超越容器，在 macOS 上使用 Kubernetes 编排 LLM

唐威强 摩尔线程
叶晓冬 摩尔线程


SHANGHAI
KCD
GROWING CLOUD NATIVE TOGETHER
Kubernetes Community Days Shanghai 2024

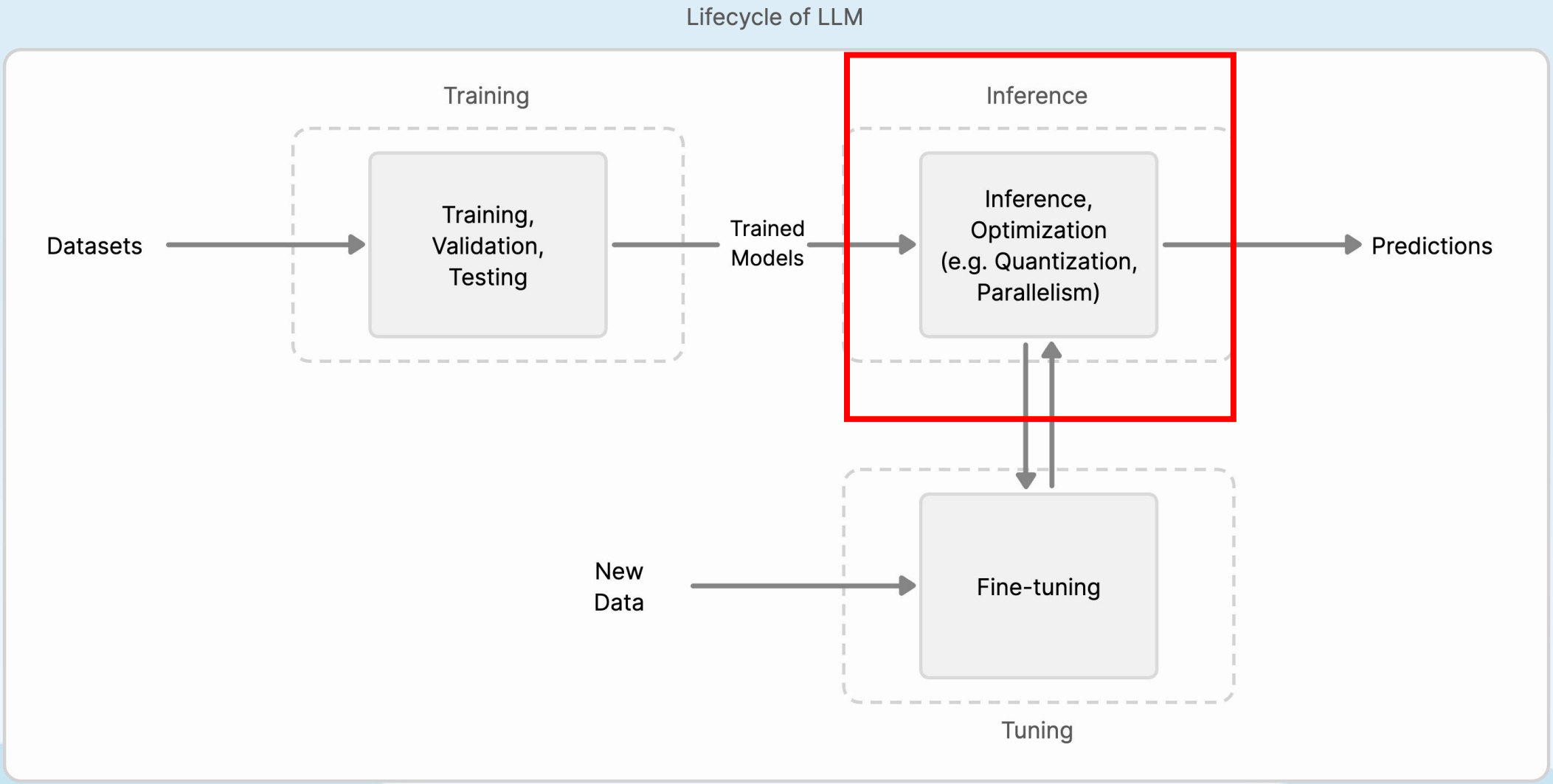
Beyond Containers, Orchestrate LLMs with Kubernetes on macOS

