Sustainability in Computing

Energy Efficient Placements of Edge Workloads

Kaiyi Liu

Software Engineer Intern

Parul Singh

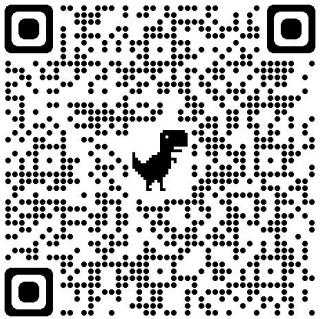
Senior Software Engineer

Red Hat

Who We Are

CLOUD NATIVE
COMPUTING FOUNDATION

- Community based initiatives on environmental sustainability
- Proposal: **CNCF TAG Environmental Sustainability**
- Carbon Aware Scaling with KEDA
    - a community based initiative; investigates how to use electricity carbon intensity to make workload scaling decisions.
- CLEVER:
    - Container Level Energy-efficient VPA Recommender for Kubernetes



Agenda

- Background
- Introduce our Sustainability stack
  - Kepler
  - Model Server
- Demo

Red Hat

Background

According to Gartner, “In 2021, an ACM technology brief estimated that the information and communication technology (ICT) sector contributed between 1.8% and 3.9% of global carbon emissions.

Red Hat

Background

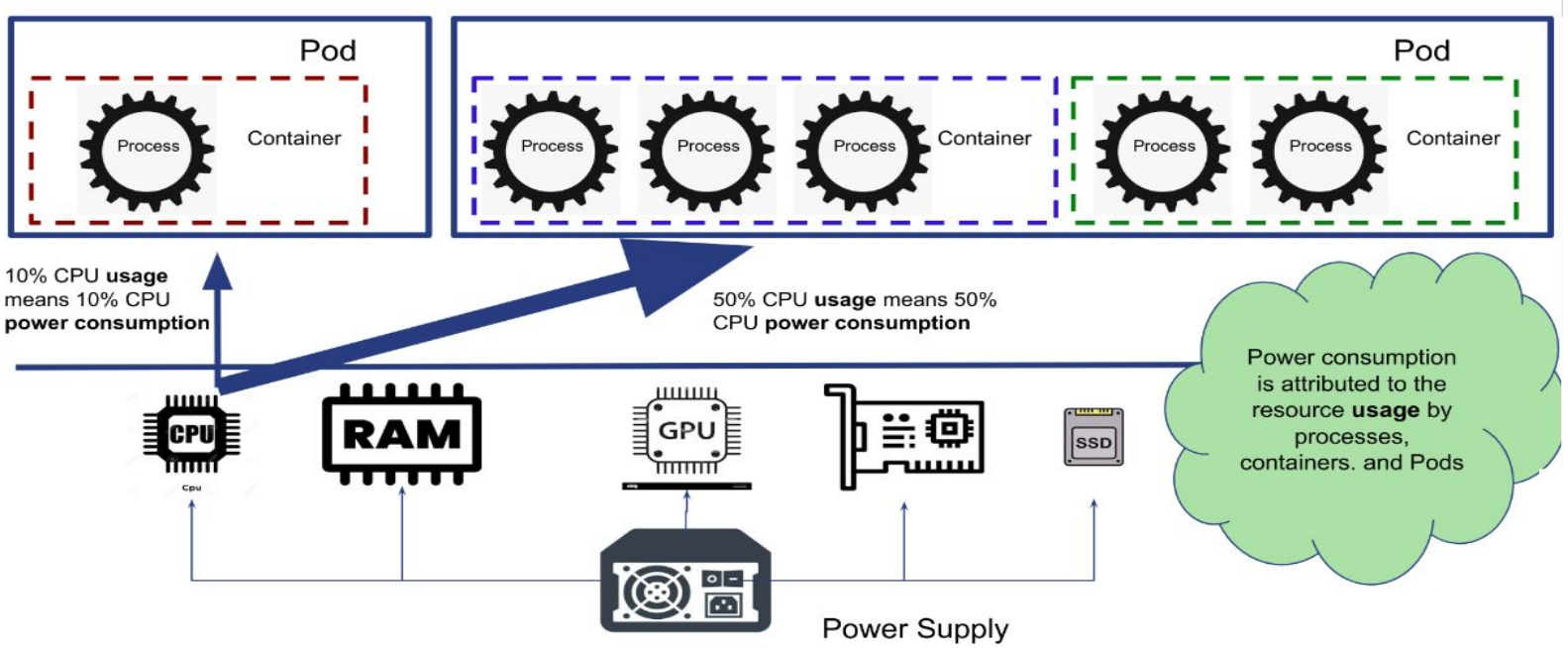
- How to measure energy consumption indirectly?
- How to measure energy consumption of workloads?
- How to attribute power on share resources to processes, containers or Pods?

