

Akash Network

Running Containers in a Distributed System



The views, thoughts, and opinions expressed in this presentation are solely those of the individual author and do not necessarily reflect the official policy or position of PennyMac

Nathan Moore
VP, App Dev



Usual Disclaimer

The views, thoughts, and opinions expressed in this presentation are solely those of the individual author and do not necessarily reflect the official policy or position of PennyMac

This presentation was made with assistance from Google's Nano Banana

Nathan Moore
VP, App Dev
PennyMac



The name “Akash” comes from Sanskrit:

ākāśa (आकाश)

“the basis and essence of all things in the material world; the first element created.”

-Wiki

Meaning of Akash

<https://en.wikipedia.org/wiki/Akasha>

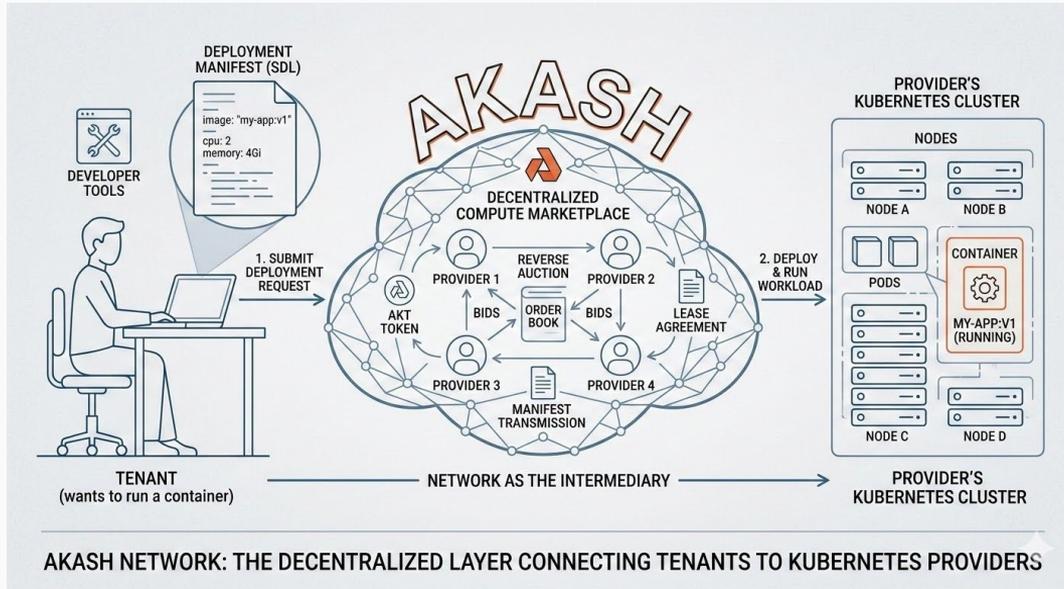


Akash is an open source project, licensed under Apache License 2.0

There is also a governance structure provided by a decentralized community, with decisions made by AKT token holders

The original creator was Overclock Labs, which remains the largest contributor to the project

Akash Network is the middle layer between people who want to run a container and people who want to host a container



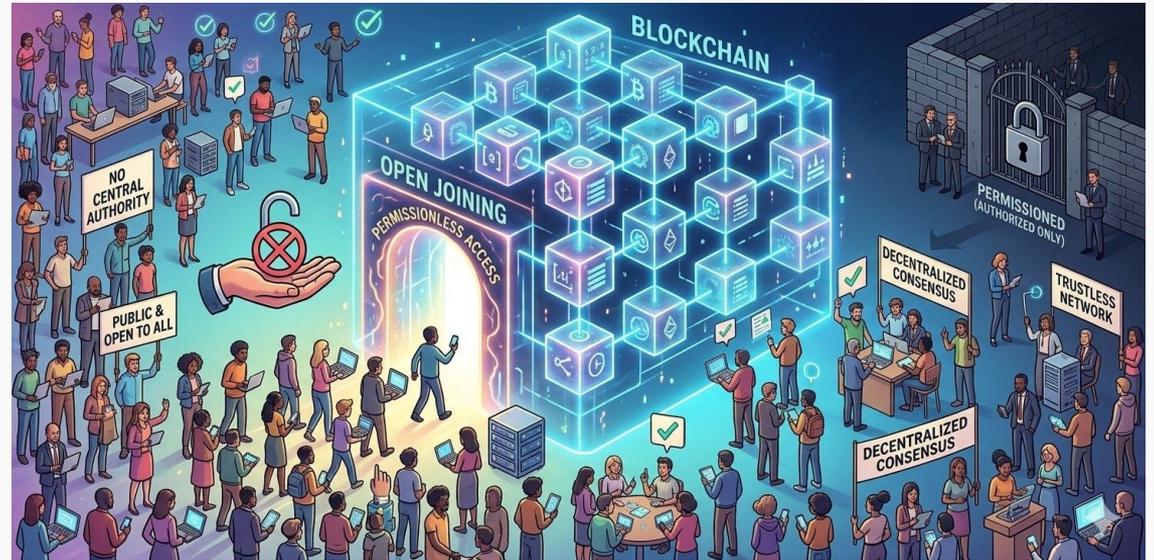
Akash Network is the middle layer between people who want to run a container and people who want to host a container

Anyone can participate to provide compute resources

Anyone can participate to consume compute resources

This requires “permissionless” access on both sides

It also requires “trustlessness” with strong encryption for secure compute, where compute jobs cannot be corrupted



The mission is to democratize both usage and provision of compute resources

- Application Containers, \$3.5Bn Revenue in 2025
- 27% CAGR growth, expect \$39.1Bn Revenue in 2035
- Cloud vs. On-Prem hosting
 - 75.5% vs. 24.5%

Global Application Container Marketing Notes

<https://www.grandviewresearch.com/industry-analysis/cloud-computing-industry>

<https://www.researchnester.com/reports/application-container-market/8266#:~:text=Application%20Container%20Market%20Outlook:.evaluated%20at%20USD%204.5%20billion>



- Founded in 2015
 - Greg Osuri and Adam Bozanich
- Mainnet launch in 2020
- Utilizes Cosmos SDK
- GPU support in 2023
- Active Discord:
 - <https://discord.com/invite/akash>
 - About 24k people in Discord

Akash Network History



Network Summary

USD spent (24h) ?

