A Multi-Cloud Infrastructure with Kubernetes, Prometheus and other CNCF Projects

Ricardo Rocha
Computing Engineer, CERN

Data Science Portugal

Ricardo Rocha

Computing Engineer, CERN Cloud Team

- Kubernetes and Containers, Networking and SDN
- GPUs and other Accelerators, Machine Learning

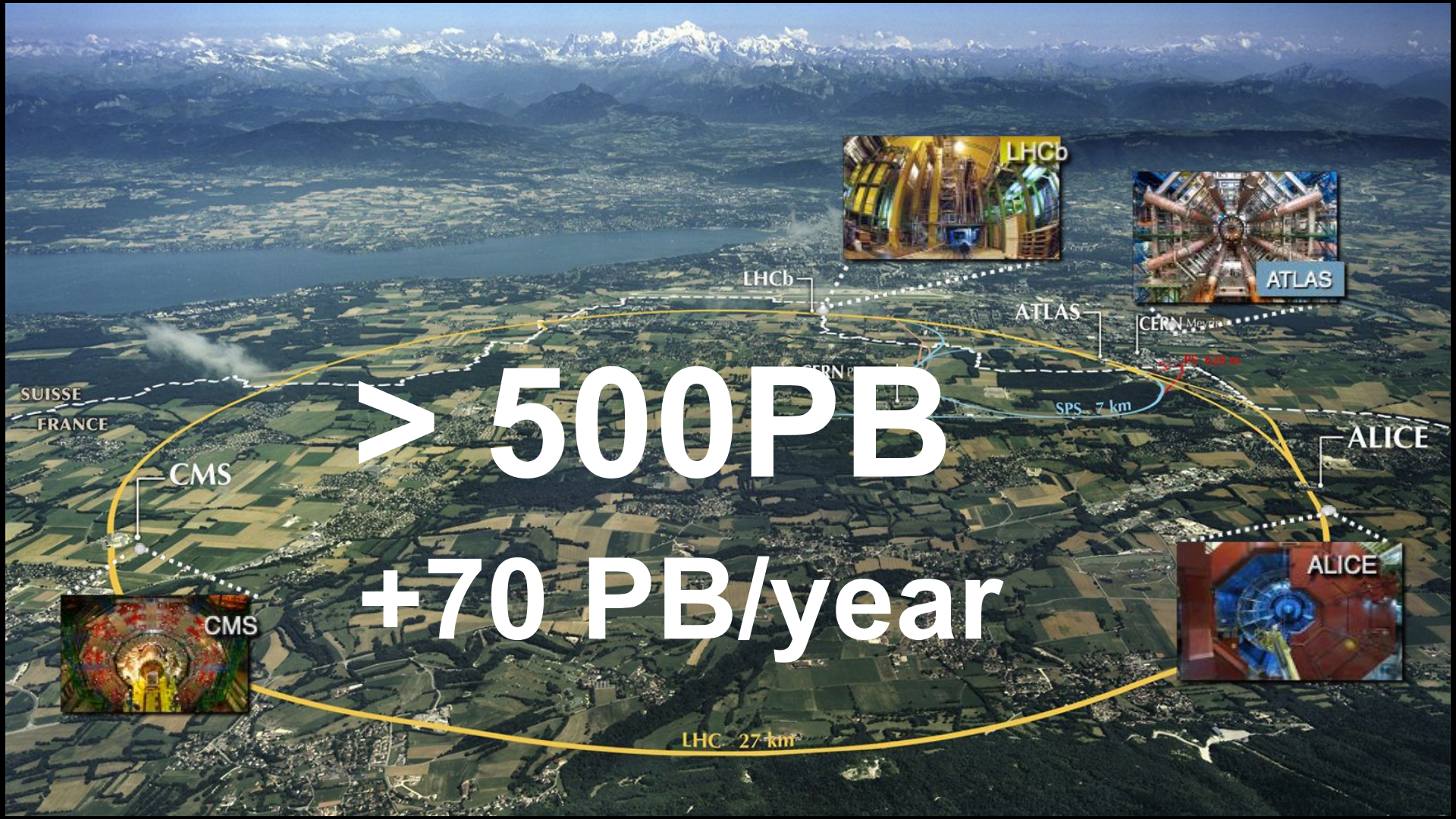
Cloud Native Computing Foundation (CNCF)

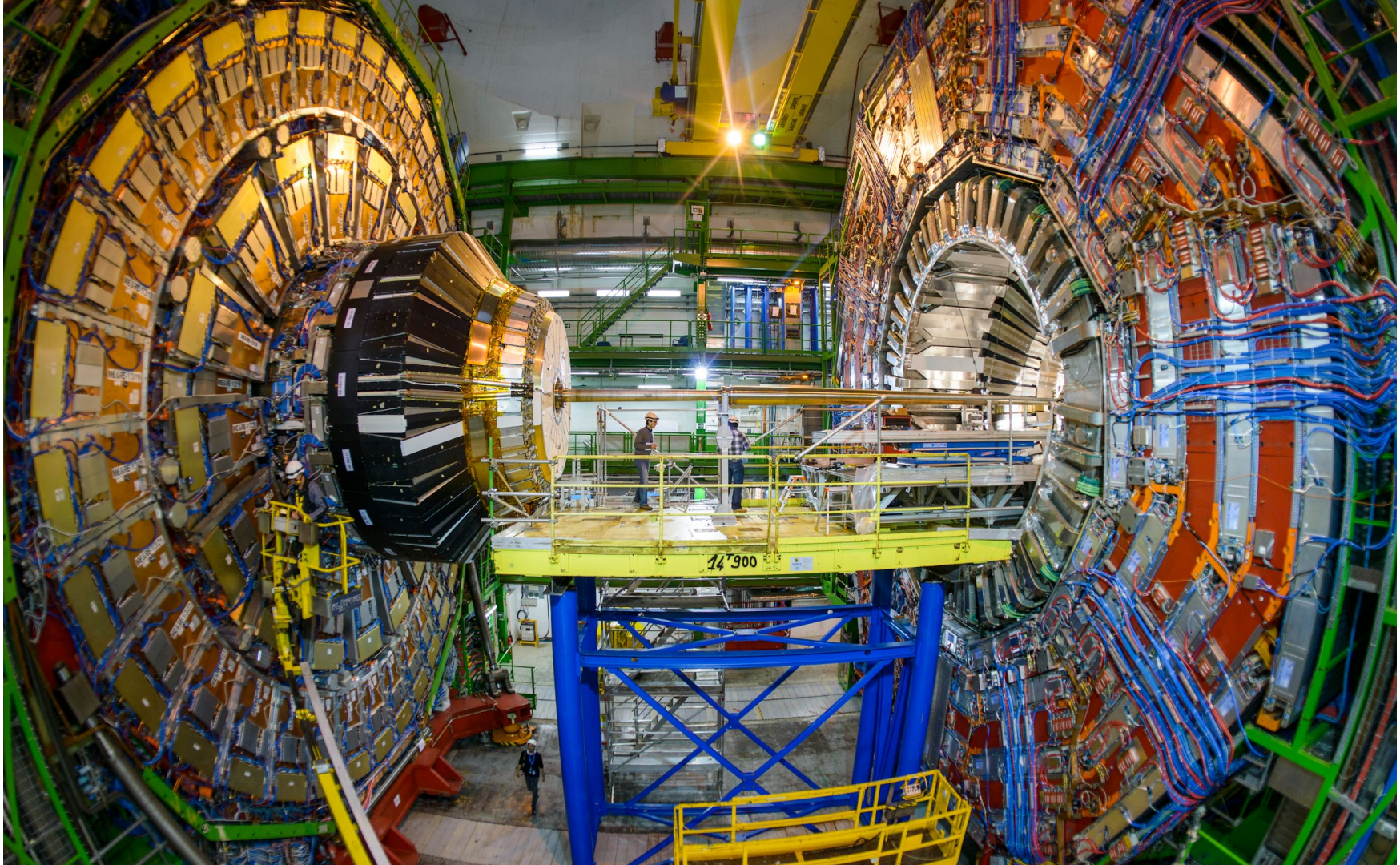
- Representative of CERN in the CNCF and End User Community
- Member of the CNCF Technical Oversight Committee (TOC)
- https://www.cncf.io/people/technical-oversight-committee/
- Lead of the CNCF Research User Group
- https://github.com/cncf/research-user-group

What is 96% of the universe made of?

What was the state of matter just after the Big Bang?

Why isn’t there anti-matter in the universe?


LHCb
ATLAS
ALICE
CMS
SUISSE
FRANCE
SPS 7 km
LHC 27 km
> 500PB
+70 PB/year


14 T 900

CERN OpenStack overview

Last 30 days Refresh every 15m

Cloud resources

Used	Available	Used	Available	Used	Available
279.3 K cores	283.0 K cores	754.8 TiB RAM	915.1 TiB RAM	9.0 PiB disk	14.8 PiB disk

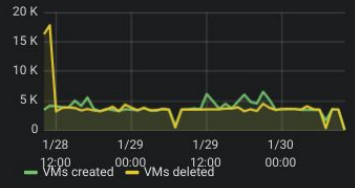
Openstack services stats

Users	Projects	VMs	Magnum clusters	Hypervisors	Images	Baremetal nodes
3528	4609	34458	423	8746	3260	1562

Volumes	Volume size	Fileshares	Fileshares size
6278	1.95 PiB	412	230 TiB

Resource overview by time

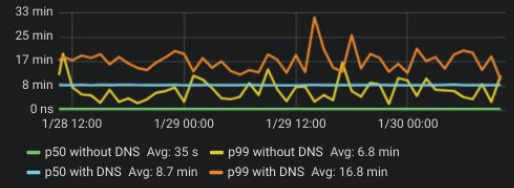
VMs created/deleted



Shared cells availability

Total VMs

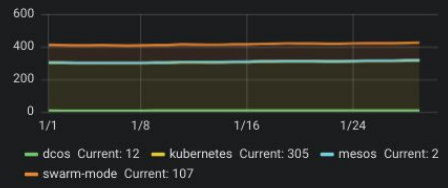
Average VM boot time



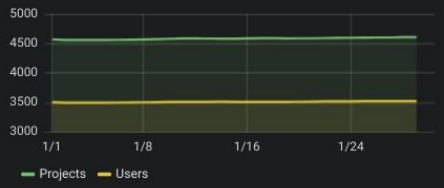
VM changes

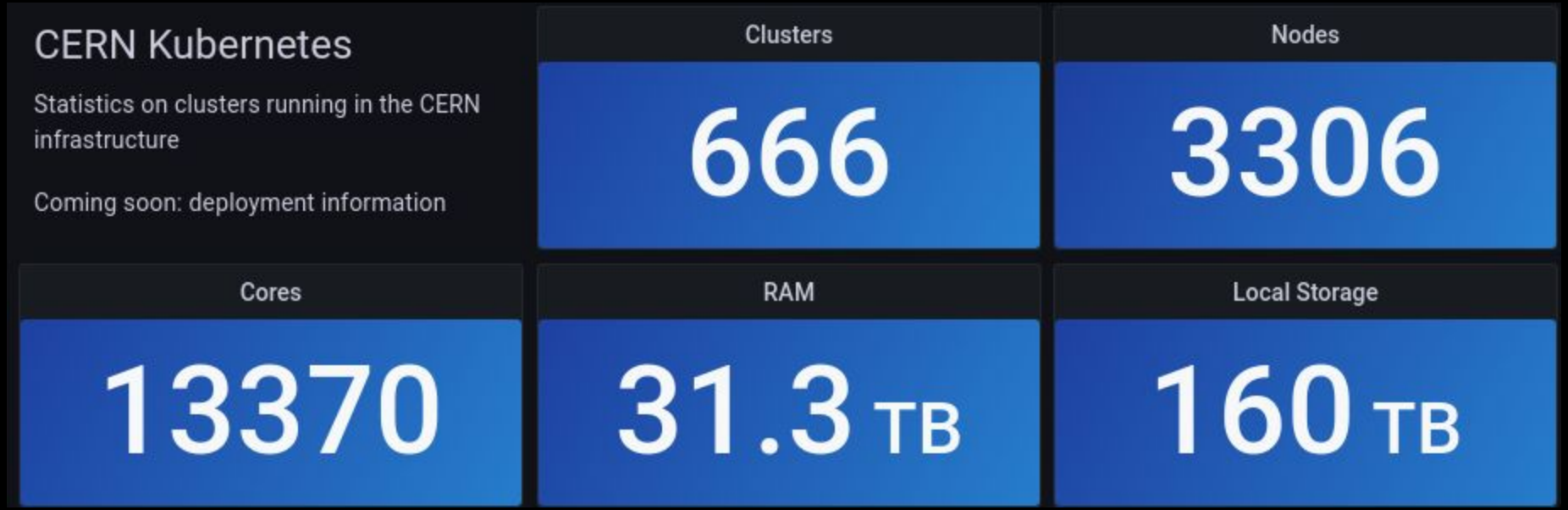
Hypervisors

Magnum clusters



Projects and users




CERN Kubernetes
Statistics on clusters running in the CERN infrastructure
Coming soon: deployment information
Clusters
666
Nodes
3306
Cores
13370
RAM
31.3 TB
Local Storage
160 TB

