IEEE ICBC'24 1570981740

NC-DHT: Towards a Robust and Anonymous DHT

***Abstract*—Distributed Hash Tables (DHT) is one of the most popular peer-to-peer (P2P) systems. It provides the underlying overlay for supporting communication in a wide spectrum of practical systems, such as file sharing, content distribution, and distributed search engines. However, the decentralized nature of DHT introduces security and privacy vulnerabilities. Significant efforts have been devoted to making DHT's more robust and augmenting DHT's privacy guarantees. This paper proposes a novel design, namely NC-DHT, which has two main features: (i) NC-DHT tolerates Byzantine nodes, and (ii) provides *initiator anonymity*, *query unlinkability* and *target privacy*. To the best of our knowledge, NC-DHT is the first DHT system to support all these important features, which makes it suitable as an overlay for Blockchain systems. NC-DHT relies on a novel integration of quorum topology and network coding-based techniques.**

***Index Terms*—Byzantine, DHT, Privacy, Network Coding.**

I. Introduction

DHT is one of the most popular mechanisms that enable large-scale decentralized services, e.g., Azureu DHT, Kademlia DHT and Mainline DHT (used by BitTorrent [1]) [2]–[4]. Generally speaking, DHT implements the decentralized key-value data structure (or the hash table) – a scalable mechanism to map keys onto values. Two main APIs are:

- *Insert*: write a value v to key k, and
- *Retrieval*: given key k, read its value v.

The core of DHT is a scalable *routing* or *lookup* scheme that can handle a high churn rate: *given a key, how to find the node (or peer)*[1] *that is responsible for storing and serving the data in a system with dynamic membership?*

Several structured DHT's have been proposed and deployed in practice, e.g., Chord [5], Kademlia [6], and CAN [7]. These DHTs adopt different ways of mapping data and hence, different routing mechanisms, and they are designed to provide efficient storage and coordination among decentralized nodes.

A. DHT in Recent Systems

DHT has been used in many emerging systems as well. The utilization of Distributed Hash Tables (DHTs) in blockchain technology is exemplified by recent innovations such as the Bitcoin Lightning Network (LN) [8], LightChain [9], IPFS (InterPlanetary File System) [10] and Ethereum Swarm [11]. DHTs enable decentralized storage and routing in Swarm and IPFS, whereas LN and LightChain improve performance and accessibility by storing off-chain information on DHTs.

In short, DHTs play a pivotal role in these recent systems. Ongoing developments and optimizations have bolstered the DHT's performance, security, and resilience. This has solidified its place as a fundamental component, enabling several important operations such as efficient content lookup, data redundancy, censorship-resistance, and data preservation. Therefore, DHTs have been used as an overlay for many Blockchain systems.

[1]Following the literature, we use "node" and "peer" interchangeably.

B. Making DHT Robust and Anonymous

The decentralized design makes DHT come with several vulnerabilities in security and privacy. In a typical DHT design, nodes rely on (a small number of) peers to complete the operations – usually $O(\log n)$, where n represents the total number of participating nodes in the system. This design gives malicious (or Byzantine) nodes an opportunity to provide false information to misdirect an honest node's routing [12].

With such malicious behavior, privacy can easily be compromised too. For example, malicious nodes can simply route an honest node to a bogus site and obtain its private information. Or malicious nodes can provide fake information to increase the workload of honest nodes so that the adversary can conduct traffic analysis to reveal private information.

This paper studies how to build robust and anonymous DHT. To address these vulnerabilities in security and privacy, our goal is to identify a routing design in DHT that achieves the following aspects:

- *Robustness*: communication between (groups of) nodes is reliable even if some nodes may become Byzantine faulty and have an arbitrary behavior.
- *Anonymity*: no initiator information is revealed (the initiator is the node that searches or updates the data).

In this work, we focus on how to provide *target anonymity*, i.e., no routing destination is learned by the adversary. In Section VI, we briefly discuss how to augment our system to provide initiator anonymity and query unlinkability.

C. State-of-The-Art DHT's

Significant efforts have been devoted to make DHT robust and/or anonymous, e.g., [13]–[23]. (Section III discusses other related work.) There are mainly two categories of approaches:

- *Distributed* (e.g., [20]–[22]): These systems use variants of *quorum topology* to tolerate Byzantine nodes. A quorum is a set of $O(\log n)$ peers.[2]
- *Centralized* (e.g., [15], [23]–[25]): These systems use variants of centralized mechanisms for manage nodes' identity and surveillance of nodes' behavior.

[2]The notion of "quorum" in the DHT literature is referring to a group of peers, and should not be confused with the one used in distributed computing and database literature, where two quorums must overlap to ensure consistency or correctness. Here, such overlap feature is not necessary [18]–[22]. In fact, each node can be in only one quorum in most designs.

