EasyChair Preprint

№ 4045

A Secure Domain Name Resolution and Management Architecture Based on Blockchain

Wenfeng Liu, Yu Zhang, Lu Liu, Shuyan Liu, Hongli Zhang and Binxing Fang

August 16, 2020

A secure domain name resolution and management architecture based on blockchain

Wenfeng Liu, Yu Zhang, Lu Liu, Shuyan Liu, Hongli Zhang, Binxing Fang
Computer Science and Technology
Harbin Institute of Technology
Harbin, China
{15b903031, yuzhang}@hit.edu.cn, 19s003112@stu.hit.edu.cn, {1130320121, zhanghongli, bxfang}@hit.edu.cn

***Abstract*—The domain name system (DNS) is the infrastructure of many services and applications, thus the availability and consistency of the domain name resolution process are crucial but have long troubled DNS. The availability problem is caused by a denial-of-service (DoS) attack or a single point of failure (SPOF). The consistency problem originates from the lack of a forced data synchronization mechanism between authoritative server replicas or between parent/child authoritative servers. We proposed a novel blockchain-based domain name resolution and management architecture named FI-DNS to solve the above problems fundamentally. FI-DNS solves availability and consistency problems in the name resolution process from the mechanism level and guarantees the authenticity and integrity of name resolution results by using public-key cryptography. FI-DNS also supports root zone collaborative management based on smart contracts, which is compatible with the current governance model led by Internet Corporation for Assigned Names and Numbers (ICANN). We implemented the prototype system to prove the feasibility and effectiveness of the FI-DNS architecture. We built an experimental environment with real domain name data, evaluated the name resolution performance and stability of the FI-DNS prototype system, and compared the prototype system with DNS.**

***Index Terms*—DNS, name resolution architecture, blockchain, availability, consistency**

I. Introduction

DNS is the key infrastructure that provides services for domain name resolution on the Internet. Internet applications that use domain names, such as the web and e-mail, rely on these services to translate domain names into IP addresses correctly. At present, DNS continues to evolve and play broad roles on the Internet, such as providing search engines with anti-spam [1], replica server selections for the content delivery network (CDN) [2], and DNS-based authentication of named entities (DANE) [3]. The wide range of applications highlights the importance of domain name resolution. Two critical issues with the domain name resolution process: *availability issue* and *consistency issue* have persisted for a long time, which are limited by the DNS architecture itself and have yet to be fundamentally solved.

The name resolution availability problem refers that authoritative servers of a domain cannot respond to the name resolution request. The cause of unavailability can be a single point of failure (SPOF) risk (e.g., domain *.microsoft.com was not available on 24-Jan-2001 [4]) or suffer a denial of service (DoS) attack [5]. However, the primary reason for both is the insufficient deployment number of servers. The DNS specification (section 5 in [6]) requires a domain operator to deploy more than two authoritative server replicas and to ensure that replicas are in different physical locations/sites and use different DNS software implementations. However, maintaining multiple heterogeneous authoritative servers in different locations is too expensive for small operators to afford.

The name resolution consistency problem refers that the resource record set (RRSet) of a domain name is inconsistent in different storage locations. Inconsistency includes the following three cases: (1) the glue records in the parent zone is inconsistent with the storage in the child zone. This phenomenon called "lame delegation" has been studied by [7], which we also call it "delegation inconsistency" phenomenon. (2) The same RRSet is inconsistent between different authoritative server replicas, which we call the "replica inconsistency" phenomenon. (3) The cached RRSet in the recursive resolver is inconsistent with that stored in the authoritative server, which we call the "cache inconsistency" phenomenon.

Cases (1) and (2) usually caused by misconfiguration, and case (3) is caused by the DNS that only guarantees weak consistency. Nevertheless, cache abuse behaviors [8] [9] and cache poisoning attacks [10] will amplify the degree of the inconsistency (3). In summary, the primary reason for the above three inconsistencies is that DNS stores RRset of a domain name in multiple DNS components (replica servers and cache servers) to provide hierarchical domain name resolution (case (1)) and accelerate name resolution process (cases (2) and (3)). However, there is a lack of an inherent forcing mechanism in current DNS architecture to ensure that the RRset in different storage locations keeps consistent.