Kubernetes

Lingua franca of the cloud

Managed services offered by all major public clouds

Multiple options for on-premise or self-managed deployments

Common declarative API for basic infrastructure : compute, storage, networking

Healthy ecosystem of tools offering extended functionality

Public Cloud Motivation

Cover for **periodic load spikes**

Access to a (much) larger amount of otherwise scarce resources: **GPUs**

Access resources not available on premises: **TPUs**, IPUs

Evaluate resources in advance, prepare for tenders: GPUs, ARM, …

Disaster Recovery

Previous Work (2019)

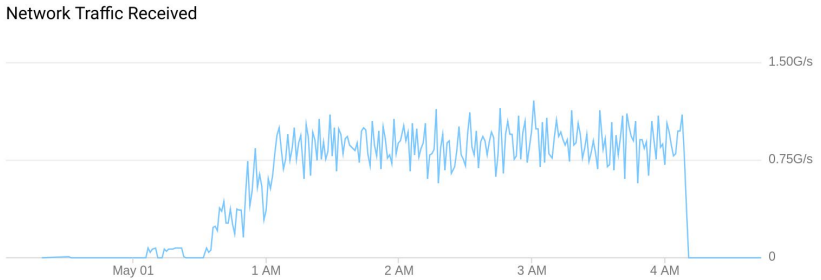
Kubecon Higgs Analysis Demo: **70TB in 5min**

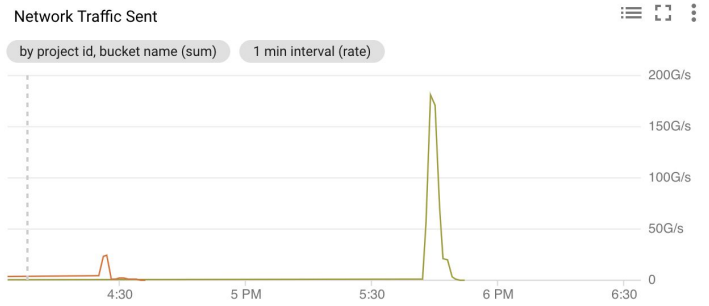
Deep Dive: https://indico.cern.ch/event/828111/

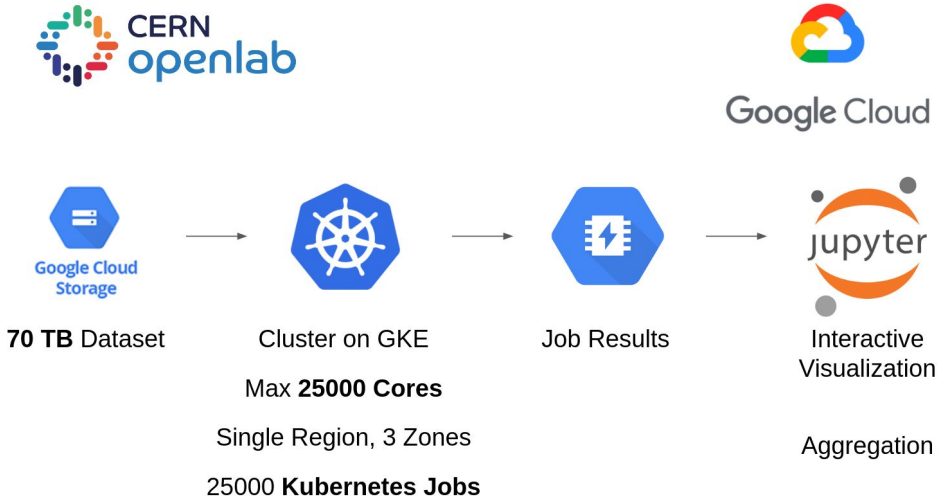
Throughput up to **32Gb per VM**

In memory storage used to handle this

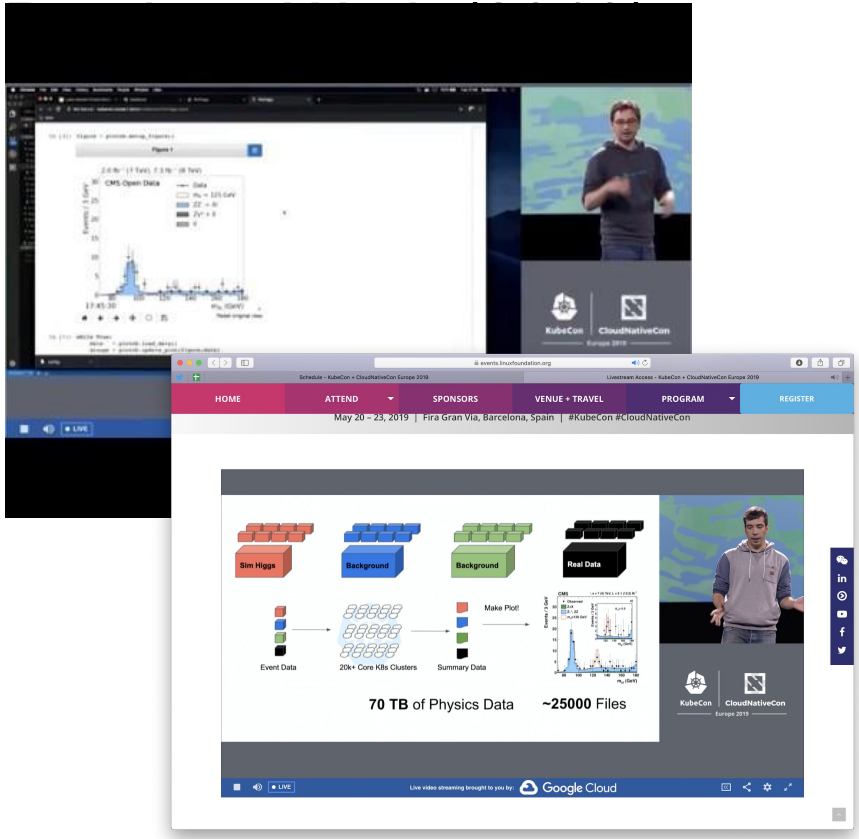
Peak at close to **200GB/s**

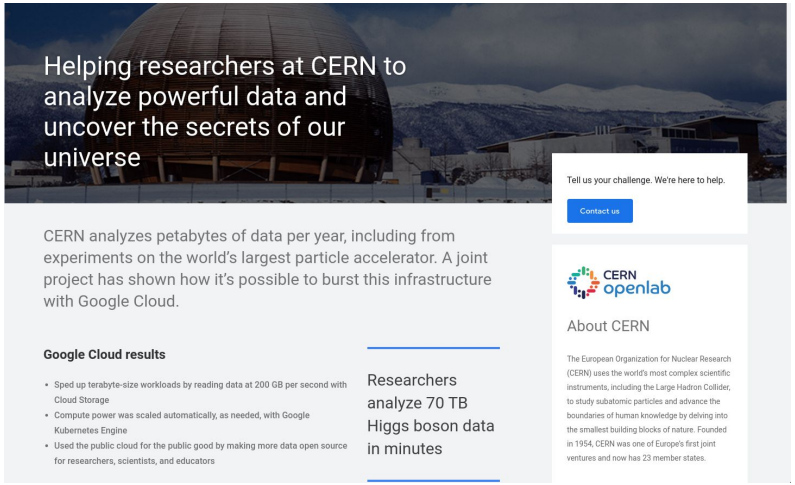


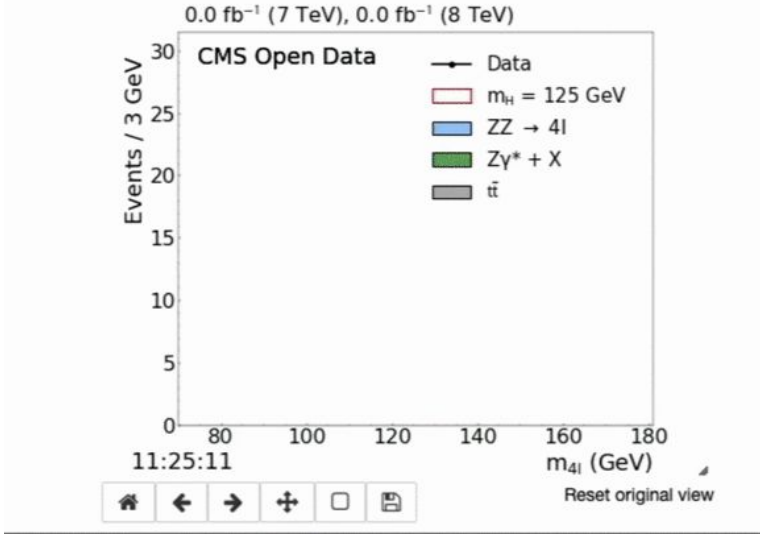




https://cloud.google.com/customers/cern







https://www.youtube.com/watch?v=CTfp2woVEkA

But this got us less than halfway...

Support multiple cloud providers, regions

Integrate with our usual in house end user tools

Centralized monitoring, logging, tracing is a requirement

Accounting, costing at a fine grain level is another

… add here , and here , and here …

Cloud Native

“”” Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

*These techniques enable **loosely coupled systems that are resilient**, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil. “””*

Cloud Native Landscape

v0.9.9

App Definition & Development

Database & Data Analytics | Streaming | SCM | Application Definition | CI/CD

Orchestration & Management

Scheduling & Orchestration | Coordination & Service Discovery | Service Management

Runtime

Cloud-Native Storage | Container Runtime | Cloud-Native Network

Provisioning

Host Management / Tooling | Infrastructure Automation | Container Registries | Secure Images | Key Management

Cloud

Public | Private

Platforms

PaaS / Container Service

Serverless/Event-based

Observability & Analysis

Monitoring

Logging

Tracing

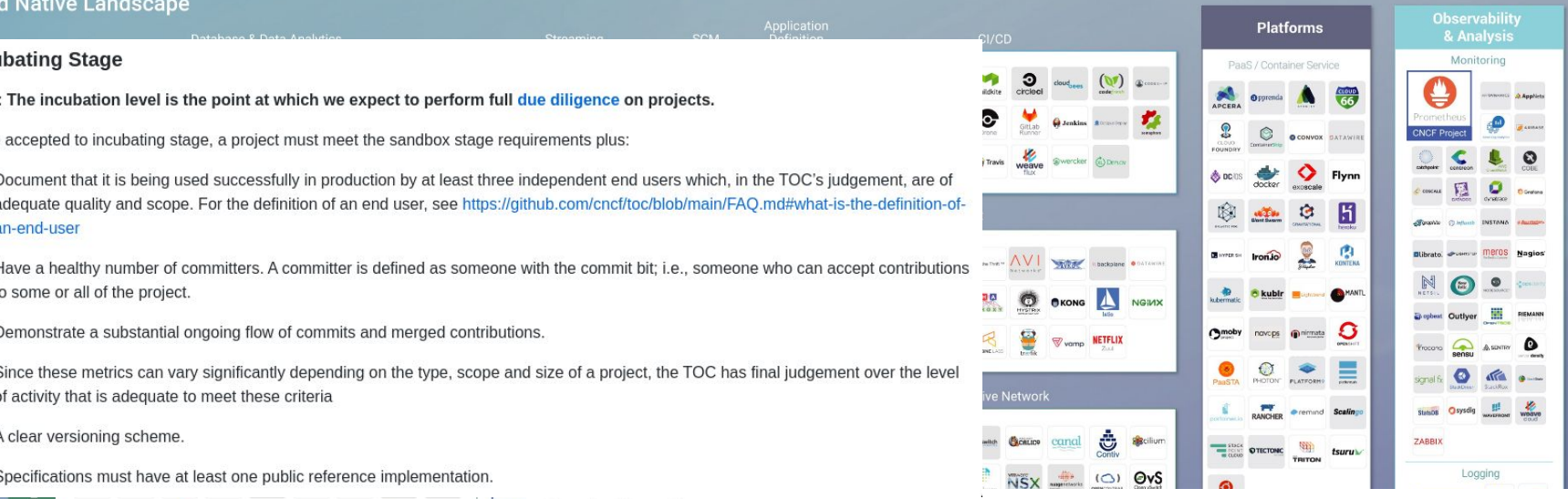
github.com/cncf/landscape

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint | Amplify Partners

Greyed logos are not open source



Incubating Stage

Note: The incubation level is the point at which we expect to perform full due diligence on projects.

