

# TITAN NETWORK Whitepaper 2025

v1.0.0

IDLE RESOURCE  
PROBLEM

TURN IDLE DEVICES  
INTO INCOME

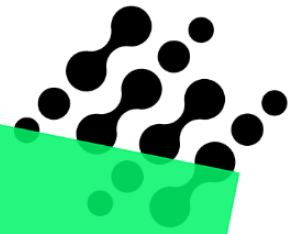
TRANSPARENT  
EARNINGS VS BLACK  
BOX SYSTEMS

SECURITY

REAL BUSINESS USE  
CASES

HYPER-LOCAL  
INFRASTRUCTURE

WEB3 INFRASTRUCTURE  
LAYER



Titan

# TITAN NETWORK Whitepaper 2025

IDLE RESOURCE  
PROBLEM

TURN IDLE DEVICES  
INTO INCOME

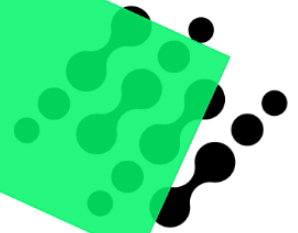
TRANSPARENT  
EARNINGS VS BLACK  
BOX SYSTEMS

SECURITY

REAL BUSINESS USE  
CASES

HYPER-LOCAL  
INFRASTRUCTURE

WEB3 INFRASTRUCTURE



Titan

# **Titan Technical Whitepaper 1.0**

# Table of content

|                                                                         |           |
|-------------------------------------------------------------------------|-----------|
| <b>Table of content</b> .....                                           | <b>3</b>  |
| <b>1. Introduction</b> .....                                            | <b>6</b>  |
| 1.1. Problem space.....                                                 | 8         |
| 1.2. Paper Organization.....                                            | 9         |
| <b>2. Idle Edge Resources</b> .....                                     | <b>10</b> |
| 2.1. Characteristics.....                                               | 10        |
| 2.1.1. Diversity (Heterogeneity).....                                   | 10        |
| 2.1.2. Stability.....                                                   | 11        |
| 2.1.3. Regionality.....                                                 | 11        |
| 2.1.4. Vast Quantity (Scale).....                                       | 11        |
| 2.1.5. Unification Complexity.....                                      | 11        |
| <b>2.2. Concerns of Supply Side</b> .....                               | <b>12</b> |
| 2.2.1. Infrastructure Security Risks.....                               | 12        |
| 2.2.2. Lack of Income Transparency.....                                 | 12        |
| <b>2.3. Concerns of Demand Side</b> .....                               | <b>13</b> |
| 2.3.1. Reliability of DePIN infrastructure and Technical Overhead.....  | 13        |
| 2.3.2. Security of DePIN infrastructure.....                            | 13        |
| 2.3.3. Costs and Transparency.....                                      | 13        |
| <b>2.4. Titan Solutions</b> .....                                       | <b>14</b> |
| 2.4.1. Addressing Idle Resource Diversity.....                          | 14        |
| 2.4.2. Addressing Idle Resource Stability and Reliability.....          | 14        |
| 2.4.3. Addressing Idle Resource Regionality.....                        | 15        |
| 2.4.4. Addressing Idle Resource Scale.....                              | 15        |
| 2.4.5. Addressing Idle Resource Sharing and Renting Security Risks..... | 16        |
| 2.4.6. Addressing Idle Resource Costs and Reward Distributions.....     | 17        |
| <b>3. Titan Network Components</b> .....                                | <b>18</b> |
| 3.1. Architecture.....                                                  | 18        |
| 3.2. Titan Agent.....                                                   | 20        |
| 3.2.1. Support for Multiple Device Types.....                           | 20        |
| 3.2.2. Idle Resource Standardization.....                               | 21        |
| 3.2.3. Resource Quota Scheduling.....                                   | 21        |
| 3.3. Titan DAO.....                                                     | 22        |
| 3.4. Titan Harbour.....                                                 | 23        |
| 3.4.1. Accepting Supervision from Titan DAO.....                        | 24        |
| 3.4.2. Resource Onboarding and Management.....                          | 24        |

|                                                                   |           |
|-------------------------------------------------------------------|-----------|
| 3.4.3. Resource Validation and Reputation Vector.....             | 25        |
| 3.4.4. Resource Scheduling System.....                            | 26        |
| 3.4.5. Resource Earnings System.....                              | 27        |
| 3.4.6. Resource Provider Data Service.....                        | 27        |
| 3.4.7. Titan Client Data Service.....                             | 27        |
| 3.4.8. O&M WEB Platform.....                                      | 28        |
| 3.5. Titan Blockchain.....                                        | 29        |
| 3.6. Titan Smart Contracts.....                                   | 30        |
| 3.6.1. Resource Management Smart Contract.....                    | 30        |
| 3.6.2. Customer Task Smart Contract.....                          | 31        |
| 3.6.3. Harbor Governance Smart Contract.....                      | 31        |
| 3.7. Titan Interfaces.....                                        | 32        |
| 3.7.1. Resource Provider Interface.....                           | 32        |
| 3.7.2. Resource Consumer Interface.....                           | 33        |
| <b>4. Protocol Coordination.....</b>                              | <b>35</b> |
| 4.1. Titan Protocol.....                                          | 35        |
| 4.1.1. Idle Resource Access Protocol.....                         | 36        |
| 4.1.1.1. Agent Login.....                                         | 36        |
| 4.1.1.2. Agent and Harbor Configuration/Data Synchronization..... | 36        |
| 4.1.1.3. Agent State Transition and State Synchronization.....    | 36        |
| 4.1.1.4. Keep-Alive.....                                          | 36        |
| 4.1.1.5. Resource Plane Formats.....                              | 36        |
| 4.1.1.6. Event Subscription.....                                  | 37        |
| 4.1.1.7. Business Scheduling.....                                 | 37        |
| 4.1.2. Resource Provider Service Protocol.....                    | 38        |
| 4.1.2.1. Access Authorization.....                                | 38        |
| 4.1.2.2. Bind or Unbind Idle Resource.....                        | 38        |
| 4.1.2.3. Configure Idle Resource.....                             | 38        |
| 4.1.2.4. View Idle Resource Information.....                      | 39        |
| 4.1.3. Customer Service Protocol.....                             | 39        |
| 4.1.3.1. Access Token.....                                        | 39        |
| 4.1.3.2. Query Harbor Idle Resource Overview.....                 | 39        |
| 4.1.3.3. Query Owned Resource Information.....                    | 39        |
| 4.1.3.4. Order VM Resource.....                                   | 40        |
| 4.1.3.5. Order Container Resource.....                            | 40        |
| 4.1.3.6. Query Customer Task Contract.....                        | 40        |
| 4.2. Resource Validation System.....                              | 41        |

|                                                                      |           |
|----------------------------------------------------------------------|-----------|
| 4.2.1. Titan Validation Network.....                                 | 41        |
| 4.2.2. AI-Generated Service Loads.....                               | 42        |
| 4.2.3. Effectiveness of the Titan Validation System.....             | 42        |
| 4.2.4. Titan's Validation Process and Protocol.....                  | 43        |
| 4.2.4.1. Titan Validation System Application Case Study A.....       | 43        |
| 4.2.4.2. Titan Validation System Application Case Study B.....       | 45        |
| 4.3. Resource Scheduling System.....                                 | 47        |
| 4.3.1. Fairness of Scheduling Policies.....                          | 47        |
| 4.3.2. Specificity of Scheduling Policies.....                       | 48        |
| <b>5. Titan Network Use Cases.....</b>                               | <b>49</b> |
| 5.1. Smart Edge CDN.....                                             | 49        |
| 5.2. Edge Computing.....                                             | 50        |
| 5.3. IP Proxy.....                                                   | 50        |
| 5.4. Web Crawler Applications.....                                   | 51        |
| 5.5. Resource Infrastructure for Various Web2 and Web3 Projects..... | 51        |
| 5.6. Regional Testing & Active Monitoring.....                       | 52        |
| 5.7. Edge GPU Computing and AI Agents.....                           | 52        |
| 5.8. Data Sovereign Storage.....                                     | 53        |
| 5.9. Content and compute localization.....                           | 54        |
| <b>6. Future Development.....</b>                                    | <b>55</b> |
| 6.1. Greater Public Transparency.....                                | 55        |
| 6.2. Better Titan Blockchain Core.....                               | 55        |
| 6.3. Preventing Resource Misuse for Malicious Activities.....        | 56        |
| 6.4. Titan Client Data Security.....                                 | 57        |
| 6.5. Aggregating More Types of Idle Resources.....                   | 57        |
| 6.6. Supporting Rich Transaction Modes.....                          | 58        |
| 6.7. Landing More Types of Customer Businesses.....                  | 58        |
| <b>7. Conclusion.....</b>                                            | <b>60</b> |

# 1. Introduction

The digital economy is accelerating worldwide. As industries adopt technologies such as AI, blockchain, IoT, and real-time applications, compute, storage, and bandwidth have become essential infrastructure, much like electricity in the industrial era. These resources not only power innovation but also form the foundation for scalable digital services.

Despite technological progress, a growing mismatch exists between the demand for digital infrastructure and how resources are actually utilized. A 2023 Global Data Center Energy Efficiency Report found average server utilization in traditional data centers hovering at just 12%-18%. Meanwhile, more than 50% of personal devices remain idle for most of the day. The world is surrounded by underutilized devices - home PCs, routers, mobile phones, even enterprise servers - that consume energy but contribute little value, resulting in sunk costs amounting to hundreds of billions of dollars in idle resources.

At the same time, traditional Cloud Infrastructure faces multiple challenges:

- Emerging technologies such as artificial intelligence (AI), blockchain, and edge computing exhibit explosive growth in demand for elastic, low-cost resources
- Centralized Internet Data Center (IDC) deployments struggle to adequately cover low-latency scenarios like edge computing and real-time rendering
- Issues of resource wastage and energy consumption become increasingly severe

Moreover, centralized control over infrastructure means users must compromise on flexibility, privacy, and locality. These trade-offs are becoming increasingly unacceptable nowadays, and all of this is being forced on the public while digital infrastructure monopolies hinder access to affordable computing resources for small and medium-sized developers, thereby limiting their innovation potential.

Luckily, a parallel trend is emerging: decentralization. Driven by Web3, individuals and organizations are reclaiming control over data, identity, and infrastructure. The concept of data sovereignty, the right to own and govern one's digital assets, is becoming a fundamental expectation. Centralized cloud systems cannot satisfy these emerging demands. People want infrastructure that is not only low-cost and performant but also transparent, censorship-resistant, and community-owned.

In this environment, a significant opportunity exists: aggregating idle compute resources into a new, decentralized digital infrastructure layer. The raw supply is there, but right

now it is dispersed, unused, and globally distributed. What's missing is the coordination layer that can:

- Standardize these heterogeneous devices
- Secure and validate their availability
- Incentivize participation fairly
- Deliver reliable services to end-users

Titan Network is a decentralized protocol and infrastructure network that transforms idle devices into usable compute, storage, and bandwidth. By combining blockchain coordination, smart contracts, and containerized runtime environments, Titan enables anyone to contribute or consume digital infrastructure in a transparent, secure, and efficient way. Through its open-source and modular design, Titan enables a new generation of developers, enterprises, and communities to access compute infrastructure that is decentralized, cost-effective, and close to end consumers.

## 1.1. Problem space

Modern computing faces a paradox of **idle supply vs. growing demand**. Billions of personal devices (PCs, smartphones, IoT devices, and home servers) worldwide sit unused for large portions of the day, representing vast wasted computing potential. At the same time, demand for distributed computing and low-latency services is increasing, yet centralized cloud providers still dominate, with high costs and limited geographic reach. This imbalance leads to:

- **Underutilized Infrastructure:** Huge numbers of computing devices consume energy without performing useful work. The aggregate idle capacity, if harnessed, could provide enormous computing power.
- **Centralization Pains:** Enterprises and users relying on a few cloud providers face high costs, latency issues for distant users, data privacy concerns, and vendor lock-in. Many everyday workloads do not require ultra-high-performance supercomputers, but rather distributed, moderate compute with low latency to end-users.

The emerging Decentralized Physical Infrastructure Network (DePIN) concept offers a way to bridge this gap. By networking and incentivizing these idle resources, one can create a distributed cloud that is cost-effective, scalable, and powered by the community. Titan Network targets this opportunity by providing the technical and economic framework to unlock idle edge resources at scale.

Vast quantities of idle resources distributed worldwide, if fully utilized, could not only prevent resource waste but also generate income for resource owners. More importantly, they can complement traditional centralized resource services, alleviating pressure and addressing their shortcomings.

## 1.2. Paper Organization

This paper is organized to build a progressive understanding of the Titan Network. From the problem it addresses, to its architecture, protocol design, and practical applications. Each chapter deepens the exploration of a specific dimension of the system:

- **Chapter 1:** Introduction - outlines the broader context of digital infrastructure, identifies key inefficiencies in the current model, and introduces Titan as a solution to unlock idle edge resources at scale.
- **Chapter 2:** Idle Edge Resources - analyzes the nature and challenges of using idle infrastructure, then breaks down device characteristics (heterogeneity, stability, locality, scale), concerns from both supply and demand sides, and how Titan systematically addresses each of them.
- **Chapter 3:** Titan Network Components - describes the core building blocks of the Titan Network, including the Titan Agent, Titan Harbour, Titan DAO, smart contracts, and interfaces. The chapter then defines how resources are onboarded, validated, scheduled, and monetized.
- **Chapter 4:** Protocol Coordination - details how Titan's protocols govern resource access, service interactions, validation, and scheduling. Includes the complete breakdown of communication flows between agents, Harbors, and customers, as well as Titan's approach to network security through a robust validation layer.
- **Chapter 5:** Titan Network Use Cases - demonstrates how Titan can support real-world applications, such as edge CDN, compute, IP proxy, decentralized storage, AI agents, and highlights both Web2 and Web3 integration opportunities.
- **Chapter 6:** Future Development - discusses planned system enhancements, including transparency, blockchain upgrades, security measures, and broader support for additional workloads and transaction modes.
- **Chapter 7:** Conclusion - summarizes Titan's role in rebalancing the digital infrastructure market by enabling decentralized, verifiable, and community-owned compute.

