Smart Layer: A Decentralized Integration Protocol for the Next-Generation Web

1st Septempber, 2023

Abstract

In the evolving landscape of the decentralized web, the need for a robust integration layer that bridges the gap between siloëd services and tokenized assets has become paramount. Smart Layer emerges as a decentralized network designed to facilitate the next generation of web use-cases, addressing the inherent limitations of the current web. This whitepaper delves into the architecture, design, and potential of Smart Layer and its token, aiming to revolutionize the way we perceive and interact with the web.

Introduction

The evolution of the web has been marked by periods of rapid innovation, leading to an era of unprecedented connectivity and information exchange. Observations of tech giants like Google have underscored the potential benefits of integrated services. However, despite these advancements, the broader web landscape has remained fragmented. Centralized entities have emerged as dominant forces, creating isolated ecosystems that limit true integration. While blockchain technology introduced a new paradigm with its emphasis on decentralization and trustless transactions, its primary focus has been on asset tokenization.

In this context, Smart Layer emerges—a decentralized protocol that aspires to redefine the web's architecture. Envisioned as an integration bus, Smart Layer facilitates seamless interactions between diverse services, akin to how websites today leverage platforms like Google for a myriad of functionalities. Beyond acting as a mere bridge, Smart Layer introduces the concept of Smart Tokens. These are tokenized digital rights and products/services that can be seamlessly integrated across various web use-cases, transcending the limitations of centralized systems and leveraging the strengths of blockchain. This protocol is designed to function as a distributed network, serving as the backbone for the next generation of the web.

Integral to the functioning of Smart Tokens within the Smart Layer is TokenScript, an OASIS standard work in progress. While Smart Layer provides the infrastructure and environment, TokenScript outlines how these smart tokens should be packaged, distributed, referenced, composed and executed. It's a complementary framework that ensures smart tokens operate optimally within the defined parameters of privacy, secure storage, and cost accounting.

Problem Statement

The modern web, for all its advancements, grapples with a fundamental challenge: the 'Limit of 3' problem. Despite the web's inherent nature of integration, most websites today are confined to three primary types of integrations: login, social media posting, and checkout. This limitation stems from a combination of privacy concerns, integration costs, and the absence of a secure, privacy-preserving mechanism to facilitate expansive integration.

Moreover, the web's siloëd nature has led to fragmented user experiences. Consider the example of an airplane ticket. In the current web paradigm, this ticket, while representing a token of value within its issuing platform, remains isolated. The potential for this ticket to integrate with other systems – updating travel statuses on social media, guiding users via mapping services, or communicating flight changes to hotel booking systems – remains largely untapped. Such straightforward integrations, though long overdue, are hindered by the web's compartmentalized structure, where centralized entities offer piecemeal solutions.

This state of affairs underscores the need for a paradigm shift, a move towards a more dynamic and interconnected web ecosystem. The following sections will delve into how Smart Layer addresses these challenges, laying the foundation for the next-generation web.

Protocol Requirements

The design and functionality of Smart Layer are driven by specific protocol requirements, tailored to enable the unique capabilities of smart tokens. These requirements are not merely a reflection of standard practices for distributed networks but are intricately linked to the challenges and goals of the Smart Layer ecosystem. The key areas of focus include:

- **Serviceability**: This encompasses continuous uptime, redundancy, and load balancing. While mature industrial technology can meet these requirements, their application within Smart Layer is influenced by other interconnected requirements.
- **Privacy and Security**: Smart tokens distribute their logic between user agents (like decryption of sensitive data) and server-side logic (such as triggers set within the tokens). This paper primarily addresses the server-side logic executed by the Smart Layer network.
- **Token Lifecycle Management**: This pertains to the management of smart tokens throughout their existence. Considerations include the duration a flight ticket smart token resides on a node and the mechanisms to reinstate tokens that are in dormant states, such as a car-insurance token awaiting activation.
- **Inter-node Collaboration**: Nodes within the Smart Layer network are expected to work together to facilitate specific token functions. For example, a smart car token's status could be influenced by its registration, insurance, and maintenance tokens, potentially managed on different nodes.

