BCDNS: A Secure DNS Authority Server Based on a Decentralized Blockchain

Joseph E Gersch
Invykta LLC
Loveland CO USA
jgersch@invykta.com

Stephen C Hayne
SCH & Associates
Fort Collins CO USA
stephen.hayne@gmail.com

***Abstract*— A blockchain implementation of a DNS Authority Server is described. Metrics for size, performance and cost are provided. Web3 design principles have been followed, yet adherence to conventional DNS semantics, governance, and workflow is explained. Utilities are defined for parsing standard zone files directly into blockchain records as well as to bridge the blockchain to conventional DNS systems and web browsers. The blockchain implementation provides operational advantages and simplifications compared to conventional DNS. These include *data integrity* by means of automated digital signatures; *DDoS resistance* based on a public, widely distributed peer-to-peer blockchain network; *data privacy* for DNS queries; and elimination of complex administrative tasks such as maintenance of DNSSEC signatures and management of primary and secondary name servers requiring AXFR, TSIG and serial number synchronization.**

Keywords — DNS, DNS Security, DNS Privacy, DNS Authority Server, Blockchain, Smart Contract, Decentralization

I. Introduction

A. Motivation

The Domain Name System (DNS) is the Internet Service that maps human-readable domain names into IP addresses and other associated data. The DNS is well-established, adheres to Internet governance, and has evolved over time to address new capabilities and overcome many vulnerabilities. In essence, the DNS is critical to the proper functioning of the Internet.

Nevertheless, many issues with the DNS still exist today [1]. These include susceptibility to data integrity attacks [2], privacy concerns [3], and denial-of-service attacks [4]. Many conventional DNS Authority Servers, for example, are deployed with only a primary server and one or two secondary servers; this may be insufficient to withstand a large DDoS attack. User queries are transmitted without encryption and can be logged by ISPs thereby enabling user surveillance or sale of user browsing habits.

DNSSEC security extensions were developed in 2005 to protect DNS data integrity. The actual deployment and use of DNSSEC has been disappointing, however. Statistics cited [5] indicate that only 3.9% of *.com* domains use DNSSEC as of March 2023. Despite advances in automation, DNS operators are often reluctant to deploy DNSSEC due to operational complexity and the risk of taking a zone offline should mistakes be made. The result is that most DNS messages remain vulnerable to counterfeiting.

This paper describes the design of a DNS Authority Server based on a decentralized public blockchain with many world-wide peers. Our blockchain DNS (BCDNS) architecture and implementation directly addresses the issues described earlier with respect to data privacy, integrity, and availability. It also simplifies administrator workflow when compared to the procedures required to manage conventional DNS redundancy or to maintain DNSSEC signatures. BCDNS is not meant to be a wholesale replacement for the existing DNS; it has instead been designed for compatibility and interoperability. Given its benefits, BCDNS is proposed as an improvement to current DNS authority server implementations that can fit well into the existing DNS ecosystem.

B. Design Goals

These objectives are discussed in greater detail throughout this document:

- **Compatibility & Interoperability** with standard DNS.
- **Scalability** to billions of DNS records.
- **Data Integrity** - automated digital signatures.
- **DDoS Resistance** - highly decentralized nodes.
- **Data Privacy** - TLS encryption for DNS queries.
- **Ease of Use** - simplifies DNS zone administration.
- **Dynamic** - supports updates to DNS records.
- **High Performance** - eliminates recursive lookups.
- **Low Cost** – wrt gas costs and storage requirements.
- **Transparency** – all transactions on the blockchain.
- **Safety** - key management and recovery.
- **Censorship Prevention** - via decentralization.
- **Governance** - registrant/registrar/registry model as well as strict rule based self-registration for top level domains.

C. Organization

The remainder of this paper is organized as follows. Section II describes related work with respect to blockchain naming systems. Section III gives the architecture and methods that were developed to implement BCDNS on an Ethereum public or private blockchain. Section IV describes the associated ecosystem, including DNS workflow and administrator utilities. Section V discusses data integrity and privacy. Section VI provides metrics with respect to performance, storage, and cost. Conclusions and a summary of future work required are given in Section VII.