Each chapter is designed to stand on its own while contributing to a complete understanding of Titan's technical and economic framework.

## 2. Idle Edge Resources

Titan Network is built on a simple yet powerful idea: the world already has more infrastructure than it uses - it's just idle, fragmented, and uncoordinated. These idle edge resources, though individually limited, collectively represent an immense and underutilized infrastructure layer. This chapter examines the characteristics of these resources, the concerns faced by both supply-side contributors and demand-side users, and how Titan is designed to bridge this gap by transforming idle capacity into verifiable, trusted, and productive infrastructure through decentralized coordination.

### 2.1. Characteristics

Idle edge resources differ significantly from centralized infrastructure. They vary in hardware capabilities, network stability, geographic location, and availability. Understanding these characteristics is essential to designing a system that can effectively onboard, manage, and utilize such resources.

#### 2.1.1. Diversity (Heterogeneity)

A wide variety of idle resources exist in reality. From a location perspective, there are idle resources in homes, IDCs, office environments, and countless other settings. Considering device-based idle resources within homes, various devices are often present that are idle: smartphones, tablets, laptops, desktops and other smart devices.

These home devices are not only numerous but also possess rich resources and capabilities. For example, many smartphones feature:

- 1GB to as high as 16GB of RAM.
- 16GB to as high as 1+TB of storage.
- CPU/GPU computational capabilities are approaching those of laptop CPUs/GPUs.

If these resources could be fully utilized, substantial value could be generated.

These idle resources could be further differentiated across multiple dimensions

- Operating System Type: Windows, Linux, macOS, etc., with various versions inside
- CPU Architecture: x86, ARM, MIPS architectures, etc.
- Memory Type and Quantity: DDR5/4/3 types, quantities like 4GB/8GB/16GB, etc.
- Storage Type and Quantity: HDD, SSD, quantities like 256GB/512GB/1TB, etc.
- Connectivity and Network Types: Static/Dynamic IP, NAT1/2/3/4, Fiber/5G/SAT, etc.

### **2.1.2. Stability**

A key characteristic of idle resources is their relative lack of stability in the following areas:

- Consumer-grade vs commercial-grade equipment reliability
- Power supply stability (e.g., lack of Uninterruptible Power Supply (UPS))
- Broadband and internet stability (e.g., shared broadband facilities among multiple households)
- Availability of resources due to device utilization (e.g., fewer available resources due to user device utilization)

### **2.1.3. Regionality**

Multi-regionally distributed idle resources, from a physical distance perspective, can effectively serve demands within the same region due to proximity. However, consolidating idle resources across multiple regions for utilization is challenging due to potentially unacceptable network latency incurred by cross-regional communication.

### **2.1.4. Vast Quantity (Scale)**

The primary source of idle resources is household devices. Even within a single region, such as a city, the scale of idle resources is enormous. From a resource perspective, substantial network bandwidth is required for communication with idle resources, and powerful computational capabilities are needed for scheduling these resources to provide services.

### **2.1.5. Unification Complexity**

While the diversity, scale, and geographic distribution of idle edge resources present immense potential, they also introduce real-world complexity. These devices are often perceived as unstable and difficult to manage consistently across regions. For owners and potential consumers of such resources, this complexity raises significant concerns regarding security, reliability, and economic transparency. The following section examines these challenges from the supply side, focusing on what prevents device owners from contributing their idle capacity today.

## **2.2. Concerns of Supply Side**

From the perspective of idle resource owners, allowing resources to sit idle is indeed regrettable. However, if using these idle resources requires taking risks, they would rather not use them at all to avoid potential harm.

### **2.2.1. Infrastructure Security Risks**

Allowing external use of idle resources involves running external programs. If these programs are malicious, data security issues can arise. This security risk endangers not only the idle resource itself but also its entire local area network (LAN). If the idle resource is located within a home, all devices on the home LAN could be at risk; if located in an IDC, all devices within that IDC might be compromised.

### **2.2.2. Lack of Income Transparency**

Resource owners often see a lack of transparency in the income generated from their idle resources. Numerous predecessor projects have aggregated idle resources to provide services, generating revenue and distributing a portion back to resource owners. The inherent problem lies in the black-box nature of this process; resource owners lack visibility into the actual earnings and their share, potentially receiving only a fraction of the actual revenue.

This black-box model is equivalent to deception, as it harms the interests of both resource owners and resource users (clients). Consequently, resource owners are unwilling to support such black-box models and are hesitant to contribute their idle resources to such projects.

## **2.3. Concerns of Demand Side**

### **2.3.1. Reliability of DePIN infrastructure and Technical Overhead**

The ease of deploying, managing, and updating software on distributed idle nodes is a critical concern. Customers need intuitive interfaces like dashboards, similar to traditional cloud solutions for monitoring, deployment, versioning, and troubleshooting across many edge resources. Without efficient tooling and automation, operational overhead can outweigh the benefits of lower-cost infrastructure.

### **2.3.2. Security of DePIN infrastructure**

Deploying proprietary code or content on a decentralized network of community-owned devices raises serious concerns around data integrity, confidentiality, and execution trust. Without sufficient security or clear accountability, customers may hesitate to adopt DePIN solutions. There is a need to ensure isolated access points, verifiable execution, and strong security layers to give users confidence that their applications and data remain safe across a globally distributed infrastructure with programmable confidence.

### **2.3.3. Costs and Transparency**

Customers, particularly large enterprises, seek large volumes of compute and network resources at minimal cost. While idle resources offer affordability, opaque or black-box aggregation models often inflate prices or misallocate fees, resulting in poor value added for customers. Demand-side actors require a transparent, open platform where pricing is predictable, intermediaries are minimized, and cost reflects actual resource quality and usage.

## **2.4. Titan Solutions**

Addressing the characteristics of idle resources and the concerns of demand and supply, Titan considers the perspectives of both resource providers and users, proposing corresponding solutions for each key point.

### **2.4.1. Addressing Idle Resource Diversity**

To manage the diversity of idle resources, Titan Protocol has been designed to abstract heterogeneous idle resources across multiple dimensions, minimizing the differences between various types.

Titan Protocol standardizes the top-level abstraction of idle resources, establishing resource planes primarily based on containers (K8S containers, Titan lightweight containers) and Virtual Machines (VMs). Various idle resources provide services to Titan clients through these Titan Protocol-defined resource planes.

Simultaneously, the Titan Protocol defines the access and management protocols for idle resources joining the Titan ecosystem. Therefore, any idle resource needs only to adhere to the Titan Protocol specifications to join the ecosystem and accept Titan's scheduling. In Chapter 3.2. Titan Agent is explained in detail, which implements the resource planes and access specifications defined by the Titan Protocol. As the Titan ecosystem evolves, the community will release Titan Agents supporting a wider range of idle resource types, enabling broader participation.

### **2.4.2. Addressing Idle Resource Stability and Reliability**

To counteract the insufficient stability of idle resources, the Titan ecosystem employs multiple strategies.

Firstly, Titan assigns a stability score and rating to each idle resource. After an idle resource joins the ecosystem, Titan continuously monitors metrics across multiple dimensions and scores the resource quality. Based on this score, the idle resource is classified into one of several stability levels.

Secondly, the Titan ecosystem allows resource owners to pledge collateral against their idle resources. If stability issues arise, a certain proportion of the collaterals may be deducted. In essence, resource owners guarantee the stability of their resources through a pledge.

Within the Titan ecosystem, Titan Harbor is responsible for resource scheduling. Its

resource scheduling system fully considers factors such as the stability level of idle resources and whether they have a stability guarantee (i.e., pledge) during resource allocation.

For example, for stateless Titan client workloads requiring 99% reliability, Titan Harbor could select three idle resources, each with 90% reliability. Statistically, this combination can meet the client's reliability requirement. For stateful Titan client workloads, which are difficult to migrate smoothly and thus demand highly reliable resources, Titan Harbor can select guaranteed (pledged) idle resources to fulfill the client's needs.

As the Titan ecosystem continues to develop and mature, more methods for addressing the reliability shortcomings of idle resources will be introduced and implemented. For instance, machine learning and AI could be used to predict the stability of idle resources over a future period based on historical stability data.

### **2.4.3. Addressing Idle Resource Regionality**

To address the regional nature of idle resources, the Titan ecosystem prioritizes the regional attribute as its primary attribute. Each idle resource connects to the Titan Harbor responsible for its region based on its location. Each region has at least one Titan Harbor responsible for managing the idle resources within that area. The Titan ecosystem avoids cross-Harbor resource scheduling during resource allocation.

Additionally, idle resources possess an IDC-UUID attribute that identifies the specific IDC facility where the resource is located. Idle resources sharing the same IDC-UUID value are grouped by Titan Harbor as resources within the same IDC, implying they can interconnect via a high-speed LAN.

Currently, regions within the Titan ecosystem are defined geographically at the country or continental levels. As the ecosystem grows with more idle resources joining and demands for finer granularity increase, the physical scope of regions may be refined, potentially down to the city level.

### **2.4.4. Addressing Idle Resource Scale**

Firstly, the Titan ecosystem deploys Titan Harbors based on physical regions, with each Harbor managing only the idle resources within its designated area.

Secondly, each Titan Harbor is a cluster system composed of multiple service programs. Each service program supports multi-instance distributed deployment. This not only reduces the load on individual instances but also mitigates the risk of service

interruption due to single points of failure.

Furthermore, critical core data related to idle resources, such as earnings, is anchored to the Titan blockchain via smart contracts. This ensures that even in the worst-case scenario of a Titan Harbor failure, the earnings of resource owners are protected.

#### **2.4.5. Addressing Idle Resource Sharing and Renting Security Risks**

Regarding the security of idle resources, the Titan ecosystem achieves this through thorough isolation. Titan client workloads run within a sandbox environment, ensuring they cannot breach the sandbox to interact with the external environment. This ensures the security of the idle resource itself and the security of the local area network (LAN) on which it resides.

Currently, the Titan ecosystem implements and supports the following four types of sandboxes. As the ecosystem evolves, more sandbox types will be supported:

1. **K8S Containers:** What are containers? Containers are standardized, lightweight, and secure program execution environments. Because they share the kernel with the host machine's operating system, they exhibit extremely fast startup times and low resource consumption, making them lightweight compared to virtual machines (VMs). Simultaneously, a container provides an environment isolated from the host OS. Titan client business programs within a container cannot illicitly access facilities outside the container. K8S containers refer to containers managed by the K8S container orchestration system. Titan clients can efficiently utilize the idle resources of the Titan ecosystem through K8S or multi-tenant container orchestration systems.
2. **Virtual Machine (VM):** A virtual machine creates a virtualized instance of a computer on a host, running a separate operating system. The VM is isolated from the host machine, preventing Titan client business programs inside the VM from illegally accessing facilities outside it. The advantage of VMs is the ability to run an operating system completely different from the host's, determined by the Titan client's requirements. The disadvantage, stemming precisely from needing to run an independent OS, is that VMs require more resources and have slower startup times compared to containers. The Titan ecosystem primarily utilizes KVM virtualization on Linux platforms, VirtualBox on Windows and macOS platforms, and also supports VMWare VMs.
3. **JS/WASM Container:** This container differs from the K8S containers mentioned above. It cannot run typical container images. Instead, it is a JavaScript/WASM runtime environment built on Chromium's V8 JavaScript engine, capable of

executing programs written in languages such as JavaScript and WASM. The Titan ecosystem built this JS/WASM runtime environment based on Worker-D (an open-source runtime environment based on Chromium V8), ensuring isolation from the host machine. Titan client business programs, developed using JavaScript/WASM, run within Titan's JS/WASM containers and thus cannot access facilities on the host operating system in an unauthorized manner. The advantage of JS/WASM containers is their extremely lightweight nature, minimal resource consumption, and rapid startup. However, the disadvantage is the requirement for Titan client businesses to write programs specifically in JavaScript or WASM, which represents a significant overhead for many clients.

4. **TERMUX Sandbox:** This type of sandbox primarily exists for idle Android phones within the Titan ecosystem. It is built upon libraries provided by the renowned Android community terminal emulator, TERMUX. Its purpose is to allow Android phones to run console programs within an isolated environment. With the TERMUX sandbox, Titan client workloads can run as Android console programs on idle Android phones within the Titan ecosystem. The client's program cannot access phone facilities outside the TERMUX sandbox.

#### **2.4.6. Addressing Idle Resource Costs and Reward Distributions**

To ensure transparency of income from idle resources, the Titan ecosystem anchors core earnings data to the blockchain via smart contracts. The earnings from idle resources and their distribution are made completely open and transparent by publishing them on the Titan blockchain.

