
KubeCon
CloudNativeCon
North America 2022
BUILDING FOR THE ROAD AHEAD
DETROIT 2022
TiKV
TiKV
HELM

Sustainability Research *the Cloud Native Way*

KubeCon | CloudNativeCon
North America 2022
BUILDING FOR THE ROAD AHEAD
DETROIT 2022

KubeCon | CloudNativeCon
North America 2022
BUILDING FOR THE ROAD AHEAD
DETROIT 2022

October 24-28, 2021

Chen Wang
IBM Research
@wangchen615

Huamin Chen
Red Hat
@root_fs

Cloud Native Sustainability

Energy Efficiency = MC^3

Metrics

Container Level Power Metrics Reporting

Project KEPLER

KEPLER (Kubernetes-based Efficient Power Level Exporter) uses eBPF to probe energy related system stats and exports as Prometheus metrics

Correlation

Energy Efficienciency Aware Workload Scheduling

Project PEAKS

PEAKS (Power Efficiency Aware Kubernetes Scheduler) Scheduler uses metrics exported by KEPLER and correlation between energy efficiency, resource utilization, and hardware characteristics to schedule workload to achieve optimal performance per Watt

Correction

Online Workload Tuning for Energy Efficienciency

Project CLEVER

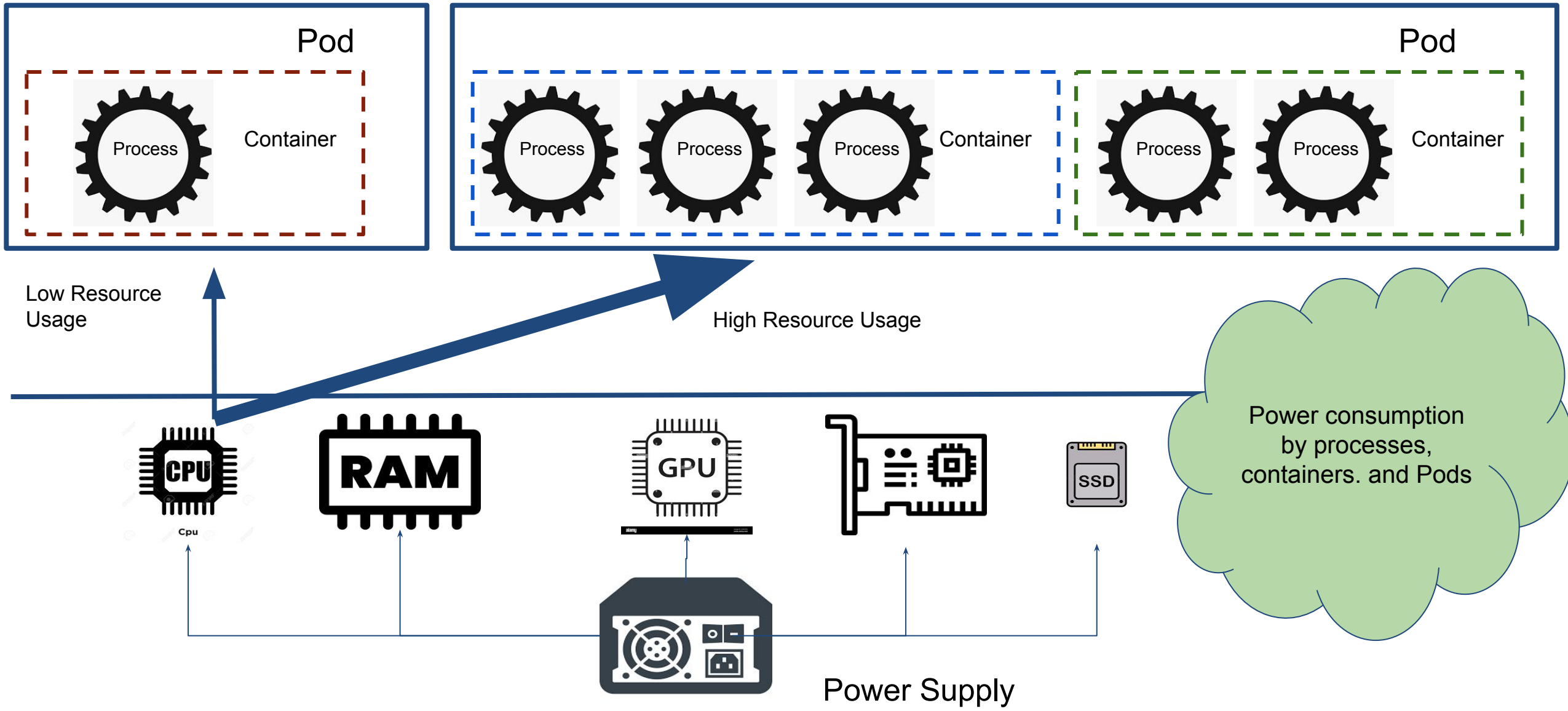
CLEVER (Container Level Energy-efficient VPA Recommender) uses ML models to predict and correct Pod resource usage to optimize performance per Watt