To fundamentally solve the availability and consistency problems, we propose an name resolution architecture and implement the corresponding prototype system (named FI-DNS). In comparison with the traditional DNS architecture, FI-DNS has the following characteristics (Figure 1):

Eliminate SPOF risk and Mitigate DoS attacks. FI-DNS consists of a blockchain network and a distributed file network, which contains plenty of peer nodes. Domain name data (RRSet) is stored in multiple peer nodes of distributed networks, and each peer node can independently provide responses to queries against a name service. The failure of

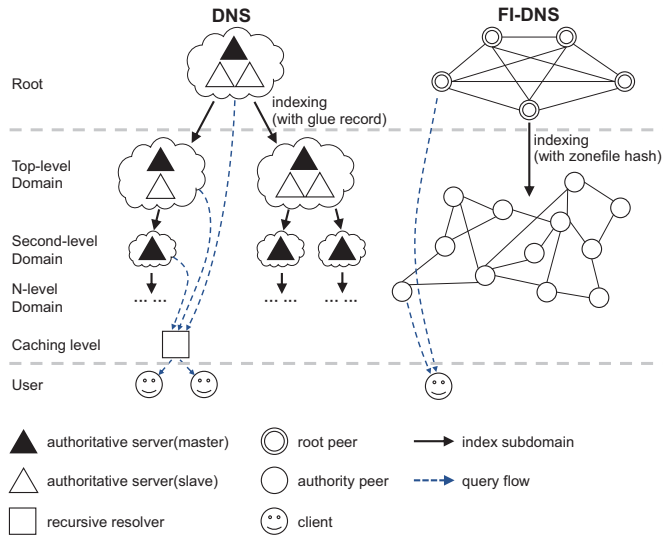


Fig. 1. Characteristics comparison between DNS and FI-DNS name resolution architecture. FI-DNS consists of a two-tiered distributed network, the first tier is a permissioned blockchain network and the second tier is a permissionless distributed file network.

a few nodes will not affect the services provided by others.

Eliminate delegation inconsistency. FI-DNS no longer stores glue records in parent zone but by storing the hash value (of the subdomain) instead. Indexing subdomains with unique hash values avoids the delegation inconsistency caused by storing glue records in authoritative servers of both parent and child domains.

Eliminate replica inconsistency. FI-DNS uses a consensus algorithm to synchronize data between peer nodes in the blockchain network. Zone files (store RRSet) are indexed by file hash values from peer nodes to ensure the client queries are consistent among all peer nodes.

Eliminate cache inconsistency. FI-DNS cancels the use of recursive resolver to avoid inconsistency caused by caching. [11] demonstrated that removing recursive resolver would only add $\leq 10\%$ name resolution delay to the client.

The main contributions of this paper include:

1) A new secure domain name resolution architecture is proposed, which attempts to fundamentally solve the availability and consistency problem through an enhanced mechanism of the architecture.
2) A new collaborative domain name management architecture compatible with the current ICANN-led governance model, which utilizes identity-based permissioned blockchain technology and blockchain smart contracts.
3) A prototype system named FI-DNS is developed, which proves the feasibility of the name resolution architecture. FI-DNS serves as a platform that provides the resolution-as-a-service, thus empowers small domain name operators with the ability to resist DoS attacks and SPOF risk. We designed an experimental simulation environment with real domain data to perform a performance evaluation and compared it with current DNS.

II. Background and Related Work

DNS provides a mapping of domain names to values. The value type can be an IP address, a hostname, or any text string. An authoritative server is a database for storing domain name data. The data of the domain name in each level are stored in their corresponding authoritative servers. Taking the DNS root domain as an example (Figure 1, left), the root zone data includes two parts, the domain data (of the root domain) and the index data (of the top-level domain). DNS currently uses the glue record [10] to index domain data of the subdomain, and the glue record contains the hostname and the IP address of the authoritative servers which stores the domain data of the subdomain.

A recursive resolver maintains a cache database that stores responses to all queries that it has issued and forces all clients to share the cache (Figure 1, left). The cache-shared recursive resolver is the notorious security weakness in DNS, and the Kaminsky cache poisoning attack [10] is a typical example of exploiting the vulnerability. Numerous studies have focused on solving security vulnerabilities brought by the caching mechanism. Part of that are devoted to increasing the cryptographic security protection of DNS, such as DNSSEC [12] and DNS-over-DTLS [13]. Some other studies [11] propose to eliminate the vulnerability by removing the recursive resolver itself, and our architecture borrows this idea.

Studies on the use of blockchain technology for DNS are relatively few. Namecoin [14] is the first (name, value) pair registration and transfer system based on Bitcoin technology. Blockstack Naming Service (BNS) [15] is an improvement of Namecoin. Limited by the technical characteristics of the permissionless blockchain network, Namecoin and BNS use completely acentric name resolution architecture. This unsupervised name system is entirely incompatible with the existing ICANN-led "apply-approve-supervision" governance model and cannot avoid damage caused by malicious cybersquatting to the current domain name owners. Unlike Namecoin and BNS, our architecture is designed based on the identity-based permissioned blockchain that is compatible with ICANN-dominated TLD management models. In addition, Bitcoin's slow transaction speed and low transaction confirmation efficiency have been criticized for a long time. Thus the performance of the BNS system based on Bitcoin will be completely limited by Bitcoin.