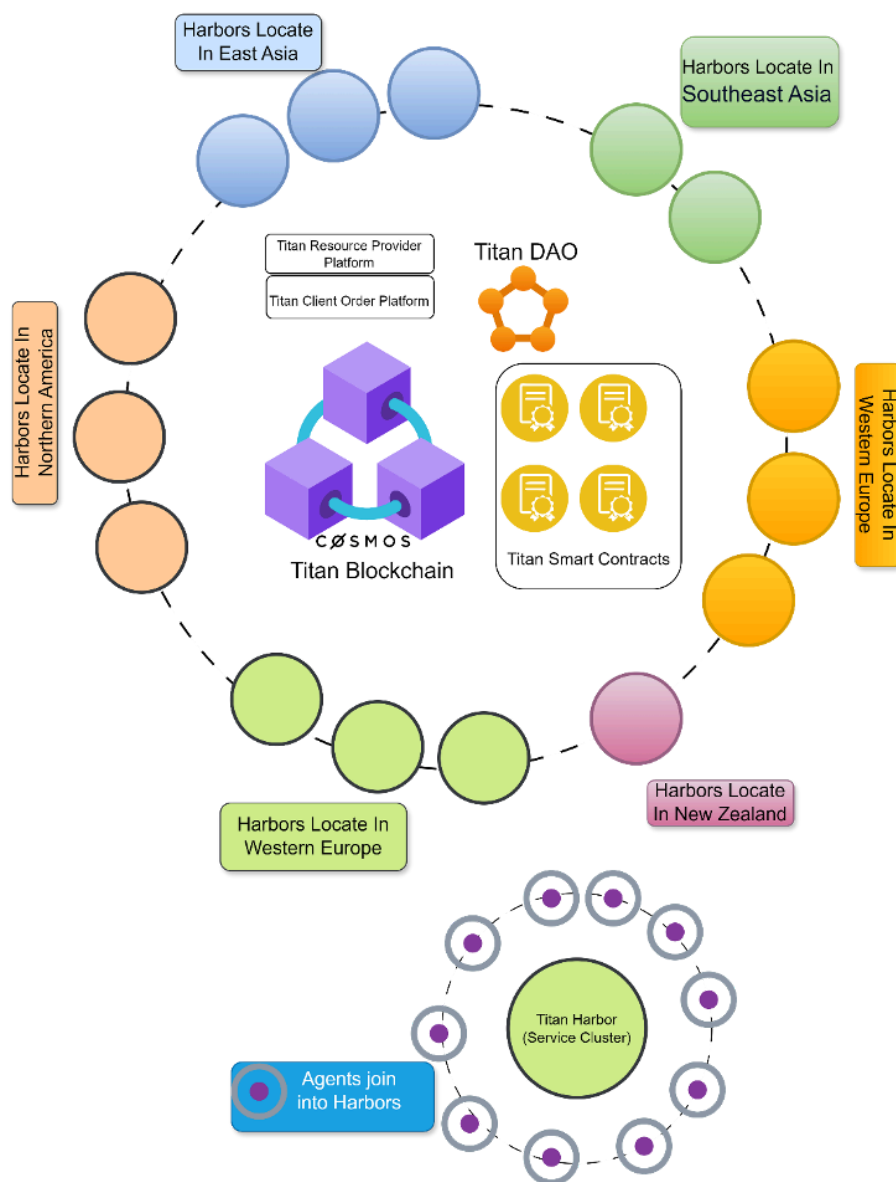
The payment amount and cost of service calculation bases must also be publicly transparent. The Titan ecosystem uses Titan DAO to set unit prices, discounts, and other pricing information for various resource types in each region. This information is published to the Titan blockchain via smart contracts. The payment made by the Titan client is calculated precisely based on the pricing information residing on the Titan blockchain and the quantity of resources ordered.

Each Titan client's order is published to the Titan blockchain as an order smart contract. For each idle resource, information detailing who it provided services for, the duration of service provision, and the corresponding earnings received will also be published to the Titan blockchain. With this publicly transparent data residing on the blockchain, both resource owners and Titan clients can clearly understand where value originates and where it flows.

# 3. Titan Network Components

This section provides a high-level overview of Titan's core subsystems. These subsystems are essential components forming the Titan ecosystem. They utilize the Titan Protocol as the foundational communication protocol, collaborating to provide high-quality services to participants, including idle resources, resource owners, and resource users.

## 3.1. Architecture



[Image Description: The diagram shows various Idle Resources connecting via Titan Agents to their respective regional Harbors. Harbors manage and schedule these resources. Clients interact via the Titan Customer Service Platform to submit resource requests. Harbors fulfill these requests using the managed idle resources. Client tasks are anchored as Smart Contracts on the Titan Blockchain. Titan DAO governs the entire ecosystem, including Harbors and Service Platforms.]

The figure above illustrates Titan's deployment architecture.

Various types of idle resources, after being standardized by Titan Agents, converge at the Harbor corresponding to their respective regions.

Harbor is responsible for managing and scheduling these idle resources, providing resource services to clients.

Clients, through the Titan customer service platform, publish their business resource requirements. One or more Harbors schedule the idle resources under their management to fulfill the client's resource needs.

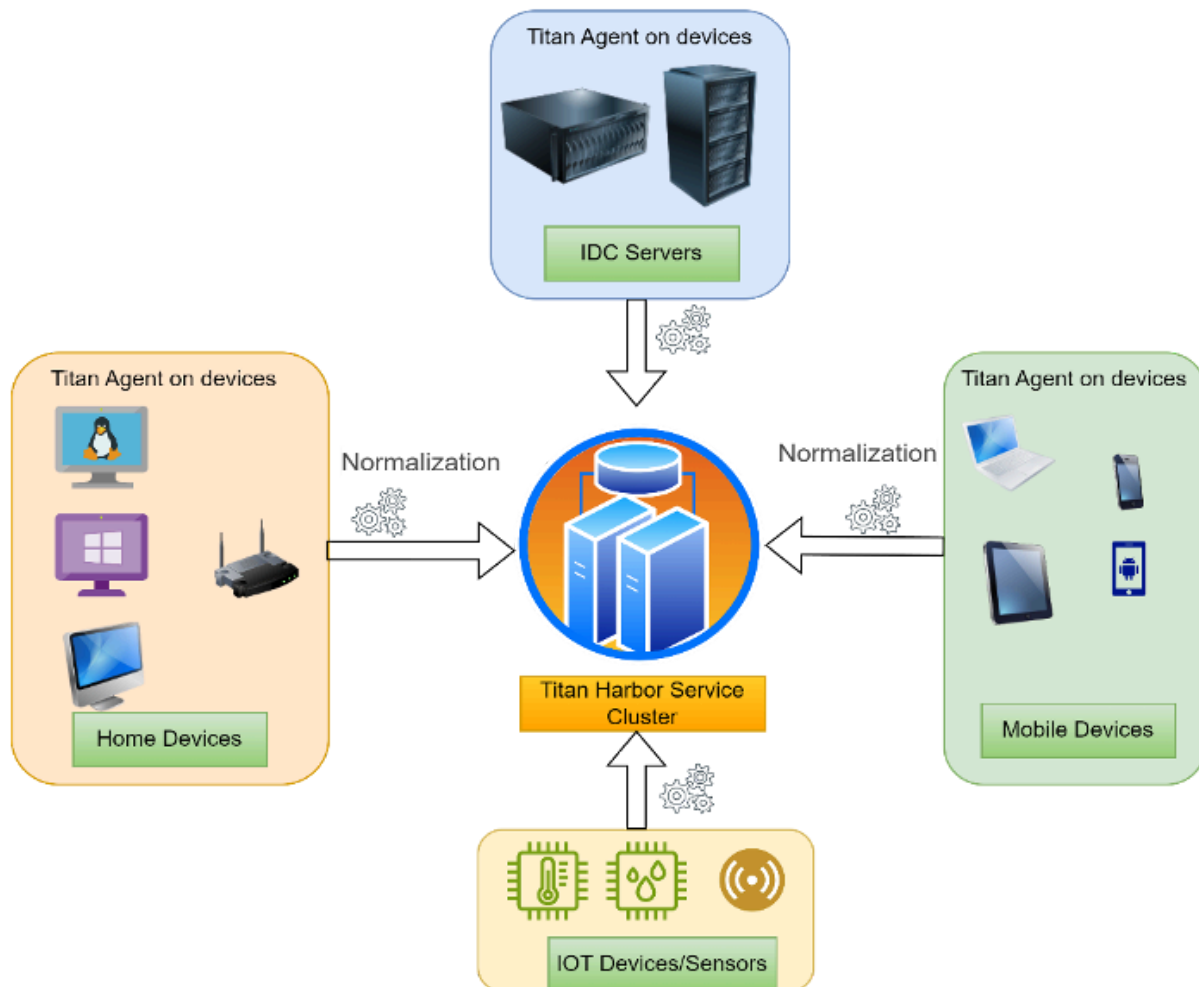
Each client task is anchored to the Titan blockchain through a programmable smart contract, ensuring transparency throughout the entire process. The smart contract contains not only the detailed requirements but also data information and logic code pertaining to the conditions necessary for contract completion.

Titan DAO, serving as the core of ecosystem governance, implements community co-governance through a modular governance framework. Key responsibilities of Titan DAO include, but are not limited to:

- Titan governance, such as supervising Titan Harbors and promoting the continuous development of the Titan ecosystem.
- Dispute resolution, including handling disputes related to client service quality and idle resource earnings.
- Ecosystem supervision, ensuring the appropriate use of Titan's resources and the healthy development of the Titan ecosystem.

## 3.2. Titan Agent

The Titan Agent is an open-source software suite designed to standardize idle resources and connect them to a Titan Harbor according to the Titan Protocol. Only after this process can idle resources be scheduled and utilized by Titan.



### 3.2.1. Support for Multiple Device Types

The types of devices that can serve as idle resources are varied. The Titan Agent can run on various hardware (e.g., Windows, Linux, macOS PCs, datacenter-grade infrastructure) as well as mobile devices (e.g., Android phones, tablets, IoT). Therefore, the Titan Agent can standardize various idle resources and integrate them into the Titan idle resource network, allowing them to contribute to the collaborative development of the Titan ecosystem.

### **3.2.2. Idle Resource Standardization**

Standardizing idle resources is required to comply with the constraints of the Titan Protocol. Due to the diversity of resources, it is necessary to abstract away underlying differences as much as possible to provide clients with a smooth user interface, enabling them to utilize idle resources conveniently.

From the resource provider's perspective, they are concerned not only with the revenue generated by the Titan ecosystem but also with resource security. Standardizing idle resources ensures security, as the Titan Protocol requires that standardization occur in an isolated manner, independent of the host machine.

The Titan Agent selects appropriate standardization technologies based on the type and characteristics of the idle resource. For example, for Windows PC-type idle resources, the Titan Agent uses VM technology for standardization. For mobile phone-type idle resources, it uses JS/WASM containers. For Linux PCs or servers, it utilizes Kubernetes (K8S) containers, and so forth.

### **3.2.3. Resource Quota Scheduling**

A resource quota plan is needed not only to show the available quota of idle resources but also to indicate the time periods during which they are available. Titan Agent needs to dynamically adjust Titan's resource usage limits based on the resource quota plan.

Idle resource owners can set a Titan availability quota plan. For example, home bandwidth resources can be allocated quotas based on time slots. Different time slots allow Titan to use varying bandwidth quotas: e.g., during late-night rest periods, Titan might utilize almost the full bandwidth, but during times when family members require bandwidth, Titan must release sufficient capacity to avoid impacting their experience.

### **3.3. Titan DAO**

Titan DAO is the central governance hub of the Titan ecosystem, shouldering the responsibility for its sustainable development. Titan DAO supervises the operations and service quality of the two core service platforms within the ecosystem (Resource Provider Service Platform and Customer Service Platform) and approves their quality improvement plans.

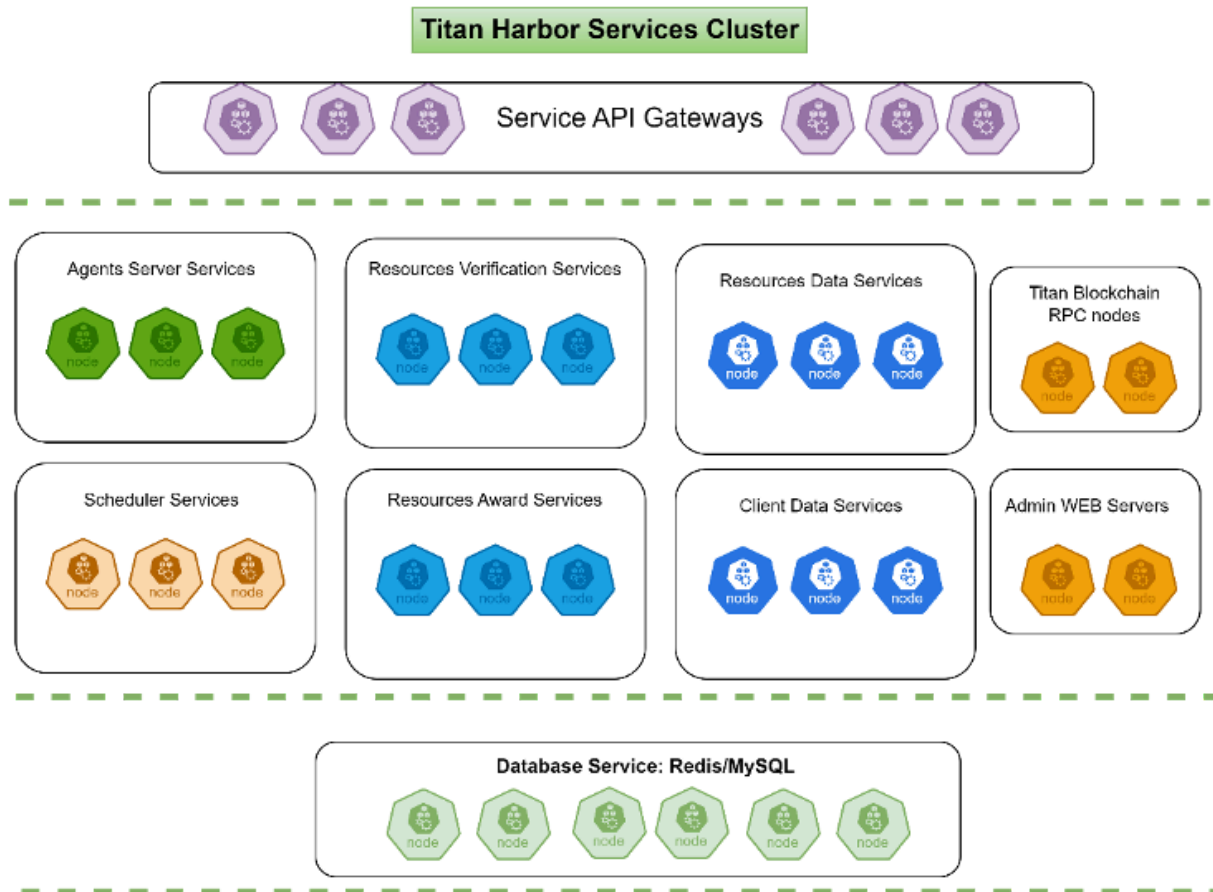
Based on the evolving needs of the Titan ecosystem, Titan DAO approves plans for new Titan Harbor operations or revokes existing ones. Titan DAO continuously monitors the operational and service quality of Titan Harbors within the ecosystem and approves their quality improvement plans.