Introducing the Cloud Native Sustainability Stack

1. Kepler
2. Kepler Model Server

Red Hat

Energy Consumption Attribution Methodology



Reference: https://lca.ece.utexas.edu/pubs/bircher-TC2012.pdf

Red Hat

Kepler

Kubernetes based Efficient Power Level Exporter

Red Hat

Kepler: Kubernetes based Efficient Power Level Exporter

Uses software counters to measure power consumption by hardware resources and exports as Prometheus metrics

Red Hat

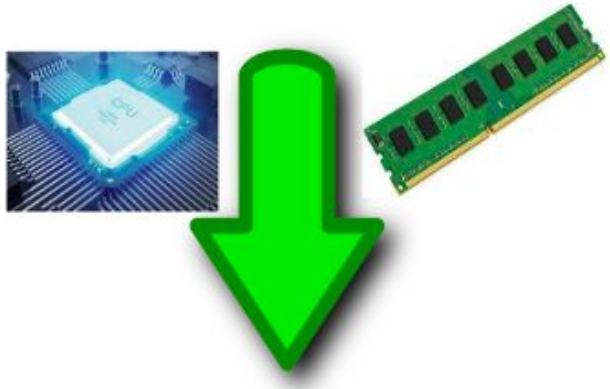
Reporting

- Per Pod level energy consumption reporting, including **CPU/GPU, RAM**
- Support **bare metal** as well as **VM**
- Support **Prometheus**

Red Hat

Reporting

- Per Pod level energy consumption reporting, including **CPU/GPU, RAM**
- Support **bare metal** as well as **VM**
- Support **Prometheus**



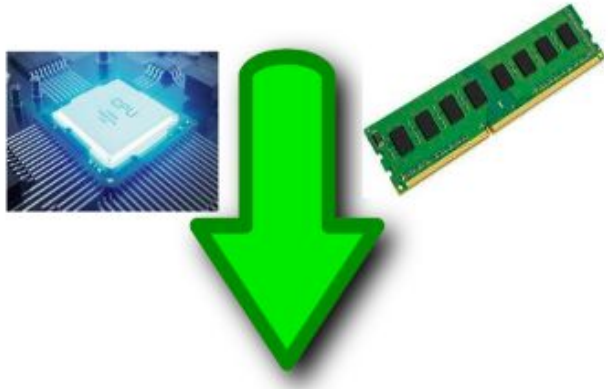
Reduction

- Reduced computational resource used by the probe
- Using **eBPF**

Red Hat

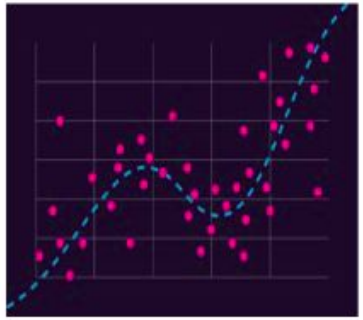
Reporting

- Per Pod level energy consumption reporting, including **CPU/GPU, RAM**
- Support **bare metal** as well as **VM**
- Support **Prometheus**



Reduction

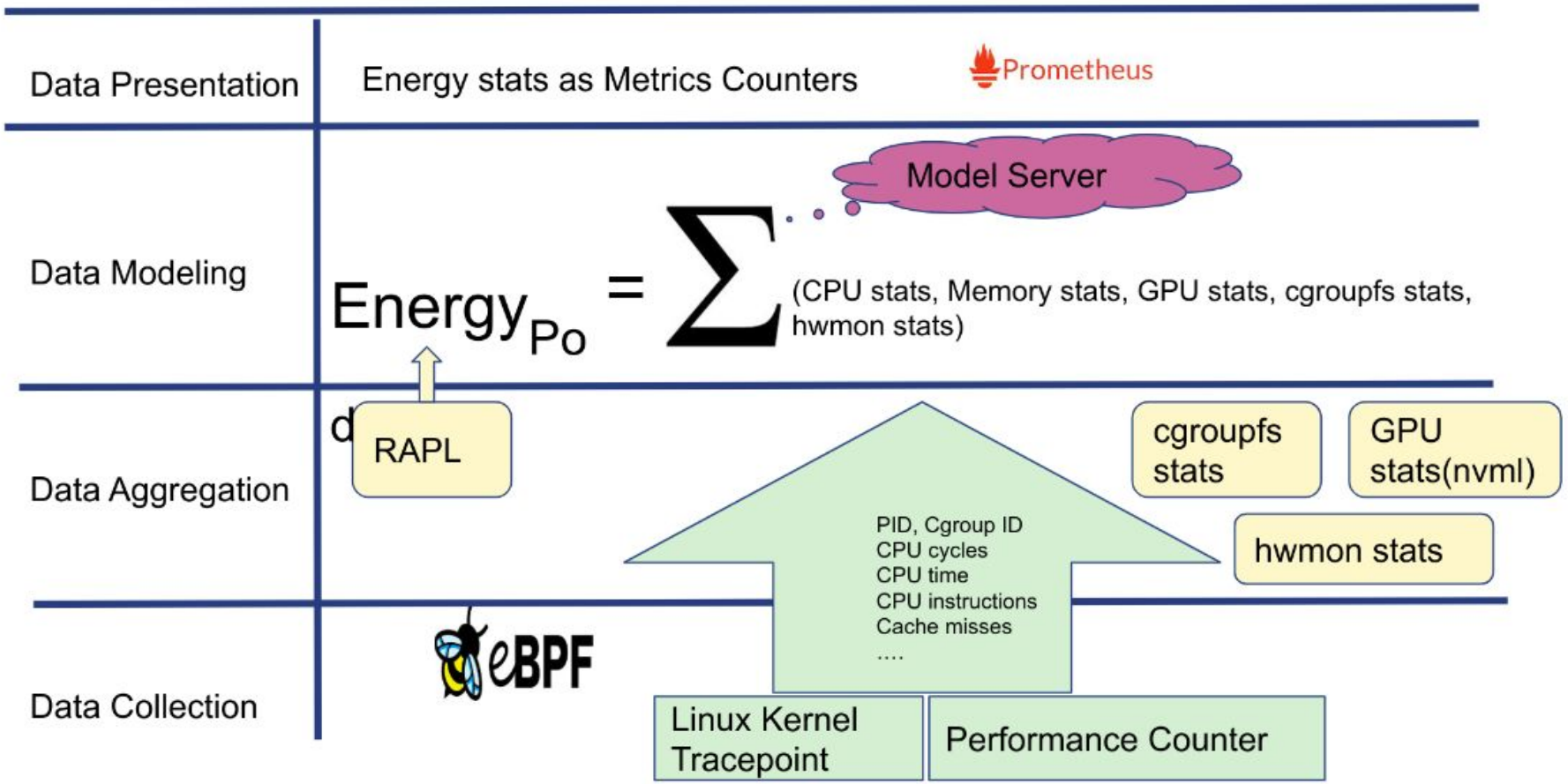
- Reduced computational resource used by the probe
- Using **eBPF**



