WhanauSIP: A Secure Peer-to-Peer Communications Platform

by

Raymond Cheng

Submitted to the Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2010
[September 2010]

Author .... [illegible] ....
Department of Electrical Engineering and Computer Science
August 20, 2010

Certified by ..... [illegible] ....
Christopher Lesniewski-Laas
Ph.D. Candidate
Thesis Supervisor

Certified by .... [illegible] ....
Dr. Frans Kaashoek
Professor of Computer Science
Thesis Supervisor

Accepted by .... [illegible] ....
Dr. Christopher J. Terman
Chairman, Department Committee on Graduate Theses

WhanauSIP: A Secure Peer-to-Peer Communications Platform

by

Raymond Cheng

Submitted to the Department of Electrical Engineering and Computer Science
on August 20, 2010, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis presents a novel mechanism for achieving secure and reliable peer-to-peer communications on the Internet. WhanauSIP merges a Sybil-proof distributed hash table with the session initiation protocol (SIP) to enable instant messaging, audio chat, and video conferencing that is resilient to censoring, eavesdropping, and forgery. Performance and security evaluations performed on the PlanetLab network demonstrate that the majority of resource lookups return within five seconds. These results indicate that WhanauSIP delivers practical performance with respect to call session initialization latency for voice-over-IP, without having to rely on any central servers. Furthermore, the tests demonstrated that lookup performance was minimally affected during a Sybil key-clustering attack, illustrating the network's resilience to malicious adversaries. This thesis delivers three software packages for public use: a general Whanau distributed hash table implementation, a WhanauSIP gateway, and a desktop IM/VoIP client.

Thesis Supervisor: Christopher Lesniewski-Laas
Title: Ph.D. Candidate

Thesis Supervisor: Dr. Frans Kaashoek
Title: Professor of Computer Science

Acknowledgments

This project began a few years ago, when I had a "there must be a better way" moment. We live in a digital world where there is a growing ability and willingness to invade privacy. Additionally, countries and organizations can use their leverage over centralized services to monitor and censor their citizens. I believe information should be free and people have a right to free speech, but I didn't know how to make it so. It was really in the world of PDOS that I gained the skills that I needed to fulfill my vision.

I owe a great amount of gratitude to Chris Lesniewski-Laas for his deep insights, novel ideas, and continuing support. Every meeting with him not only brought clarity to the project, but also clarity to my vision. He has helped me grow tremendously and I am lucky to have had him as an advisor.

I would also like to thank Professor Kaashoek. He has a magical ability to turn complex ideas into a flowing story. His comments and suggestions have been wonderfully useful and it has truly been an honor to work with him.

Last but not least, I need to thank my pea. Serena, you have helped me stay focused. You have helped me with my writing and presentation skills. You have inspired, motivated, and kept me excited. You have supported me day in and day out, and I could not thank you enough for just being by my side.

Contents

List of Figures

Chapter 1

Introduction

Real-time communications on the Internet have been growing in demand, fueling a major shift in how people communicate. Whether on their mobile device, laptop, or desktop computer, any user should be able to connect to their friends by addressing a unique string of characters. Centralized protocols provide a convenient mechanism to map users to physical machines, enabling organizations to bring voice-over-IP (VoIP) and instant messaging (IM) to the market quickly using protocols such as XMPP [30] and SIP [29]. As VoIP and IM services proliferate, users are becoming increasingly aware of the problems that come with centralized services. Rather than trusting service providers and policy-makers to protect users, the technology should have built-in mechanisms to protect privacy, resist attacks, and prevent censorship. This thesis presents an alternative to centralized communications, leveraging a variety of peer-to-peer (P2P) technologies to protect data privacy, integrity, and reliability.

Providing a secure method of user and resource location has several unique characteristics. The user's identification or address-of-record (AoR), such as *alice@mit.edu*, does not change. However, the mappings between address and physical location can change frequently. Address-to-location mappings are small and entail low storage cost. These records must be easily updated as users move around and change their location or device. The records must be stored in a way so that any user can reliably look up all other users in the system. The lookup scheme must have low latency, such that when Alice decides to call Bob, it takes a short time for the software to locate Bob and initialize a session. The system should also handle a uniform load. That is, each lookup may be searching for a needle in a haystack, but it must be done reliably and quickly. Additionally, the system should have a flexible and reliable bootstrap mechanism to lower the start up cost and ease adoption.