To be accepted to incubating stage, a project must meet the sandbox stage requirements plus:

- Document that it is being used successfully in production by at least three independent end users which, in the TOC's judgement, are of adequate quality and scope. For the definition of an end user, see https://github.com/cncf/toc/blob/main/FAQ.md#what-is-the-definition-of-an-end-user
- Have a healthy number of committers. A committer is defined as someone with the commit bit; i.e., someone who can accept contributions to some or all of the project.
- Demonstrate a substantial ongoing flow of commits and merged contributions.
- Since these metrics can vary significantly depending on the type, scope and size of a project, the TOC has final judgement over the level of activity that is adequate to meet these criteria
- A clear versioning scheme.
- Specifications must have at least one public reference implementation.

Graduation Stage

To graduate from sandbox or incubating status, or for a new project to join as a graduated project, a project must meet the incubating stage criteria plus:

- Have committers from at least two organizations.
- Have achieved and maintained a Core Infrastructure Initiative Best Practices Badge.
- Have completed an independent and third party security audit with results published of similar scope and quality as the following example (including critical vulnerabilities addressed): https://github.com/envoyproxy/envoy#security-audit and all critical vulnerabilities need to be addressed before graduation.
- Explicitly define a project governance and committer process. This preferably is laid out in a GOVERNANCE.md file and references an OWNERS.md file showing the current and emeritus committers.
- Have a public list of project adopters for at least the primary repo (e.g., ADOPTERS.md or logos on the project website). For a specification, have a list of adopters for the implementation(s) of the spec.

Deployment

Deployments

Automate and manage centrally full deployment

Clusters, base components, end user applications

Support clusters in **multiple clouds, regions**, zones

Include **base services** for monitoring, *tracing*, policy, software distribution, cost, …

Support multiple ways to define workloads / services: **helm, kustomize, plain**, …

Deployment

Includes 3 groups of components

Underlay, handles the cluster(s) deployment and lifecycle

Infrastructure, handles the basic infra services running on all clusters

Services / Applications, the actual workloads running in one or more clusters

All managed in one single git repo and relying on **ArgoCD**

Plain yaml, Helm, Kustomize

Hides a lot of the quite complex deployment details

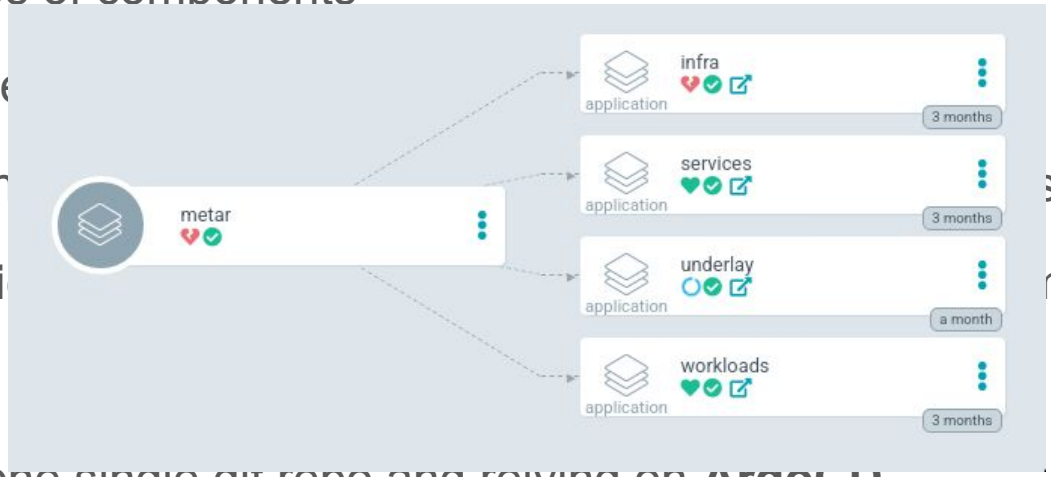
Deployment

Includes 3 groups of components

Underlay, handle

Infrastructure, h

Services / Appli



sters

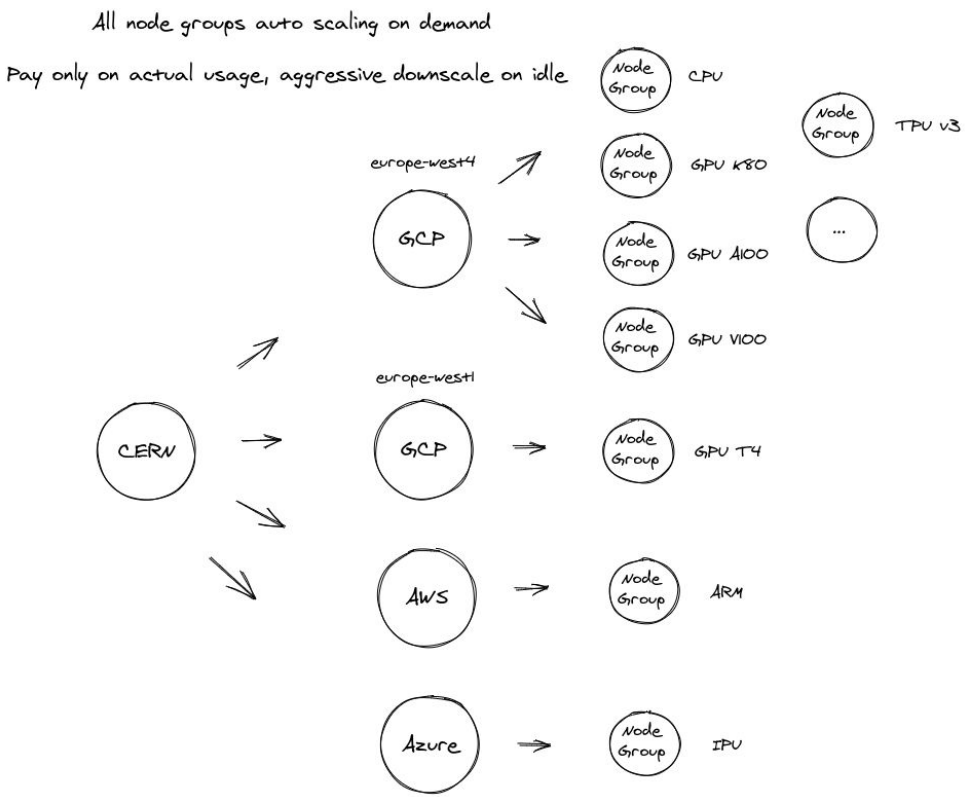
more clusters

All managed in one single git repo and relying on **ArgoCD**

Plain yaml, Helm, Kustomize

Hides a lot of the quite complex deployment details

Deployment: Underlay



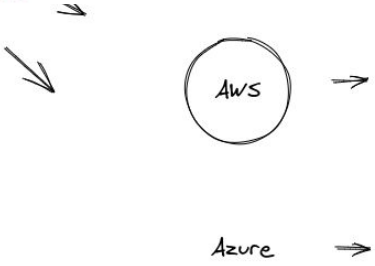

```
clusters:
  gke-europe-west4-a-1:
    cloud: gcp
    autoprovisioned: true
    config: gcpconfig
    providerConfig:
      location: "europe-west4-a"
      enableTpu: true
      initialClusterVersion: "1.18.12-gke.1210"
    pools:
      default:
        diskSize: 120
        diskType: pd-ssd
        machineType: n1-standard-4
        nodeCount: 2
      a100:
        accelerator:
          count: 1
          type: nvidia-tesla-a100
        autoscaling:
          enabled: true
          minCount: 0
          maxCount: 10
        diskSize: 120
        diskType: pd-ssd
        machineType: a2-highgpu-1g
        nodeCount: 0
        preemptible: true
```

Deployment: Underlay

All node groups auto scaling on demand

Pay only on actual usage, aggressive downscale on idle

```yaml
apiVersion: container.gcp.crossplane.io/v1beta1
kind: GKECluster
metadata:
  name: {{ $cluster }}
spec:
  providerConfigRef:
    name: {{ $clusterdata.config }}
  writeConnectionSecretToRef:
    name: {{ $cluster}}-secret
    namespace: default
```




```yaml
apiVersion: container.gcp.crossplane.io/v1alpha1
kind: NodePool
metadata:
  name: {{ $cluster }}-{{ $pool }}
spec:
  providerConfigRef:
    name: {{ $clusterdata.config }}
  forProvider:
    clusterRef:
      name: {{ $cluster }}
    locations:
      - '{{ $clusterdata.location }}'
    initialNodeCount: {{ $pooldata.nodeCount }}
    {{- if $pooldata.autoscaling }}
    autoscaling:
      autoprovisioned: {{ $pooldata.autoscaling.autoprovisioned | default "false" }}
      enabled: {{ $pooldata.autoscaling.enabled | default "false" }}
      maxNodeCount: {{ $pooldata.autoscaling.maxCount }}
      minNodeCount: {{ $pooldata.autoscaling.minCount }}
    {{- end }}
    config:
      {{- if $pooldata.accelerator }}
      accelerators:
        - acceleratorCount: {{ $pooldata.accelerator.count }}
          acceleratorType: {{ $pooldata.accelerator.type }}
      {{- end }}
      diskSizeGb: {{ $pooldata.diskSize }}
      diskType: {{ $pooldata.diskType }}
      imageType: {{ $pooldata.imageType | default "cos_containerd" }}
      {{- if $pooldata.labels }}
      labels:
        {{ $pooldata.labels }}
      {{- end }}
      machineType: {{ $pooldata.machineType }}
```

Crossplane

Get Started on GitHub | Documentation | Slack Channel | Blog | Twitter

Compose cloud infrastructure and services into custom platform APIs

Crossplane is an open source Kubernetes add-on that enables platform teams to assemble infrastructure from multiple vendors, and expose higher level self-service APIs for application teams to consume, without having to write any code.