\$2.17K ↓ 11.27%

Graph ↗

Total spent USD ?

\$4.95M ↑ 0.04%

Graph ↗

New leases (24h) ?

240 ↓ 28.36%

Graph ↗

Total leases ?

428K ↑ 0.06%

Graph ↗

Resources leased

Active leases ?

516 ↓ 7.86%

Graph ↗

Compute

1.94K CPU ↓ 25.65%

Graph ↗

Graphics

70 GPU ↑ 22.81%

Graph ↗

Memory

4.83 TB ↑ 5.35%

Graph ↗

Storage

20.33 TB ↑ 1.07%

Graph ↗

Network Capacity

Active providers ?

53 ↓ 1.85%

Graph ↗

Compute

8.2K CPU ↓ 4.65%

Graph ↗

Graphics

182 GPU ↓ 1.09%

Graph ↗

Memory

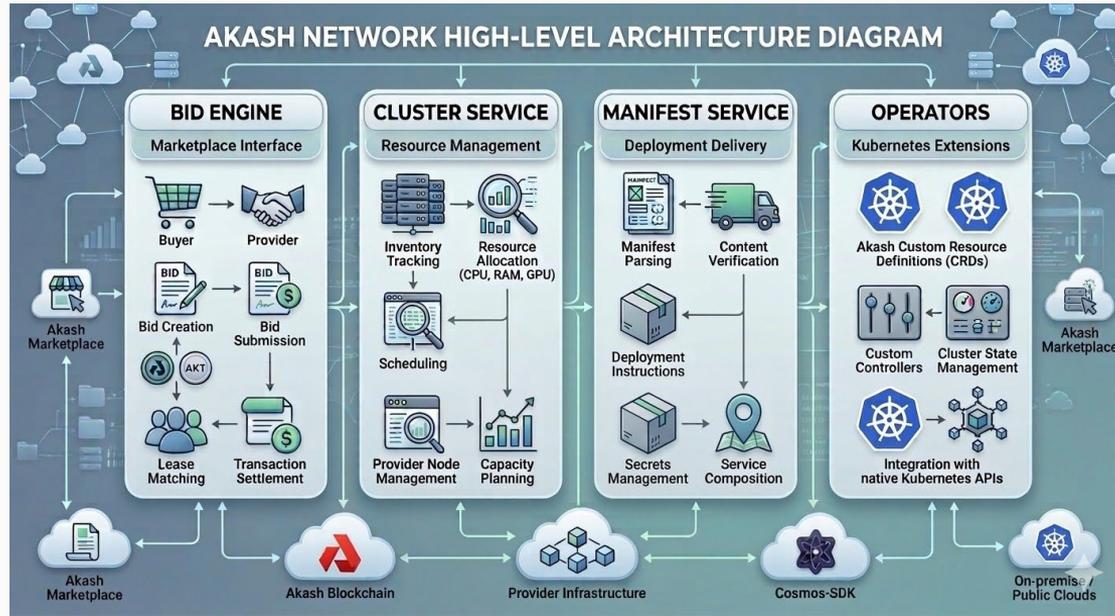
52.95 TB ↓ 1.6%

Graph ↗

Storage

514.65 TB ↓ 5.74%

Graph ↗



Bid Engine

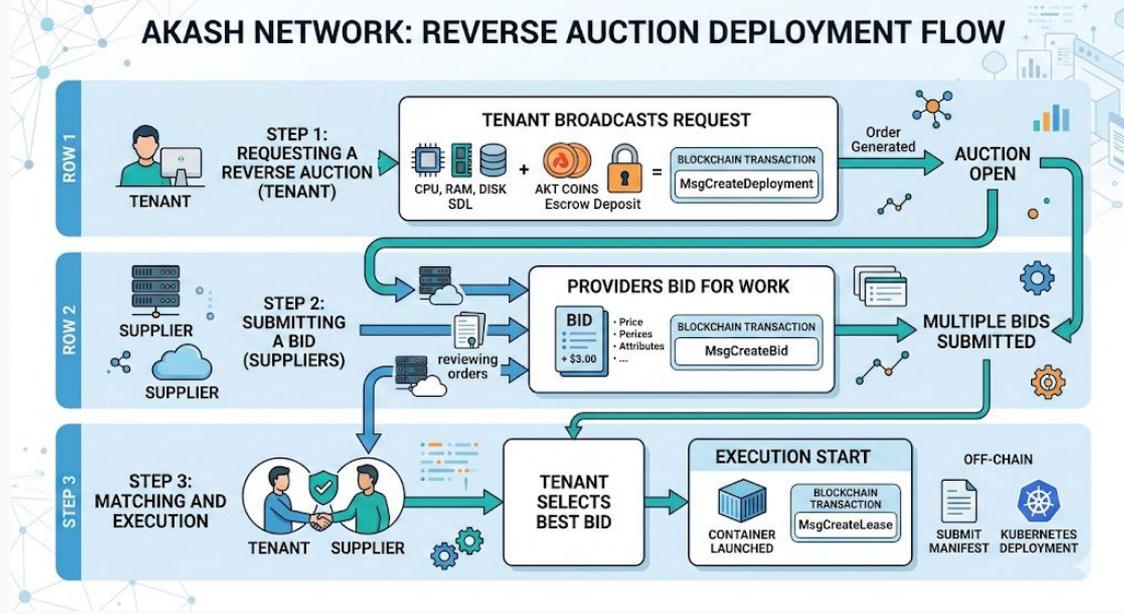
Reverse Auction

An auction where multiple providers all submit bids for how much they are want to complete the service, while a single buyer chooses which bid to accept