Many of the popular schemes in use today rely on centralized means of control and proprietary protocols. While current tools provide convenience, there remains a number of unsolved problems.

Privacy: Many prevalent services require that users trust them with their data. As processing power and data storage become cheaper, service providers have gained an unprecedented ability to catalog the activities of their users. As a consequence, there have been a myriad of concerns regarding user privacy and control over their data [26]. For example, there have been instances of employee misconduct in violating user privacy [14]. Warrantless seizures of data are also possible in a number countries, including in certain cases in the United States [1]. Technology should protect its users without having to rely on government regulation, and proper service provider conduct. While some technologies allow encryption for security, encryption may not always be between the two end-users. Some services only encrypt traffic between servers and clients, leaving open the possibility for monitoring of messages [35].

Censorship: Some services do protect privacy by encrypting the data from user to user [39]. End-to-end encryption prevents the traffic from being read during any point of transmission between the two users. However, centralized mechanisms are vulnerable to being censored. For example, SIP allows encryption to protect the contents of conversations, but all users in a domain connect to a central login server. By using traditional blocking techniques such as DNS-hijacking or IP-blocking, countries can shut down entire services [16]. There have been instances of such censorship in the past [15, 23].

Malicious Attacks: Some have tried to decentralize infrastructure even further, to prevent a simple takedown via DNS-hijacking. The address-to-location mappings can be moved away from a central server and distributed amongst the users in the system in a peer-to-peer (P2P) fashion. P2P systems have the advantage of being highly scalable, fault-tolerant, robust, and self-organizing. For example, current P2P-SIP schemes use distributed hash tables (DHT) to perform user registration and lookup. However, current implementations of DHTs have been shown to be vulnerable to the Sybil attack. In this attack, adversaries create numerous false identities to influence the system's behavior. For example, a malicious user could misroute information, destroy information, or generally mislead any other clients. This attack has been shown in practice in systems such as Unvanish [37]. Existing P2P-SIP implementations use central authorities to thwart the Sybil attack, which reintroduce a single point of failure [13].

This thesis introduces WhanauSIP, a new combination of technologies to provide a secure communications platform that is resistant to attacks and censorship. Whanau, a distributed hash table (DHT) that is provably secure against the Sybil attack, stores the mappings between users' addresses and locations [20]. New user registrations will be closely tied into the join procedure of Whanau, providing its security benefits. Upon call initialization, users request the appropriate record in the DHT, and use existing mature SIP [29] technology to create a secure direct connection for voice and text data.

WhanauSIP requires that each user in the system store $O(\sqrt{n}\log(n))$ entries in their routing tables, where n is the number of users in the system. Whanau is a one-hop DHT, provably providing $O(1)$ lookup time [20]. Currently, two applications of WhanauSIP exist. The first application is a plug-in to SIP Communicator [6] that implements the WhanauSIP protocol. SIP Communicator is an audio/video Internet phone and instant messenger that supports several popular protocols, including SIP, XMPP, AIM/ICQ, MSN, Yahoo, and IRC. Using this desktop client, users can initiate chats and calls using a familiar graphical user interface. The second application is a WhanauSIP gateway daemon that is currently running across the PlanetLab network [5]. The gateway enables unmodified SIP clients to initiate calls to WhanauSIP clients.

This thesis is divided into the following sections: Chapter 2 will illustrate the design goals of WhanauSIP, Chapter 3 will discuss the design decisions and how they meet the intended goals, Chapter 4 will describe the software implementation, Chapter 5 will evaluate WhanauSIP's performance, Chapter 6 will compare other IM and VoIP protocols, and Chapter 7 will conclude the thesis.

Chapter 2

Goals

WhanauSIP is designed to be a practical instant messaging and voice-over-IP platform. Accordingly, it must contain all of the functionality that users expect from a robust communications client. Users register their accounts to establish a global identity. Then, they can establish conversations with any individual that has properly registered in the system using media in the form of text, audio, or video. If users disconnect from the network, there must be mechanisms for the users to rejoin the network. This chapter describes the goals that motivated the design of WhanauSIP.

