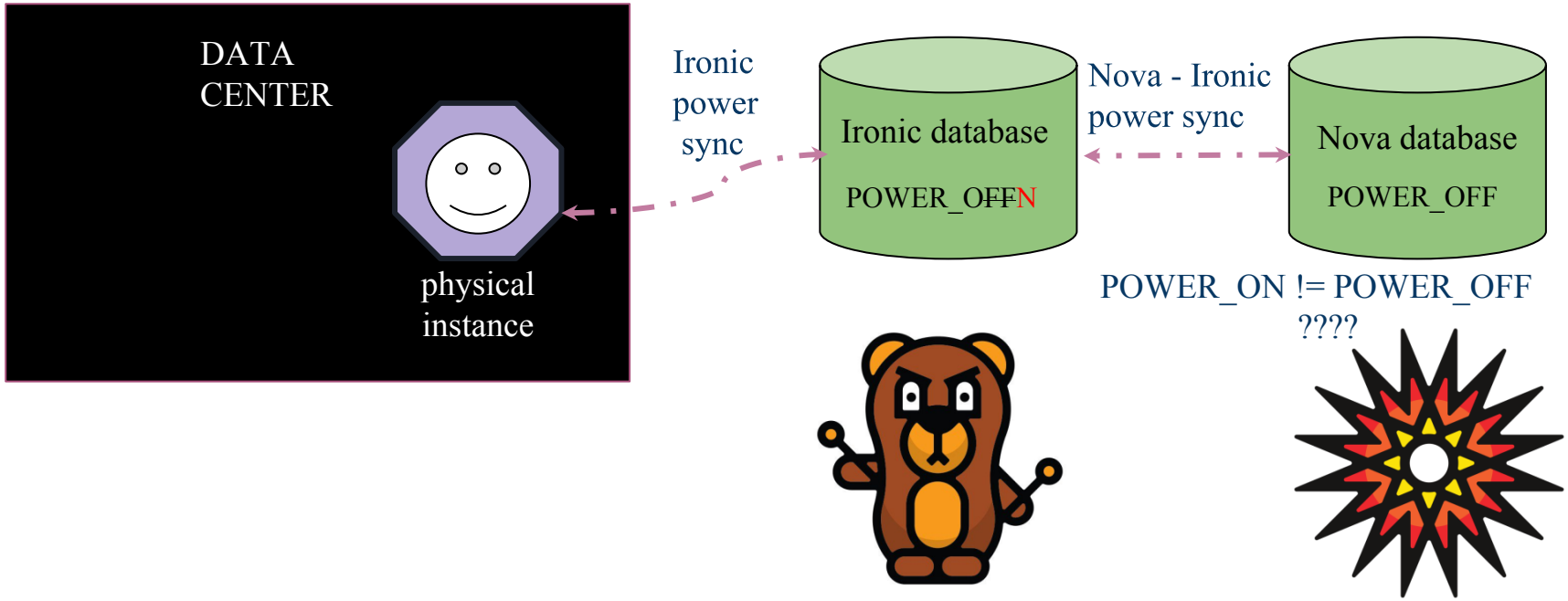
Repair The Nova / Ironic Power Sync

Problem Statement



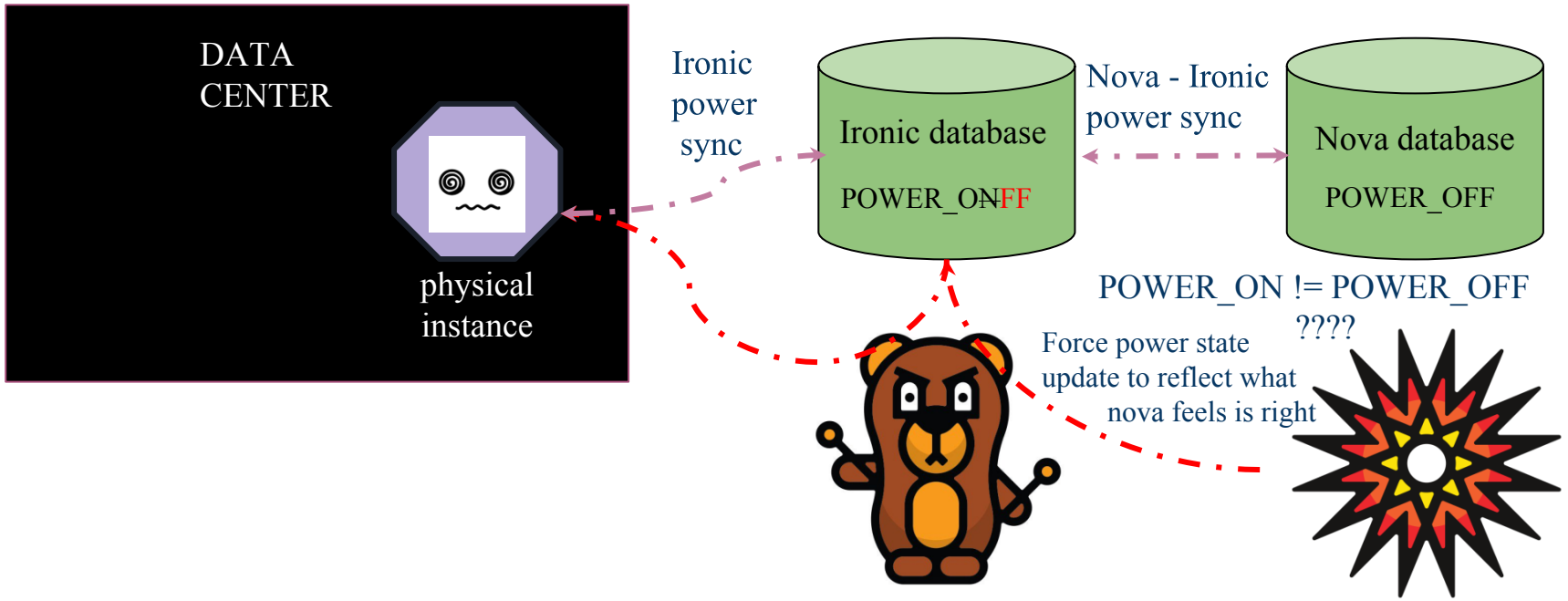
Repair The Nova / Ironic Power Sync

Problem Statement