Integrations expect smooth interactions between nodes, allowing them to concentrate on the capabilities provided by smart tokens rather than managing their intricacies.

- **Incentive Structure**: The cost of operating a token is typically borne by the integration. For instance, if a health token is used by a website to optimize a user's shopping list, the e-commerce platform incurs the cost. However, token issuers play a pivotal role in ensuring the availability and operability of their tokens on the network. They must incentivize the network to maintain the token's availability, even as the actual operation costs are met by integration points.

It's essential to differentiate these requirements from those of TokenScript. While TokenScript outlines how smart tokens should be packaged, distributed, and executed, Smart Layer emphasizes the real-time execution of the server-side components of these scripts. This ensures they function optimally within the defined parameters of privacy, secure storage, and cost accounting. The features specific to TokenScript, currently under standardization in collaboration with OASIS, are not covered in this document.

Smart Layer Architecture

Smart Layer's architecture is rooted in mature protocols and algorithms that have been proven effective in distributed systems, including blockchain itself, distributed hash table, load balancing and service level objective monitoring and the use of Merkle tree in data integrity verification. These foundational technologies provide the basis for building Smart Layer as a robust, decentralized network tailored for token operations. The innovation primarily stems from the creation of an integration service platform and a conducive environment for smart tokens, encouraging existing web infrastructure to transition towards a token-centric architecture.

The primary serviceability requirement determined that the network can't be built like a blockchain, where consensus serve to determine truth; instead, services must be monitored and load balanced in real time. This leads to the need of anchoring nodes.

Anchoring Nodes and Distributed Smart Token Instances

Smart Layer's emphasis on serviceability sets it apart from traditional blockchains that lean heavily on consensus mechanisms. This focus demands real-time monitoring and load balancing, which is where anchoring nodes come into play. These nodes serve as the network's guardians, ensuring consistent service availability and stepping in for pivotal operations. The Distributed Hash Table (DHT), shared among these anchoring nodes, is instrumental in determining which node is responsible for a specific smart token instance. This decentralized approach not only mitigates potential attacks that might arise from matching node IDs with token IDs but also guarantees prompt responses to integration

queries.

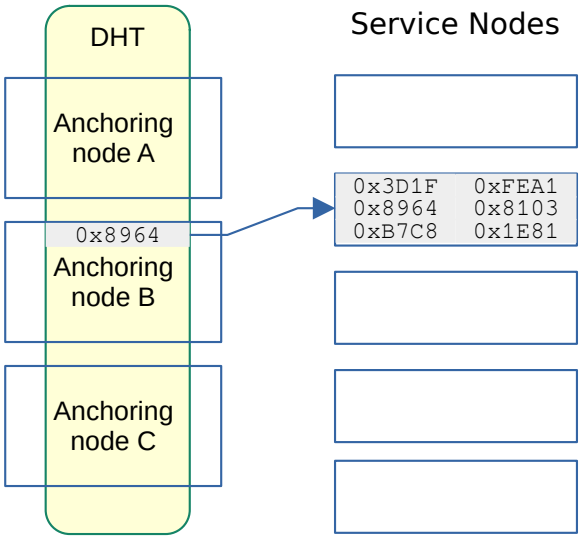

Figure 1: Mapping Token ID to its service node

Token Status Propagation and Execution

Smart tokens, as envisioned in the Smart Layer network, have a dynamic status that can be influenced by various factors. These factors can range from attestations to smart contract updates and node messages. While some of these updates are deterministic, others can be non-deterministic, leading to potential complexities in the network's operation.

Deterministic vs. Non-Deterministic Status A deterministic status update is one that, given the same input, will always produce the same outcome. For instance, with a flight ticket as a smart token, a flight delay leading to an automatic lounge access reward for a passenger is deterministic. However, not all updates are so straightforward. Consider the scenario of the same smart token rebooking a hotel through a web API. The outcome might be a successful booking attestation, a timeout, or even a server-side error. Such non-deterministic outcomes present challenges, especially when integrating with existing web2 systems. While there are pure blockchain based solutions that completely do away the status branching, such as rebooking through hotel smart contracts backed by a hotel's precommitment, the integration of web2 systems with smart contract-enabled platforms will remain a challenge for the foreseeable decade.

