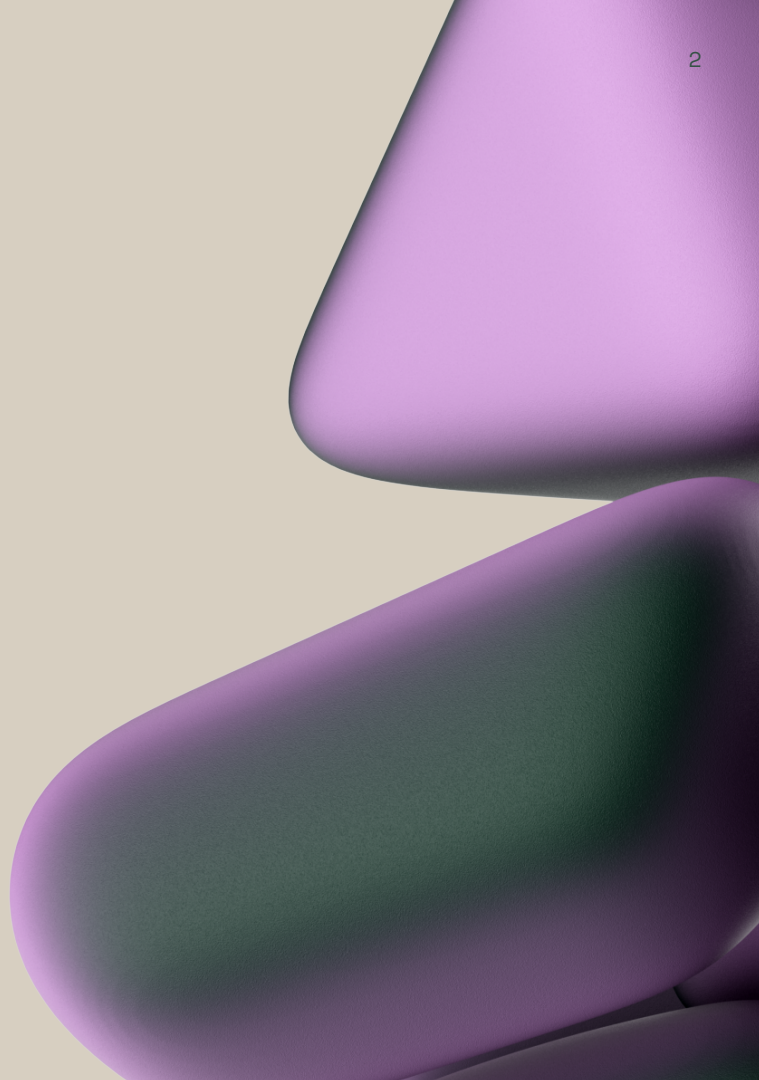
cohere

Mastering LLM Delivery in Private Clouds

A journey to **seamless** deployments with **Kubernetes** and **OCI**



A journey to **seamless** deployments with **Kubernetes** and **OCI**

A journey to **seamless** deployments with **Kubernetes** and **OCI** (Open Container Initiative)

Not Oracle Cloud Infrastructure

cohere

Autumn Moulder

Director, Infrastructure & Security

cohere

Marwan Ahmed

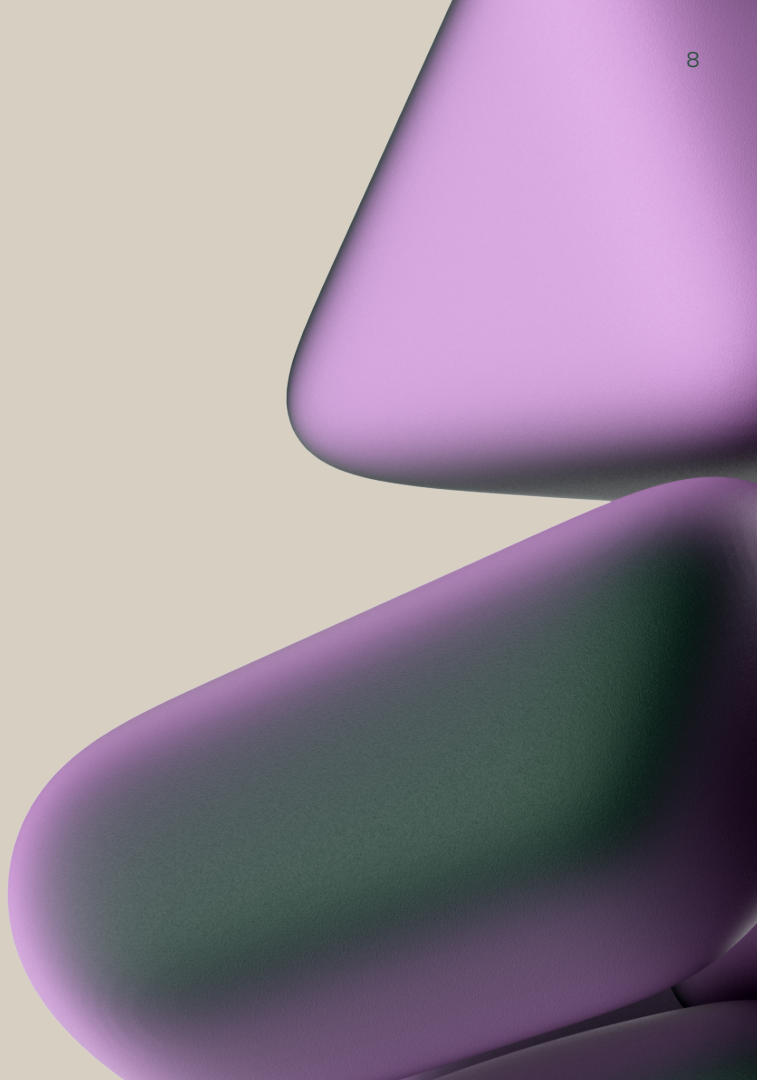
Member of Technical Staff

Site reliability engineer?

Data scientist / MLE?

....?

What is a Large Language Model?



cohere coral

CORAL DASHBOARD DOCUMENTATION PLAYGROUND COMMUNITY

Large Language Models

GROUNDING

Let's dive into it. Search the web or give me a task.

what is a large language model?

A large language model is a type of artificial intelligence that uses statistical models to analyze vast amounts of data, learning the patterns and connections between words and phrases. This allows them to generate new content, such as essays or articles, that are similar in style to a specific author or genre.

How does the performance of large language models compare to other types of AI models?

Message...

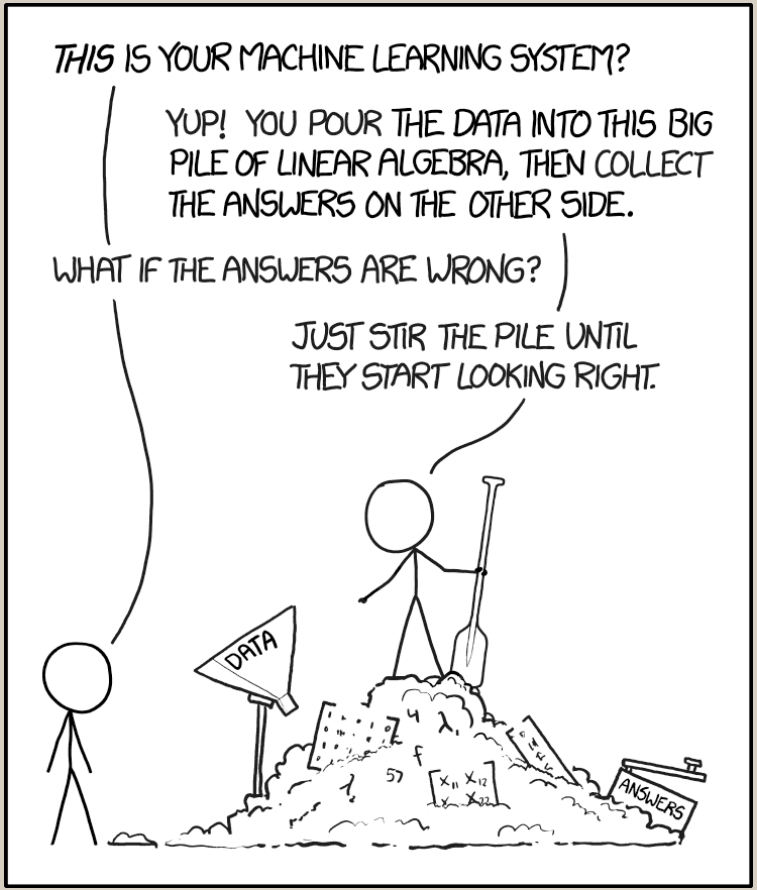
CORAL IS IN DEVELOPMENT AND USES AN ENGLISH-ONLY MODEL. HELP US IMPROVE BY RATING ANSWERS.

3 references

TECHTARGET
WHAT ARE LARGE LANGUAGE MODELS...

