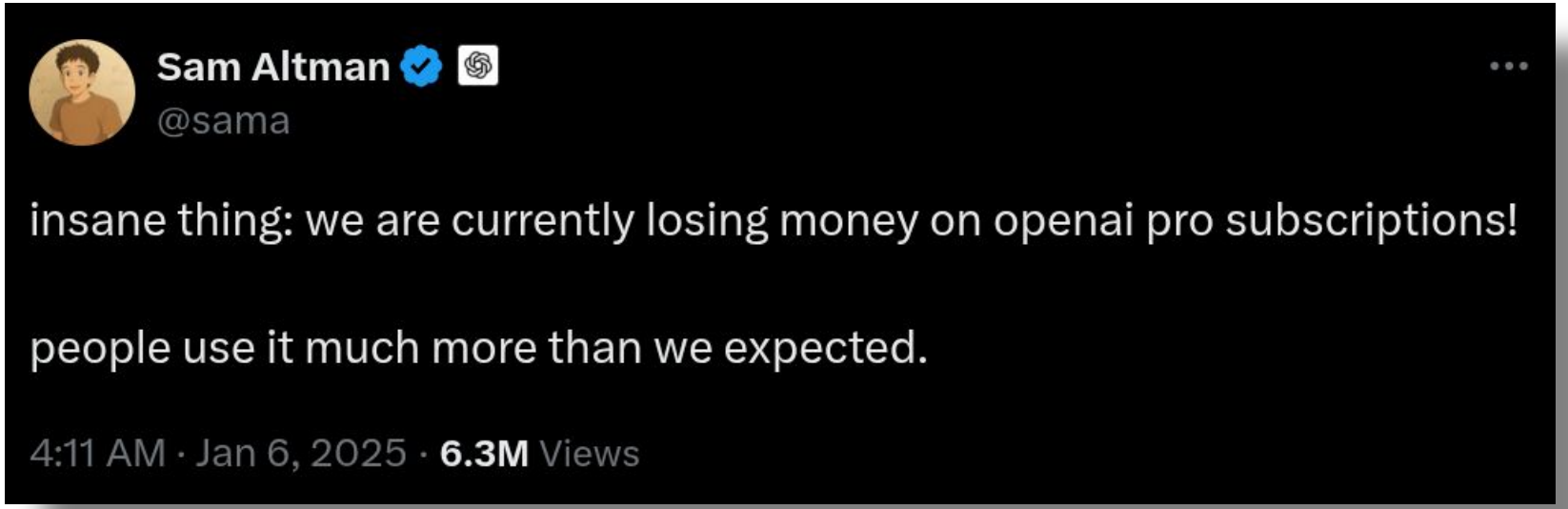
Scalable Inference

with llm-d and KServe

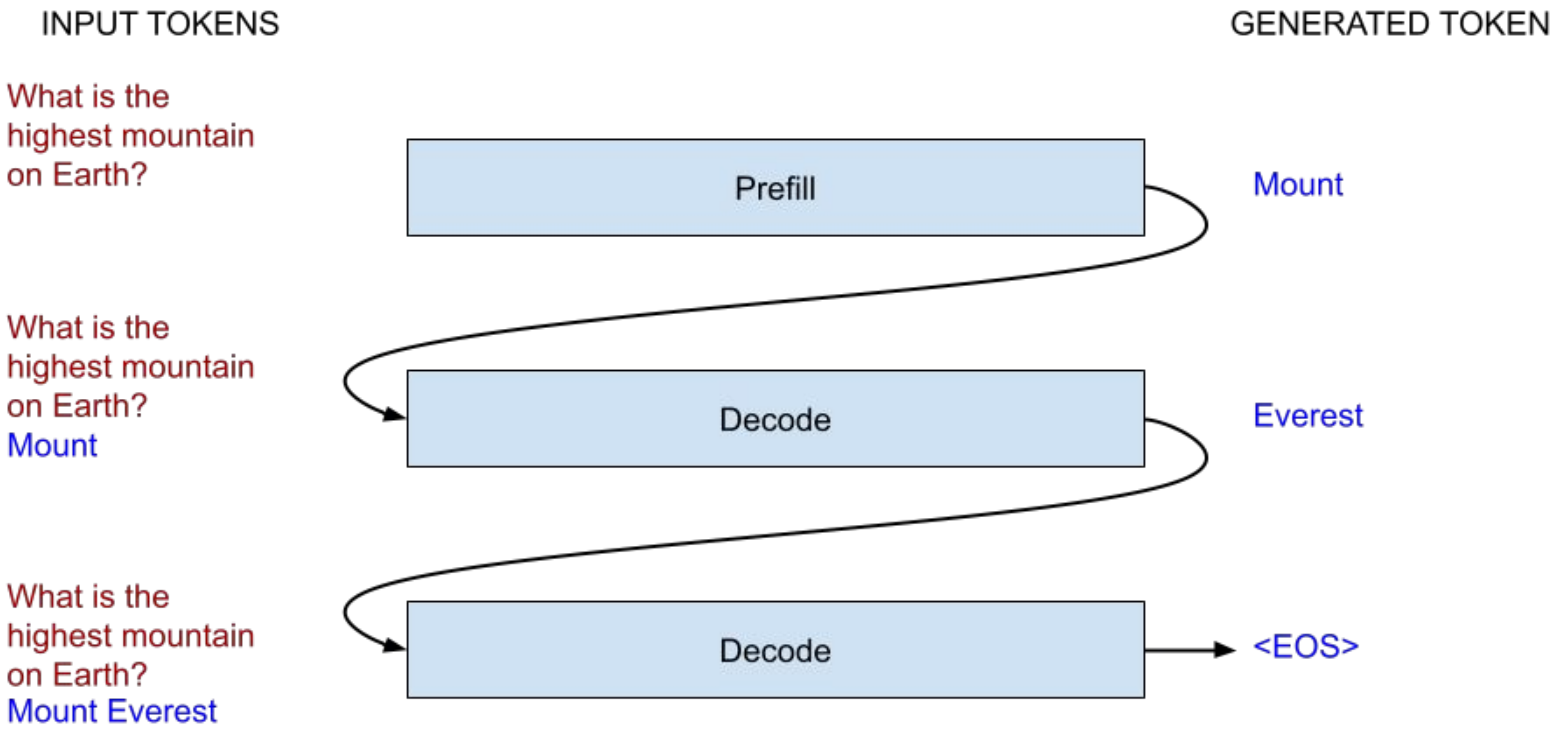
Bartosz Majsak

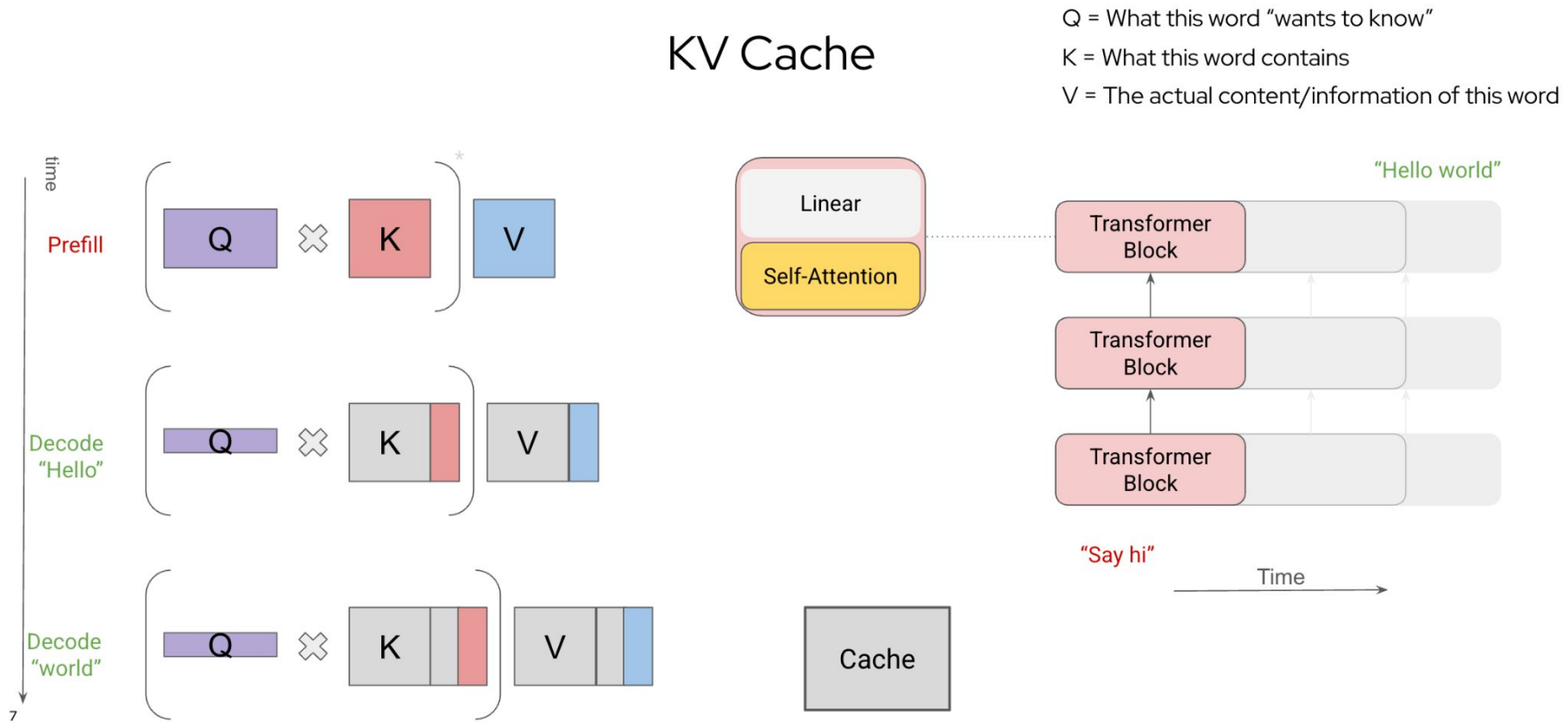
Scaling LLMs is hard

Scaling LLMs is hard



Red Hat



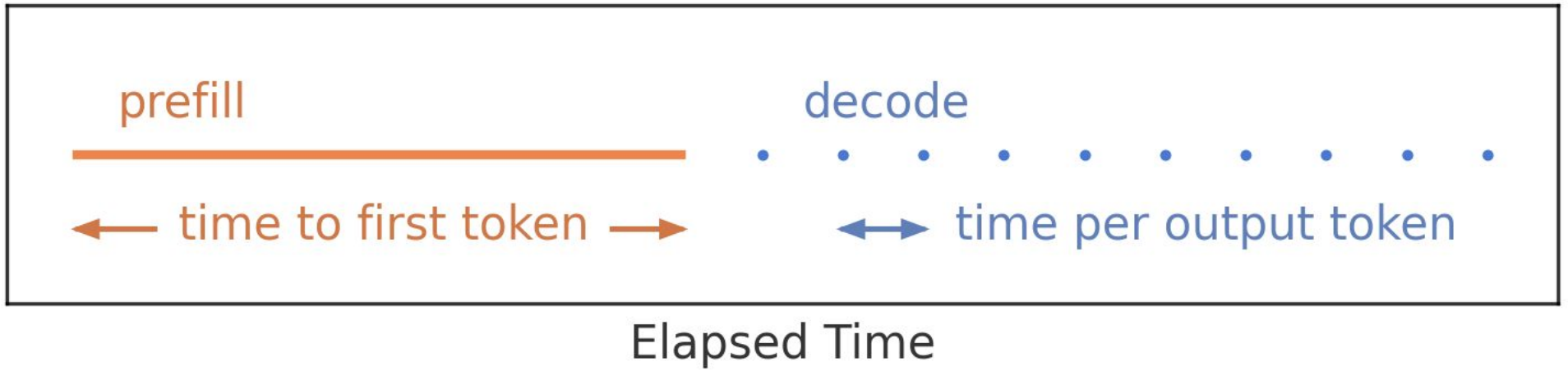


Prefill

- Processes input tokens and computes