II. Related Work

Previous efforts to define namespaces via decentralized blockchains include *Namecoin* [6][7], the *Ethereum Name System* [8][9], and *Handshake* [10][11][12].

Namecoin is a key-value registration system based on a clone of bitcoin technology. Users can purchase *Namecoin* tokens (NMC) to create names ending with a *.bit* suffix. User must run a full *Namecoin* node to resolve names securely or may use a lighter-weight SPV client with lower security. The *ncdns* daemon [3] creates a bridge between *Namecoin* and the DNS so that *.bit* websites can be resolved. *Namecoin* is notable as a first mover but has several issues. Anonymous self-registration of names has resulted in malware and spam emanating from *.bit* domains. This caused *OpenNIC* [13], an organization that supports alternative TLD naming, to drop support for *Namecoin*. In contrast, BCDNS addresses anonymity with its governance and accountability mechanisms. Additionally, *Namecoin* may suffer from limited redundancy depending on the number of private peer nodes available. BCDNS is highly distributed and decentralized by employing the already existing global infrastructure of Ethereum.

The *Ethereum Name System* (ENS) runs as a smart contract on the public Ethereum blockchain. Its primary use model is to map human-readable names to Ethereum account addresses. ENS is a first-come, first-served namespace in which names are assigned a *.eth* suffix. ENS has been extended to support DNS record types and supports DNS gateways. This permits an Ethereum client such as Mist to resolve the address of a traditional website, or the mail server for an email address, from a blockchain name. The *.bit* suffix prevents ENS from encroaching on DNS Top-Level Domains (TLDs). ENS does not specify how domains should be registered or updated, or how systems can find the owner responsible for a given domain. BCDNS, in contrast, provides accountability by storing registrar information for domains directly in the blockchain.

Handshake, based on a highly modified clone of bitcoin, is an alternative for creating top-level domain names. *Handshake* supplements the existing DNS by allowing users to bid, register, and trade TLDs without an intermediate registrar or other authority. Existing TLDs (e.g. *.com, .edu,* etc.*)* are pre-allocated in the *Handshake* blockchain so that the resolution of a traditional domain will be directed back to ICANN's root servers. *Handshake* has also populated its blockchain with the top 100,000 most popular domains which can be claimed by the original domain owner. New TLD names can be proposed by anyone and placed into an auction in which the highest bidder claims ownership of the new domain. Handshake has also created its own protocol, lightweight client, and plugin for the *Unbound* recursive DNS server. *Handshake* has limited support for DNS record types as it is only meant to support TLD registration. In contrast, BCDNS is designed for DNS interoperability at all levels of the domain name hierarchy.

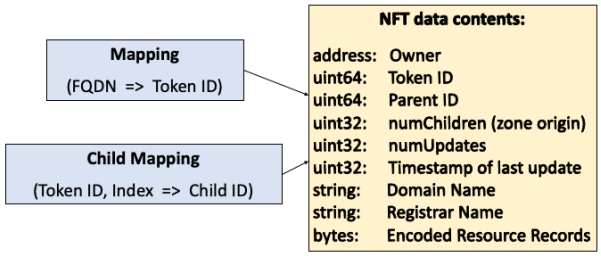

Figure 1: NFT data can be retrieved with a single lookup that maps an FQDN to its token ID. Child subdomains are retrieved using the Child Map and an index.

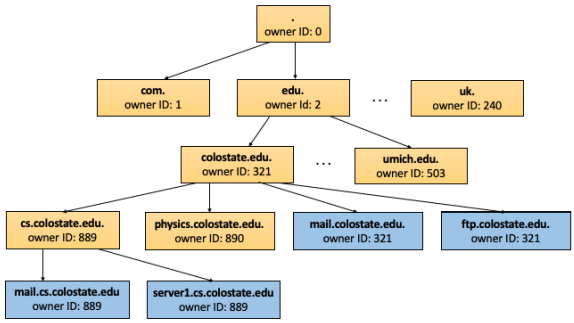

Figure 2: NFTs mirroring the DNS naming hierarchy by means of parent-child relationships. A blue background indicates a simple subdomain and orange indicates a delegation. Each NFT has an Ethereum account owner with public/private keypairs.