Repair The Nova / Ironic Power Sync

Problem Statement



Repair The Nova / Ironic Power Sync

Problem Statement

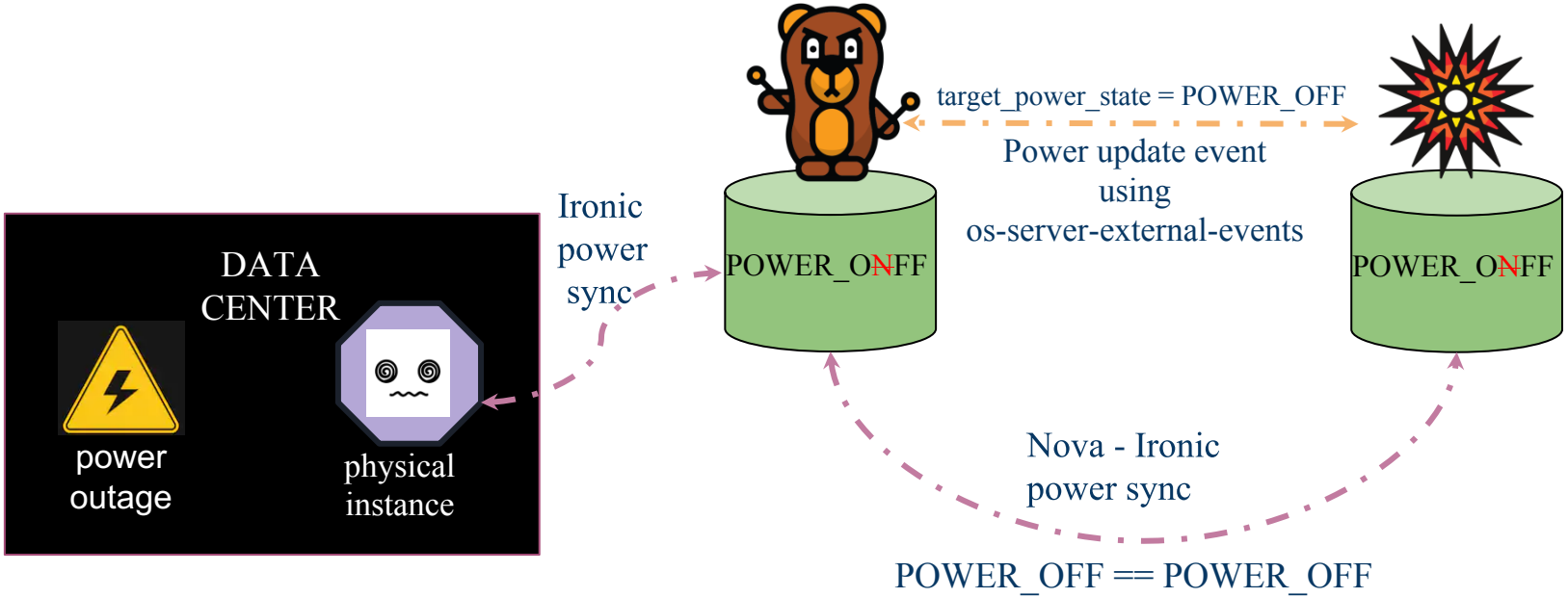
➔ **Unforeseen events like power outage:**