Regression

- Support **ML** models to estimate energy consumption
- Science based approach

Red Hat


Data Presentation
Energy stats as Metrics Counters
Prometheus
Data Modeling
Model Server
Energy_Po = Σ
(CPU stats, Memory stats, GPU stats, cgroupfs stats, hwmon stats)
Data Aggregation
RAPL
cgroupfs stats
GPU stats(nvml)
hwmon stats
PID, Cgroup ID
CPU cycles
CPU time
CPU instructions
Cache misses
....
Data Collection
eBPF
Linux Kernel Tracepoint
Performance Counter

Red Hat

Kepler Model Server

Red Hat

About Kepler Model Server

- Default: Kepler uses supported power meter tools to measure node level energy metrics (CPU core, DRAM)
- Problem: No supported power meter for Kepler
- Model Server Goal: Provide Trained Models for Kepler that use Software Counters/Performance metrics to predict missing energy metrics
- Current Tech Stack: Tensorflow Keras, Flask, Prometheus

Kepler Model Server's Models

- CPU Core Energy Consumption Model: Linear Regression
  - Label: **CPU Core Energy Consumption**
  - Features: **cpu_architecture, curr_cpu_cycles, curr_cpu_instructions, curr_cpu_time**
- Online Learning

Red Hat

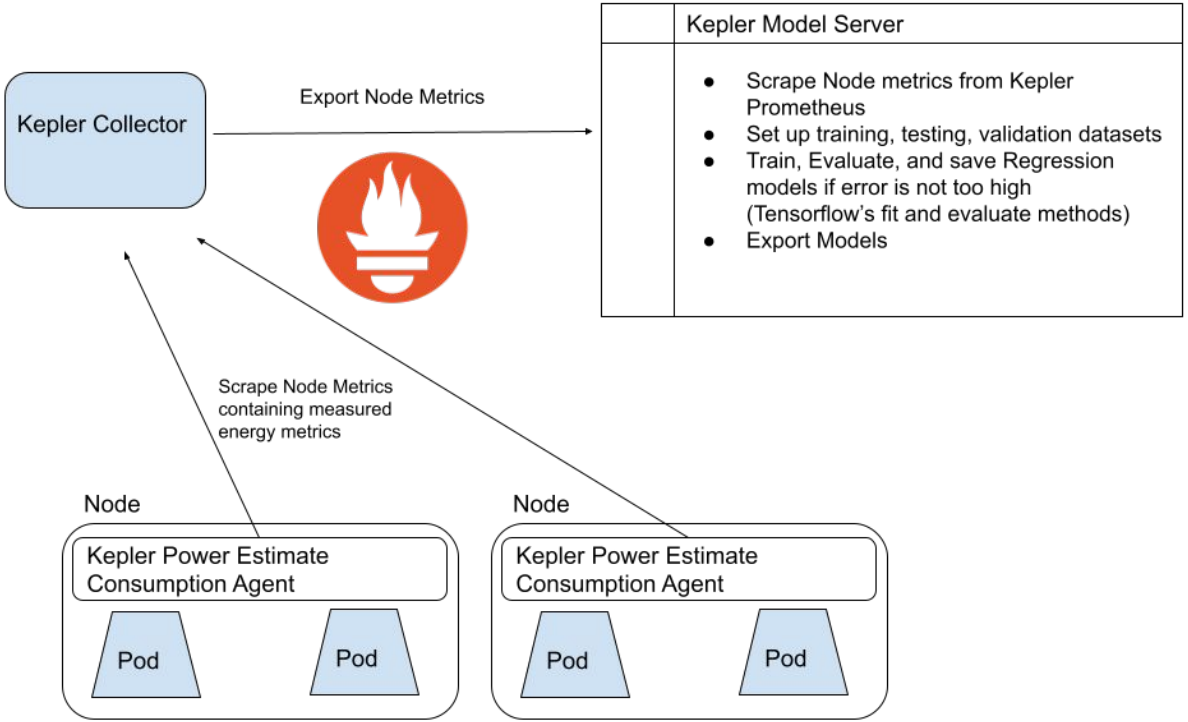
Kepler Model Server's Models Cont

- Dram Energy Consumption Model: Linear Regression
  - Label: **DRAM Energy Consumption**
  - Features: **cpu_architecture, curr_cache_misses, memory_working_set**
- Online Learning

Red Hat

Model Server and Kepler

Training Phase

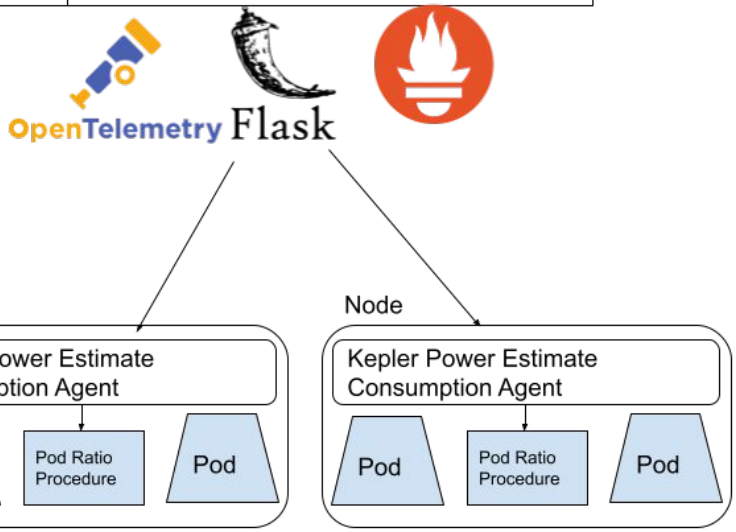


Red Hat

Model Server and Kepler

Exporting Phase

	Kepler Model Server
	• Scrape Node metrics from Kepler Prometheus • Set up training, testing, validation datasets • Train, Evaluate, and save Regression models if error is not too high (Tensorflow's fit and evaluate methods) • Export Models