DecDNS [16] is a distributed DNS data storage solution based on blockchain technology, which is aimed to store domain name data in the blockchain system. FI-DNS focuses on solving the availability and consistency problems in the current DNS. FI-DNS changes the process of indexing subdomain data and provides consistency capabilities with the help of blockchain consensus mechanisms. FI-DNS not only stores domain name data through the blockchain, but also implements a collaborative root zone management architecture based on blockchain smart contracts, which is compatible with the current governance model.

III. Design of FI-DNS Resolution Architecture

In this section, we introduce the design of the FI-DNS resolution architecture (figure 2), including its two-tier distributed network, hash-based hierarchical data indexing, data storage format, domain name resolution and verification processes, and the smart contracts for root zone management.

A. Architecture Overview

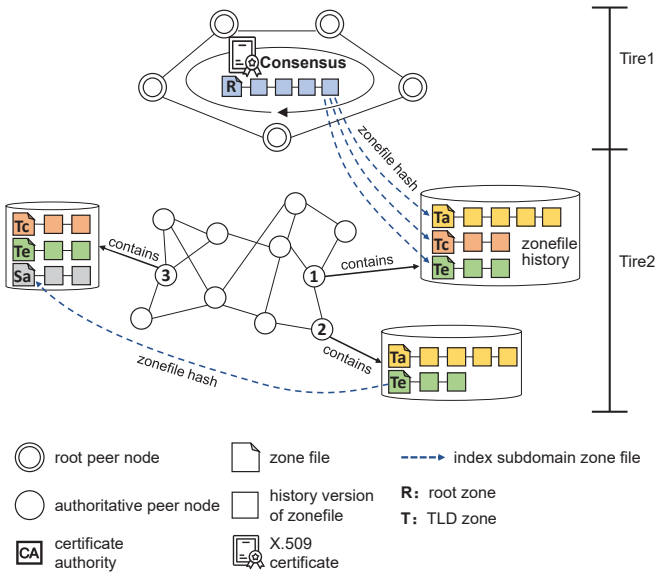


Fig. 2. FI-DNS name resolution architecture

The first tier is a permissioned network consisting of root peer nodes. The root peer node acts like a root server in DNS, responding to queries of root zone data. Besides, the root peer node joins the permissioned blockchain network in an identity-checked manner. All root peer nodes perform operations that describe the actions of the entity (root/TLD Authority) on the resource (TLD), such as TLD delegation and domain data publication. Operations are defined by consensus-based smart contracts of all root peer nodes. Accordingly, all entities that initiate operations through root peers are constrained by same smart contracts. Each entity is allowed to control the TLD resources that have been delegated to itself. Moreover, each entity can participate in the collaborative management of other entities' resources based on contract constraints.

The second tier is a permissionless distributed file network consisting of authoritative peer nodes. The role of this node is similar to the authoritative servers at various levels in DNS, providing authoritative responses to all stored domain data. In this network, all nodes can act as authoritative servers for all domain names(such as TLDs, second-level domains (SLDs), etc., other than root domain). The peers that store domain name data no longer distinguish between the master and the slave roles, all peers store data with equal identity. Figure 2 exhibits two authoritative peer nodes for the TLD *Ta* (yellow) and *Tc* (orange), three peers for the TLD *Te* (green), and one peer for the SLD *Sa* (gray). Each peer node can act as an authoritative node for multiple domains. For example, Peer No. 1 is the authoritative peer of three domains (TLD a/c/e), and Peer No. 3 is the authoritative peer of three domains (TLD c/e and SLD a).

Index subdomain data The prerequisite for implementing a hierarchical index of domain name data is that the index information of the subdomain's zone file must be included in the zone file of the parent domain. Only in this way can the client obtain the resolved data of the target domain name through multiple iterations of querying. DNS currently indexes zone file of subdomain by storing the IP address (glue record) of the authoritative servers that store the zone file of the subdomain. Since the authoritative server does not have a binding relationship with the stored data, the inconsistency between the parent and child domains (lame delegation) cannot be completely avoided. Figure 2 demonstrates that FI-DNS indexes the zone file of the subdomain by indexing the hash value of the zone file. The hash value of the zone file obtained by the hash function has a natural binding relationship with the zone file due to the hash function characteristics. When the client gets the hash value of the subdomain zone file from the parent domain, the zone file storing the subdomain data is uniquely determined.