Get Started on GitHub

Join Slack Channel

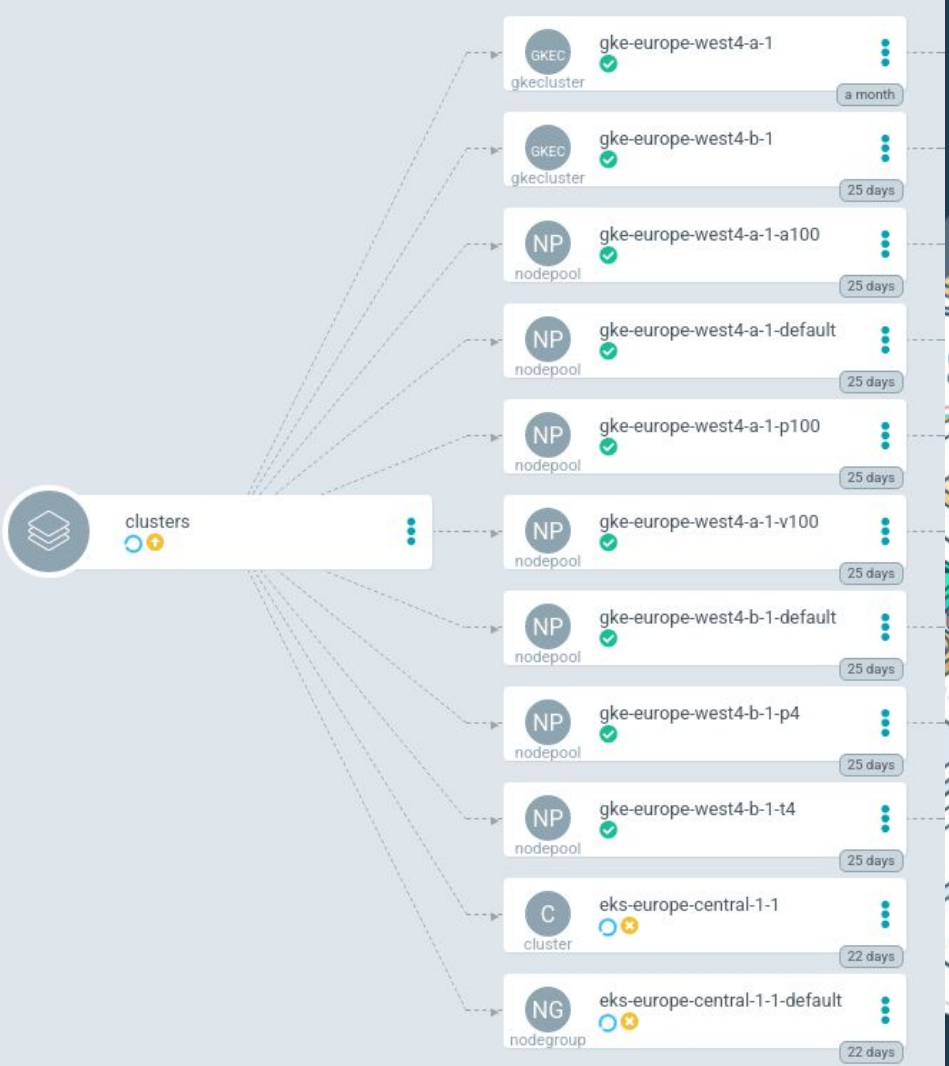
Crossplane

Slack Channel Blog Twitter

Compose c
infrastructu
services int
custom pla
APIs

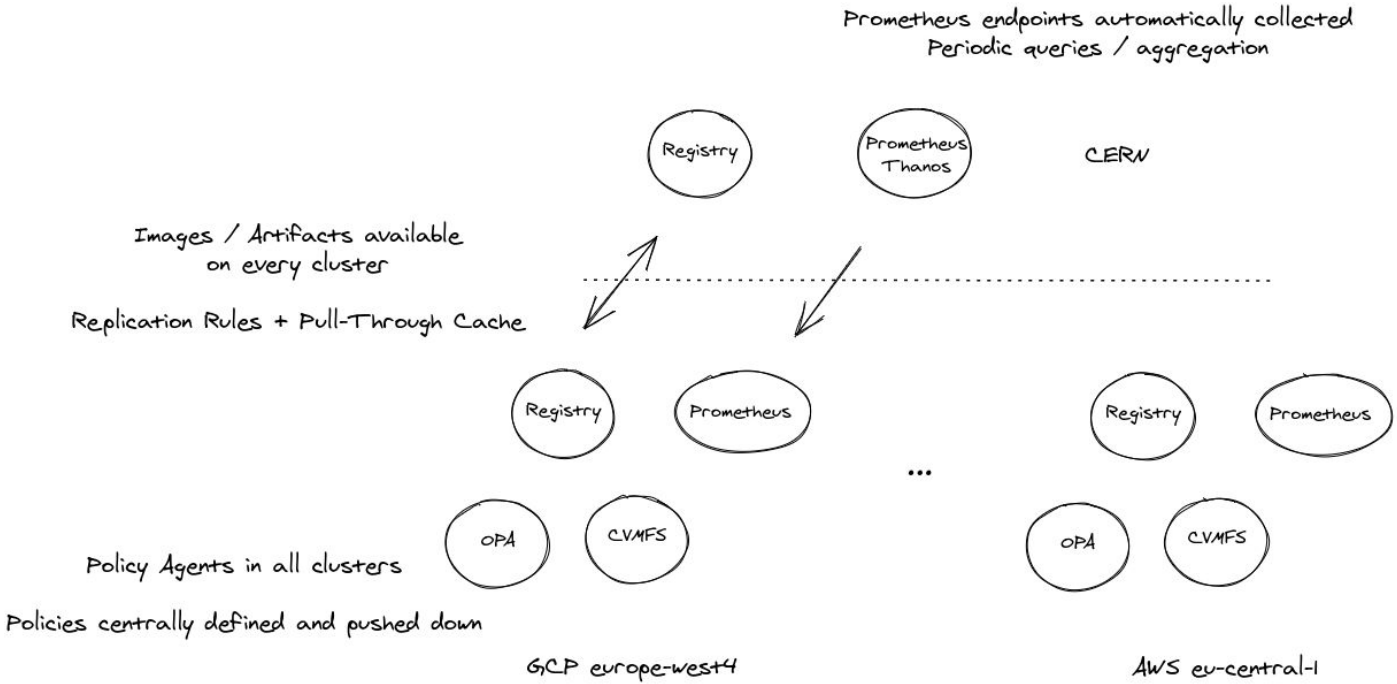
Crossplane is an open sour
that enables platform team
infrastructure from multiple
higher level self-service AP
to consume, without having

Get Started on GitHub



Deployment: Infrastructure

Base services **automatically deployed on all clusters**

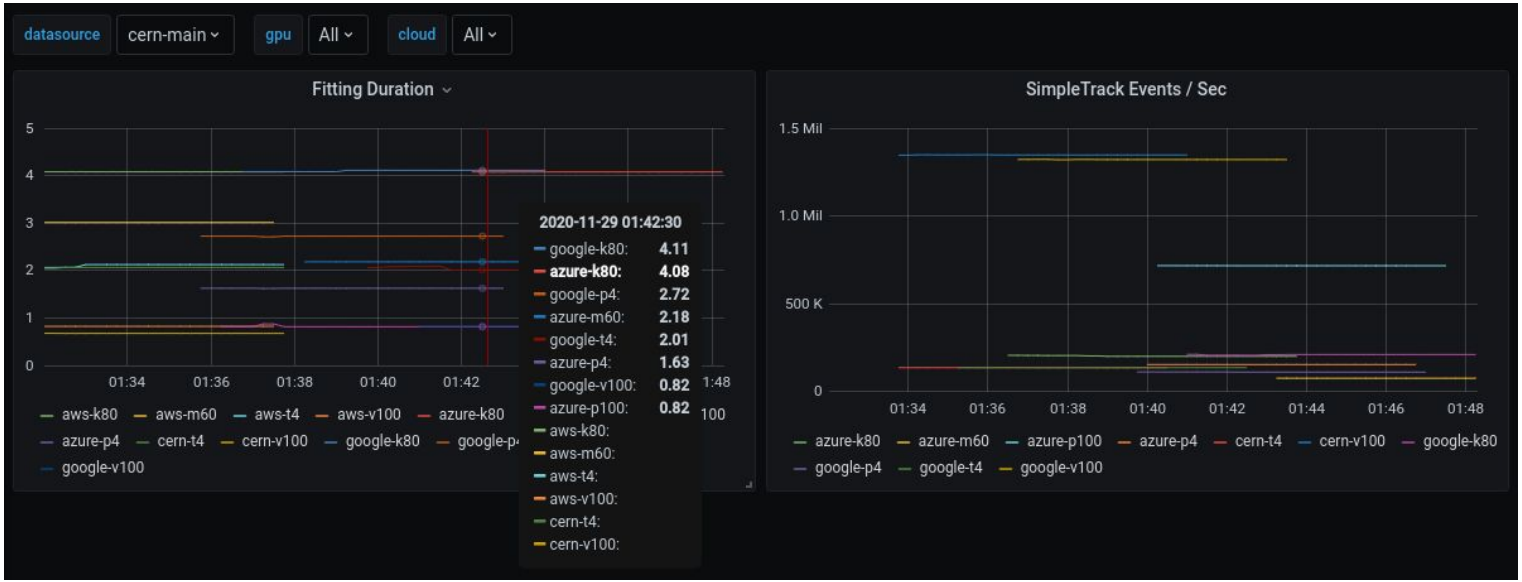


Use Cases

Use Cases: Accelerated Workloads

Measure performance across a wide range of GPUs

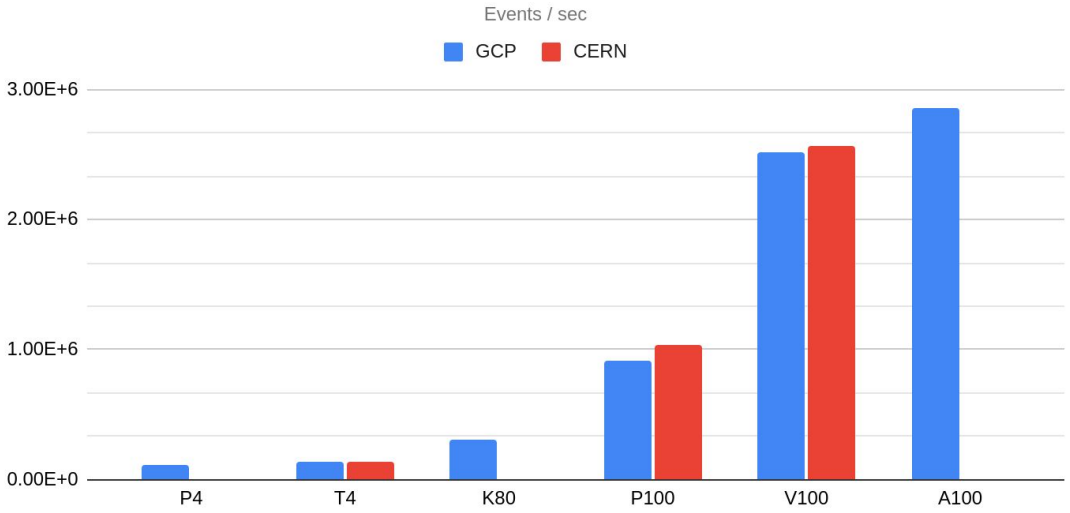
Example with generation of simulation data



Use Cases: Accelerated Workloads

Measure performance across a wide range of GPUs

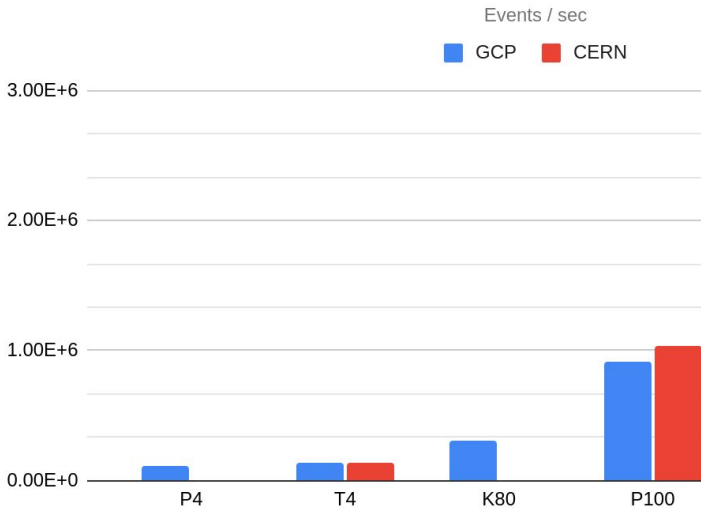
Example with generation of simulation data