WIKIPEDIA
LARGE LANGUAGE MODEL - WIKIPEDIA

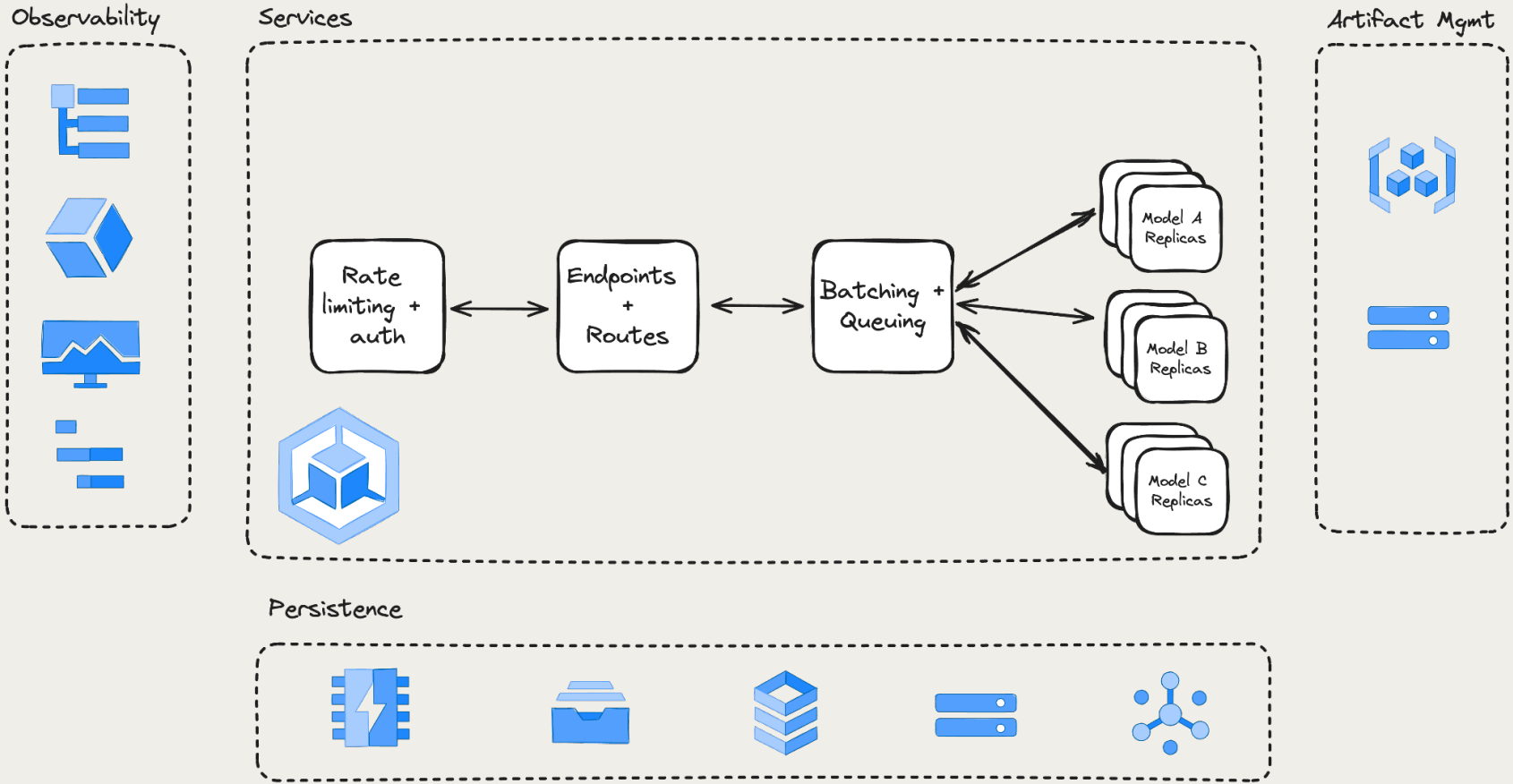
BOOST
WHAT ARE LARGE LANGUAGE MODELS ...



