THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
EUROPE

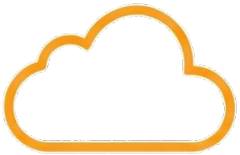
Embedded Linux Conference

Look MA, No GPUs! Experiment and Develop LLMs Without GPU Support

Alexon Oliveira
Senior Principal Technical Account Manager
Red Hat

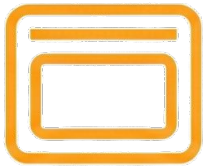
THE LINUX FOUNDATION

The Inference Dilemma: Why Serving LLMs is Hard



Massive Resources

Large models require gigabytes of VRAM. High-end GPUs (A100s/H100s) are scarce and expensive.



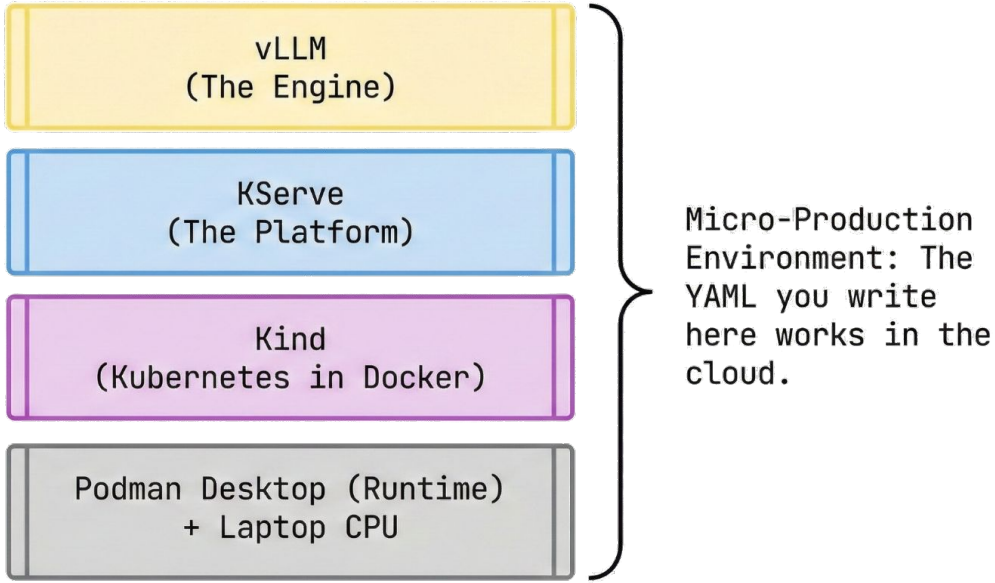
Complex State

Unlike stateless microservices, LLMs manage ‘KV Cache’ for every user session, making load balancing difficult.

The Scaling Trap

Traffic spikes require furious scaling without dropping requests, but models take minutes to load into memory.

Meet Your Local Cloud Stack

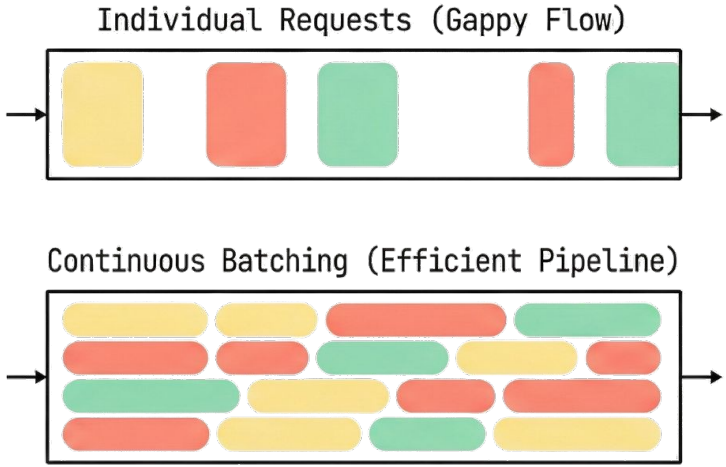


The Engine: Why vLLM?

PagedAttention: Treats GPU/CPU memory like OS virtual memory. Eliminates fragmentation.

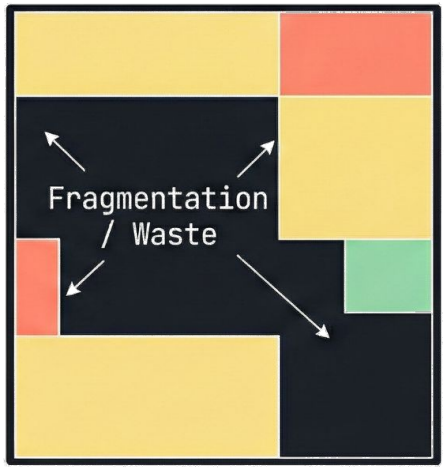
Continuous Batching: Processes incoming requests immediately rather than waiting for batch windows.