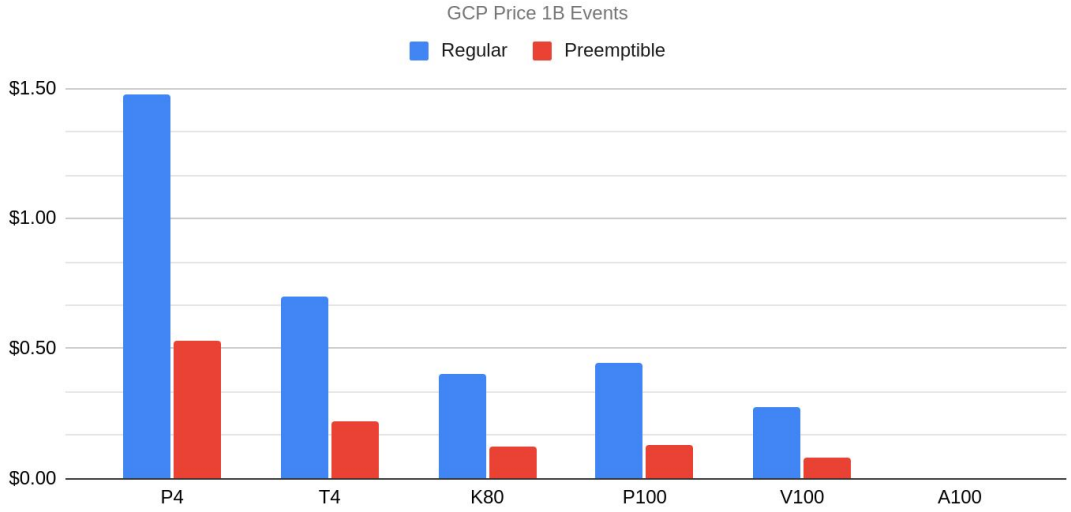


Use Cases: Accelerated Workloads

Measure performance across a wide range of GPUs

Example with generation of simulation data

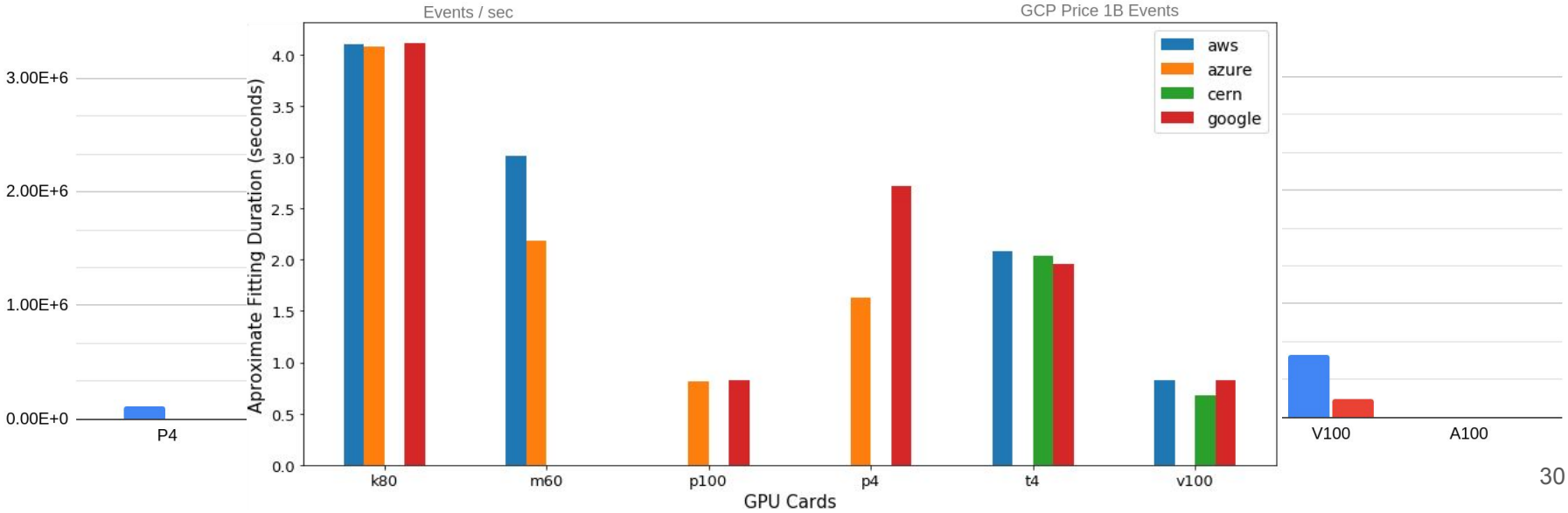




Use Cases: Accelerated Workloads

Measure performance across a wide range of GPUs

Example with generation of simulation data



Use Cases: GitLab Runners

https://gitlab.cern.ch/rbritoda/gitlab-runner-public

Experimental setup with runners using public cloud resources

Cluster(s) auto scaling on demand

- Slight delay (<2min) for job start when a fresh node is required

Large numbers and types of resources available

- **GPUs**: Nvidia A100, V100, P100, K80, P4, T4, M60
- **TPUs**: v2-8, v3-8, can add larger instances if needed
- **ARM**: Graviton2 (m6g.xlarge, m6g.16xlarge, m6g.metal)

Pipeline Needs **Jobs 13** Tests 0

Status	Job ID	Name		Coverage
Test				
passed	#12061346 docker-arm	sample-arm	00:00:05 2 hours ago	
passed	#12061347 docker-arm-graviton2	sample-arm-graviton2	00:00:08 2 hours ago	
passed	#12061338 docker-gpu	sample-gpu	00:00:05 2 hours ago	
passed	#12061339 docker-gpu-nvidia	sample-gpu-nvidia	00:00:15 2 hours ago	
passed	#12061340 docker-gpu-a100	sample-gpu-nvidia-a100	00:00:15 2 hours ago	
passed	#12061345 docker-gpu-k80	sample-gpu-nvidia-k80	00:00:06 2 hours ago	
passed	#12061342 docker-gpu-p100	sample-gpu-nvidia-p100	00:00:05 2 hours ago	
passed	#12061344 docker-gpu-p4	sample-gpu-nvidia-p4	00:00:12 2 hours ago	
passed	#12061343 docker-gpu-t4	sample-gpu-nvidia-t4	00:00:06 2 hours ago	
passed	#12061341 docker-gpu-v100	sample-gpu-nvidia-v100	00:00:12 2 hours ago	
passed	#12061348 docker-tpu	sample-tpu	00:03:03 2 hours ago	
passed	#12061349 docker-tpu-v2-8	sample-tpu-v2-8	00:03:01 2 hours ago	
passed	#12061350 docker-tpu-v3-8	sample-tpu-v3-8	00:02:58 2 hours ago	

Pipeline Needs **Jobs 13** Tests 0

```
Waiting for pod gitlab-runner/runner-hdqec7c-project-108952-concurrent-1fbmnv to be running, status is Pending
       ContainersNotReady: "containers with unready status: [build helper]"
       ContainersNotReady: "containers with unready status: [build helper]"
Running on runner-hdqec7c-project-108952-concurrent-1fbmnv via gitlab-runner-tpu-v3-8-gitlab-runner-5d54df77b-wcgq8...
Getting source from Git repository                                                                    00:01
Fetching changes with git depth set to 50...
Initialized empty Git repository in /builds/rbritoda/gitlab-runner-public/.git/
Created fresh repository.
Checking out 828259f8 as load...
Skipping Git submodules setup
Executing "step_script" stage of the job script                                                       00:12
```

```python
#!/usr/bin/python3
import tensorflow as tf
import os
from tensorflow.python.profiler import profiler_client

endpoint = os.environ['KUBE_GOOGLE_CLOUD_TPU_ENDPOINTS']
print("Connecting to TPU at %s\n" % endpoint)

tpu = tf.distribute.cluster_resolver.TPUClusterResolver()
print('Running on TPU ', tpu.cluster_spec().as_dict()['worker'])

tf.config.experimental_connect_to_cluster(tpu)
tf.tpu.experimental.initialize_tpu_system(tpu)

print(profiler_client.monitor(endpoint.replace('8470', '8466'), 100, 2))
```

```
2021-02-09 21:01:12.702455: I tensorflow/core/distributed_runtime/rpc/grpc_server_lib.cc:390] Started server with target: grpc://localhost:30018
Connecting to TPU at grpc://10.116.18.178:8470
Running on TPU  ['10.116.18.178:8470']
  Timestamp: 21:01:22
  TPU type: TPU v3
  Utilization of TPU Matrix Units (higher is better): 0.000%
Cleaning up file based variables                                                                      00:00
Job succeeded
```

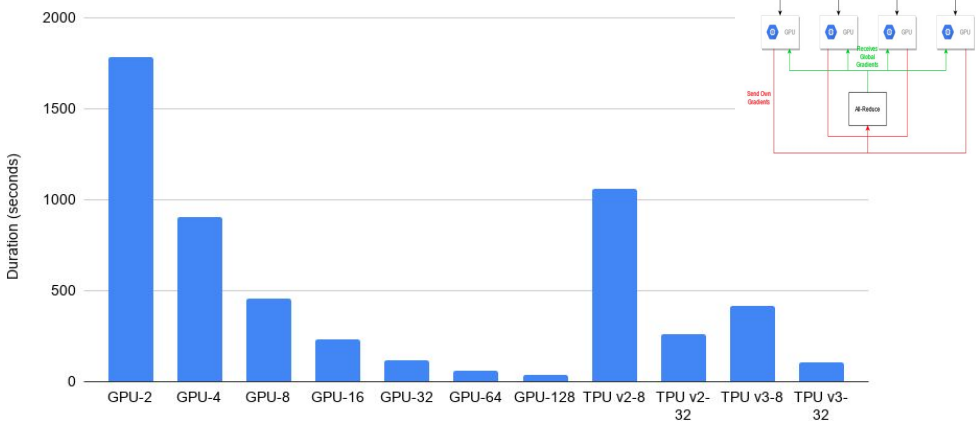
```
Running on runner-yawjs17o-project-108952-concurrent-18ffzd via gitlab-runner-arm-graviton2-gitlab-runner-575dd7ff87-tlpbc...
Getting source from Git repository                                                                                    00:00
Fetching changes with git depth set to 50...
Initialized empty Git repository in /builds/rbritoda/gitlab-runner-public/.git/
Created fresh repository.
Checking out 828259f8 as load...
Skipping Git submodules setup
Executing "step_script" stage of the job script
$ lscpu
Architecture:                    aarch64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
CPU(s):                          2
On-line CPU(s) list:             0,1
Thread(s) per core:              1
Core(s) per socket:              2
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       ARM
Model:                           1
Model name:                      Neoverse-N1
Stepping:                        r3p1
BogoMIPS:                        243.75
L1d cache:                       128 KiB
L1i cache:                       128 KiB
L2 cache:                        2 MiB
L3 cache:                        32 MiB
NUMA node0 CPU(s):               0,1
Vulnerability Itlb multihit:     Not affected
Vulnerability L1tf:              Not affected
Vulnerability Mds:               Not affected
Vulnerability Meltdown:          Not affected
Vulnerability Spec store bypass: Mitigation; Speculative Store Bypass disabled via pr
Vulnerability Spectre v1:        Mitigation; __user pointer sanitization
Vulnerability Spectre v2:        Not affected
Vulnerability Srbds:             Not affected
Vulnerability Tsx async abort:   Not affected
Flags:                           fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid asimdrdm lrcpc dcpop asimddp sscbs
Cleaning up file based variables                                                                                      00:00
Job succeeded
```

Use Cases: Machine Learning / **Kubeflow**

Scale out / distributed training

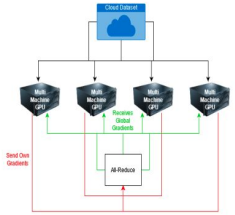
Example: Fast Simulation with 3D GANs (TensorFlow based)