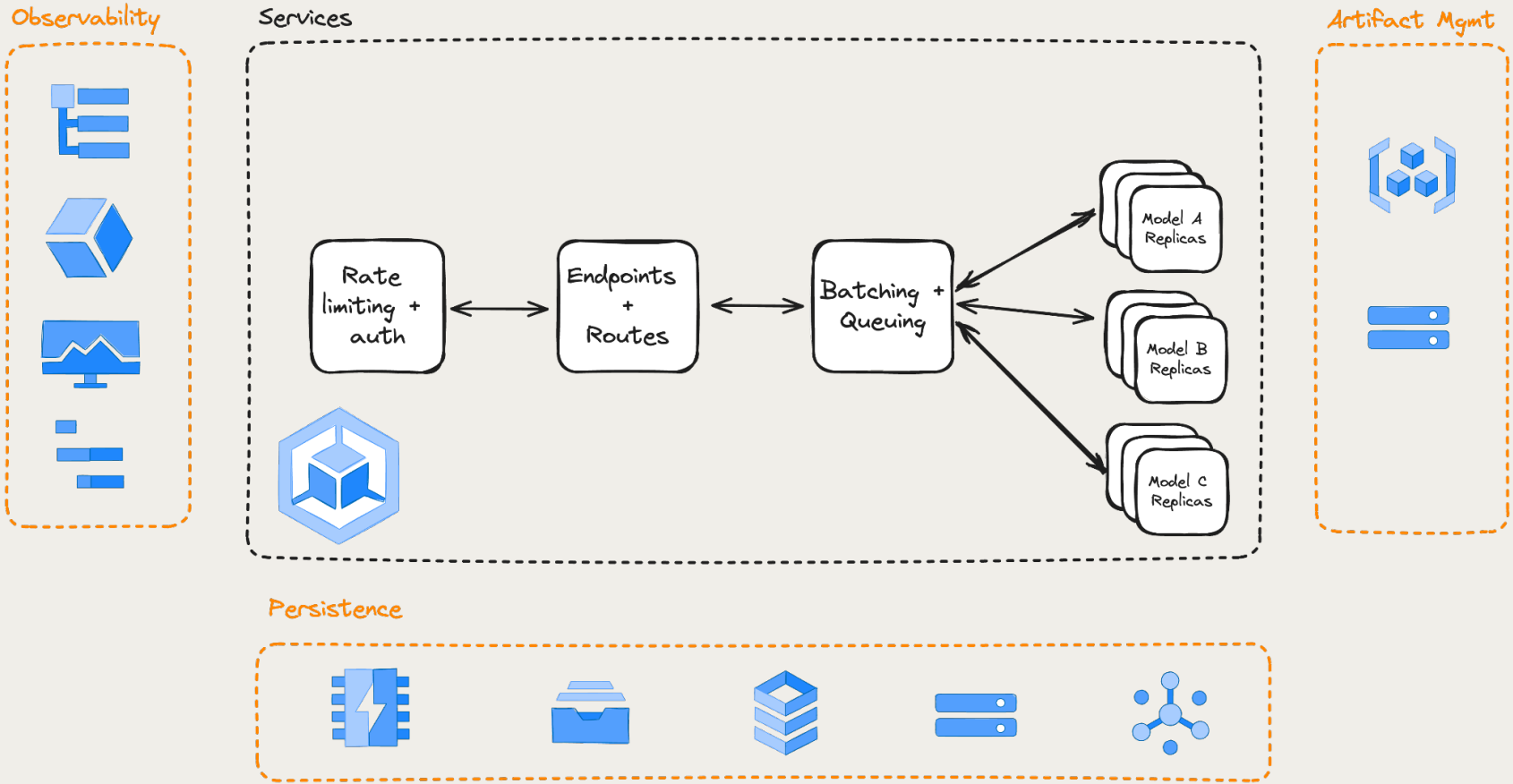
HTTPS://XKCD.COM/1838/

LLM Serving Stack

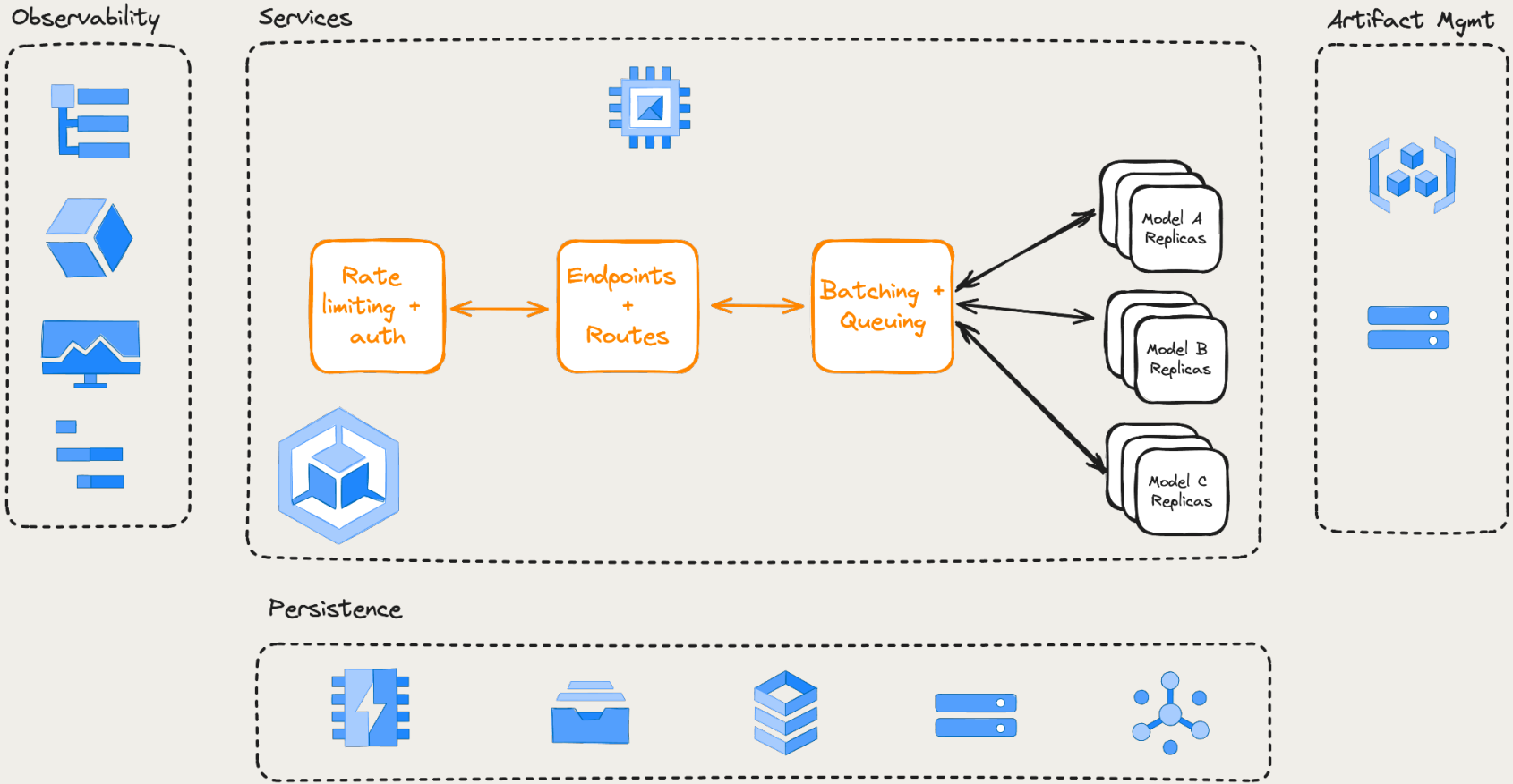
System Components



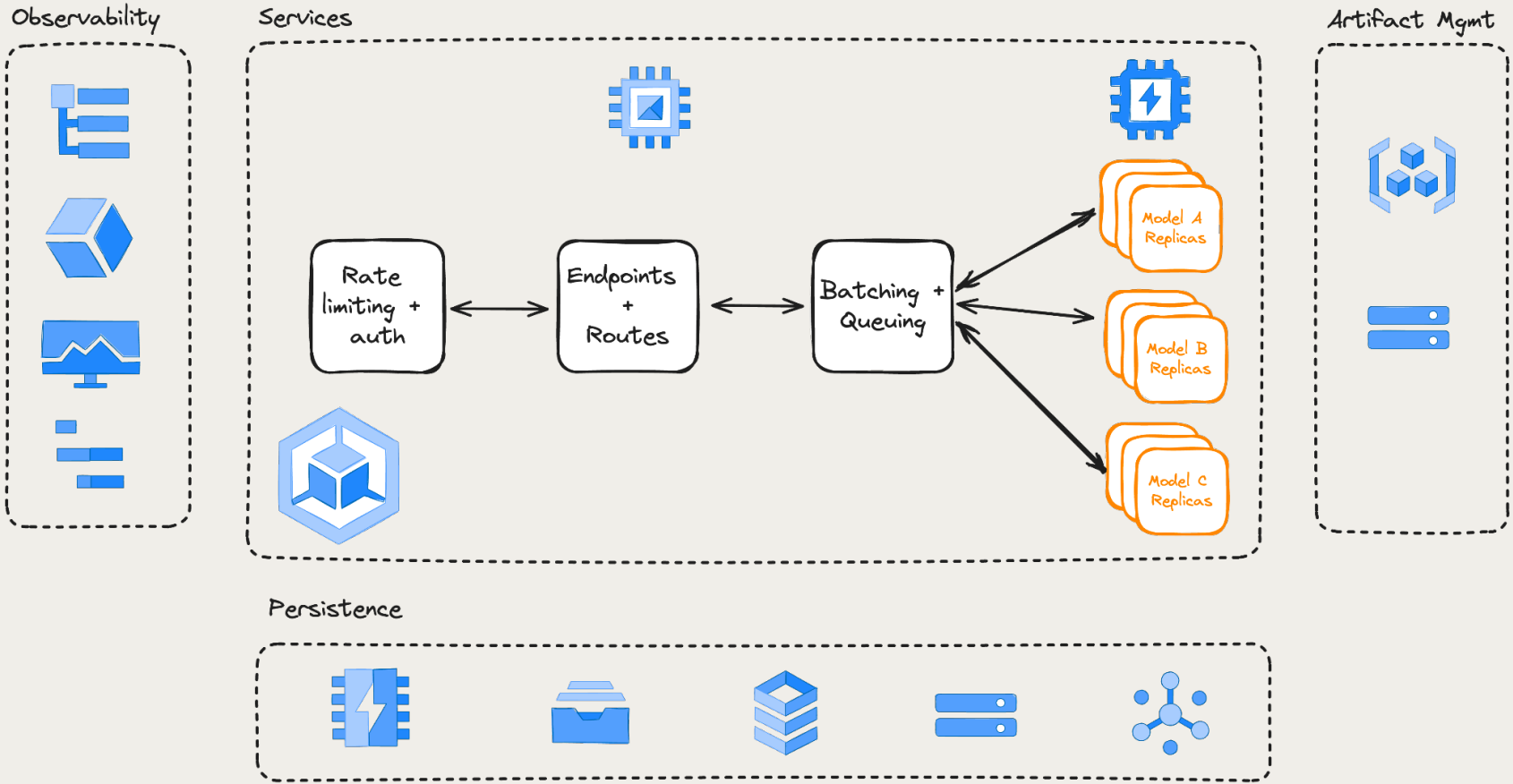
System Components

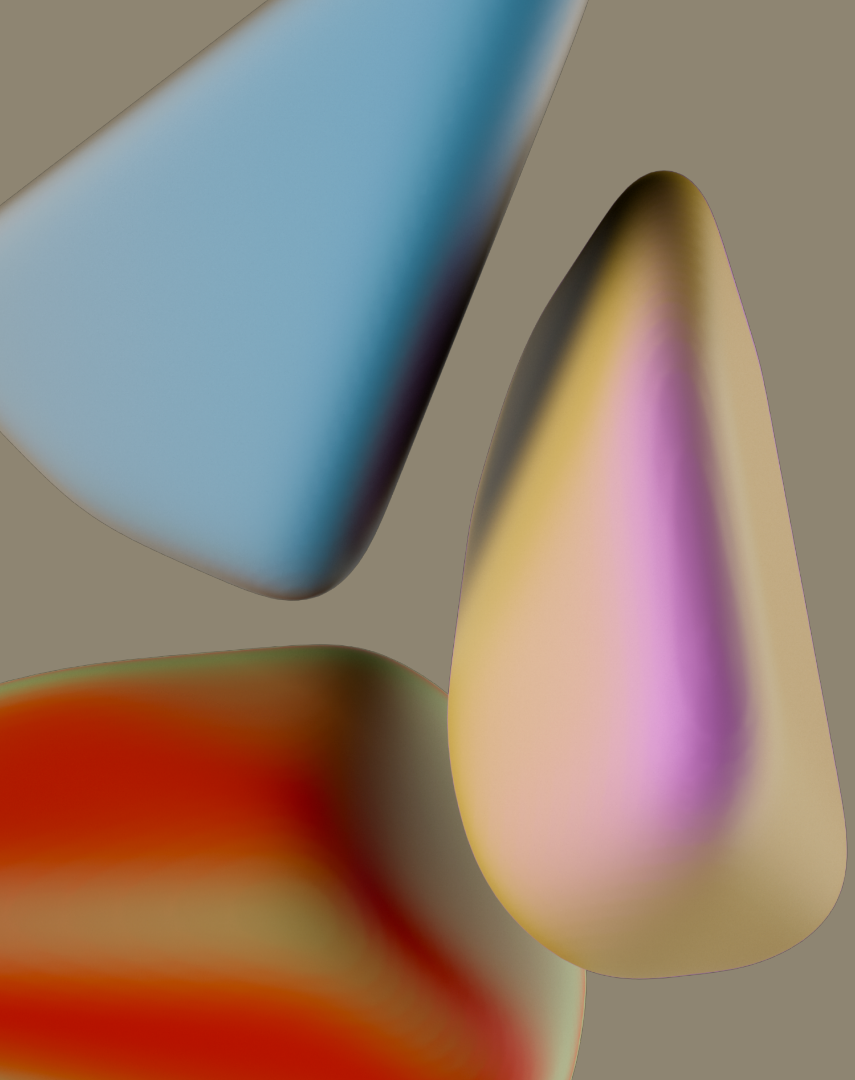


System Components



System Components



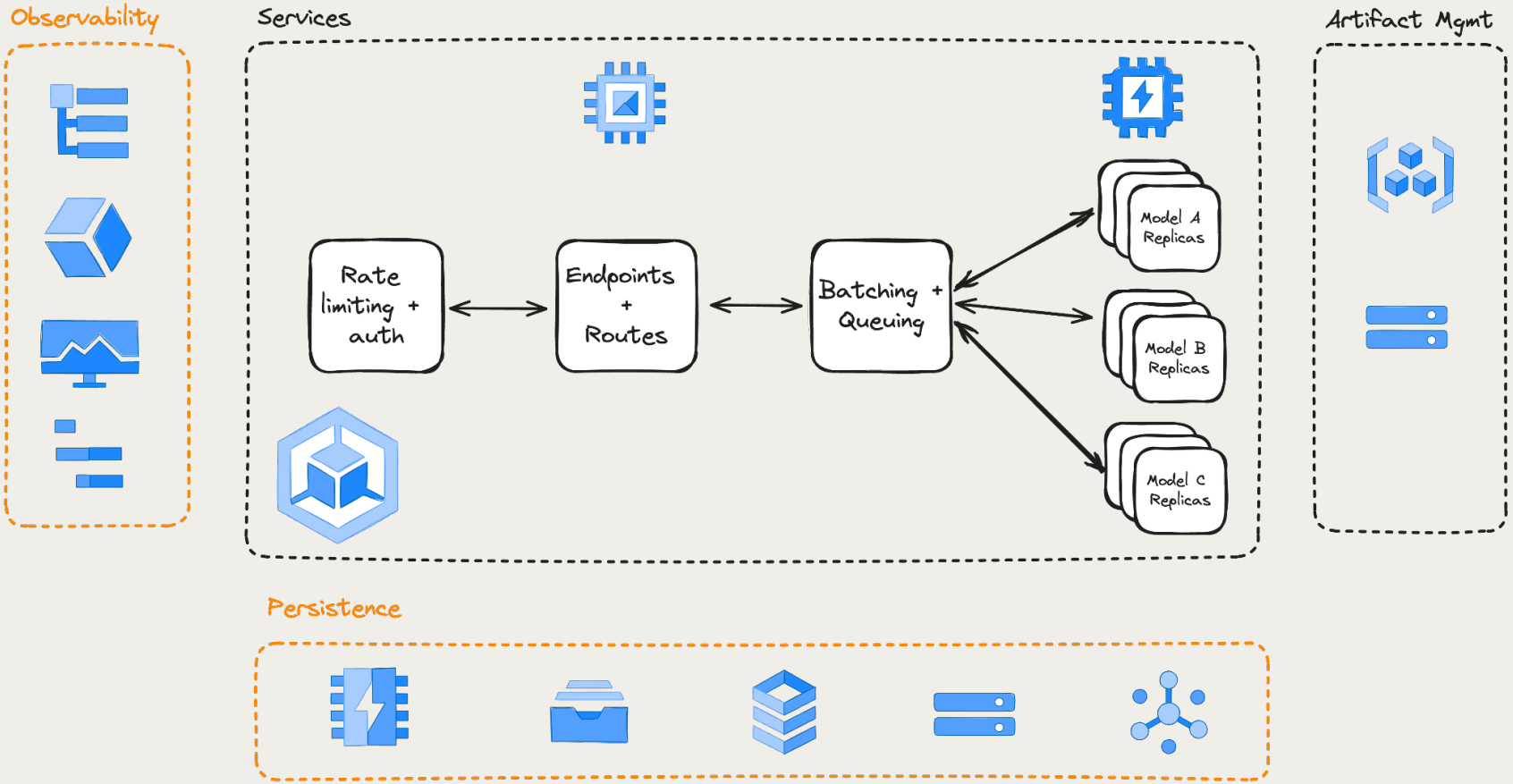


Why do companies need **private LLM** deployments?

Why private deployments?

- Compliance & security controls
- Data volumes & latency
- Single plane for availability and reliability

Challenges


Observability
Services