2.1 Security Goals

WhanauSIP is designed to be a global system capable of handling a network that spans around the world. Because access is unconstrained, the underlying protocol must be secured to resist to a range of attacks such as the Sybil attack [37], Eclipse attack [34], and the key-clustering attack [20]. Users should not be able impersonate each other and users must have a decentralized mechanism for verification of data integrity. In order to be truly decentralized and peer-to-peer, security must be achieved without central elements, such as a certificate authority.

Some existing systems use puzzles and CAPTCHAs to rate-limit malicious users [10], where users are forced to solve a cryptographic problem or recognize words in an image before joining the network. However, this thesis aims to do better. Computational and networking resources are becoming less expensive by the day. It is questionable whether a rate-limiting scheme is effective against a sizable botnet. Even a small number of bogus identities gives an attacker considerable power to influence the system. Furthermore, WhanauSIP should resist the predominant forms of censorship, namely IP-block, DNS-hijacking, and filtering by traffic analysis [16].

2.2 Performance Goals

WhanauSIP should be able to scale properly such that even when millions of users are active in the system, lookups are still efficient. Lookup times must scale sublinearly and the protocol must entail sublinear storage costs per node in the distributed hash table. The Whanau distributed hash table performs lookups in $O(1)$ time [20].

However, it is not enough to say that a system has $O(1)$ time lookups. In order to be a practical system for use, a hard limit must be imposed on call latency. For example, when a user wants to chat, the time it takes to locate the other user and initialize a complete session must take less than five seconds. Although arbitrary, any additional wait could cause users to quit in frustration. Ideally, wait time should be as low as possible.

2.3 Usability Goals

In order to ease adoption, the user must be able to interact with the software in the same way that users interact with existing popular client software. The software must provide chat, voice, and video capabilities, as well as maintain presence data. Any additional effort the user must take to learn a new protocol may act as a barrier to using the software. On a similar note, the service must bootstrap without user intervention.

Additionally, cross-compatibility with existing SIP networks is a natural fit. WhanauSIP users should be able to contact any SIP user. Likewise, all SIP users should be able to contact any WhanauSIP user without installing any additional software or configuration using existing SIP URI semantics [29].

Chapter 3

Design

In WhanauSIP, the address-to-location mappings are distributed across the users of the system in a distributed hash table (DHT) overlay, using the Whanau protocol. This property marks the main difference from centralized mechanisms, where a server contains the address-to-location mappings for each user in the system. WhanauSIP can effectively be split into two parts, shown in Figure 3-1. Participants must first find the appropriate address-to-location entry in the DHT. Once the location of the contact is acquired through the DHT, traditional SIP stacks are used to establish the session.

Each user begins by establishing network links with their friends, now called network peers. When a user wants to look up a user in the DHT, he sends a request that travels across the social network formed by all the network links [20]. Effectively, each user only accepts routing information from the people that they trust. Participants may join or leave, causing changes in the social graph. Furthermore, new network links can be formed with new users that were found through the DHT.

The chapter will cover each design decision in detail, as well as explain how each goal is satisfied by the decisions.

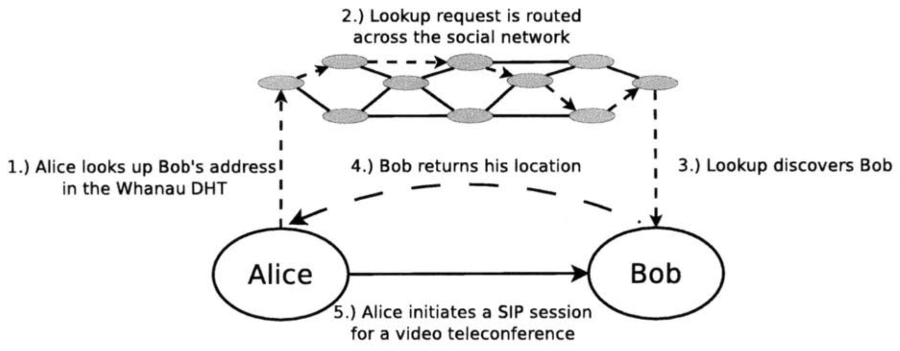

2.) Lookup request is routed across the social network
1.) Alice looks up Bob's address in the Whanau DHT
4.) Bob returns his location
3.) Lookup discovers Bob
Alice
Bob
5.) Alice initiates a SIP session for a video teleconference

Figure 3-1: Sequence of Events to Initiate a Call in WhanauSIP

3.1 Challenges

Trust: Secure peer-to-peer communications pose several challenges. First and foremost without a central authority that all users can mutually trust, users need another mechanism for protecting themselves. Early distributed hash tables simply trusted any routing information from other nodes in the system [33], making them vulnerable to the Sybil attack. Papers such as Whanau DHT [20] and SybilGuard [38] explored the possibility of explicitly defining trust relationships, using social networks as a protection mechanism against the Sybil attack.