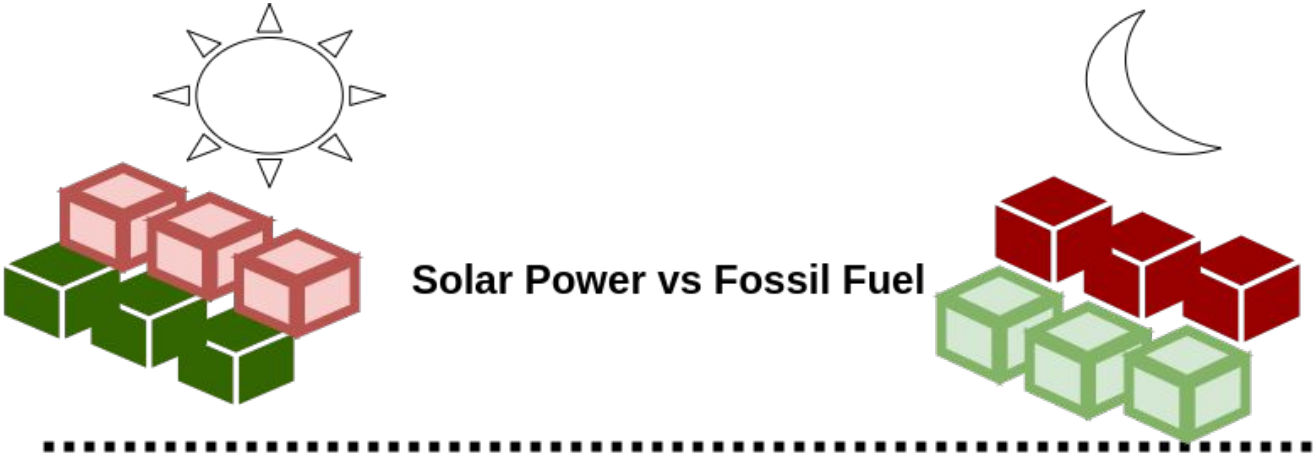


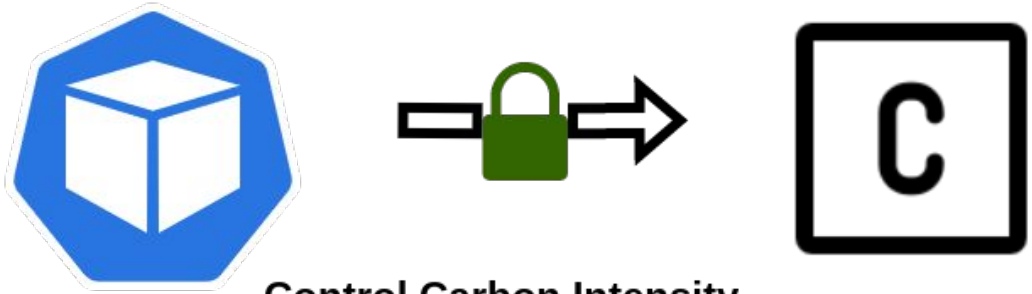
Red Hat

Carbon Intensity Aware Scheduling

Red Hat

Use Cases





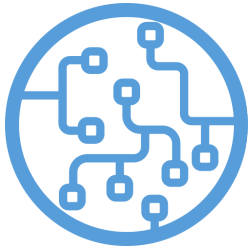
Red Hat

Use Case Premise

- Multi-node cluster
- Nodes in different zones
- Long running batch/ML workloads

Demo Set Up

- 3 Nodes OpenShift Cluster
- Monitoring: Prometheus