On one hand, two major drawbacks of the distributed DHTs (e.g., [22]) are (i) the lack of support of the existing privacy-preserving techniques to provide initiator anonymity and query unlinkability, and (ii) high computation and high communication overhead for its routing mechanism.

On the other hand, centralized DHTs (e.g., [23]) relies on the stable and fast communication among nodes to reduce the number of false alarms [26]; moreover, they tend to require background mechanisms for surveillance which induce extra communication overhead.

D. Our Contributions

To overcome the aforementioned shortcomings, we neatly integrate the techniques from both categories and propose a novel Chord-based system, *NC-DHT*, which is both robust and provides anonymous guarantees. More concretely, NC-DHT tolerates a fixed fraction of Byzantine peers (in each quorum), and provides *target anonymity* with low communication and computation overhead.

More importantly, NC-DHT supports existing privacy-preserving techniques, such as multiple anonymous paths [23], so that our system can be easily augmented to provide *initiator anonymity* and *query unlinkability*. Finally, due to the redundant nature of the quorum topology, NC-DHT does not have the issue of false alarms and do not require that communication among peers are stable and fast.

On a high-level, NC-DHT combines network coding-based techniques with the concept of quorum topology [20], [21] and anonymous query [22] to perform routing with desirable properties. For bootstrapping and maintaining the quorum topology, NC-DHT relies on the use of Certificate Authority (CA) [15], [23]. Due to our novel design, CA's workload is light; thus, our system can support a reasonable number of peer churning simultaneously.

II. Preliminary

We first present key ideas from the literature [20]–[22].

A. Quorum Topology

A quorum is a set of nodes and behaves as an "atomic" unit for executing DHT operations, such as join/leave and routing. On a high-level, a quorum of nodes behaves as a single node in a non-fault-tolerant DHT. A DHT structure based on the concept of quorums is called "quorum topology."

The quorum topology provides two key features: (i) Byzantine behavior is mitigated by majority voting, and (ii) data corruption is recoverable due to data redundancy.

Quorum topology is usually embedded in DHT that has Chord-like structure [18], [20]–[22]. The topology is assumed to maintain the following *invariants* at all time.

- *Membership:* Every peer is in at least one quorum.
- *Goodness:* $< 1/3$ of the peers in a single quorum are Byzantine faulty.
- *Intra-Quorum Communication:* Every peer is able to communicate "directly" to all the other peers in the same quorum. That is, each peer knows all the identifies of the other peers belonging to the same quorum.
- *Inter-Quorum Communication:* If quorums Q_i and Q_j are neighbors in the specific DHT quorum topology, then each peer in Q_i can communicate "directly" with any peers in Q_j and vice versa.

B. Efficient Routing

Quorum topology incurs expensive routing operation – given a key, identify a route to the quorum that stores the data. Naive lookup operation incurs $O(\log^3 n)$ message complexity as elaborated in [20]–[22].

Young et al. proposed a novel system – RCP-I – which achieves the robust routing with $O(\log^2 n)$ messages (in expectation) based on recursive retrieving routing information in DHTs using threshold cryptography [20], [21]. One important feature of RCP-I is that it tolerates malicious peers $< 1/3$ of a quorum in the asynchronous system and $< 1/2$ of a quorum in the synchronous system.

Subsequently, Backes et al. integrated oblivious transfer (OT) with RCP-I to hide the identity of routing destination, i.e., to *provide target anonymity* [22], [27]. In the discussion below, we refer to Backes' schemes as OT-RCP-I. In OT-RCP-I, OT-RCP-I had several drawbacks as identified in [26]:

- OT-RCP-I does not support the existing privacy-preserving techniques, such as multiple anonymous paths [23], and thus it is not clear how modify it to provide initiator anonymity and query unlinkability.
- OT-RCP-I boosts the total number of messages by a constant factor due to the overhead of OT protocol.
- OT-RCP-I's oblivious transfer protocol has a substantial computation overhead – $O(\log n)$ in each hop.

Our system NC-DHT adopts the network coding techniques to have a more efficient routing than OT-RCP-I. Compared to RCP-I, NC-DHT provides anonymous guarantees and is equally efficient.

III. Related Work

Since the seminal works on using lightweight routing to build scalable DHT, e.g., Chord [5], Kademlia [6], and CAN [7], lots of different attacks have been identified and used, e.g., polluting data [28], polluting routing indices by creating fake data ID [29], and injecting fake data blocks [30]. Therefore, researchers have been studying how to make DHT robust.

Castro et al. [25] were among the first to use *redundancy* to make DHT robust – contacting multiple nodes to ensure that the routing information is not corrupted. Subsequent systems like Salsa [13], Halo [14], and Cyclone [31] propose and improve redundancy-based routing to provide better robustness. Myrmic [15] relies on a central authority that takes the responsibility of maintaining and updating node certificates. However, these systems do not provide anonymity guarantees.