- Highly parallelized, efficient GPU utilization

Decode

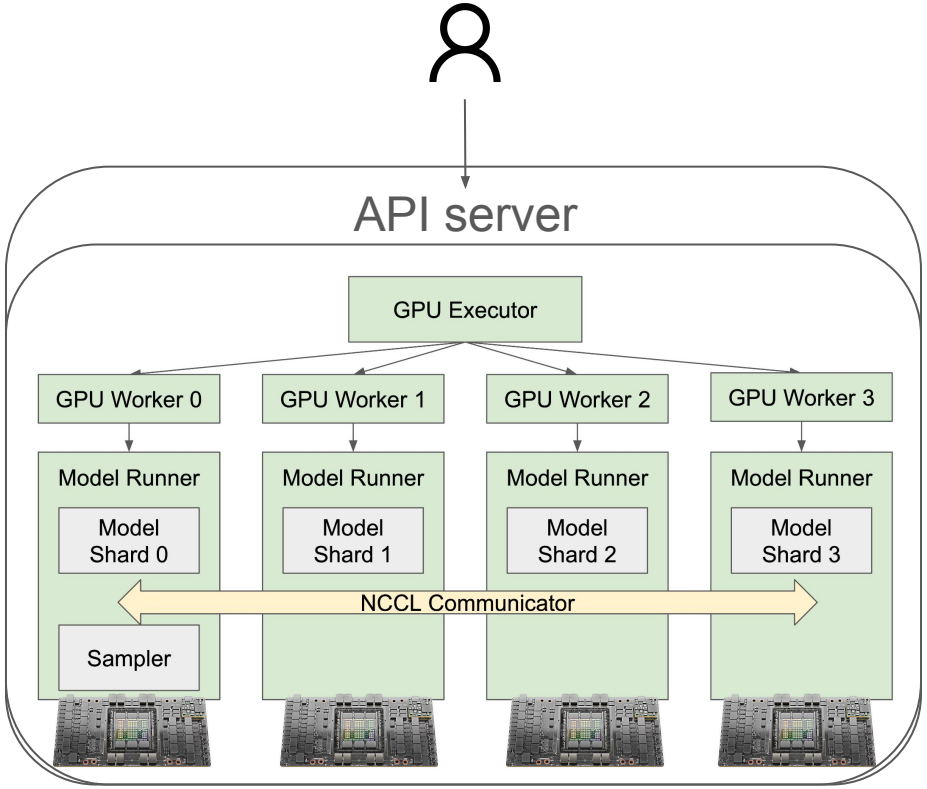
- Generates output tokens one at a time
- Sequential processing, based on previous tokens
  - potentially slower
  - memory-bound



https://huggingface.co/blog/tngtech/llm-performance-prefill-decode-concurrent-requests