Carbon

Carbon Aware Workload Dispatching

Workloads are dispatched to zones that have low current and projected carbon emission footprint to ensure the sustainability objectives are retained during the workloads' lifecycle.

Cloud Native Power Measurement: Accuracy, Fair, Multi-granular

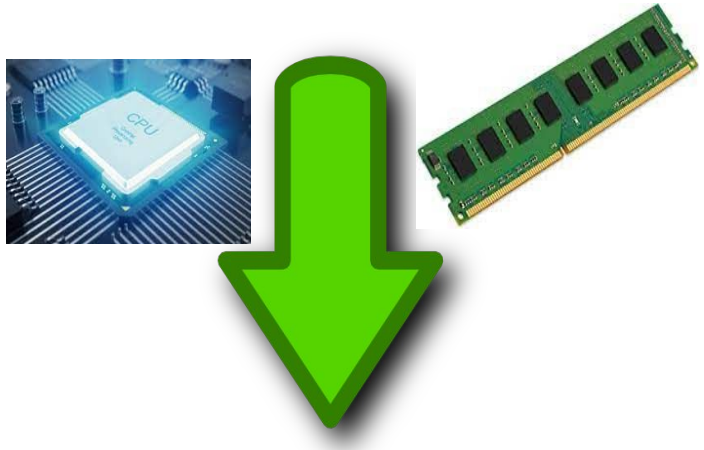


Kepler: Kubernetes-based Efficient Power Level Exporter

