Dante Network：The "Internet protocol stack" of Web3

Abstract

Dante Network is the middleware of Web3 for collaboration among multiple ecosystems. In Dante Network, we define and implement a protocol stack for Web3 to realize the interconnection and interoperability, which will bring innovative experience for Web3, just as an "Internet protocol stack" is for the current Internet. Based on the protocol stack instance realized by Dante Network, in the future, not only token circulation among multi-chain ecosystems could be easily achieved, but comprehensive information perception and barrier-free interoperability of smart contracts will also be seamlessly feasible.

1. Introduction

1.1 Internet protocol stack

In 1969, the Advanced Research Projects Agency (ARPA) established ARPANet, which realized the interconnection of four large computers located at UCLA, Uc Santa Barbara, Stanford University, and the University of Utah.

This was the forerunner of the Internet, and in the decades that followed, the protocol stack and infrastructure of the Internet, such as the famous TCP/IP, were introduced and gradually became one of the indispensable cornerstones of the Internet protocol stack. Besides TCP/IP, there are many basic protocols, such as ARP, DNS, and the popular HTTP/HTTPS.

The current Internet is built on these protocols, which we call the ”Internet protocol stack” in general, along with the routing and switching networks that actually execute them. The ”Internet protocol stack” is rarely felt directly by the users, but without

them, the entire Internet and everything built on it will collapse such as online social platforms, online games, video sites, e-commerce, live streaming, etc.

1.2 Blockchain and Web3

Blockchain brings new possibilities to the development of the Internet due to its decentralization, transparency, and tamper resistance, along with unique features like open-source and DAO. All these promote Web3 gradually from concept to reality. Inevitably, as does any innovation, blockchain has also raised new concerns for Web3 in terms of connectivity and communication.

Ethereum's initial vision was to build a world computer. Regardless of its low TPS and high gas fee, it does, in a way, build a whole new paradigm of computers, with features of distributed, decentralized, Turing-complete, data storage, and operational dApps. As for the performance and cost issues mentioned above, many developers are trying to address them by constantly introducing innovative methods such as Proof of stake (PoS) consensus, sharding, Rollup, etc.

However, there is more to web3 than just Ethereum, just like there are billions of computers in the world than just one computer. Until now, there are many "world computers" similar to Ethereum, and they all have relatively stable technical architecture and mature ecosystems, such as Near, Avalanche, Flow, Filecoin, PlatON, etc. Some of them have higher TPS and lower gas fees, while others have special capabilities, such as scalable storage or privacy-preserving computation. Overall, the existence of these public chains expands the World of Web3's possibilities.

Web3 is currently in a state similar to that of computers prior to the Internet, when they were either isolated or had a limited range of connectivity. Equivalently, blockchains are not widely and effectively connected to one another, and data is

isolated from one another as well. This isolation is a natural consequence of blockchain's technical architecture. To ensure a reliable consensus in a trustless context, a blockchain only records and validates transactions that occur within its network. . However, this mechanism ignores other transactions that occur outside of the native chains, thus, isolation happens.

This isolation has restricted the Web3 composability and complementary between multiple blockchain ecosystems. We believe that in years to come, we will realize the need for a protocol stack for Web3 connections, just as an "Internet protocol stack" is for today's Internet.

1.3 A middleware in Web3

We hope to build Dante Network as a middleware that will enable the collaboration between multiple ecosystems of Web3. The middleware will consist of a protocol stack containing a series of multi-chain interoperability protocols and a network which is the instance of the protocol stack.

For developers, this middleware will make it easier for them to achieve diverse composability of business across multiple blockchains, allowing them to focus more on business development itself rather than basic and cumbersome issues such as data communication. For users, this middleware will assist them to move freely between multiple ecosystems, regardless of which ecosystem their current assets and identity are created.

2. Dante Network

2.1 Goals

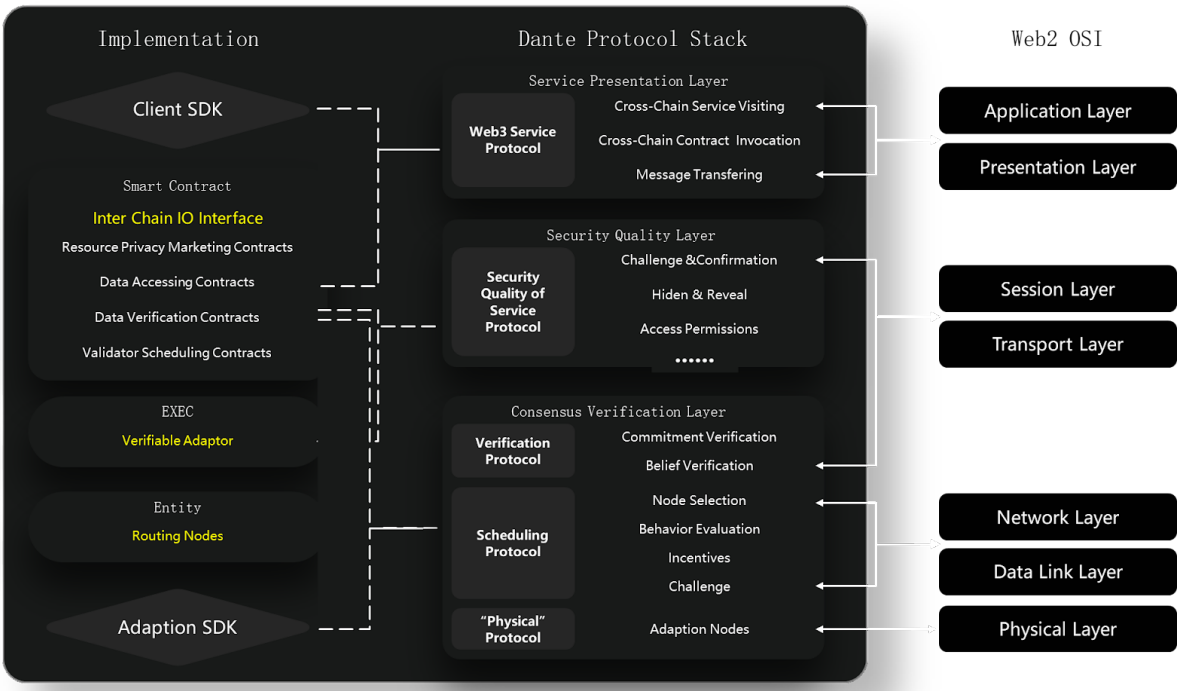
With cross-chain issues as a matter of Internet and communication, the requirements for middleware become more stringent and comprehensive rather than just app-specific.

At the outset of constructing the Dante Network, we believed that a good middleware should have fundamental and general features.. This led us to formulate two basic goals for the Dante Network:.

- Comprehensive interconnection: Dante Network will ensure the consistency of multi-chain environment and single-chain environment as far as possible, achieving not only the free cross-chain token circulation, but also the mutual perception of multi-chain information, and most importantly, the mutual invocation of cross-chain smart contracts.
- Agile configuration: Dante Network will allow the agile configuration adjustment on cross-chain efficiency, security, and decentralization on the condition of ensuring the fundamental functions in order to adapt to multiple scenarios.

2.2 Protocol Stack

We call Dante Network's protocol stack "World Tree Protocol Stack", similar to the Internet protocol stack, which attempts to define a basic, common and universal standard framework that can help interconnect various public chain ecosystems in the web3 world. The protocol stack is divided into three layers: service presentation layer, security quality layer, and consensus verification layer.



Pic1 Dante Protocol Stack

2.2.1 Service presentation layer

From an information science perspective, all interactions between multiple chains can be abstracted as data-based services. Whether it's storage, computing, or asset straddling. We have harmonized these actions with a set of standard service agreements called Web3 service.

Web3 service specifically defines message presentation for inter-ecological collaboration across multiple chains, similar to HTTP, FTP, SMTP, etc., in the Internet protocol; the calls for cross-ecological access to smart contracts, similar to RPC, and service presentation methods, similar to RESTful web services.

The service presentation layer can be compared to the application and representation layers in the Internet OSI model, which contains standard protocols for coding and decoding a range of service-related information such as data ontologies, data descriptions, service execution models, data validation, and information verification standards, and secure quality-of-service configuration information.

2.2.2 Security quality layer

The essence of cross-chain behavior is actually a communication category, and any communication protocol must consider the issue of Quality of Service (QoS), which is specifically reflected in TCP/IP and other Internet protocols.