vLLM : a LLM Inference & Serving Engine

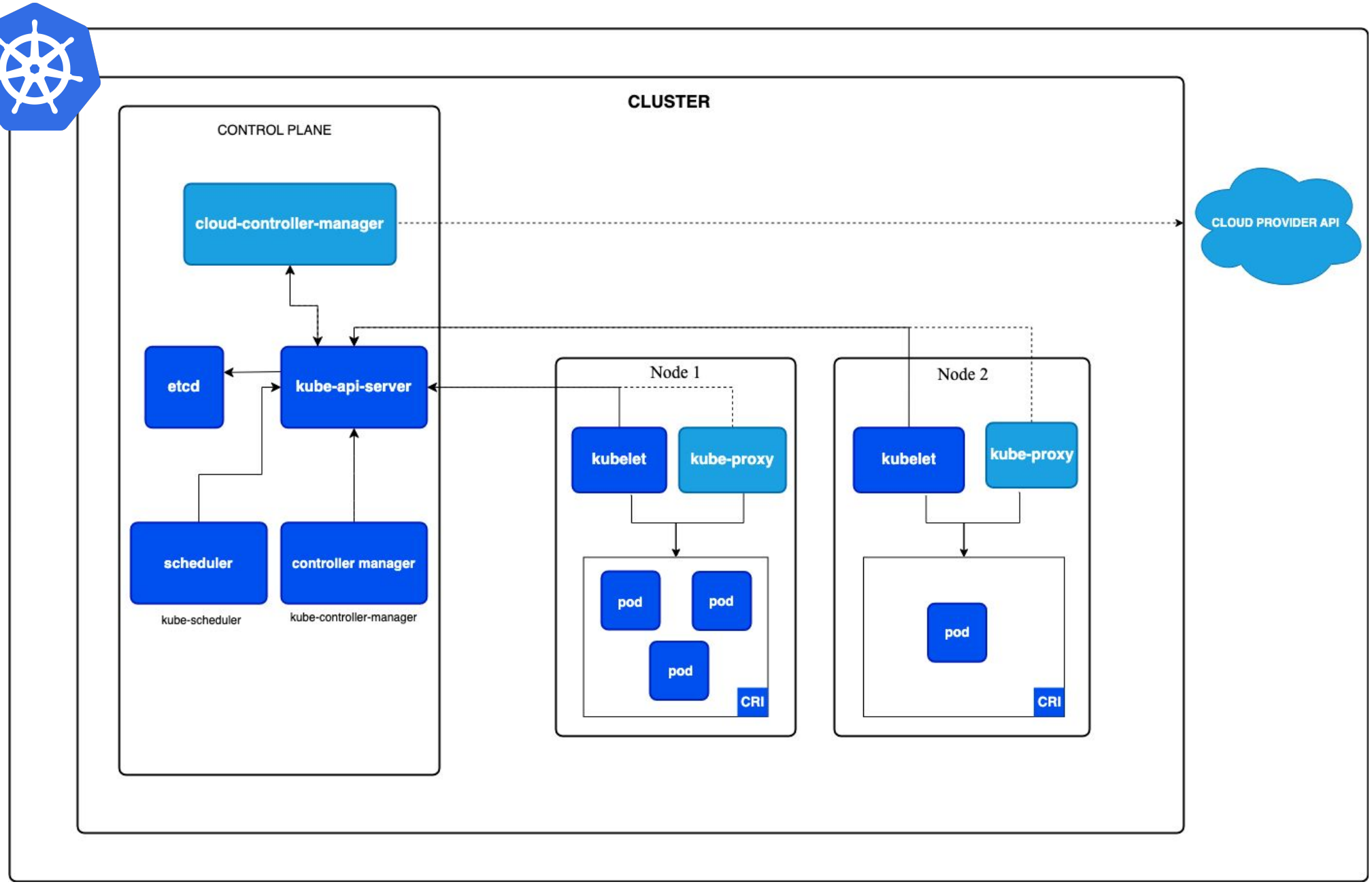
- Library or REST API (OpenAI API specification)
- Allowing users to chat with one model with:
  - **Text, Images, Videos or Audio** as input
  - **Text-only** as output (no diffusion models)
- **100+** model architectures supported
- ~400 PR/months
- Supports “all” GPUs/accelerators
  - NVIDIA, AMD, Intel, IBM Spyre, Google TPU, AWS Trainium/Inferentia, etc...



Red Hat OpenShift

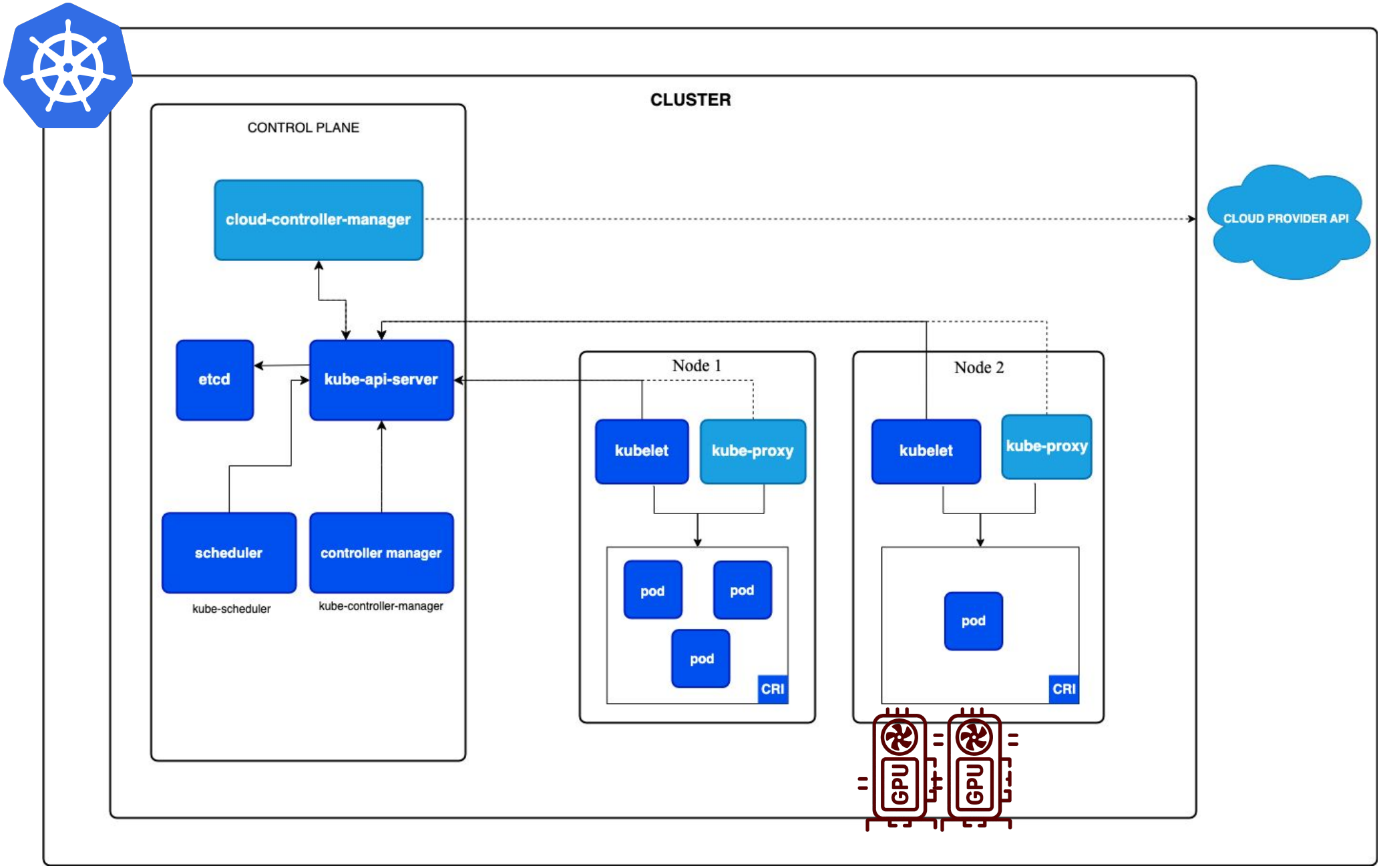
Kubernetes GPU management

Red Hat



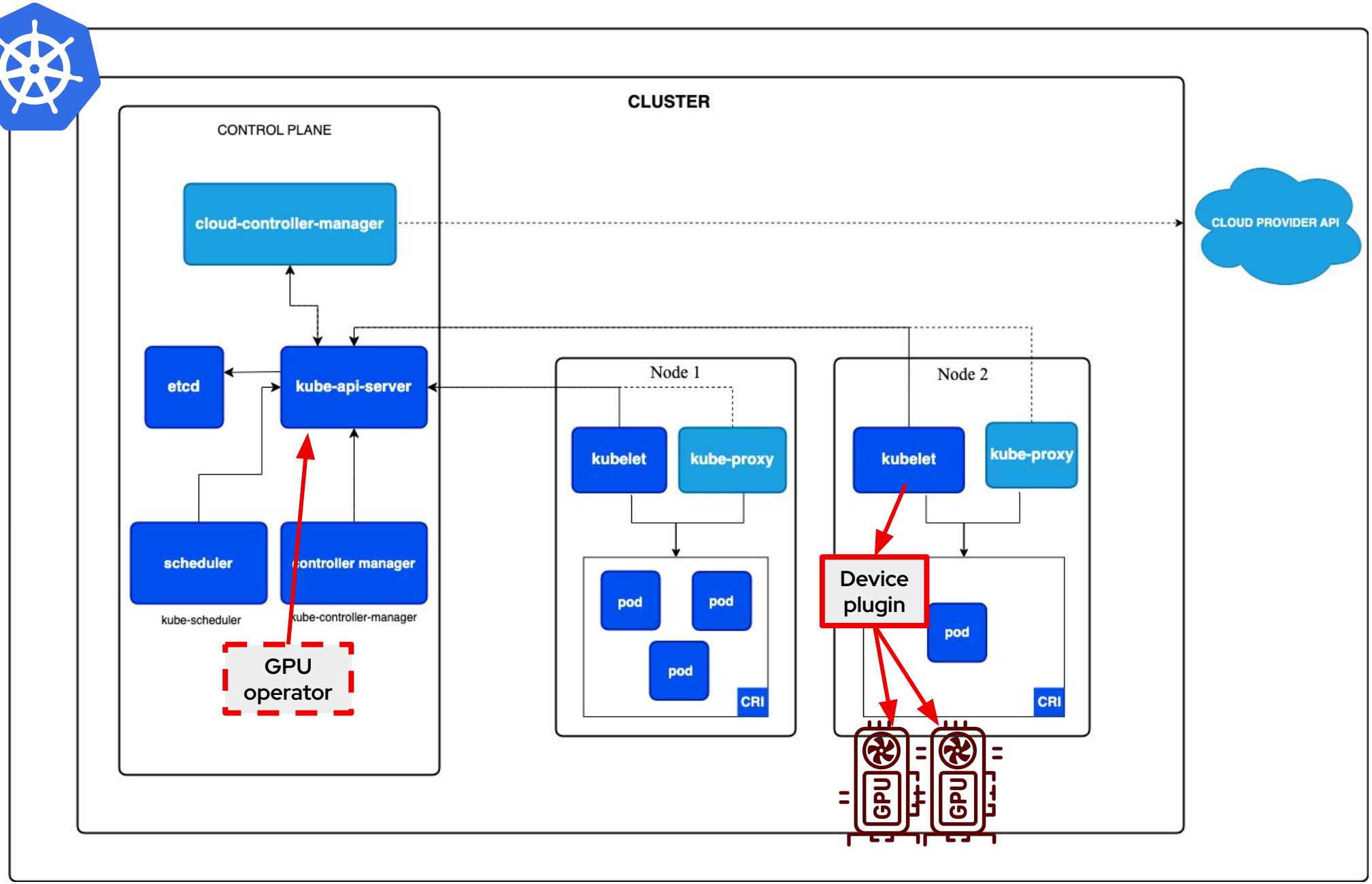
https://kubernetes.io/docs/concepts/architecture/