Artifact Mgmt
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
Persistence

Cloud Agnostic Serving

Managed Services

Multiple Targets

Lesson 1 - Invest In Code Abstractions

- Uniform API for service access
- Functional Options Pattern

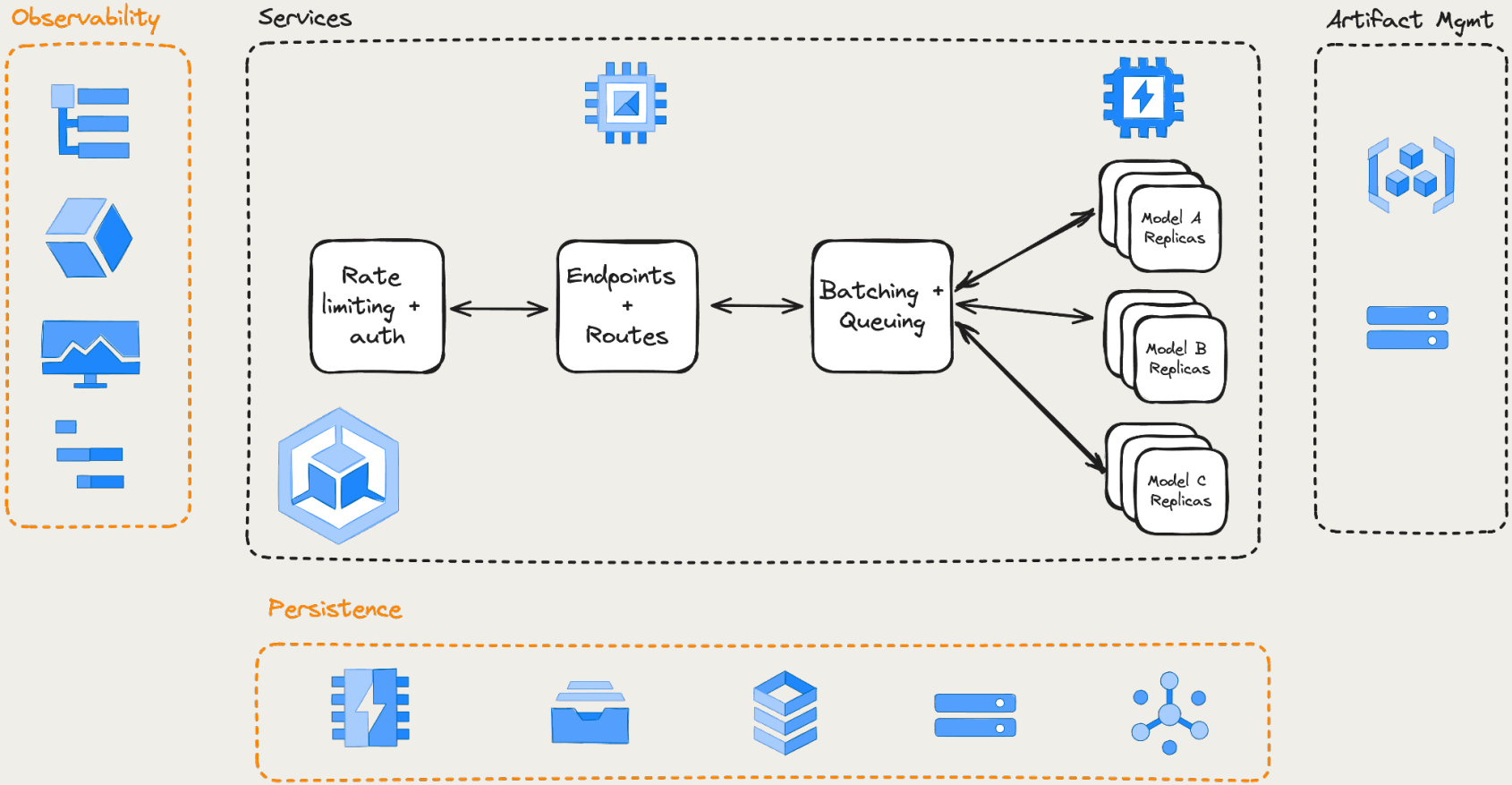
```
type ServerOption func(*Server)
func NewServer(opts ...ServerOption) *Server {
     server := &Server {
     // initialize some defaults
     }
     // Apply the options on the target struct
     for _, opt := range opts {
          opt(server)
     }
     return server
}
```