III. System Architecture

BCDNS is based on an Ethereum smart contract that manages all data storage and retrieval. Every DNS domain name is stored as a unique ERC-721 Non-Fungible Token (NFT) [14] [15].

Ethereum NFTs were selected as the appropriate data storage object because they are well-defined, highly scalable, industry tested, and have flexible usage models. We exploit NFT mechanisms to store domain data and to assign or transfer ownership of the NFT. Retrieval of an NFT is efficient, as shown in Fig 1. A single hash table lookup will map a Fully Qualified Domain Name (FQDN) to its appropriate NFT. Recursive lookups among multiple DNS nameservers are no longer necessary.

BCDNS mirrors the standard DNS naming hierarchy by organizing NFT tokens are into parent-child relationships. Each token has an owner and is designated as either a subdomain or as a delegation. Only the token owner can create child tokens. A child delegation token can be transferred to a new owner who in turn has permission to create its own child tokens.

Fig. 2 gives an example NFT hierarchy. Domain names follow standard DNS hierarchical naming for FQDNs. The tokens for *colostate.edu*. and *umich.edu*. are both children of the *edu.* token. Likewise, the tokens for *cs.colostate.edu*. and *physics.colostate.edu*. are children of the *colostate.edu*. token.

The blockchain itself provides data security and integrity. All transactions that store, modify or delete domain data on the blockchain require the use of Ethereum accounts based on public/private keypairs. All transactions are digitally signed and verified automatically. Account IDs provide proof of ownership for NFTs and bind with permission mechanisms in the smart-contract API. The API has functions to:

- Register, update, or delete a DNS domain and its associated data and resource records (e.g., *A, CNAME, TXT*, etc.). {zone owner only}
- Retrieve resource records. {anyone}
- Safely transfer a domain to a new owner using an *authorize/accept* protocol sequence. {zone owner only}

Careful consideration was given to minimizing the size of DNS Resource Records to be stored within an NFT. Each NFT contains the entire RRset for a domain. RRsets are usually small, containing only one or two *A* (*Address)* records. RRsets can become quite large, however, especially if there are multiple RRtypes or numerous *TXT* records associated with the domain.

An encoded data structure was designed to minimize RRset size). Prior to embedding in BCDNS, each RRset is evaluated by encoding it in several formats with and without compression and then choosing the smallest size format. If necessary, resource records can also be stored off-chain in IPNS/IPFS [16][17] or other digitally signed decentralized data storage. This is important for dynamic DNS data with frequent updates. Off-chain data prevents the blockchain from growing endlessly in this case.

Not all RRtype records need to be stored in BCDNS. There is no need to store DNSSEC records such as *RRSIG, NSEC, DS,* or *DNSKEY.* The NFT itself has already been digitally signed by the zone owner. Furthermore, there is no need to store *SOA* and *NS* records. There are no *SOA* serial numbers, zone transfers nor timeout information required. Conventional DNS uses *NS* records to perform delegation from server to server.. Delegation in BCDNS is defined as the relationship between parent zone to a child zone, not to another server that needs to be recursively queried. The entire DNS namespace is stored on a single blockchain using fully qualified domain names (FQDN).

IV. DNS Ecosystem, Governance, & Workflow

BCDNS cannot exist as just a standalone smart contract. For completeness, it must interact with conventional DNS APIs as well as with Web3 applications. The overall system should conform to well-known administrative workflows that hasten adoption and provide familiarity.