Red Hat OpenShift



https://kubernetes.io/docs/concepts/architecture/

Red Hat OpenShift



https://kubernetes.io/docs/concepts/architecture/

Red Hat OpenShift

I need GPU!

```yaml
spec:
  containers:
  - name: app
    image: ...
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
        nvidia.com/gpu: 2
      limits:
        memory: "128Mi"
        cpu: "500m"
```

Photo by Timur Garifov on Unsplash

But... a node with GPU is expensive

Don’t waste it!

```yaml
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  ...
spec:
  replicas: 1
  selector:
    ...
  template:
    ...
    spec:
      ...
      taints:
        - key: restrictedaccess
          value: "yes"
          effect: NoSchedule
```

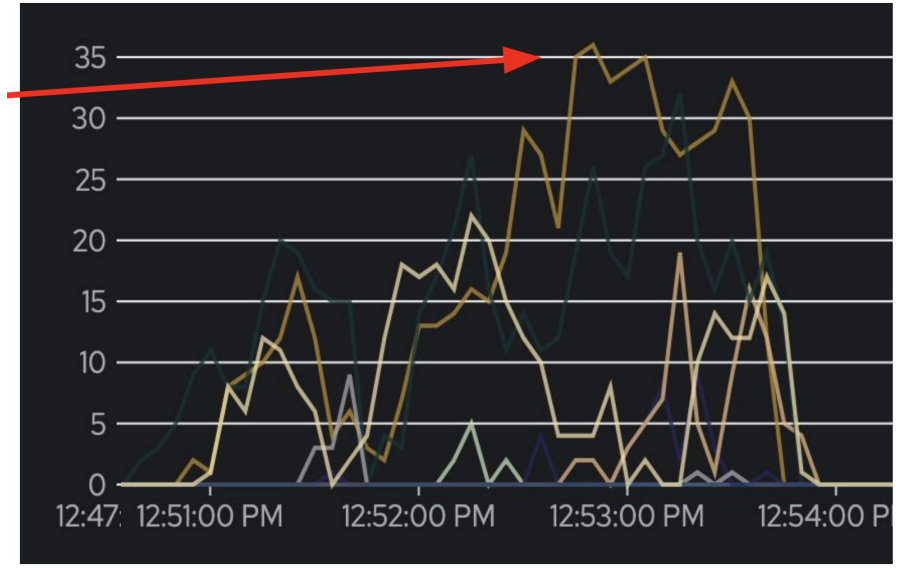
```yaml
apiVersion: nvidia.com/v1
kind: ClusterPolicy
metadata:
  ...
  name: gpu-cluster-policy
spec:
  ...
  daemonsets:
    ...
    tolerations:
      - effect: NoSchedule
        key: restrictedaccess
        operator: Exists
  sandboxWorkloads: ...
  gds: ...
  vgpuManager: ...
  vfioManager: ...
  toolkit: ...
...
```

https://ai-on-openshift.io/odh-rhoai/nvidia-gpus/

Red Hat OpenShift

Disaggregated serving

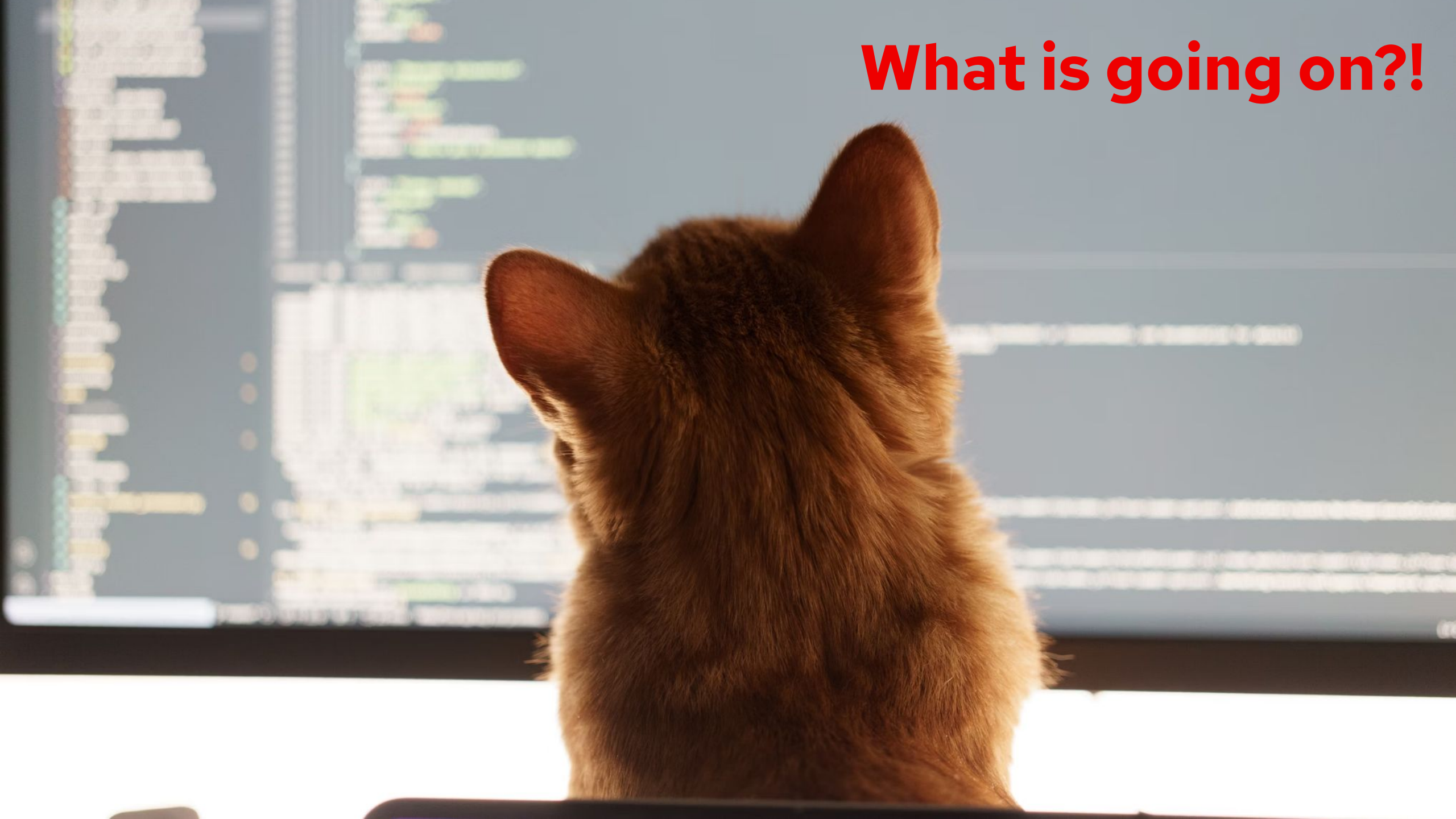
When a single instance is not enough...



vLLM metric: num_requests_waiting
(raw-deployment)

https://kccncna2024.sched.com/event/1i7lb

Red Hat
OpenShift


What is going on?!