- Taints/Tolerations/NodeSelectors
- Carbon Intensity Forecaster

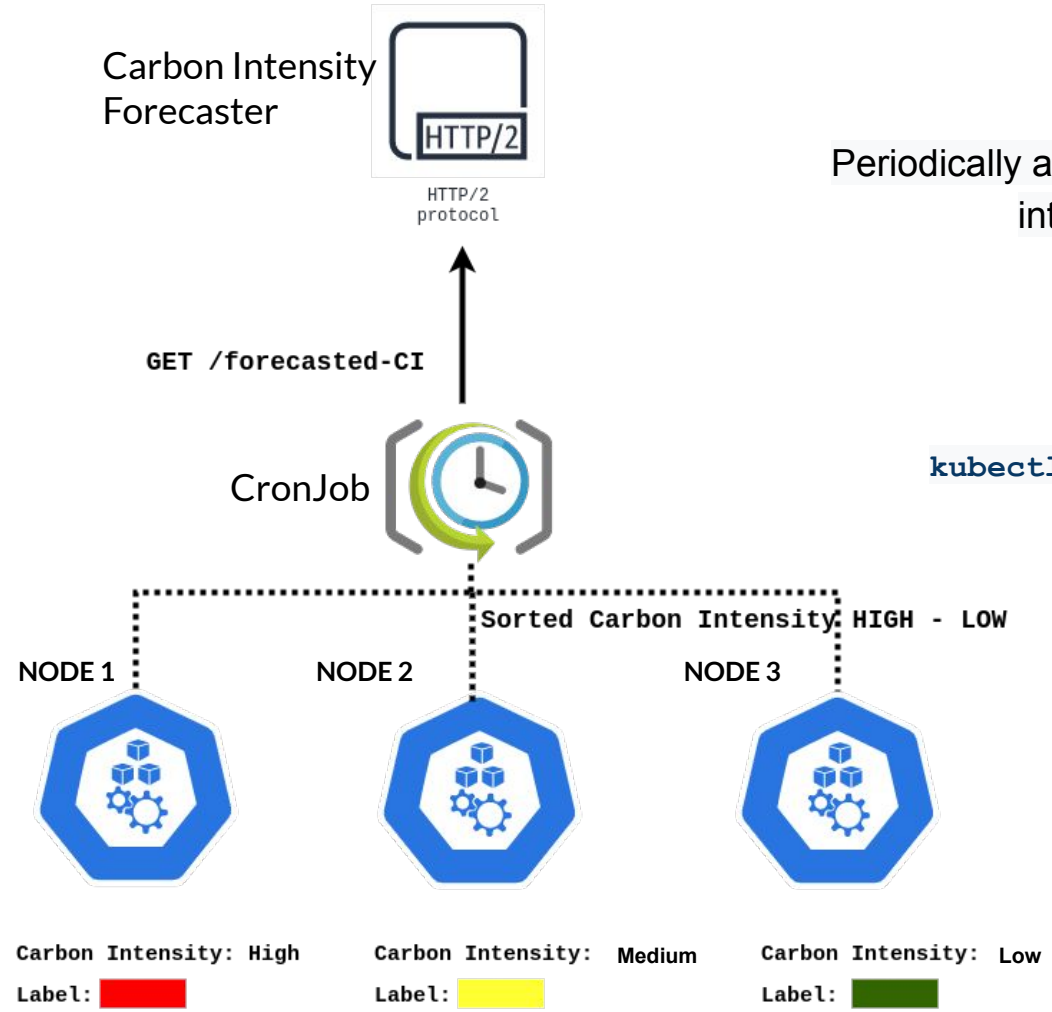


Red Hat

Carbon Intensity Forecaster

- Exporter scrapes from Public Energy APIs (ex. Electricity Map) and exports as Prometheus metrics
- Scrapes prometheus metrics from the exporter to update models for each node
- Carbon Intensity Forecaster and Exporter are extendable interfaces

