# Guard: Secure Routing in Skip Graph

Sanaz Taheri Boshrooyeh and Oznur Ozkasap
Department of Computer Engineering, Koç University, İstanbul, Turkey
{staheri14, oozkasap}@ku.edu.tr

***Abstract*—Skip Graph is a distributed hash table (DHT) which acts as the underlying infrastructure in various P2P applications such as cloud storage and online social networks. The basic operation in Skip Graph is the search which is done in a fully decentralized manner. Any misbehavior of peers during the routing of a search query heavily degrades the system functionality. Security of search queries is the missing feature of Skip Graph, and applying existing DHT-based solutions on a Skip Graph reduces the search operation's efficiency and degrades the performance of query processing and response time. In this work, we propose Guard, the first fully decentralized secure search mechanism for Skip Graph, that provides authenticated and reliable search operation in a fully decentralized manner. *Guard* secures Skip Graph against the Sybil attack and routing attacks in the presence of the malicious colluding nodes. In contrast to the existing solutions, security of our design is formally proven.**

## I. PROBLEM DEFINITION AND MOTIVATION

Skip Graph [1] is a DHT-based routing infrastructure, which used in several P2P systems [4], [5], [2], [7], and also as an alternative infrastructure in many DHT-based services such as online social networks [9], search engines [3], and P2P storage systems [8]. In Skip Graph, each node is known by its two identifiers: *numerical ID* and *name ID*. The search for numerical ID [1], done in a fully decentralized manner, is the most common operation of Skip Graph. Nodes joining and accessibility to each other as well as their data items rely on the search for numerical ID. Any malicious behavior of nodes toward corruption of search operation directly affects the functionality of the system. Skip graph is vulnerable to *routing attacks* where malicious nodes try to drop, manipulate, misdirect or give the wrong reply while routing a search query, as well as the *Sybil attack* where a malicious node inserts an arbitrary number of fake nodes to the system to perform a large-scale attack.

Authentication of search queries is a missing feature of Skip Graph. The existing applicable DHT-based solutions are either probabilistic by conducting the same search in different paths and go with the majority of responses, or increase the response time and communication overhead by frequently pinging all the neighbors of each node to verify the node's trustworthy [12], or centralized by benefiting from a trusted third party (TTP) that frequently checks the correctness of search queries [11], [10]. As a solution to the search authentication problem in Skip Graph we propose *Guard* mechanism.

## II. GUARD

*Guard* employs identity based threshold signature scheme (IBTHS) [6]. Each peer is associated with a private and public key, where the private key is used for signing a message and the public key is utilized to verify the authenticity of both the signed message and the signer. Any publicly known identity (e.g., peer IP address) can be used as public key. The corresponding private keys are issued by TTP. In a (t,k) IBTHS, the signature key under a single identity is shared among $k$ parties such that any subset of $t$ parties can jointly provide a valid signature on a given message. *Guard* consists of two main entities: a TTP and nodes (i.e., peers).

**Registration**: Each node of Skip Graph is assigned a unique numerical ID by TTP. Numerical IDs are bounded to the nodes' physical identities such as MAC address. Then the TTP sends the node the signature key of the numerical ID associated with that physical identity. Node's name ID is the hash of the node's numerical ID, where the hash function is collision resistant, selected and publicized by TTP.

**Skip Graph construction**: Nodes come together to construct the Skip Graph. We assume that the Skip Graph is full i.e. all the name IDs in the name ID space are available in the Skip Graph. The nodes' connection in Guard is as in a regular Skip Graph with the following slight difference. The extreme nodes of level zero (the nodes with the smallest and largest numerical IDs) are also connected to each other (i.e., the nodes form a circular linked-list at level zero, the reason is explained in guard assignment). Upon the determination of each node's lookup table, the node gets a signature per each entry of its lookup table from the neighbor addressed by that entry. The message that each neighbor signs contains the numerical ID of the owner of the lookup table, the level and the position of the neighbor in that lookup table e.g. Level2, Right. The set of generated signatures per lookup table is called the table proof. We assume that all the nodes behave honestly during the Skip Graph construction i.e. ultimately, each node possesses an intact lookup table and table proof.