Titan DAO is responsible for handling disputes within the Titan ecosystem, such as those raised by idle resource providers or service quality disputes from Titan clients.

When the Titan ecosystem lacks sufficient quantities of a specific type of idle resource to meet demand, Titan DAO could release solicitation plans for that resource type to the community and formulate corresponding incentive schemes.

### 3.4. Titan Harbour

Harbors act as the central hubs of the Titan ecosystem. They interface northbound with the customer service platform, responding to client resource demands, and southbound with the idle resources represented by Titan Agents, onboarding and managing all Titan Agents within their region.



Harbor is a service cluster composed of various service processes, including but not limited to:

- Agent Management Service (Agent-Server)
- Resource Validation Service (Agent-Reputation)
- Resource Scheduling Service (Scheduler)
- Resource Earnings Service (Awardee)
- Data Service for Resource Providers (Provider-Data-Server)
- Data Service for Titan Clients (Customer-Data-Server)
- O&M WEB Platform

The Harbor service cluster implements the protocols defined by the Titan Protocol, allowing external entities to access and interact with the various services within Harbor by adhering to these protocols.

To prevent single points of failure, all service processes within the Harbor service cluster support multi-instance deployment. These service processes run on a K8S container cluster.

### **3.4.1. Accepting Supervision from Titan DAO**

Multiple Harbors exist within the Titan ecosystem, each managing a certain number of idle resources. This distributed deployment both distributes load and avoids single-point-of-failure risks.

Titan DAO is responsible for managing various aspects of Titan Harbors, such as determining the region and quantity of Harbors (e.g., how many Harbor clusters to deploy in which region). It also oversees Harbor service quality monitoring and improvement, tracking metrics like response speed to client resource requests and the quality of idle resource scheduling, requiring Harbors to improve these indicators continuously.

### **3.4.2. Resource Onboarding and Management**

Titan Harbor accepts connection requests from Titan Agents. Agents register their resource attributes with Titan Harbor according to the constraints of the Titan Protocol.

Titan Harbor allocates corresponding data storage space for each Agent to record its resource attributes, such as device type (home PC or IDC server), operating system (Windows or Linux), CPU architecture (ARM or x86), etc.

When an Agent is online, Harbor marks its resources as available and adds them to the corresponding resource pools based on various dimensions. When an Agent goes offline, its resources are marked as unavailable by Harbor and removed from the relevant resource pools.

Harbor manages the current online/offline status of each Agent. Simultaneously, Agents periodically generate reports on their resource status and send them to Harbor, allowing Harbor to promptly update its database to reflect changes in resource availability.

Harbor knows which client tasks are allocated to which Agent and the running status of these tasks. Through this data, Harbor maintains a real-time understanding of its overall

resource status, including resources currently in use, available idle resources, and more.

Harbor constructs various resource pools based on multiple dimensions and categorizes idle resources into these pools to facilitate rapid resource scheduling. For example, based on the deployment type dimension of client tasks, idle resources can be categorized into K8S Pod resource pools, Linux VM resource pools, etc.

### **3.4.3. Resource Validation and Reputation Vector**

The authenticity and validity of resources are crucial to the Titan ecosystem. For example, if a resource provider promises 100 Mbps of upstream bandwidth but actually provides only 1 Mbps, allocating this resource to a client will negatively impact the client's business operations and degrade the customer experience.

To maximize assurance of resource authenticity and validity, Titan needs to implement a multidimensional, multilayered validation system. The need for multiple dimensions and layers arises because Titan ecosystem resources are diverse (e.g., compute, bandwidth, storage), requiring different validation methods. Consequently, the current validation methods must be considered at multiple levels to minimize resource cheating.

Furthermore, by leveraging the large number of home and mobile devices within the Titan ecosystem, Titan has developed an Availability Validation Service (AVS). Clients can use Titan's AVS to periodically validate their services, ensuring their operational effectiveness.

Titan's validation system randomly initiates validations on idle resources and anchors the results to the Titan blockchain via smart contracts, ensuring public transparency.

Validation results ultimately influence the idle resource's reputation vector. The reputation vector is a multidimensional set of metrics, each indicating the resource's reliability in a specific dimension (e.g., online indicator, CPU capability indicator, memory availability indicator, bandwidth availability indicator, etc.).

When Titan Harbor schedules idle resources for client tasks, it prioritizes resources with better reputations. Therefore, idle resources with higher reputations have a greater chance of serving clients and earning more revenue.

### 3.4.4. Resource Scheduling System

Titan clients typically establish service requirements through Titan's customer service platform. When creating a requirement, clients can actively select one or more Harbors to provide idle resources for their service. If the client does not make an active selection, the customer service platform automatically assigns one or more suitable Titan Harbors based on matching criteria across multiple dimensions (e.g., the target operational region of the client's task).

Upon receiving a resource requirement from the customer service platform, Titan Harbor's idle resource scheduling system begins the resource allocation process for the client task. This allocation involves a combination of policies and algorithms that evolve over time, continuously improving under the supervision of Titan DAO.

Currently, Titan Harbor's resource scheduling strategy considers the following aspects:

- Selection based on the runtime environment type chosen by the client task. For example, a container environment requires selecting idle resources that support container runtimes, while a virtual machine environment requires selecting resources that support virtual machines (VMs). The supported runtime environment of an idle resource is determined during its standardization by the Titan Agent based on the resource's nature.
- Prioritization of idle resources with better reputation vectors. For instance, a higher online indicator in the reputation vector signifies greater stability, whereas a lower value suggests a higher likelihood of the resource going offline during service, forcing the scheduling system to reschedule to meet the client's needs again.
- Resource allocation requires thorough consideration of redundancy, given that the dependability of idle resources doesn't measure up to commercial-grade resources. For example, if a client task requires 1 GB of bandwidth, Titan Harbor might allocate an additional 200 MB (20% redundancy) during scheduling.
- Continuous tracking of resource changes while the client task is running, responding promptly when rescheduling is necessary to minimize disruption to the client's business.
- Potential collaboration between the scheduling systems of multiple Titan Harbors to achieve better overall resource scheduling.

Additionally, Titan Harbor's resource scheduling system must support scheduled tasks, meaning performing resource allocation for customer services at a designated future time. This requires the system to assess the projected availability of Harbor-managed idle resources for that specific future point, thereby preventing potential failures in

executing the scheduled allocation when the time arrives.

In summary, Titan Harbor's resource scheduling subsystem continuously strives to identify more optimal solutions for resource allocation. In the future, Titan plans to incorporate Machine Learning (ML) and Artificial Intelligence (AI). These advancements will leverage the analysis of diverse data within the Titan Harbor database to predict idle resource availability and propose superior resource allocation strategies.

### **3.4.5. Resource Earnings System**

The resource earnings system is responsible for calculating the earnings for idle resources managed by Harbor and periodically anchoring this earnings data to the Titan blockchain via smart contracts.

Idle resource earnings include incentive rewards set by Titan DAO, earnings from contributing resources to Titan client tasks, staking rewards, etc.

The resource earnings system is also responsible for calculating penalties incurred by idle resources, such as failing spot checks by Harbor's validation system or failing to provide the promised service quality for Titan client tasks.

### **3.4.6. Resource Provider Data Service**

The primary responsibility of the Resource Provider Data Service is to implement the data service protocols defined in the Titan Protocol, providing data access and modification services for resource providers.

Idle resources connect to Titan Harbor via the Agent. Harbor, through its Resource Provider Data Service, offers interfaces for resource providers to manage their owned idle resources, including but not limited to:

- Managing the association between Resource Providers and idle resources, specifically binding and unbinding.
- Configuration of owned idle resources by Resource Providers.
- Allowing Resource Providers to view details of the idle resources they own.

### **3.4.7. Titan Client Data Service**

The primary responsibility of the Titan Client Data Service is to implement the data service protocols defined in the Titan Protocol, providing data access and modification services for Titan clients.

Titan Harbor's resource scheduling service allocates idle resources for Titan clients. Through the Client Data Service, Titan Harbor allows clients to operate the idle resources they have purchased, including but not limited to:

- Querying information about purchased idle resources.
- For VM-type idle resources, providing access methods (e.g., IP address/port, account credentials).
- For container-type idle resources, providing entry points to container orchestration systems, allowing clients to deploy workloads via these systems.

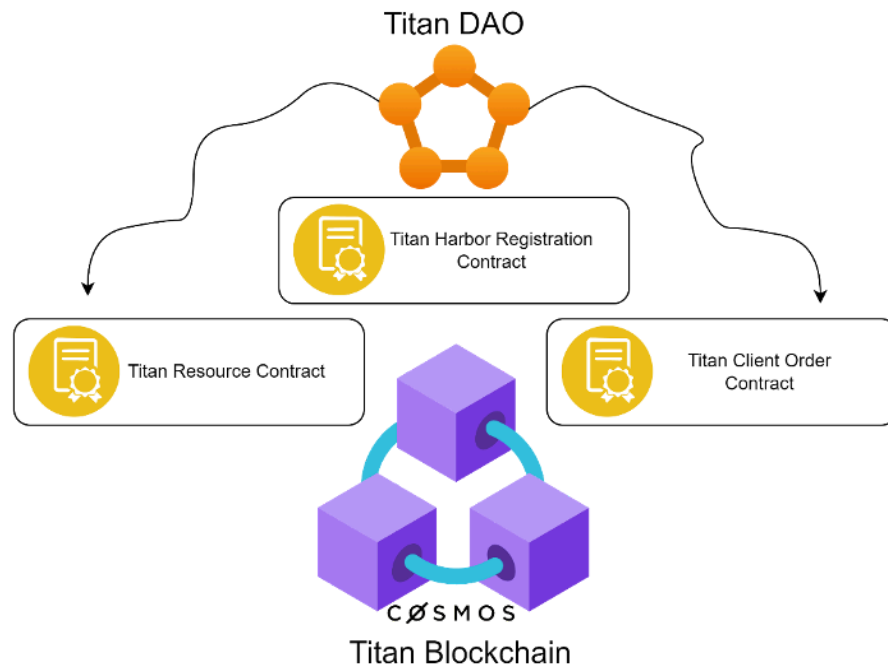
### **3.4.8. O&M WEB Platform**

The O&M WEB Platform is a set of WEB servers used for the operation and management of Titan Harbor. Through this platform, Titan Harbor administrators can view the current runtime status of the Harbor, examine alert events, modify runtime configurations, etc.

Combined with common operational and monitoring tools like Prometheus, this ensures that Titan Harbor can provide reliable idle resource management and scheduling services 7x24.

### 3.5. Titan Blockchain

Titan L1 blockchain technology forms the foundation for Titan's decentralized governance and community collaboration.



Titan blockchain is built upon Cosmos. Cosmos meets the current TPS (Transactions Per Second) requirements and existing functionality relevant to the Titan ecosystem on mainnet launch. However, as the number of idle resources and Titan clients in the ecosystem continues to grow, the demand for blockchain TPS will increase significantly. At that time, Titan DAO will, as needed, transition to a blockchain base with higher TPS, potentially employing mechanisms like DAG-based block synchronization to replace the current Cosmos block synchronization mechanism.

When selecting data to be recorded on-chain, Titan must take into account the following aspects:

- **Data Volume:** Minimizing storage requirements while ensuring core data is on-chain.
- **On-chain Frequency:** Reducing TPS demand as much as possible to avoid pressuring the current Titan blockchain.

## 3.6. Titan Smart Contracts

The two most essential smart contracts within the Titan ecosystem are the Resource Management Smart Contract and the Customer Task Smart Contract. The Resource Management Smart Contract manages available resources and resource providers, while the Customer Task Smart Contract serves Titan clients.

### 3.6.1. Resource Management Smart Contract

Each Titan Harbor possesses its own dedicated Resource Management Smart Contract to avoid overwhelming a single contract with data management pressure from a large number of idle resources.

Every idle resource creates a Titan wallet upon initialization. This wallet address serves as the identity representation of the idle resource on the Titan blockchain.

After an idle resource connects to a Titan Harbor and successfully joins the Titan ecosystem, the Harbor invokes the resource initialization interface of the Resource Management Smart Contract to complete the idle resource's initialization on the Titan blockchain.

Titan Harbor will only register an idle resource onto the Resource Management Smart Contract when the idle resource's wallet has a sufficient balance to cover the GAS fees for contract interface calls.

The dataset for each idle resource within the Resource Management Smart Contract includes, but is not limited to:

- The owner of the idle resource (i.e., the Resource Provider).
- Pledge data for the idle resource.
- Earnings data for the idle resource.