Red Hat

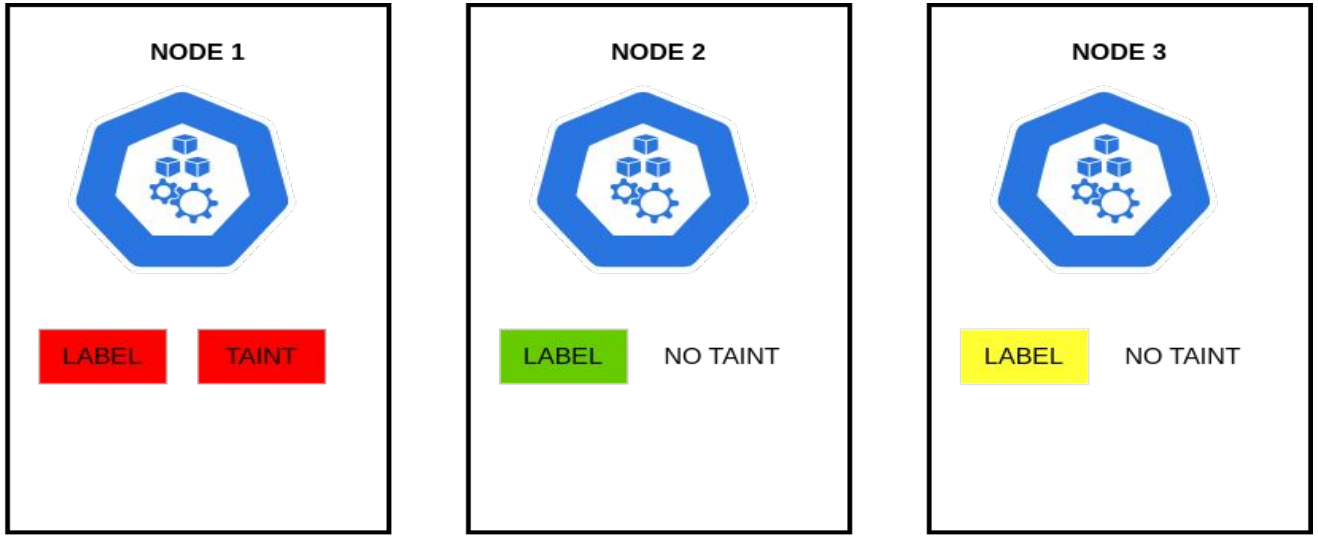


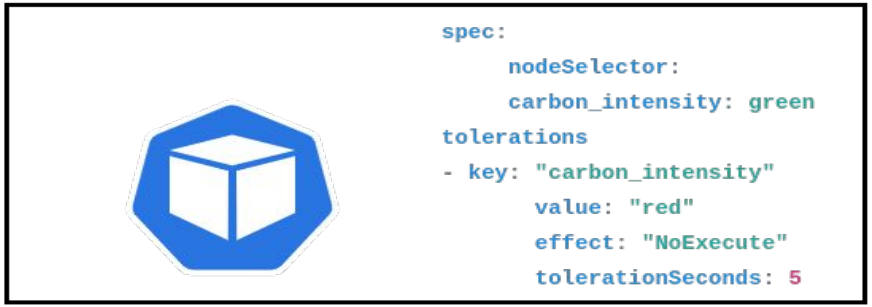
Node labels

Periodically assign node labels according to forecasted carbon intensity of the zone the node is present

```
kubectl label nodes ip-10-0-169-34.ec2.internal
carbon_intensity=green
```