**Guard Assignment**: Each node is assigned three randomly selected nodes (among the existing nodes) named **guards**. The guards are responsible for checking the node's behavior while routing the search requests and to prevent any corruption. Each node has one main guard and two side-guards. In order to prevent a malicious node from circumventing its immediate neighbors, the node's side-guards are the main guards of its left and right neighbors at level zero. It is easy to verify that if a malicious node can not circumvent its immediate neighbors, he would not be able to circumvent any other node as well. We require that extreme nodes at level zero to be connected to each other so that all the nodes would have three guards. The name ID of each node's main guard is computed by applying a keyed permutation function on the node's name ID. The key of the permutation function is only known to TTP (hence nodes are not aware of their guards before asking TTP). In the first step of guard assignment, each node authenticates itself to TTP where TTP generates and sends some random strings

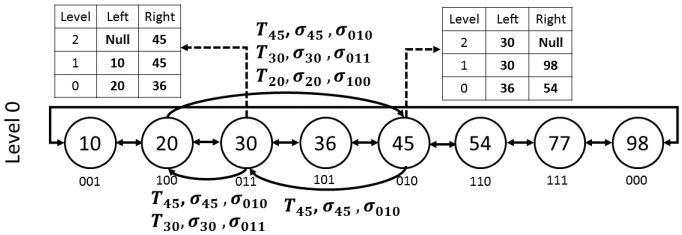
ISBN 978-3-901882-94-4 

to the node. If the node generates valid signature per string under its numerical ID's signature key, TTP authenticates the node. Afterward, the node delivers its lookup table and table proof to TTP. TTP verifies the validity of the table proof then computes the name IDs of the node's main and side-guards (using the nodes' name ID and the entries of the node's lookup table at level zero). Node searches its guards (by named ID) and asks them to connect to TTP. Guards are also required to authenticate themselves to TTP. Then, using a (3,3) identity based threshold signature scheme, TTP shares the signature key of the nodes' name ID (which was not already issued) among the node's guards such that the presence of all the guards is necessary to generate a valid signature. Guards also receive a copy of the node's lookup table and table proof.

**The search phase:** Nodes follow the regular search algorithm of Skip Graph. However, while routing a search request, nodes also generate and transmit information to provide provable security against routing attacks. Each search query is associated with a unique random string nonce. A fresh random nonce is generated per initiated search query to prevent the replay attack. While routing a search query, each intermediate node (including the search initiator) generates a routing transcript as a string with the following format $R||F||T||I||Q||N$. $R$ is the numerical ID of the routing node i.e. the generator of the transcript, $F$ and $T$ denote the numerical IDs of the preceding and subsequent nodes (relative to the routing node) on the search path ($F$ is set to NULL for the search initiator), $I$ denotes the numerical ID of the search initiator, $Q$ corresponds to the queried numerical ID and $N$ is the nonce. Routing node signs its routing transcript under its numerical ID's signature key. The routing node also obtains a signature for the transcript under the name ID' signature key from its guards. Guards provide the signature if the transcript matches to the node's lookup table (a copy of the lookup table is sent to the guards during the guard assignment). The combination of the transcript and the two signatures on the transcript is called the node's routing proof. Then, the node forwards its transcript and the routing proof alongside with all the transcripts and proofs generated by the preceding nodes, to the next hop. This procedure continues until the result of the search is determined. The last node on the search path (who is either the owner of the queried numerical ID or is the node whose numerical ID is the greatest numerical ID less than or equal to the queried numerical ID) creates its transcript and sets the $T$ field to null. It sends back all the transcripts and proofs to the search initiator. Figure 1 depicts a sample search in Guard.