Fig. 3 shows an ecosystem allowing DNS and Web3 applications to interact with BCDNS. We will survey the workflow and utilities needed within this ecosystem to create a new DNS zone with origin *mydomain.com*. Assume that the root zone "." exists and that all TLDs have been registered and

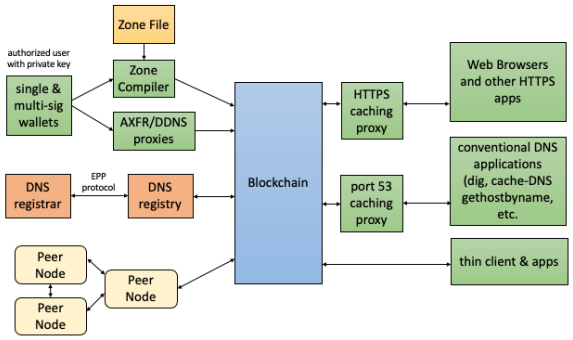

Figure 3: Example BCDNS ecosystem with direct blockchain access and bridges to conventional DNS applications.

transferred to accounts owned by the appropriate registries. The steps to be examined involve (A) user-onboarding, and (B) domain registration, zone compilation, and data access.

A. Blockchain Wallets, Accounts, & Onboarding

The first step in using BCDNS requires the domain owner to initialize a blockchain wallet with account IDs. This could be a daunting task for a DNS administrator who has no prior blockchain experience. Skilled blockchain users are familiar with wallets such as *Metamask* [18] or *Gnosis Safe* [19]. DNS administrators have likely heard of neither.

The onboarding process can be confusing and even dangerous if the DNS administrator were to lose account credentials. To avoid this, BCDNS is deployed using a multi-factor wallet system that makes onboarding as easy as signing on to an email or web banking account. The wallet follows ERC-4337 [20][21] standards and incorporates features for creating and managing accounts, purchasing Ethereum currency, key derivation using multi-factor simplified login [22], shared accounts with multi-signature, optional key escrow, and a method for bundling multiple transactions with a single approval (important for zone compilation).

B. Domain Registration and Zone File Compilation

As is typical with conventional DNS, the creator of *mydomain.com* contacts an accredited registrar to register the new domain. In turn, the registrar will use the EPP Extensible Provisioning Protocol to inform the registry owner for *.com* to process the registration and create the new NFT via the *registerDomain* API.

The registerDomain function also records the name of the registrar within the NFT. This is to complement the information normally stored in the WHOIS database. The registry then authorizes the transfer of the domain to the domain registrant.

NS records in the registry's conventional DNS server can be created for interoperability, but these records would point to BCDNS port 53 anycast proxies (see section V) as the authority for *mydomain.com.*

At this stage only the domain origin for *mydomain.com* is present in the blockchain. The domain registrant must finish the safe-transfer handshake and accept ownership of the domain.

The next task is to create and register all subdomains and resource records for *mydomain.com* zone. This is managed in bulk with a zone compiler utility. DNS zones are typically managed by means of textual zone files. DNS administrators are familiar with this process and would be reluctant to switch to other methods for maintaining a zone. To accommodate this familiar behavior, the BCDNS zone compiler is invoked as either a stand-alone program or through a web interface. The *zonec* utility parses the zone file and creates NFTs for all subdomains and stores them into the blockchain.

Updates to a zone are made by simply editing the zone file and re-compiling or by using a Dynamic-DNS (DDNS) proxy to update a record. Note that the DNS administrator no longer need to manage and maintain DNSSEC signatures in the zone. All signature management and verification are handled automatically by the blockchain. Also note that BCDNS decentralization means that the DNS administrator no longer needs to deal with secondary authority servers and can eliminate the maintenance and use of AXFR/IXFR and TSIG.

After zone compilation, the *mydomain.com* zone data is available on a world-wide basis via blockchain decentralization. There are various methods for accessing the zone data, each with its own security model and privacy considerations. These methods are discussed in section V.

Users looking for an open and democratic alternative to the DNS root may wish to self-register TLD domains without going through the normal registrant/registrar/registry governance process. OpenNIC supports this objective as a user owned and controlled top-level Network Information Center offering a non-national alternative to traditional Top-Level Domain registries such as ICANN.

BCDNS has incorporated, but not yet enabled, a secure blockchain oracle to accommodate self-registration of TLDs. The oracle is intended to be open and fair, but also must adhere to a set of strict rules to avoid trademark infringement and to provide transparency and accountability. The future design work will explore possibilities such as cooperation with OpenNIC and with Handshake regarding domain auctions, or to validate requests based on ownership of publicly verifiable credentials.