- ➢ **Physical instance goes down**
  - **Nova puts the instance into SHUTDOWN state**
    - **through the `` `_sync_power_states` `` periodic task**
    - **hypervisor regarded as the source of truth**
- ➢ **Physical instance comes back up without Nova knowing**
  - **Nova again puts the instance back into SHUTDOWN state**
    - **through the `` `_sync_power_states` `` periodic task**
    - **database regarded as the source of truth**

Nova should not force the instance to POWER_OFF when it comes back up

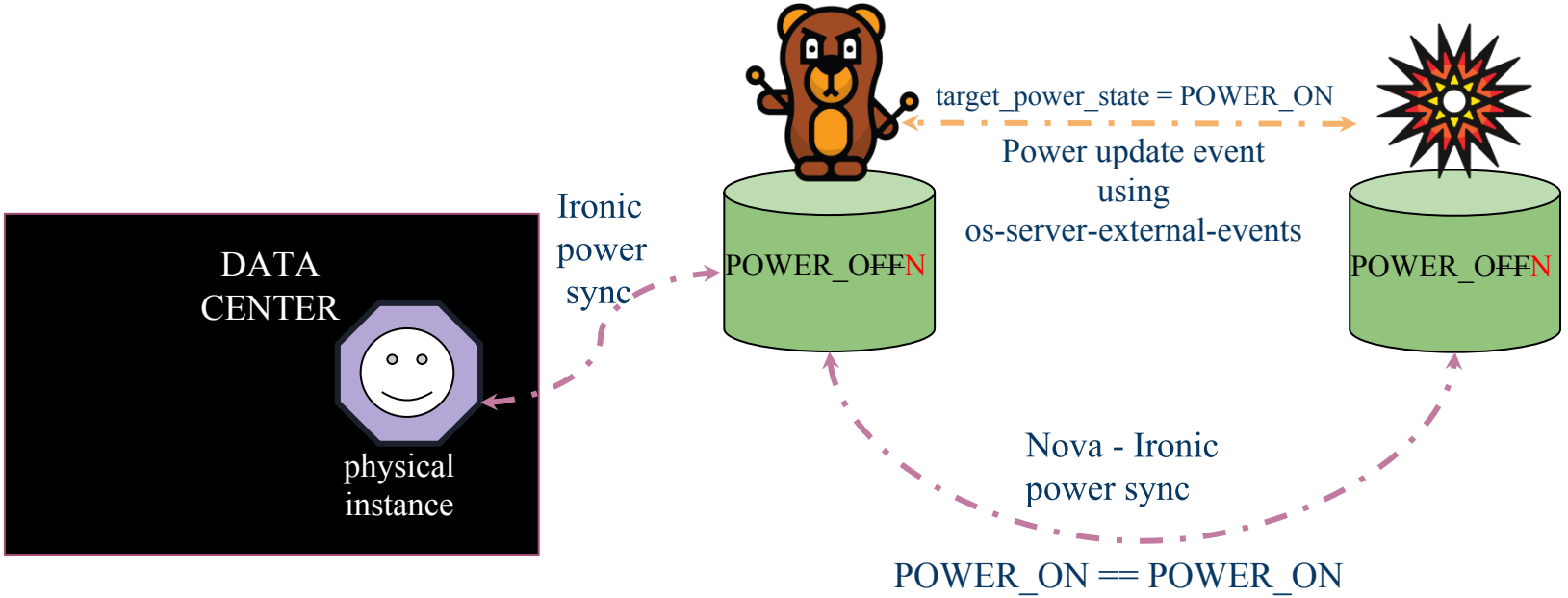
Repair The Nova / Ironic Power Sync

Implemented Solution



Repair The Nova / Ironic Power Sync

Implemented Solution



There can be race conditions depending on the sequence of occurrence of events.