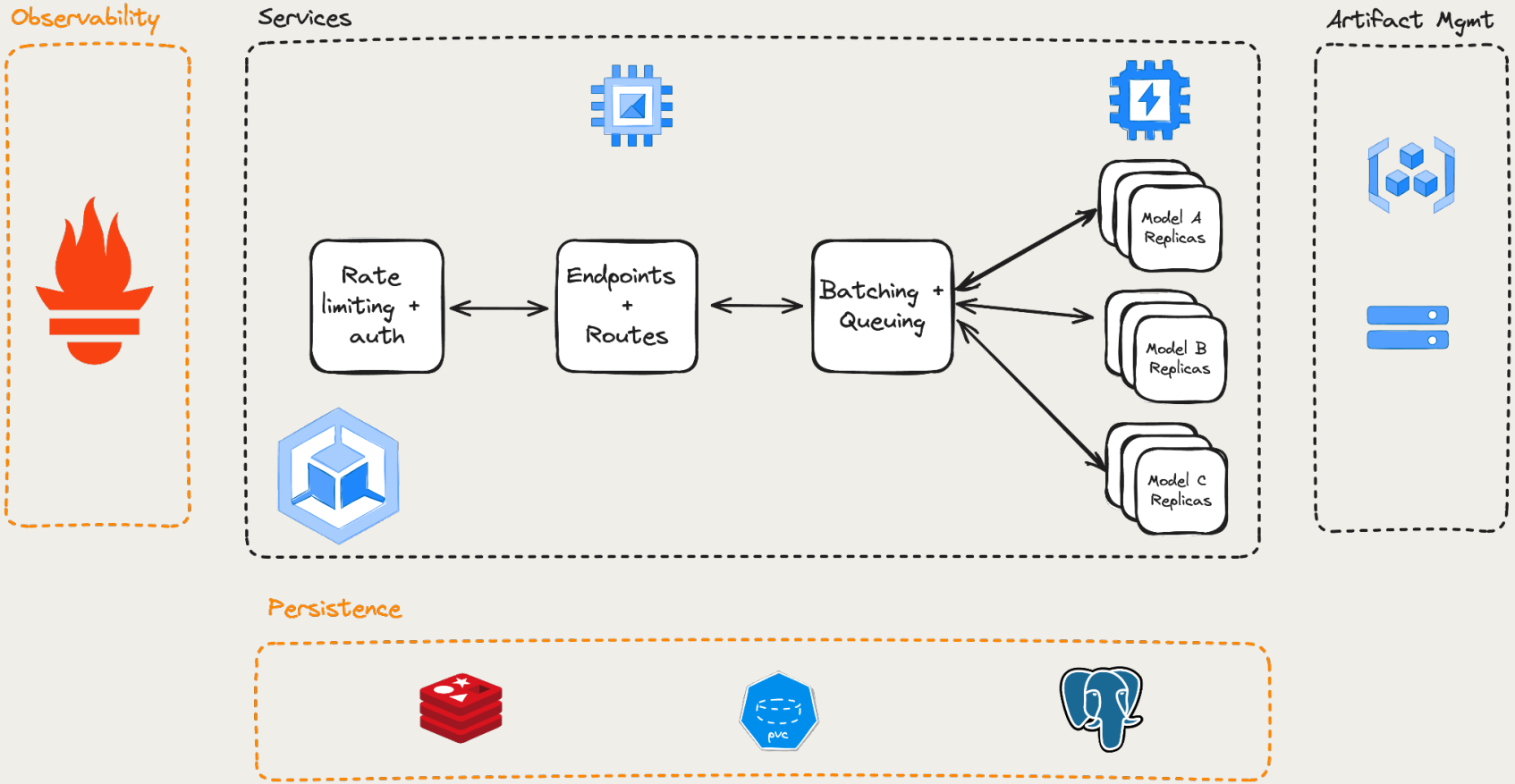
Lesson 1 - Invest In Code Abstractions

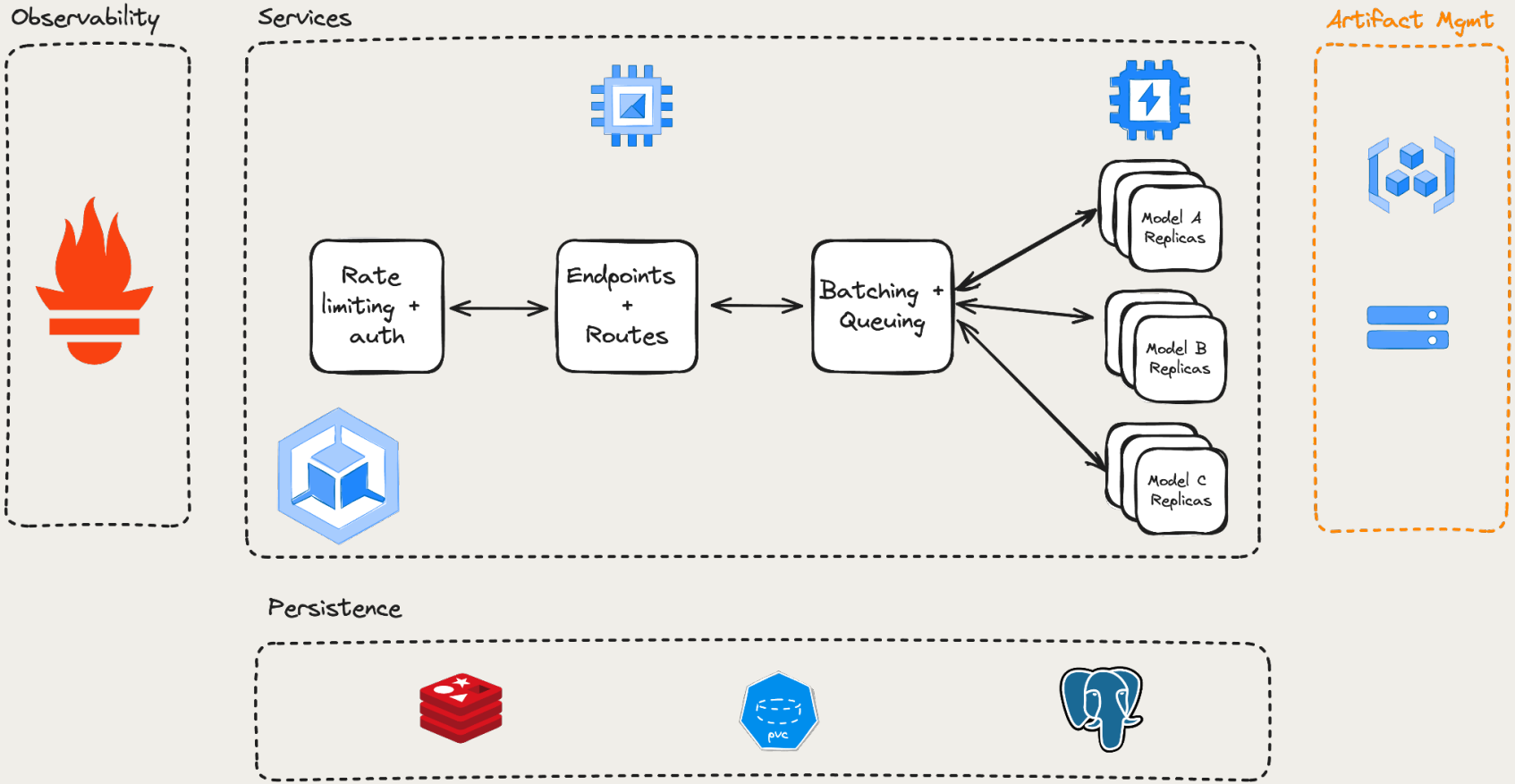

```
func WithCache(cache *kvcache.Cache) ServerOption {
    return func(s *ServerOption) {
        s.cache = cache
    }
}
func WithFeatureFlagLib(ff *featureflag.Client) ServerOption {
    return func(s *ServerOption) {
        s.ffClient = ff
    }
}
```

Lesson 2 - Don’t Reinvent the Wheel

- Object Storage Abstraction (Thanos objstore)
- User-provided PVCs for storage
- Metrics standardized through Prometheus
- Workload identity


Observability
Services
Artifact Mgmt
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
Persistence


Observability
Services
Artifact Mgmt
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
Persistence
pvc


Observability
Services
Artifact Mgmt
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
Persistence
pvc

Delivering Model Weights Securely

- Authentication
- Scalability
- Cost

Option 1

Exploring Signed URLs

- Short-lived URL
- Can only be for exact object URIs
- Our model weights are quite large

Option 2

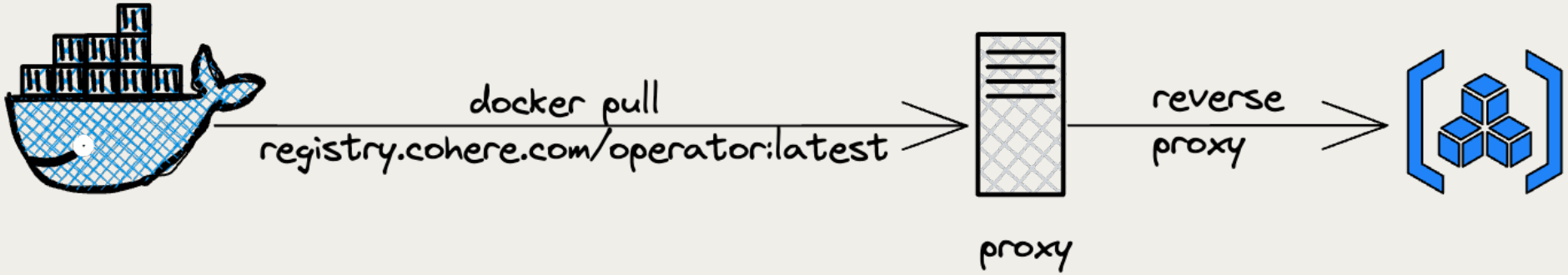
Delivering Artifacts to Users' Object Storage

- Workload Identity Federation to grant access
- Required unique handling per customer
- Supporting the solution for on-prem is difficult

Option 3

Bundling with Container Images

- Container images are delivered through a proxy



Option 3

Bundling with Container Images

- Less flexible to release due to tight coupling
- Negates the NFS optimizations for large models
- Longer time to patch CVEs in the runtime image

Lesson 3 - OCI Artifacts are a powerful standard

- Utilizes the same infrastructure as containers to store arbitrary files
- Consolidates security and management into one solution
- Registries are evolving as generic artifact stores

Lesson 3 - OCI Artifacts are a powerful standard

- De facto tool for storing and managing OCI artifacts in registries
- Focuses on non-image artifacts
- Rich library support for building custom registry clients

OCI Artifacts - Naive Approach

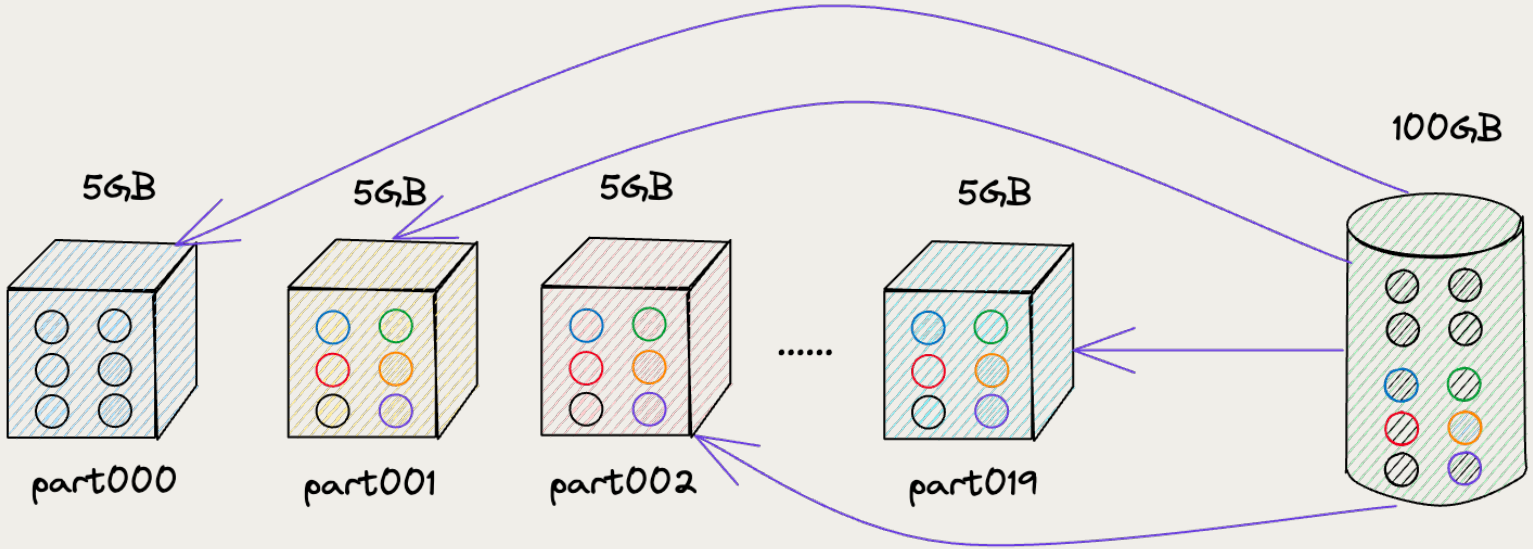
- Dump the model weights as one layer
- ORAS compresses directories into gzipped tarballs and marks it to be unpacked when pulled

