Livepeer Network-as-a-Product

Context

Based on this vision, this doc outlines a Product and Engineering Roadmap of how we get there one milestone at a time.

In the vision, there are three core deliverables identified to ensure the success:

1. Ability for users to deploy realtime AI workflows and request inference.
2. Industry-leading latency for realtime AI and world model workflows.
3. Cost-effective scalability, allowing users to pay as they go and the network to automatically deliver required scale.

Different from traditional infra providers for the same space (media inferences), the Livepeer is uniquely identified with following values:

1. **Open access and permission-less nature:** As a protocol on a public blockchain, anyone can build on the network, reducing platform risk.
2. **Open source and community oriented:** The project is open source, allowing users to control, update, and contribute, and benefit from network improvements as token owners.

Eric Tang 1

Another val
on real time
reflected in
SLA, and w
mechanism
deployment

What is it: LP Network-as-a-Product?

What is it?

LP Network-as-a-Product is a **service-oriented product** that consists of three key components:

#1. At its core, it is the Permission-less Livepeer Protocol enabling Orchestrators to provide their GPU capacity as a service on the network.

> *as an orchestrator, I am able to enroll my GPUs to the Livepeer Network by knowing what SLAs it must comply with, what demands this GPU is to satisfy, and what compensation it pays.*

#2. A set of public monitor-able real-time video inference SLAs that define what the network offers.

> *as users of the network, I am ensured by all my inference services requests will be met with a pre-agreed, and published SLAs.*

#3. A workload management utility to manage the lifecycle of various real-world model workloads.

> *as users of the network, I have a set of utilities allowing me to manage my workloads, from deployment, execution, analyzing to workload sunset.*

Who is it for?

LP Network-as-a-Product serves 3 user groups:

#1. **Public Orchestrators , who** provide essential GPU computing capacity to the network by meeting SLAs

#2 and Gateway Providers, who provide permission-less O management utility, and the workload management utility.

#3. **Inference service and tooling providers** (such as Daydream), who deploy workloads to the network and manage services for their users, typically through a self-hosted Gateway, and serviceable APIs.

Roadmap

Overview

There are four major product engineering milestones to deliver what the vision describes, as following:

- **Milestone 1** is the Network-as-a-Product (NaaP) MVP, the goal is to make NaaP key characteristics measurable, and monitor-able, in realtime.
- Milestone 2 is to enable NaaP, to be self-adaptive, and scalable, based on the core set of SLAs.
- Milestone 3 is to provide a toolkit, to enable community manage the NaaP, through a well defined gateway API spec. Community can implement their own version of the API spec, to provide their own utility.

> All above 3 milestones had one implicit workload topology assumption, it is 1 or many to 1 mapping between stream and GPUs. it is stream:GPU = n:1

- Milestone 4 to extend the paradigm from n:1 (1, or more stream per GPU), to 1:m (1 stream to many GPUs), where GPU cluster can provide more sophisticated workload management for redundancy and in-workload scalability

The following outlines the major goal for each milestone

Milestone 1: NaaP MVP


Doug Petka

Generally I
supporting
for a job. H


Qiang Han

The goal is to deliver SLAs metrics reporting using the existing analytic infra architecture, as (GPU + Trickle Protocol + Livepeer Gateway + Kafka cloud + Clickhouse cloud), through a hosted gateway driven test runner, and reporting API which is publicly access-able.

As an Orchestrator, from the explorer SLA dashboard, I can have the full visibility on

- what GPU SLAs criteria to meet for providing competitive services
- what GPUs realtime SLAs is in realtime, per GPU
- what network demands are throughout the time, to inform the demand vs. supply gaps

Milestone 1.0: Design Spec RFC

RFC for the technical document on SLAs, and its technical Architecture

Taking Daydream as design partner, to publish Phase 1 RFCs

This should include two set of metrics
- GPUs metrics on (core performance, reliability, and workload) per GPU in realtime, per workload

(treat the test loads as real loads, and maintain minimum 20% usage at all times)