B. Storage

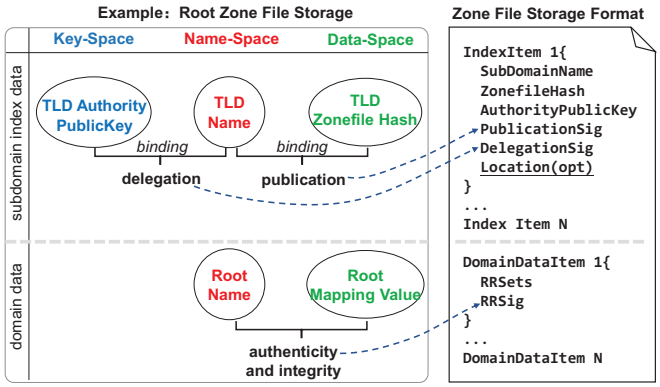


Fig. 3. FI-DNS zonefile storage format. Take the root zone file as an example to display the data stored in zone file.

FI-DNS uses the authority public key of the domain in the name resolution architecture to protect the authenticity of the name resolution results, which is reflected in the data structure of the zone file. The following are the three types of data stored in the FI-DNS zone file: 1) the domain name belonging to the hierarchical namespace, 2) the domain data representing the mapping value of the domain name, and 3) the public key representing the authority identity of the domain name. Take the zone file of the **root domain** as an example (Figure3):

The structure *IndexItem* is used to store index information for TLD. *IndexItem* uses five necessary fields, namely, *SubDomainName*, *ZonefileHash*, *AuthorityPublicKey*, *PublicationSig*, and *DelegationSig* to index TLD zonefile. The *SubDomainName* field stores the TLD name. The *ZonefileHash* field stores the hash value of the subdomain zone file. The

AuthorityPublicKey field stores the public key that uniquely identifies the TLD authority entity. The *DelegationSig* field is a digital signature generated by the root authority, and represents a delegation action, which means that the TLD (*SubDomainName*) is delegated to the entity holding the key (*AuthorityPublicKey*). The *PublicationSig* field is a digital signature generated by the TLD authority and is used to ensure the authenticity and integrity of the hash value *ZonefileHash* used for indexing TLD zone data.

The structure *DomainDataItem* stores data belonging to the root domain name. *DomainDataItem* contains two necessary fields. *RRSet* represents a resource record set with a format that conforms to the DNS protocol and consists of five fields: name, class, type, ttl, and rdata. The *RRSig* field is a digital signature generated by the root authority signing the RRSet field to protect the authenticity and integrity of the domain data. The format is in accordance with the DNSSEC protocol.

Optimize large zone file storage. FI-DNS performs the name resolution process by retrieving and downloading the zone file through hash values. For large zone file (such as $\geq 100MB$), the peer node takes lots of time downloading zone file, resulting in slow name resolution. To alleviate this situation, we propose a zone file partition algorithm to achieve the goal of resolving the target domain name by the only need downloading a small block of the zone file.

For a large zone file, we split it into *N* small zone file blocks for uploading and retrieving. For each *IndexItem* (figure 3) that constitutes the zone file is taken out, and the hash number is calculated according to its *SubDomainName* field. The *IndexItem* is recombined into the new zone file blocks based on the calculated hash number.

C. Domain Name Resolution and Verification

FI-DNS verifies while resolving. The red arrow indicates the verification of the domain name delegation. This step is implemented by verifying the delegation result (*DelegationSig*) of the subdomain name, *Verify([parent]AuthorityPublicKey, DelegationSig) = True* is required to guarantee the subdomain name delegated to correct registrant. The blue arrow indicates the authenticity check of the published zone file of the subdomain, implemented by verifying the publication result of the subdomain name, *Verify ([child]AuthorityPublicKey, ZoneFileHash) = True* is required to guarantee the authenticity of the hash index used to retrieve subdomain zone file.

Figure 4 shows an example of how to resolve the IP address for the target domain name *"www.example.com"* in the FI-DNS architecture and ensure the authenticity of the name resolution process. To resolve *"www.example.com"*, the root peer node searches the root zone file, then returns the *IndexItem* containing the TLD *"com"* to the client. The client first verifies (leftmost red arrow) that the TLD *"com"* has been delegated to the public key that represents the TLD authority entity. Thereafter, the client verifies (leftmost blue arrow) the zone file published by TLD *"com"* authority with the correct hash value. The client then uses the zone file hash to index the TLD *"com"* zone file for finding the