How It Works: Reverse Auction

AKASH NETWORK: REVERSE AUCTION DEPLOYMENT FLOW



When someone (tenant) wants to have their container run, on request, all available providers who want to bid on running this container submit a bid, and the buyer chooses which one, after which container execution starts

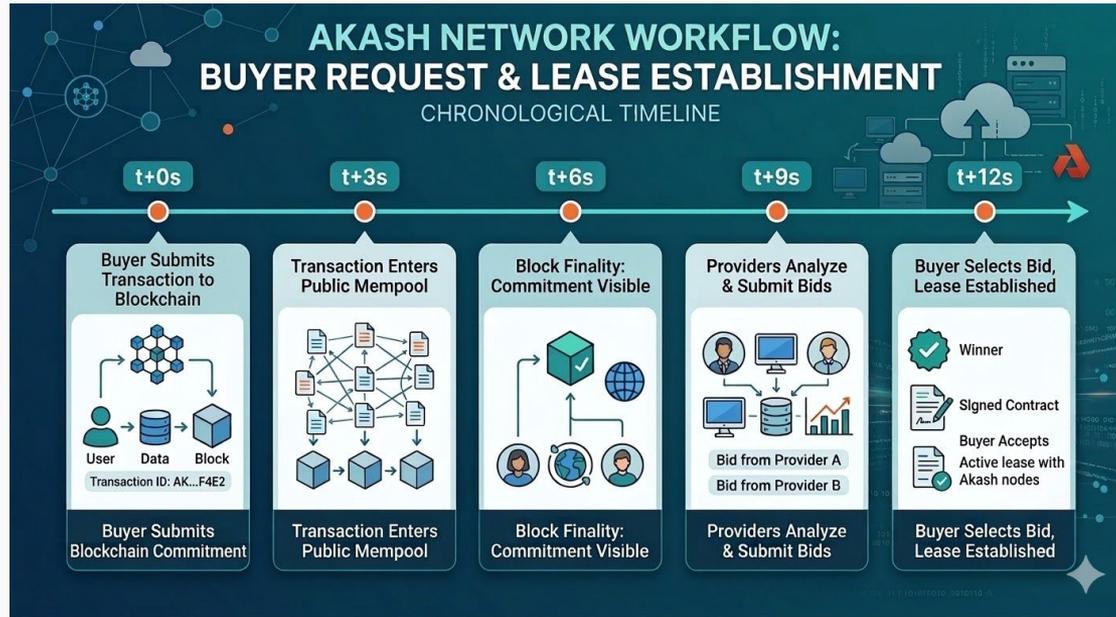
Blockchain semantics

Every participant has a unique wallet address, and can sign transactions to cryptographically validate them.

1. MsgCreateDeployment (Tenant commits transaction)
2. MsgCreateBid (Suppliers commit transactions)
3. MsgCreateLease (Tenant signs transaction of chosen supplier)

Buyer (Tenant) and Supplier (Provider)

- Cosmos SDK
 - 6 second blocktime
 - Gossip protocol
 - Consensus protocol
 - Tendermint
 - Byzantine Fault Tolerance
 - Instant finality
 - Providers can run a Validator node
 - Proof-of-Stake



Timeline from Lease Request to Lease Establishment



- Public Ledger
 - Every bid, pricepoint, and hardware promise is permanently recorded and readable by all
- No gatekeepers
 - Anyone can deploy or provide resources globally

This is part of “Permissionlessness”, where providers cannot engage in price discrimination or prevent others from competing, as all competitors have full visibility into the market clearing price

Transparency and Fairness

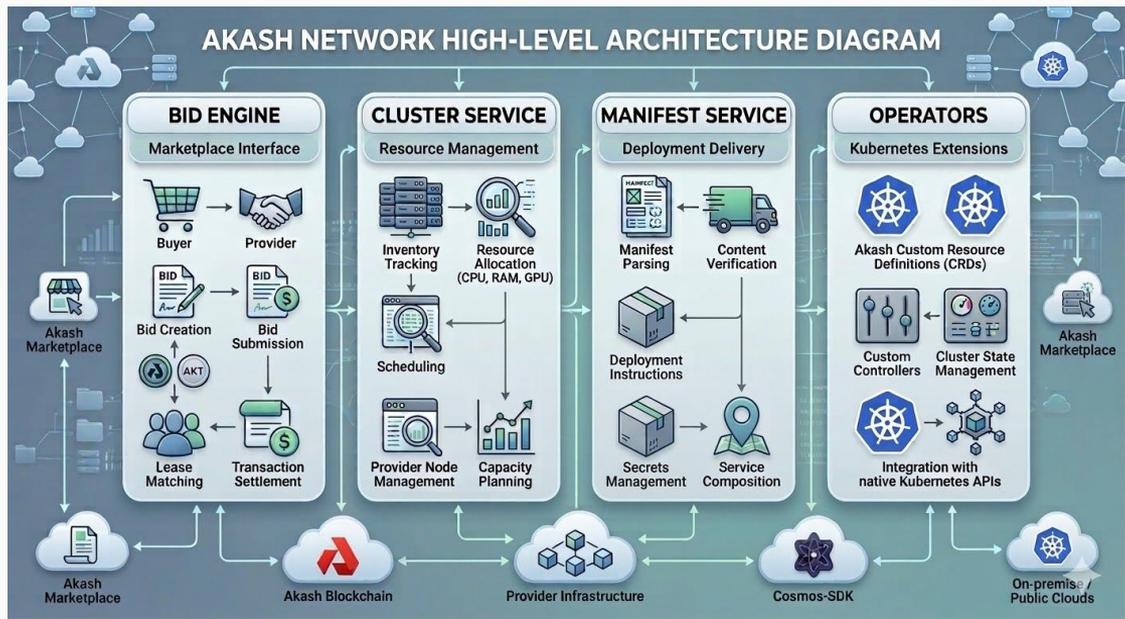


- To pay for compute, rent is due every block cycle (6 seconds)
 - This is usually a fraction of a penny (~\$0.0004)