Xiaodong Ye, Moore Threads
Weiqiang Tang, Moore Threads

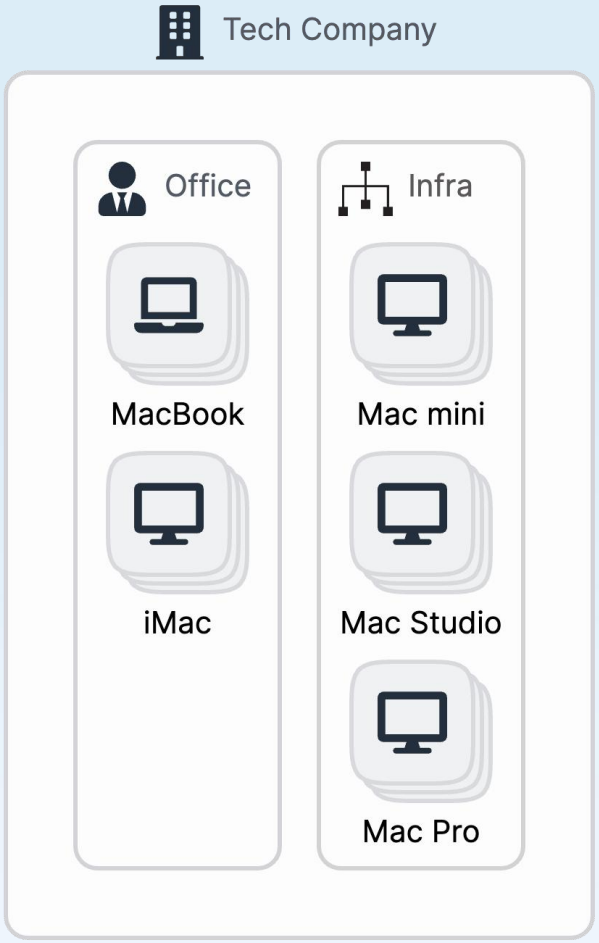
Agenda

- The Opportunity with Apple Silicon
- Kubernetes with LLMs - An Evolving Landscape
- Embracing macOS for LLM Inference
- Technical Deep Dive
- Demo: Deploying Open-Source Foundation Models
- Future Outlook
- Related links

In this talk


Lifecycle of LLM
Training
Inference
Datasets
Training,
Validation,
Testing
Trained
Models
Inference,
Optimization
(e.g. Quantization,
Parallelism)
Predictions
New
Data
Fine-tuning
Tuning

The Opportunity with Apple Silicon Cost



- GPUs are expensive and hard to buy in China (no warranty in some cases)
- Macs are free to buy and have warranty of 2 years in China!
- There are many Macs with idle GPUs in the tech company - reuse them as Kubernetes nodes