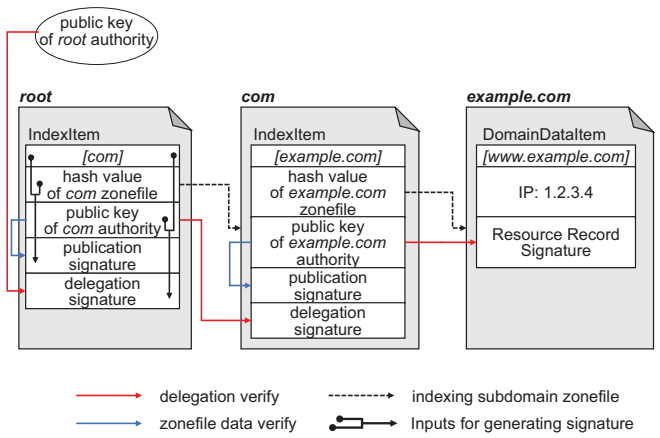


Fig. 4. Example of secured name resolution for domain 'www.example.com'. The red solid arrow indicates delegation verification, the blue solid arrow indicates data (domain name data or index data) verification, and the black dotted arrow indicates that the subdomain file is indexed with a hash value.

IndexItem containing SLD *"example.com"*. The client repeats the verification operation until the RRSet and RRSig of the IP address of *"www.example.com"* are obtained. Accordingly, the target domain name is securely resolved. The client uses zone file hash to retrieve TLD *"com"* zone file for finding the *IndexItem* containing SLD *"example.com"*. The client repeats the verification operation until the RRSet and RRSig of the IP address of *"www.example.com"* are obtained. At this point, the target domain name is securely resolved.

D. Smart Contract for Root Zone Management

TABLE I
SEMANTICS OF FI-DNS SMART CONTRACT

FI-DNS Operation	
Smart Contract	**Example**
	Initiator $\rightarrow$ **Output**$_{signer}$
DelegationPublication	RA $\rightarrow < TLD, TAPubKey >_{RA}$
DelegationTransition	TA $\rightarrow < TLD, TAPubKey >_{TA}$
DelegationRevocation	RA $\rightarrow < TLD, \emptyset >_{RA}$
DelegationRenewal	TA $\rightarrow < TLD, ValidTo >_{TA}$
DelegationRedemption	TA $\rightarrow < TLD, TAPubKey >_{TA}$
DataPublication	TA/RA $\rightarrow < ZoneData >_{TA/RA}$
DelegationValidation	FI-DNS $\rightarrow$ True/False
DataValidation	FI-DNS $\rightarrow$ True/False
RevokeOP	RA/TA $\rightarrow < TLD, OP\text{-}ID >_{RA/TA}$
ConfirmOP	RA/TA $\rightarrow < TLD, OP\text{-}ID >_{RA/TA}$

Smart contracts define all the operations that an authority can perform on a domain, as well as the precondition and results of those operations. The entity refers to all participants of the first tier network, that is, the root and each TLD authorities in the current DNS.

Table I shows the operation of all smart contract definitions supported by the current FI-DNS. In the second column of the table, RA represents the root authority, TA represents the TLD authority, and Signer represents the entity that digitally signs the operation output. All operations defined in the first Column of table I that can be divided into two categories:

The first category are the delegation operations and data publication operations. *DelegationPublication* binds the TLD to a registrant, making the registrant become the authority of the TLD. *DelegationTransition* replaces the authority of the current TLD. *DelegationRevocation* revokes the delegation for TLD, giving TLD an undelegated status. *DelegationRenewal* extends the TLD delegation period. *DelegationRedemption* redelegates the TLD that has expired to the original registrant. The above operations are consistent with the current ICANN-dominated DNS governance model [17]. The second category is data publication operation. The root authority publishes domain data through *DataPublication* operation, and TLD authorities publish indexing data (Figure 3) through it.

The second category are two basic operations, which are ComfirmOP and RevokeOP. ComfirmOP is used to vote on a submitted controversial operation, use the majority vote principle to resolve controversial operation, and reduce the long-time suspension of pending operations. RevokeOP is used to revoke previously initiated operations for the initiator.

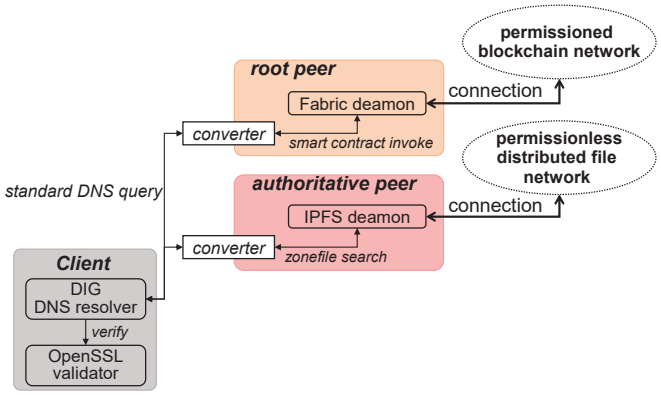


Fig. 5. The prototype system of the FI-DNS name resolution architecture

IV. Implementation

The FI-DNS prototype system consists of the following four functional parts, which are: root peer, authoritative peer, client, and converter. Figure 5 describes the implementation of the architecture.