Once an idle resource completes its on-chain initialization, Titan Harbor will periodically submit the operational data of the idle resource, such as daily earnings, to the Titan blockchain.

Resource Providers can call the pledge interface of the Resource Management Smart Contract to pledge collaterals against their idle resources, thereby improving the resource's reputation vector metrics and enabling it to meet the requirements of more Titan client tasks.

### **3.6.2. Customer Task Smart Contract**

When a Titan client purchases resource services through the Titan ecosystem's customer service platform, and Titan Harbor completes operations like resource allocation and confirms the order successfully, Harbor generates a new Customer Task Smart Contract and anchors it to the Titan blockchain. Titan Harbor continuously monitors the service quality of the client task and updates the on-chain data in the smart contract.

The Customer Task Smart Contract includes, but is not limited to, the following datasets:

- Titan Client Identifier
- Titan Harbor Identifier
- Payment Strategy
- Ordered Resource Information

The payment strategy considers various options, including Pay-As-You-Go models and monthly or periodic payment methods.

The payment strategy also accounts for potential payment disputes, such as a client's unwillingness to pay due to service quality issues or requests for refunds. Such disputes often require arbitration by Titan DAO.

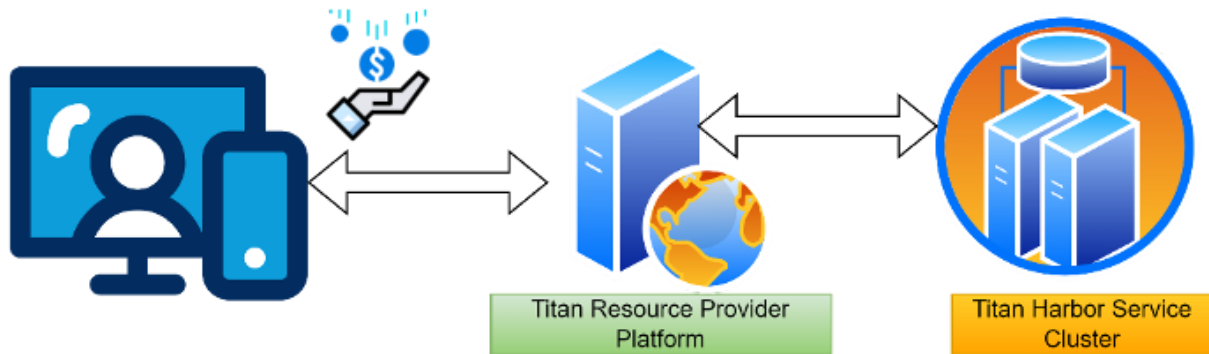
### **3.6.3. Harbor Governance Smart Contract**

The Harbor Governance Smart Contract is published by Titan DAO and serves to register all Harbors. It includes information such as the Harbor's Public Key, the set of incentive parameters for the idle resources managed by the Harbor, and the set of price parameters for the idle resources managed by the Harbor. Titan DAO determines the various parameter sets within the Harbor Governance Smart Contract.

## 3.7. Titan Interfaces

### 3.7.1. Resource Provider Interface

Titan's Resource Provider Service Platform, comprising web services and the Titan Resource Provider Data Service, is a service cluster designed for Titan resource providers.



It is a data platform managed by Titan DAO, designed to provide data services related to resource information, configuration operations, and resource earnings, among others.

The WEB service acts as the graphical user interface for the Titan Resource Provider Data Service. Titan resource providers interact with the data service through the WEB pages.

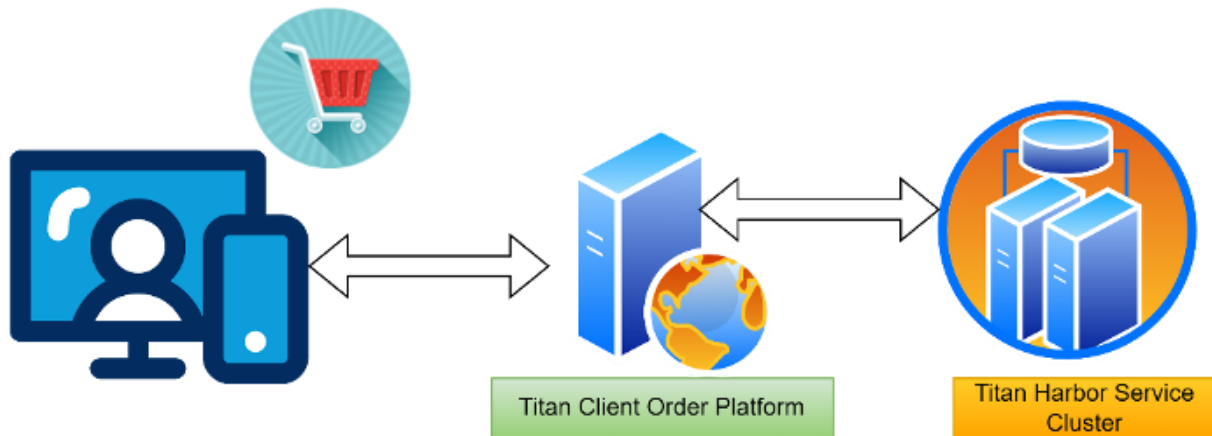
The Titan Resource Provider Data Service implements the data service protocols of the Titan Protocol, integrating resource provider data from various Titan Harbors within the ecosystem to offer consistent data services externally. For example, through this service, a Titan resource provider can browse and manage their idle resources distributed across different Titan Harbors from a unified WEB page, without needing to access each Harbor's provider data service individually.

Resource providers can view information about all their idle resources from the WEB page, including status, earnings, etc.

Resource providers can manage and configure all their idle resources from the WEB page, such as setting idle resource quota plans.

### 3.7.2. Resource Consumer Interface

Titan's Customer Service Platform, comprising WEB services and the Titan Customer Data Service, is a service cluster oriented towards Titan clients.



It is a comprehensive service platform managed by Titan DAO, serving as the customer entry point and value exchange hub for the Titan ecosystem. It aims to provide Titan ecosystem clients with services such as data browsing, order management, resource management, application deployment, etc.

Web service functionality:

- Navigate graphical user interface for the Titan Customer Data Service
- Browse and manage the rented idle resources
- Overview of data from Titan Harbors
- Overview of personal account data

The WEB service serves as the graphical user interface for the Titan Customer Data Service. Titan clients interact with the data service through the WEB pages.

The Titan Customer Data Service implements the data service protocols of the Titan Protocol, integrating customer data from various Titan Harbors within the ecosystem to provide consistent data services externally. For example, through the Titan Customer Data Service, a Titan client can browse and manage the idle resources they have purchased, distributed across different Titan Harbors, from a unified WEB page, without needing to access each Harbor's customer data service individually.

Titan clients can view an overview of data from Titan Harbors within the ecosystem on the webpage, including information on idle resource types, quantities, location

distribution, usage status, and more.

Titan clients can view data related to their accounts on the webpage, such as an overview of currently used resources, the operational status of various deployed projects, etc.

Titan clients can select and purchase the idle resources they need from the WEB page. Selection is based on multiple dimensions, such as the region where the idle resource is located, CPU type, bandwidth size, etc.

Titan clients can use the idle resources they have purchased from the WEB page, for example, by deploying projects onto these resources.

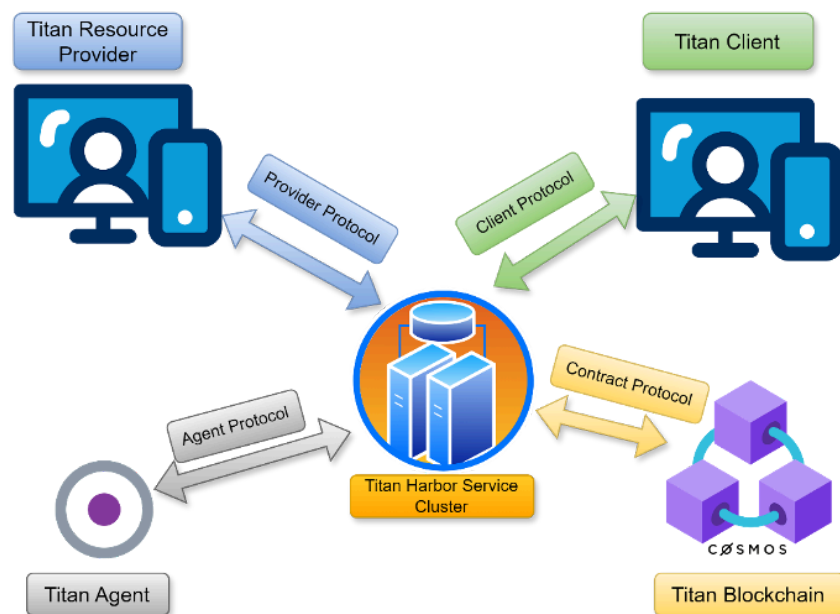
## 4. Protocol Coordination

Titan Network relies on a coordinated set of protocols to connect idle edge resources with user workloads in a secure, scalable, and efficient way. These protocols define how devices communicate, how resources are verified, and how tasks are scheduled across a decentralized network. This chapter focuses on three key technical areas:

- **Titan Protocol:** handles communication between agents, harbors, and clients
- **Resource Validation System:** ensures resources are real, available, and trustworthy
- **Resource Scheduling System:** matches user demand to the most suitable available resources. Together, these systems form the technical backbone of Titan's decentralized infrastructure.

### 4.1. Titan Protocol

The Titan ecosystem comprises multiple core systems, such as Titan Agent and Titan Harbor, mentioned in the "System Architecture" section. These systems need to communicate and collaborate. The Titan Protocol defines the collaboration protocols between them and establishes the norms for interaction among various systems. Due to space constraints, this paper only outlines several key sub-protocols of the Titan Protocol.



### **4.1.1. Idle Resource Access Protocol**

The Titan Protocol defines the set of protocols that idle resources must implement to access the Titan ecosystem. All types of Titan Agents need to implement this protocol set to connect.

This part primarily focuses on the communication and collaboration between Agent and Harbor, enabling the Agent to log into Titan Harbor, synchronize management data from Harbor, and respond to scheduling commands from Harbor.

Below are some core protocols listed:

#### **4.1.1.1. Agent Login**

After establishing a network connection with Harbor, the Agent sends a login request. The input set for the login request must include two core parameters: the unique ID of the Agent's device and a signature generated using the Agent's key. These two parameters are mandatory for Agent identity authentication. The device's unique ID and key are currently obtained from Titan's Resource Provider Service Platform.

#### **4.1.1.2. Agent and Harbor Configuration/Data Synchronization**

After logging in, the Agent needs to synchronize data with Harbor. For example, aligning the local quota plan with the one on Harbor. This step may take a relatively long time (e.g., one minute) if the configuration or data needs to be downloaded from Harbor.

#### **4.1.1.3. Agent State Transition and State Synchronization**

After completing configuration and data synchronization, the Agent can switch its state to 'ready' and synchronize this new state with Harbor.

#### **4.1.1.4. Keep-Alive**

The Agent needs to periodically send keep-alive packets to Harbor to indicate it is online. If Harbor does not receive keep-alive packets for an extended period, it assumes the Agent has gone offline.

#### **4.1.1.5. Resource Plane Formats**

Previous sections mentioned the need to standardize idle resources for use by Titan clients. Currently, the Titan Protocol specifies three forms of resource planes:

1. VM (Virtual machine) format.
2. K8S Container format.
3. Titan lightweight container (WASM/JS isolated runtime) format.

Among these, VMs are the heaviest, demanding the most resources, and are suitable for idle resources like home computers or data centers.

K8S Containers are relatively lightweight, suitable not only for home computers but also for devices with limited resources, such as tablets and router boxes.

Titan lightweight containers are the lightest, requiring the fewest resources, but can only run restricted WASM and JS programs.

As the Titan Protocol evolves, more resource plane formats will be added. For instance, the Titan ecosystem has recently added a simulated terminal format for Android phone Titan Agents, based on the TERMUX Library, capable of running various ARM CPU type console programs.

#### **4.1.1.6. Event Subscription**

After completing the login, the Agent needs to establish a persistent WebSocket connection with Harbor. The primary function of this long connection is to promptly receive and respond to event notifications from Harbor. For example, when the quota plan is updated, Harbor sends an event notification to the Agent via the long connection. The Agent then needs to update the quota plan from Harbor and adjust its local quota policy accordingly.

#### **4.1.1.7. Business Scheduling**

When the idle resource represented by the Agent is selected by Harbor to provide services for a client, Harbor issues a business scheduling event to the Agent based on the client's task configuration.

Upon receiving the scheduling event, the Agent prepares the necessary resources, retrieves the business configuration from Harbor, and then deploys the business locally according to the configuration. For example, Harbor might request the Agent to create a VM with resource specifications designated by Harbor; alternatively, Harbor might request the Agent to create containers and run a specific container image, again with resource configurations specified by Harbor.

The Agent also needs to periodically synchronize the business status with Harbor to maintain alignment.

Generally, CPU/memory/bandwidth resource configuration requirements are relatively easy for the Agent to handle. However, network environment requirements are more

complex, potentially requiring specific network types, such as NAT1 or those with public IP addresses. Therefore, adapting the network environment between Agent and Harbor requires additional toolsets, such as executing specific iptables commands or installing particular network plugins.

### **4.1.2. Resource Provider Service Protocol**

The Titan Protocol also defines a set of protocols serving resource providers. Through these protocols, resource providers can interact with Titan Harbor to manage their idle resources.

Currently, Titan's Resource Provider Service Platform uses this protocol set to communicate with Titan Harbors, presenting idle resource data and management operations to resource providers.