The Opportunity with Apple Silicon Performance

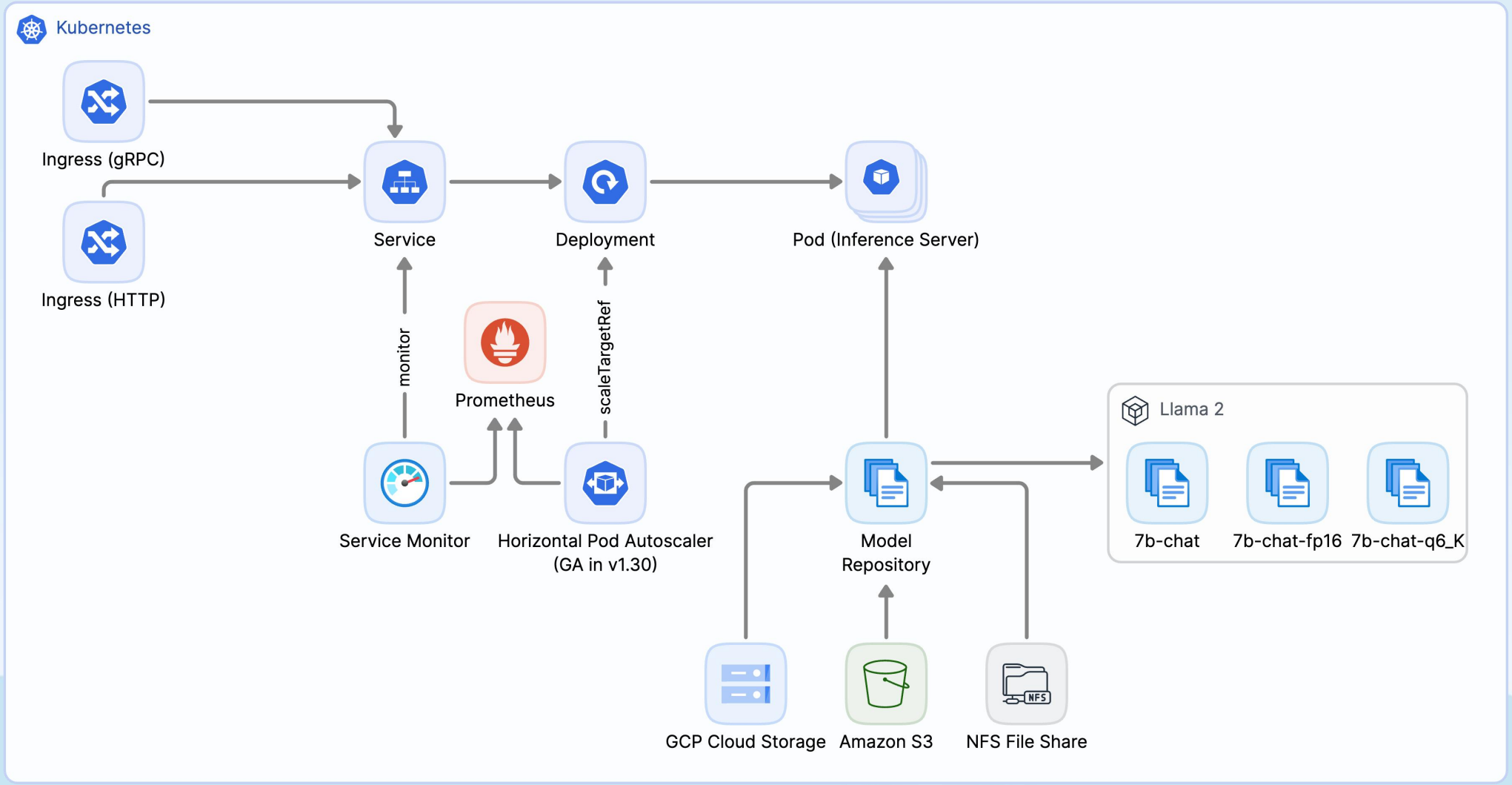
- Killer Features
  - Unified Memory Architecture (UMA)
    - Up to 192GB (~155GB can be accessed by Metal. Llama 2 7b ~ 3.8GB/13b ~ 7.4GB/70b-fp16 ~ 138GB)
    - 800GB/s of memory bandwidth
  - High performance GPU and Neural Engine
- llama.cpp on Apple Silicon
  - First-class citizen - optimized via ARM NEON, Accelerate and Metal frameworks
  - Performance
    - PP = Prompt processing (bs=512)