Settlement occurs on every block instead of a monthly billing cycle. Termination of lease requires 1 block to elapse, ceasing billing immediately after

This delivers a flexible billing, pay-as-you-use model, which either side can cancel at any time

Rent is due



Cluster Service



Provider-Side cluster management

- Choose your hardware setup
 - Minimum requirements
 - Availability
 - Resource Allocation
 - Where/how you want to host
 - Network

Cluster Service



Minimum requirements:

Single Server (Worker & Control Plane Combined)

- x86_64 only
- CPU 8 cores
- RAM 16GB
- Storage 150GB SSD
- Internet connectivity
- Dynamic DNS or static public IP

Cluster Service

<https://akash.network/docs/providers/getting-started/hardware-requirements/>



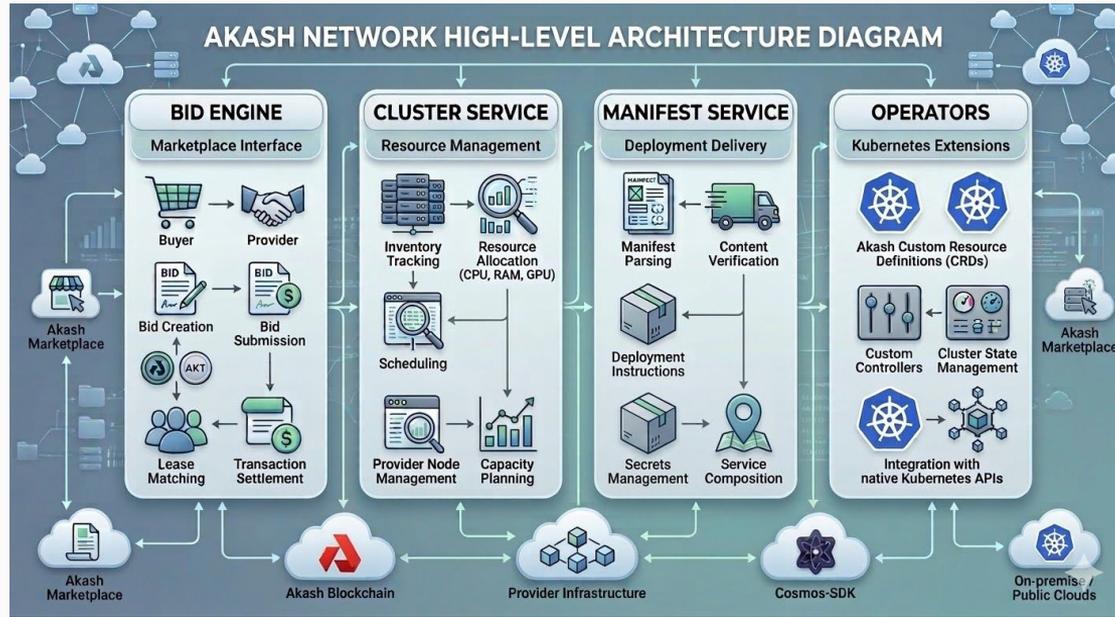
Deploy your cluster via Ansible Playbook, Kubespray, or Provider Console

Requirements

- Ubuntu 24.04 LTS
- Root or sudo access on all nodes
- SSH access to all nodes

Cluster Service

<https://akash.network/docs/providers/setup-and-installation/provider-playbook/>



Manifest Service



Stack Definition Language (SDL)

Akash's custom YAML-based declarative language used to define:

- Configuration
- Resource requirements
- Deployment parameters

Configures a “Manifest” file, similar to a Docker Compose file

Manifest Service

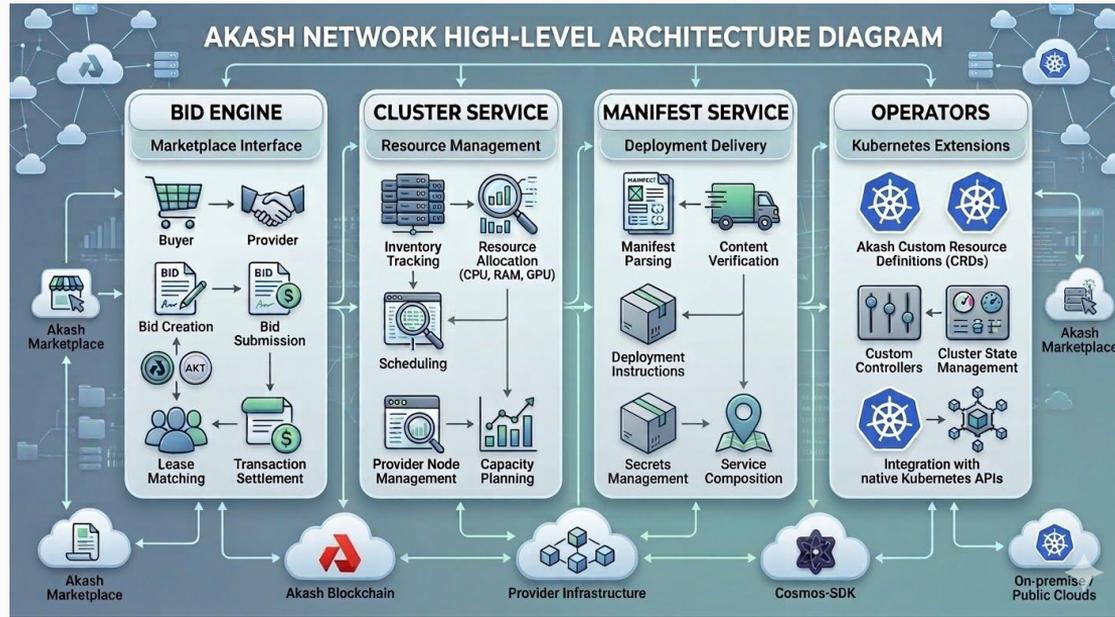
<https://akash.network/docs/providers/setup-and-installation/provider-playbook/>