- **Root peer** provides root zone data storage/resolution service and root zone data management service, which is implemented on the open-source blockchain development platform Hyperledger Fabric [18] (version 1.4).
- **Authoritative peer** provides authoritative zone file storage/resolution, which is implemented on the distributed file platform IPFS [19] (version 0.4.22).
- **Client** is responsible for domain name resolution and makes cryptographic verification to the name resolution process. The client implementation relies on two open-source components — Domain Information Groper (DIG) is used to issue queries compatible with DNS protocol and receive corresponding responses. OpenSSL is used to verify the digital signature of the response to ensure authenticity and integrity.
- **Converter** encapsulates the root peer nodes and the authoritative peer nodes, enabling all peer nodes to provide a unified interface (standard DNS query/response) for name resolution service.

V. Evaluation

We built an experimental environment for FI-DNS prototype system to evaluate the performance and stability of name resolution process. Real domain data was used to compare the FI-DNS prototype system with DNS.

A. Experimental Environment

The FI-DNS prototype system experimental environment (Figure 7) is configured as follows: 5 cloud servers and 1 local server. 1 cloud server from Linode cloud service provider, is used to simulate 13 root peer nodes. 4 cloud servers are used to simulate authoritative peer nodes, which are distributed in 4 data centers of Vultr cloud service provider. 1 local server is used as a client to initiate a domain name resolution query and verify the resolution process. Detailed server configuration and data initialization information list in Table II.

The 13 root peer nodes form a permissioned blockchain network, and the consensus copies of the root zone data are stored in all root peer nodes. The four authoritative peer nodes join the IPFS public network. To simulate various name resolution scenarios, the zone files of 100 TLDs (each zone file contains 10-10000 SLDs) are randomly stored in 3 authoritative peer nodes exclude Singapore node.

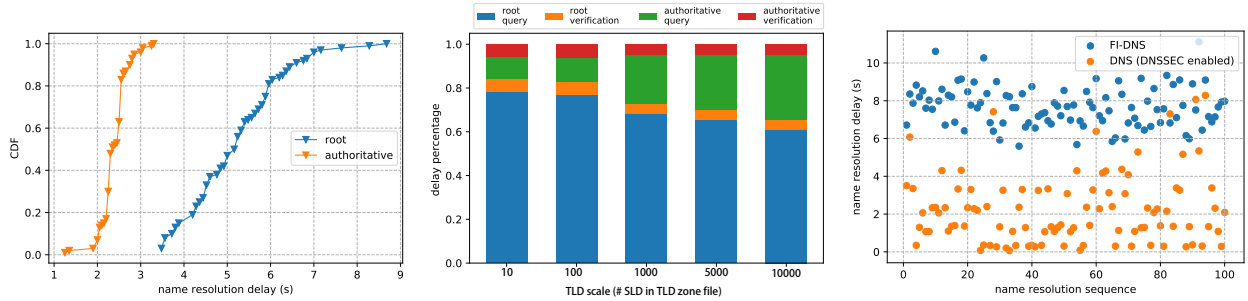
B. Data Collection

The FI-DNS name resolution process consists of 4 steps. The delay of name resolution can be divided into two parts: root delay and authoritative delay. Root delay is divided into a root query delay (Figure 7, step 1) and a verification delay (step 2). Authoritative delay can be divided into an authoritative query delay and a verification delay (step 3,4).

We use a collector installed on the client to obtain the time consuming of each step of name resolution. We performed name resolution tests for the same domain names on FI-DNS and DNS, summarized the collected data and plot in Figure 6.

C. Performance Analysis of Name Resolution Process

The name resolution performance is mainly evaluated by the name resolution delay. Figure 6(a) shows the cumulative distribution function (CDF) of the root and the authoritative delays in name resolution tests. Test data are obtained by randomly resolving names from 100 different TLDs. The results show that the root delay is distributed in the [3.51s, 8.74s] interval, the average is 5.25s. The authoritative delay is distributed in the [1.26s, 3.32s] interval, and the average is 2.43s. The measurement results show that the latency of the root resolution (layer-1 network) is approximately twice that of the authoritative (layer-2 network) resolution delay. This is because the blockchain network needs to call the docker virtual machine and execute a smart contract each time the data is resolved. Calling the virtual machine while executing the smart contracts will consume lots of time.