LLaMA 7B

	BW [GB/s]	GPU Cores	F16 PP [t/s]	F16 TG [t/s]	Q8_0 PP [t/s]	Q8_0 TG [t/s]	Q4_0 PP [t/s]	Q4_0 TG [t/s]
✅ M1 [1]	68	7			108.21	7.92	107.81	14.19
✅ M1 [1]	68	8			117.25	7.91	117.96	14.15
✅ M1 Pro [1]	200	14	262.65	12.75	235.16	21.95	232.55	35.52
✅ M1 Pro [1]	200	16	302.14	12.75	270.37	22.34	266.25	36.41
✅ M1 Max [1]	400	24	453.03	22.55	405.87	37.81	400.26	54.61
✅ M1 Max [1]	400	32	599.53	23.03	537.37	40.2	530.06	61.19
✅ M1 Ultra [1]	800	48	875.81	33.92	783.45	55.69	772.24	74.93
✅ M1 Ultra [1]	800	64	1168.8	37.01	1042.95	59.87	1030.04	83.73
✅ M2 [2]	100	8			147.27	12.18	145.91	21.7
✅ M2 [2]	100	10	201.34	6.72	181.4	12.21	179.57	21.91
✅ M2 Pro [2]	200	16	312.65	12.47	288.46	22.7	294.24	37.87
✅ M2 Pro [2]	200	19	384.38	13.06	344.5	23.01	341.19	38.86
✅ M2 Max [2]	400	30	600.46	24.16	540.15	39.97	537.6	60.99
✅ M2 Max [2]	400	38	755.67	24.65	677.91	41.83	671.31	65.95
✅ M2 Ultra [2]	800	60	1128.59	39.86	1003.16	62.14	1013.81	88.64
✅ M2 Ultra [2]	800	76	1401.85	41.02	1248.59	66.64	1238.48	94.27
🟥 M3 [3]	100	8						
🟨 M3 [3]	100	10			187.52	12.27	186.75	21.34
🟨 M3 Pro [3]	150	14			272.11	17.44	269.49	30.65
✅ M3 Pro [3]	150	18	357.45	9.89	344.66	17.53	341.67	30.74
✅ M3 Max [3]	300	30	589.41	19.54	566.4	34.3	567.59	56.58
✅ M3 Max [3]	400	40	779.17	25.09	757.64	42.75	759.7	66.31
🟥 M3 Ultra	800	60						
🟥 M3 Ultra	800	80						

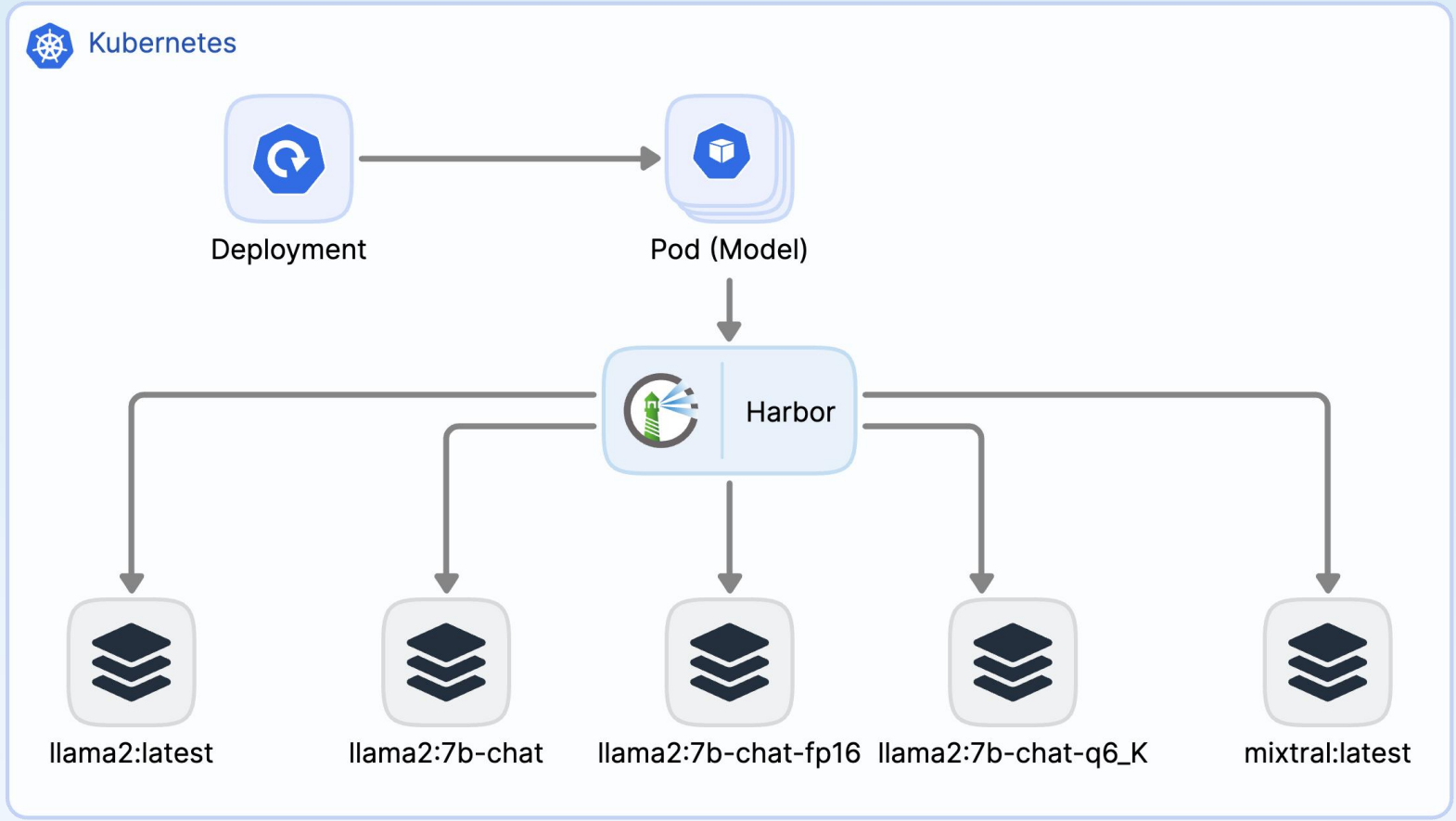
Kubernetes & LLMs - An Evolving Landscape