OCI Artifacts - Naive Approach


```
{
 "schemaVersion": 2,
 "config": {
  "mediaType": "application/vnd.cohere.models.v1",
  "digest": "sha256:34f392306782dc14f91762f17ae8e5b94a8f3ea7a7ed90e681e121478affce4c",
  "size": 2
 },
 "layers": [
  {
   "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
   "digest": "sha256:17ec76743262480504cf365e43d526f965acdd4a2654dfcb8b25ef08280eabd7",
   "size": 787,
   "annotations": {
    "io.deis.oras.content.digest":
"sha256:6efa1c3cfcd1bbb5f916ac9fb67831e81191a6c30ff09552905064dcde024b43",
    "io.deis.oras.content.unpack": "true",
    "org.opencontainers.image.title": "cache"
   }
  }
 ]
}
```

OCI Artifacts - Binpacking Into Layers

- Separate files into folders of equal size



OCI Artifacts - Binpacking Into Layers

- Call ORAS Copy API with the list of folders
- Use a PostCopy callback to rebuild the directory structure
- Annotate the layers during copy time with a “hint”

OCI Artifacts - Binpacking into Layers


```
{
 "schemaVersion": 2,
 "config": {
  "mediaType": "application/vnd.cohere.models.v1",
  "digest": "sha256:34f392306782dc14f91762f17ae8e5b94a8f3ea7a7ed90e681e121478affce4c",
  "size": 2
 },
 "layers": [
  {
   "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
   "digest": "sha256:17ec76743262480504cf365e43d526f965acdd4a2654dfcb8b25ef08280eabd7",
   "size": 4914009886,
   "annotations": {
    "io.deis.oras.content.digest":
"sha256:6efa1c3cfcd1bbb5f916ac9fb67831e81191a6c30ff09552905064dcde024b43",
    "io.deis.oras.content.unpack": "true",
    "org.opencontainers.image.title": "part000"
    "com.cohere.moveup": "true",
   }
  },
```

OCI Artifacts - Binpacking into Layers

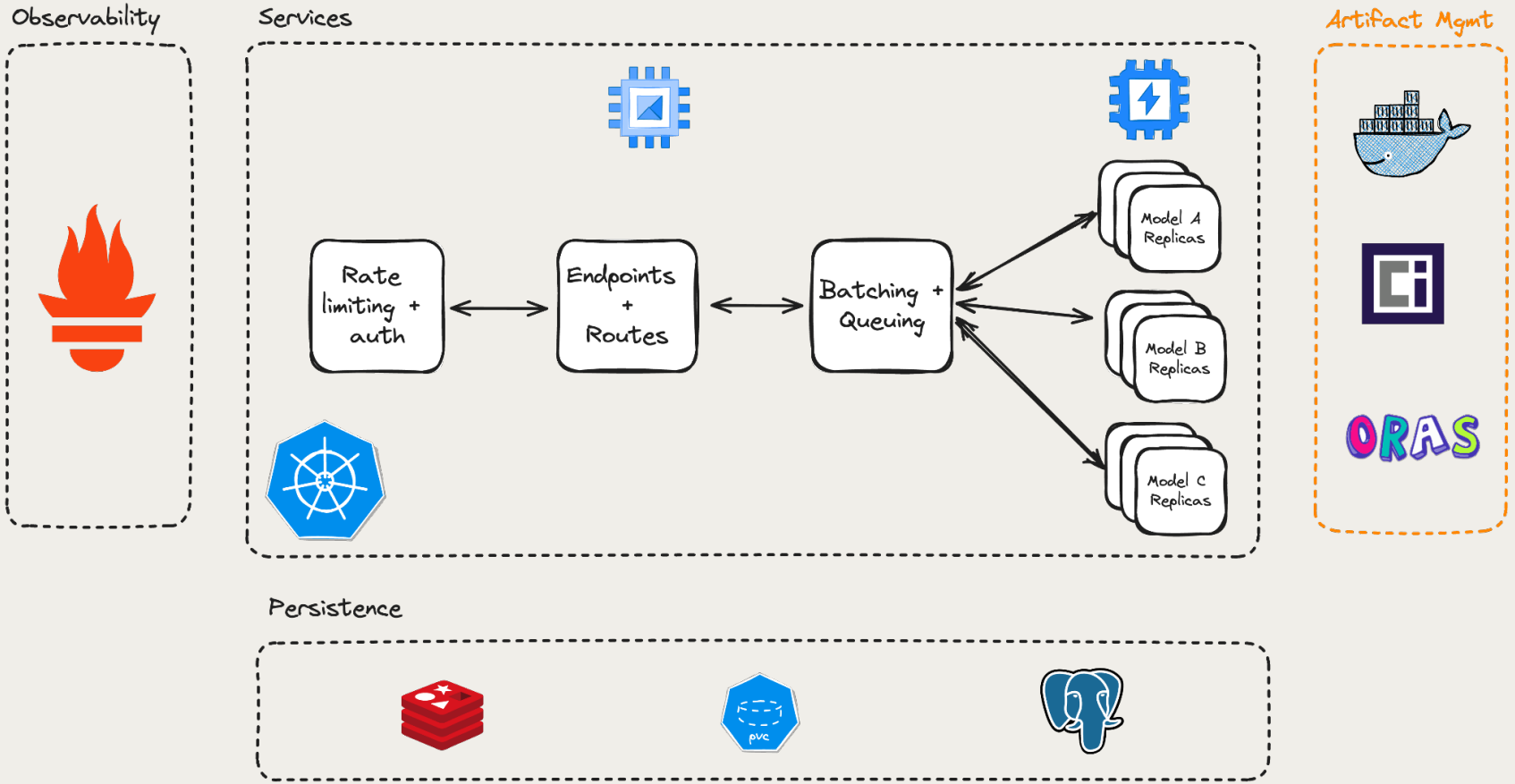

```
  {
   "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
   "digest": "sha256:ab63f8760e931e2c70dec6e9e0a5fad0487bffdbffb6b52065aa98618490811a",
   "size": 4927301871,
   "annotations": {
    "io.deis.oras.content.digest":
"sha256:6efa1c3cfcd1bbb5f916ac9fb67831e81191a6c30ff09552905064dcde024b43",
    "io.deis.oras.content.unpack": "true",
    "org.opencontainers.image.title": "part001"
     "com.cohere.moveup": "true",
   }
  },
       ..... Up to part019
 ]
}
```

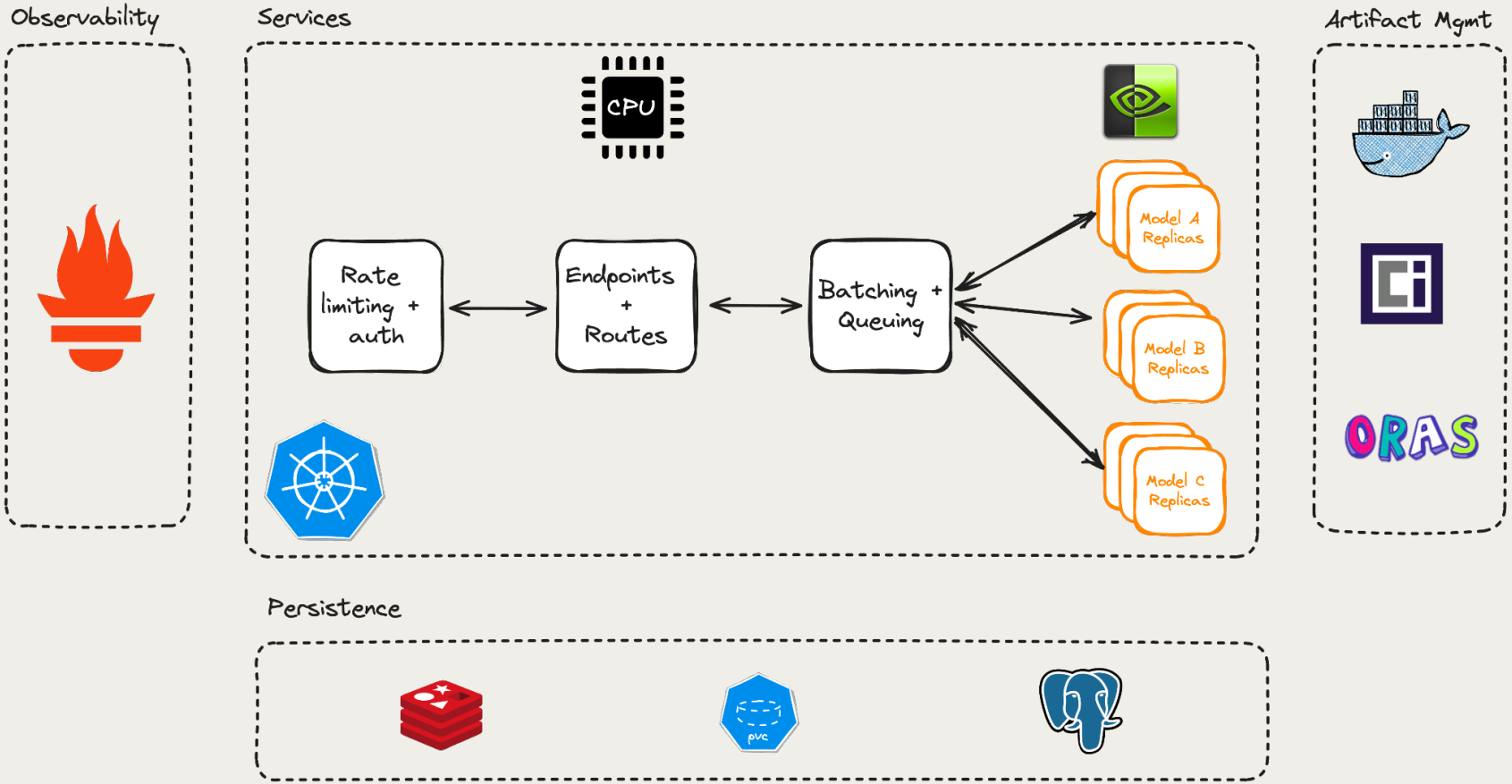
OCI Artifacts - Binpacking Into Layers


```
// Add files to a file store
fileDescriptors := make([]ocispec.Descriptor, len(filePaths))
for i, f := range filePaths {
            fileDescriptor, _ := fileStore.Add(ctx, f, mediaType, "")
            fileDescriptor.Annotations[k] = annotations
            fileDescriptors[i] = fileDescriptor
}

// Pack the files and tag the packed manifest
manifestDescriptor, err := oras.Pack(ctx, fileStore, artifactType,
fileDescriptors, oras.PackOptions{
            PackImageManifest:   true,
})

// Download Time
copyOptions.PostCopy = func(ctx context.Context, desc ocispec.Descriptor) error {
     if _, ok := desc.Annotations["com.cohere.moveup"]; ok {
            // move the downloaded contents up one level in the filestore
     }
}
```