https://github.com/sustainable-computing-io/kepler

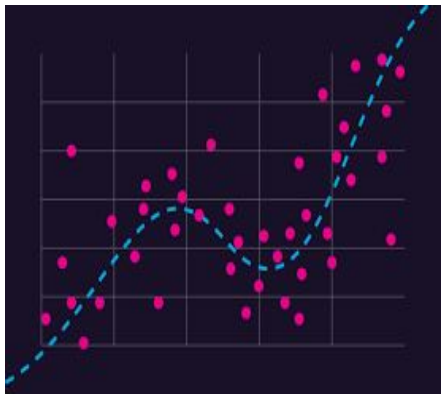
Reporting

- 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**

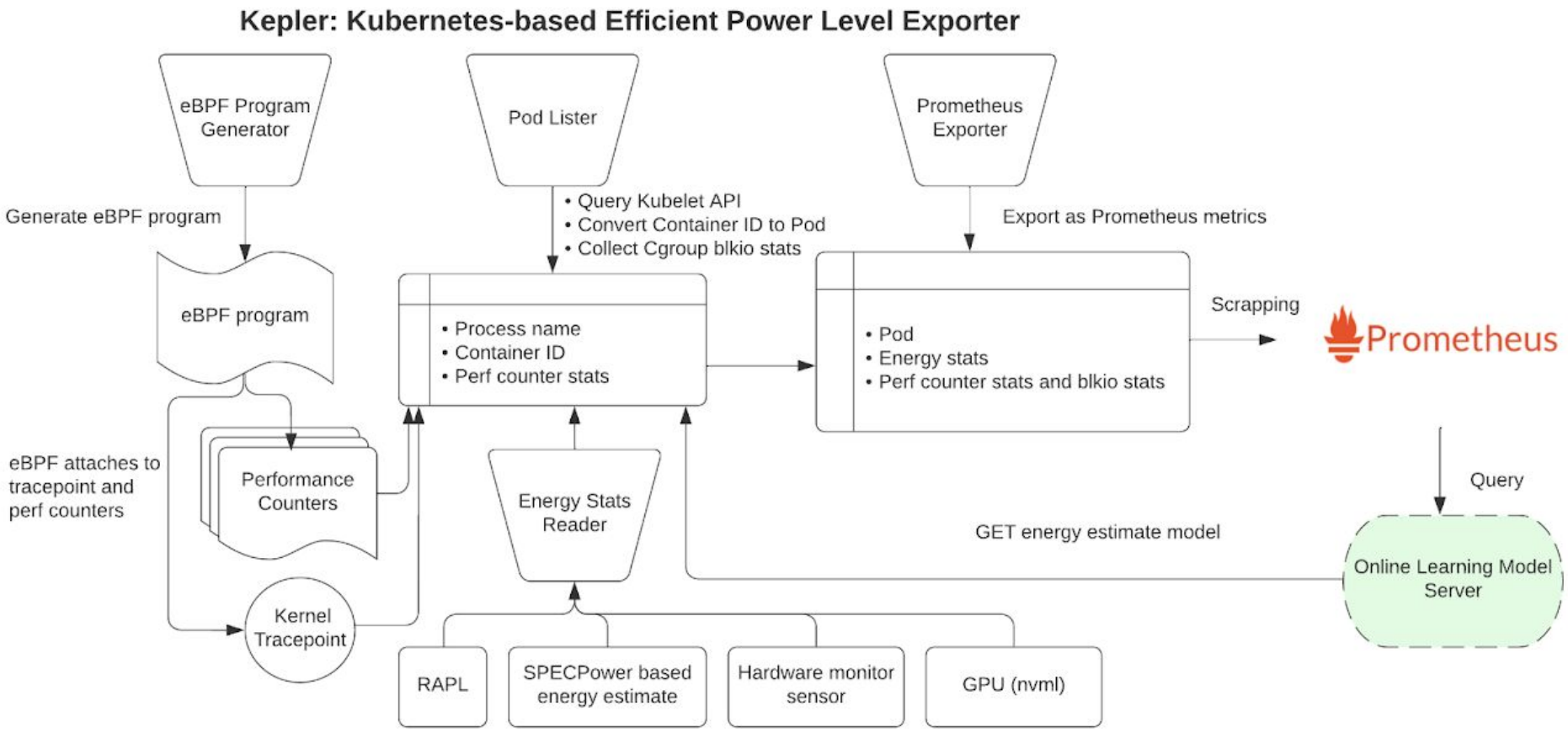


Regression

- Scientific research based **ML** models

$$\sum_{i=1}^{NumCPUs} 9.25 + (35.7 - 9.25) \times PercentActive_i + 4.31 \times \frac{FetchedUops_i}{Cycle}$$