(a) Distribution of root delay and authoritative delay in FI-DNS

(b) Delay comparison of name resolution for different TLD sizes in FI-DNS

(c) Comparison of name resolution delay fluctuation between FI-DNS and DNS (DNSSEC enabled)

Fig. 6. The FI-DNS prototype system name resolution performance evaluation

TABLE II
DETAILED TEST ENVIRONMENT LIST, INCLUDING SERVER CONFIGURATION AND INITIAL STORAGE STATUS OF DOMAIN NAME DATA

Role	Location	Cloud provider	IP address	Initial status	Hardware specification
Root peer	Frankfurt, Germany	Linode	172.104.139.115	With Root Data	4vCPU, 8GB memory, 160GB hard disk
Authoritative peer	Dallas, United State	Vultr	149.28.245.24	With TLD data	1vCPU, 1GB memory, 25GB hard disk
	London, United Kingdom		95.179.197.70		
	Sydney, Australia		108.61.184.203		
	Singapore, Singapore		45.76.186.113	No TLD data	
Client	Shenzhen, China	N/A	210.22.22.141	No Cache	4vCPU, 8GB memory, 120GB hard disk

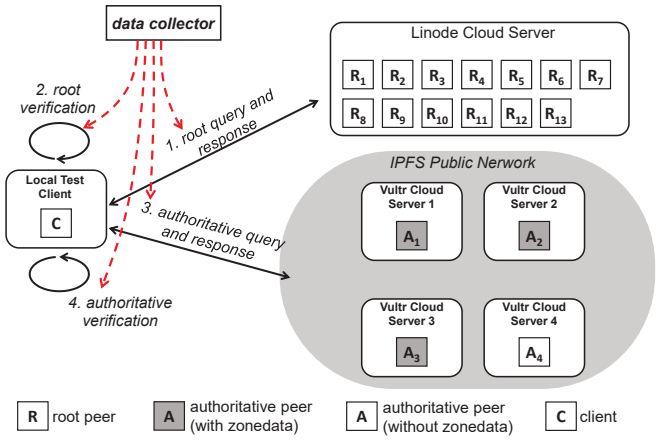


Fig. 7. Domain name resolution experimental environment

We then analyze the time proportion of 4 steps of the FI-DNS resolution process (describes in section V-B). Figure 6(b) shows the delay of each resolution step that constitutes the total name resolution delay. From the measurement results, the root query delay plus the authoritative query delay account for 90% of the total name resolution delay. The query delay is determined by the size of zone file and the network environment of the cloud service provider. The delay for local signature verification (root and authoritative verification defined in Figure 7 only occupies 10% of the name resolution delay.

Finally, we compared the performance differences between FI-DNS and DNS by resolving a set of identical domain names. The DNS resolution delay is obtained by querying Google caching resolver (IP:8.8.8.8) using the client. Figure 6(c) shows the name resolution delay comparison between FI-DNS and DNS (with DNSSEC-enabled), and the experimental data is obtained by randomly selecting 100 SLDs for name resolution tests. The measurement results show the following: FI-DNS and DNS delays are respectively distributed in the [5.60s, 11.12s] interval and the [0.06s, 8.28s] interval. The 90th percentile of DNS delay is 5.60s, which is the minimum resolution delay of FI-DNS, and the average resolution delay of the DNS is 28.8% of FI-DNS. The Google caching resolver reduces the name resolution delay by large-scale deployment and using cache pools. However, we removed the use of the cache server in our architecture, and this is the main factor that causes the average name resolution delay gap between the two.

D. Stability Analysis of Name Resolution Process

Name resolution stability refers to the delay fluctuation of domain name resolution during the test sequence. Figure 6(c) shows the comparison of the name resolution delay fluctuation between FI-DNS and DNS. From the distribution of points on the y-axis, the name resolution delay fluctuations of FI-DNS and DNS are close, and FI-DNS is slightly better. The variance of the FI-DNS name resolution delay is 1.12, which is 67.3% lower than the variance of the DNS. Because FI-DNS uses hash values to index files in subdomains, this indexing method usually has a more stable number of queries than DNS. Once the DNS resolver selects an authoritative server authorized outside the domain, multiple queries will be added, which will

make the name resolution delay of the same domain name unstable.