OCI & Oras - Benefits

- Authenticity and integrity
- Unlocking encryption scenarios
- Storage space reduction
- ORAS built in retries for failed layer downloads
- On-the-fly config changes
- Easier path to air-gapped


Observability
Services
Artifact Mgmt
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
ORAS
Persistence
pvc


Observability
Services
Artifact Mgmt
CPU
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
ORAS
Persistence
pvc

GPUs Across Clouds

Provisioning

Small vs large scale

Lesson 4 - Provisioning GPUs Reliably is tricky

- Nvidia Drivers
- Nvidia Device Plugin
- DCGM Exporter
- (Somewhat) Solution: Nvidia GPU Operator

Lesson 5 - Inconsistent Identifiers

- Device name fetched through NVML is inconsistent across clouds

```
root@aks:/# nvidia-debugdump --list
Found 4 NVIDIA devices
            Device ID:       0
            Device name:      NVIDIA A100 80GB PCIe
root@gcp:/# nvidia-debugdump --list
Found 4 NVIDIA devices
            Device ID:       0
            Device name:      NVIDIA A100-SXM4-80GB
```

Lesson 5 - Inconsistent Identifiers

- Harder to control and binpack workloads due to the lack of common node labels
- Solution: Node/GPU Feature Discovery

```
nvidia.com/cuda.runtime.major=11

nvidia.com/gpu.product=NVIDIA-A100-SXM4-40GB

nvidia.com/cuda.driver.major=470

nvidia.com/mig.capable=true

nvidia.com/gpu.count=8M4-80GB
```

Lesson 6 - Node Upgrades Are Bound To Fail

- Striking the balance between speed and availability for surge upgrades
- Blue/Green is ideal but GPU costs
- GPU Capacity crunch makes it difficult
- Solution: new pool, taint and set max nodes to 1 for the old pool, slowly HPA onto the new pool

Lesson 7 - Autoscaling LLMs is Challenging

- Hacky hardcoded assumptions in autoscaler

```
unschedulablePodWithGpuTimeBuffer = 30 * time.Second
```

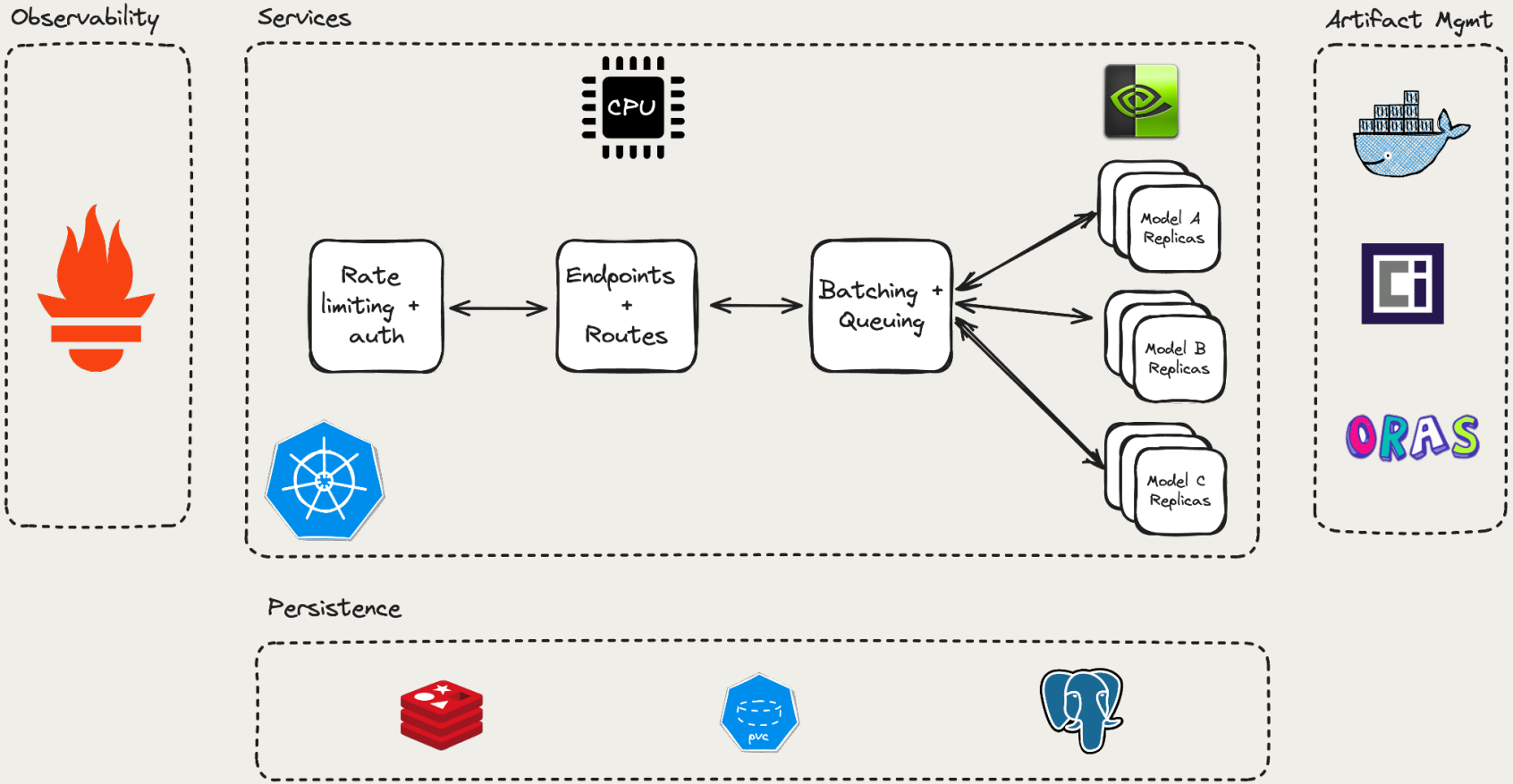
- Back-off delays can lead to long scale up times
- Recommendation: tweak autoscaler params

```
--initial-node-group-backoff-duration
--max-node-group-backoff-duration
--node-group-backoff-reset-timeout
```

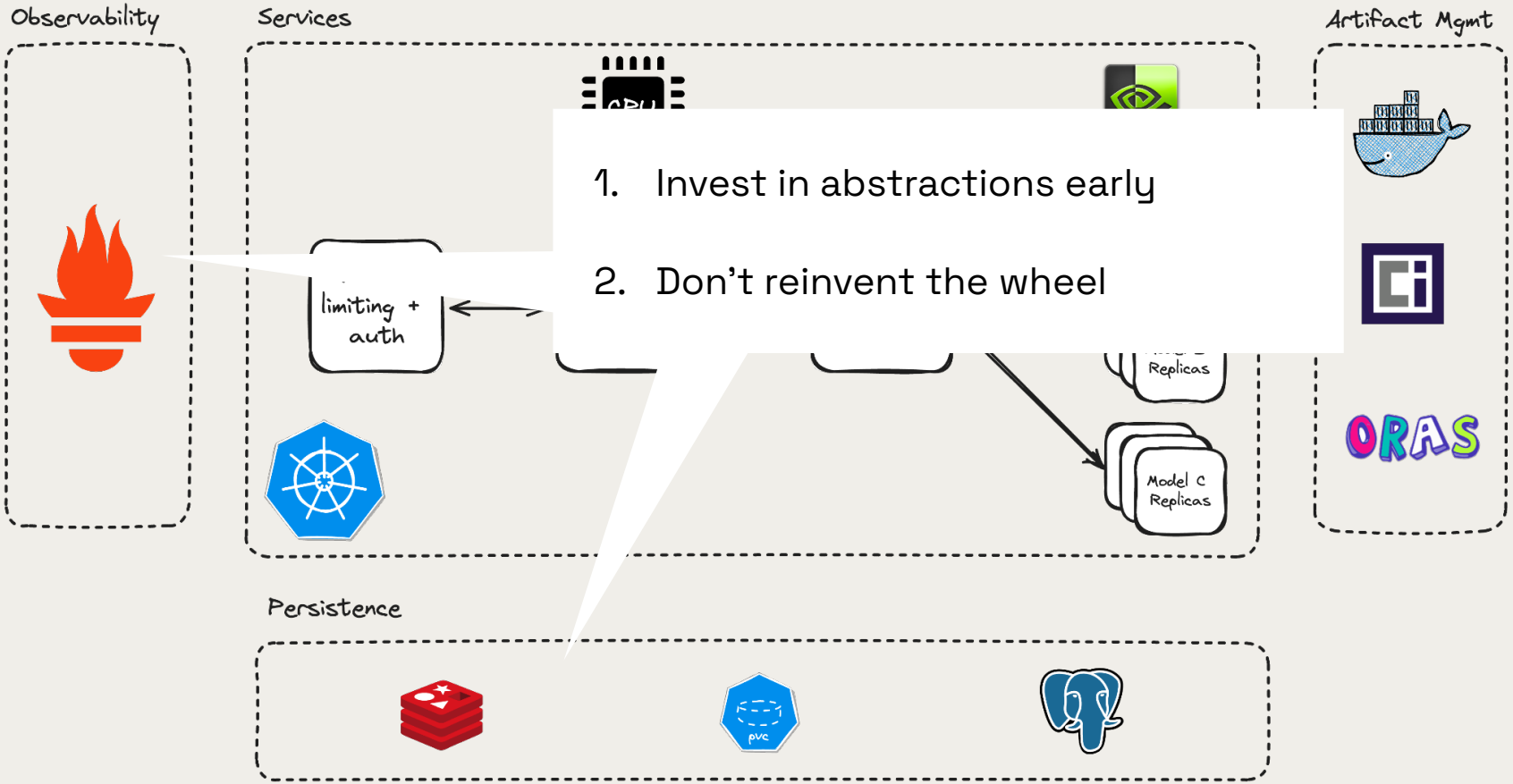
Lesson 7 - Autoscaling LLMs is Challenging