Kepler Design Overview



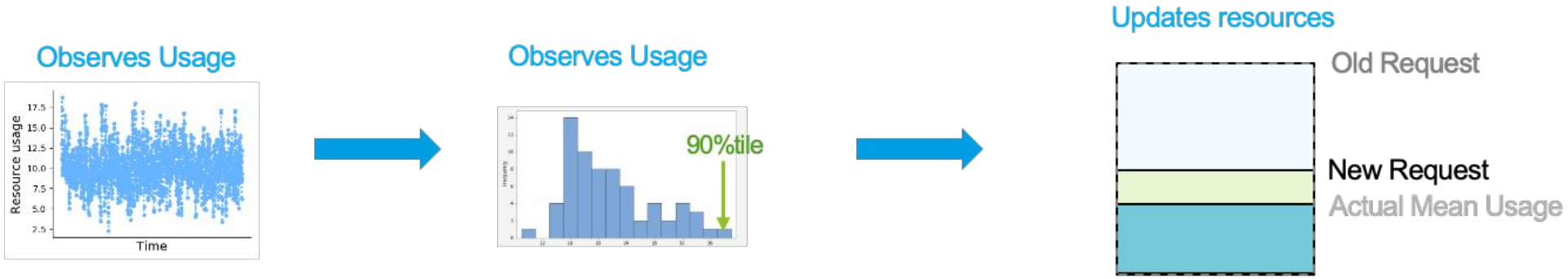
Kepler Dashboard

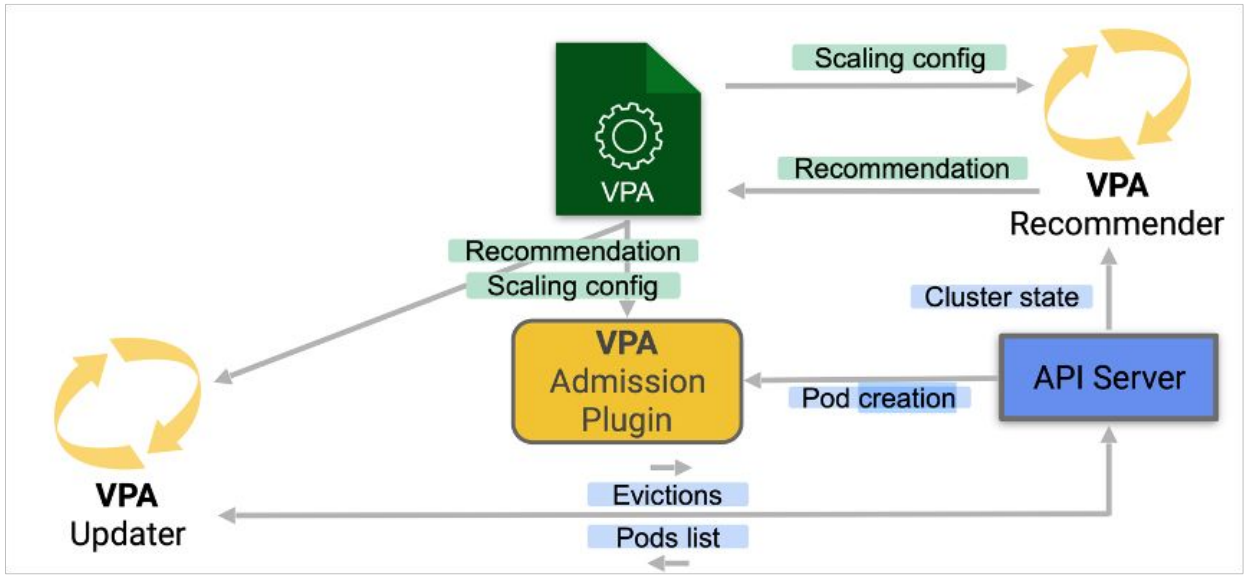
Kepler is currently being contributed by developers from the following companies, in chronological order. It has applied for CNCF Sandbox

Problem: Frequency Awareness in Vertical Scaling

When frequency tuning controllers increase/decrease CPU frequencies, how can we guarantee the performance of workload stay the same?