Performance: Up to 24x higher throughput than standard Transformers.



Under the Hood: The Power of PagedAttention

Legacy Memory



PagedAttention

Splits KV Cache into blocks stored anywhere in memory.
Result: Near-zero waste and higher concurrency.

The Platform: Why KServe?

Standardized Protocol

Implements V2 Inference Protocol (OpenAI compatible API).

Serverless Capabilities

Native Scale-to-Zero and Autoscaling based on traffic.

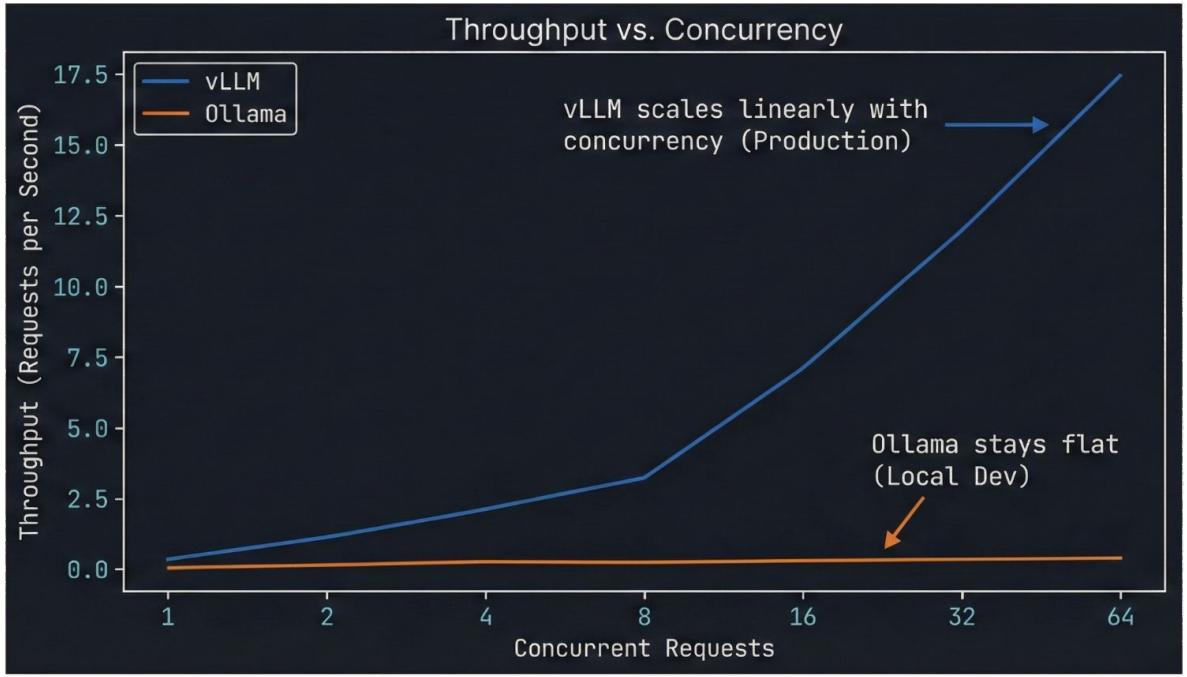
Pluggable Runtimes

Switch backends (Triton, TGI, vLLM) without rewriting infrastructure code.

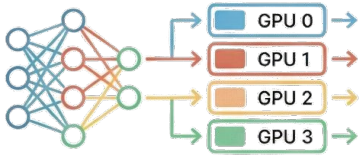
Abstraction

Manages networking, Ingress, Istio, and revision rollouts automatically.

Benchmarks: vLLM vs. Ollama

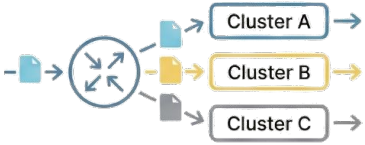


Extending to Production: Advanced Scaling



Tensor Parallelism (TP)

Split massive models (e.g., Llama-3-70B) across multiple GPUs.



Distributed Inference (llm-d)

Intelligent request routing across clusters.

Speculative Decoding

Use small draft models to predict tokens for big models.

Hardware Agnostic

NVIDIA, AMD, Intel Gaudi, and AWS Inferentia.

The Enterprise Context: Red Hat OpenShift AI

The 'Lab Stack' you just built is the core engine of this enterprise platform.