NISAN [32] is a DHT with robustness and anonymity guarantees. In NISAN, each queried node to provides an entire routing table to the initiator. Consequently, the initiator can use the technique of *bound checking* on the table to

limit manipulation in routing information. This combined with redundant copies of data ensure both target anonymity and robustness. However, Wang et al. [33] identified an attack called *range estimation* to compromise the target anonymity. Torsk [24] combines DHT and Myrmic [15] to provide anonymous communication. The key mechanism for the routing initiator is to perform a random walk on DHT to find a node that can perform the routing on its behalf. However, as suggested in [23], Torsk's performance is limited due to its usage of Myrmic.

ShadowWalker [17] integrates the idea of redundancy (or shadow) into the DHT's structure. The shadows are nearby nodes in the constructed DHT structure, which help verify each hop of the routing. However, as analyzed in [34], ShadowWalker is not robust. This is because the entire set of shadows for a certain node may be compromised by a specially designed attack. Octopus [23] proposed a novel surveillance mechanism to discover attacker identities, and to remove malicious nodes. The mechanism significantly limits the adversary's ability . For anonymity, Octopus used redundant and dummy routing queries for ensuring various notions of anonymity, including target anonymity, initiator anonymity, and query unlinkability [23]. Unfortunately, it was identified in [26] that Octopus may induce high false alarm when nodes are not stable and the communication among peers is not reliable. After a thorough evaluation in practical settings [26], Octopus is found practical when communication among nodes are stable and fast. One reason is that when communication is unstable, Octopus may induce high false alarm due to its surveillance mechanism. This causes Octopus to remove honest nodes and downgrade performance.

NC-DHT resolves these limitations by using the integration of network coding techniques and quorum topology.

IV. Design Goals and System Model

A. Design Goals

The major goal is to make the routing (or lookup) in DHT robust and private. Concretely, for robustness, we achieve the following property, which are identical to the one in [20], [21].

- *Routing Correctness:* In the presence of an adversary that compromises up to $1/3$-fraction of any quorum, an initiator that looks up some key k should be able to locate the correct target that holds the value of key k.

Note that NC-DHT has the same message complexity as the ones in [20] [21].

For privacy, NC-DHT achieve the same privacy level as in [22], i.e., providing *target anonymity*, with low computation and computation overhead.

- *Target Anonymity:* given an initiator and its lookup request, the information about the target (of the query) should not be revealed.

In Section VI, we briefly discuss how our solution can be augmented for other privacy guarantees [23]: *initiator anonymity, query unlinkability, Fingertable correctness*, and *Fingertable trustiness*.

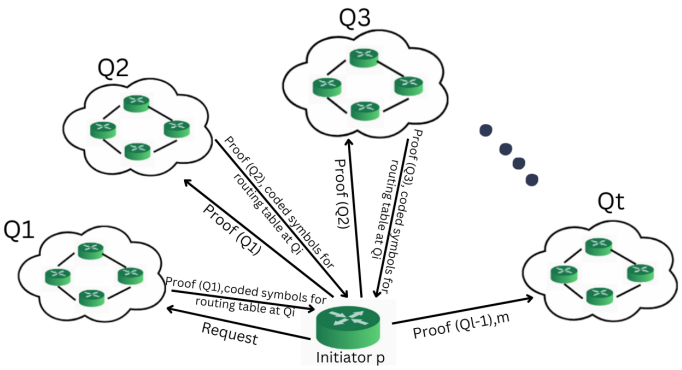

Fig. 1. **Illustration of NC-DHT.** Initiator p contacts quorums to obtain proof and routing table so that it can eventually reach the target quorum Q_t.

B. System and Fault Model

We consider an asynchronous system with the presence of Byzantine nodes. The communication is asynchronous in the sense that any message may suffer an infinite delay (or equivalently such a message is considered lost). The processing speed of a node is also arbitrary.

Additionally, we assume a strong (omniscient) adversary that has complete knowledge of the algorithm specification and the network topology. At most $1/3$ of nodes in the system may be compromised by the adversary. The compromised nodes are said to be faulty, and can have arbitrary behavior, such as dropping, sending incorrect and inconsistent messages, or colluding with each other. The adversary is assumed to be computationally bounded so that it cannot break the threshold signature used in RCP-I [20], [21].

Similar to previous works on anonymous communication [17], [22]–[24], [35], we assume that adversary *cannot* observe all the communication in the system; however, a faulty peers is able to observe the message sent to all the peers in the quorum that it belongs to, and share such information with other faulty peers with low transmission delay.