Repair The Nova / Ironic Power Sync

Implemented Solution

Ironic sends power state change callbacks to Nova

- ➔ Operator has to set [nova].send_power_notifications config option to `True`
- ➔ JSON request body sent from Ironic

```
{
    "events": [
        {
            "name": "power-update",
            "server_uuid": "3df201cf-2451-44f2-8d25-a4ca826fc1f3",
            "tag": target_power_state
        }
    ]
}
```

- ➔ Done via the os-server-external-events Nova api
- ➔ Read power_update spec and documentation for more details

Repair The Nova / Ironic Power Sync

Implemented Solution

Ironic sends power state change callbacks to Nova

➔ **JSON response body sent from Nova**

```
{
    "events": [
        {
            "code": 200,
            "name": "power-update",
            "server_uuid": "3df201cf-2451-44f2-8d25-a4ca826fc1f3",
            "status": "completed",
            "tag": target_power_state
        }
    ]
}
```

Available from

➔ **Nova updates its database to reflect the power state change**

➢ **Ironic regarded as source of truth**

Retire The End of the Cycle

- **A simple procedure for now**
  - Deleting instances triggers cleaning
  - Setting maintenance avoids instance creation
  - The maintenance reason marks them for removal

- **No explicit “node retirement” support in Ironic**
  - Time windows allow for re-use
  - Explicit tagging would be helpful

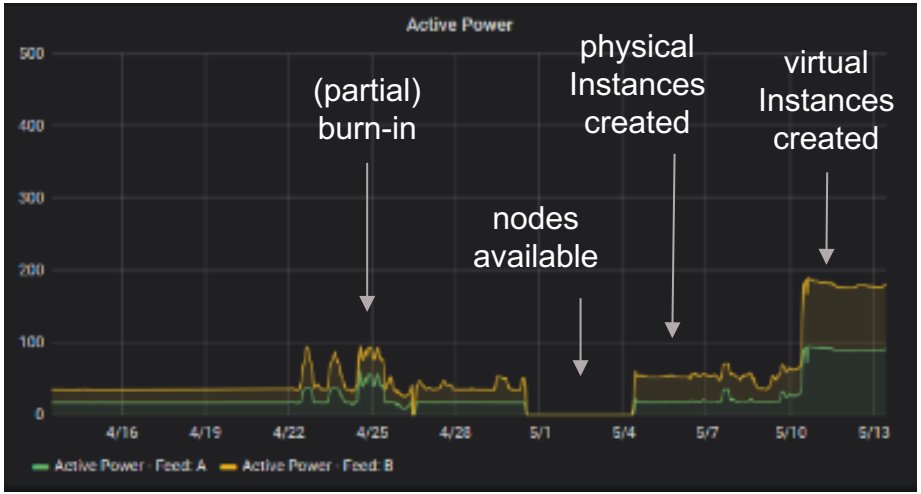
- **Proposals to introduce a retirement flag (or even state)**
  - Spec: “Add support for node retirement” https://review.opendev.org/#/c/656799/

A Pain Point: Resource Tracking

- **The resource tracker loops sequentially over compute nodes**
  - Holds a semaphore and hence blocks instance creation
  - OK for virtual machines, a scaling issue for bare metal deployments
- **This creates a noticeable dead time for instance creation**
  - For 1 node with 3500 physical nodes the turn-around time is ~60 mins
  - … this is already with upstream and local patches to reduce the overhead
- **Stop-gap solution: adapt the resource tracker cycle**
  - Compromise between blockage and placement updates
- **Potential solutions: n-compute sharding and per-instance locking**

Summary

- ➔ **At CERN, Ironic is used for the majority of a server's life cycle steps**
- ➔ **We work on the remaining steps, more features and we plan to pass +10k servers!**



Power consumption in container DC during allocation of a new delivery

謝謝啦

Thank you!