The Lab Setup: CPU-Only Inference

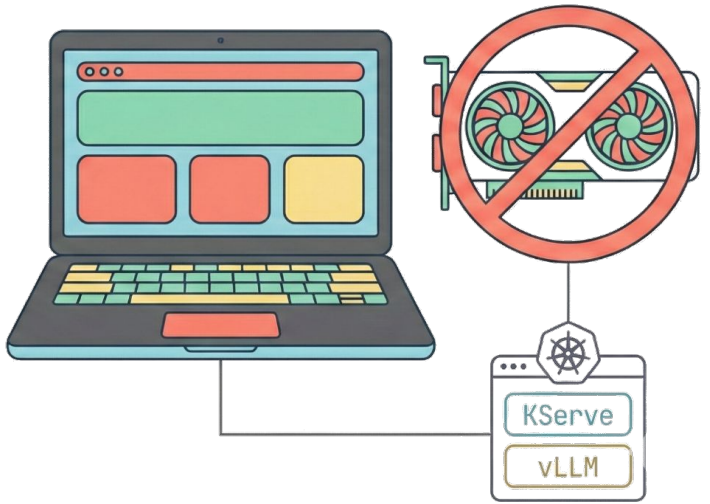
No A100? No Problem.

Constraint: Standard Laptop CPU

Model: google/gemma-3-270m

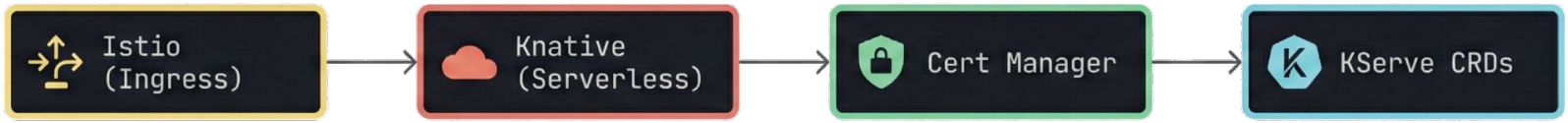
Goal: Validate KServe + vLLM pipeline

Prerequisites: Podman Desktop, Kind Cluster, Hugging Face Token



Step 1: Installing the KServe Stack

```
curl -s 'https://raw.githubusercontent.com/kserve/kserve/release-0.15/hack
  /quick_install.sh' | bash
```



Step 2: The InferenceService (The Critical Config)

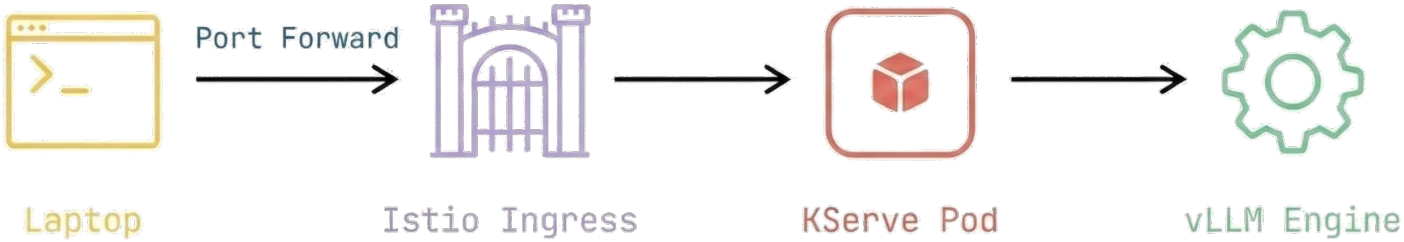
```yaml
apiVersion: serving.kserve.io/v1beta1
kind: InferenceService
metadata:
  name: gemma-3-270m
spec:
  predictor:
    image: kserve/huggingfaceserver:latest
    args:
      - --model_name=gemma-3-270m
    env:
      - name: VLLM_ATTENTION_BACKEND
        value: 'CPU_ATTN'
    resources:
      limits:
        cpu: '4'
        memory: '8Gi'
```

CRITICAL: Unlocks local testing without GPU

Sets bounds for laptop hardware

Uses standard HuggingFace Server runtime

Step 3: Connecting the Pipes



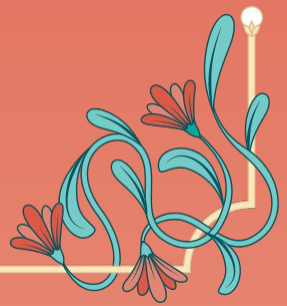

```
# Check Port Forward
kubectl port-forward svc/istio-ingressgateway ...
# Identify Host
export SERVICE_HOSTNAME=$(kubectl get inferenceservice ...)
```