SDL example, OpenClaw

```
README View SDL
< OpenClaw Deploy
1 ---
2 version: "2.0"
3 services:
4   openclaw:
5     image: ghcr.io/zjuuu/openclaw-docker:openclaw-v2026.2.12@sha256:54078e1fc41c781a94f48846cf286f18d0632a4e49c69d78aee799ba938fffa0
6     env:
7       # REQUIRED: Set a strong password for the setup UI
8       - SETUP_PASSWORD=CHANGE_ME_TO_STRONG_PASSWORD
9       - OPENCLAW_CONFIG_PATH=/data/.openclaw/openclaw.json
10    expose:
11      - port: 8080
12        as: 80
13        to:
14          - global: true
15        http_options:
16          max_body_size: 104857600
---
```

Manifest Service

<https://console.akash.network/templates/akash-network-awesome-akash-openclaw>



Operators

Inventory Operator

- Discovers and tracks cluster resources (CPU, GPU, memory, storage) in real time
 - Supports hardware feature discovery (GPU models, storage classes)
 - Integrates with Nvidia device plugin for GPUs
 - Updates K8s node labels with capabilities
 - Publishes inventory to provider service for bidding

Operators

IP Operator

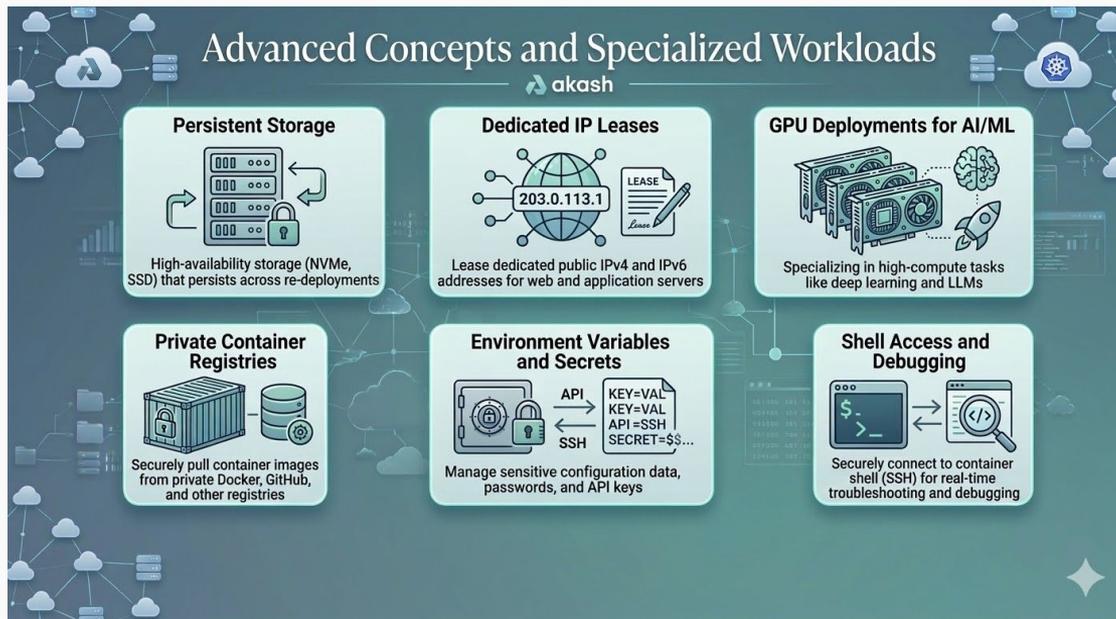
- Manages static IP address allocation for deployments via MetalLB integration
 - Allocates dedicated IPs for deployments
 - Supports IP sharing across services (same tenant)
 - Tracks IP pool usage and availability
 - Handles IP lifecycle (create, update, delete)
 - Exposes HTTP API for monitoring

Operators

Hostname Operator

- Manages custom hostname assignments and creates ingress resources for deployments
 - Integrates with Ingress controllers
 - Handles hostname conflicts and validation
 - Supports wildcard hostnames
 - Monitors hostname health

Operators



Advanced Concepts and Specialized Workloads

Default storage is ephemeral

- Lost on container restart

If provider supports it, can deploy to persistent storage instead.

Note, persistent storage is provider-local. Switching providers loses data

There are three storage classes:

1. NVMe storage ("Beta3")
2. SSD storage ("Beta2")
3. HDD storage ("Beta1")

Persistent Storage

Default deploys use shared IPs, mapped via provided Hostname

Custom IP leases allow

- Custom domains
- Static IPs
- non-HTTP services

IP leases are available on a per-Provider basis, at additional cost.