Prefill and decode

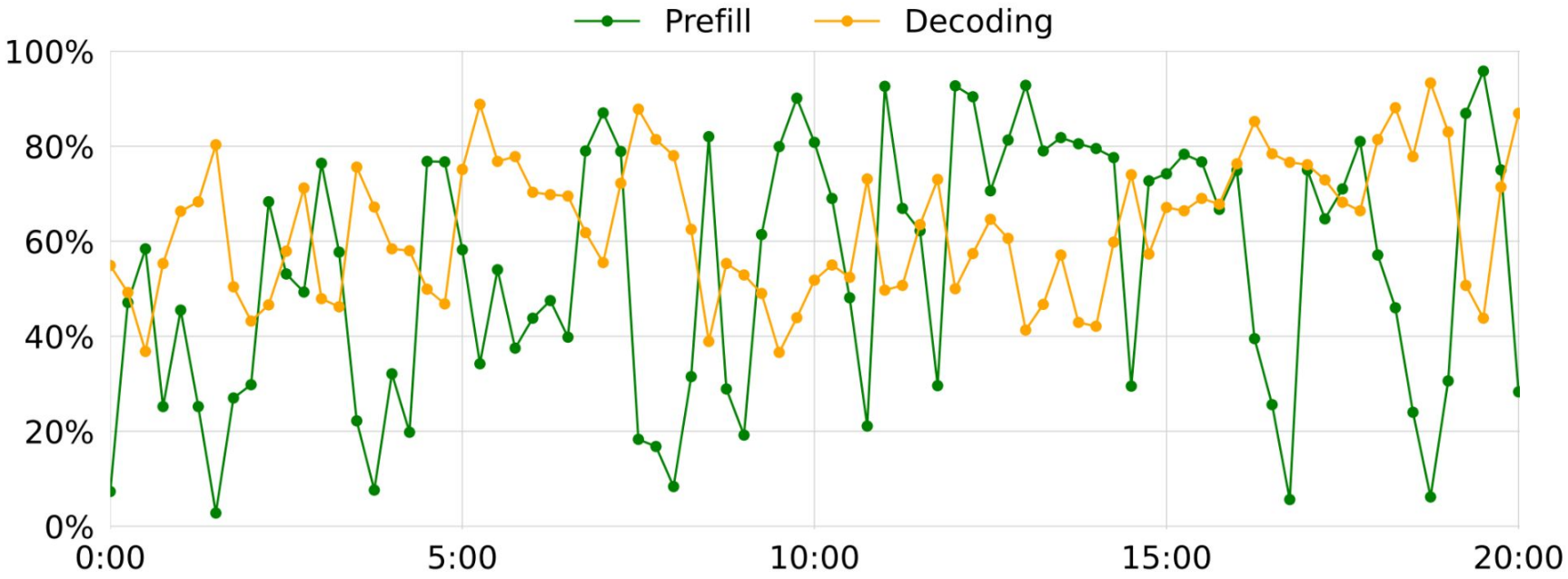


Figure 9: The load of prefill and decoding instances over 20 minutes, before using the prediction-based early rejection.