Churn: Secondly, these networks exhibit potentially great amounts of churn, dynamically changing as nodes may leave or join in different places. Churn is inherently more difficult to manage than merely interacting with a central server at a static location. The implementation must be optimized to handle frequent failures and moderate churn while still delivering reasonable performance.

SIP Integration: Users in WhanauSIP are identified by a hash of their public keys (*i.e.* `whanausip:14AB35FD23C`) for reasons described in Section 3.2.1. However, SIP URIs follow very different semantics (*i.e.* `sip:alice@mit.edu:5060`) [29]. Not only are the lookup mechanisms in WhanauSIP and SIP very different, but the URI semantics differ substantially. WhanauSIP must address these differences for smooth interoperation.

Usability: WhanauSIP uses a unique DHT that requires additional parameters that would not normally be required of instant messaging (IM) services. For example, nodes must be able to explicitly differentiate mere contacts from the peers that it trusts to receive routing information. These additional requirements of the DHT must be satisfied while masked behind normal functions that the user performs.

Node	An abstract entity that represents the user in the social network Each nodes' functions are performed by the user's device
Peer	A neighboring node on the social network
Contact	Any node in the system that can be contacted using WhanauSIP
URI	Uniform Resource Identifier - a string of characters that identifies a WhanauSIP or SIP user (*i.e.* `sip:alice@mit.edu`)

Figure 3-2: Common Terminology Defined

3.2 Resource Location

3.2.1 User Identification

Users of WhanauSIP start by generating a cryptographic public/private key pair. These keys are used for encryption, signatures, and authentication. When users publish their location on the DHT, they store a self-certifying key-value pair (`public key hash, signed IP address`) [36, 24, 25]. A SHA1 hash of the user's public key is used as the key, while the IP address signed by the private key, is stored as the value. By using standard cryptographic techniques, any user can verify the validity of any record stored in the DHT. Consequently, the DHT only needs to ensure availability, since any client can verify the record. As an optimization, nodes in the DHT can drop any records where the self-certification fails.

The public/private key is also used to establish SSL/TLS links between nodes [17]. SSL provides mechanisms for verifying that each side has the proper keys. Even if a client were to receive the wrong address, a man-in-the-middle attack would be detected during the SSL handshake. Without cracking current cryptographic schemes, attackers would not be able to impersonate another user without the user's private key.

The hash of the user's public key is now the user's address-of-record (AoR), which sidesteps the issue of fairly distributing names with semantic value. This thesis recognizes that in order to fairly delegate human-readable addresses, a distributed system needs a central authority. Certificate authorities can use X509 certificates to map chosen addresses like `alice@mit.edu` to public keys. RELOAD currently uses central certificate authorities to perform this task and access control [19].

3.2.2 Call Setup Process

WhanauSIP users are located using the URI syntax: `whanausip:public_key_hash`. When Alice wants to talk to Bob, she will look up Bob's public key hash in the DHT. Bob's IP address is returned in the form of a SIP URI, `sip:<Bob's public key hash>@IP-address`.

SIP allows users to initialize direct connections to each other in registrar-less mode, if the IP addresses are known, bypassing all SIP proxies and registrars. WhanauSIP runs a full SIP stack in this mode, allowing users to connect directly with the target using the SIP URI returned from the DHT. Furthermore, this stack allows all WhanauSIP users to contact any SIP address. Unmodified SIP clients can also contact a WhanauSIP user by their direct SIP URI containing the IP address, but not by WhanauSIP URI. The next section will discuss the possibility of gateways for SIP clients to perform resource location in WhanauSIP, without modifying their existing software.

Note that contact with a user does not imply a peering link in the DHT social network. For example, users may try to contact URIs that are published on a website, but they may not trust routing information from them.