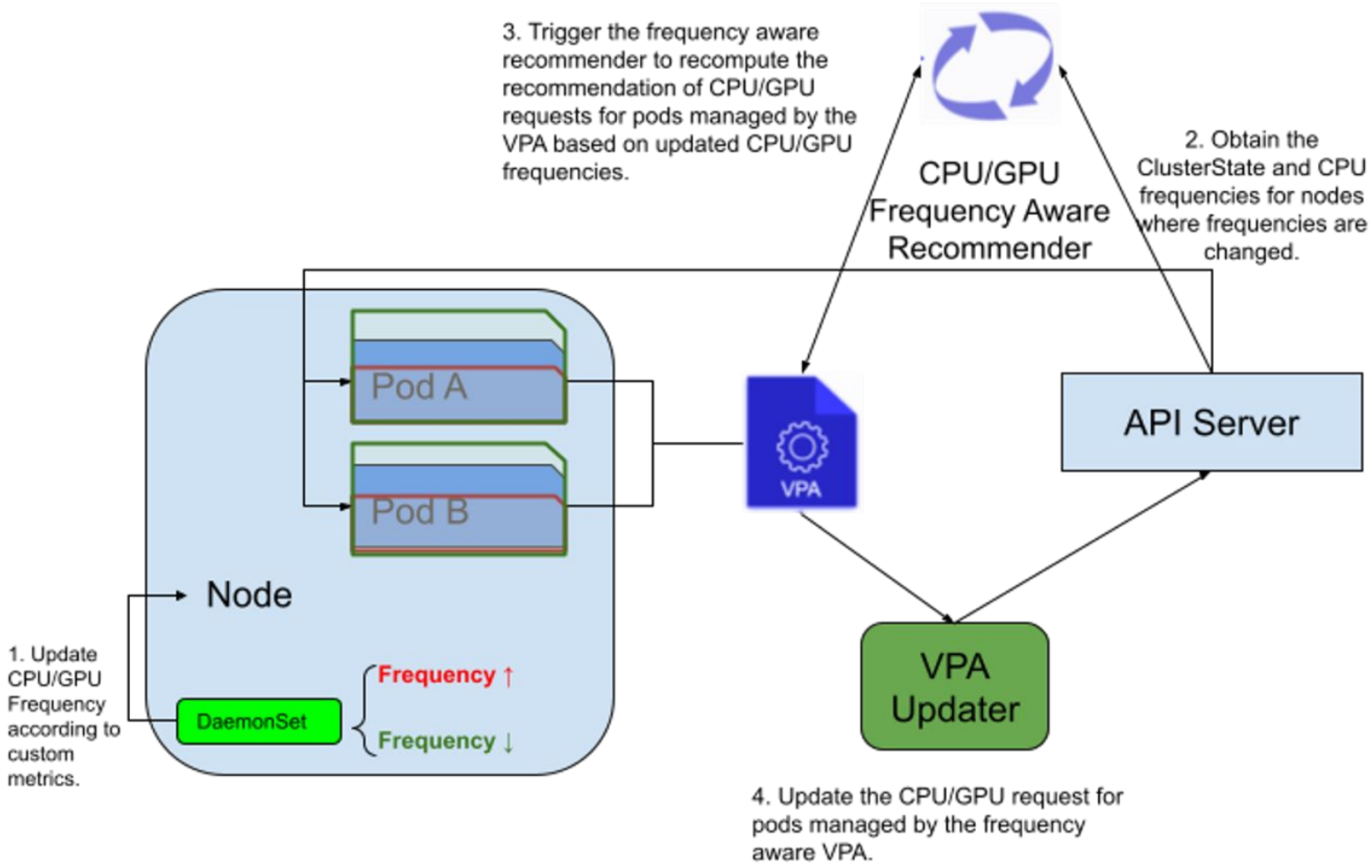
Background: Vertical Pod Autoscaler on Kubernetes





[1] Kubernetes Vertical Pod Autoscaler, https://github.com/kubernetes/autoscaler/tree/master/vertical-pod-autoscaler

CLEVER: Container Level Energy-Efficient VPA Recommender



Carbon footprint can be reduced via improving the objective of performance per watt for containers, namely finding efficient way to guarantee a certain QoS/performance for a workload.