Below are some core protocols listed:

#### **4.1.2.1. Access Authorization**

Resource providers use this protocol to obtain access authorization for Titan's data services. Upon successful authorization, the provider receives an authorization token used for subsequent data access. The token contains permission sets, validity periods, etc. When the token is nearing expiration, the provider can request a refresh from Titan's data service to avoid repeating the authorization process.

#### **4.1.2.2. Bind or Unbind Idle Resource**

An idle resource has one and only one owner. A resource provider becomes the owner through a binding operation. Only the owner can configure the idle resource, modify its service status, etc. If the idle resource is already bound by another provider, duplicate binding is not allowed. A provider uses the unbinding operation to relinquish ownership of the resource. After successful unbinding, the idle resource returns to an ownerless state and no longer participates in Titan Harbor's scheduling. If the current state of the idle resource prevents unbinding (e.g., it is actively serving a Titan client), the provider must first change the resource's service status before proceeding with unbinding.

#### **4.1.2.3. Configure Idle Resource**

Resource providers use this protocol to configure their owned idle resources. Providers should select the correct region based on where the idle resource is located, as the region directly determines which regional Titan Harbor the resource should connect to. Cross-regional connections can affect scheduling (e.g., higher latency). Providers need to set the available quota and quota plan for the idle resource to prevent overutilization.

#### **4.1.2.4. View Idle Resource Information**

Through the protocol, Resource Providers can retrieve details about available idle resources. Beyond standard configuration data, this includes supplementary information provided by Titan Harbor. This supplementary data reveals how the idle resource is currently being scheduled, offering insights into its utilization, for example, the consumed quotas for different resource types and the number of active Titan client workloads. Another crucial piece of information is the reputation vector. This is a score assigned by Titan Harbor based on data from multiple dimensions, including the Titan validation system and the service quality delivered to Titan client tasks. The reputation vector directly impacts the idle resource's earnings. The Titan ecosystem community will provide guidance on how providers can improve their resources' reputation and earnings.

#### **4.1.3. Customer Service Protocol**

Titan Protocol also defines a set of protocols for Titan resource users, i.e., Titan clients. Through these protocols, clients can utilize the idle resources of the Titan ecosystem.

Currently, Titan's Customer Service Platform acts as the usage entry point for clients. This platform uses the protocol set to communicate with Titan Harbors, presenting Harbor capabilities to clients.

Below are some core protocols listed:

##### **4.1.3.1. Access Token**

Through this protocol, Titan clients gain authorization to access Titan's data services. Successful authorization grants the client an access token, which they use for all subsequent data access requests. The access token includes details like the scope of permissions (permission set) and its expiration time (validity period). To avoid repeating the full authorization flow, clients can request a token refresh from Titan's data service shortly before the current token expires.

##### **4.1.3.2. Query Harbor Idle Resource Overview**

Clients use this protocol to get an overview of the idle resources managed by a Titan Harbor, such as total resource types and quantities, amounts already used, and remaining available resources. This information helps clients order resource services.

##### **4.1.3.3. Query Owned Resource Information**

Clients use this protocol to query information about the resources they have ordered. The protocol supports querying overview information for all resources as well as

detailed information for individual resources. For VM-type resources, detailed information includes the VM's current state and runtime information like CPU, memory, storage, and network usage. For container-type resources, detailed information includes the number of containers, resource usage (CPU, memory, storage, network), and the running status of services deployed within the containers.

#### **4.1.3.4. Order VM Resource**

Clients use this protocol to order VM-type idle resource services from the Titan ecosystem. They need to specify the required VM characteristics, such as CPU, memory, storage, and network requirements, as well as potentially rules like requiring pledged resources or a minimum reputation vector level. Upon successful ordering, Titan Harbor generates a customer task smart contract for this service order and anchors it to the Titan blockchain.

#### **4.1.3.5. Order Container Resource**

Clients use this protocol to order container-type idle resource services. They need to specify container requirements, such as runtime type (K8S container, Titan lightweight container, trusted container, etc.), resource needs (CPU, memory, storage, network), and potentially requirements for staking or minimum reputation levels. Upon successful ordering, Titan Harbor generates a customer task smart contract for this service order and anchors it to the Titan blockchain.

#### **4.1.3.6. Query Customer Task Contract**

Clients use this protocol to request Titan's data service to retrieve basic information and the current status of their ordered service contracts from the Titan blockchain. It supports querying overview information for all service contracts as well as detailed information for a single service contract.

## 4.2. Resource Validation System



The primary responsibility of idle resources in the Titan ecosystem is to serve Titan clients, thereby earning income paid by those clients. As long as an idle resource correctly fulfills its duty by providing services meeting client expectations, Titan considers it honest. Therefore, Titan's validation system focuses on verifying whether Titan client services are operating as expected. To achieve this, Titan needs to build a validation network. This network involves the participation of many idle resources within the Titan ecosystem, aiming to verify whether services run by Titan clients within the ecosystem are functioning as anticipated.

### 4.2.1. Titan Validation Network

The Titan validation network involves collaboration among observers, Titan Harbor, and Titan clients.

Clients submit their service validation requirements to Titan Harbor, typically specified when ordering idle resources.

Titan Harbor formulates a validation plan based on the client's requirements and directs observers to execute it. The validation plan accompanies the lifecycle of the client's service, ensuring the service benefits from Titan's validation throughout its duration.

Observers are any idle resources within the Titan ecosystem that meet the validation requirements, including numerous mobile devices. Notably, some devices in the ecosystem, such as mobile phones, can opt out of resource scheduling and act solely

as observers in the validation network, executing validation plans and receiving corresponding work rewards.

When executing a validation plan, Titan Harbor randomly selects several observers and deploys a validation program onto their devices. The observers' devices run this program. The validation program requests the client's service to be validated, records the service's response speed and content, and sends these results back to Titan Harbor.

Titan Harbor collects validation results from multiple observers to assess whether the idle resource is running the client's service as expected. The assessment result impacts the idle resource's reputation vector and ultimately its earnings.

#### **4.2.2. AI-Generated Service Loads**

The Titan ecosystem also utilizes AI-generated service programs to simulate Titan client tasks. The random nature of AI generation helps prevent malicious resource providers from cheating specific validation programs. The similarity generated by AI also makes it harder for malicious providers to identify and cheat validation programs. These AI-generated service programs are deployed onto idle resources in a manner similar to Titan client services. Titan Harbor can then direct observers to validate them.

AI-generated service loads offer several benefits: Firstly, they do not depend on actual Titan client tasks, meaning an idle resource can be validated even if not currently scheduled for a client. Secondly, they are not controlled or influenced by Titan clients.

Crucially, these AI-generated service loads change continuously; for example, Titan Harbor might generate different loads daily.

The Titan ecosystem also leverages idle GPU resources within the ecosystem as part of the compute power for generating these AI service loads.

#### **4.2.3. Effectiveness of the Titan Validation System**

From the perspective of a malicious resource provider, distinguishing between genuine Titan client tasks and validation programs is difficult. As noted, the validation program might be the client task itself or an AI-generated service load. Therefore, identifying and tampering with validation programs to evade or deceive the validation process is a challenging task.

Observers are selected randomly, and they represent any participant in the Titan

ecosystem, making their number vast. Malicious actors cannot control a majority of observers.

From the perspective of a malicious Titan client, cheating to reduce payments is also tricky, as they cannot easily distinguish whether requests originate from their actual customers or Titan observers. Furthermore, Titan Harbor can prove the validity of idle resources by validating AI-generated service loads.

When formulating validation plans and directing validation execution, Titan Harbor must strive to avoid disrupting the normal operation of Titan client services, for instance, by not consuming excessive resources or taking too long.

#### **4.2.4. Titan's Validation Process and Protocol**

Titan prioritizes observers from that region during selection to avoid factors like intercontinental network latency affecting validation execution, focusing on different resource types and client task types that influence the specifics of the plan. Titan DAO is responsible for establishing a long-term, evolvable validation process and protocol, continuously improving validation quality, including details regarding observer rewards for executing validation plans, the composition of these rewards, calculation formulas, etc.

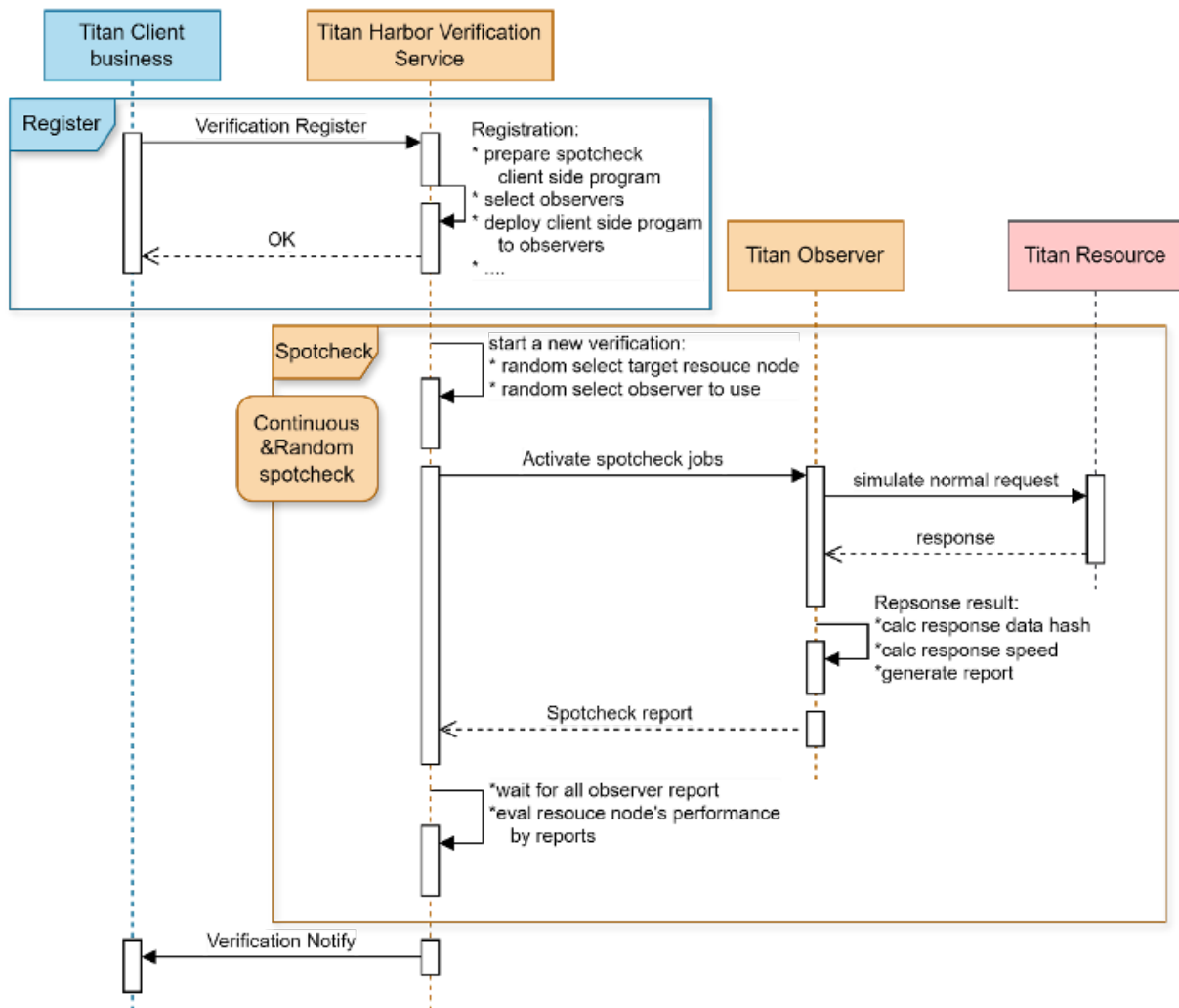
For further explanation, two validation cases are highlighted.

##### **4.2.4.1. Titan Validation System Application Case Study A**

Project A is a decentralized storage project that utilizes the Titan ecosystem's idle resources as its foundation. It stores content across multiple idle resources within Titan, utilizing techniques such as erasure coding to prevent data loss due to single-node failures.

Project A leverages Titan's validation system for periodic validation of its storage service. This process validates both the effectiveness of the idle resources and the effectiveness of Project A's service. For instance, it can detect service quality degradation due to idle resource cheating and service failures caused by storage service program crashes.