V. DNS Query Security and Privacy

There are multiple methods to access data in BCDNS, some being highly secure and private, and some being completely insecure. The most direct method is for data queries to make blockchain GET requests to a local copy of a blockchain node. Queries could also be made indirectly through an HTTPS provider such as *Infura* [23] or *Alchemy* [24].

Bridging from BCDNS to conventional port 53 DNS protocol is required so that existing caching servers and applications will work without being re-written. Alternatively, a blockchain node could be embedded within a DNS caching server such as Unbound [25] via plug-ins.

Each of these techniques vary in their ability to provide privacy and data integrity guarantees. Before exploring these query methods further, we must first set some context and explore issues related to DNS privacy and security in general.

A. Conventional DNS Privacy Issues - State of the Art

A fundamental limitation in enforcing query privacy in conventional DNS is the centralization of DNS services offered by corporations and Internet open resolvers. The configuration and use of the DNS servers is under site control. Sending a query to a corporate hosted resolver or open resolver leaves the end user vulnerable to the collection of the DNS query and the IP address from which it was sent.

Tracking client DNS queries is a rich source of surveillance capital [26]. It gives advertisers, governments, and hackers information about the web sites visited by an end user. This data can be used by third parties to analyze end users web browsing behaviors resulting in targeted advertisements, censorship, and end user profiling. In short, it is a stark invasion of privacy.

To prevent surveillance of DNS queries in flight, several technologies have been developed to encrypt DNS queries from a client to the DNS resolver: DNS over HTTPS (*DoH*), DNS over TLS (*DoT*), and Oblivious DoH (*ODoH*). These technologies are designed to protect client privacy in the DNS lookup path between the end system and a DNS resolver.

DoH and DoT prevent intercept of the DNS query over a network but do not prevent the DNS resolver from logging the IP source and DNS query. A resolver -- as part of its normal operation -- necessarily decrypts the DNS query to perform a look up at which point the data is in the clear and can be logged and archived.

To prevent a resolver from tracking the IP client address of the DNS query, Oblivious DoH was proposed (ODoH) [27]. The ODoH solution inserts a proxy between the client and server with the intention of preventing the resolver from knowing the IP address of the client query, thus anonymizing the source. While this enhances end to end client privacy, the solution is implemented by centralized servers any of which can log the information despite assurances from ODoH service providers that DNS query data is not collected.

Most browsers embed resolvers -- Mozilla for example has a configurable trusted recursive resolver or TRR -- which provides options for securely requesting DNS lookups [28][29]. Features such as DoH or ODoH (currently experimental) are configurable in the network setting preferences. The TRR Configuration Mode default is to enable trusted resolution using DoH with Cloudflare as the provider. Providers are vetted in accordance with the Mozilla Security/DOH-resolver-policy [30]. The Mozilla browser defaults to the use of DoH and DNS queries are encrypted over the network via the DoH HTTPS connection. Otherwise, the resolver falls back to the native resolver where all queries are in the clear on the wire.

B. BCDNS Privacy Techniques

To address vulnerabilities related to DNS queries, BCDNS decentralizes access to the domain name system and provides several data access methods. One access method is to use a commercial blockchain access provider. The provider submits the query to one of the nodes in their array of blockchain servers. The query is resolved, and the response is sent to the provider end point and relayed to the client using TLS encryption. The *bcdig* query utility is an example of a program using this access method.

While the use of a commercial Ethereum provider is a convenient way to access the blockchain, it has two limitations: 1) provider query rate limiting and 2) queries still pass through a centralized corporate owned gateway which may or may not perform logging.

A solution guaranteeing complete privacy would be a system in which each client system installs a local blockchain node (full or lightweight) to which a DNS query could be sent. This mitigates the rate limiting concern and eliminates the corporate middleman. The size of even a lightweight peer node is considerable, however, especially for devices with limited storage and memory. Nevertheless, these direct-access methods provide the strongest privacy and assurance of data integrity.

Future work involves integration of BCDNS access directly into the Mozilla TRR Trusted Resolver. Direct access of the TRR to the blockchain would provide a high level of security and privacy to web browsing which is the most common use case and largest consumer of DNS services.