- A typical LLM inference service deployed in Kubernetes



Embracing macOS for LLM Inference

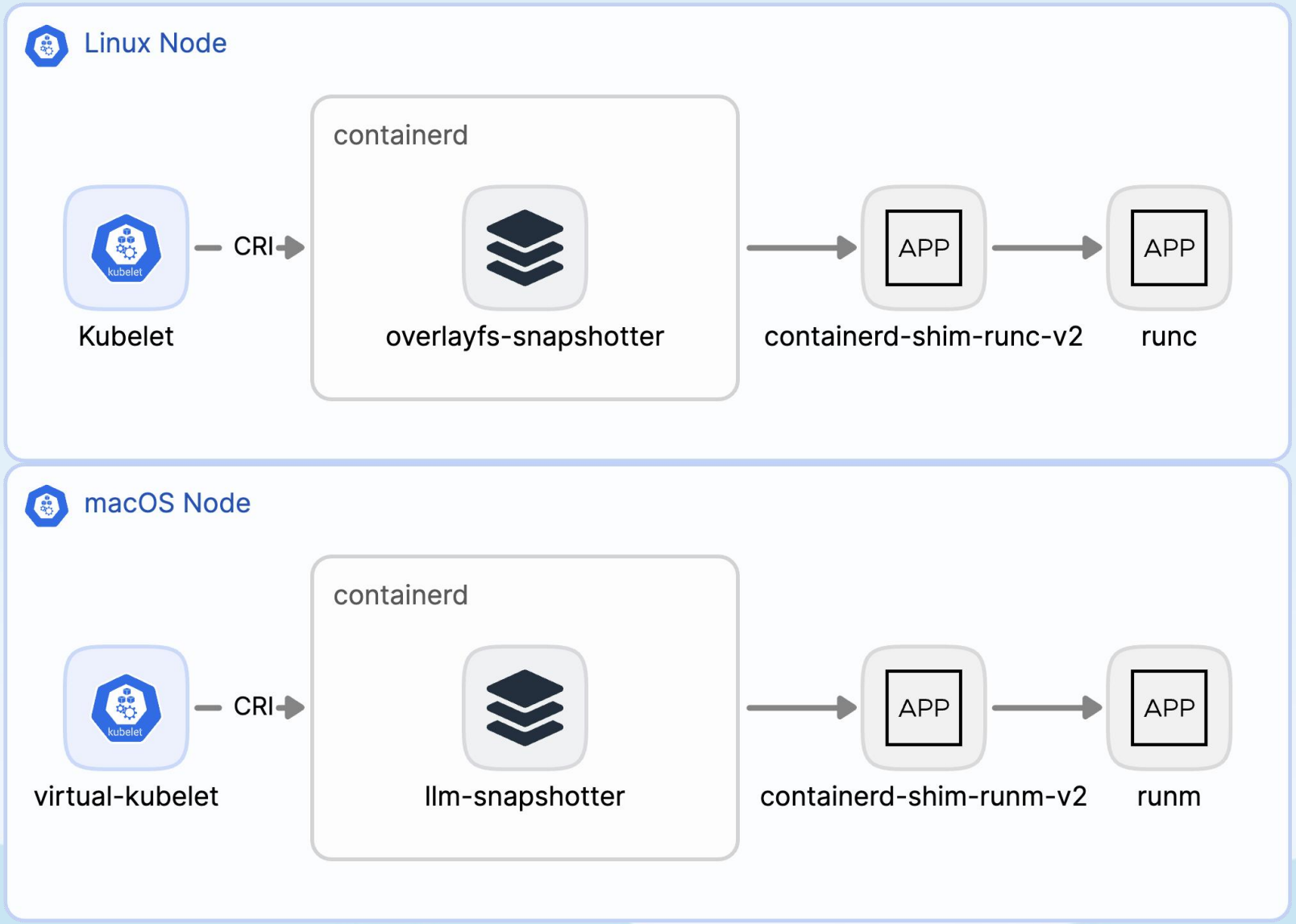
- Distribute LLMs by OCI image format
  - Get multiple benefits (layering, hashing, Auth & RBAC, retrial mechanism, etc)
  - No cloud native stack on macOS (Every container starts with a OCI image)
  - Proven by Ollama on local LLM