The validation flow is as follows:



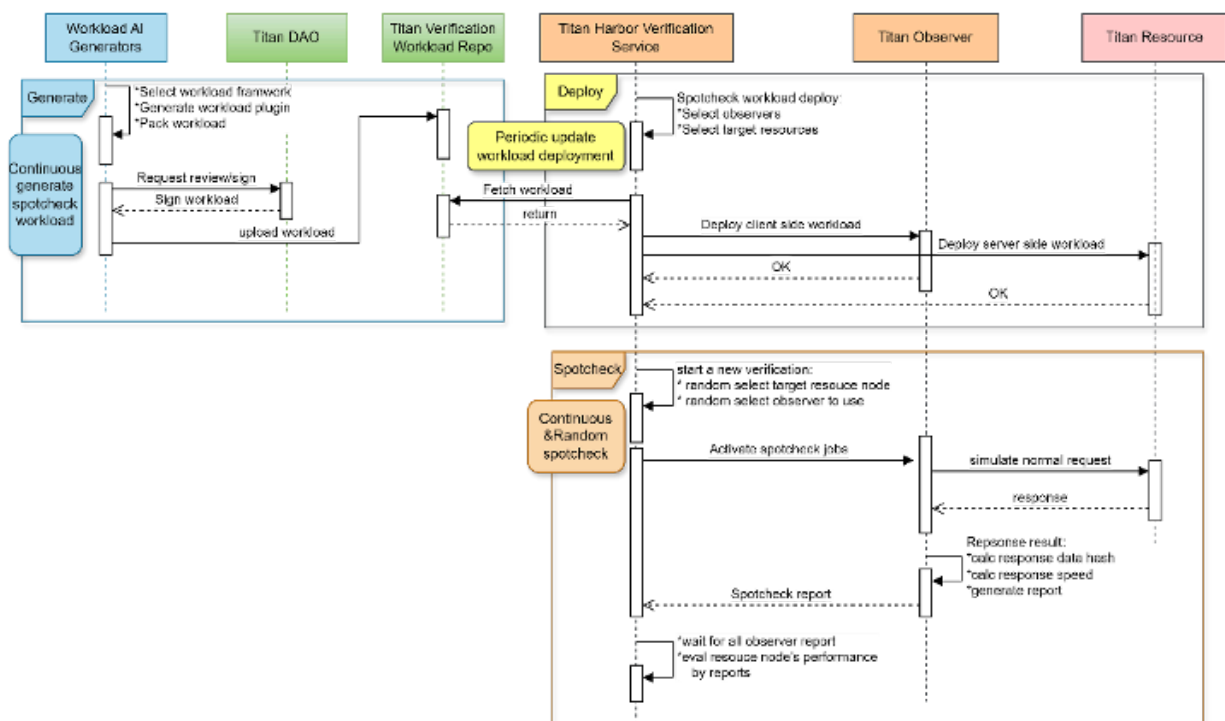
1. Titan Harbor executes validation according to plan. It starts by requesting Project A's server to obtain the set of nodes to be validated, along with the names and hashes of the stored content.
2. Harbor plans validation pairings. For example, storage node set X contains 10 nodes storing identical content. Harbor selects observer set Y containing 100 nodes. Observer set Y is responsible for validating the storage node set X.
3. Within observer set Y, every 10 observer nodes randomly check one node from storage node set X.
4. Observer nodes simulate real clients by requesting the storage node. The storage node cannot distinguish requests from observers versus actual clients. The storage node sends the specified content, completing the request.
5. Upon reception, the observer node calculates the content hash and measures the bandwidth, sending these results to Titan Harbor.

6. After validation completion, Harbor collects the results, assesses the service validity and quality of the storage nodes based on the results, scores the idle resources of the storage nodes, and updates their reputation vectors.
7. Harbor notifies Project A of the validation results. Based on the results (e.g., identifying failed nodes), Project A needs to make adjustments.
8. Harbor records the contributions of the observers for subsequent rewarding.

#### 4.2.4.2. Titan Validation System Application Case Study B

In the previous case, the project needed to follow Titan's validation process and protocol, cooperating with the validation system. The advantage is verifying both resource and service effectiveness simultaneously; the disadvantage is the extra workload for the project.

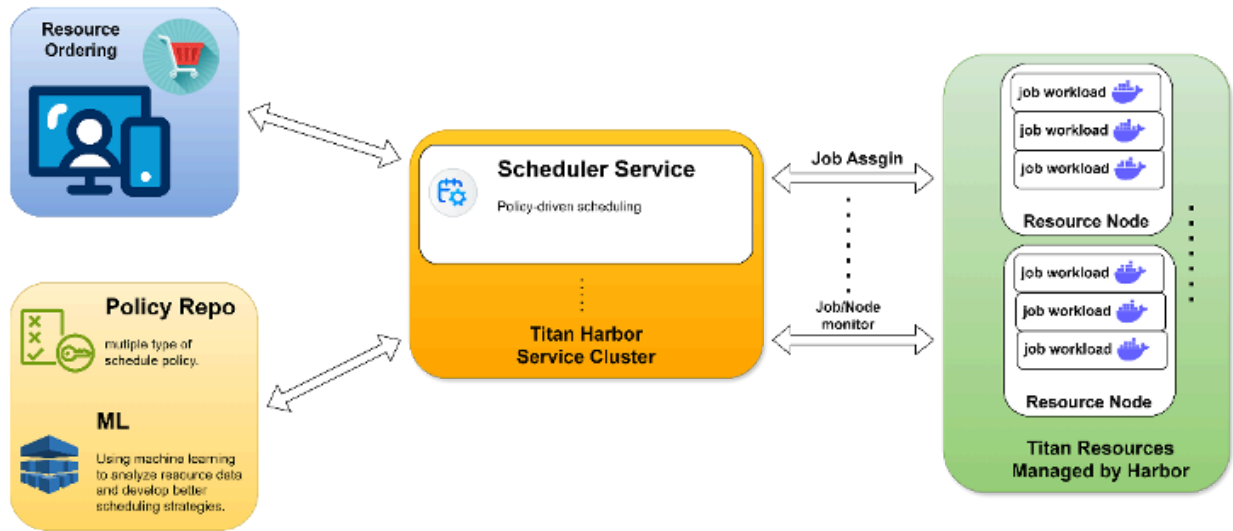
An alternative approach is for the project not to participate. The Titan ecosystem completes the validation process independently, ensuring only resource effectiveness is considered. The flow is as follows:



1. Titan Harbor executes validation according to plan. It starts by deploying an AI-generated service load onto the idle resource node being validated.
2. The idle resource node cannot distinguish whether the service load is AI-generated for validation or a real Titan client's service load.

3. Harbor plans validation pairings, selecting observer set  $Y$  (e.g., 10 nodes) responsible for validating this idle resource node.
4. Observer nodes simulate real clients, requesting the AI-generated service load on the idle resource node, which completes the request and sends back a response.
5. Upon receiving the response, observer nodes calculate the required validation results.
6. After validation completion, Harbor collects results, assesses the resource effectiveness and quality of the idle resource based on the results, scores the idle resource, and updates its reputation vector.
7. Harbor records the observers' contributions for subsequent rewarding.

## 4.3. Resource Scheduling System



Idle resources in the Titan ecosystem are diverse. Scheduling must consider not only type and capacity but also stability.

The stability of idle resources from home devices is influenced by multiple factors: the inherent lower stability of consumer-grade devices, less stable home power supplies compared to IDCs, and potential variations in resource availability at different times.

If a Titan client orders VM resources, the provided idle resources must be sufficiently stable; otherwise, the VM might fail to meet client requirements. Therefore, when scheduling resources for VM-type client tasks, the system selects IDC server-level equipment or home devices with substantial stakes and high reputation vectors.

If a client orders container resources, for stateful containers that are difficult to migrate, the system selects highly stable devices. For easily migratable stateless containers, the system can choose less stable devices, as issues can be quickly resolved by finding replacements from the available pool and completing the migration rapidly.

When numerous less stable idle resources (e.g., home devices) serve Titan clients, the scheduling system must continuously monitor their availability status. When they become unavailable, the system must promptly find replacements from the available pool.

### 4.3.1. Fairness of Scheduling Policies

Every Titan client hopes to select the highest quality idle resources. A simple strategy

might allocate the best available resources from the pool for each task. However, this leads to issues: later clients tend to receive lower-quality resources than earlier ones, and lower-quality resources might remain unused for extended periods.

To meet client quality requirements while ensuring scheduling fairness, the current system approach involves classifying idle resources into several tiers based on quality upper and lower bounds. Resources within the same tier receive equal scheduling opportunities. When scheduling for a client task, the system selects resources from each tier's pool in specific proportions.

For example, suppose a client orders container resources requiring 30 containers with 99% reliability. The scheduling system, targeting 99% overall reliability, might select 10 resources from the top tier (each >99% reliable), 30 (3x10) from the second tier (each >90% reliable), and 40 (4x10) from the third tier (each >80% reliable). Statistically, this is equivalent to providing the client with 30 resources, achieving 99% reliability.

The idle resource scheduling system will implement various scheduling policies to better match different client task requirements. However, fairness must be a consideration in every policy.

The system generates detailed logs for each scheduling decision, stored as individual files. The hash of each file is anchored to the Titan blockchain. The Titan community can retrieve the log file using its hash from Titan Harbor for auditing and traceability.

Community review and analysis of scheduling logs enable supervision and improvement of the scheduling system's policies.

#### **4.3.2. Specificity of Scheduling Policies**

Real-world Titan client tasks may have unique scheduling requirements not met by the system's existing automated policies. In such cases, a pathway for human intervention is necessary.

Therefore, the resource scheduling system also supports a hybrid mode, combining semi-automation with manual intervention.

Once the Titan ecosystem community gains sufficient understanding of a specific type of special scheduling requirement, new policies can be designed and implemented for it, enabling fully automated handling of such scheduling needs in the future.

## 5. Titan Network Use Cases

The flexibility and scalability of the Titan Network enable it to support a wide range of real-world applications across Web 2.0 and Web 3.0. By aggregating idle edge resources into a people-powered, programmable cloud, Titan can meet diverse compute, bandwidth, and localization needs, often at lower cost and with greater geographic reach than traditional infrastructure. This chapter highlights practical use cases where Titan's architecture delivers clear value, from edge computing and CDN to AI agents, regional testing, and privacy-preserving data services.

### 5.1. Smart Edge CDN

Services like video streaming, online education, and game downloads rely on Content Delivery Networks (CDNs) to reduce access latency. However, traditional CDN providers have limited node coverage, leading to high costs in remote areas. Furthermore, sudden traffic surges (e.g., sports live streams) can easily overload central nodes.

Enterprises can leverage the Titan ecosystem to aggregate vast numbers of distributed idle bandwidth resources (e.g., home bandwidth, idle IDC bandwidth) to build intelligent edge CDNs. Compared to traditional centralized CDNs, smart edge CDNs have the following advantages:

- **Cost Reduction:** Built on Titan's cost-effective idle resources, significantly lowering content distribution costs for enterprises.
- **Flexibility & Intelligence:** Titan's resource standardization planes (containers, VMs) allow flexible deployment and execution of enterprise-specific scheduling and distribution logic, making content delivery more timely and intelligent. For example, it can perceive real-time bandwidth pressure and hot content on the enterprise's central service cluster, autonomously caching and offloading distribution for popular content.
- **Proximity & Performance:** Edge CDNs provide "last-mile" content delivery, closer to users than traditional centralized CDNs. This can result in superior network latency metrics, enhancing user experience.

Enterprises can utilize smart edge CDNs as a supplement to their centralized CDNs, thereby reducing costs and enhancing distribution quality.

Companies with an existing smart edge CDN application can directly deploy it onto the extensive network of geographically distributed idle bandwidth resources aggregated

within the Titan ecosystem. Conversely, businesses without their own smart edge CDN application can leverage the development framework provided by Titan. This enables them to create their own smart edge CDN solution by simply adding a minimal amount of code.

## **5.2. Edge Computing**

Scenarios like the Internet of Things (IoT), smart wearables, and autonomous driving generate massive demand for edge computing. Traditional cloud computing centers, due to physical distance and high transmission latency, struggle to meet millisecond-level response requirements. Additionally, transmitting all data back to the cloud for processing incurs substantial bandwidth and compute costs.

Enterprises can build their dedicated edge computing networks by deploying compute services onto the container/VM resource planes of Titan's aggregated multi-regional, near-user idle resources. Requests from IoT devices, user data processing tasks, etc., can be routed to this Titan-based edge network for handling.

An enterprise-specific edge computing network built on Titan not only reduces the enterprise's data center costs (bandwidth, compute) but also offers lower network latency and faster service responses due to its proximity to the user side, thereby improving the overall user experience.

## **5.3. IP Proxy**

Many applications today restrict user access based on their IP address, effectively creating geographical limitations. When users need to bypass these geo-blocks, using an IP proxy is a common method.

The Titan ecosystem aggregates idle resources from around the globe. These resources adhere to the Titan Protocol's resource plane specifications, standardizing them to run various client applications, particularly Titan's lightweight containers. A key feature is that these containers are loaded onto the idle resources only when actively in use.

Titan clients can run their IP proxy applications within these lightweight containers. When accessing a target application, the connection is routed through this proxy, leveraging the IP address and network bandwidth of the underlying Titan idle resource. This effectively avoids the target application's IP-based restrictions.

Because Titan's lightweight containers utilize idle resources and operate on a pay-per-use duration model, they offer significant cost savings. Clients avoid the expense of a full-time IDC VPS, which typically requires payment for 24/7 availability, even if they only need proxy access for short periods, like an hour a day. This makes Titan's lightweight containers a highly cost-effective solution.

Furthermore, the Titan ecosystem community often shares templates for common IP proxy applications. This allows clients to simply select idle resources in their desired geographical region and deploy a proxy application with a single click, making it ready for immediate use.

## **5.4. Web Crawler Applications**

Web crawling needs can be met by deploying distributed crawlers based on Titan's multi-regional architecture, which utilizes numerous idle resources.

Traditional crawlers based on IDC servers face limitations in bandwidth and compute capacity. Moreover, their IP address ranges often belong to specific IDC providers and are easily blocked entirely by target websites.

Titan's idle resources are multi-regional, close to the user side, and numerous. These characteristics enable distributed crawlers built on Titan to possess greater bandwidth, compute, and storage capabilities. Their user-proximate IP addresses are less likely to be blocked by target websites.

Distributed crawlers built on Titan, after data scraping, can also utilize the idle resources' compute power for data preprocessing, finally compressing and orderly transmitting the results back to the Titan client's central service for further processing.

Titan DAO requires KYC processes and reviews for distributed crawler applications to prevent malicious DDoS attack programs from masquerading as crawlers to attack target websites.

## **5.5. Resource Infrastructure for Various Web2 and Web3 Projects**

The vast pool of diverse, multi-regional idle resources aggregated by the Titan ecosystem can serve as a foundational resource for various blockchain projects.

For example, Actively Validated Services (AVS) type blockchain projects require

numerous multi-regional nodes for validating service effectiveness. They can utilize Titan's abundant multi-regional idle resources as validation nodes.

Notably, within Titan, Harbor's validation system uses idle resources within the ecosystem as validators to verify the effectiveness of target idle resources.