Execution Models

Load balanced execution vs Single Executor through Election: Most token interface are accessed in a read-only manner, as integrated websites read token status and gets update from the token. However, for actions that actively with the external world, it's essential to have deterministic execution. This means that only one node should execute a particular action to avoid discrepancies. Examples will be provided in the next section.

The election of this single executor node is determined before any read/write operation. Among the nodes where a smart token is instantiated, one is elected as the execution node. This election isn't done by the node itself but through the network of anchoring nodes, ensuring a quick and unbiased selection.

Handling Failures: Failures, whether they're a standard part of the smart token's tokenscript or indicative of potential malicious activity, need to be addressed promptly to ensure service level objectives are met. If a node fails in its execution duties, the anchoring nodes step in. They can either arbitrate disputes or reallocate the smart token to a different, more reliable node. Only anchoring nodes can provide attested failures, however, they are not expected to take over the execution, hence its role is often the provision of attestation to the failure to acquire needed attestations to move to the next state. This is exemplified in the next section.

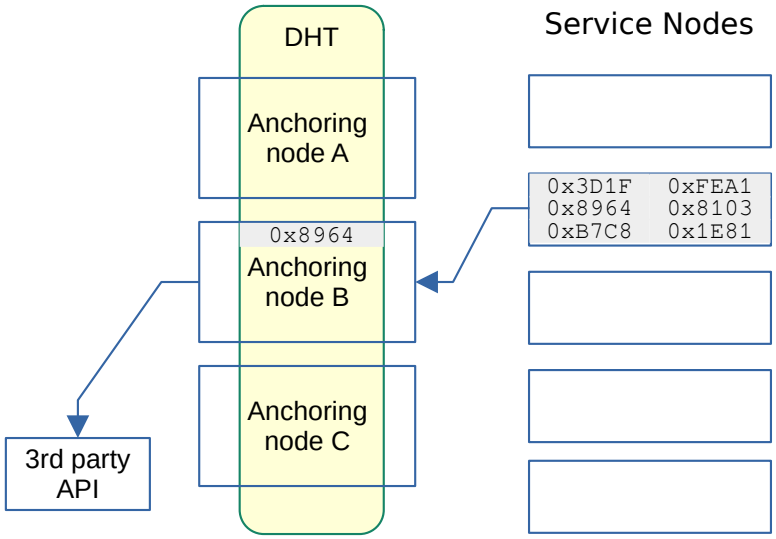

Figure 2: After a failure, a service node requests an anchoring node to route its traffice in order to get a failure attestation

Real-World Application: The Flight Ticket Smart Token

To better understand the intricacies of the Smart Layer network, let's delve into a real-world example: the flight ticket smart token.

Read-Only Access Imagine the user booked a car rental at the rental website, using the smart token on this website. This creates an authorization for the car rental to access the flight ticket smart token interfaces. When the car rental service wants to verify if your flight is on time, any node that has this smart token instance can provide a read-only API to share the flight's current status, and the selection of the node is largely the matter of load balancing. This process is facilitated by any of the nodes where the smart token is instantiated, selected at random.

Single Execution by Elected Node Now, consider a significant flight delay. This delay might trigger the smart token to rebook your hotel reservation. Since this action interacts with the external world and can have financial implications, it's crucial that only one node executes it. Once the rebooking is successful, the hotel system might generate a new booking attestation, confirming the change.

However, what if there's a failure? What if the node trying to rebook the hotel faces multiple timeouts when accessing the hotel's API? After a set number of failures, from the network point of view, an event occurred that the token failed to move to the next state. The traffic to the hotel system API would be rerouted through the anchoring node that elected the original executor. This anchoring node can then verify server timeouts or other errors, and provide a failure attestation to move the smart token to the next state. Frequent routing of this nature triggers service level agreement to scruitinise the node that was responsible for execusion.