IP Leases



GPUs are available on a per-provider basis

Tenants can request specific GPU models including multiple GPUs, and match against what providers have available on lease instantiation

Providers offer H200 on down, at different price points

Supports CUDA runtime

GPU Deployments for AI/ML

<https://akash.network/docs/learn/core-concepts/gpu-deployments/>

Specify private container registry credentials inside of SDL

Any Docker-compatible private registry is supported, including

- ECR
- GCR
- GHCR
- etc

Private Container Registries

Environmental variables are set within SDL

Use Secret Management Tools

- AWS Secrets Manager
- HashiCorp Vault
- Google Secret Manager
- Azure Key Vault

Have application fetch at runtime, use SDL to specify which tool

Environmental Variables and Secrets



Prerequisites:

- Active deployment with a lease
- Akash CLI (provider-services) installed
- Wallet public key accessible
- Deployment Sequence Number (DSEQ)
- Provider address

Use the “provider-services” package (the Akash CLI) to authenticate and access the remote container

Shell Access and Debugging

<https://akash.network/docs/learn/core-concepts/shell-access/>

Akash Deployment Guide with No Crypto Wallet

 akash

Use Free Credits to Start

Sign Up & Get Credits



Sign Up & Get Credits

Create a free Akash account and automatically receive 100 free Akash testnet AKT credits. No credit card or wallet required.

 akash



Configure Your Deployment

Upload your deployment file (SDL) and configure resources (CPU, RAM, GPU) using our intuitive web console.

Deploy Your Application



Deploy Your Application

Review bids from decentralized providers and select the best one to instantly deploy your app to the cloud.



Akash Public Cloud



Provider Infrastructure



Your App URL



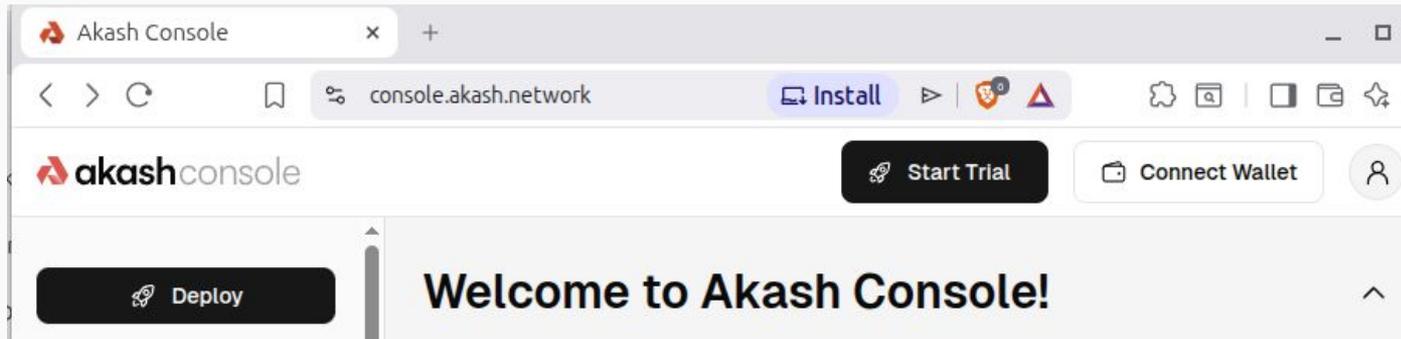
Deployment Guide – No Crypto Wallet

<https://akash.network/blog/introducing-credit-card-payments-in-akash-console/>



<https://console.akash.network>

Select "Start Trial"



Deployment Guide - No Crypto Wallet

<https://akash.network/docs/learn/core-concepts/shell-access/>



<https://console.akash.network>

Follow instructions

A screenshot of the Akash console's 'Start Your Free Trial' page. At the top, there is a progress bar with five steps: 1. Free Trial (highlighted), 2. Create Account, 3. Verify Email, 4. Payment Method, and 5. Welcome. Below the progress bar, the Akash console logo is displayed. The main heading is 'Start Your Free Trial'. The text below reads: 'Experience the power of decentralized cloud computing with Akash Network. Deploy your applications on a global network of providers with our free trial.' At the bottom, there is a dark button labeled 'Start Free Trial'.

Deployment Guide – No Crypto Wallet

<https://akash.network/docs/learn/core-concepts/shell-access/>



 Deploy

-  Home
-  Deployments
-  Templates
-  SDL Builder
-  Providers
-  Alerts

Welcome to Akash Console!



Getting started with Akash Console

Learn how to deploy your first docker container on Akash in a few clicks using Console.



Explore the marketplace

Browse through the marketplace of pre-made solutions with categories like blogs, blockchain nodes and more!



Learn more about Akash

Want to know about the advantages of using a decentralized cloud compute marketplace?

Your account

+ Add Funds

 Deploy

Available Balance 

\$100.00

\$0.00 used in deployments

Active Deployments 

0

Total Cost 

\$0.00 / hour

\$0.00 / month

Deployment Guide - No Crypto Wallet

<https://akash.network/docs/learn/core-concepts/shell-access/>

Deploy OpenClaw

Confirm deployment creation?

openclaw:ghcr.io/zjuuu/openclaw-docker:openclaw-v2026.2.12@sha256:54078e1fc41c781a94f48846cf286f18d0632a4e49c69d78aee799ba938fffa0

3 CPU 6 Gi RAM 5 Gi Disk

To create a deployment, you need to have at least \$5 in an escrow account. [Learn more.](#)

Trial

Amount Balance: 100 USD

USD 5

Cancel

Buy credits

Continue

Search provider

Search provider...