However, in most current cross-chain services, little consideration is dedicated to this related aspect. In Dante Network, we formally propose Secure Quality of Service (SQoS). This protocol specifically defines the collaboration model of the web3 multi-chain ecosystem. From a functional point of view, this is similar to the session layer in the OSI model, as well as to some of the capabilities of the transport layer. With different SQoS configurations, users can choose flexibly between security, scalability, and decentralization according to the characteristics of their business, which is a very flexible way to cope with the blockchain trilemma in distributed systems.

2.2.3 Consensus verification layer

Blockchain itself is a distributed and trustless system. These features are further amplified in a multi-chain scenario, which is something that needs to be systematically considered for any cross-chain solution. In this regard, we plan a consensus verification layer to handle this problem, in which we will build a multi-node

collaborative consensus protocol stack. Following this protocol stack, multiple routing nodes can work in a trustless model. The protocol specifies the verification methods for various cross-chain collaboration services, the routing methods for data delivery, and the access and operation methods of the routing nodes that actually perform the related work.

The consensus verification layer is similar to the transport layer, network layer, data link layer, and physical layer in the OSI model. Its protocol stack contains the following main design aspects.

- Verification Protocol

This includes commitment verification and belief verification. The verification protocol defines in which way the legitimacy of the service provided by the working node will be judged. The specific node performing the service will submit the service result as defined in this protocol. Besides, in the commitment verification, a proof of completion of the service is also required. This protocol will also verify the submitted service results, and service proofs in the on-chain smart contract.

a) Committed Verification: it defines the relevant model for performing committed verification. Committed verification is a deterministic verification method that does not require any underlying routing and is able to guarantee the legitimacy of the service execution process and results through cryptographic algorithms. It is a relatively efficient implementation but has a limited application scenario.
b) Belief Verification: it defines the relevant model for performing belief verification. Belief verification is an uncertain verification approach where each service needs to be completed redundantly by multiple nodes, without trusting any of them, and the service outcome depends

on the aggregation of beliefs of multiple copies of the service. It is a relatively inefficient execution, but a widely adaptable authentication method.

- Routing Protocol

It defines the selection, evaluation, incentive, and challenge mechanisms for routing nodes when performing a specific service. Similar to the network layer, the data link layer in the OSI model.

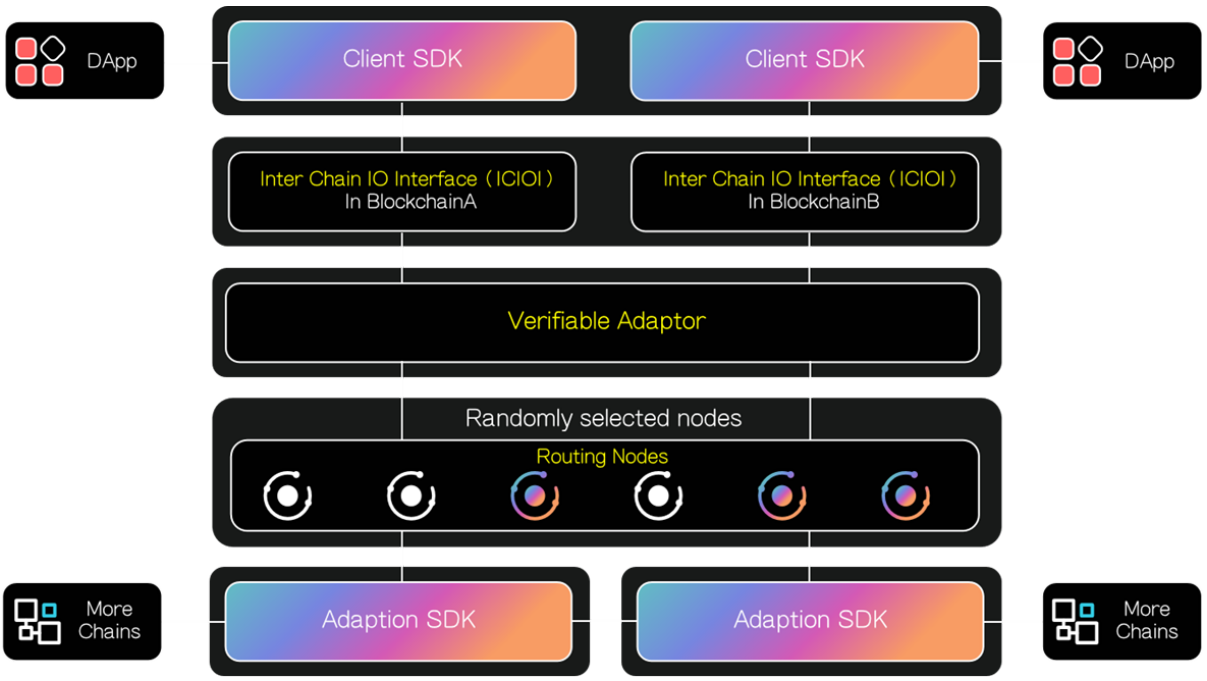
a) Routing selection mechanism: it defines the routing approach when executing a given service. For example, when performing commitment verification services, because commitment verification can cryptographically guarantee that the result is trusted, there are not many security constraints on the underlying routing, and a strategy with high execution efficiency and low routing security requirements can be used; however, when performing belief verification services, since all parties are trustless, multiple routing nodes are required to complete the tasks simultaneously. A certain degree of randomness in the selection is required to ensure collusion.
b) Behavior evaluation mechanism: it defines how to evaluate the behavior of node services during execution. Nodes that successfully complete the task, i.e. pass the verification, will receive positive feedback. Nodes that fail to perform the task, or fail the verification, will receive negative feedback. The feedback information will be expressed in the node's credibility, which will affect the likelihood of the node being selected.
c) Incentive/slashing mechanism: nodes need to perform the task by staking. Nodes that succeed in performing the task will receive a network incentive and the nodes that fail to complete the task will be slashed.
d) Challenge mechanism: it is a safety mechanism for system security that defines how to initiate challenges to abnormal network behaviors. Any node can issue a challenge to activities that it considers anomalous, like malicious deceptions

such as erroneous cross-chain messaging and the creation of information out of nothing.

- Node Adaption Protocol

This defines the framework and specifications for executing operations and submitting proofs for network routing nodes. In the OSI model, it is similar to the physical layer.

2.3 Framework



Pic 2 Framework of Dante Network

2.3.1 Routing Node

A routing node is a physical node entity that performs the physical routing of data from the source chain to the target chain. Routing nodes are open and decentralized, and users who meet certain basic requirements can be elected as routing nodes and provide services for multi-chain interconnection.

2.3.2 Verifiable Adapter

The verifiable adapter is an off-chain executable program that runs on top of the routing node and performs data routing at the software level, including collecting, aggregating, and classifying data from different chains, converting the classified data into information format readable by the target chain, and feeding it to the corresponding target chain based on the currently set pattern.

2.3.3 Inter-chain IO Interface (ICIOI)

The inter-chain IO interface is the smart contract cluster deployed on each chain. It is the gateway to other blockchains, through which the data between multiple chains will be sent and received.

In addition, this smart contract cluster could be regarded as the brain of cross-chain task scheduling, as it could set and command the logic of this cross-chain execution according to the specific cross-chain business requirements, including matching the source and target chains, specifying the confirmation mode of data, selecting and scheduling the routing nodes, making the final confirmation of data, etc.

Due to diverse EVM architectures and supported programming languages, the deployment form of smart contracts may differ from chain to chain, even though the business logic is the same.

2.3.4 Developers and SDK

We will package and provide two types of SDKs, namely Client SDK and Adaption SDK, for dApp developers and node/community developers respectively, in order to allow users to conveniently use the services provided by Dante Network and better enjoy the infinite possibilities brought by multi-chain interoperability. The Client SDK provides dApps development support, through which dApps will be able to directly

call Dante Network's multi-chain services, thus enabling dApps to realize information synchronization and smart contract calls among multiple chains.

The Adaption SDK provides node development support. Developers can carry out secondary development based on this SDK and include the link into the Dante Network connection scope for chains that Dante Network does not currently support, as long as they meet the requirements of the protocol standard.

2.4 See Dante Network from different perspectives

2.4.1 From the perspective of web3

Web3 is characterized by decentralization and self-organization, but this does not necessarily imply isolation. If we have ever deeply felt the incredible achievements of full interconnection for Web2, it is not difficult to imagine the significance of interconnection and interoperability for Web3 development.