Existing applications such as the *dig* command, *gethostbyname,* or even full-featured DNS caching servers require a bridging mechanism between the BCDNS system and standard DNS protocols. To accomplish this, a port53 proxy has been developed which translates wire-format DNS queries into blockchain data requests, returning data as conventional wire-format DNS responses. A port53 proxies can be run either locally or as a public service on a distributed anycast network.

When a port53 proxy is run locally the consumer is in complete control of privacy and security. Public proxies, on the other hand, communicate across the Internet in plaintext and all privacy is lost.

Therefore, the port53 proxy program is available as open-source for local implementation and deployment.

C. Signature Verification

Unlike DNSSEC, the end consumer of a BCDNS query does not receive an RRSIG record attached to the response for a requested resource record. We instead rely on the blockchain digital signatures and the data access methods to guarantee end-to-end data integrity. This is the fastest and most convenient data access method.

There may be some situations, however, in which the consumer would prefer an additional guarantee of data correctness. In this case, the normal blockchain method to retrieve the digital signature for an NFT transaction can be directly obtained from the blockchain to verify the public/private keys match with the data requested.

D. Private Blockchain Access

Some organizations may want to benefit from the enhancements offered by BCDNS but have private or split-brain DNS data that should not be accessible to the outside world. Examples include government, military, or internal corporate systems. In these situations, a private Ethereum blockchain containing the BCDNS smart contract can be deployed by the organization and access restricted to their own use. All API calls and applications would work in the same manner as the public blockchain BCDNS.

Table I: A comparison of gas consumption, ether fees and dollar prices for transactions on two blockchain networks: the Goerli Testnet and the Level 2 Arbitrum Testnet. Arbitrum is typically 100x to 200x lower in cost due to lower gas fees.

	Ethereum gas consumed	**Ethereum fee** (20 gWei/gas)	**Arbitrum gas consumed**	**Arbitrum fee** (0.1 gWei/gas)
Deploy contract	6,882,000	0.01376 $25	10,612,000	0.00106 $1.91
Register Domain	212,000	0.00042 $0.76	473,000	0.000047 $0.09
Transfer Domain	47,000	0.00009 $0.18	141,000	0.00001 $0.01
zonec 35 domains	8,211,000	0.01642 $30	19,686,000	0.00197 $3.54
zonec 179 domains	41,281,000	0.08256 $148	50,275,000	0.00503 $9.05

VI. Metrics

A. Blockchain Storage & Gas Costs

BCDNS must be affordable. Its operational costs should be comparable or lower than the cost of hosting conventional DNS servers. This section examines the cost of storing data on both level-1 and level-2 public blockchains. The level-1 blockchain measurements were made on the *Ethereum Goerli Testnet.* The second set of measurements were performed on the level-2 *Arbitrum Goerli Testnet* [31]. Both platforms use Ether as their native currency. As expected, the level-2 *Arbitrum* platform is much less expensive and faster than the level-1 platform.

The only consistent metric regarding blockchain transactions is the amount of gas that a transaction consumes. All other measurements vary by market value. Gas fees have historically ranged from 10 gWei to 500 gWei. Typically, gas prices hover close to 20 gWei. The market value for Ethereum also varies. At the time of this writing, 1 ether is valued at $1800. Ether prices have ranged from $100 to $4600 over its lifetime.

Table I gives measurements for various transactions on the two platforms. The cost for registering a domain on the *Goerli* platform was typically 230,000 gas units, or about 83 cents per domain given the fees and ether price quoted earlier. As can be seen, registering a zone with 179 domains becomes somewhat costly at $148.

The *Arbitrum* measurements showed 100x to 200x lower cost due to the cost of gas (0.05 to 0.1 gWei per gas unit). The price for storing the 179-domain zone dropped from $148 to a much more affordable price of $9.

In practice, the Arbitrum costs might be too low. Blockchain price structures are not just there to pay for infrastructure costs, but also to discourage users from data flooding on the blockchain. Accordingly, the BCDNS smart contract has incorporated a variable fee structure to balance costs to appropriate levels.

Of course, deployment on a private blockchain could institute any pricing structure the system demanded, from 0 gWei per gas unit on up.