While the flight ticket smart token serves as a tangible example, it's essential to understand that the Smart Layer network is not limited to this application. The principles discussed here apply to a myriad of potential smart tokens, each with its unique challenges and solutions.

Attestation Gossiping and Queuing in Smart Layer

In the Smart Layer architecture, the dynamic status of smart tokens is predominantly updated through attestations. These attestations, essentially cryptographic proofs, vouch for the validity of a particular state or action. Given the decentralized nature of the system and the real-time requirements for token status updates, there arises a need for an efficient mechanism to disseminate these attestations across the network.

The Need for a Hybrid Mechanism Traditional gossip protocols, inspired by the way information (or gossip) spreads in social networks, have been a staple in distributed systems. They ensure that data is disseminated quickly

and efficiently across a network. However, the unique requirements of Smart Layer, especially the need for ordered delivery of attestations and the ability to request missing attestations, demand a more sophisticated approach.

The unique requirement include a selective gossiping where an attestaion covers a type of smart tokens or a subset of smart tokens in that type, the timely update needed for integration to function, order and integrity, plus compatibility with a subscription model. This calls out for a hybrid mechanism, which marries the strengths of gossip protocols with the features of systems like Apache Kafka. While gossip protocols ensure rapid dissemination, systems akin to Apache Kafka ensure that these attestations are delivered in order and allow nodes to request specific attestations they might have missed.

How It Might Work Imagine a scenario where an airline releases a series of attestations updating flight arrival times. Nodes in the Smart Layer network that have smart token instances depending on that flight would subscribe to these attestations. As these attestations are gossiped through the network, the Kafka-like system ensures they're received in the correct order. If a node misses an attestation, it can request that specific piece of information, ensuring data integrity and consistency across the network.

This hybrid approach addresses the challenges of both rapid dissemination and data consistency. The decentralized nature of Smart Layer presents unique challenges, and we're committed to innovating and iterating on these solutions to ensure a robust and efficient system. As we progress, we'll continue to refine and adapt our approach to best serve the needs of the Smart Layer ecosystem.

Secure Execution Environment

Given the dynamic nature of smart tokens, it's imperative to have a secure environment for executing token code. Sandboxing techniques are employed within Smart Layer nodes, allowing token scripts to run in isolated environments. This ensures that the broader network remains unaffected by potentially malicious or faulty token scripts.

Smart Layer Token Types

The tokenomics of Smart Layer is intricately designed to ensure the sustainability, efficiency, and robustness of the network. Central to this design is the dual-token system, which serves distinct yet interconnected purposes:

Service Token

The Service Token is the primary medium of exchange within the Smart Layer network. It facilitates all micro-transactions associated with token operations, from querying token data to more resource-intensive modifications. Integrations typically bear the costs associated with token activation, which are transacted

using the Service Token. This token ensures that the network remains agile, with real-time settlements and efficient resource allocation.

Stake Token

The Stake Token represents a stake in the Smart Layer network. Holders of this token have a vested interest in the network's growth and governance. Depending on the network's design, Stake Token holders might influence governance decisions, propose changes, or even earn rewards based on network activity. This token ensures that the network remains decentralized, with stakeholders actively participating in its evolution.

The tokenomics of Smart Layer is designed to ensure the sustainability, efficiency, and robustness of the network. It strikes a balance between incentivizing token issuers, integrations, and the nodes that power the network. Here's a deep dive into the tokenomics structure:

Service Toke's Tokenomics

Token Issuer Rent

Every token issuer pays a nominal fee termed as "rent." This rent ensures that the TokenScript associated with the token remains available on the network. Additionally, it guarantees that any smart contracts linked to the token are actively monitored. While the rent is minimal, it serves a crucial purpose: it ensures that the network remains primed for any token activations, ensuring the readiness and responsiveness of the Smart Layer.

However, this rent doesn't necessarily translate to a financial burden for token issuers. On the contrary, they can potentially profit from the broader ecosystem, particularly from the "business" operations, which will be elaborated upon subsequently.