- Request-based
- GPU utilization (duty_cycle)
- Inference Server Queue Time
- Global Queue based on batches running

In Summary....

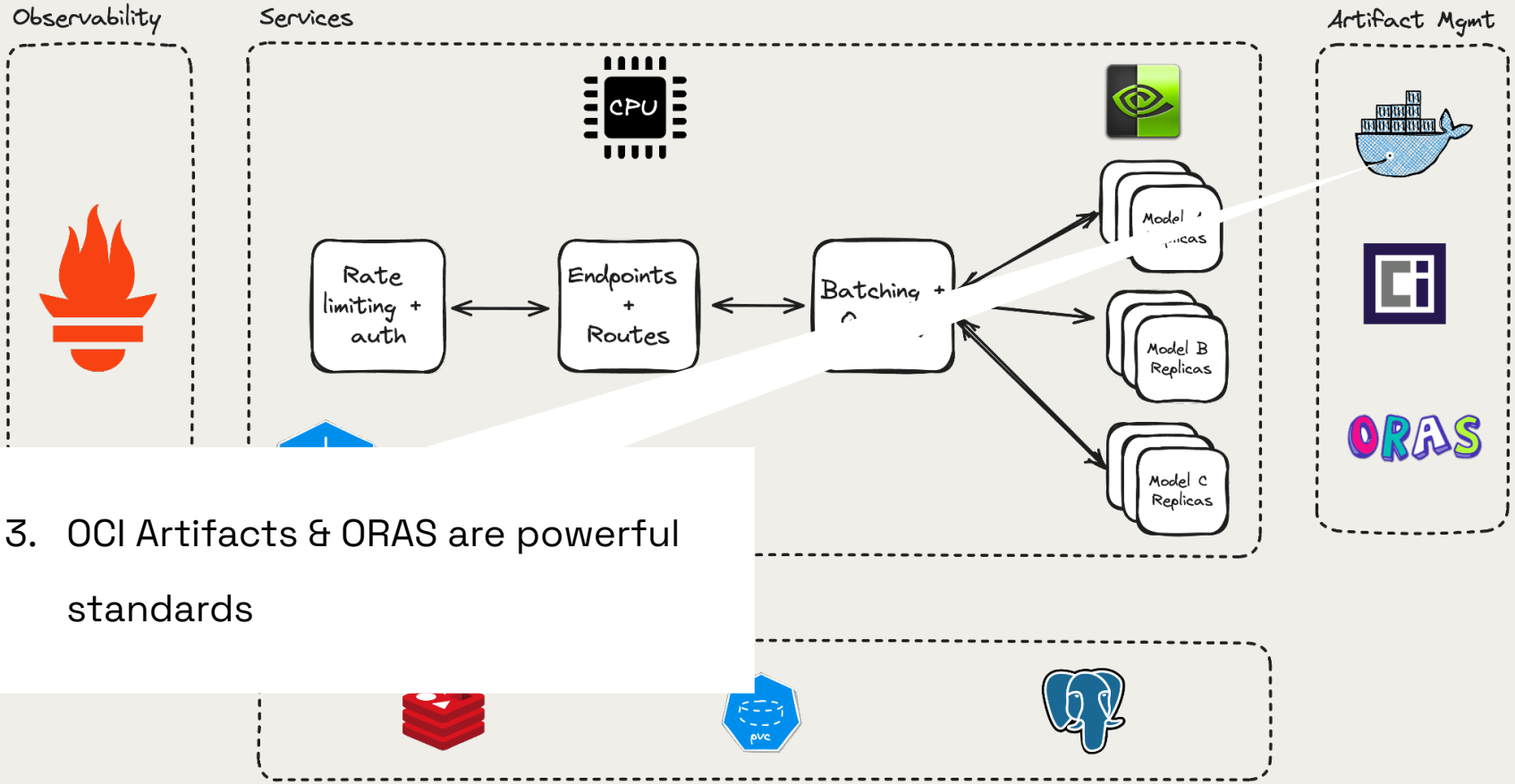

Observability
Services
Artifact Mgmt
CPU
Rate limiting + auth
Endpoints + Routes
Batching + Queuing
Model A Replicas
Model B Replicas
Model C Replicas
ORAS
Persistence
PVC

Common Challenges



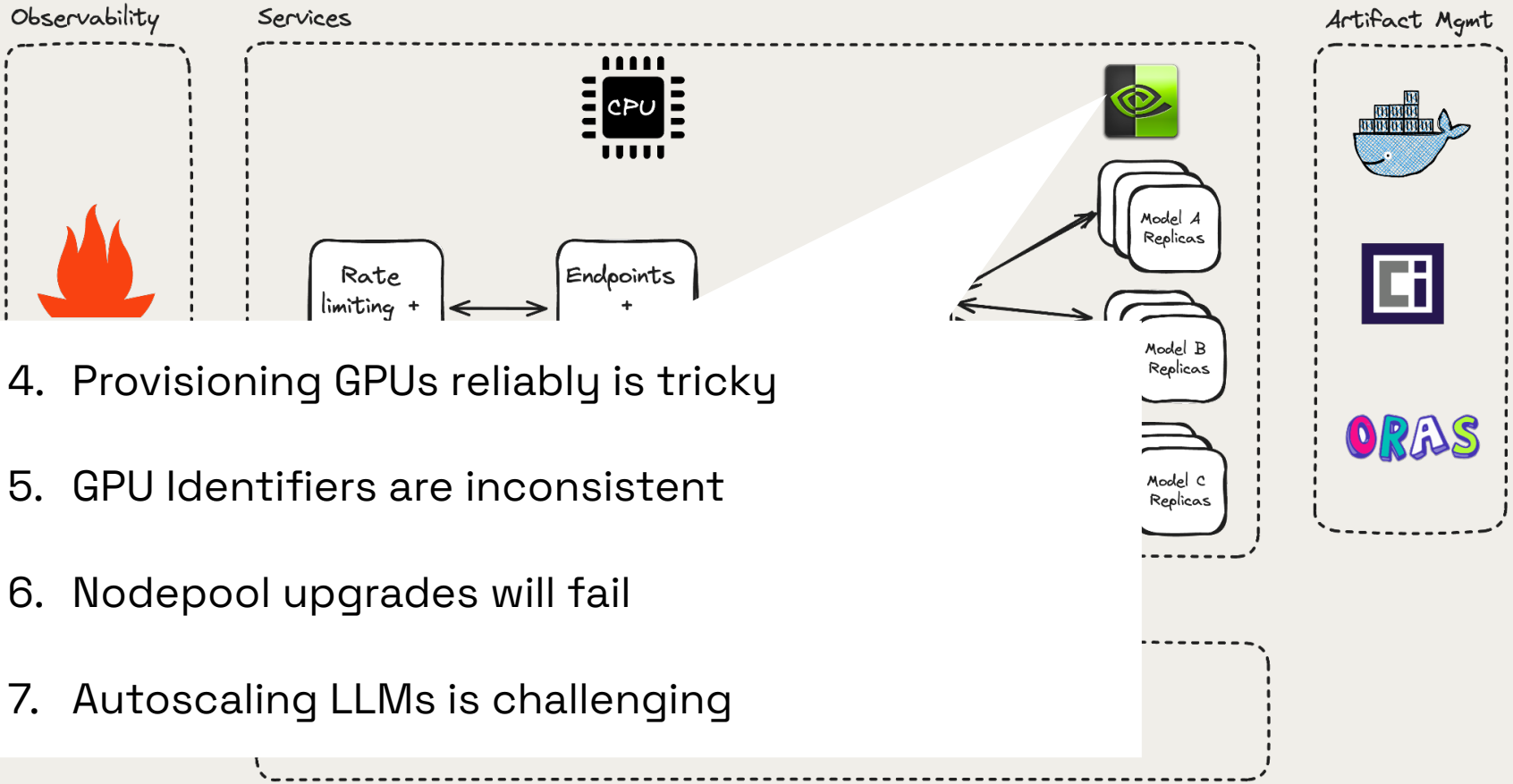
1. Invest in abstractions early
2. Don't reinvent the wheel

Delivering Artifact Weights Securely



3. OCI Artifacts & ORAS are powerful standards

GPUs Across Clouds



4. Provisioning GPUs reliably is tricky
5. GPU Identifiers are inconsistent
6. Nodepool upgrades will fail
7. Autoscaling LLMs is challenging

cohere

Thank you for joining!