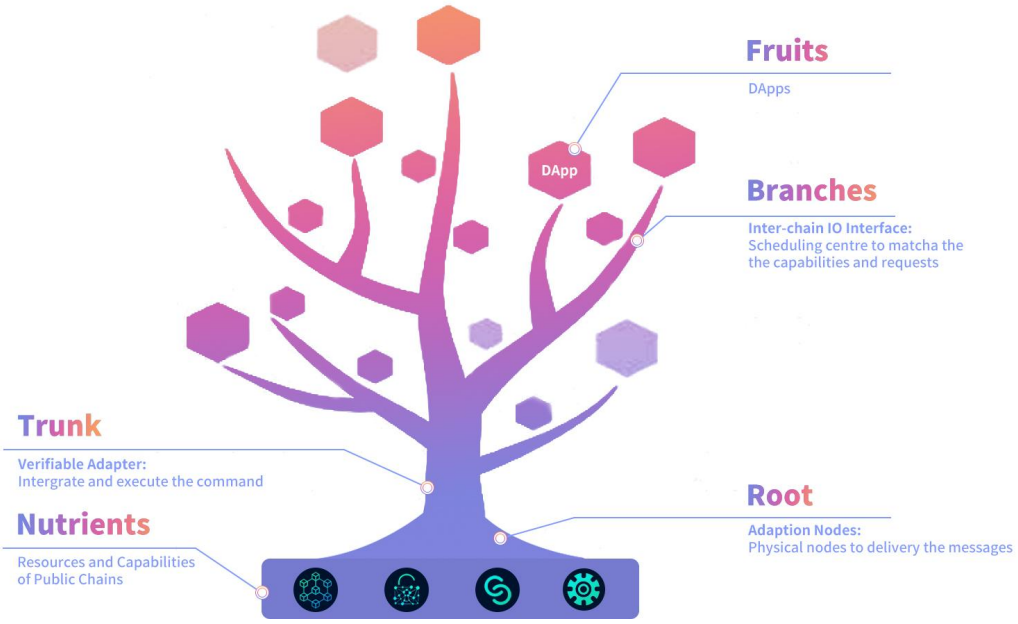
From the perspective of Web3, Dante Network defines the "Internet Protocol" that belongs to the native Web3. Our original intention of this protocol is to explore and implement the future of Web3. The protocol attempts to describe and define the specification of interconnection and interoperation in the Web3 world, and there could be different methods and examples for the specific implementation of the protocol. The world will make its own choice, we just need to do our best work.

2.4.2 From the perspective of dApps

When Dapp developers choose to develop and deploy their applications based on a certain chain, they usually consider ecological support, community prosperity, technology stack adaptation, infrastructure construction status, and so on.

However, in practice, for a specific public chain ecosystem, these elements are usually not available at the same time, which means that developers must frequently select and trade-off among them, and to some extent, choosing a certain chain involves sacrificing some options. In the context of cloud-native, each public chain essentially provides resources and capabilities, and the dApp's task is to integrate these resources and capabilities to support its own business logic. Before multi-chain interconnection and interoperability became a reality, each dApp could only have access to the resources and capabilities of the chain it was currently deployed on, .but the ultimate purpose of Dante Network is to break this limitation.

If we consider Web3 as fertile soil, Dante is a world tree, and the capabilities and resources of each public chain, such as the arithmetic power of ETH to execute smart contracts, the storage space of Filecoin, the TEE resources of PlatON, and the arithmetic power of privacy computing, are the nutrients in the soil. And many dApps on it are the fruits that are borne. The important role of the world tree is to absorb nutrients from the soil and transform them into the energy needed for the fruit to grow.



Pic3 World Tree in Web3

3. Technical Implementation

3.1 Security Quality of Service (SQoS)

SQoS defines what level of secure quality of service cross-chain operations will have. For any distributed system, the blockchain trilemma is an inevitable problem, but we are trying to work around it. In fact, we believe that not all scenarios require security, scalability, and decentralization at the same time in specific applications. Therefore, we would provide users with the ability to configure their preferences according to their own business needs. SQoS consists of a set of security-related and quality-of-service configuration items. In some cases, users can choose very high security at the expense of scalability, as one would choose TCP for reliable transmission in web2, while in other cases, users may not need very high security and resolve to favor scalability, as one would choose UDP for "connectionless" communication in web2.

Based on the research and analysis of a large number of application scenarios, we have designed the following configurable items for the SQoS part of the Dante protocol stack.

a) Challenge Confirmation: it describes whether the cross-chain service needs to confirm the wait, and the window size is the waiting time, which is the number of blocks. Challenge nodes can issue a challenge within the confirmation window. confirmation window in which the challenge node could initiate a challenge.
b) Hidden & Reveal: it describes whether the cross-chain service is executed in hidden & reveal mode. In this mode, the explicit content of the cross-chain service is not revealed until it is totally executed. This is to prevent nodes executing after the cross-chain operation from "plagiarizing" the "answer" of nodes that have already completed their execution.
c) Verification Threshold: it describes the credibility threshold of the message content when performing belief validation. If the verification threshold of the most "trustworthy" message content exceeds this threshold after verification is performed, the message will be accepted. If the threshold is set to 100%, all copies of the message must be identical in order for the verification to pass.
d) Priority: there may be factors that affect the process orders of cross-chain services, such as waiting time, types of messages, or price of service fees.
e) Exception rollback: multi-chain interoperation will lead to a breakdown of service call atomicity, and when an error occurs, the error message cannot be returned directly to the initial caller. Exception rollback is applied to cope with this situation.
f) Anonymous transactions: it is similar to the encrypted transmission in TLS/SSL. Actions in web3 occur through transactions, and anonymous

transactions will provide the ability to hide transaction information to a certain extent.

g) Identity traceability: multi-chain interoperation will lead to a disruption of service invocation atomicity, where the target chain will not be directly informed of the identity of the source coupling caller, and identity traceability will provide this capability, similar to identity verification in TLS/SSL.

h) Send/receive isolation: send/receive isolation will increase the difficulty of routing nodes colluding to do evil, regardless of whether the service and acknowledgment are done by the same routing nodes.

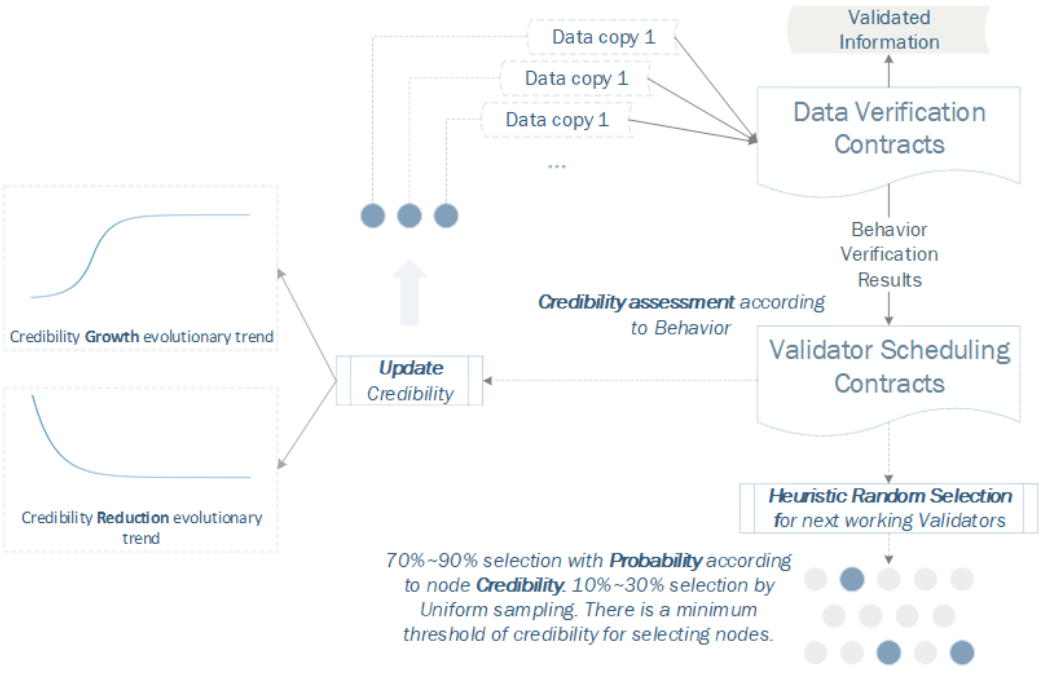
i) Cross-verification: this is an option specifically designed for a very high level of security in SQoS. Although the Dante protocol stack is designed and implemented with a more rigorous verification approach, we believe that the world of web3 is a world of co-construction. Therefore, we also allow users to access other similar infrastructures like Axelar, LayZero, or ChainLink to double-verify the message. This compatibility is also the reason why Dante protocol stack could be called a "multi-chain co-built protocol stack". Cross-verification provides a higher security capability than all existing single authentication methods and could be used in scenarios that require a very high-security level, e.g. large amount of multi-chain swap, DeFi, Token transfer, etc., ensuring that the Dante Network remains secure and effective even if one verification method is attacked and becomes unavailable, or is maliciously hijacked. Of course, there is no such thing as a free lunch, this result is obtained by sacrificing efficiency and paying higher service fees.