B. *Impact on Blockchain Size*

In one sense, BCDNS is decentralized, because there are thousands of redundant nodes all over the world to share the load. But on the other hand, BCDNS data is "centralized" in the sense that potentially all of the DNS namespace is present on each blockchain node. There is no "fan-out" of delegations to multiple authoritative servers, each with its own small portion of the DNS namespace.

How will this impact the overall size of a blockchain node?

As of December 2022, there are 351.5 MM domain names [32]. As a rough approximation, if each domain had an average of 20 subdomains, there would be a total of 7 billion FQDN names. Assuming each FQDN requires 150 bytes of blockchain storage, 1.05 terabytes would be required to store the entire DNS namespace and data.

As of March 2023, The Ethereum blockchain has 16.9 million blocks and requires 350 gigabytes of storage to support a node. Adding BCDNS would quadruple this size to 1.4 terabytes. This requirement can easily be accommodated with today's standard disk and SSD drives. Nevertheless, Ethereum researchers are investigating ways to handle very large scale blockchains including pruning, sharding and partitioning. Additionally, BCDNS could be implemented on its own private blockchain, but only if sufficient redundancy and global access were guaranteed.

C. *Performance*

Blockchains were not originally designed for extremely high transaction performance. Ethereum nodes are busy doing many things simultaneously: they keep track of pending transactions from peers and validate new blocks, they store and read data from the local disk, and they handle RPC API requests from applications. Given this workload, Ethereum was originally designed to handle 20-40 writeable transactions/second.

As a level-2 blockchain implementation, Arbitrum can handle 40k TPS. The new Ethereum design goal is 100k TPS.

These rates are sufficient for filling the blockchain with DNS data in a timely manner. 7 billion domain names could be stored within a few days. This is an unrealistic scenario, of course. Actual adoption rates would be smaller and be spread out over many months and years.

Of more interest is the ability for blockchains to handle the DNS query load – the READ rate. The worldwide DNS query rate is unknown, but likely to be close to a billion QPS. Most of these queries are answered by DNS caching servers which typically have a cache miss rate between 5 and 10%. This would result in a total of 50MM to 100MM QPS to BCDNS authoritative servers distributed across the globe. Supporting this estimate:

- In 2018, Google's DNS caching servers were getting 1.2 trillion queries per day. In 2022 they were handling about 20 million QPS. Assuming a 5% to 10% cache miss ratio results in approximately 1 to 2 million QPS sent to the respective authoritative DNS servers.
- Add to this the DNS load from world-wide ISPs. Large telephony carriers typically provision their caching servers for a load ranging from 10k to 1MM QPS. A 10% cache miss ratio results in 1k to 100k authoritative server QPS.

Can a blockchain network support this load? The Infura blog, *What Does it Take to Handle Two Billion Eth_calls Per Day?* [32] provides some interesting insights. Infura measured read performance for an individual node at 5249 eth_calls per second (450MM/day). Using multiple nodes, load balancing, data caching and other techniques, Infura has scaled to gets 30k reads/second and continues to expand its performance.

A global DNS query load of 50MM reads/second is 1500 times the current Infura load. A simplistic model would assume that at least 1500 Infura-sized nodes would be required. This is certainly possible, and is in fact much, much smaller than the total number of DNS servers currently deployed around the world.

Nevertheless, more research on scaling needs to be performed via live experiments and by simulations before we can have confidence in a global deployment solution.

VII. Conclusions and Future Work

A working version of the BCDNS blockchain has been created, refined, and deployed to both the Ethereum test net, the Arbitrum test net and to a private blockchain created on AWS. Utilities including the zone compiler, bcdig, port53 proxy and other tools have been written and tested. The design objectives regarding compatibility, interoperability, data integrity, privacy and accountability have been met. Further research needs to be performed to ensure massive scalability. Other future work desired includes development of the following components, programs, and processes:

- Unbound Resolver plugin
- Multi-factor ERC 4733 Wallet
- Mozilla TRR Trusted Resolver
- Oracle TLD validation & policy
- An Investigation of the merits of deploying BCDNS on a private Ethereum blockchain and incentives for global deployment (e.g., bundling with caching servers) and comparisons to public blockchain regarding cost, impact, scalability, privacy, security.

References

[1] The Five Big DNS Attacks and How to Mitigate Them. (https://www.networkworld.com/article/3409719/worst-dns-attacks-and-how-to-mitigate-them.html)

[2] What is DNS Cache Poisoning | DNS Spoofing? (https://www.cloudflare.com/learning/dns/dns-cache-poisoning/

[3] Introduction to DNS Privacy. (https://www.internetsociety.org/resources/deploy360/dns-privacy/intro)

[4] DDoS DNS Attacks are Old-School, Unsophisticated… and They're Back. (https://www.theregister.com/2023/03/29/ddos_dns_attacks_are_oldschool/?td=rt-9bq)

[5] Dnssecstat (https://rick.eng.br/dnssecstat/)

[6] Namecoin (https://namecoin.org)

[7] Namecoin DNS bridge (https://github.com/namecoin/ncdns)

[8] ENS Decentralized Naming (https://ens.domains)

[9] Ethereum Name Service EIP-137 (https://github.com/ethereum/EIPs/blob/master/EIPS/eip-137.md)

[10] Handshake Decentralized Naming and Certificate Authority (https://handshake.org)

[11] Handshake Architectural Design Paper (https://handshake.org/files/handshake.txt)

[12] The Ambitious Plan to Reinvent How Websites Get Their Names, MIT Technology Review. (https://www.technologyreview.com/2019/06/04/239039/the-ambitious-plan-to-make-the-internets-phone-book-more-trustworthy/)

[13] The OpenNIC project (https://www.opennic.org)

[14] ERC-721 Non-Fungible Token Standard. (https://eips.ethereum.org/EIPS/eip-721)

[15] OpenZeppelin ERC-721 Contract. (https://docs.openzeppelin.com/contracts/3.x/erc721)

[16] Inter-Planetary File System (IPFS). (https://ipfs.tech/)

[17] Inter-Planetary Name System (IPNS). (https://docs.ipfs.tech/concepts/ipns/#mutability-in-ipfs)

[18] Metamask, A Crypto Wallet and Gateway to Blockchain Apps (https://metamask.io)

[19] Gnosis Safe. (https://safe.global)

[20] ERC-4337: Account Abstraction Using AltMempool. (https://eips.ethereum.org/EIPS/eip-4337)

[21] Let's Understand EIP-4337. (https://medium.com/neptune-mutual/lets-understand-eip-4337-559ba14761cc)

[22] Multi-Factor Key Derivation Function for Fast, Flexible, Secure and Practical Key Management; Vivek Nair and Dawn Song, University of California, Berkeley. (https://arxiv.org/pdf/2208.05586.pdf)

[23] Infura. (https://www.infura.io/)

[24] Alchemy, the Web3 Development Platform. (https://www.alchemy.com/)

[25] NLNetLabs Unbound DNS Caching Resolver (https://nlnetlabs.nl/projects/unbound/about/)

[26] The Age of Surveillance Capitalism, Shoshana Zukoff; 2019 Hachette Book Group; page 8.

[27] Improving DNS Privacy with Oblivious DoH. (https://blog.cloudflare.com/oblivious-dns/)

[28] Mozilla Wiki: Trusted Recursive Resolver. (https://wiki.mozilla.org/Trusted_Recursive_Resolver)

[29] Mozilla Wiki: Security/DNS over HTTPS. (https://wiki.mozilla.org/Security/DNS_Over_HTTPS)

[30] Mozilla Wiki: Security/DoH-Resolver-Policy. (https://wiki.mozilla.org/Security/DOH-resolver-policy)

[31] A Gentle Introduction to Arbitrum. (https://developer.arbitrum.io/intro/)

[32] Ultimate Domain Name Stats. (https://siteefy.com/how-many-domains-are-there/)

[33] Infura blog: What Does it Take to Handle Two Billion Eth_calls Per Day? (https://blog.infura.io/post/what-does-it-take-to-handle-two-billion-eth_calls-per-day)