Technical Deep Dive

- Linux Node vs macOS Node
- LLM Image and New LLM SnapShotter
- runm - A Lightweight Inference Solution Derived from ggerganov/llama.cpp
- A specific network

Linux Node vs macOS Node



LLM Image and New LLM SnapShotter - 1

- OCI artifacts as LLM image

harbor.mthreads.com/llm/tinyllama:latest

Layers

"mediaType": "application/vnd.ollama.image.model",
"digest": "sha256:2af3b81862c6be03c769683af18efdadb2c33f60ff32ab6f83e42c043d6c7816",
"size": 637699456

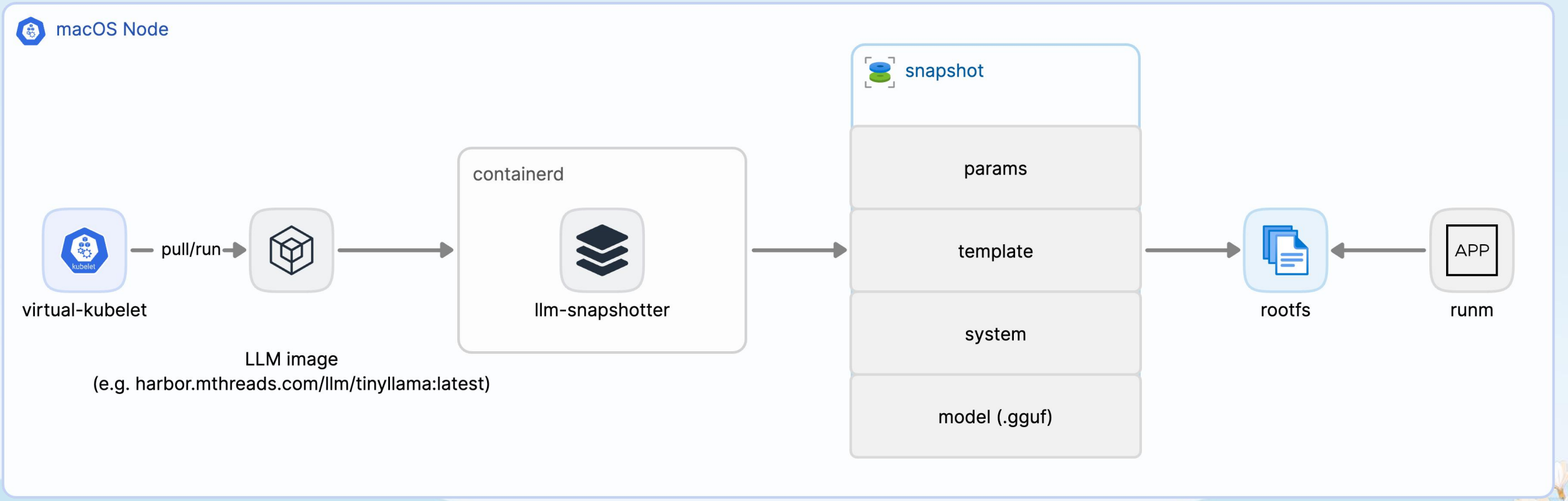
"mediaType": "application/vnd.ollama.image.template",
"digest": "sha256:af0ddbdaaa26f30d54d727f9dd944b76bdb926fdaf9a58f63f78c532f57c191f",
"size": 70