## III. Security Analysis

**Security against Routing Attacks**: A routing attack results in a biased search response, either false positive or false negative. A false positive response happens if a malicious node forges a valid signature for a non-existing node. It is impossible due to the security of the signature scheme. The false negative response occurs if a malicious node arbitrarily routes the search query to circumvent the queried numerical ID. This is infeasible since the malicious node's lookup table is shared with its guards that check the correctness of node's routing behaviour. The guards have an intact copy of a malicious node's lookup table due to the following reason. Each node




$T_{45}$=45|| Null ||30||45||20||nonce $T_{30}$=30||45||20||45||20||nonce $T_{20}$=20||30||Null||45||20||nonce

Fig. 1: Guard: Sample search query initiated by node 45. Each circle represents a node with numerical ID embedded, and name ID at the bottom. Figure shows the connection of Skip Graph nodes at level zero. Left and right tables correspond to the lookup tables of nodes 30 and 45, respectively. $T_{45}$, $T_{30}$ and $T_{20}$ are the routing transcripts generated by node 45, 30 and 20, respectively. The value of nonce is determined by node 45. The concatenation operation is denoted by ||. The $\sigma$ with the subscript of the numerical ID and name ID represents signatures that the corresponding owner of IDs issues on its own transcript using the signature key of the IDs.

has two common guards with each of his immediate neighbors at level zero (the main guard of each node is the side-guard of his left and right neighbors). Thus, as soon as the neighbors of the malicious node share their lookup tables with their guards, the inconsistency between the malicious nodes' lookup table and its neighbors is detected by guards. It implies that the malicious node has to deliver the original lookup table to its guards. We argue that colluding between a malicious node and its guards (or immediate neighbors at level zero) is unlikely. First, recall that the malicious node cannot control the selection of his guards and neighbors. Now, assume that a malicious node colludes with $f$ other nodes where $f$ is a small constant due to the Sybil attack protection. Hence, only with a negligible probability $(\frac{f}{n})^3$ all the node's guards (or $\frac{f}{n}$ probability one of its neighbors) might be selected from the node's colluding set ($n$ is the total number of existing nodes).

**Security against Sybil Attack**: In *Guard*, nodes' numerical IDs are bound to their unique and unforgeable physical identities. Thus, no malicious node controls more than one node i.e. numerical ID in the Skip Graph. Hence, the system becomes resistant against Sybil attack.

In the poster presentation, we aim to provide the details of the Guard algorithms, the security analysis and extensive comparison with the related work.

## References


[1] J. Aspnes and G. Shah. Skip graphs. *TALG*, 2007.

[2] S. Batra and A. Singh. A short survey of advantages and applications of skip graphs. *IJSCE*, 2013.

[3] B. Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of P2P systems*, 2003.

[4] S. El-Ansary, L. O. Alima, P. Brand, and S. Haridi. Efficient broadcast in structured p2p networks. In *IPTPS*. Springer, 2003.

[5] Y. Hassanzadeh-Nazarabadi, A. Küpçü, and O. Ozkasap. Laras: Locality aware replication algorithm for the skip graph. In *IEEE NOMS 2016*.

[6] F. Li, W. Gao, G. Wang, K. Chen, and X. Wang. Efficient identity-based threshold signature scheme from bilinear pairings in standard model. *IJIPT 7*, 2014.

[7] T. Shabeera, P. Chandran, and S. Kumar. Authenticated and persistent skip graph: a data structure for cloud based data-centric applications. In *ACM CSS 2012*.

[8] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 2001.

[9] S. Taheri-Boshrooyeh, A. Küpçü, and Ö. Özkasap. Security and privacy of distributed online social networks. In *IEEE ICDCSW, 2015*.

[10] G. Urdaneta, G. Pierre, and M. V. Steen. A survey of dht security techniques. *ACM Computing Surveys (CSUR)*, 43(2):8, 2011.

[11] P. Wang, I. Osipkov, N. Hopper, and Y. Kim. Myrmic: Secure and robust dht routing. *U. of Minnesota, Tech. Rep*, 2006.

[12] Q. Wang and N. Borisov. Octopus: A secure and anonymous DHT lookup. In *IEEE ICDCS*, 2012.