https://arxiv.org/abs/2407.00079

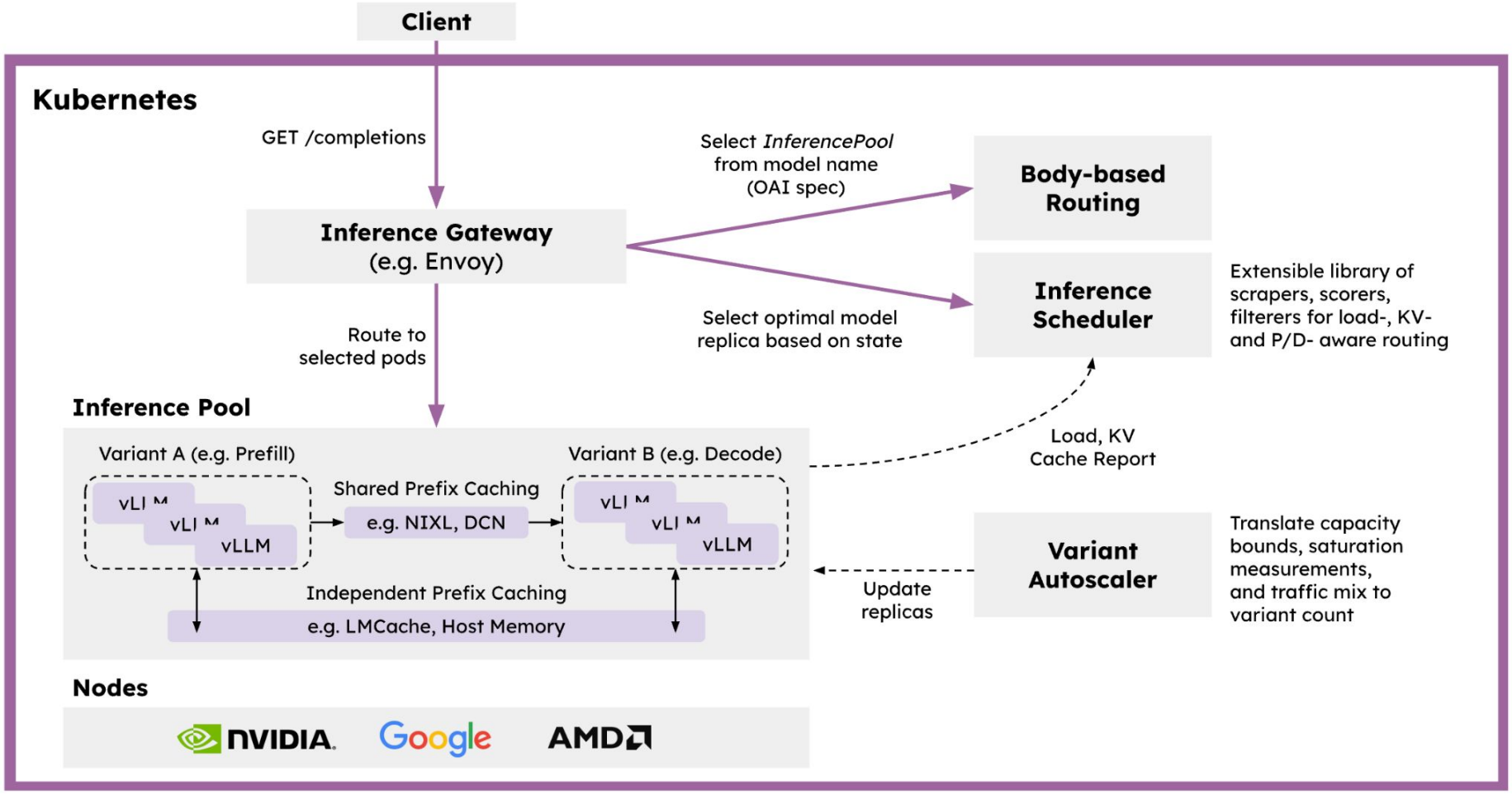
Red Hat OpenShift

What is going on? - Prefill and decode

*All ~~animals~~ requests are **NOT** equal*

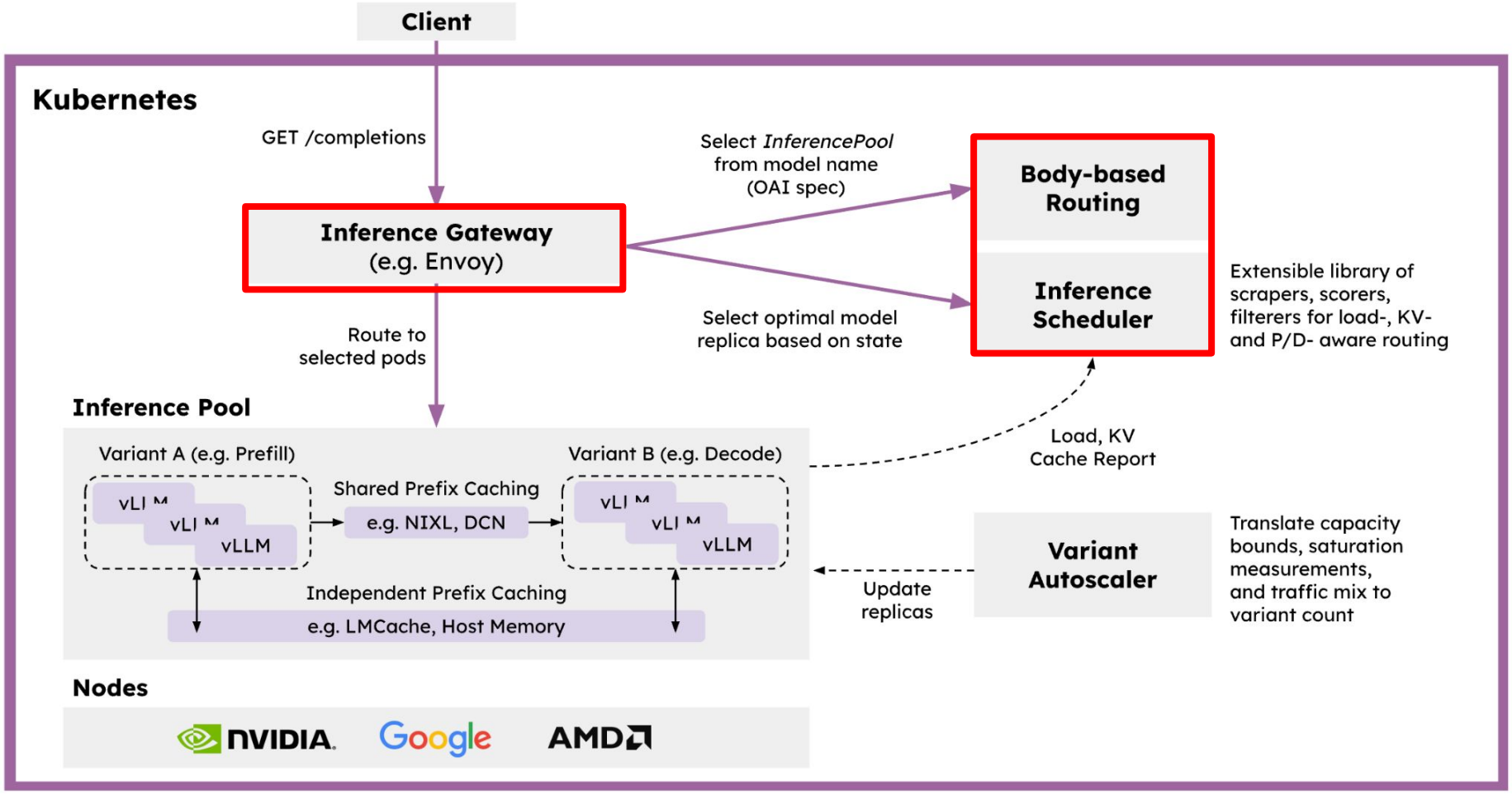
- **Prefill** and **Decode** phases have **different scalability challenges** and latency implication.
- **Prefill phase** processes the prompt produces the first token (**Time To First Token**)
- **Decode phase** produces all the other tokens (**Inter Token Latency**)