- Lower Frequency: Reducing Energy Consumption.
- Increase CPU allocation: guarantee QoS

What is target Performance/QoS?

Computation Intensive Workload

$$IPS = \frac{Frequency_{CPU}}{N_{cycles}/instruction} \times Request_{CPU}$$

Target Performance/QoS

$$IPS_{target} = \frac{Frequency_{Max}}{N_{cycles}/instruction} \times Request_{default}$$

==

Actual Performance/QoS

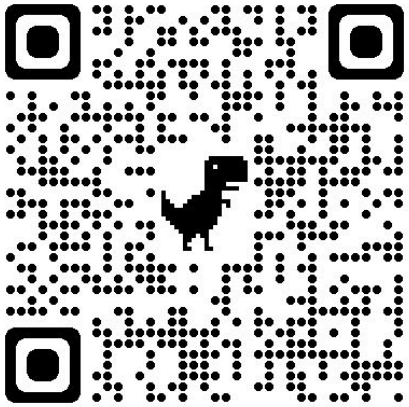
$$IPS_{current} = \frac{Frequency_{current}}{N_{cycles}/instruction} \times Request_{current}$$

KEPLER

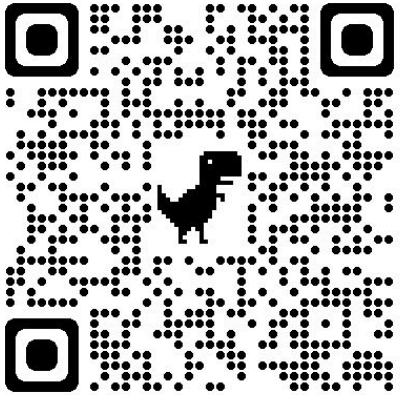
$$Request_{current} = \frac{Frequency_{Max}}{Frequency_{current}} \times Request_{default}$$

Demo

1. Deploy Kepler
2. Show Kepler dashboard
3. Deploy VPA
4. Deploy Clever Recommender
5. Test workload using VPA with Clever Recommender