Accept Bid →

Favorites Audited 

Time Remaining: 04:44 ⓘ

Waiting for more bids... ⓘ

GSEQ: 1  3 CPU  6.44 GB  16.11 GB

1 of 1

Price	Region	Uptime (7d)	Provider	Audited	Select
\$6.77 / month ⓘ	21, HR	99.99%	 provider.europlots.com 	Yes 	<input type="radio"/>
\$6.97 / month ⓘ	TX, US	98.14%	 provider.akashprovid.com 	Yes 	<input type="radio"/>
\$7.30 / month ⓘ	MI, US	98.08%	 provider.boogle.cloud 	Yes 	<input type="radio"/>

Deployment Guide - No Crypto Wallet

<https://akash.network/docs/learn/core-concepts/shell-access/>



Success

< **OpenClaw** ... **Add funds** Auto top-up

Balance	\$5.00	Status	active Trial
Cost	\$6.77 / month	Time left	~23 days (Trial: in 1 day)
Spent	\$0.00	DSEQ	25837102

Leases Alerts Logs Shell **Events** Update

Services
openclaw **Stick to bottom** **Download events**

```
1 [c9a4dtuhkpbaf0158f0bmhgg6g.ingress.europlots.com]: [Normal] [Sync] [Ingress] Scheduled for sync
2 [c9a4dtuhkpbaf0158f0bmhgg6g.ingress.europlots.com]: [Normal] [Sync] [Ingress] Scheduled for sync
3 [openclaw]: [Warning] [FailedScheduling] [Pod] 0/5 nodes are available: pod has unbound immediate PersistentVolumeClaims. preemption: 0/5
4 [openclaw]: [Normal] [Scheduled] [Pod] Successfully assigned pp6u0nflia2mojmetonqmjthc0toddnmjidrgd360tu8s/openclaw-0 to node1
5 [openclaw]: [Normal] [SuccessfulAttachVolume] [Pod] AttachVolume.Attach succeeded for volume "pvc-77a52d46-839c-49d5-b19f-dad644e4c00b"
6 [openclaw]: [Normal] [Pulled] [Pod] Container image "ghcr.io/zjuiu/openclaw-docker:openclaw-v2026.2.12@sha256:54078e1fc41c781a94f48846cf2"
7 [openclaw]: [Normal] [Created] [Pod] Created container openclaw
8 [openclaw]: [Normal] [Started] [Pod] Started container openclaw
9 [openclaw]: [Normal] [Provisioning] [PersistentVolumeClaim] External provisioner is provisioning volume for claim "pp6u0nflia2mojmetonqmjthc0toddnmjidrgd360tu8s/openclaw-0"
10 [openclaw]: [Normal] [ExternalProvisioning] [PersistentVolumeClaim] Waiting for a volume to be created either by the external provisioner
11 [openclaw]: [Normal] [ProvisioningSucceeded] [PersistentVolumeClaim] Successfully provisioned volume pvc-77a52d46-839c-49d5-b19f-dad644e4c00b
12 [openclaw]: [Normal] [SuccessfulCreate] [StatefulSet] create Claim openclaw-data-openclaw-0 Pod openclaw-0 in StatefulSet openclaw successful
13 [openclaw]: [Normal] [SuccessfulCreate] [StatefulSet] create Pod openclaw-0 in StatefulSet openclaw successful
```

Deployment Guide – No Crypto Wallet

<https://akash.network/docs/learn/core-concepts/shell-access/>

OpenClaw Details

< **OpenClaw** 🔄 ⋮ Add funds 🔌 Auto top-up ℹ️

Balance	\$5.00	Status	active ● Trial 🔌
Cost	\$6.77 / month	Time left	~23 days (Trial: in 1 day)
Spent	\$0.00	DSEQ	25837102 📄

Leases Alerts Logs Shell Events

active ● GSEQ: 1 OSEQ: 1 View logs

🔌 3 CPU 📄 6.44 GB 📄 16.11 GB

Price: \$6.77 / month 🔌

Provider: provider.europlots.com 📄 ☆ 🔄

Group: openclaw 🔌 ✓

Available: ● Ready Replicas: ● Total: ●

URI(s):
c9a4dtuhkpbaf0158f0bmqg6g.ingress.europlots.com 📄 📄

c9a4dtuhkpbaf0158f0bmqg6g.ingress.europlots.com/get-started/ ▶ 🔒 🚩

OpenClaw on Akash

Enter your setup password to continue

PASSWORD

Enter SETUP_PASSWORD

Login

Deployment Guide – No Crypto Wallet

<https://akash.network/docs/learn/core-concepts/shell-access/>

Using Akash with a Crypto Wallet



Connecting & Transacting on Akash

Set Up & Fund Your Wallet



Set Up & Fund Your Wallet

Step 1: Download an Akash-compatible crypto wallet (e.g., Keplr, Leap) and fund it with AKT tokens. No testnet credits here—this is real. Ensure your keys are secure.



Authorize & Configure

Step 2: Connect your wallet to the Akash Console and authorize a deployment by signing a transaction. Define your deployment file (SDL) as usual. Your wallet funds the lease.

Confirm & Transact



Confirm & Transact

Step 3: Submit your transaction, approve the lease cost with a single final signature, and watch your application take off. Payments are via AKT tokens. Your app is live!



Decentralized Akash Network



Provider Infrastructure



Your Live App URL



Using Akash with Crypto Wallet



Switch to Wallet Payments

- Work natively with AKT
- Gain greater control
- Exit free trial so deploys last longer than 24 hours

Trial | \$95.00

Free Trial

Credits Remaining:	\$95.00
Deposits:	\$5.00

What's this?

Once your Free credits run out, deployments will automatically close. To continue, create an account and add funds with your credit card. Deployments from your Free Trial get transferred when creating a new account.