Another example includes DePIN projects, such as those requiring GPU resources. They can leverage the large number of idle GPUs aggregated by Titan as their resource pool. Such a DePIN project simply needs to deploy its program onto Titan's idle resources to access these GPUs.

## **5.6. Regional Testing & Active Monitoring**

Ensuring an excellent user access experience across multiple target regions is crucial for enterprise network application services. Passively waiting for user feedback on access issues often means that the negative impact has already occurred. Therefore, proactive monitoring to detect problems before users do is essential.

The Titan ecosystem aggregates idle resources from various regions, including numerous devices naturally close to the user side, like home computers and personal phones. Enterprises can deploy their monitoring programs onto these near-user devices within Titan. These programs run periodically, monitoring the access experience of the enterprise's application services and sending aggregated result data back to the enterprise's servers. This data helps enterprises understand service performance in each region and identify issues promptly.

The multi-regional idle resources on Titan provide a cost-effective and vast resource base for enterprises to build multi-regional, round-the-clock active monitoring systems. Enterprises can further reduce costs by implementing monitoring programs as Titan lightweight container applications, benefiting from pay-per-use pricing.

Similarly, enterprises can utilize Titan's multi-regional idle resources to build their own multi-regional intelligent testing systems. These systems can run enterprise-specific test programs, simulating user interactions from multiple regions to test application services.

## **5.7. Edge GPU Computing and AI Agents**

Many home users purchase high-end GPUs for optimal gaming experiences. However, gamers might spend most of their day studying or working, with only a fraction devoted to gaming. Even after gaming sessions, these GPUs often remain idle.

These idle GPUs can be securely aggregated into the Titan ecosystem via the Titan Agent.

Enterprises can leverage the numerous multi-regional GPU resources aggregated in Titan to build edge GPU computing capabilities, routing AI inference requests to this edge network. For instance, some state-of-the-art AI models, after significant optimization by the open-source community, can already run inference tasks on GPUs in home computers.

An enterprise-specific edge GPU computing network built on Titan reduces data center costs (network bandwidth, GPU resources). Furthermore, Titan's idle GPU resources are closer to users, offering lower latency and faster responses, thus enhancing the overall user experience.

## **5.8. Data Sovereign Storage**

Traditional distributed data storage solutions are mostly enterprise-deployed and managed facilities located in data centers (e.g., storage services from Cloud providers like Google and AWS). These services are controlled by the enterprises, and users may face risks of censorship or service termination in the future.

Even some blockchain-based storage services have miners predominantly running in IDCs, facing similar risks of censorship and shutdown (e.g., IDCs terminating services for miners).

Titan ecosystem aggregates vast multi-regional storage resources. While some are in IDCs, the majority are from homes and individuals, making them inherently resistant to censorship and centrally mandated shutdowns – a unique characteristic of the Titan ecosystem.

The industry and community can build censorship-resistant storage services with data sovereignty, leveraging Titan's extensive idle storage resources.

Data sovereign storage built on Titan is not just about storage itself. Combined with Titan's multi-regional, abundant idle network bandwidth resources, it provides clients with truly usable and effective data sovereign storage services.

This differs from some existing blockchain storage projects, which often focus solely on storage without providing adequate network bandwidth, making them less practical for real client needs.

## 5.9. Content and compute localization

As governments around the world implement stricter data residency and infrastructure localization policies, enterprises and developers are increasingly required to ensure that content and compute workloads remain within specific geographic or jurisdictional boundaries. Traditional cloud providers often lack the granularity, flexibility, or affordability to meet these evolving localization demands, especially in regions without nearby data centers.

Titan Network, with its globally distributed pool of idle resources—many of which are located in residential or non-IDC environments—offers a unique solution. By enabling users to deploy workloads directly onto resources within a targeted region or country, Titan allows compliance with data sovereignty laws while maintaining low latency and cost efficiency.

For example, an enterprise operating in Brazil can selectively deploy content delivery, compute, or monitoring tasks to Titan nodes physically located within those countries. This localized execution model ensures legal compliance, enhances performance for regional users, and reduces cross-border data transfer costs.

## **6. Future Development**

### **6.1. Greater Public Transparency**

One of Titan's core capabilities is data openness and transparency, built upon the Titan blockchain.

In the current phase, Titan primarily focuses on standardizing the resource plane of idle resources, their management, and client tasks. Specifically, to quickly validate the design and implementation of core systems like Titan Agent and Harbor, and to verify the value of idle resources within the ecosystem and establish the value conversion chain, Titan has selected a few typical client business types (e.g., PCDN). These businesses have been successfully implemented within the Titan ecosystem. The Titan community identifies idle resources that meet these business needs, standardizes them using Titan Agent for management by Titan Harbor, which then schedules these resources for client tasks, providing cost-effective resource services.

In the future, Titan needs to analyze each process, putting key steps and results on-chain, including but not limited to:

- Idle resource validation process.
- Resource scheduling decision-making process.
- Business earnings distribution process.
- Client task service quality monitoring.

Some code currently implemented within Titan Harbor services may need to be reimplemented as smart contracts to make the entire process publicly transparent. Titan envisions that, supported by a future Titan blockchain with extremely high TPS, all processes within Titan Harbor could be implemented in smart contracts, and all result data anchored via smart contracts. Ultimately, the core systems of the entire Titan ecosystem would operate in a truly decentralized and publicly transparent manner.

### **6.2. Better Titan Blockchain Core**

The Titan ecosystem will require more data on-chain in the future, demanding increasingly higher TPS from the Titan blockchain. The current Cosmos-based blockchain, due to its consensus protocol and block propagation/synchronization mechanisms, will struggle to meet future TPS needs.

The Titan ecosystem needs to introduce more efficient consensus protocols and block

propagation mechanisms. For example, adopting a DAG (Directed Acyclic Graph)-based consensus protocol and restructuring the current blockchain base could enable a new Titan blockchain to meet the high TPS demands.

It is anticipated that as the Titan ecosystem develops and matures, many Web3 projects will deploy onto it. These projects will also require blockchain services, and a Titan blockchain with ultra-high TPS would be their ideal choice. Titan can build dedicated smart contract SDKs and blockchain access SDKs for each type of Web3 project, providing one-stop, high-quality services.

For instance, for Web3 games, Titan could offer not only compute, storage, and bandwidth resources but also oracle services based on the Titan blockchain.

## 6.3. Preventing Resource Misuse for Malicious Activities

Malicious actors exploiting Titan's abundant and cost-effective idle resources to launch DDoS attacks against targets would cause immense harm to the ecosystem.

Titan can address this from two perspectives in the future:

Firstly, effective remediation measures are needed. When the Titan community receives a complaint, it must respond immediately. Titan Harbor must be able to quickly shut down the malicious actor's workload, halt the attack, and disable their Titan client account. This requires Harbor to rapidly identify the malicious workload, possibly by analyzing network logs from idle resources during a specific period to see if the workload accessed the target.

Secondly, preventative measures need establishment and improvement, such as:

- **Access Blacklist:** Titan establishes a blacklist of sensitive, protected targets. Idle resources are forbidden from accessing targets on this list.
- **KYC for Large Orders:** Client business requirements involving ordering large quantities of idle resources must undergo Titan DAO's KYC process and receive approval before ordering is permitted.
- **Usage Restrictions for New and Unverified Clients:** New clients and those without Titan DAO KYC approval are subject to restrictions on resource usage scenarios, particularly when employing an outbound whitelist mechanism, which allows access only to declared targets. If Harbor detects numerous workloads declaring the same access target, it may suspect a potential attack target and take

action to prevent incidents. As Titan grows, it will inevitably attract more malicious actors attempting illicit activities with its resources. The fight against misuse will be continuous, requiring ongoing improvement of countermeasures to safeguard the ecosystem's development.

## **6.4. Titan Client Data Security**

Titan Protocol's resource standardization (containers, VMs, lightweight containers) isolates Titan client programs, ensuring the security of the resource provider's idle resources and other assets. This protection is vital for providers, giving them the confidence to aggregate resources into Titan for earning income.

However, from the client's perspective, concerns about external intrusion, theft, or tampering of their data are also valid. For example, if a client's program runs in a container on a provider's idle home computer, container isolation protects the provider's computer from any potential issues. However, if the computer hosts malware, that malware may potentially infiltrate the container environment (e.g., via debugging) and compromise the client's program and data.

Currently, Titan mitigates client concerns by aggregating idle resources with TEE (Trusted Execution Environment) capabilities, offering trusted container and VM environments for data-sensitive clients. The drawback of TEE is significantly reduced performance for programs executed within it, impacting service quality. To maintain quality, clients might need to order many such trusted environments.

Another potential approach is for Titan to release an open-source hypervisor. A transparent hypervisor, aided by trusted components like TPM on the computer, could perform self-integrity checks upon boot and then load the host OS of the provider's idle resource. Since the hypervisor has the highest privilege, ensuring its trustworthiness allows it, through technical means, to guarantee the trustworthiness of the containers and VMs it manages.

## **6.5. Aggregating More Types of Idle Resources**

For a considerable period, Titan will primarily aggregate device-based idle resources like home computers, routers, tablets, phones, IDC servers, etc. These devices possess compute, storage, and network resources. Aggregated, they form a massive capability cluster meeting diverse client needs, serving both individual and enterprise-level tasks cost-effectively.

In the future, Titan could aggregate novel device-based resources, such as temperature/humidity sensors, air quality sensors, etc. Distributed globally, these sensors could provide rich data services to the ecosystem.

Beyond devices, Titan could aggregate other categories of idle resources, such as human resources. Leveraging its blockchain base, Titan could build transparent value transfer channels for idle human capital and related demands.

## **6.6. Supporting Rich Transaction Modes**

Currently, Titan clients order resources via the Customer Service Platform. In the future, Titan will support more transaction modes, such as tendering and bidding. Clients could publish tenders via the platform, and multiple Titan Harbors could submit bids.

Implementing new transaction modes and supporting new resource allocation needs can attract more clients to the Titan ecosystem.

## **6.7. Landing More Types of Customer Businesses**

The Titan ecosystem needs to support client businesses with more diverse requirement types in the future. These businesses will place new demands on Titan's idle resources. For example, Web3 games require integrated use of compute, storage, and network bandwidth in a data center cluster format. These resources require high-speed collaboration, which is achievable only via low-latency, high-bandwidth LAN connections within the same facility. Titan must be able to pool and manage unused resources within the same IDC to meet these needs.

Titan must also consider scenarios where clients want Titan resources integrated into their private Cloud management systems. For instance, a client orders idle resources from Titan, and these resources join the client's own K8S container orchestration system as K8S PODs, rather than forcing the client to use Titan's provided system.

Titan should provide resource ordering and management SDKs for client businesses in the future. Clients could use these SDKs to directly order and manage Titan resources without logging into the service platform.

Titan needs to proactively capture and explore business needs, collecting requirement information from the community and various industries. It should actively build specific SDKs, ordering and deployment templates, and other "demand adhesives" to provide the most convenient pathway for new businesses to utilize Titan's idle resources.

An example is the smart edge CDN development framework built by Titan, which enables enterprises with content distribution needs to implement their own edge CDN program with minimal code additions on top of this framework. They can then deploy it on Titan's aggregated resources and build their dedicated edge CDN network.

## 7. Conclusion

Titan Network is a decentralized resource platform that bridges the global imbalance of compute, storage, and network capacity by matching the world's idle resources with areas of high demand. By crowdsourcing heterogeneous resources from everyday devices into a unified cloud, Titan delivers web services that create an alternative or supplementary solution to traditional providers like AWS and Google Cloud. This uniquely positions Titan to lead the next wave of decentralized physical infrastructure networks (DePIN), effectively transforming unused capacity into valuable infrastructure at a global scale.

At the core of Titan's architecture is the Titan Agent, an open and lightweight software client that standardizes diverse resources via open protocols and virtualization technologies. Through containerization and virtual machines, Titan abstracts away hardware differences, isolating workloads on each device and enforcing a uniform Titan Protocol across the network. This approach provides strong security isolation and a consistent interface for services, ensuring that any contributed compute, bandwidth, or storage can be utilized reliably regardless of its origin.

The Titan ecosystem is structured as a modular, multi-tier network that efficiently coordinates resources at scale. Titan Agents run on user devices (from smartphones and PCs to servers), contributing idle capacity at the network edge, while "Harbors coordinate and manage scheduling across the system. On top of this foundation, Titan's service platforms deliver cloud-like services – from decentralized content delivery to AI inference workloads – by harnessing the pooled resources. This layered design allows each component to operate independently yet harmoniously, resulting in a flexible and efficient decentralized cloud infrastructure.

Titan also directly addresses the key challenges of trust, security, and fairness in a distributed network. A blockchain-based governance layer underpins all coordination: resource contributions and usage are validated and tracked on Titan's ledger, and smart contracts automatically handle task assignment and reward distribution. This approach ensures critical functions like proof of resource availability, workload scheduling, and result verification are handled transparently without centralized control. Notably, Titan's virtualization sandbox model reinforces security by strictly isolating client workloads from the host devices, and all transactions and metrics are recorded on an immutable ledger for auditability. As a result, resource providers are compensated according to the actual usage of their hardware, while clients pay only for what they consume – an economic model that guarantees fair monetization for contributors and cost-efficient

access for consumers (with Titan's deployments already showing up to 75% cost savings for enterprise clients).

In summary, Titan offers a practical, scalable and transparent foundation for turning idle capacity into usable infrastructure, positioning itself as a prime DePIN infrastructure layer for Web3 and beyond. It addresses the structural contradictions of resource supply and demand in the new era and redefines production relations in the digital age, transforming "idle" into "productivity." It empowers individuals and enterprises to shift from passive resource consumers to co-creators of value.