First real attempt at scaling TPUs

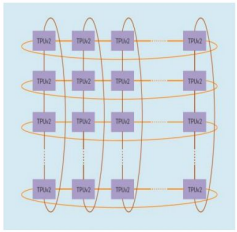


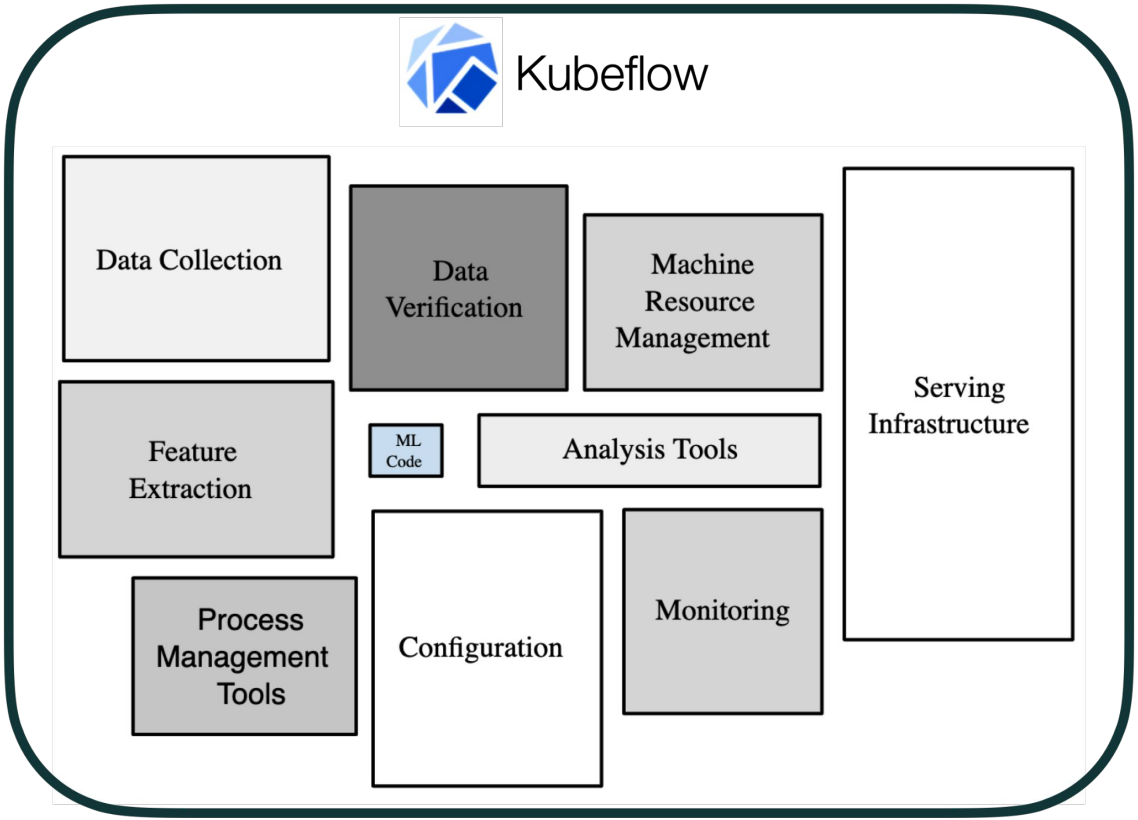
Mirrored Strategy

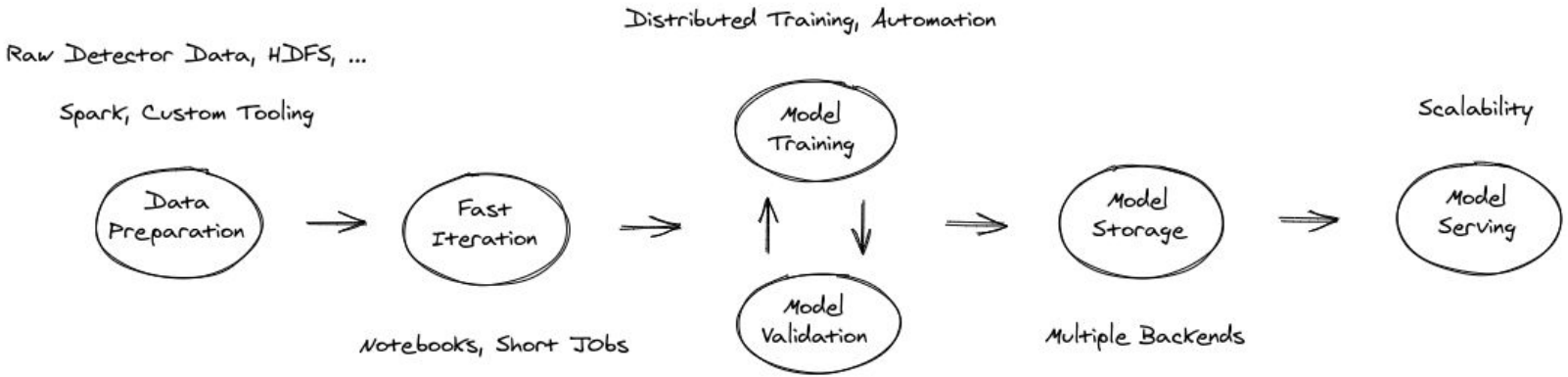
Multi Worker Mirrored Strategy

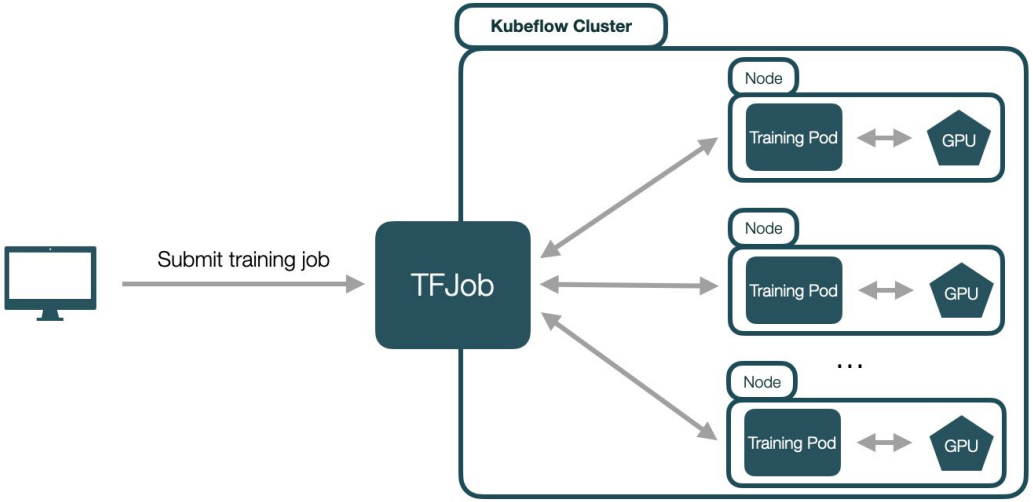


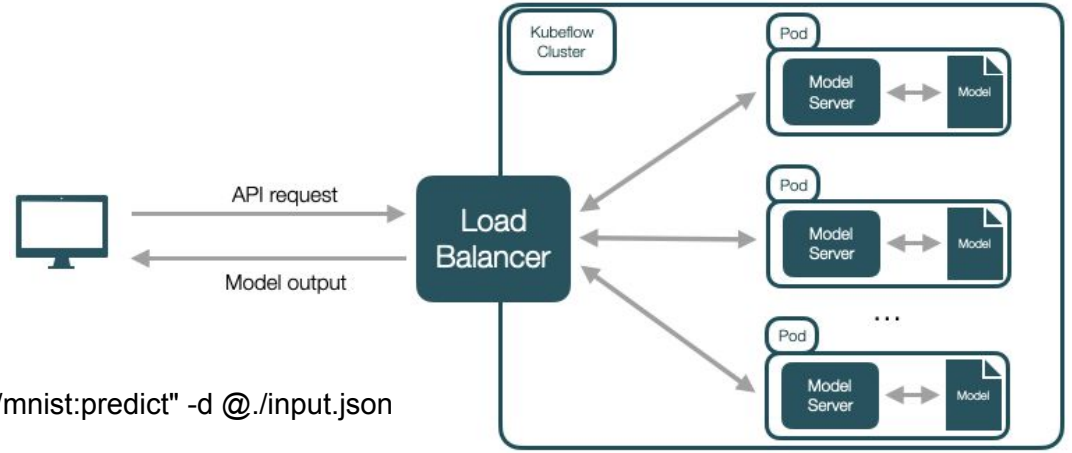
TPU Strategy




Kubeflow
Data Collection
Data Verification
Machine Resource Management
Serving Infrastructure
Feature Extraction
ML Code
Analysis Tools
Process Management Tools
Configuration
Monitoring


Distributed Training, Automation
Raw Detector Data, HDFS, ...
Spark, Custom Tooling
Scalability
Data Preparation
Fast Iteration
Model Training
Model Validation
Model Storage
Model Serving
Notebooks, Short Jobs
Multiple Backends





Inference: curl -v -H “Host: host” “http://host_ip/v1/models/mnist:predict" -d @./input.json

On resources and cost...

Resources are not infinite, even in the public cloud

- CPUs kind of feel like infinite, at least at O(10k) scale
- GPUs (preemptibles) need more care at O(1k)

Pre-emptibles / spot can be very cost effective

- Up to 90% cheaper, a lot (most?) of the workloads we have shine on these

All possible, but **need to stay flexible** on clouds, regions, zones of choice

On resources and cost...

Beware of straight, definite answers

> *“It’s X times more expensive than on-premises”*
>
> X can be anything from 3, to the most common 6, to 20, or 600

Better Answer: It’s complicated

> Workload dependent, negotiable, small changes give big gains, …
>
> Pre-emptibles / spot, Sustained / committed use discounts, egress deals, …

On resources and cost...

Taking our ML example presented earlier…

Running one epoch on 1 GPU takes ~3550 seconds (~1h)

- Reserved Cost: $2.53 USD , Preemptible cost : $0.73 USD
- 100 epochs would take over 4 days: 355000 seconds

Workload particularly well behaved, scales almost linearly without much effort

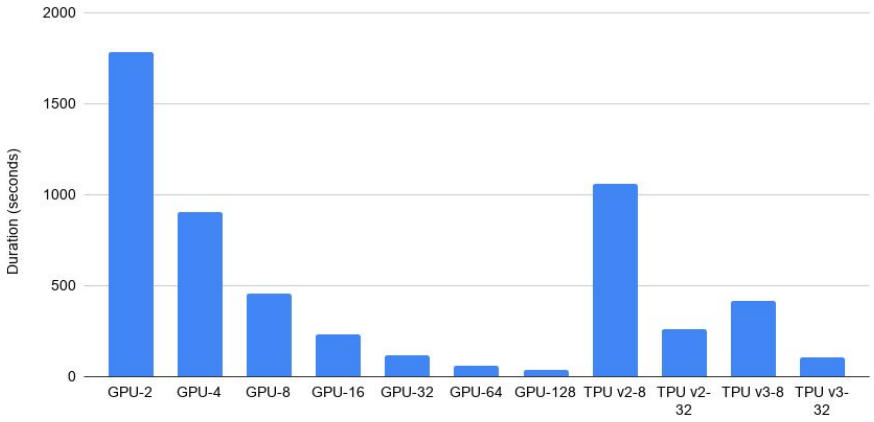
- Larger models ~~might not~~ won’t scale the same way

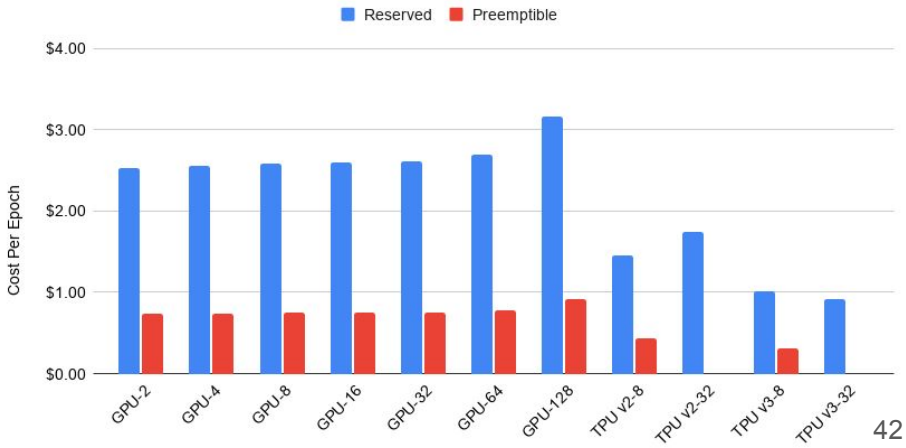
On resources and cost...