"mediaType": "application/vnd.ollama.image.system",
"digest": "sha256:c8472cd9daed5e7c20aa53689e441e10620a002aacd58686aeac2cb188addb5c",
"size": 31

"mediaType": "application/vnd.ollama.image.params",
"digest": "sha256:fa956ab37b8c21152f975a7fcdd095c4fee8754674b21d9b44d710435697a00d",
"size": 98

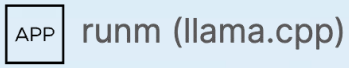
LLM Image and New LLM SnapShotter - 2

- Store LLM image locally as snapshot
- Create rootFS for runm



runm - A Lightweight Inference Solution Derived from ggerganov/llama.cpp

- Derived from ggerganov/llama.cpp
- Provide OpenAI API Compatibility



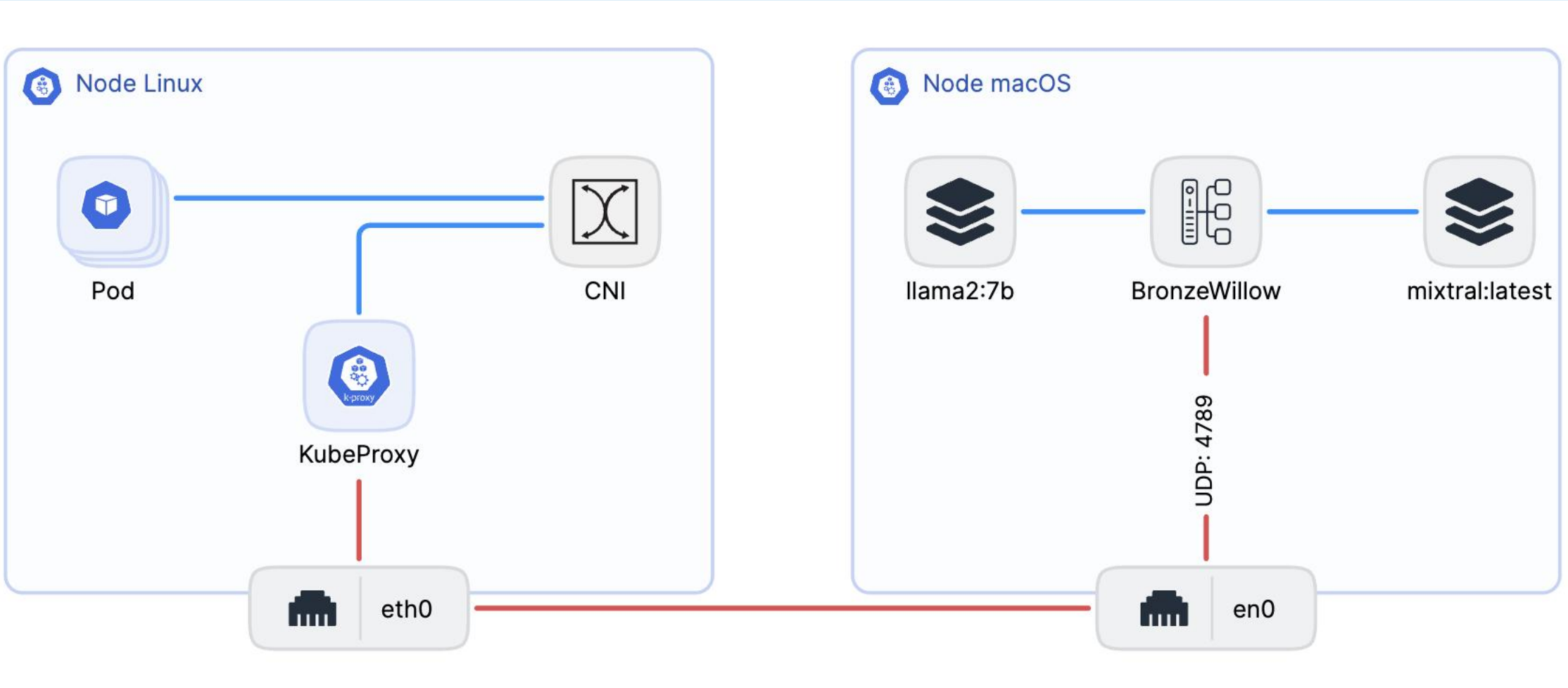
otool -L runm

```
/usr/lib/libresolv.9.dylib (compatibility version 1.0.0, current version 1.0.0)
/System/Library/Frameworks/CoreFoundation.framework/Versions/A/CoreFoundation (compatibility version 150.0.0, current version 2420.0.0)
/usr/lib/libc++.1.dylib (compatibility version 1.0.0, current version 1700.255.0)
/System/Library/Frameworks/Foundation.framework/Versions/C/Foundation (compatibility version 300.0.0, current version 2420.0.0)
/System/Library/Frameworks/CoreGraphics.framework/Versions/A/CoreGraphics (compatibility version 64.0.0, current version 1774.4.3)
/System/Library/Frameworks/Metal.framework/Versions/A/Metal (compatibility version 1.0.0, current version 343.14.0)
/usr/lib/libobjc.A.dylib (compatibility version 1.0.0, current version 228.0.0)
/System/Library/Frameworks/Security.framework/Versions/A/Security (compatibility version 1.0.0, current version 61123.100.169)
/usr/lib/libSystem.B.dylib (compatibility version 1.0.0, current version 1345.100.2)
```