Token Activation and Operation Costs

While the token issuer pays the rent, the costs associated with token activation are typically borne by the integrations. An integration, in this context, refers to any stakeholder or entity that leverages the smart token for a specific use-case.

For instance, consider a health token. An e-commerce platform might utilize this token to optimize a user's shopping cart. In such a scenario, the e-commerce platform would cover the costs associated with accessing the token's interface. Conversely, a visa office might adopt a different approach. If an applicant provides data, such as a flight ticket in smart token format, the visa office might require the applicant to attach a small payment to cover the token operation costs.

It's worth noting that the cost of querying token data is relatively low, facilitated through a state channel. However, modifying token data, given its reliance on

gossiping and node synchronization, is more resource-intensive and, consequently, costlier.

State Channels and Settlement

State channels play a pivotal role in the Smart Layer tokenomics. Whenever an integration interacts with a smart token, a state channel is opened. This channel keeps track of all the micro-transactions associated with the token operations. Integrations have the flexibility to close these channels and settle the accumulated costs at regular intervals, such as monthly.

Anchoring nodes, which play a central role in the network's operation, maintain these state channels. They ensure that payments are routed correctly to the nodes that provide the services. At the end of a settlement period, the anchoring node consolidates all the transactions within the state channel and processes the payment to the respective service-providing nodes.

Incentive Structure and Profit Mechanism

The tokenomics is structured in a way that token issuers, while paying a nominal rent, can still profit from the broader ecosystem. This profit can be derived from various "business" operations facilitated by the smart tokens. For instance, a token issuer might collaborate with multiple integrations, each offering a unique service or benefit associated with the token. Every interaction with the token, whether it's a query or an update, translates to a micro-transaction. Over time, the cumulative value of these micro-transactions can result in substantial revenue for the token issuer.

Furthermore, integrations, by leveraging smart tokens, can offer enhanced services to their users. This not only improves user experience but also opens up new revenue streams for the integrations. For instance, an e-commerce platform can offer personalized shopping recommendations based on a user's health token, leading to increased sales and customer satisfaction.

In conclusion, the tokenomics of Smart Layer is meticulously designed to ensure a win-win scenario for all stakeholders. It promotes network sustainability, incentivizes participation, and fosters a vibrant ecosystem where token issuers, integrations, and nodes collaboratively drive the next generation of the web.

Conclusion

The digital realm is undergoing a transformative shift, with the decentralized web poised to redefine our online experiences. Amidst this evolution, Smart Layer emerges as a beacon of innovation, aiming to bridge the chasm between isolated services and tokenized assets. This whitepaper has illuminated the intricate architecture and potential of Smart Layer, a decentralized protocol that transcends the limitations of the current web.

At its core, Smart Layer is not just a technological marvel but a vision for a more integrated, dynamic web ecosystem. It addresses the 'Limit of 3' problem, breaking the shackles of limited integrations and offering a solution to the fragmented user experiences that plague the modern web. By introducing Smart Tokens, it paves the way for tokenized digital rights and services that can be effortlessly integrated across diverse web scenarios.

The protocol's architecture, rooted in proven distributed systems technologies, is tailored for token operations, ensuring serviceability, privacy, and security. The introduction of anchoring nodes, the emphasis on deterministic execution, and the innovative hybrid mechanism for attestation dissemination are testaments to Smart Layer's commitment to robustness and efficiency.

Furthermore, the tokenomics of Smart Layer is a masterclass in balancing incentives and sustainability. By ensuring that all stakeholders, from token issuers to integrations and nodes, have a stake in the network's success, it fosters a collaborative ecosystem. This ecosystem not only promises enhanced user experiences but also opens doors to new revenue streams and business models.

In essence, Smart Layer is more than just a protocol; it's a vision for the future of the web. A future where services are seamlessly integrated, where user experiences are enriched, and where the true potential of a decentralized, tokenized web is realized. As we stand on the cusp of this new era, Smart Layer beckons us to embrace the next generation of the web, promising a journey filled with innovation, integration, and limitless possibilities.