VI. Discussion

Here, we give some high-level comparative discussions on the design of FI-DNS name resolution architecture to other designs. Firstly, FI-DNS is entirely different from Namecoin [14] and Blockstack [15], both of which want to create a new unsupervised, uncentered domain name system. For this purpose, they need to isolate from the existing ICANN-managed domain namespace and introduce a new namespace, like .bit, but this will inevitably cause compatibility issues. The FI-DNS solution is devoted to systematically solving the availability and consistency problems in the existing resolution system. It is an enhancement of the existing resolution architecturethus FI-DNS completely inherits the current namespace. Secondly, FI-DNS is the first work that explores the possibility of a distributed operation model for root zone management. In current DNS, ICANN and TLD registries collaborate to generate a root zone file, but the data needs to be aggregated to a central data station, root zone data will be released after auditing, thus there is a single point of failure risk contains in the management process. To eliminate it, we need to use permissioned blockchain technology and smart contract to help us achieve this goal.

Next, we compare the compatibility of different designs with DNS. Compatibility includes four aspects: namespace compatibility, server-side compatibility, client-side compatibility, and communication protocol compatibility. Namecoin [14] and Blockstack [15] do not meet all the four compatibilities above. DecDNS [16] uses the blockchain only for storage purpose, so it meets all the above four compatibilities at the same time. Because our solution changes the subdomain indexing format, we need the resolvers that understand this new index method to perform name resolution, which causes the only client-incompatibility.

VII. Conclusion and Future work

In this paper, we propose a domain name resolution architecture (FI-DNS) based on blockchain technology to solve availability and consistency problems in DNS. We implemented the prototype system to prove its feasibility and effectiveness. Performance evaluation results show that the resolution delay of FI-DNS is more than twice that of the DNS due to the slow execution efficiency of the blockchain. In our future work, we will attempt to improve the server end resolution efficiency of the FI-DNS resolution architecture to increase its practicality.

Acknowledgment

This work was supported in part by the National Key R&D Program of China under Grant SQ2018YFB180078

References

[1] C. Lewis and M. Sergeant, "Overview of best email dns-based list (dnsbl) operational practices," Internet Requests for Comments, RFC 6471, January 2012.

[2] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante, "Drafting behind akamai (travelocity-based detouring)," in *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4. ACM, 2006, pp. 435–446.

[3] P. Hoffman and J. Schlyter, "The dns-based authentication of named entities (dane) transport layer security (tls) protocol: Tlsa," Internet Requests for Comments, RFC 6698, August 2012.

[4] N. Brownlee, K. C. Claffy, and E. Nemeth, "Dns measurements at a root server," in *GLOBECOM'01. IEEE Global Telecommunications Conference (Cat. No. 01CH37270)*, vol. 3. IEEE, 2001, pp. 1672–1676.

[5] C. Deccio, J. Sedayao, K. Kant, and P. Mohapatra, "Measuring availability in the domain name system," in *2010 Proceedings IEEE INFOCOM*. IEEE, 2010, pp. 1–5.

[6] R. Elz, R. Bush, S. Bradner, and M. Patton, "Selection and operation of secondary dns servers," Internet Requests for Comments, BCP 2182, July 1997.

[7] V. Pappas, Z. Xu, S. Lu, D. Massey, A. Terzis, and L. Zhang, "Impact of configuration errors on dns robustness," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4. ACM, 2004, pp. 319–330.

[8] K. Schomp, T. Callahan, M. Rabinovich, and M. Allman, "On measuring the client-side dns infrastructure," in *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 2013, pp. 77–90.

[9] T. Callahan, M. Allman, and M. Rabinovich, "On modern dns behavior and properties," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 3, pp. 7–15, 2013.

[10] D. Kaminsky, "Black ops 2008: It's the end of the cache as we know it," *Black Hat USA*, vol. 2, 2008.

[11] K. Schomp, M. Allman, and M. Rabinovich, "Dns resolvers considered harmful," in *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, 2014, pp. 1–7.

[12] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "Protocol modifications for the dns security extensions," Internet Requests for Comments, RFC 4035, March 2005.

[13] T. Reddy, D. Wing, and P. Patil, "Dns over datagram transport layer security (dtls)," Internet Requests for Comments, RFC 8094, February 2017.

[14] A. Loibl and J. Naab, "Namecoin," *namecoin. info*, 2014.

[15] M. Ali, J. Nelson, R. Shea, and M. J. Freedman, "Blockstack: A global naming and storage system secured by blockchains," in *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, 2016, pp. 181–194.

[16] J. Liu, B. Li, L. Chen, M. Hou, F. Xiang, and P. Wang, "A data storage method based on blockchain for decentralization dns," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2018, pp. 189–196.

[17] "Bylaws for internet corporation for assigned names and numbers," http://www.icann.org/resources/pages/governance/bylaws-en, accessed: 2019-10-25.

[18] "The hyperledger fabric project," https://hyperledger.org/projects/fabric, accessed: 2019-10-25.

[19] "The interplanetary file system (ipfs) project," https://ipfs.io, accessed: 2019-10-25.