Network - Overview

- A Specific CNI
  - A CNI, a VTEP, a L7 proxy with userspace network stack
- No privilege needed
- Using overlay network VXLAN/GENEVE
  - Supported by the most CNIs: Cilium, Calico, Antrea, .etc.
- Built in Rust
- Named BronzeWillow

Network - Traffic



Demo: Deploying Open-Source Foundation Models

- Create a Kubernetes Cluster (v1.30 with Antrea CNI)
- Join the macOS Node to the Cluster
    - Start the llm-containerd, llm-kubelet and BronzeWillow on macOS Node
- Create a tinyllama Deployment with 2 replicas
- Create a Service of the Deployment
- Create a Pod with *mods* (an AI CLI tool) built in
- Use *mods* with tinyllama Service

~/demo (-zsh)

```
(base) ➜ demo
```

~/demo (-zsh)

```
(base) ➜ demo
```

k9s (k9s)

```
Context: kubernetes-admin@kubernetes
Cluster: kubernetes
User:    kubernetes-admin
K9s Rev: v0.32.4
K8s Rev: v1.30.0
CPU:     n/a
MEM:     n/a
```

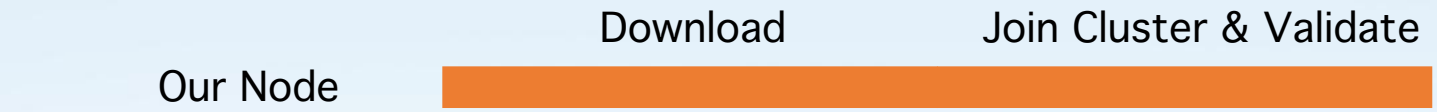
<c> Cordon <u> Uncordon
<y> YAML
<r> Drain
<?> Help

Demo 视频请查看：

https://www.bilibili.com/video/BV1YE421L7nz

Nodes(all)[2]

NAME↑	STATUS	ROLE	TAINTS	VERSION	PODS	AGE
k8s-node-control-plane	Ready	control-plane	1	v1.30.0	9	24h
k8s-node-worker-1	Ready	worker	0	v1.30.0	2	24h



This is just the start...

Download a Model Implement a service Build Image Distribute Image

Linux Node

Loop

Our Node

Maintenance is the real problem.

Future Outlook

- Built-in support of RAG and langchain composition
- Resource quota management of macOS

- Train model on KUAE and distribute models on

Related links

- https://github.com/containerd/containerd
- https://github.com/virtual-kubelet/virtual-kubelet
- https://github.com/ggerganov/llama.cpp
- https://github.com/jmorganca/ollama
- https://github.com/antrea-io/antrea
- https://github.com/charmbracelet/mods
- https://www.mthreads.com/product/KUAE
- https://kccnceu2024.sched.com/event/1YeMp
- https://kccnceu2024.sched.com/event/1YeMh
- https://www.bretfisher.com/kubernetes-vs-docker/
- https://github.com/yeahdongcn/kcd-shanghai-2024

Thanks.