[Add Funds](#)

[Switch to Wallet Payments](#)

Select your wallet

Keplr

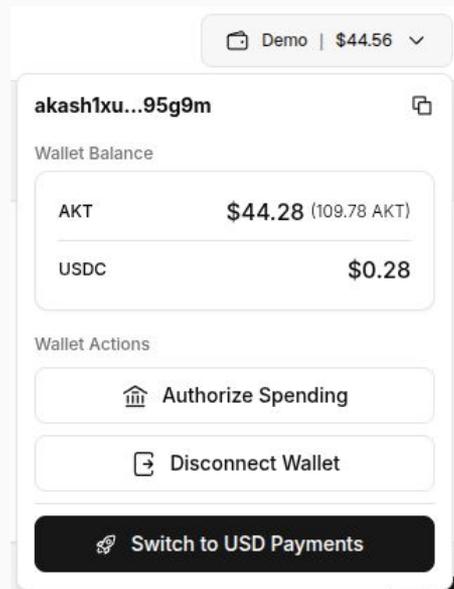
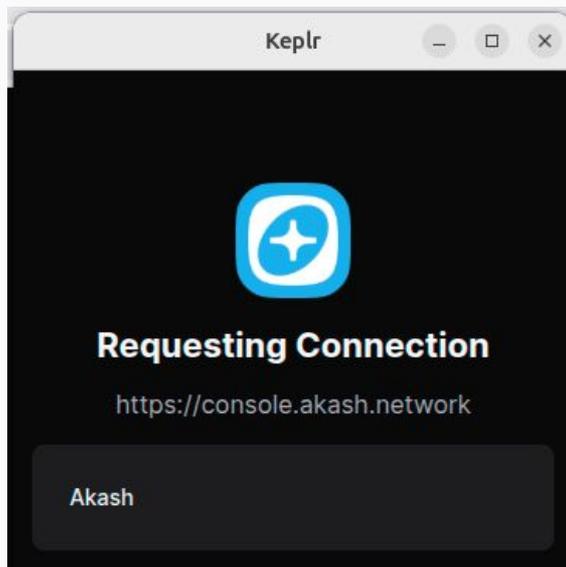
Leap

Leap Cosmos MetaMask SNAP

Cosmostation

Cosmos MetaMask Extension SNAP

Using Akash with Crypto Wallet



Using Akash with Crypto Wallet



DeFi semantics

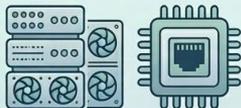
- Install preferred wallet (Keplr)
- Fund your wallet with ATOM (Cosmos coin)
 - Fund from On-ramp (CeFi or other)
- Bridge ATOM to Osmosis
- Use Osmosis DEX to swap ATOM for AKT
- Withdraw AKT back from Osmosis to Keplr

Using Akash with Crypto Wallet

Becoming an Akash Provider



1. Prepare Your Hardware



1. Prepare Your Hardware

Step 1: Dedicate your powerful CPU and high-speed network. Ensure stable power and connectivity. Minimum system requirements: [Example: 8-core CPU, 16GB RAM, 100Mbps Uplink]. Your resources are essential.

2. Install & Configure Software



2. Install & Configure Software

Step 2: Install the Akash client (akash) and configure your provider manifest. Set up resource pricing, availability, and attributes. Use your Akash wallet to sign the manifest. Your node is registered.

3. Offer Resources & Earn



3. Offer Resources & Earn

Step 3: Submit your transaction, approve resource commitments, and watch deployments begin. Your node automatically receives bid requests and matches workloads. Payments are via AKT tokens. Your node is earning!



Decentralized Compute Pool



Provider Network Nodes



Resource Marketplace

Becoming an Akash Provider



Hardware:
Intel I7700K (8 core)
16GB RAM
1TB NVMe

OS:
Ubuntu 24.04 LTS



Becoming an Akash Provider

<https://akash.network/docs/providers/setup-and-installation/provider-playbook/>



Install via provided Ansible Playbook

- Interactive guide
- Flexible config
- Infrastructure as Code (configs versioned)

```
Akash Provider Setup Script

This script will help you set up and configure your Akash Provider.
Answer a few questions and we'll handle the rest!

Select which playbooks you want to run:

[?] Install Kubernetes? (Required for new setup) [y/n]: y

Choose your Kubernetes distribution:
[1] K8s (Kubespray) - Production-grade, full-featured Kubernetes
[2] K3s - Lightweight, single binary, ideal for edge/IoT

[?] Select distribution (1/2) [1]: 2
[?] Do you use a separate storage location for container images? [y/n]: n
[?] Run OS playbook for system optimizations? [y/n]: y
[?] Are there GPU nodes in the cluster? (Will install NVIDIA drivers and container toolkit) [y/n]: n
[?] Install Akash Provider service? [y/n]: y
[?] Set up Tailscale for secure remote access? [y/n]: n
[?] Install Rook-Ceph for storage management? [y/n]: n

You have selected the following playbooks:
[0] K3s - Kubernetes installation using K3s (lightweight, single binary, ideal for edge/IoT)
[0] OS - Basic OS configuration and optimizations
[x] CPU - Are there GPU nodes in the cluster?
[4] Provider - Akash Provider service installation and configuration
[x] Tailscale - Tailscale VPN for secure network access
[x] Rook-Ceph - Rook-Ceph storage operator installation and configuration

[?] Proceed with these selections? [y/n]: y
```

Becoming an Akash Provider

<https://akash.network/docs/providers/setup-and-installation/provider-playbook/>



Utilize an automated pricing script to optimize pricing

Akash provider pricing script with per model GPU pricing

`price_script_generic.sh`

Raw

```
1 #!/bin/bash
2 # WARNING: the runtime of this script should NOT exceed 5 seconds! (Perhaps can be amended via AKASH_BID_PRICE_SCRIPT_PROCESS_T
3 # Requirements:
4 # curl jq bc mawk ca-certificates
5 # Version: Sept-08-2023
6 set -o pipefail
7
8 # # Specify the output file path in the /tmp directory
9 # var_output_file="/tmp/env_variables.txt"
10
11 # # Loop through all environment variables and append them to the output file
12 # for var in $(printenv); do
13 #   echo "$var" >> "$var_output_file"
14 # done
```

Becoming an Akash Provider

<https://gist.github.com/chainzero/33978bf221eb35f10a7392ed9bae8caa>

Questions?



akash