Daydream can specify a set of target SLAs as part of the dashboard as

Core Performance KPIs

- target output fps / per model
- minimum GPU requirements (gpu type, vram size, per gpu)
- prompt-2-first-frame latency (≤ xx ms)
- e2e stream latency (from ingest, to egress)
- jitter coefficient (standard dev fps/mean fpg) per workflow
- startup (cold) time
- network bandwidth up/down

Reliability KPIs

- inference failure rate
- swap rate (% gateway has to swap)

Cost economics

currently, re
by swappin
through coll

Doug Petka

Is this “per
idea that an
these core

Show 1 rep

Qiang Han

since GPUs
workload th
without kno

- CUDA utilization efficiency
  - CUE (Cuda Utilization Efficiency) = Achieved FLOPS/Peak FLOPS×100%
    - Whether the GPU is **compute-bound** or **I/O/memory/network-bound**
    - How well your **kernels are optimized** for that GPU architecture
    - Whether you're getting **expected value per $** of GPU usage

-Network historical demand distribution statistics per (gateway Geo, workflow, GPU type, O)

Daydream as a gateway provider publishes the following as demand measurements

- total streams per(O, workflow, region, time range)
- total inference mins per(O, workflow, region, time range)
- total inference mins per (GPU type)
- unit price/fee per workload

as a working prototype dashboard in metabase

Timeline: 2 weeks

Working Group:

Evan, Qiang, Doug

Rick, One representative from Cloud SPE, One representative from network advisory board

Need work with Evan, Dough to complete

Network-as-a-Product MVP – SLA Metrics & Analytics Infra

Review by:

Community, and Inc engineering

in 2 weeks, concluded by end of Nov.

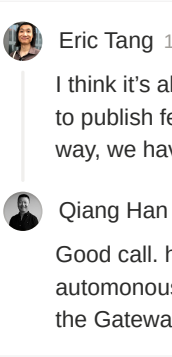
Milestone 1.1 : Ref Implementation

Goal: Launch Explore AI SLA monitoring dashboard with detailed realtime Orchestrator's SLA reporting with dims of (per O, per region, per workload, per period and now)

Recommendation: Evan and Cloud SPE

Timeline: 4 weeks (many works have been done)

Requires a detailed scoping exe (can be led by Qiang)



Implementation team main tasks:

- Explorer O SLAs Reporting Dashboard (one option is to directly embed metabase dashboard)
- Gateway load test scheduler, GPU Metric gateway reporting (leveraging current daydream load testing toolkit)
  - a dedicated network testing gateway, to ensure SLA
- Public Git Repo for Load test datasets, as
  - good prompt

    workload specific dataset that can have consistent workflow outputs from a given GPU spec
  - random prompt

    workload tries to simulate real world traffic types, with variants of prompts (bad, edge, and average)
- SLA data pipeline, Analytic Model and O reporting
- API for Explorer SLA dashboard

Milestone 1 Kick-off meeting

Milestone 2: SLAs Gateway

(Scoring, Selection and Incentive)

The goal is to have a well defined Gateway API Spec and reference implementation, on

- SLA Score Computing
- Selection API
- SLA based payment API

Milestone 2.1 : Design Spec RFC

RFC, for SLAs based O scoring and selection algorithms, and incentive algorithms that take into consideration of workflow varieties, and its underlying GPUs spec, including

(A user story doc is missing to describe what this API needs to accomplish for whom), roughly what should include as following:

- how to calc SLA scoring based on Milestone 1 deliverables

> What is considered “fair”, by establishing the SLA weighted scoring algorithm

- what is selection/swap algorithm in realtime per workload
  - how to dynamically publish the benchmark, as workflow publishers must specify this (for example, daydream)

- what is the dynamic incentive algorithms, to enable more diversified workload participation
  - since different workloads have different demands, and pricing for end users, how to distribute that revenue, through the protocol?
- a published API spec, including
  - Query API, to retrieve any Public O SLA scoring in realtime/historic records
  - SLAs initialization API to get a O started with initial fair SLA Score
  - Min Qualifying SLA score Query API and update API (allow the minimum SLA score to be queried, updated, by Gateway)