V. NC-DHT: Adding Target Privacy

In this section, we discuss how NC-DHT ensures lookup correctness and target privacy by using quorum topology and error detection code.

A. Error Detection Code in NC-DHT

The lookup operation makes use of an error detection code. We first describe the code and its properties. With a suitable choice of parameter c, we will use a $(s_Q, s_Q - t_Q)$ Reed-Solomon code over Galois Field GF(2^c), where s_Q is the size of quorum Q and t_Q is the upper bound on the number of faults in each quorum. By assumption, $t_Q/s_Q < 1/3$.

More precisely, the constant c is chosen large enough such that $s_Q \leq 2^c - 1$. Denote by D the number of bits of a routing table. Alternatively, D can be viewed as consisting of $s_Q - t_Q$ "*data* symbols" from the field $GF(2^c)$. Thus, each of

Algorithm 1 NC-DHT: Steps at Initiator $p \in Q_1$

Initial Step:

1: p sends the message (or request) below to all peers in Q_1:

$$[ID_p|add_p|REQUEST|ts_1]$$

2: p receives and interpolates all signature shares, which results into:

$$S_1 \leftarrow [ID_p|add_p|REQUEST|ts_1]_{k_{Q_1}}$$

Intermediate Step:

3: **for** i from 2 to $l-1$ **do**

4: p sends S_{i-1} and ts_i to all peers in Q_i and requests a proof, which consists of (i) signature S_i, (ii) public key $K_{Q_{i+1}}$, and (iii) **coded symbols** for the routing table

5: p receives and interpolates all signature shares, which results into:

$$S_i \leftarrow [ID_p|add_p|REQUEST|ts_i]_{k_{Q_i}}$$

6: p uses K_{Q_i} to verify if S_i is valid

7: **if** S_i is invalid **then**

8: p sends signature shares to each peer in Q_i

9: p constructs the routing table using coded symbols, determines the next quorum Q_{i+1} and uses the valid shares to construct a new valid S_i, which shows to peers in the next quorum that p's operation is legitimate

Final Step:

10: p sends S_{l-1} to Q_l, which is the destination, to prove the legitimacy of the lookup

Algorithm 2 NC-DHT: Quorum Peer $q \in Q_j$

Upon receiving a request by p:

1: **if** p's request is legitimate **then**

2: q sends its signature share to p

Upon receiving S_{i-1} **and** ts_i **from** p:

3: q verifies S_{i-1} using $K_{Q_{i-1}}$ and validates ts_i

4: **if** signature is valid **then**

5: q sends its signature share, public key of next quorum $K_{Q_{i+1}}$ and the q-th **coded symbol**

Upon receiving signature shares from p:

6: q verifies each share using public key share $\hat{K_{Q_i}}$ and sends valid shares to p

these symbols can be represented with c bits. Consequently, $D = c(s_Q - t_Q)$.

Given the $(s_Q - t_Q)$ data symbols corresponding to a certain D-bit routing table, s_Q "coded" symbols in the corresponding codeword are obtained as linear independent combinations of the $(s_Q - t_Q)$ data symbols over $GF(2^c)$. The code specification is part of the specification of the system.

The $(s_Q, s_Q - t_Q)$ Reed-Solomon code has the following useful property: Any $s_Q - t_Q$ (coded) symbols in a codeword can be used to compute the corresponding $s_Q - t_Q$ data symbols, and therefore, the corresponding D-bit routing table.

For completeness, we summarize the relationships between the code parameters:

- s_Q coded symbols in each codeword, corresponding to $(s_Q - t_Q)$ data symbols,
- $s_Q \leq 2^c - 1$, and
- $D = c(s_Q - t_Q)$

This implies that $s_Q \leq 2^{D/(s_Q - t_Q)} - 1$, and $D \geq (s_Q - t_Q)\log_2(s_Q + 1)$. Thus, we need $D = \Omega(\log n \log\log n)$, since typically s_Q is in the order of $O(\log n)$. This statement is true in Chord-like DHTs [18], [19]. In Chord [5], there are $O(\log n)$ entries, in which each entry contains node's ID (identifier) of at least $O(\log n)$ bits. Thus, D is lower bounded by $\Omega(\log^2 n)$ bits. Consequently, such a $(s_Q, s_Q - t_Q)$ Reed-Solomon code over Galois Field GF(2^c) exists in similar structures. In particular, NC-DHT is built on top of the quorum topology considered in [20]–[22].

B. Adding Target Privacy

a) Overview: The core idea of our design is to *hide* target's identity by *not* providing lookup key to quorums. Instead the lookup initiator asks each quorum (along the lookup route) for their routing table. The naive way is for each quorum peer to transmit its entire routing table, and the initiator simply uses the majority voting to determine the correct routing table. While this incurs the same message complexity as in RCP-I, the message size is prohibitively large.