llm-d



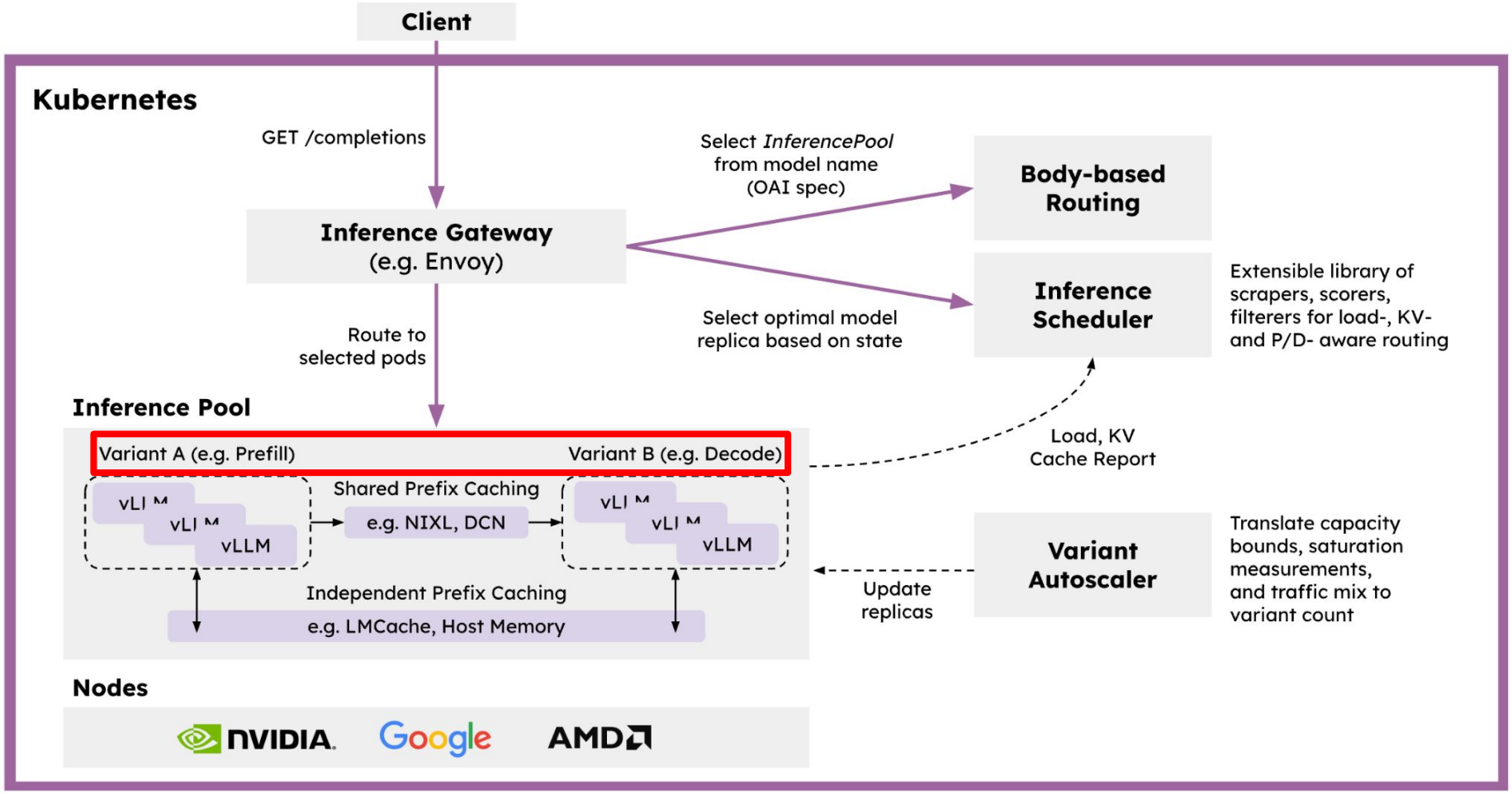
llm-d

Red Hat OpenShift



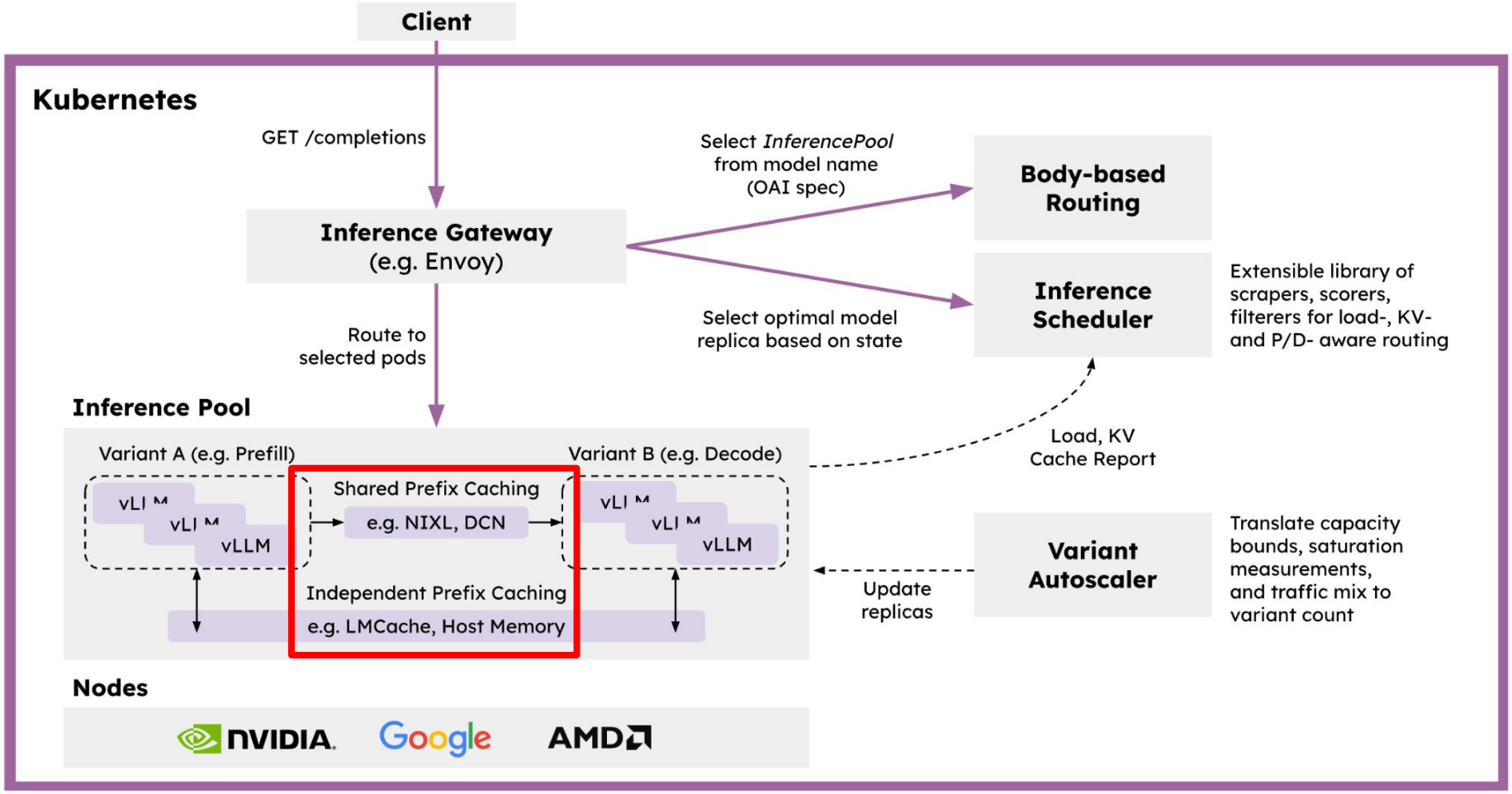
llm-d

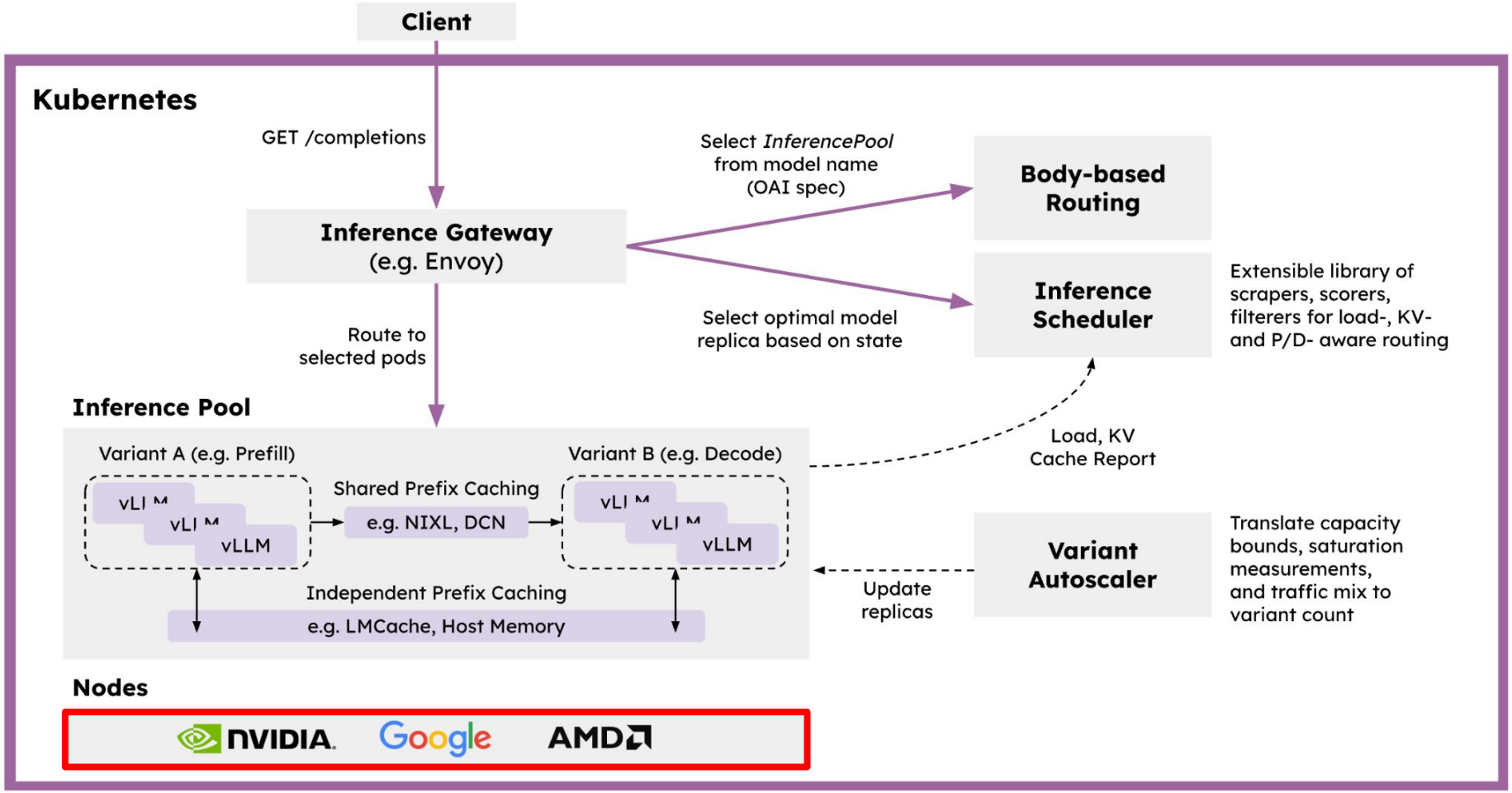
Red Hat OpenShift



llm-d

Red Hat OpenShift

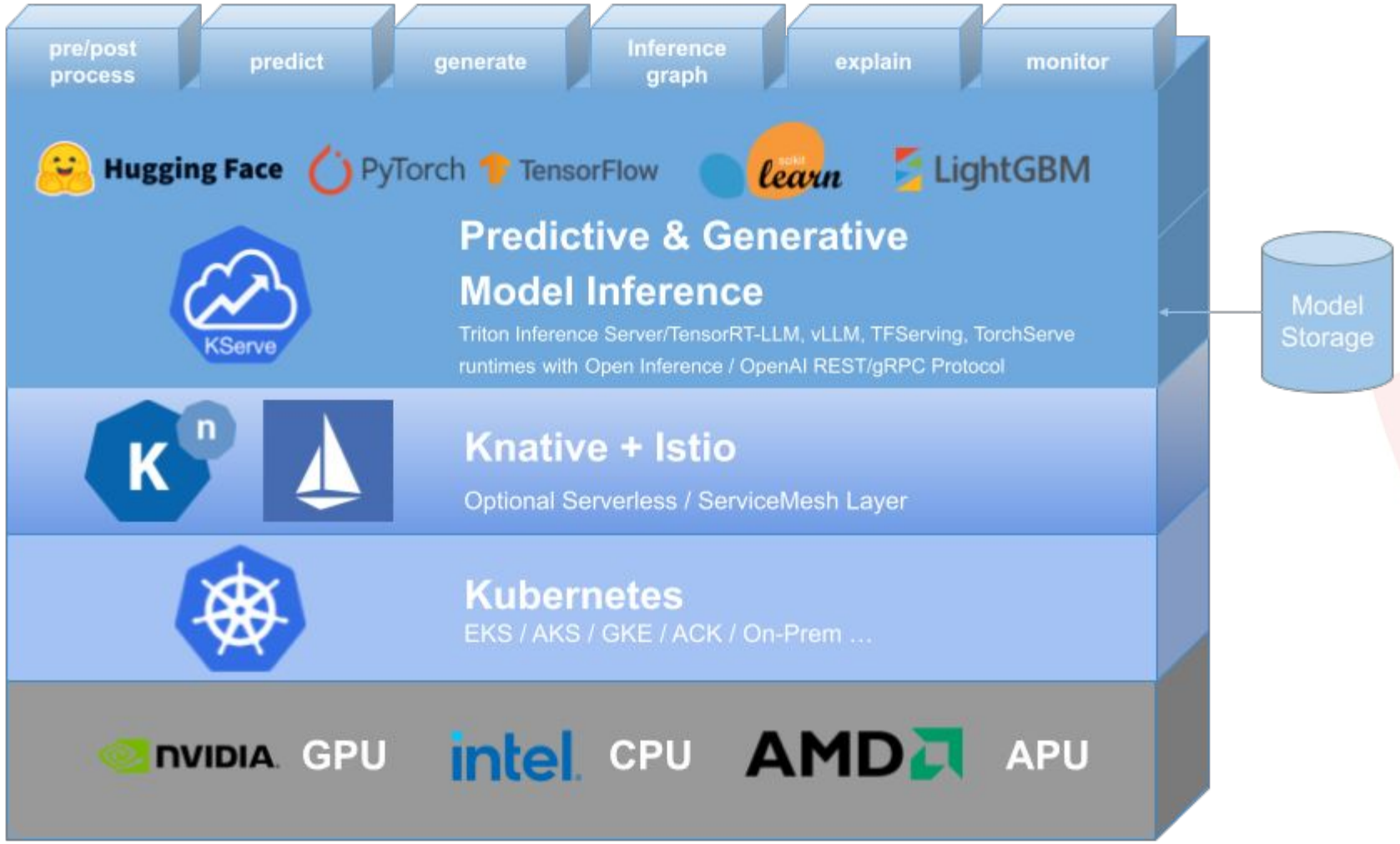

Client
Kubernetes
GET /completions
Inference Gateway
(e.g. Envoy)
Route to
selected pods
Select InferencePool
from model name
(OAI spec)
Body-based
Routing
Select optimal model
replica based on state
Inference
Scheduler
Extensible library of
scrapers, scorers,
filterers for load-, KV-
and P/D- aware routing
Inference Pool
Variant A (e.g. Prefill)
Variant B (e.g. Decode)
vLLM
vLLM
vLLM
Shared Prefix Caching
e.g. NIXL, DCN
vLLM
vLLM
vLLM
Independent Prefix Caching
e.g. LMCache, Host Memory
Load, KV
Cache Report
Update
replicas
Variant
Autoscaler
Translate capacity
bounds, saturation
measurements,
and traffic mix to
variant count
Nodes
NVIDIA
Google
AMD



llm-d

Red Hat OpenShift


KServe



```yaml
kind: LLMInferenceService
metadata:
  name: llama4-scout-17b-16e-instruct
spec:
  replicas: 2
  model:
    uri: hf://meta-llama/Llama-4-Scout-17B-16E-Instruct
  router:        # automatically create Gateway and HTTPRoute