Working Group

Dough, Evan, Qiang, Rick, 2 representatives from communities

Milestone 2.2: Ref Implementation

Goal: Launch Explorer AI Orchestrator Lead Board UI

Implementation tasks:

Livepeer Gateway SLA API Reference implementation on

- SLAs scoring
  - score update interval
  - score publishing and repo
  - score per workload (using different target for different workload)
    - for example krea model will have much lower fps, vs. streamdiff
- Selection algorithm
  - query for available GPUs using (workload, SLAs, cost)
- Selection statistics reporting based on SLAs
  - realtime score balanced with historic scores
- Payment (SLAs based incentive framework)
  - reflected by % selection, and swapping, further manifested as total inference mins
  - fee per workfload

?The main assumption is:

- different GPU has different cost base
- different workload has different GPU and power consumption
- different providers offer different levels of SLAs

How to fairly compensate the differences above, will be the main design question to consider

SLAs based incentive will bring “Protocol Changes” on how different stakeholders including Gateway provider, Orchestrators, Delegators can collaborate together, so the best economics of demand and QoS driven scalability is possible, with a self-perpetrated dynamic market mechanism.

Milestone 3 Workload Utility Control Plane

The goal is have a well defined Gateway API spec on workload management. This API will enable more workload utility to be developed. The Control Plane is the concept to integrate the workload management API in ops workflow, with an opinionated DevX Ux design.

The following core use cases are supported:

User Story #1

As a workload creator, I need to create a descriptor file (in json format) to describe the following:

- workload id
- workload model card
- where to download the AI models from and access key
- SLAs target values
- min cuda
- min vram
- supported GPUs
- min # of instances

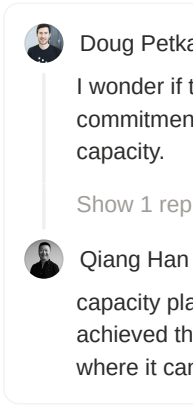
and, I can

- query the network, to discover if any available resources, and how many, and at what price
- publish my workload request to a registry for inquiry O visibility, until is fulfilled

as the network, users’ workload resource requests are published as a demand request, to produce the future demand projection. This projection together with historical demand statistics, will help Orchestrators to respond to the growth in realtime.

User Story #2

As a user, once the GPU resources are allocated, I can



- deploy the workload to the network
- start the inference services
- publish the inference services, with a gated endpoint url
- stop the inference services
- release/uninstall the workload from the allocated GPUs
- perform the SLA monitoring of the published inference services

User Story #3

As a user, I can manage the workload security by

- must perform all the above workload management tasks through an authorized api key
- i can set up api key gate for the inference end point

User Story #4

As orchestrator, I can respond to the workload request descriptor, to provide my GPUs running the workload, by deploying the descriptor. also update the demand requests, with my capacity.

Doug Petka

The below i
unsolved pr
protocol up
but my hea
idea that:
…

Milestone 3.0 Design Spec RFC

The goal is define the API spec to fulfil the three core user stories.

Team:
Qiang, Dough, Rick, 2 more members from community

first task: collect core user stories to define the scope boundary of the Spec.

Milestone 3.1 Reference Implementation

The goal is to provide first reference implementation for the core utility scope. So a user gateway dashboard can be delivered to enable gateway provider manage user-centric resource management, so called, gateway/network control plane.

Team:
need SPE, to be funded, to provide the essential talents, and resources implement this, step by step.

Milestone 4 Complex Workload handling

The goal is to enable the NaaP users, to publish, and manage more complex workloads, with higher SLAs (especially over 99.9% reliability for its core inference services). this requires a cluster based architecture to enable more robust resource orchestration during the runtime, and scalability.

This does require Public O to offer their GPUs pools in the form of cluster, not single GPU.

Let’s worry about this, until we have enterprise level use cases.