In NC-DHT, we use error detection code to transmit the routing tables so that the followings properties are ensured:

- The number of message complexity remains the same as RCP-I.
- The size of message each quorum peer transmits is only $O(1/\log n)$ of the routing table, since each peer needs to transmit a corresponding symbol (either data or coded symbol).
- The initiator is able to recover the routing table from correct quorum peers, owing to the property of error detection.

b) Threshold signature and DKG: As in [21], [33], we also assume the existence of threshold signature generated by distributed key generation (DKG). After the DHT stabilizes, a DKG instance is executed such that, at the end, each quorum Q_i has (K_{Q_i}, k_{Q_i}) – the (distributed) public/private key pair specicifically for Q_i.

Note that K_{Q_i} is only needed to be known to those peers that belong to or have links to quorum Q_i. Furthermore, every peer $p \in Q_i$ has a private key share $(k_{Q_i})_p$ of k_{Q_i}. The

corresponding public key share $\hat{K_{Q_i}}$ can be used to check each private key share and is only known to the peers in Q_i.

The quorum public/private keys, (K_{Q_i}, k_{Q_i}), is used to form a proof (for showing the legitimacy to the next quorum), and the public/private key shares (at each quorum peer), $(\hat{K_{Q_i}}, k_{Q_i})_p)$, is used to verify the information transmitted from other quorum peers.

A signature share $S_i = [m]_{(k_{Q_i})_p}$ is referred to a message m signed by p's private key share $(k_i)_p$. Due to the feature of threshold signature, in quorum Q, $t_Q + 1$ such shares from correct peers form a valid signature $S = [m]_{k_{Q_i}}$, a message m signed by the quorum's private key k_{Q_i}. Recall that by assumption, the number of faulty peers in quorum Q is upper bounded by t_Q.

c) Target anonymity in NC-DHT: In our algorithm, ID_p is p's identifier, add_p is p's IP address, $REQUEST$ is a flag indicating a lookup request, and ts is the time stamp. We also refer **coded symbol** to the symbol containing part of the routing table as discussed in Section V-A.

Algorithm 1 presents the steps to be taken by the initiator (i.e., the node that initiates the lookup request), and Algorithm 2 present the steps to be taken by the peers in each quorum. Figure 1 presents the messages exchanged between the initiator p and each quorum Q_i.

C. Analysis

1) Robustness: If all four invariants of the underlying quorum topology mentioned in Section II-A are valid, then it is clear that NC-DHT can achieve lookup correctly. The correctness comes from the fact that the initiator node can correctly reconstruct the routing table from each quorum. Since at least $s_Q - t_Q$ peers are correct in each quorum Q in the system, there are always enough coded symbols from correct quorum peers. This statement holds in every step. Then by repeated application of the statement (proof by induction), the initiator eventually reach the correct target.

2) Target Anonymity: It is also obvious that our solution provides target anonymity, since faulty quorum peers only observe the request from the initiator, but the query key (which can be used to determine the location of the target) is never revealed.

Note that NC-DHT **always** provides target anonymity, while in OT-RCP-I [22], the query key is hidden in the *cryptographic sense* due to the underlying OT protocol. In other words, if the adversary has a very high computation power, then the identity of the target may be revealed. Due to the usage of coding, our solution preserves target anonymity, as long as the threshold signature scheme is secure.

3) Message Complexity: In our solution, we do not introduce extra messages, compared to prior DHTs that use quorum topology. Thus, the message complexity for completing a lookup operation – the total number of messages transmitted and received by all correct peers – is exactly the same as in RCP-I.

- The total message complexity in the system is at most $3s + 4s(l-2)$, where the quorum size is upper bounded by s, and l is the length of lookup (i.e., the number of quorums that need to be traversed).
- The message complexity at the initiator is also at most $3s + 4s(l-2)$, since we use iterative routing, where all communication initiate from the initiator.
- Each non-initiator peer that is at the initial or the intermediate quorum needs to transmit ≤ 4 messages.

In Chord-like DHTs [18], [19], both s and l are $O(\log n)$. Thus, our solution achieves $O(\log^2 n)$ message complexity. Note that although OT-RCP-I achieves the message complexity in same magnitude, it incurs 2 more messages at each intermediate quorum peers.

4) Message Size: We show that our solution has the maximum message size roughly equal to the one in RCP-I. In [20], [21], the message size is not explicitly listed. Without loss of generality, we assume that by "routing information for Q_{i+1}," Young et al. meant that entries corresponding to all the nodes in Q_{i+1} in the routing table. We further assume that each entry contain B bits. Thus, the total bits is $O(B \log n)$.