From 1 to 128 GPUs: 3550 to 35 seconds per epoch

x100 speedup at the same total cost

TPUs seem to be particularly cost effective



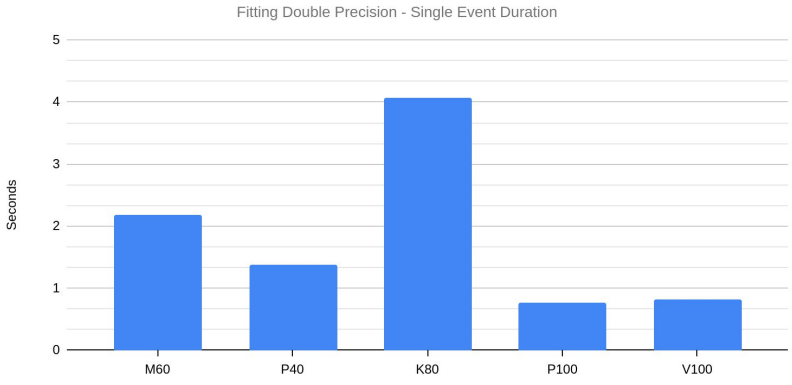


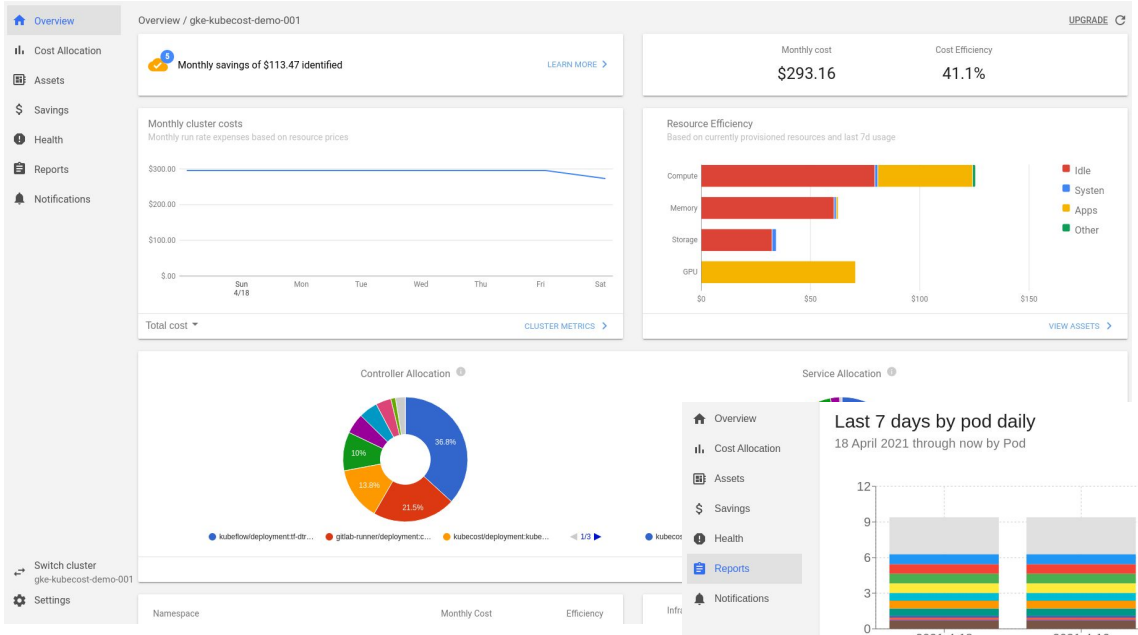
On resources and cost…

Taking another example doing statistical inference

The most powerful GPU does not give the best *value for money* (far from it)

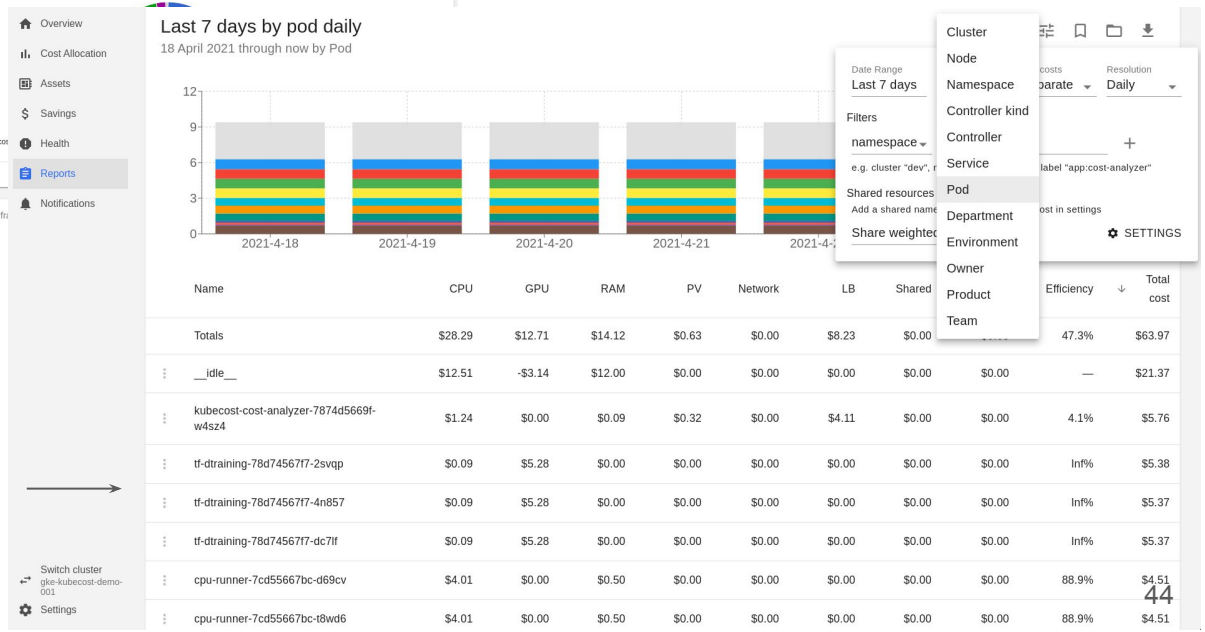
Workloads fitting the spot model shine in the public cloud





Cost efficiency evaluation

Billing down to individual workloads(s)



Name	CPU	GPU	RAM	PV	Network	LB	Shared		Efficiency	Total cost
Totals	$28.29	$12.71	$14.12	$0.63	$0.00	$8.23	$0.00		47.3%	$63.97
__idle__	$12.51	-$3.14	$12.00	$0.00	$0.00	$0.00	$0.00	$0.00	—	$21.37
kubecost-cost-analyzer-7874d5669f-w4sz4	$1.24	$0.00	$0.09	$0.32	$0.00	$4.11	$0.00	$0.00	4.1%	$5.76
tf-dtraining-78d74567f7-2svqp	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.38
tf-dtraining-78d74567f7-4n857	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.37
tf-dtraining-78d74567f7-dc7lf	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.37
cpu-runner-7cd55667bc-d69cv	$4.01	$0.00	$0.50	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$4.51
cpu-runner-7cd55667bc-t8wd6	$4.01	$0.00	$0.50	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$4.51

Conclusion

Managing **100s of clusters** on-premises, **10s** in multiple public clouds / regions

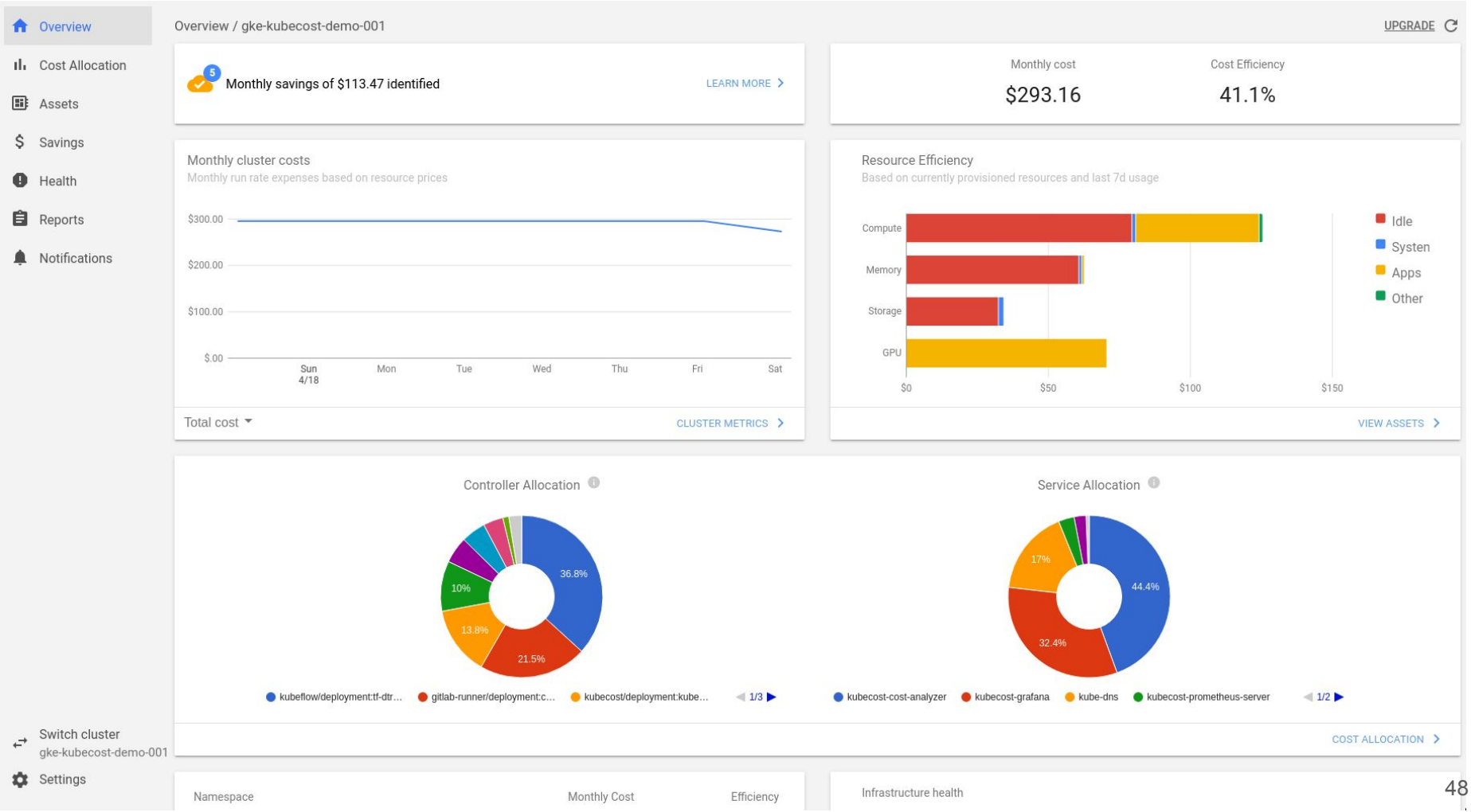
Cluster launch to full production availability in **less than 15 minutes**

Flexibility helps get to the most cost effective deployment

Kubernetes API is the **common language and enabler**

Questions?

https://events.linuxfoundation.org/kubecon-cloudnativecon-europe/


Overview
Cost Allocation
Assets
Savings
Health
Reports
Notifications
Switch cluster
gke-kubecost-demo-001
Settings
Overview / gke-kubecost-demo-001
UPGRADE
5
Monthly savings of $113.47 identified
LEARN MORE
Monthly cost
$293.16
Cost Efficiency
41.1%
Monthly cluster costs
Monthly run rate expenses based on resource prices
$300.00
$200.00
$100.00
$.00
Sun 4/18
Mon
Tue
Wed
Thu
Fri
Sat
Total cost
CLUSTER METRICS
Resource Efficiency
Based on currently provisioned resources and last 7d usage
Compute
Memory
Storage
GPU
$0
$50
$100
$150
Idle
Systen
Apps
Other
VIEW ASSETS
Controller Allocation
36.8%
21.5%
13.8%
10%
kubeflow/deployment:tf-dtr…
gitlab-runner/deployment:c…
kubecost/deployment:kube…
1/3
Service Allocation
44.4%
32.4%
17%
kubecost-cost-analyzer
kubecost-grafana
kube-dns
kubecost-prometheus-server
1/2
COST ALLOCATION
Namespace
Monthly Cost
Efficiency
Infrastructure health

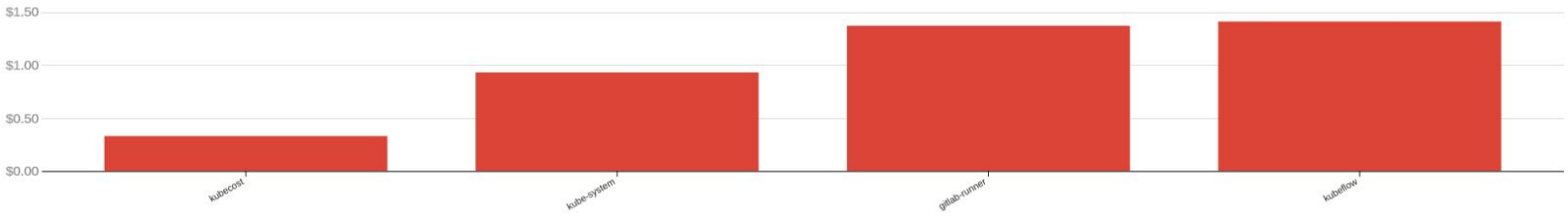
- Overview
- Cost Allocation
- Assets
- Savings
- Health
- Reports
- Notifications