Red Hat



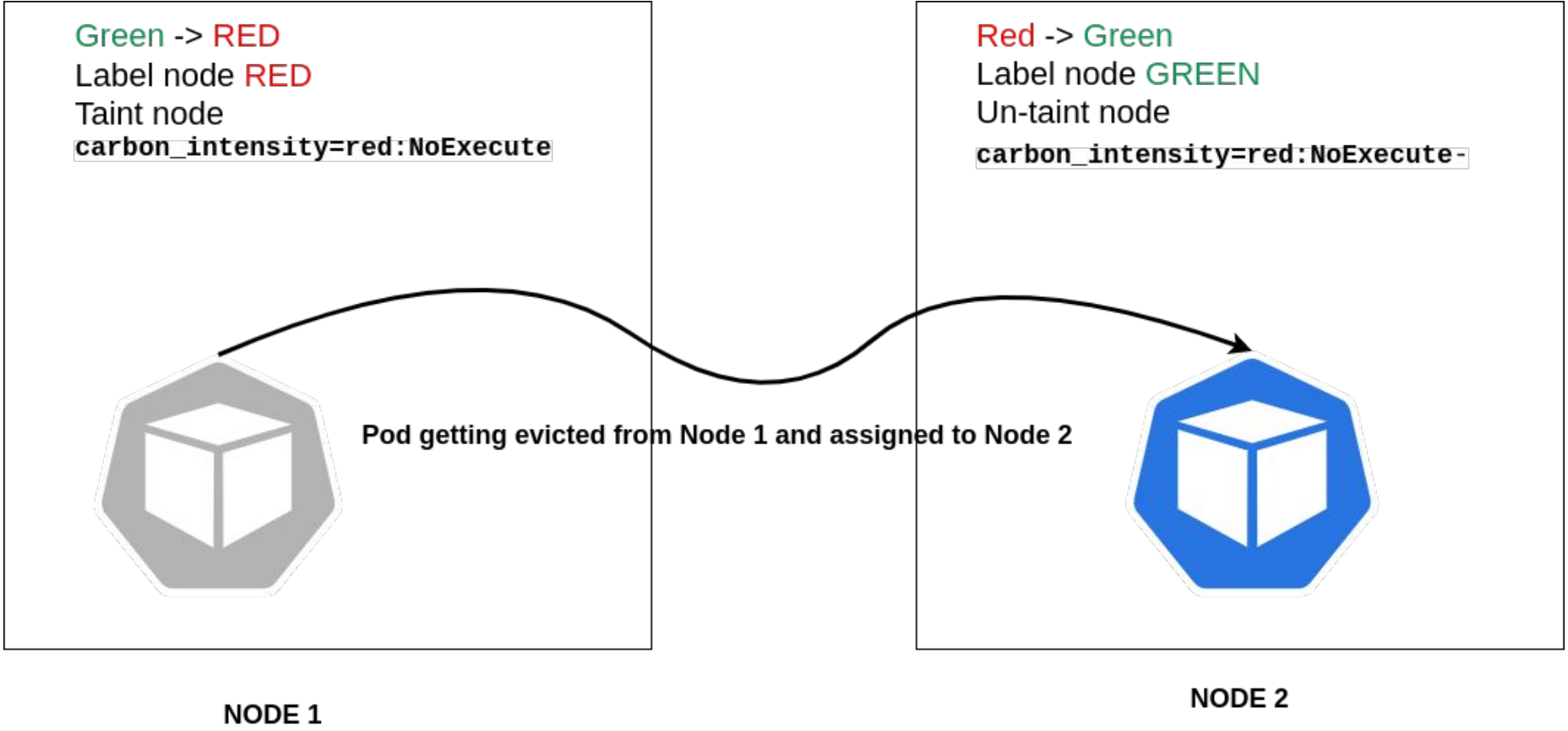


```
spec:
     nodeSelector:
     carbon_intensity: green
tolerations
- key: "carbon_intensity"
       value: "red"
       effect: "NoExecute"
       tolerationSeconds: 5
```