In NC-DHT, each coded symbol is $BE/(s_Q - t_Q)$ bits, where E is the number of entries in the routing table. In Chord-like DHTs with quorum topology, $E = O(\log^2 n)$, because each peer is connected to $O(\log n)$ quorums of size $O(\log n)$. By assumption, $s_Q - t_Q = O(\log n)$ (c.f. Section V-A). Thus, each coded symbol is $O(B \log n)$ bits, which is in the same order of magnitude as the message size in RCP-I.

5) Computation Overhead: In OT-RCP-I, the OT protocol contains s_Q exponentiations at each peer in the intermediate quorum Q, and 2 exponentiations at the initiator. The computation involves expensive operations such as hash, encryption, and group multiplication. In other words, OT protocol is slower than verifying and generating error detection code. Furthermore, when the system is relatively stable (i.e., a small churn rate), the coded symbol (of routing table) can be reused again and again to amortize the computation overhead. In contrast, OT protocol does not enjoy such amortized cost reduction.

VI. Evaluation and Discussion

Due to space limit, we briefly mention the key evaluation and extended features in NC-DHT.

A. Evaluation

Compared to prior systems, NC-DHT uses erasure coding. We demonstrate that it is indeed *not* a bottleneck. Our implementation utilizes the Reed-Solomon erasure coding algorithm from the Backblaze erasure coding library.[3] The library is employed to encode and decode the routing table at each node using Galois for finite field operations over 8-bit values. We randomly generate the routing table and measure the encoding and decoding times under different table size. We evaluate the time on our machine running on Windows 11 with Java version 20.0.1. It is equipped with 12 GB of RAM with Intel i5 4-core CPU, clocked at 2.4 GHz.

[3]https://github.com/Backblaze/JavaReedSolomon

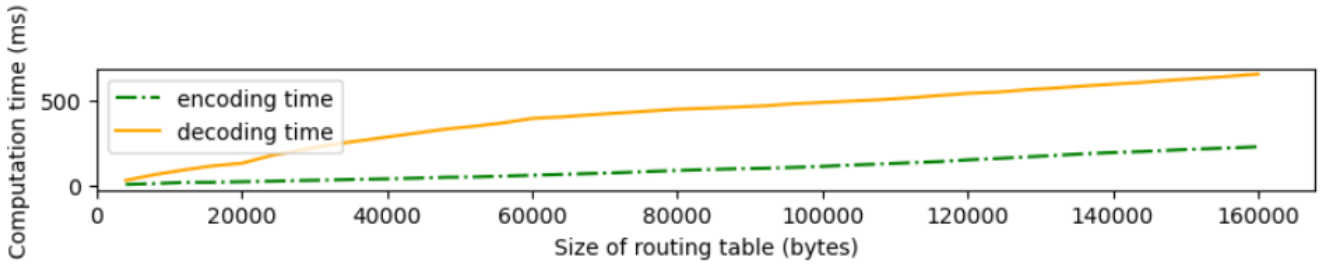

Fig. 2. **Coding/Encoding Time vs. Routing Table Size.** The configuration has $S_Q = 250$ and $t_Q = 50$.

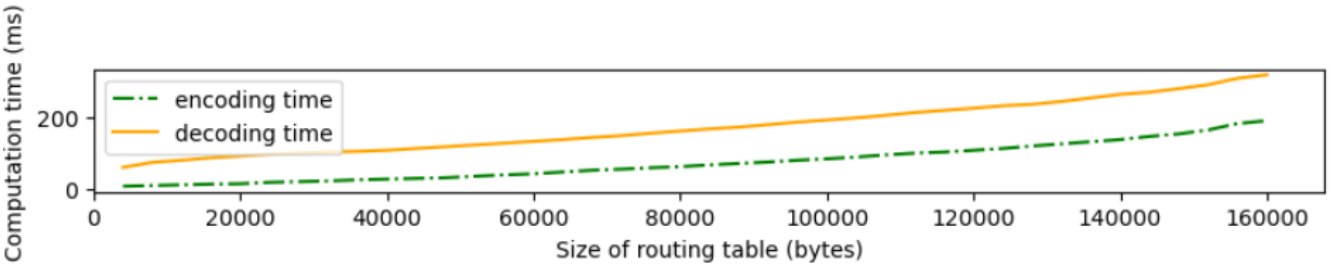

Fig. 3. **Coding/Encoding Time vs. Routing Table Size.** The configuration has $S_Q = 130$ and $t_Q = 30$.

Our implementation encodes and decodes data across multiple shards, consisting of both data and parity shards. The core components include the ReedSolomon class for encoding and decoding, supported by Matrix for matrix arithmetic, and Galois for finite field operations over 8-bit values. Random number generator for generating test data. This implementation offers simplicity and efficiency for encoding and decoding data. The implementation also includes tracking and outputting statistics related to encoding and decoding times under various conditions.