3.2 Data Routing

The essence of routing is to carry information from a source address to a destination address, and Dante Network accomplishes this task through routing nodes in the network.

Due to Web3's characteristics like decentralization, no permission, and trustlessness, the network is typically open and weakly-constrained to the possible participating routing nodes. Consequentially, when compared to Internet-based information transmission, information routing between multiple chains will require more design mechanisms, such as redundancy or cryptography-based premises assumptions, to ensure the accessibility and authenticity of information delivery. The entire routing strategy consists of a sequence of actions such as routing nodes selection, information verification, and routing result feedback.

To implement this routing policy In Dante Network, we will introduce a node belief evaluation model, a heuristic random selection strategy, and a combination of mechanisms such as staking, incentive, slashing, and challenge.



Pic4 Belief Verification

3.2.1 Node belief evaluation model

The node belief evaluation model is a probabilistic quantitative indicator of a router's legitimacy, which is expressed as the node's credibility value. When an adaption node provides belief verification services, its service quality will receive feedback in the verification session, and this feedback will be expressed as a change in its credibility. Let the credibility of each node be $\{c_0, c_1, c_2, \ldots, c_m\}$, and according to the constraints computed on-chain, the credibility can be converted into a corresponding integer to perform, as used for message verification in the belief verification above, assuming that here we define the credibility as a value of [0, 10000], which actually corresponds to [0, 100%].

Let:

$$
\begin{aligned}
&min = 0 \\
&max = 10000 \\
&middle = min + (max - min)/2 \\
&range = max - min \\
&step_{success} = about \quad 100 \\
&step_{doEvil} = about \quad 200 \\
&step_{exception} = about \quad 100
\end{aligned}
$$

The above indicates, from top to bottom, the lower limit, upper limit, middle value, and span of the range of plausibility values, the step amount of reasonable behavior, the step amount of evil behavior, and the step amount of abnormal behavior.

The step amount of reasonable behavior, the step amount of evil behavior, and the step amount of abnormal behavior will be specified at the time of going live on the main website after sufficient testing.

Constraints:

$$
\begin{aligned}
&step_* < range \\
&step_{doEvil} > step_{success} \\
&step_{doEvil} > step_{exception} \ ;
\end{aligned}
$$

The value of the credibility of the newly added nodes will be set around [3500, 5500], and the exact value will be specified when the main network goes online after sufficient testing.

3.2.2 Reasonable behaviors

When the message content, that is the hash value, "carried" by the node is consistent with the verified message, the system will consider the node's behavior reasonable and the node will receive positive feedback and its credibility will be improved.

The mathematical evolution of credibility improvement is as follows:

$$c_i^{t+1} \leftarrow f_{success}(c_i^t);$$

$$in \quad which \quad f_{success} \quad satisfy \quad c_i^{t+1} > c_i^t;$$

$f_{success}$ will be specified by considering the characteristics of the computational execution on-chain, so that the evolutionary trend $J_{growth}(c_i^0, f_{success}, t)$ has the following characteristics:

$$\frac{\partial J_{growth}}{\partial t} > 0;$$

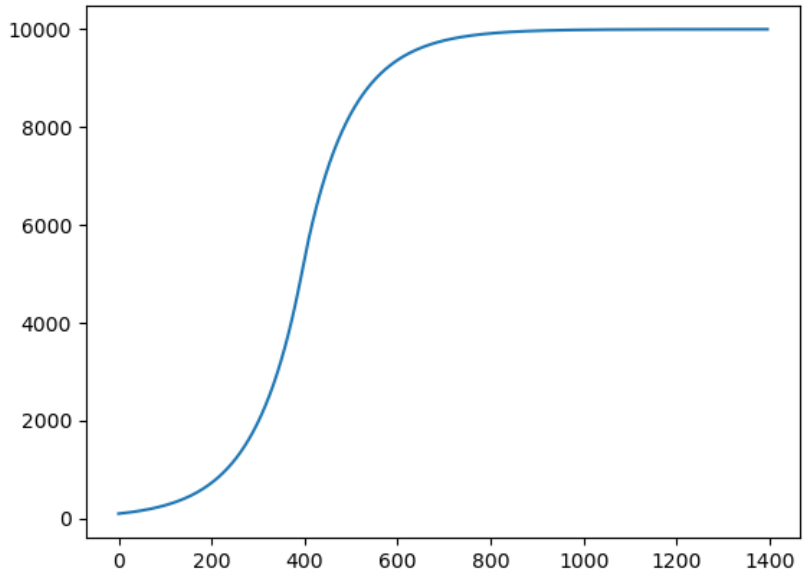
$$\begin{matrix} \frac{\partial^2 J_{growth}}{\partial t^2} > 0 \quad when \quad c_i^t < middle \\ \frac{\partial^2 J_{growth}}{\partial t^2} < 0 \quad when \quad c_i^t > middle \end{matrix} \quad , \quad c_i^t = f_{success}^{(t)}(c_i^0) \quad ;$$

Where $f_{success}^{(t)}(c_i^0) = f\left(f_{success}^{(t-1)}(c_i^0)\right) \quad , \quad f_{success}^{(0)}(c_i^0) = c_i^0$;

This will make it possible for nodes with very low credibility to increase their credibility slowly at first but will grow faster and faster. The fastest growth will occur when the credibility is close to the median. The initial value of newly added nodes is close to the median, which is an incentive for new nodes. After the credibility exceeds the

median, the growth rate will gradually slow down. The growing credibility will eventually converge to the upper limit indefinitely.

The simulation results of the evolutionary trend of the continuous growth of the credibility of reasonable behavior $J_{growth}(c_i^0, f_{success}, t)$ are shown in the following figure, where the vertical coordinate is the credibility and the horizontal coordinate is t, representing the evolutionary step.



Pic5 Increasing trend in the credibility

The purpose of this design is to give new nodes the opportunity to quickly improve their credibility, while for nodes with low credibility due to evil, more effort is required to improve their credibility to normal levels, indirectly increasing the cost of evil.

3.2.3 Evil behaviors

When the message content, the hash value, "carried" by the node does not match the message content derived from the verification, the system will consider the node to be behaving badly, and the node will receive negative feedback and its credibility will be reduced.

The mathematical evolution of the credibility reduction of the misbehavior is as follows:

$$c_i^{t+1} \leftarrow f_{doEvil}(c_i^t);$$

$$in \quad which \quad f_{doEvil} \quad satisfy \quad c_i^{t+1} < c_i^t;$$

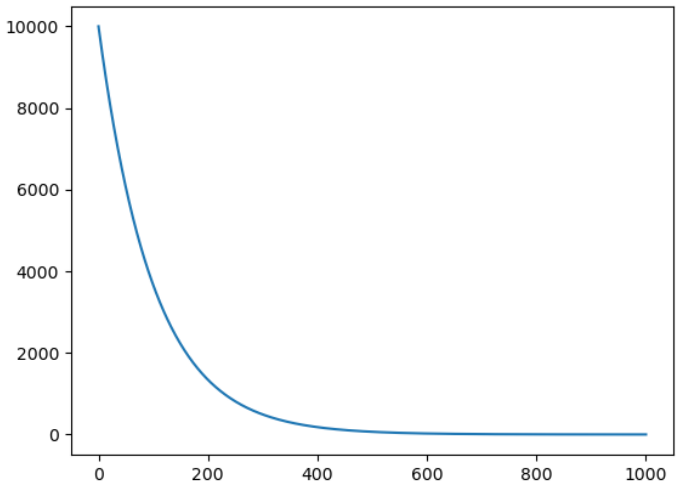
f_{doEvil} will be specified by considering the characteristics of the computational execution on-chain, so that the evolutionary trend $J_{reduction}(c_i^0, f_{doEvil}, t)$ has the following characteristics:

$$\frac{\partial J_{reduction}}{\partial t} < 0;$$

$$\frac{\partial^2 J_{reduction}}{\partial t^2} > 0;$$

It is easy to see that the higher the credibility, the more severe the punishment will be once the evil is done. This is because the higher the credibility, the higher the probability of being selected in the selection of routing nodes. And the rate at which the credibility of evil behavior decreases is greater than the rate at which the credibility of successful behavior increases.

The simulation results of the evolutionary trend of the continuous decrease in the credibility of the evil behavior $J_{reduction}(c_i^0, f_{doEvil}, t)$ are shown below.



Pic6 Decreasing trend of credibility

Such a design aims to raise the cost of node evildoing so that it is financially more uneconomical for nodes with high credibility to do evil. A single act of evil will not only cost the staking but also require multiple successful and reasonable operations to make up for its loss in credibility.

3.2.4 Abnormal behaviors

When a message is verified and no consistent message can be derived, the system determines that the message behaves abnormally and all related nodes of this cross-chain message will get negative feedback and their credibility will be reduced. The reduced value is influenced by the weight q_i of the group to which it belongs.

The mathematical evolution of the credibility reduction of the anomalous behavior is as follows:

$$c_i^{t+1} \leftarrow f_{exception}(c_i^t, q_i);$$

$$in \quad which \quad f_{exception} \quad satisfy \quad c_i^{t+1} < c_i^t;$$

$f_{exception}$ will be specified by considering the characteristics of the computational execution on-chain, so that the evolutionary trend $J_{exception}(c_i^0, f_{exception}, t, q_i)$ has the following characteristics, where t is the evolutionary step.

$$\frac{\partial J_{exception}}{\partial t} < 0$$
$$\frac{\partial J_{exception}}{\partial q_i} < 0;$$

$$\frac{\partial^2 J_{exception}}{\partial t^2} > 0;$$

The decrease in credibility due to abnormal behavior will be less than the evil behavior in the same situation and will depend on the group weight to which the message delivered by the node belongs at the time of this message verification. The higher the group weight, which means that the message executed by the node is more consistent with the content of the message executed by more, or more trustworthy nodes, the lighter the penalty will be for the node and the smaller the decrease in credibility value.

Similar simulation results for the evolutionary trend of persistently decreasing credibility of abnormal behaviors $J_{exception}(c_i^0, f_{exception}, t, q_i)$ and the evolutionary trend of decreasing credibility of evil behaviors $J_{reduction}$.

The purpose of this design is to provide some protection for nodes that are relatively "legal" after abnormal behavior occurs.

3.3 Heuristic random selection

Since the joining of nodes is completely open-ended, if the nodes are allowed to perform the operation completely freely, it may lead to some tasks that are inefficient due to too many participating nodes. And some tasks fail to meet the requirements of

belief verification due to few participating nodes. This project adopts a heuristic random selection strategy to select the execution nodes for each task. The strategy is executed in on-chain smart contracts so that the consensus mechanism provided by the public chain can be used to ensure the fairness of the selection strategy execution.

The share of each selected service node will be divided into two parts, credibility selection, and random selection. Let the total number of selected service nodes be N_s, and the share of credibility selection N_c accounts for the system upper limit of S^+ and lower limit of S^-, then the share of credibility selection each time takes the value space of $[S^-, S^+]$. Meanwhile, we set the minimum threshold of node credibility each time to P_c^{min}, and nodes below this threshold will no longer be selected. The nodes with credibility greater than $P_c^{trustworthy}$ are called trustworthy nodes.

Detailed algorithms are as follows:

In credibility selection, the probability of a node being selected as a service node is influenced by its credibility, and let the credibility of all nodes greater than the minimum threshold of credibility be $\{c_0, c_1, c_2, \ldots, c_n\}$, the selection probability of node i credibility mapping is:

$$p_i = \frac{c_i}{\sum_{j=0}^{n} c_j};$$

The total percentage of all trustworthy nodes is:

$$P_{all}^{trustworthy} = \sum_{j=0}^{k} p_j \quad , where \quad p_j \geq P_c^{trustworthy}$$

.

Then the credibility selection shares at this point are:

$$N_c = N_s \times max\{min\{P_{all}^{trustworthy}, S^+\}, S^-\};$$

while the random selection shares are:

$N_r = N_s - N_c$.

Selecting N_c nodes as service nodes by sampling with probability $\{p_0, p_1, p_2, \cdots, p_n\}$ from all nodes with credibility greater than or equal to P_c^{min}.

Then select N_r nodes as service nodes by uniform sampling from the remaining nodes with credibility greater than or equal to P_c^{min}.)

3.4 Incentives/Slashing

The node executing the message routing needs to stake a certain amount of deposit, and if its execution result is finally verified and adopted, it can get the corresponding incentive, and if it fails to pass the verification, the staked deposit will be slashed.

The specific incentive and slashing parameters will be described in a special document.

3.5 Challenge

For operations with higher security settings in SQoS, like asset cross-chain, a task confirmation window will be set when cross-chain information verification occurs abnormally, and within that time window, the challenger can challenge the abnormal behavior. At this time, Dante Network will randomly select multiple nodes to confirm the challenge, for example, in the cross-chain transfer scenario, query whether the transaction really exists on the source chain, unless all nodes in the network are faulty nodes, it is possible to determine the authenticity of the challenged object. If the result is true, the challenge fails and the challenger is slashed, while, if the result is false, the challenge succeeds and the relevant route nodes that performed the task

before will be slashed. Additionally, if the result is inconsistent, the message will be withdrawn. This is to avoid an extreme case where the network is over-represented by malicious nodes, in which case it is better to fail the cross-chain sending than to let the attacker succeed.

Challengers are also required to stake a certain number of Tokens to initiate a challenge. At this point, the contract will reselect multiple nodes to perform the secondary verification, and the nodes that perform this verification will be incentivized.

This challenge mechanism is actually a secondary verification mechanism, and the nodes are also reselected by heuristic random selection, but the parameters are set to prefer nodes with higher confidence, unless all nodes in the network are malicious nodes, the probability that the two randomly selected adaption nodes happen to be all malicious nodes is small.

Let us consider an extreme case where the nodes in the network are all bad and are all colluded in advance. At this point, let the percentage of malicious nodes in the network of adapted nodes be P_{evil}, the total number of adapted nodes in the network be N_a, and the number of nodes selected to perform the task each time be N_{s-1}. For scenarios of high importance, we set that all message copies must be identical to pass the verification. The number of nodes selected to perform secondary confirmation verification at each time is N_{s-2}, and the secondary verification itself requires all confirmation results to be consistent in order to confirm the challenge result. Therefore, when:

The probability that all nodes selected by the task verification are malicious nodes $P_{1-allEvil}$ is

$$P_{1-allEvil} = \frac{C_{\lceil N_a \times P_{evil} \rceil}^{N_{s-1}}}{C_{N_a}^{N_{s-1}}};$$

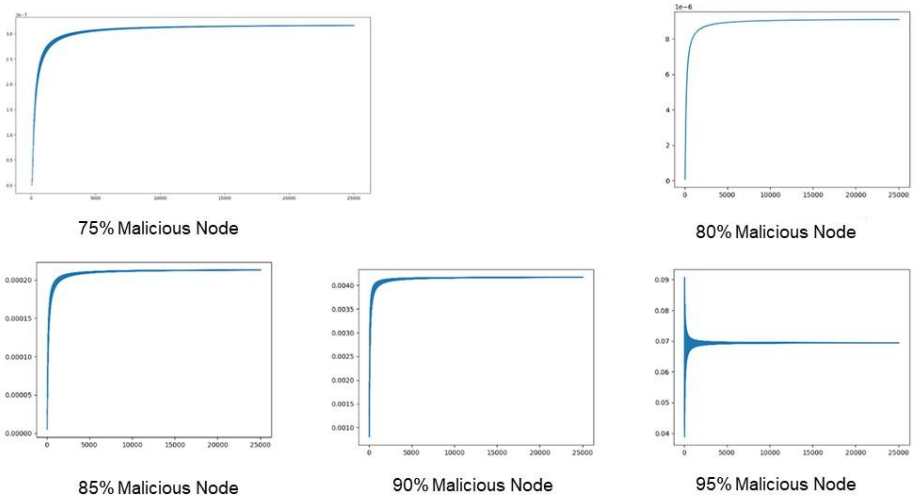
the probability that all nodes selected for secondary validation are malicious nodes is $P_{2-allEvil}$ as:

$$P_{2-allEvil} = \frac{C_{[N_a \times P_{evil}]}^{N_{s-2}}}{C_{N_a}^{N_{s-2}}};$$

Then, the probability that this belief verification is successfully attacked by a malicious node resulting in a verification error is P_{failed} :

$$P_{failed} = P_{1-allEvil} \times P_{2-allEvil}.$$

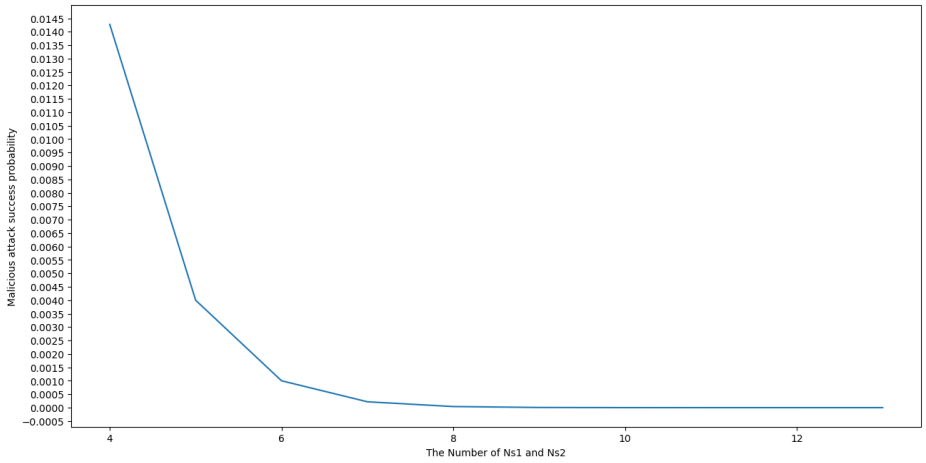
We assume that in the extreme case, the number of nodes N_{s-1} selected to perform the task each time is 21, the number of nodes selected for secondary confirmation verification N_{s-2} is 31, and the malicious nodes account for 75%, 80%, 85%, 90%, and 95% at different total number of nodes N_a, noted, most consensus mechanisms of the blockchain actually do not work anymore when the malicious nodes exceed 51%, and the simulation results of the probability of successful attacks by malicious nodes are as follows, the horizontal coordinates are the values of N_a, and the vertical coordinates are the probability of successful attacks.



As can be seen, the probability of a successful attack is almost negligible when the malicious nodes account for less than 80%. It is still only up to 0.02% (2 in 10,000) at 85%; up to 0.4% (4 in 1,000) at 90%; and will not be negligible at 95%, up to 9%. Of

course, we are simulating under very extreme assumptions, that is, assuming that all malicious nodes collude in advance, however, the engineering cost of such a scenario actually happening in a real situation is very large, in other words, it requires a very large engineering cost for collaborative collusion. In fact, a decentralized network with 51% or more malicious nodes is unlikely to be financially profitable for virtually all participants in the network, including the malicious nodes.

In addition, even in the early stage of Dante Network, assuming that we set a minimum of 21 nodes are needed before the main network will go online, at this time, N_{s-1} is set initially to 3, gradually increasing, N_{s-2} is the same as N_{s-1}, which is a very loose setting, when there are 60% of malicious nodes, which is already much larger than the percentage of malicious nodes that can be tolerated by the conventional blockchain consensus, the simulation results of the probability of successful attack by evil nodes are as follows (the horizontal coordinates are the number of N_{s-1} and N_{s-2}):



As shown in the figure, the probability of a successful attack, in the early stage of Dante Network with only 21 nodes, even with a completely open node joining model, and in the meantime, we are under very loose security settings (at this time N_n is fixed 21, and N_{s-2} is the same as N_{s-1}), as long as the number of selected initial execution nodes and secondary confirmation verification nodes exceeds 8, even if there are

60% of malicious nodes, the probability of their colluding in a successful attack is less than 0.02% (2 in 10,000).

We also simulated the initial network with a total number of 21 adapted nodes, where 70%, 80%, and 90% of malicious nodes. In the case of 70% malicious nodes, as long as the number of execution nodes and secondary verification nodes is not less than 8, the probability of a successful malicious attack does not exceed 0.1% (1 in 1000); in the case of 80%, the probability of a successful malicious attack is 1.5% when the number of execution nodes and secondary verification nodes is 8, and 0.4% (4 in 1000) when there are 10 nodes; in the case of 90% of malicious nodes, more than 16 execution nodes and secondary confirmation verification nodes are required at this point to make the probability of successful attacks small (the attack success rate is about 0.3%).

Information verification

In the Internet, credibility verification of data is not mandatory, and it is usually the application layer that decides whether to perform the verification work according to its own business characteristics. In the usual scenario, only packets are generally verified to ensure the integrity of data transmission, and in some scenarios, the integrity of data is not even judged, such as UDP. However, since the blockchain network itself and with other chains are in a trustless state, data verification is a mandatory point of consideration in the cross-chain scenario.

In abstract terms, we believe that the verification approach can be divided into two different modes based on the driving modality of the data.

One is commitment verification, in which the data requestor submits a data requirement in advance, which contains a specific data commitment, and the data responder is required to submit a data proof along with the data delivery. This data proof can be cryptographically verified against the data commitment, and if the

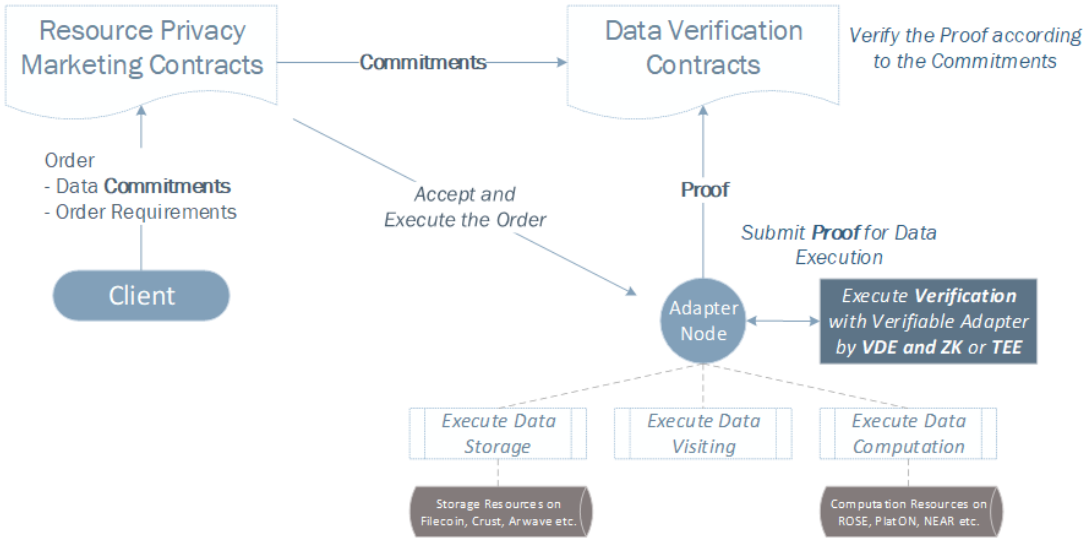
verification is true, the data provided by the responder can be considered cryptographically trustworthy. In this project, commitment verification supports diverse manifestation forms, such as hash values corresponding to the data itself, Merkle roots for block-stored data, VDE (verifiable delayed encoding) processing, commitments in homomorphic hidden form, hidden commitments in trusted execution environments, etc. Also, in this project, when verifying against these commitment submission proofs, ZK Proof generation will be considered to shorten the proof length. The implementation of commitment verification has some technical difficulties, but its execution process can be done independently by only one node, so the execution is relatively efficient.

The other is belief verification, a model in which either there is no direct data requestor or the data requestor cannot give cryptographic data commitments in advance. We will encounter this situation when listening to messages that are carried cross-chains, or messages that are generated entirely off-chain, e.g., the results of executing some complex off-chain computation, the engineering cost of which can be very high if the complex computation is to be arithmeticity and a ZK Proof is generated. Since the off-chain process performed by a single node is not trusted, in this mode we need to select multiple nodes at a time to complete the operation together. This project uses a probabilistic belief-based model, i.e., each message is verified by comparing and aggregating the contents of the copies provided by all relevant execution nodes. Belief verification must consider the presence of malicious nodes, at which point multiple results with inconsistent content will be obtained, along with the associated credibility of each result. Credibility represents the distribution of our beliefs about the results.

Commitment Verification

This is a type of proof that is cryptographically verifiable as to its authenticity. Each proof is proof against a particular commitment, i.e., the verification of the proof is a

calculation of whether the proof satisfies the conditions of some submitted commitment in advance. This type of proof is more efficient and we will implement it based on ZK Proof. In the initial stage, it may be considered to be implemented by hardware facilities such as TEE.



The technology principles for commitment verification are as follows:

The requester makes a data request, which can be a request for data storage, a request for data visiting, a request for data computation, and, depending on the specifics of the data, uses a commitment function $f_{commitment}(x_{data})$ to create a data commitment and submits.

$$D_{commitment} = f_{commitment}(x_{data});$$

where x_{data} is the relevant input for generating the commitment, e.g. when the commitment is a Merkle root, x_{data} is a data block vector.

$f_{commitment}(x_{data})$ has the following characteristics:

Hiddenness: the inverse function of $f_{commitment}(x_{data})$ is not available, i.e., $D_{commitment}$ is known and x_{data} cannot be computed, e.g., a hash function.)

Bondability: $D_{commitment}$ can uniquely represent x_{data}, i.e., different x_{data} will yield different $D_{commitment}$.

After the responder performs the relevant data operation, the result is submitted along with the associated $Proof_x$, which is verified by $f_{verification}(D_{commitment}, Proof_x)$.

$$\begin{cases} f_{verification}(D_{commitment}, Proof_x) = True & \text{if} \quad passed \\ f_{verification}(D_{commitment}, Proof_x) = False & \text{if} \quad failed \end{cases};$$

$f_{verification}(D_{commitment}, Proof_x)$ has the following characteristics:

Hiddenness: it is sufficient to provide $Proof_x$, without disclosing the plaintext of the specific data x_{data}, while x_{data} cannot be calculated based on $Proof_x$.

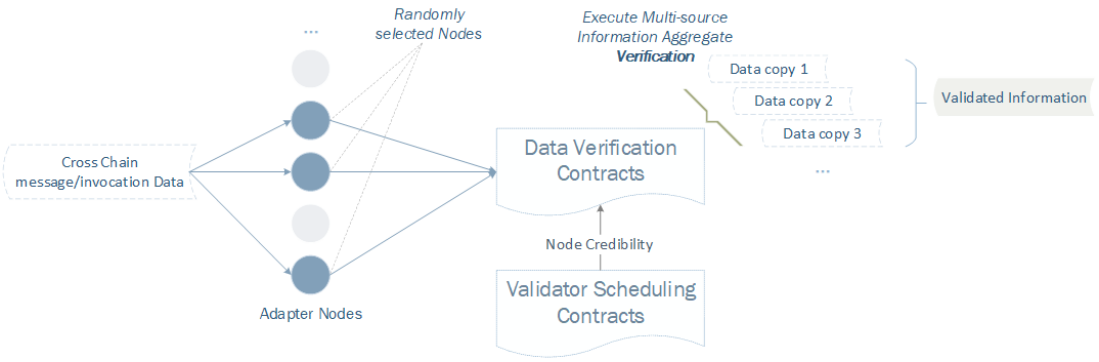
Bondability: $Proof_x$ can only be proved against the unique $D_{commitment}$, i.e., mismatched proofs and promises will yield a False conclusion.）

Belief verification

Belief verification is a redundancy-based verification approach. In simple terms, the receiver's judgment of a message depends on the belief it computes based on the multiple copies of the message it receives. Each proof is against the message itself, and the verification of each message requires receiving multiple copies of the message from multiple nodes. The message receiver cannot trust each individual message copy and needs to aggregate the judgment on the content of message copies from multiple sources to complete the verification. Conflicts among the information copies may result in not getting a definitive message for the receiver.

In belief verification, each message is operated on by multiple nodes (e.g., a node carries a cross-chain message). Each node delivers a copy of the carried message to the destination chain, and the message is verified in Data Verification Contracts based

on the content hash of each message copy, along with the credibility of the operating node.



The technical principles of belief verification are as follows:

Message copy aggregation: Aggregate the message copies according to the hash value, and place the message copies with the same hash value together as a group. If more than one group exists, the groups are numbered according to the number of message copies in each group, from most to least, with a starting number of 0.

Each cross-chain node has a value that identifies its credibility, which is in the range of [0, 100%] and will be converted to an integer value accepted by the contract when it is processed in the on-chain contract, assuming that here we convert the value to [0, 10000] (which actually corresponds to [0, 100%]), where 9586 represents 95.86%.

Let there be a total of K groupings, each grouping contains k_i message copies, $i \in [0, K)$, the message copies in group i are from nodes $\{n_{i_0}, n_{i_1}, n_{i_2}, \ldots, n_{i_{k_i-1}}\}$, and the credibility of each node is $\{n_{i_0}, n_{i_1}, n_{i_2}, \ldots, n_{i_{k_i-1}}\}$, respectively, and the message credibility measure v_i in group i is calculated by:

$$v_i = \sum_{j=0}^{k_i-1} c_j$$

The sum of message confidence measures V for all subgroups is calculated as:

$$V = \sum_{i=0}^{K-1} v_i$$

The message credibility weight q_i for subgroup i is:

$$q_i = \frac{10000 \times v_i}{V} = \frac{10000 \times v_i}{\sum_{j=0}^{K-1} v_j} = \frac{10000 \times v_i}{\sum_{j=0}^{K-1} \sum_{l=0}^{k_i-1} c_l}$$

Let the currently set message verification threshold be T. If there exists message credibility weight $q \geq T$, the grouping with the highest credibility weight is used as the adopted grouping of the current message, and the content of the message corresponding to this grouping is the adopted object; if the credibility of all grouped messages is less than the message verification threshold T, the consistency result cannot be derived and the current message verification cannot pass, at which time the credibility of all the adaption nodes associated with this message will be affected, in the manner detailed in the key technology heterogeneous behavior scheduling.

The credibility of the adaption nodes corresponding to message copies with the same content as the adopted message will be enhanced; meanwhile, the credibility of other adaption nodes corresponding to message copies with inconsistent content with the adopted message will be reduced. The specific method is described in detail in the key technique heterogeneous behavior scheduling.

Use Cases

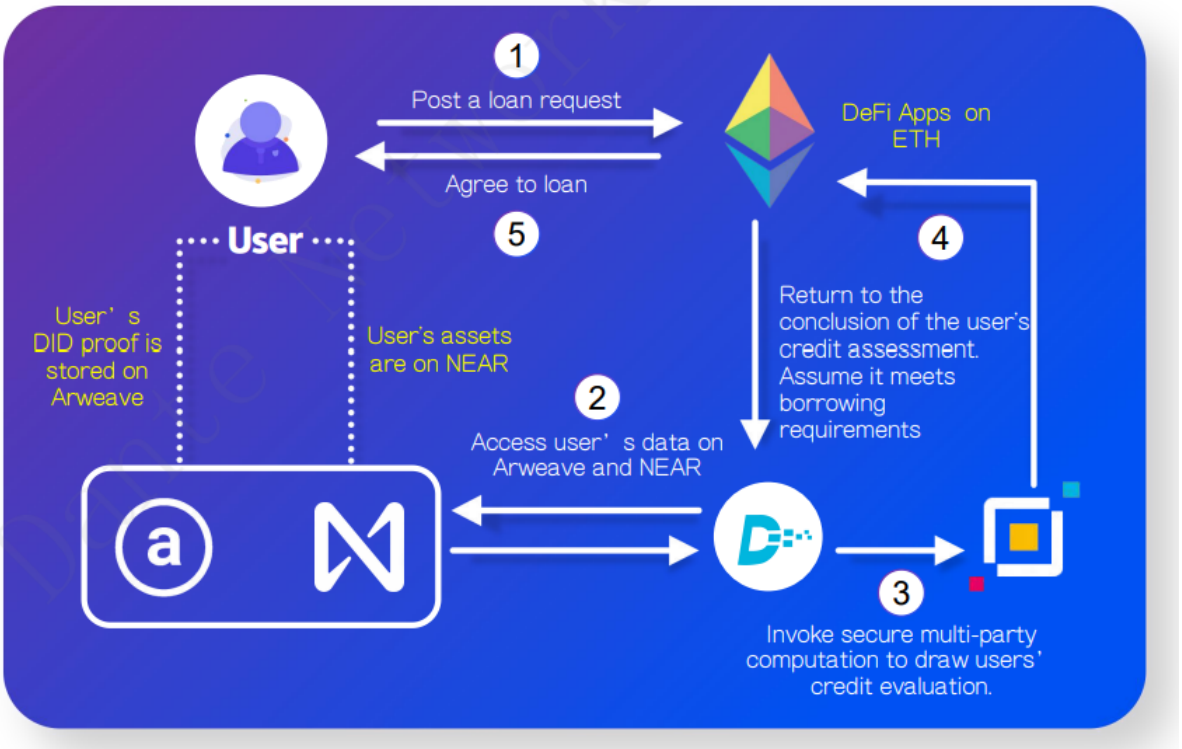
Credibility Loan in DeFi

Credit loan is a common approach in traditional finance. The borrower only needs to present his qualified credit history to the lender, without additional collateral, in order to lend a loan that meets his credit limit. This credit history is derived from a person's identity information, behavior history, and own assets.

In reality, such information often comes from different places. Your identity information may come from a government or a large organization's authorization, your behavior record may come from your credit report, and your asset information may come from your home's title deed, or your stock holdings. You don't need to turn all of that over, but you do need to prove that you own it all. For example, your property, otherwise that would be a mortgage, and you just want to show that you are a creditworthy person and you have the ability to repay the loan. Even in the real world, it's not an easy thing to get these authorizations and proofs from different agencies, but it can at least be done anyway.

Unfortunately, in the world of crypto, this is rather made more difficult due to the natural isolation of blockchain from blockchain and the decentralized nature of it all. We know that loan is an important application in the DeFi, but the most established lending methods are currently collateralized loans, which require the lending user to overcollateralize in order to lend a portion of their liquidity. Mortgages are certainly a good way to solve the liquidity problem to some extent. However, credit lending is a more important way to achieve financial inclusion. We've undoubtedly felt all of this in real life. And we feel that World Tree can help DeFi achieve all of this.

Here is the application scenario：



A DeFi application deployed on Ethereum, has as main function to review the user's information and generate a credit assessment that corresponds to the corresponding loan amount. In order to more fully assess the user's credit and control the risk of the loan, it requires the user to provide it at least two valid pieces of information, and identity from the DID application and asset information.

We assume that this identity information is permanently stored on the Arweave network, while the user's main assets are placed on other public chains such as Near or Avalanche. Dante can help verify and submit this information to Ethereum. However, we think this may have some pitfalls because current DIDs usually bind the identity information on-chain to the real identity information to ensure the uniqueness of the identity. It is not a friendly act to rashly publicize the user's information and its associated asset information on the chain without a breach event.

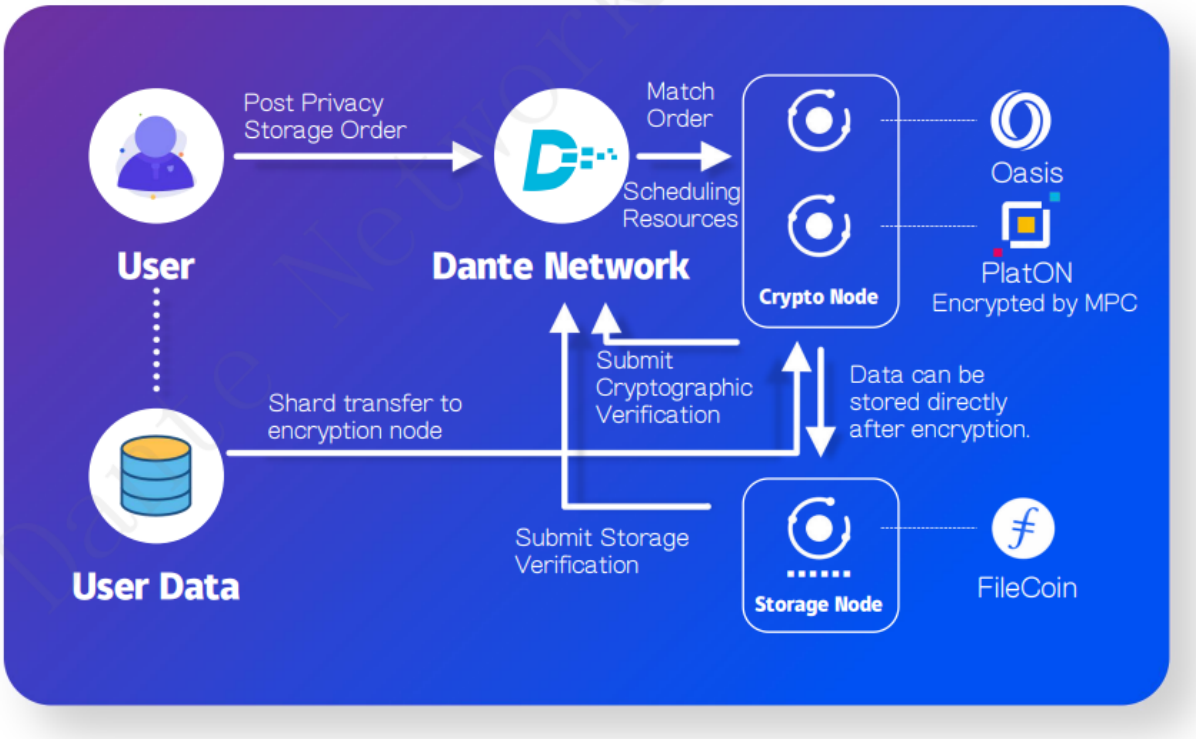
Considering that what the DeFi application actually really needs is not the complete information of the user, it only needs the credit evaluation conclusion derived from this information, we can borrow privacy computing to complete this process and avoid directly exposing the user's privacy. A reassuring phenomenon is that there are already many public chains trying to solve the problems in privacy computing, such as Oasis, or PlatON, so we don't need to build the wheel again and again.

Dante Network can help import this information into these privacy computing public chains, invoke their secure multi-party computing power to complete the execution of the credit assessment algorithm, and ultimately arrive at a credit assessment conclusion. And ultimately the DeFi app can grant or deny loans to users based on this information.

Decentralized Encryption

In real-world scenarios, data owners own the data but do not necessarily have the ability to perform encryption. Many end devices, such as a large number of cell phones, IoT sensors, etc., are often the source of continuous data generation, but given their lightweight nature, expecting them to accomplish complex encryption against attacks, homomorphic encryption, etc., is a difficult task. Therefore, encryption of data can be accomplished in a decentralized manner, such as through secure multi-party computing, at nodes that have the ability to perform cryptographic computation without exposing the data. Some public chains naturally have the resources and capabilities of such secure multi-party computing, and the applications on them can invoke such capabilities well, and the users can also enjoy the benefits conveniently. However, it is not universal for public chains. In fact, due to the limitation of professional direction, most of the public chains we know now do not have such resources and capabilities.

In a closed world, this is an acceptable norm, however, in the more open Web3 world, we believe this needs to be broken. Dante Network can help import these resources and capabilities into more Web3 public chains, making it possible for users or dApps to initiate requests about privacy computing anywhere and get a response. As shown in the diagram, this is a case study about storing raw data after decentralized encryption.



In the whole process, the interaction between the user's original data, a privacy computing public chain, such as Oasis or PlatON, and a storage public chain, such as Filecoin, will be completed, ensuring that the original data privacy is not compromised through proper storage.

NFT Display

Considering the popularity of NFT at the moment, the current release of a set of NFT-related on-chain work is well established, and likewise, the public presentation of NFT is nothing too difficult, for example, the famous NFT platform OpenSea on Ethereum is well qualified to do this. Thanks to the popularity of Ethereum and the maturity of the ecology, the related similar supporting facilities are so popular that we often take it for granted. Very often, we overlook the fact that many emerging public chains have very little infrastructure in the early days of the ecosystem, and NFT platforms like OpenSea are just one of them.

When the creator of an NFT project publishes a set of NFTs on an emerging public chain, it will be very difficult to show the real value of the NFTs if the infrastructure related to the NFT presentation is not perfect and there is a lack of a certain user-based NFT presentation platform. Not all users have the ability to check the source code of the smart contract to find out what this NFT is. If this NFT is a work of art, then users will want to be able to start enjoying the painting directly, rather than being told that you have to go to the link to inspect the code.

Of course, NFT issuers can choose to go to other public chains with relatively mature NFT infrastructure to publish, but if the NFT has a very close relationship with this emerging public chain, e.g. documenting important milestones in the development of the public chain, or coalescing the creation of native value within the ecosystem), then publishing on that emerging public chain becomes the most appropriate choice.

Its specific use process is as follows.

1. Deployment of specific NFT contracts by users on emerging public chains (native chains), such as digital collections that record important milestones in the project's development history.
2. deployment of relevant standardized NFT mapping contracts by the user on ETH, with all relevant operations on the native chain, synchronization of which is done through Dante Network cross-chain contract calls.
3. Mapping contracts deployed on ETH are verified for operational privileges through the privilege management mechanism in Dante Network cross-chain contract calls.
4. The standardized NFT contracts deployed on ETH can be directly displayed in Opensea through search.

The NFT display is just one of the very small cases. From a broader perspective, we hope Dante can provide similar convenience to more emerging public chains now and in the future, and thus help them get through the startup period when the ecological infrastructure is not yet perfect.