Show cumulative costs by namespace for last 1d | Filters (0) | Allocate idle costs

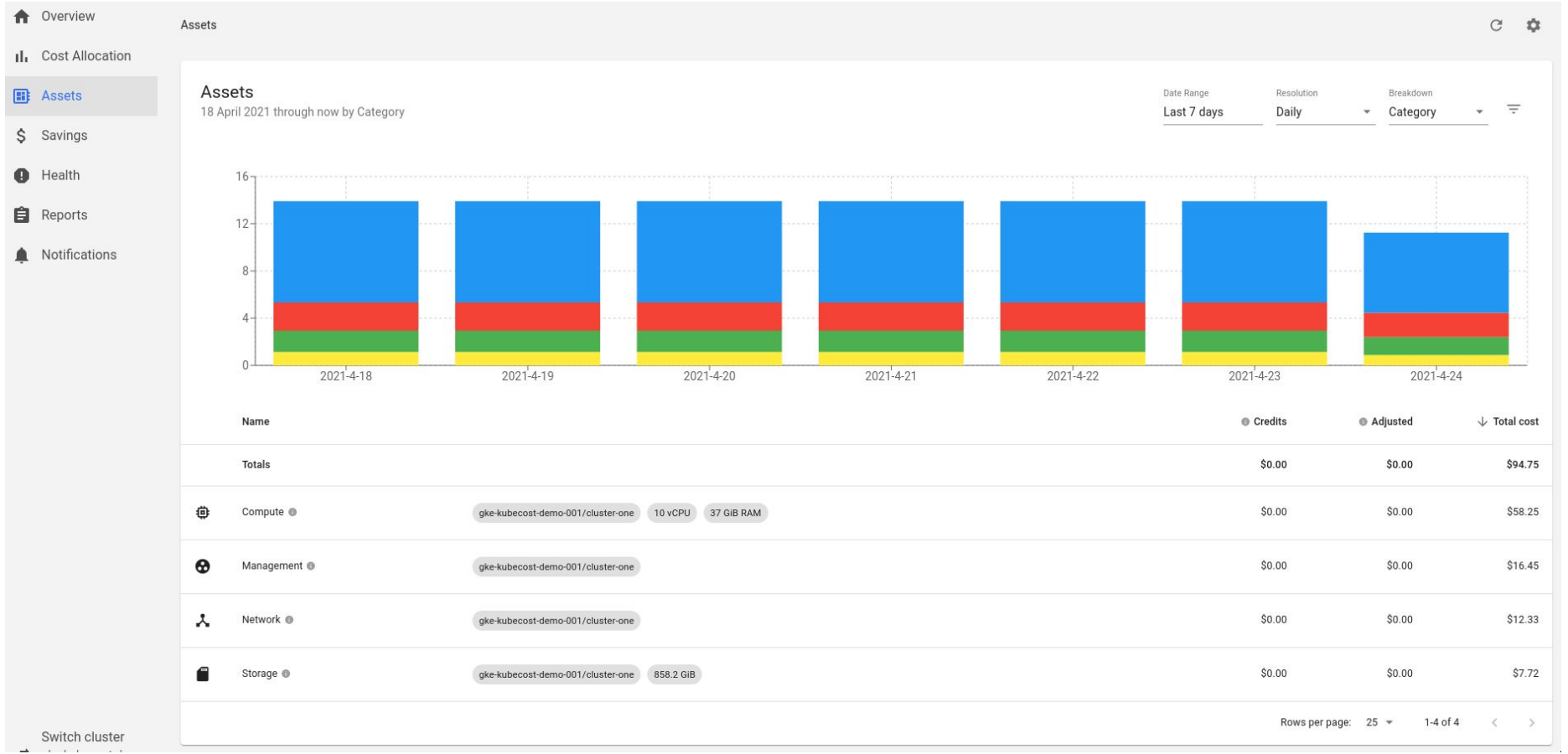
Note: this page is actively being replaced. View Reports for the new UI.

Total measured cost

Top 25 results measured over time window

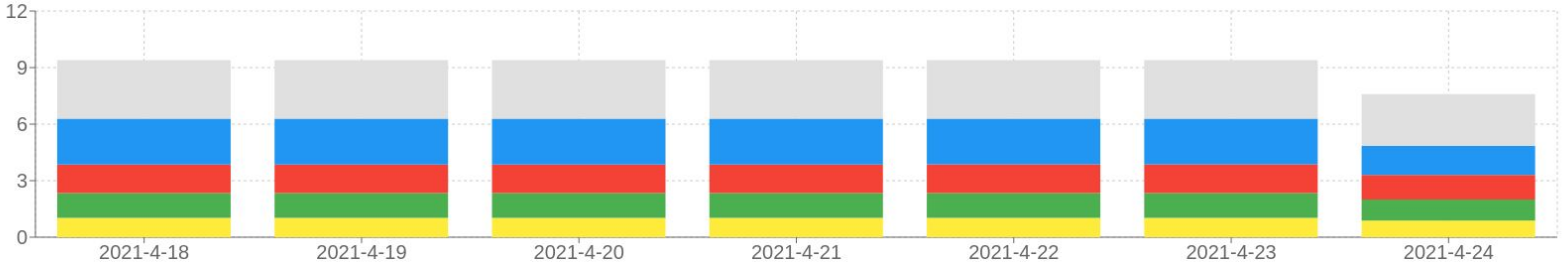


Namespace	Mem Cost	CPU Cost	Efficiency	Network Cost	PV Cost	GPU Cost	External Cost	Total Cost
kubeflow	$0.00	$0.03	*100%*	Add network	$0.00	$1.37	Add Key	$1.41
gitlab-runner	$0.15	$1.22	*88.9%*	Add network	$0.00	$0.00	Add Key	$1.37
kube-system	$0.11	$0.81	*9.3%*	Add network	$0.00	$0.00	Add Key	$0.93
kubecost	$0.03	$0.20	*25.1%*	Add network	$0.10	$0.00	Add Key	$0.33
Totals	**$0.29**	**$2.26**		**$0.00**	**$0.10**	**$1.37**	**$0.00**	**$4.04**


Overview
Cost Allocation
Assets
Savings
Health
Reports
Notifications
Switch cluster
Assets
Assets
18 April 2021 through now by Category
Date Range
Last 7 days
Resolution
Daily
Breakdown
Category
16
12
8
4
0
2021-4-18
2021-4-19
2021-4-20
2021-4-21
2021-4-22
2021-4-23
2021-4-24
Name
Credits
Adjusted
Total cost
Totals
$0.00
$0.00
$94.75
Compute
gke-kubecost-demo-001/cluster-one
10 vCPU
37 GiB RAM
$0.00
$0.00
$58.25
Management
gke-kubecost-demo-001/cluster-one
$0.00
$0.00
$16.45
Network
gke-kubecost-demo-001/cluster-one
$0.00
$0.00
$12.33
Storage
gke-kubecost-demo-001/cluster-one
858.2 GiB
$0.00
$0.00
$7.72
Rows per page:
25
1-4 of 4

Last 7 days by namespace daily

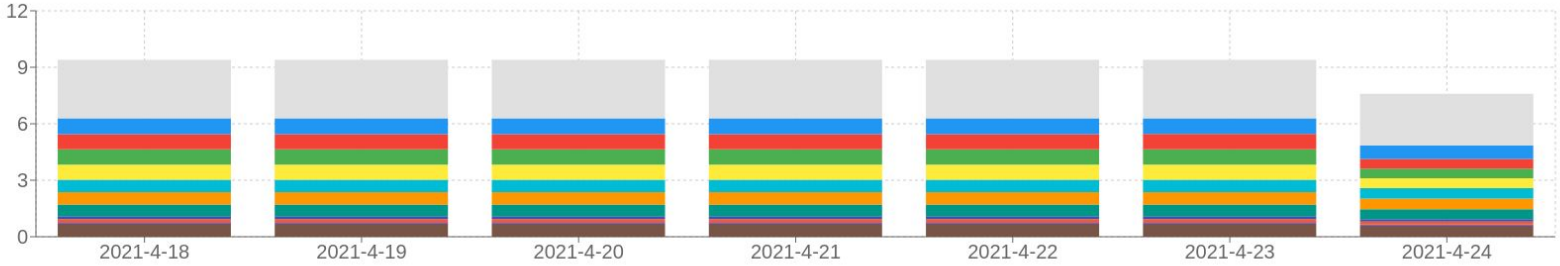
18 April 2021 through now by Namespace



Name	CPU	GPU	RAM	PV	Network	LB	Shared	External	Efficiency	Total cost
Totals	$28.29	$12.71	$14.12	$0.63	$0.00	$8.23	$0.00	$0.00	47.3%	$63.97
__idle__	$12.51	-$3.14	$12.00	$0.00	$0.00	$0.00	$0.00	$0.00	—	$21.37
kubeflow	$0.27	$15.84	$0.01	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$16.12
kubecost	$1.32	$0.00	$0.21	$0.63	$0.00	$8.23	$0.00	$0.00	34.8%	$10.40
gitlab-runner	$8.02	$0.00	$1.00	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$9.02
kube-system	$6.16	$0.00	$0.90	$0.00	$0.00	$0.00	$0.00	$0.00	7.8%	$7.06

Last 7 days by pod daily

18 April 2021 through now by Pod



Name	CPU	GPU	RAM	PV	Network	LB	Shared	External	Efficiency	Total cost
Totals	$28.29	$12.71	$14.12	$0.63	$0.00	$8.23	$0.00	$0.00	47.3%	$63.97
__idle__	$12.51	-$3.14	$12.00	$0.00	$0.00	$0.00	$0.00	$0.00	—	$21.37
kubecost-cost-analyzer-7874d5669f-w4sz4	$1.24	$0.00	$0.09	$0.32	$0.00	$4.11	$0.00	$0.00	4.1%	$5.76
tf-dtraining-78d74567f7-2svqp	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.38
tf-dtraining-78d74567f7-4n857	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.37
tf-dtraining-78d74567f7-dc7lf	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.37
cpu-runner-7cd55667bc-d69cv	$4.01	$0.00	$0.50	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$4.51
cpu-runner-7cd55667bc-t8wd6	$4.01	$0.00	$0.50	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$4.51

- Overview
- Cost Allocation
- Assets
- Savings
- Health
- Reports
- Notifications

Switch cluster
gke-kubecost-demo-001

Settings

Last 7 days by pod daily

18 April 2021 through now by Pod

Date Range
Last 7 days

Filters
namespace
e.g. cluster "dev", r

Shared resources
Add a shared name
Share weighted

costs
parate

Resolution
Daily

label "app:cost-analyzer"

ost in settings

SETTINGS

- Cluster
- Node
- Namespace
- Controller kind
- Controller
- Service
- Pod
- Department
- Environment
- Owner
- Product
- Team

Name	CPU	GPU	RAM	PV	Network	LB	Shared		Efficiency	Total cost
Totals	$28.29	$12.71	$14.12	$0.63	$0.00	$8.23	$0.00		47.3%	$63.97
__idle__	$12.51	-$3.14	$12.00	$0.00	$0.00	$0.00	$0.00	$0.00	—	$21.37
kubecost-cost-analyzer-7874d5669f-w4sz4	$1.24	$0.00	$0.09	$0.32	$0.00	$4.11	$0.00	$0.00	4.1%	$5.76
tf-dtraining-78d74567f7-2svqp	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.38
tf-dtraining-78d74567f7-4n857	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.37
tf-dtraining-78d74567f7-dc7lf	$0.09	$5.28	$0.00	$0.00	$0.00	$0.00	$0.00	$0.00	Inf%	$5.37
cpu-runner-7cd55667bc-d69cv	$4.01	$0.00	$0.50	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$4.51
cpu-runner-7cd55667bc-t8wd6	$4.01	$0.00	$0.50	$0.00	$0.00	$0.00	$0.00	$0.00	88.9%	$4.51