Figure 3 presents the encoding and decoding time for different routing table size (in Bytes). The top plot has $s_Q = 250$ and $t_Q = 50$, which translates to 200 data shards and 50 parity shards. In other words, at most 50 peers could by Byzantine faulty, and the initiator can still recover the necessary information. The bottom plot has the configuration with $S_Q = 130$ and $t_Q = 30$.

Compared to typical communication latency in wide-area network (in the order of hundred milliseconds), we conclude that our mechanism is indeed lightweight.

B. Enhanced Robustness

All prior solutions of building DHTs with quorum topology does not always satisfy *Goodness*. That is, with small probability $(1/n)$, some quorum may contain more than $1/3$ of faulty peers. In this case, RCP-I and OT-RCP-II fails.

On the contrast, in our solution, it is possible to reduce this probability by comparing routing tables from adjacent quorums. This is possible due to the following two observations:

- An initiator can obtain the whole routing table from the quorum, and
- In Chord-like DHTs, the routing tables at adjacent nodes (quorums) have lots of overlaps

Therefore, in our design, the initiator can obtain many routing tables from different quorums and choose the quorum that appears in most routing tables as the next hop.

In short, NC-DHT is more robust in the sense that the initiator is able to retrieve information from more quorums (to identify false information by quorums that fail the Goodness property).

C. Enhanced Privacy

In both RCP-I and RCP-II, the initiator only obtains the identity of next quorum, while in our solution, it obtains the entire routing table. Consequently, the initiator has the freedom to choose the next quorum in our system.

This important feature allows us to apply existing techniques to further hide initiator identity, e.g., [23], which include *initiator anonymity, query unlinkability, Fingertable correctness*, and *Fingertable trustiness*. Furthermore, due to the assumption of *Inter-Quorum Communication*, peer can issue an "inspection" to check whether the routing table at the predecessor quorums is correct or up-to-date. Thus, the check helps identify faulty peers and helps stabilize routing tables at each quorum.

On a high-level, quorums can periodically use the inspection to identify faulty nodes in each quorum and make sure the Fingertable (used for performing lookup) is correct and trustworthy. Moreover, the initiator can ask other (correct)

nodes to perform lookup and query to hide its identity and query.

VII. Summary

This paper presents a new DHT system, NC-DHT. Our system has two main salient features: (i) NC-DHT tolerates Byzantine faults, and (ii) it provides several anonymity guarantees such as initiator anonymity, query unlinkability and target privacy. To the best of our knowledge, NC-DHT is the first DHT system that provides all these important features. The key design behind our system is the integration of quorum topology and network coding-based techniques.

References

[1] Bittorrent. BitTorrent.com.

[2] Jarret Falkner, Michael Piatek, John P. John, Arvind Krishnamurthy, and Thomas Anderson. Profiling a million user dht. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 129–134, New York, NY, USA, 2007. ACM.

[3] Moritz Steiner, Taoufik En-Najjary, and Ernst W. Biersack. A global view of kad. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 117–122, New York, NY, USA, 2007. ACM.

[4] Liang Wang and Jussi Kangasharju. Measuring large-scale distributed systems: case of bittorrent mainline dht. In *P2P*, pages 1–10. IEEE, 2013.

[5] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *Networking, IEEE/ACM Transactions on*, 11(1):17 – 32, feb 2003.

[6] Petar Maymounkov and David Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 53–65, London, UK, UK, 2002. Springer-Verlag.

[7] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '01, pages 161–172, New York, NY, USA, 2001. ACM.

[8] Ahmet Kurt, Suat Mercana, Enes Erdin, and Kemal Akkaya. Enabling micro-payments on iot devices using bitcoin lightning network. In *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–3. IEEE, 2021.

[9] Yahya Hassanzadeh-Nazarabadi, Alptekin Küpçü, and Öznur Özkasap. Lightchain: Scalable dht-based blockchain. *IEEE Transactions on Parallel and Distributed Systems*, 32(10):2582–2593, 2021.

[10] Randhir Kumar and Rakesh Tripathi. Implementation of distributed file storage and access framework using ipfs and blockchain. In *2019 Fifth International Conference on Image Information Processing (ICIIP)*, pages 246–251. IEEE, 2019.

[11] Kazim Rifat Ozyilmaz and Arda Yurdakul. Designing a blockchain-based iot with ethereum, swarm, and lora: The software solution to create high availability with minimal security risks. *IEEE Consumer Electronics Magazine*, 8(2):28–34, 2019.

[12] Dan S. Wallach. A survey of peer-to-peer security issues. In *Proceedings of the 2002 Mext-NSF-JSPS international conference on Software security: theories and systems*, ISSS'02, pages 42–57, Berlin, Heidelberg, 2003. Springer-Verlag.

[13] Arjun Nambiar and Matthew Wright. Salsa: a structured approach to large-scale anonymity. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS '06, pages 17–26, New York, NY, USA, 2006. ACM.

[14] Apu Kapadia. Halo: High-assurance locate for distributed hash tables. In *In The 15th Annual Network and Distributed System Security Symposium (NDSS), To Appear*, 2008.

[15] Peng Wang, Nicholas Hopper, Ivan Osipkov, and Yongdae Kim. Myrmic : Secure and robust dht routing. Technical report, Digital Technology Center, University of Minnesota at Twin Cities, 2007.

[16] Moni Naor and Udi Wieder. A simple fault tolerant distributed hash table. In *In Second International Workshop on Peer-to-Peer Systems*, pages 88–97, 2003.

[17] Prateek Mittal and Nikita Borisov. Shadowwalker: peer-to-peer anonymous communication using redundant structured topologies. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09, pages 161–172, New York, NY, USA, 2009. ACM.

[18] Amos Fiat, Jared Saia, and Maxwell Young. Making chord robust to byzantine attacks. In *Proceedings of the 13th annual European conference on Algorithms*, ESA'05, pages 803–814, Berlin, Heidelberg, 2005. Springer-Verlag.

[19] Baruch Awerbuch and Christian Scheideler. Towards a scalable and robust dht. In *Proceedings of the eighteenth annual ACM symposium on Parallelism in algorithms and architectures*, SPAA '06, pages 318–327, New York, NY, USA, 2006. ACM.

[20] Ian Goldberg Maxwell Young, Aniket Kate and Martin Karsten. Practical robust communication in dhts tolerating a byzantine adversary. In *30th International Conference on Distributed Computing Systems (ICDCS 2010)*, 2010.

[21] Ian Goldberg Maxwell Young, Aniket Kate and Martin Karsten. Towards practical communication in byzantine-resistant dhts. *IEEE/ACM Transactions on Networking (ToN)*, 2012.

[22] Michael Backes, Ian Goldberg, Aniket Kate, and Tomas Toft. Adding query privacy to robust dhts. *CoRR*, abs/1107.1072, 2011.

[23] Qiyan Wang and Nikita Borisov. Octopus: A secure and anonymous dht lookup. In *32th International Conference on Distributed Computing Systems (ICDCS 2012)*, 2012.

[24] Jon McLachlan, Andrew Tran, Nicholas Hopper, and Yongdae Kim. Scalable onion routing with torsk. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09, pages 590–599, New York, NY, USA, 2009. ACM.

[25] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. Secure routing for structured peer-to-peer overlay networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):299–314, December 2002.

[26] Benjamin Fabian and Tobias Feldhaus. Privacy-preserving data infrastructure for smart home appliances based on the octopus dht. *Computers in Industry*, 65(8):1147–1160, 2014.

[27] Michael Backes, Ian Goldberg, Aniket Kate, and Tomas Toft. Adding query privacy to robust dhts. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, pages 30–31, New York, NY, USA, 2012. ACM.

[28] J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in P2P File Sharing Systems. In *IEEE Infocom*, Miami, FL, USA, March 2005.

[29] Jian Liang, Naoum Naoumov, and Keith W. Ross. *The index poisoning attack in P2P file sharing systems*. 2006.

[30] Prithula Dhungel, Di Wu, Brad Schonhorst, and Keith W. Ross. A measurement study of attacks on bittorrent leechers. In *Proceedings of the 7th International Conference on Peer-to-peer Systems*, IPTPS'08, pages 7–7, Berkeley, CA, USA, 2008. USENIX Association.

[31] Marc Sánchez Artigas, Pedro García López, Jordi Pujol Ahulló, and Antonio F. Gómez-Skarmeta. Cyclone: A novel design schema for hierarchical dhts. In Germano Caronni, Nathalie Weiler, Marcel Waldvogel, and Nahid Shahmehri, editors, *Peer-to-Peer Computing*, pages 49–56. IEEE Computer Society, 2005.

[32] Andriy Panchenko, Stefan Richter, and Arne Rache. Nisan: Network information service for anonymization networks. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 141–150, New York, NY, USA, 2009. ACM.

[33] Qiyan Wang, Prateek Mittal, and Nikita Borisov. In search of an anonymous and secure lookup: attacks on structured peer-to-peer anonymous communication systems. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 308–318, New York, NY, USA, 2010. ACM.

[34] Max Schuchard, Alexander W. Dean, Victor Heorhiadi, Nicholas Hopper, and Yongdae Kim. Balancing the shadows. In Ehab Al-Shaer and Keith B. Frikken, editors, *WPES*, pages 1–10. ACM, 2010.

[35] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: the second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.