    route: {}
    gateway: {}
    scheduler: {}
```

```
kind: LLMInferenceServiceConfig
metadata:
  name: nvidia-a100
spec:
  template:
    nodeSelector:
      nvidia.com/gpu.product: NVIDIA-A100-PCIE-40G
    containers:
    - name: main
      image: vllm-image…
      resources:
        requests:
          nvidia.com/gpu: “1”
```

KServe: LLMInferenceService - with nvidia A100 node selector

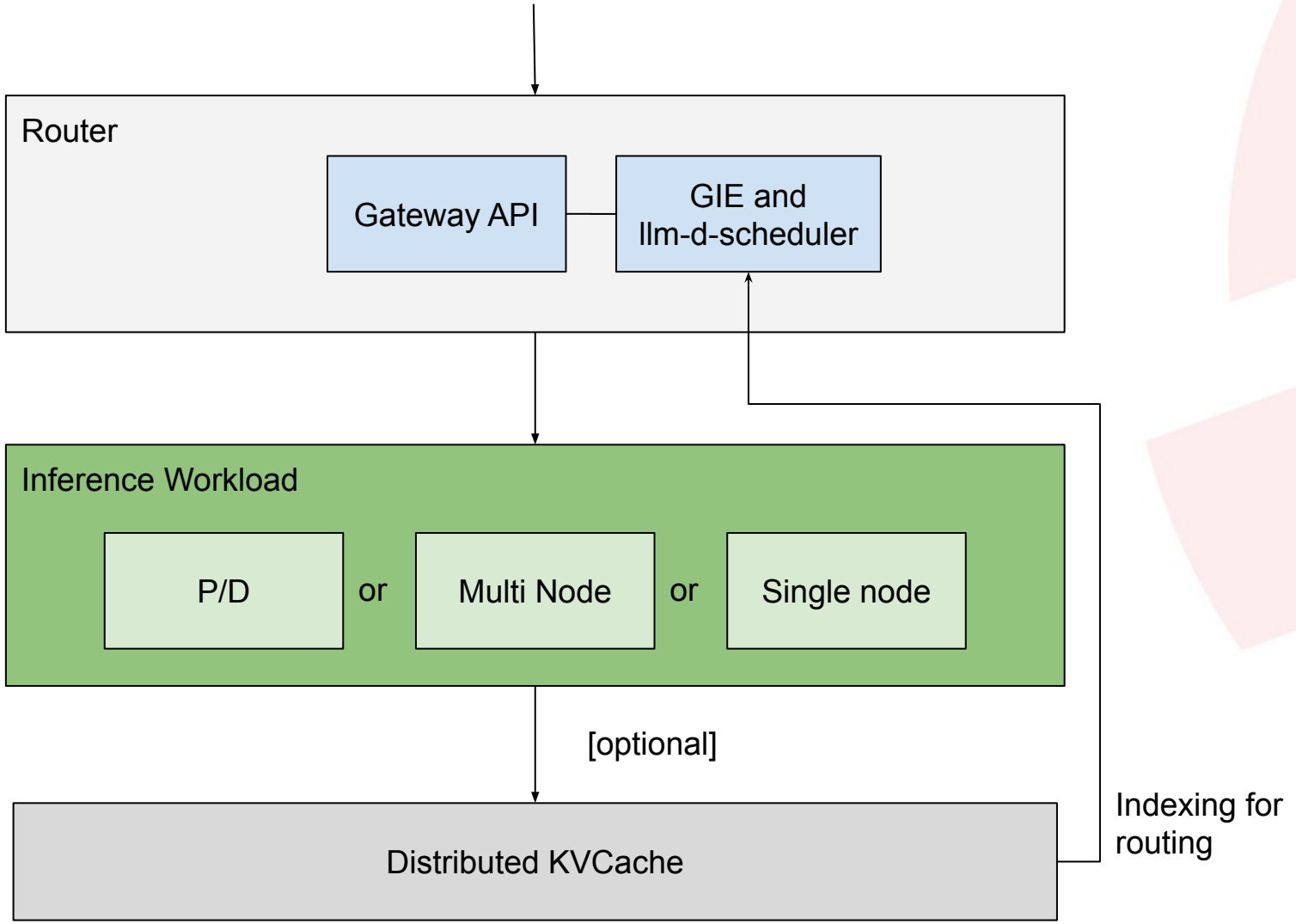
```yaml
kind: LLMInferenceService
metadata:
  name: llama4-scout-17b-16e-instruct
spec:
  replicas: 2
  model:
    uri: hf://meta-llama/Llama-4-Scout-17B-16E-Instruct
  router:
    route: {}
    gateway: {}
    scheduler: {}
  baseRefs:
    - nvidia-a100
```


KServe llm-d

Red Hat OpenShift

```
spec:
  router: # …

  template: # …
  worker:   # …
  prefill:  # …
```



KServe llm-d

Red Hat OpenShift


Demo!

Helpful links

- https://kserve.github.io/website/
- https://llm-d.ai/
- https://docs.vllm.ai/en/stable/
- https://ai-on-openshift.io/
- https://www.redhat.com/de/products/ai/openshift-ai
- https://gateway-api.sigs.k8s.io/
- https://gateway-api-inference-extension.sigs.k8s.io/

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat