UDID Network
A Unified Decentralized Identity System

Technology Whitepaper v1.1.0

Sponsor: dnsDAO

Email: team@dnsdao.org

Abstract

A decentralized naming system should have a function to map existing and future blockchain system identities (a string generated by the current blockchain identity public key) to names that are easy to remember and use. The naming system can also not be independent of existing naming systems and must be compatible with existing ones. The traditional naming system - or domain name system as we know it - is limited by the processing power of the conventional information system and has a typical tree structure. The traditional naming system did not consider the problem of security, which led to the emergence of attacks. Another centralized system CA was introduced to solve the problem of attacks, making the traditional naming system ecologically unstable and vulnerable to control. The use of a decentralized naming system can fundamentally solve the problem of security, instability, and control. The decentralized naming system allows any person or organization to join or exit the naming system according to the code agreement, with no restrictions from any country, organization, community, or individual. We use a service-driven consensus algorithm based on which nodes get more chances to create a new block by providing external query services. We denote this consensus algorithm as DPoS (Delegated Proof-of-Stake).

Introduction

As technology advances and hardware capabilities continue to upgrade, people's requirements for information systems have slowly shifted from the initial desire for increased efficiency to the need for fairness and security. The emergence of blockchain systems comes in response to this demand. The principles of a blockchain system are decentralization, privacy, and security. Web 3.0, built on top of the blockchain system, solves the problem of large companies keeping users away from their rights brought by Web 2.0 and the issue of the centralized nodes collecting and tracking users' behavior.

A blockchain system is a system built based on a cryptographic algorithm. The current identities (blockchain user, contract, token, etc.) in the blockchain system is identified by the public key of the blockchain's cryptographic algorithm. The public key is a long, irregular string. As the blockchain system is recognized among more people and applied in more fields (e.g., the widespread acceptance of on-chain applications), this current representation of blockchain identity has dramatically hindered the broad application of blockchain.

This paper aims to design a new blockchain-based naming system that maps blockchain identities (represented in the form of public keys) to easy-to-read and remember names. At the same time, the naming system will provide end-to-end security, eliminating the need for an additional set of CA systems and simplifying usage of the system.

Basic Algorithm

1.1 Ed25519

We choose the ED25519 algorithm as the base algorithm of a decentralized naming system based on two main reasons: the first is the deficiencies of EcDSA's signature, and the second is that EdDSA has more comparative advantages.

What are the shortcomings of EcDSA? The main ones are as follows:

1) the insecure use of random numbers in the signing process (including the difficulty of implementing a secure random number generator and random numbers correct usage by programmers), which may lead to key leakage;
2) forgeability of ECDSA signatures;
3) possibility of signature forgery: if there is no need to provide a signature message, the signature value corresponding to the private key can be forged based on the existing signature value;
4) low signature efficiency, if compared to Ed25519.

Ed25519 algorithm solves the above deficiencies. Its main advantages are as follows:

1) the implementation is optimized to achieve high performance on a variety of computing platforms;
2) the signature process does not require a unique random number, which can avoid the security problems caused by random numbers;
3) there are no branching and secret data indexing operations, with better immunity to side-channel attacks, etc.;
4) the public key and signature values are small (Ed25519 public key is 32 bytes, the signature value is 64 bytes);
5) computation formula is complete, no need to perform point verification operation for untrusted points;
6) collision resist, a collision of weak hash functions will not break the signature mechanism (PureEdDSA).

1.2 X25519

X25519 algorithm is a kind of ECDH algorithm, mainly designed to exchange keys, using the horizontal coordinate point group of Ed25519 algorithm, it can naturally generate X25519 keys. Thus, it achieves the same key for both signature and key exchange and is used to encrypt end-to-end data encryption.

2. Decentralized Names

2.1 The characters allowed in decentralized names include:

1) a-z, 26 letters of the alphabet;
2) 0-9, ten numbers;
3) "-" underscore.

2.2 Decentralized naming basic principles:

1) names are not case-sensitive;
2) "-" character cannot be at the beginning or the end;
3) maximum 67 bytes;
4) names use "." character link, e.g., abc.dns.

2.3 The decentralized naming system defines names as three types:

1) top-level domain names have less than five characters (e.g. dns);
2) common top-level domain names have five characters or more (e.g., google);
3) subdomain names use "." connected names (e.g., such as abc.dns).

3. Consensus Algorithm

In blockchain systems, a consensus is a process by which individual consensus nodes agree on the outcome of transaction execution. The decentralized naming system uses the algorithm of DPoS to generate consensus. Unlike other systems, the decentralized naming system can have prosperous ecology only if a good mapping query service is provided.

3.1 Node Joining

To improve the impact of new nodes joining the network on the existing system, the decentralized naming system must pass network-wide verification when a new node joins the network. We limit the difficulty of joining new nodes by calculating the HashCash mechanism. Depending on the size of the network-wide nodes, such nodes will update the difficulty value of joining the network. The relationship between difficulty and the target value is as follows:

Difficulty value = Maximum target value / Current target value (2.1)

The maximum target value and the current target value are 256 bits long, and the maximum target value is the target value when the difficulty is 1, i.e., 2^{224}. Assuming that the current difficulty is D, the arithmetic power is P, the current target value is C, and the expected time to join the network is T, then:

$$C = 2^{224}/D \tag{2.2}$$

The arithmetic power required to join the net network at this point is:

$$2^{256}/C = 2^{32}*D \tag{2.3}$$

The relationship between arithmetic power and time is

$$P = 2^{32}*D/T \tag{2.4}$$

The decentralized naming system periodically adjusts the difficulty value of joining the network so that each newly joined node takes about 10 minutes of Hash computation time.

3.2 Consensus Nodes

Nodes that join the network generally cannot participate in the decentralized system consensus block producing; only nodes that pass the election can become consensus nodes.

The election is based on the number of names, weights, and sub-names pledged to the node, to be precise:

1) the number of collateralized top-level names (denoted as TopQuantity);
2) the number of sub-names under the pledged top name (denoted as TopSubQuantity);
3) the length of consecutive renewals of the pledged top-level name (denoted as TopTimeInterval);

4) the number of pledged normal top-level names (denoted as NormalQuantity);
5) the number of sub-names under the secured normal top-level name (denoted as NormalSubQuantity);
6) the length of the consecutive renewals of the pledged common top-level name (denoted as NormalTimeInterval);
7) the number of sub-names pledged (denoted as SubQuantity);
8) the duration of continuous renewal of the pledged sub-names (denoted as SubTimeInterval).

The decentralized naming network is re-elected every 24 hours, and the weight of the election (W) is calculated as follows

$$W = \sum_{i=0}^{n} TopSubQuantity[i] * TopTimeInterval * 128 +$$

$$\sum_{j=0}^{m} NormalSubQuantity[j] * NormalTimeInterval * 16 +$$

$$\sum_{k=0}^{l} SubTimeInterval[k]$$

In particular, the renewal time is calculated from the most recent consecutive renewal time, and once there is an interruption in between, the renewal time will be recalculated.

Using the above calculations, each node can accept a pledge of names (this pledge does not change the ownership of the domain name), and each name is allowed to be pledged to only one node. Each node can calculate (elect) whether it will be a consensus node in the next epoch based on the weight of the pledge. The 65 (possibly greater than 65) nodes with the highest weight in the entire network become consensus nodes.

The nodes elected as consensus nodes must be guaranteed to be online for a certain time. Suppose the node that should create a block at a certain moment does not perform its duties. In that case, it will be disqualified as a representative, and the system will continue voting (random algorithm selection) to elect a new representative to replace him.

The elected nodes create blocks in order and are rewarded for producing them.

Each node gets the right to create a block only by voting, which ensures the security and decentralization of the whole network.

3.3 Block creating sequence

After the election of consensus, nodes is completed, we need to decide the order of node blocking, which is executed in the order of node addresses from smallest to largest in principle. We introduce the Verifiable Random Function (VRF) algorithm to let the larger address values be sorted fairly.

The verifiable random function is an application of zero-knowledge proof, that is, in the public-private key system, the person holding the private key can use the private key and a known message to generate a random number according to specific rules, and the person holding the private key can prove to others the correctness of the random number generation without revealing the private key. The exact process of a verifiable random function is as follows:

1) Generate the key pair, A generates the public-private key pair (sk,pk) locally;
2) compute value: A takes the private key sk and the message m as input and computes:
   value = VRFval (sk,m);
3) Calculate proof: A takes private key sk and message m as input and calculates:
   proof = VRFp (sk,m);
4) verify value: B verifies the received value and proof, verify that value can be calculated from proof:
   value = VRFproof2value(proof);
5) verify proof: B to verify the received proof, verify proof can be calculated from A's private key sk and m to get:
   True/False = VRFverify (proof,sk,m).

The sorted random numbers are generated by the 7 least weighted consensus nodes, which provide the random numbers. Using the generated random numbers, the sorting order is decided.

The process of random number generation and sorting is as follows:

1) Each node (the node with the smallest weight) generates a pair of public and private keys (ski, pki) and calculates a random number m by the current epoch information and the previous epoch information, and since the information is all from the chain, the nodes all calculate to get the same random number;
2) The node uses its private key and the computed random number m. The result of the computation and the proof.
   $hash = VRF_{val}(sk,pk)$
   $\Pi = VRF_{p}(sk,m)$
3) Normalize [0,1], where hashlen is the number of bits of the computed hash
   $d = hash/2^{hashlen}$
4) Construct the binomial distribution, where p = t/w is the single probability, t is the expected value, and w is the total weight.
   $B(k;w,p) = ({}^{w}k)p^{k}(1-p)^{w-k}$
5) Constructing function:
   $$f(j) = \sum_{0}^{j} B(k;w,p)$$
6) Validation result:
   $\Pi = VRF_{proof}(pk,m)$
   $hash = VRF_{proof2value}(\Pi)$

3.4 New Era

The decentralized naming system requires the re-election of new consensus nodes every 17,280 block heights and determines the block-out order of the elected nodes.

The first sequential node of the newly elected consensus node out of blocks is n*17820+1.

If the newly elected consensus node has a faulty node, the block height will be skipped directly, so the decentralized block height is discontinuous.

During the process of electing nodes, the network needs to pause for 10 seconds out of the block time in order to ensure a safe switch to the new epoch, which will reduce the TPS of the network.

3.5 Genesis block

The decentralized naming system defines top-level names, common top-level names in the creation block. It then records in the creation block the nodes that have been created, the names that have been registered, and the consensus nodes of the first epoch, the token issued by the whole network, and the number of tokens reserved.

4. Registration and query protocols

A decentralized naming system determines that a name belongs to a user for a certain period through a registration transaction. Once registration is complete, the user can use that name to map to all mapped services supported by the system. Once a user is mapped to a service using the blockchain system, the corresponding service can query the user by name to see if the service is enabled.

4.1 Registration

Registration is a kind of transaction in the system, a special kind of name transaction that creates a unique name in the system. The system sets up three types of name registration fees (top-level names, regular top-level names, and second-level and higher names), and accounts submitted for registration need to have sufficient registration fees and a fixed transaction gas fee. To register a top-level name is to create a special contract in the blockchain that accepts other users to initiate registration of second-level and above names.

In addition to registering top-level names, within the top-level name contract, the contract provides registration of second-level and higher names, registration of second-level names, like top-level names second-level and higher names, and allows users to configure mapping services that the community allows to be open.

4.2 Mapping

Mapping refers to the user's registered name, mapped to a real-world identity that is difficult to remember. The current system can be mapped to the following types.

1. blockchain addresses

2. IP address

3. NFT address

4. mail server

255. custom type

Type	Class	TYPE	V
Blockchain Address	ETH 、 BTC 、 TRON...	ChainID	Wallet Address
Network	Etheruem...	IPv6、IPv4、MAC	Address
NFT	ETH 、 BTC 、 TRON...	ChainID	Nft description with json format
Mail Server			
Customized			

Mapping types are added to the blockchain utilizing proposals in the community. For types not supported in the blockchain at the time, users can customize them, and the naming service does not provide the service of parsing custom types; users need to parse them by themselves.

4.3 Name hash

In the blockchain system, for easy storage and calculation, all names correspond to a hash, which is obtained by calculating different levels of hashes.

The formula for calculating the name hash is as follows:

Suppose an N-level name is denoted as DNamen, the Nth-level name is denoted as Segn, and the hash of an N-level name is denoted as Hash(DNamen), then the hash of an N-level name is

Hash(DNamen) = keccak256(keccak256(Seqn)^Hash(DNamen-1))

When the name is the top-level name, that is, when N==1

Hash(DName1)=keccak256(Seq1)

Example: avatar.dns is calculated as

Hash(dns)=keccak256(dns) ==

a1ed537a640b942ced4023e4bcce52645fcedf46b28b966a036b9fe422a83e89

keccak256(avatar) ==

d1f86c93d831119ad98fe983e643a7431e4ac992e3ead6e3007f4dd1adf66343

keccak256(avatar)^Hash(dns) ==

70153fe9bc3a85b634cfca675a8df527418416d4516140890314d2358f5e5dca

Hash(avatar.dns)==

20be0f5dcb5c77f09f09ba5c7450d9bce1910b96044f8dbd114d49bed31c9ccb

4.4 Queries

The decentralized naming system provides mapped query services. To reduce the impact of the query service on the whole network, the decentralized naming system distinguishes nodes into consensus nodes and service nodes. The service nodes will provide the mapped query service.

4.4.1 Restful API

A forward query is to query the content of mapping by name, the specific request and response are as follows.

Request:

Protocl	https
Approach	GET
Port	19023
Path	
Content-type	x-www-form-urlencoded
Name	

Answer:

Content-type	x-www-form-urlencoded
Class	
TYPE	
Content	address

Reverse lookup means the use of an address (wallet address, IP address) to look up a name when the name cannot be known.

Request:

Protocol	https
Approach	GET
Port	19023
Path	
Content-type	x-www-form-urlencoded
Class	
TYPE	
Address	

Answer:

Content-type	x-www-form-urlencoded
Name	

4.4.2 DNS Protocol Queries

The traditional DNS protocol defines 5 Classes and 20 Types in RFC1034 and RFC1035, which are insufficient to represent the existing decentralized naming. Traditional DNS defines the Type:

Type（QType）	Value	Meaning
A	1	a host address
NS	2	an authoritative name server
MD	3	a mail destination (Obsolete – use MX)
MF	4	a mail forwarder (Obsolete – use MX)
CNAME	5	the canonical name for an alias
SOA	6	marks the start of a zone of authority
MB	7	a mailbox domain name (EXPERIMENTAL)
MG	8	a mail group member (EXPERIMENTAL)
MR	9	a mail rename domain name (EXPERIMENTAL)
NULL	10	a null RR (EXPERIMENTAL)
WKS	11	a well known service description
PTR	12	a domain name pointer
HINFO	13	host information
MINFO	14	mailbox or mail list information
MX	15	mail exchange
TXT	16	text strings
AXFR	252	A request for a transfer of an entire zone
MAILB	253	A request for mailbox-related records (MB, MG or MR)
MAILA	254	A request for mail agent RRs (Obsolete – see MX)
*	255	A request for all records

5 Classes defined by traditional DNS:

Class（QCLASS)	Value	Meaning
IN	1	the Internet
CS	2	the CSNET class (Obsolete - used only for examples in some obsolete RFCs)
CH	3	the CHAOS class
HS	4	Hesiod [Dyer 87]
*	255	any class

The decentralized naming system supports and inherits all Class and Type of query protocols, extends the Class and Type of DNS, proposes to increment the decentralized new types in DNS protocols from the 128th of Class, and the Type class is extended to 64bits of numbers in order to support the chainID type of blockchain.

New DNS query Class:

Class（QCLASS)	Value	Meaning
BTC	128	BTC type address
ETH	129	ETH type address
NFT	130	NFT type info

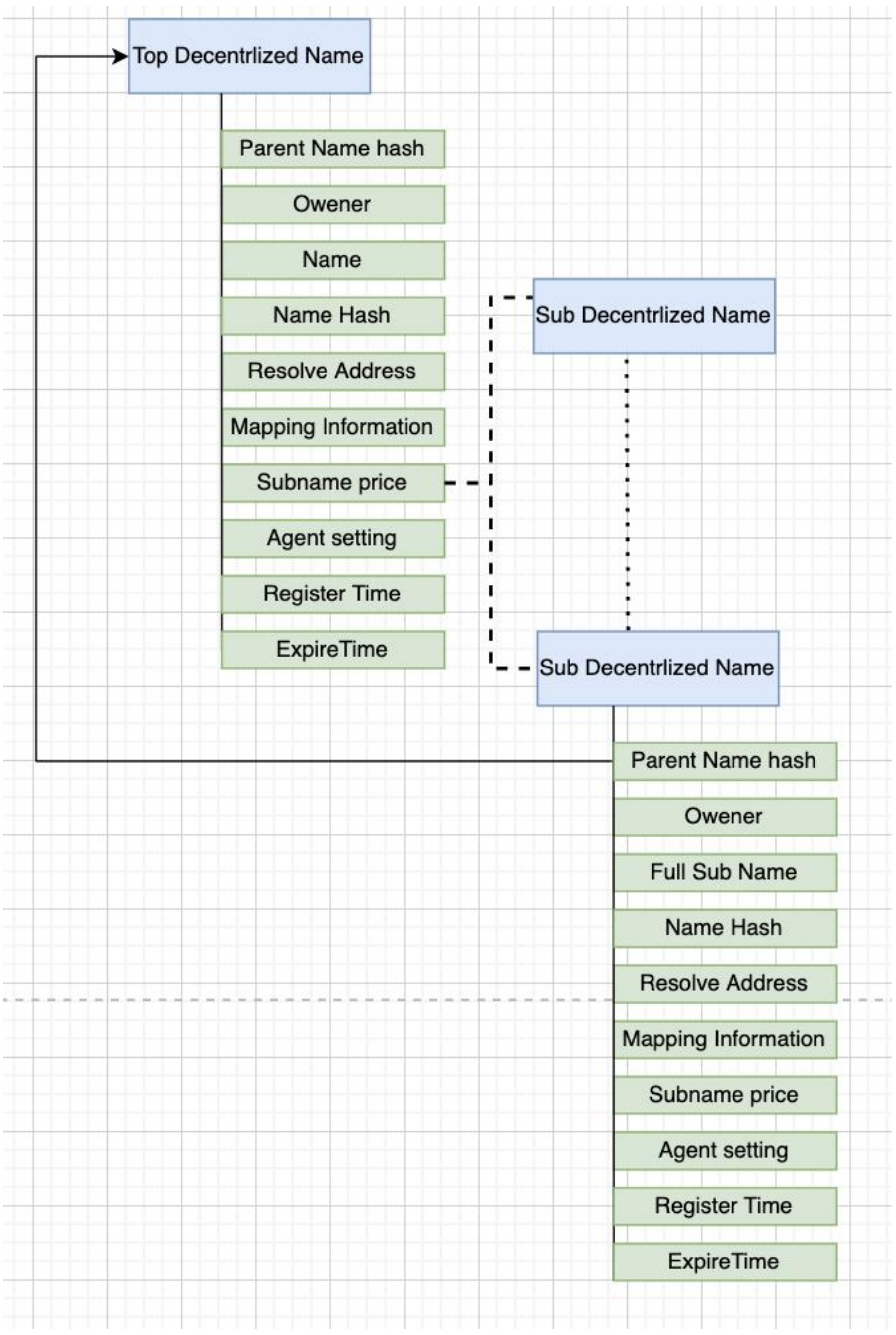
New DNS query Type:

Type（QType)	Value	Meaning
ETH	1	Eth address
Kovan	69	Kovan eth address
EOS	59	Eos address
...		

5. Transactions

The decentralized naming system has three types of transactions which are normal transactions, name transactions, and contract transactions. In the system, top-level names and sub-names are a resource that anyone can obtain by initiating a name transaction. The system creates a namespace belonging to each user who initiates a transaction and succeeds.

5.1 Decentralized Naming System Namespace



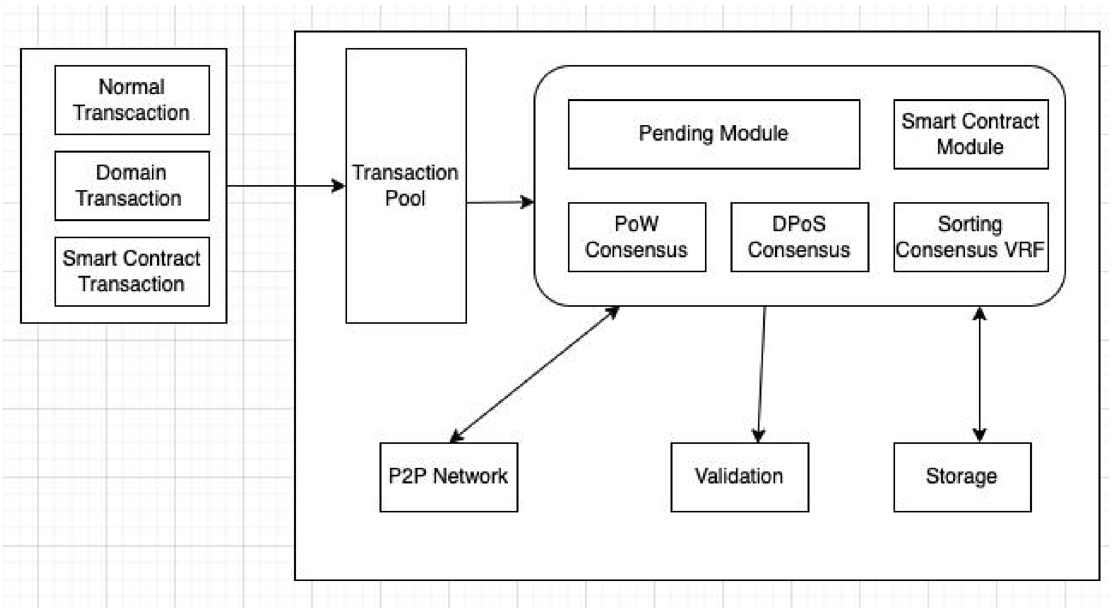
As shown above, the DNS system creates a configuration and mapping space for each successful name application, where Parent Name hash points to the hash of the name at the previous level. The owner is the current name holder, if this name is in validity, only the signature of the owner holding this name can change the owner to another address. name is the name held by the Owner, Name hash is the full name hash, which is the full hash of this name, see section 4.3 for the generation of the hash. Resolve Address is the address of the forward and reverse resolution service. Mapping information is the content that this name is mapped to, the currently supported mapping content. Refer to section 4.2, and provide the switch configuration on whether to allow reverse resolution service. Allow each holder to modify the registration price of the sub-name, allow configuration of a proxy registration, and allow other proxies to help register the sub-name. Holders hold names in relation to the time of purchase, if the Expire Time is passed and not renewed, the name will be grabbed by other users.

Currently, there is a possibility that when the registration time of a sub-name is longer than the registration time of the top-level name, there is a situation that the top-level name does not renew, but the sub-name still renews; if the sub-name still renews without the top-level name, it renews with the price set by the system.

In the case that there is no top-level name, the system allows the registration of sub-names, and at this time, the registered sub-names are rented using the price set by the system.

5.2 Decentralized name system node transaction processing

As shown in the figure below, the decentralized system collects all types of transactions and puts them into the transaction pool, executes different consensus algorithms according to the transaction types, and finally discharges the blocks.



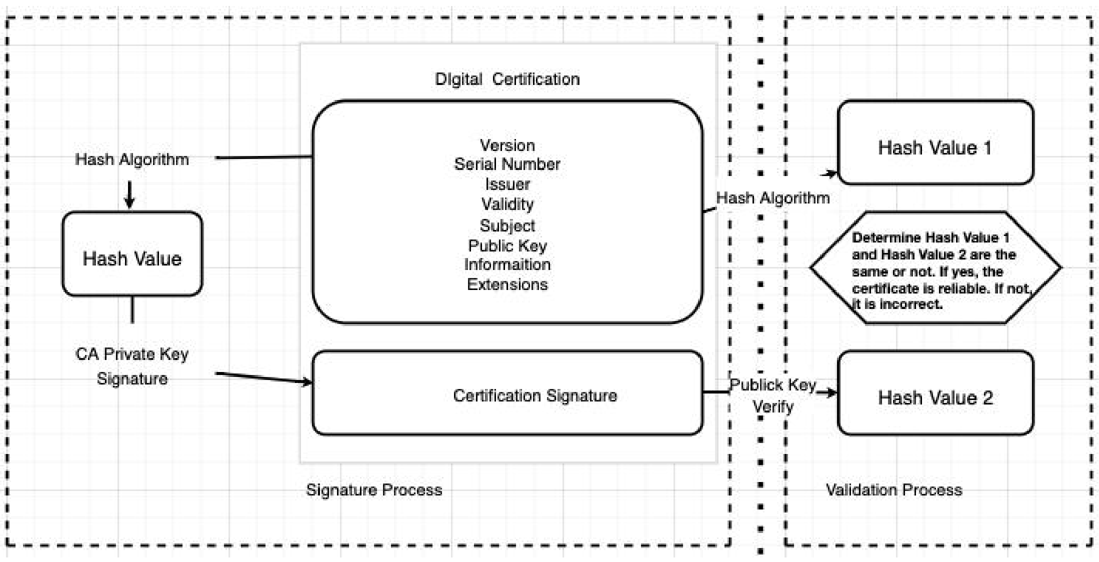
The consensus module has three parts, where the PoW algorithm is used with ordinary nodes to join the decentralized naming system, DPoS is used to elect 65 consensus nodes, and the sorting consensus VRF is used to sort the elected nodes.

6. Security

The traditional DNS system is vulnerable to attacks and has used TLS with certificate issuers to issue certificates to secure DNS from attacks over a long period of time.

A certificate issuing authority is actually a trusted centralized authority, and such an authority bears a huge credit risk and increases the cost for users.

The simple working principle of traditional CA-issued certificates is as follows:



A digital certificate is a series of documents defined using the X509 standard and composed of content according to certain criteria. The CA issues and uses the following steps.

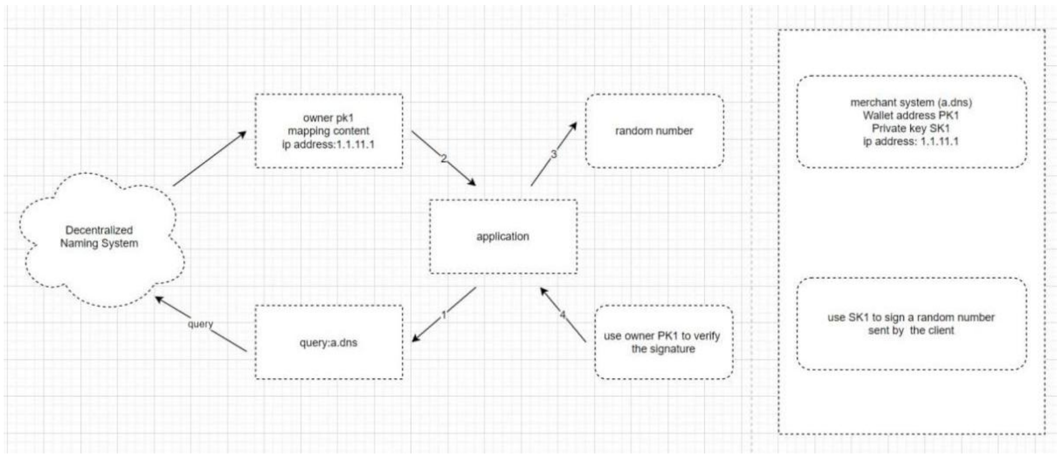
1) the CA types the holder's public key, usage, issuer, valid time, and other information into a package and then does a hash calculation on this information to get a hash value;
2) the CA signs the hash using its own private key to generate the CA's signature string;
3) the CA adds the signature string to the file certificate to form a digital certificate;
4) the client will first use the same hash algorithm of CA to calculate the hash value of the content, hash value1;
5) the browser or operating system is generally integrated with the CA public key certificate. The client receives the certificate, using the CA public key to decrypt the signature and obtain the original hash value2;
6) finally, compare the values of H1 and H2 to see if they are equal, and if they are equal, prove that the other party is trustworthy.

Using CA can ensure that an entity is a trusted one. Through TLS protocol, the proof process is encrypted to ensure that the middle does not eavesdrop.

When the decentralized naming system provides query service, the public key of the node is built into the client. When the client queries, it only needs to determine whether the signature of the service node is correct, and if it is, it will get the content of the name mapping and the public key bound to the name.

When users use the merchant system, they need to use the public key obtained from the decentralized naming system to verify the identity of the merchant system.

The process of a decentralized naming system using public key binding and replacing CA is shown in the following figure:



As shown on the way, where step 1, the communication between the client and the decentralized naming system needs to be trustworthy, which requires the client to have the ability to securely synchronize the public keys of the decentralized nodes. The community will provide some nodes for the client to synchronize the public keys of the nodes.

When the client requests a.dns, it will add random numbers to the request message; when the decentralized naming system receives the query request, it will make a hash of the requested content and sign the hash value. When the client receives the reply from the decentralized naming system, it first verifies whether the signature is correct, and if it is, the query data is proved to be reliable.

7. Incentive

The decentralized naming system maintains the operation of the system by incentivizing nodes. Unlike other systems, the naming system needs to provide excellent mapping query services, which significantly increases the operating cost of top-level names. To compensate for this cost disadvantage, the decentralized naming system rewards a large portion of the registration fee to miners.

7.1 Registration fees

Registration is the process by which a name is generated from the system. The decentralized system does not pre-determine any names (top-level names, sub-names), and generating a name in the system requires a registration fee that is higher than the gas fee. The registration fee rewards all miners except for a small portion set aside for the community. Due to the special nature of the registration fee, the registration fee is not awarded to the block nodes but to all miners in the current epoch (distributed equally).

7.2 Transaction fees

All writes in a decentralized naming system are performed by initiating transactions. In order to encourage miners to pack transactions into the system, the system needs to consume the token on the account that initiated the transaction for each instruction executed, and this fee also becomes gas. Transaction fees are rewarded to the node that gave the block.

The transaction fee also discourages certain malicious accounts from initiating attacks on the system. The transaction fee is not refunded when the execution of a transaction fails, and this design approach effectively prevents malicious attacks.

7.3 Block Creating Reward

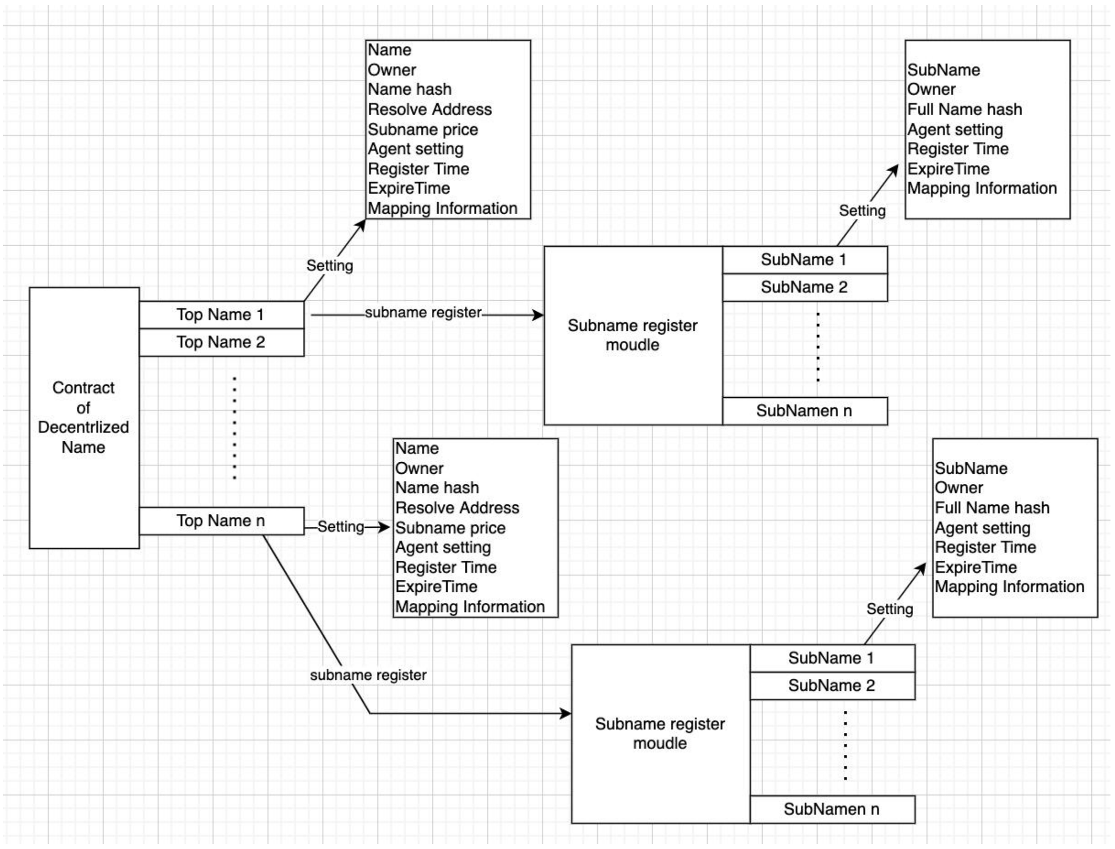
Decentralized trading system early starts nodes, due to the small rewards from transaction fees and registration fees, the decentralized naming system will provide more out of block rewards in the early stage to encourage nodes to participate in the system operation. As the network transaction fees grow, the community can adjust the block-out rewards to get a better balance.

8. Decentralized naming system based on Ethereum-like contracts

In order to verify the community governance model of the naming system, the decentralized naming system will first implement registration, management, transaction, and ecological governance of top-level names on Ether-like, including registration of sub-names, sub-name bound d-app kind of applications and more.

The decentralized name architecture implemented in Ether is the first to be able to write seamlessly to the Genesis blockchain when the decentralized naming system is launched.

The registration and mapping of decentralized names is implemented using a contract structure, as shown in the following figure:



As shown above, the contract internally implements the registration and configuration of top-level names, and the contract also implements the registration and configuration of sub-names. In each name (top-level name and sub-name), there is an owner; only the owner user provides the signature to configure the name mapping and modify the configuration; also, only the owner user can transfer the name to other users.

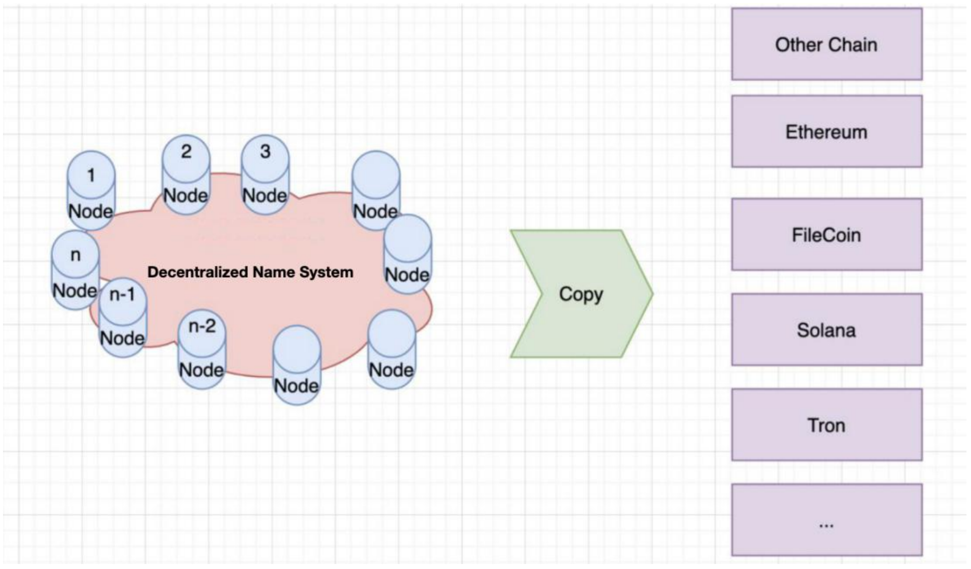
The contract allows the registration of sub-names without a top-level name, and if there is no top-level name, the registration fee required by the user to register a sub-name is determined by the default value specified in the contract (community consensus).

In addition to enabling name registration and configuration, the decentralized naming system, which allows name resale, enables users to submit their registered names to a pending order for a trading contract. The name transfer of pending orders no longer checks the owner and only requires the user proposing the transfer to provide enough tokens to transfer the traded name under their account.

9. Usability of decentralized digital identity in other public blockchains

Each public blockchain is a self-forming system in its application. If it needs to use decentralized digital identity, cross-chain mechanism needs to be used.

If each application handles the cross-chain of digital identity by itself when using decentralized digital identity, it will burden the application and is not conducive to the development of decentralized digital identity ecology. To solve the problem mentioned above, the decentralized digital identity naming system provides a mechanism by which users can easily copy their domain names to the public chain where the application is developed. As shown in the figure below



Decentralized digital identity naming system that elects signed consensus nodes at each epoch (the number of signed consensus nodes is much lower than the number of nodes in the system itself) synchronizes the signed consensus node public keys to other chains. Digital identities need to be replicated when the decentralized system provides the signatures of the signature consensus nodes and gets the signatures verified on the public chain contract.

9.1 Node Whitelisting

The digital identity synchronization public chain whitelists the public keys of the decentralized digital identity naming system consensus nodes. Each epoch synchronization of signed consensus nodes requires the signatures of two-thirds of these nodes for the synchronization to succeed.

9.2 Signature Consensus Nodes

When synchronizing decentralized digital identities, public chains check by verifying the consistent signatures of signature consensus nodes. The decentralized digital identity naming system has more consensus nodes, and public chains need to verify too many signatures when synchronizing a digital identity, which brings great signature verification cost.

In order to reduce the cost of signature verification, the decentralized digital identity system elects signature consensus nodes in each epoch and synchronizes the public keys of signature consensus nodes to other public chains. Synchronizing the signature consensus node public keys requires two-thirds of the nodes in the node whitelist for signature verification to be successfully synchronized.

9.3 Multi-signature

The elected signature consensus node needs to get the signatures of all nodes (multi-signature) to be packaged and synchronized to other public chains by the outgoing block node. When synchronized to other public chains, the digital identity needs to be signed by the signature consensus nodes to be packaged and verified on other public to be verified on other public chains for successful synchronization.

10. Summary

In this paper, we proposed a decentralized digital identity naming system, aiming to solve the current digital identity naming application challenges through its successful application.

We propose a decentralized digital identity naming system based on a blockchain incentive approach to address the shortcomings of naming digital identities through centralized institutions in the real world and solve the problems of security, instability, and vulnerability to control. Incentivize organizations and individuals who contribute to the decentralized digital identity naming system using the DPoS consensus algorithm. Stabilize node volatility in the network through PoW. Allow friendly interfacing of decentralized digital identities across major public chains using multi-chain replication.

11. References

[1] S. Satoshi Nakamoto. (2009) "Bitcoin: a peer-to-peer electronic cash system." Available at: https://bitcoin.org/bitcoin.pdf/.

[2] D. Larimer. (2014) "Delegated Proof-of-Stake (DPOS)." Available at: https://bitshares.org/technology/delegated-proof-of-stake-consensus/.

[3] A Verifiable Random Function With Short Proofs and Keys. available: https://cs.nyu.edu/~dodis/ps/short-vrf.pdf