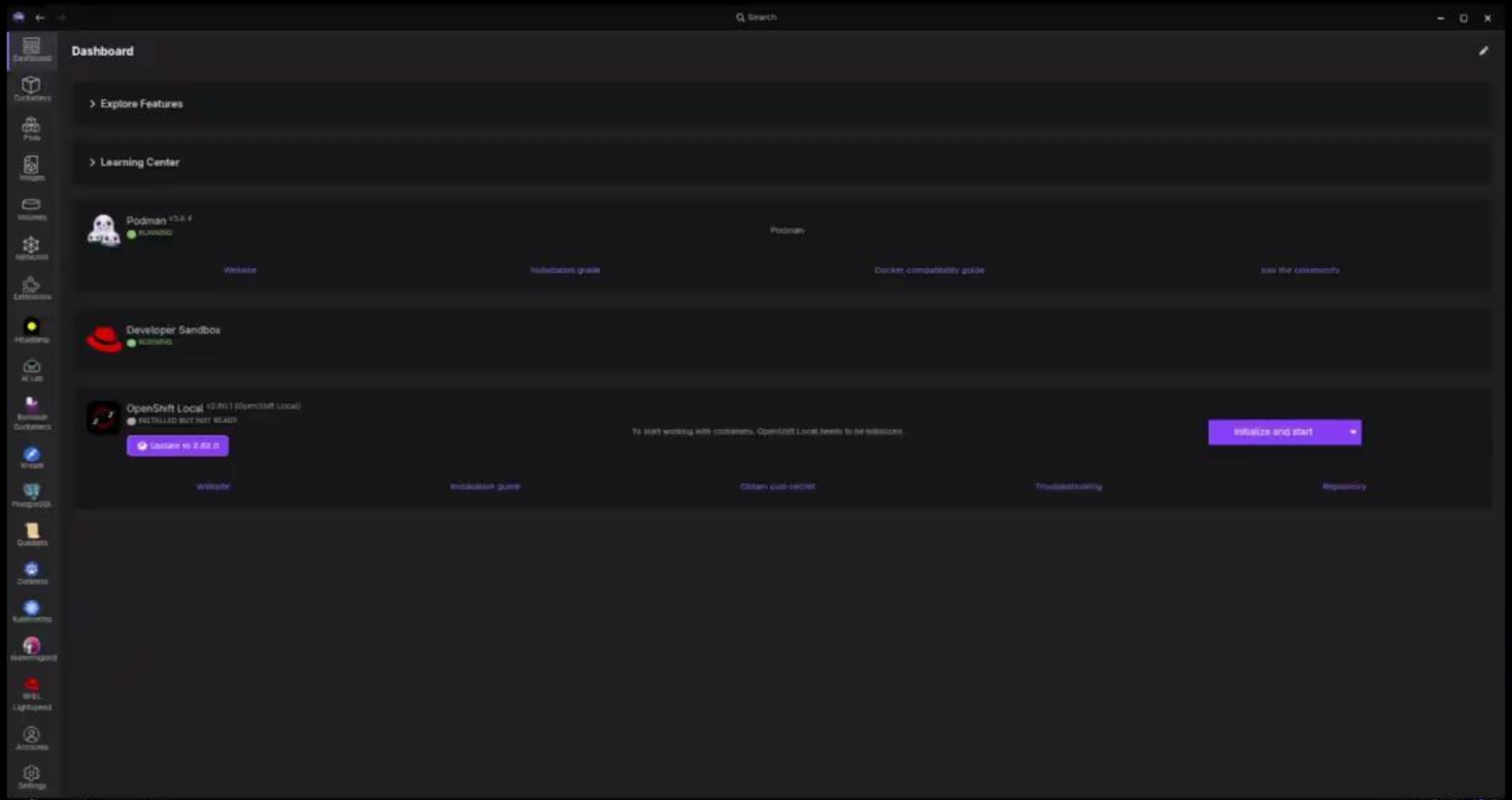
Running Inference (Live Results)

```
$ ./03-run-inference.sh
Request: 'What is the capital of France?'
{
  "id": "cmpl-8912...",
  "object": "text_completion",
  "choices": [
    {
      "text": "The capital of France is Paris.",
      "index": 0,
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 7,
    "completion_tokens": 8,
    "total_tokens": 15
  }
}
```

Success: Kubernetes-native AI inference running on CPU.

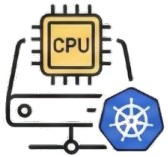
Demo



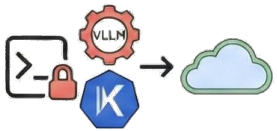

Search
Dashboard
Explore Features
Learning Center
Podman
Podman
Developer Sandbox
OpenShift Local
Initialize and start
No context
alolivei@alolivei-thinkpadp1g...
Podman Desktop

Key Takeaways & Next Steps

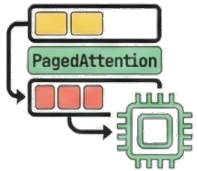
1. **Start Local:** Use 'CPU_ATTN' to learn K8s patterns without cloud bills.



2. **Use Production Tools:** vLLM + KServe locally = painless migration.



3. **Optimize:** Understand PagedAttention to maximize hardware.



Clone the Repo
(Scripts 01-03)

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT

EMBEDDED LINUX CONFERENCE

EUROPE