tolerationSeconds

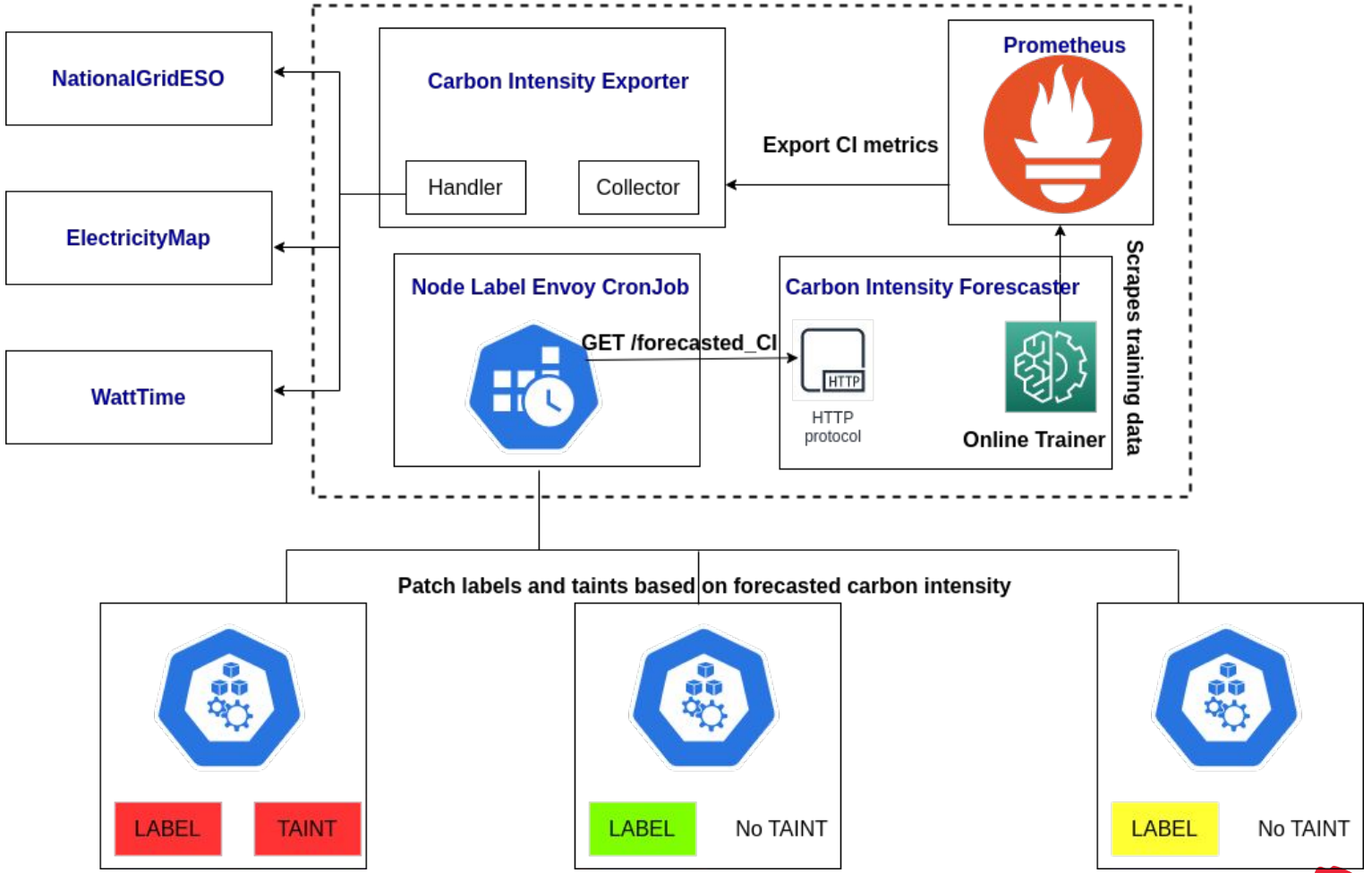
means that if this pod is running and a matching taint is added to the node, then the pod will stay bound to the node for 5 seconds, and then be evicted.

Red Hat



Tainting nodes ensure pods are evicted by the nodes if pods have no tolerations for the taint.

Red Hat


NationalGridESO
ElectricityMap
WattTime
Carbon Intensity Exporter
Handler
Collector
Export CI metrics
Prometheus
Scrapes training data
Node Label Envoy CronJob
GET /forecasted_CI
Carbon Intensity Forecaster
HTTP
HTTP protocol
Online Trainer
Patch labels and taints based on forecasted carbon intensity
LABEL
TAINT
LABEL
No TAINT
LABEL
No TAINT

Red Hat

Demo - Lessons Learnt

- Finding Zone Carbon Intensity Data
  - Some time points are missing

Red Hat

Demo - Lessons Learnt

- Finding Zone Carbon Intensity Data
- Need to support multiple query types
  - It is easy to query threshold friendly metric on Prometheus (e.g. what is the current or average carbon intensity in zone XYZ?), but hard on others (no threshold or more complicated logic)
    - Which zone has the lowest carbon intensity?
    - Is the current carbon intensity low, e.g. within the past 24 hours?

Red Hat

Demo - Lessons Learnt

- Finding Zone Carbon Intensity Data
- Need to support multiple query types
  - It is easy to query threshold friendly metric on Prometheus (e.g. what is the current or average carbon intensity in zone XYZ?), but hard on others (no threshold or more complicated logic)
    - Which zone has the lowest carbon intensity?
    - Is the current carbon intensity low, e.g. within the past 24 hours?
- Need to support multiple electricity carbon emission providers
  - Improve and integrate with Green Software Foundation carbon-aware SDK

Red Hat

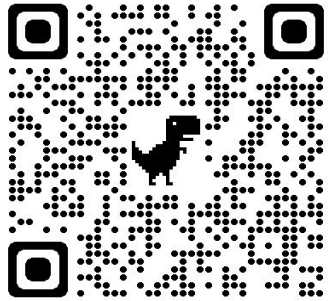
Road Ahead

- Apply to multi-cluster
  - Explore approach with kcp
- Integrate carbon-intensity awareness in kubernetes-sigs/scheduler-plugins
  - Use Trimaran TargetLoadPacking profile and integrate carbon-intensity awareness in the scheduler
  - Tune Trimaran for energy efficiency.

Red Hat

References

- How to use performance counters to estimate power consumption by cpu, memory, etc https://lca.ece.utexas.edu/pubs/bircher-TC2012.pdf
- Kepler : https://github.com/sustainable-computing-io/kepler
- The Model Server: https://github.com/sustainable-computing-io/kepler-model-server



Red Hat

Thank You