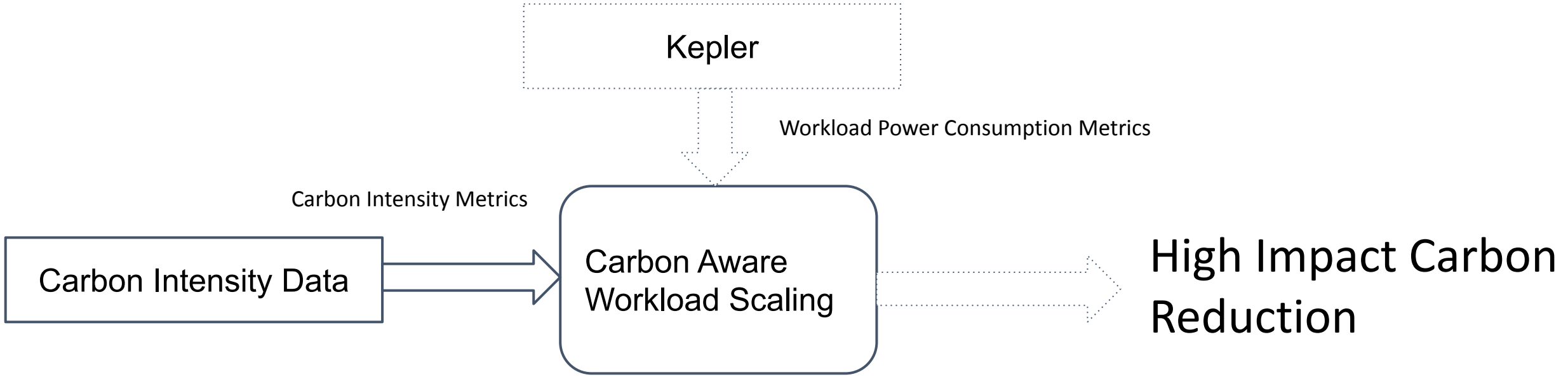
Kepler



Clever


HELP process_open_fds Number of open file descriptors.
TYPE process_open_fds gauge
process_open_fds 58
HELP process_resident_memory_bytes Resident memory size in bytes.
TYPE process_resident_memory_bytes gauge
process_resident_memory_bytes 1.677312e+07
HELP process_start_time_seconds Start time of the process since unix epoch in seconds.
TYPE process_start_time_seconds gauge
process_start_time_seconds 1.666508707083e+09
HELP process_virtual_memory_bytes Virtual memory size in bytes.
TYPE process_virtual_memory_bytes gauge
process_virtual_memory_bytes 2.69430016e+08
HELP process_virtual_memory_max_bytes Maximum amount of virtual memory available in bytes.
TYPE process_virtual_memory_max_bytes gauge
process_virtual_memory_max_bytes 1.8446744073709552e+19
HELP promhttp_metric_handler_requests_in_flight Current number of scrapes being served.
TYPE promhttp_metric_handler_requests_in_flight gauge
promhttp_metric_handler_requests_in_flight 1
HELP promhttp_metric_handler_requests_total Total number of scrapes by HTTP status code.
TYPE promhttp_metric_handler_requests_total counter
promhttp_metric_handler_requests_total{code="200"} 0
promhttp_metric_handler_requests_total{code="500"} 0
promhttp_metric_handler_requests_total{code="503"} 0
[root@clever Kepler]# cd ~/go/src/k8s.io/
[root@clever k8s.io]# ls
[root@clever k8s.io]# git clone https://github.com/kubernetes/autoscaler.git
Cloning into 'autoscaler'...
remote: Enumerating objects: 160970, done.
remote: Counting objects: 100% (2591/2591), done.
remote: Compressing objects: 100% (1072/1072), done.
remote: Total 160970 (delta 1456), reused 684 (delta 618), pack-reused 158379
Receiving objects: 100% (160970/160970), 177.83 MiB | 10.60 MiB/s, done.
Resolving deltas: 100% (101254/101254), done.
Updating files: 100% (28427/28427), done.
[root@clever k8s.io]#
Create an account or view all your
systems at https://red.ht/insights-da
shboard
Last failed login: Mon Oct 24 15:26:
50 CDT 2022 from 61.177.172.50 on s
sh:notty
There were 205 failed login attempts
since the last successful login.
Last login: Mon Oct 24 14:40:39 2022
from 12.27.2.15
[root@clever ~]# kubectl --namespace
monitoring port-forward svc/prometh
eus-k8s 9090
Forwarding from 127.0.0.1:9090 -> 90
90
Forwarding from [::1]:9090 -> 9090
Handling connection for 9090
Handling connection for 9090
monitoring port-forward svc/grafana
3000:channel 3: open failed: connect
failed: Connection refused
Forwarding from 127.0.0.1:3000 -> 30
00
Forwarding from [::1]:3000 -> 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000

Beyond Power Reporting



KEDA: Carbon Aware Scaling PoC

Looking forward

Carbon	Reduce carbon footprint by carbon aware scheduling and scaling. - At what carbon intensity to start the compute job? - At which region to start the compute job? - How many replica to scale under different carbon intensity?
Energy	Measure, manage, and optimize workload energy consumption - How much energy consumed by the workloads? - How to improve the workload performance per watt?
Water	Reduce water usage in Data Centers - Reduce heat and cooling cost, improve workload energy efficiency - Adopt metrics aware workload scheduling and scaling
Waste	Reduce overhead in compute and storage, extend device lifetime - Reduce resource requirement for workload and systems - Reduce redundancy in compute and storage