3.3 Whanau DHT

3.3.1 DHT Structure

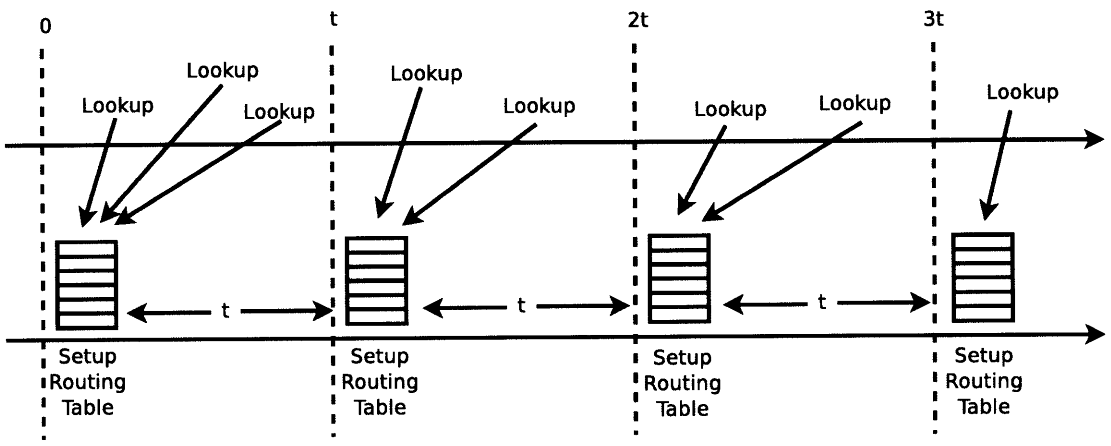


Figure 3-3: Whanau DHT Repeats Setup at Regular Intervals to Update Routing Tables

The Whanau DHT is used to store self-certifying key/value pairs containing `(public key hash, signed IP address)` for each user in the network. In effect, the DHT acts as a rendezvous service for WhanauSIP clients. The Whanau DHT protocol occurs in synchronized steps. In each step, the social network is used to create static routing tables that do not change during the entirety of the step. Once the routing table is in place, they are used by clients to perform $O(1)$ lookups. These operations are repeated at regular intervals in sequential synchronized steps. Therefore during each step, joins, leaves, and updates do not come into effect until the next step when the routing tables are updated, as shown in Figure 3-3. The Whanau DHT paper discusses the trade off in shortening step intervals and bandwidth consumption [20]. Currently, the WhanauSIP network on PlanetLab [5] runs synchronized steps every thirty minutes.

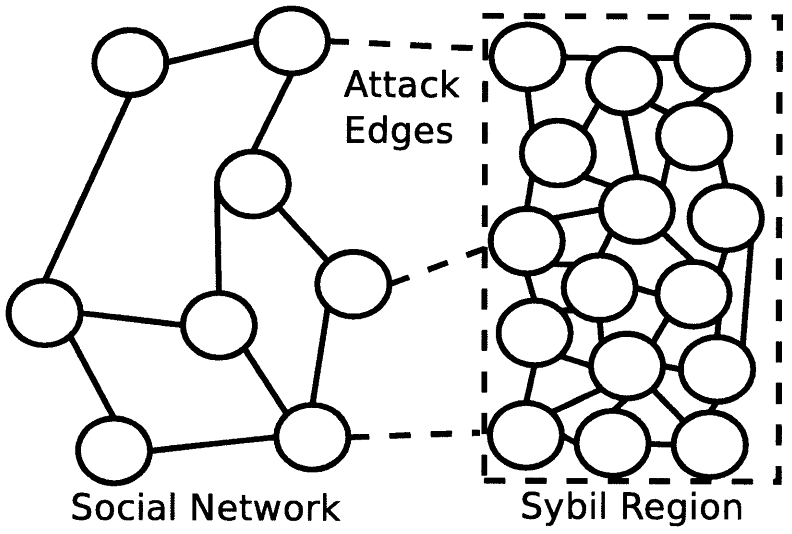


Figure 3-4: Division of the Social Network into an Honest Region and a Sybil Region

3.3.2 DHT Security

Whanau DHT was chosen as the data store for address records due to its unique resistance to the Sybil attack. Currently, no other P2P-SIP proposal to date employs a purely decentralized method to defeat the Sybil attack. An attacker must be able to convince honest users to form a social peering link in order to have any influence on the system. Each connection between an attacker and an honest node is called an attack edge. Figure 3-4 shows an example of a social graph with attack edges. The attacker's influence is limited by the number of attack edges it can form, not the number of nodes it can generate. We assume that the social engineering required to convince honest users to form trust relationships outweighs the cost to generate a Sybil. Because an attacker has control over all Sybil nodes, it has the ability to make an arbitrary number of social trust links in the Sybil region. Whanau DHT uses this property to limit an attacker's influence on the network's stability. Whanau DHT tolerates up to $O(n/\log(n))$ attack edges, where n is the number of well-connected honest nodes. Whanau DHT also requires that the social network be fast mixing, a property that can generally be found in real social networks [20].

3.4 Application Scenarios

Current implementations of WhanauSIP require that nodes actively participating in the Whanau DHT have a public IP address. They are not hidden behind a NAT and generally change their location infrequently. The following scenarios describe possible applications of WhanauSIP.

3.4.1 Desktop Chat Clients

Alice downloads a WhanauSIP chat client from the Internet. She then runs this client on a computer that has a publicly accessible IP address. If she sits behind a NAT, she will enable port forwarding to her machine. All of her friends do the same and she adds each as a contact in the software. She also needs to add the current IP address of at least one contact. From this connection, she can find the IP addresses of the rest of her contacts. For convenience, there may be tools to automate the tracking of friends' contact information using other social networks as a bootstrapping mechanism, described in Section 3.5.2.

3.4.2 WhanauSIP Gateways

WhanauSIP gateways allow unmodified SIP clients to initialize sessions with users on the WhanauSIP network, using only their public key hash and unmodified SIP clients. The gateway would join the Whanau DHT like any other node. Assume that this gateway has the domain name `foo.com`. When a SIP user, `alice@mit.edu`, wants to talk to a WhanauSIP user with URI `whanausip:1A2B3C`, Alice simply addresses `sip:1A2B3C@foo.com`. The gateway at foo.com will perform a lookup on the public key hash in the address and redirect the call to the appropriate WhanauSIP user, by returning the SIP URI `sip:1A2B3C@IP-Address`

Note that another supernode could volunteer to be a WhanauSIP gateway with domain `bar.com`. In this case, Alice can address either `sip:1A2B3C@foo.com` or `sip:1A2B3C@bar.com`. The domain merely specifies which gateway to query. The use of cryptographic SIP URIs was inspired by work done by Seedorf [31]. Since SHA1 hashes in hexadecimal are generally long, a potential future optimization is to use different encoding schemes to shorten addresses.

3.4.3 Proxy Servers

Bob has several devices that he wants to ring when someone calls him at his WhanauSIP address. He maintains a WhanauSIP proxy on a server with a publicly accessible IP address. This server may run as a virtual machine in a cloud or it may be his desktop computer at home, which he leaves on. The server acts as a traditional SIP registrar and proxy, allowing all of his mobile devices to log in and associate with the proxy using unmodified SIP clients. When he wants to make a call to Alice from his cellphone, he initiates a session using the WhanauSIP proxy, which looks up the target's address in Whanau DHT and uses SIP to complete the call.

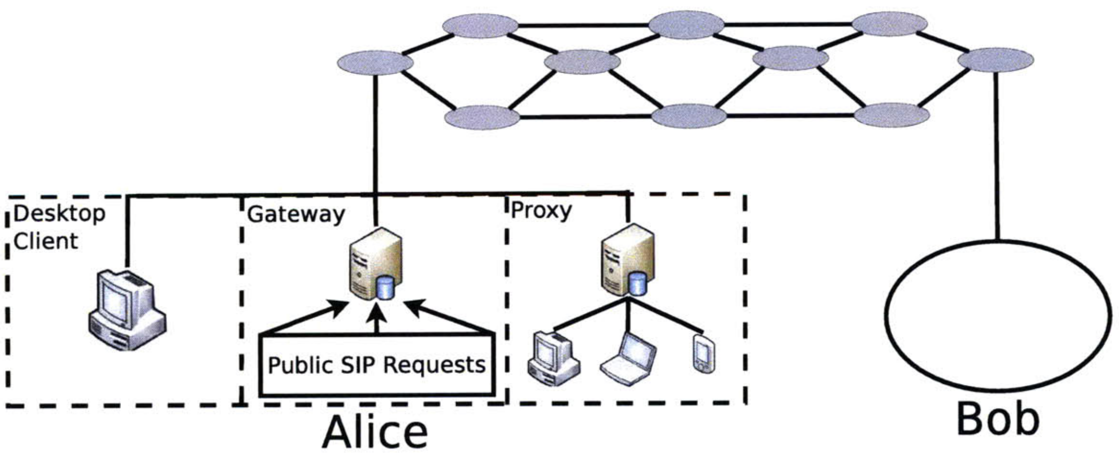


Figure 3-5: 3 Possible Endpoint Applications for WhanauSIP

3.5 Rendezvous Process

3.5.1 Independent Start

When the user first starts up the client, it should have stored the most recent IP addresses for each of its contacts in its log. If any contacts are active, the user can query this node for the addresses of the rest of its peers. This check will handle the case if some contacts and peers, but not all, have moved IP addresses.

If the user has not checked their client in a long period time such that all IP addresses are stale or if it is the first time loading the software, then there are no peers to query. In this case, the user must use a WhanauSIP gateway to perform lookups on its peers before continuing. The software can come with a list of suggested WhanauSIP gateways hardcoded into the package such that this process occurs transparently, but it is not required.

3.5.2 Bootstrapping from Other Social Networks

In order to bootstrap from other social networks, there must exist a central database that stores the public key hash and most recent IP address for each contact. When a user logs in, they update this database with their own information. Such a system would be easy to build in platforms such as Facebook API [3]. The OneSwarm project created such a facility for Google Contacts [18].

Although this feature introduces a centralized element, it is meant to be used as a one-time bootstrapping mechanism for convenience. Once the public key hashes of all of your contacts are retrieved, WhanauSIP users must operate independently of the central elements to preserve privacy and reliability. WhanauSIP can operate without using centralized bootstrapping services.

3.6 Satisfaction of goals

3.6.1 Security Goals

The Whanau DHT is used to protect the system from the Sybil and key-clustering attacks. All traffic is sent across SSL links, which encrypts and hides all information from monitors. SSL links also prevent the Eclipse attack [34] where users modify messages, as well as censoring based on traffic analysis. Because all traffic is encrypted and can run on any IP address or port, censors can only block entire IP ranges at best. The WhanauSIP gateways can be blocked for the same reasons that SIP proxies can be blocked. However with development, it may become easy enough for any node to start a gateway such that there will never be a shortage.

All private data such as text, voice, and video data is encrypted end-to-end, user-to-user using SSL. SSL ensures privacy and prevents man-in-the-middle attacks [17]. The use of self-certifying records and SSL handshakes ensures the integrity of data and prevents users from impersonating each other without knowledge of private keys.

3.6.2 Performance Goals

Whanau DHT has a $O(1)$ lookup time and requires $O(\sqrt{n}\log(n))$ entries of storage in each node's routing table [20]. Therefore it theoretically satisfies our performance goals. Refer to Section 5 for further evaluation.

3.6.3 Usability Goals

Using a full SIP stack and WhanauSIP gateways provides full SIP interoperability without any changes to existing SIP client code. Furthermore, joining and leaving a social network requires no work by the user. The locations of a users' peers are refreshed automatically using current peer contacts and WhanauSIP gateways. Bootstrapping is trivial in this case.

Finally, WhanauSIP clients contain all the functionality of SIP without any added complexity in the user interface. Users log in by loading a public/private key pair, which can be stored in a user configuration after being generated. Contacts and peers alike are listed in the buddy list. SIP and SIMPLE protocols handle presence data to check for the availability of the contacts, as well as establishing instant messages and calls to WhanauSIP and SIP users alike.

Chapter 4

Implementation

4.1 Overview

This thesis presents three software packages written in Java. Whanau DHT is implemented as a library, allowing any application to hook into the DHT for resource location. Given a set of network peers, Whanau DHT is responsible for setting up the social network, building the routing tables, and performing lookups.

The first application that uses the Whanau DHT library is a WhanauSIP gateway. The gateway acts as a node on the Whanau DHT and also listens to SIP messages on a separate port. When any SIP message is received, the WhanauSIP gateway parses out the username from the SIP URI and performs a lookup on the WhanauDHT. The result from the lookup is returned in a REDIRECT message, informing the client of the requested user's location.

The second application that uses the Whanau DHT library is the desktop client. WhanauSIP is implemented as a protocol plug-in for the SIP Communicator [6] software package. Using this graphical client, WhanauSIP users can initialize a node, form the social network by adding members to the buddy list, and initialize instant messaging, voice, and video sessions.

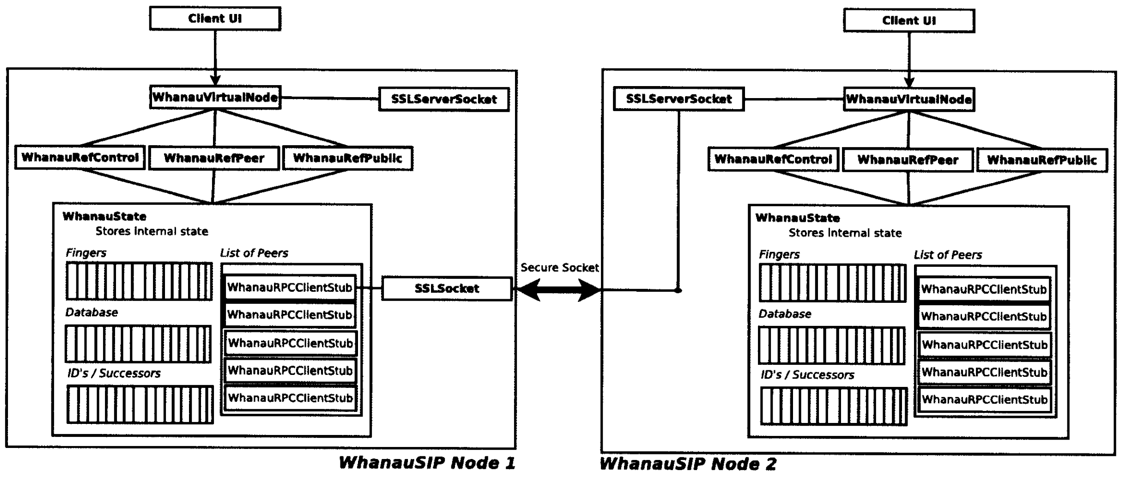


Figure 4-1: Overview of Whanau DHT Node Architecture

4.2 Organization

Figure 4-1 breaks down the overall organization of classes in a Whanau DHT node. A single instance of a Whanau DHT node consists of an interface, a data store, and core functions. WhanauState stores all of the node's state including the list of peers, finger table, database, and successors. The core functions are split into three classes, WhanauRefControl, WhanauRefPeer, and WhanauRefPublic, each containing a reference to WhanauState. WhanauRefControl performs all of the administrative functions including adding new peers and setting up the routing tables. WhanauRefPeer contains all of the functions that should only be accessed by peers on the social graph. Currently it only contains a method for random walks, which are used heavily by the Whanau DHT protocol. WhanauRefPublic contains all of the functions that are publicly accessible, such as retrieving the public key or returning layer IDs. Finally, WhanauVirtualNode is the outward facing interface. Initializing WhanauVirtualNode sets up the proper listener ports using SSLServerSocket. Refer to Appendix A for the Whanau DHT protocol pseudocode.

4.3 Secure Sockets and Access Control

Each node contains a public/private key pair, used to sign all of the records in the DHT. The same keys are used to initialize secure sockets over SSL between nodes. When node A performs an RPC to node B, A will check the public key from the SSL handshake to verify that B is actually on the other side. Similarly, B will use the peer's public key from the SSL handshake to perform access control. Any node can call methods in WhanauRefPublic, but only designated peers can call methods in WhanauRefPeer. This form of access control prevents directed graphs during any random walk. Additionally, each node specifies a set of public keys that are authorized to remotely call WhanauRefControl methods. While it may not be necessary for an ordinary desktop client, it makes remote administration more convenient and secure in environments such as the PlanetLab WhanauSIP gateway network.

It was found that Java Remote Method Invocation (RMI) [21] was ill-suited for this type of low-level access control due to the way it transported client socket factories across the network. For these reasons, RPCs are implemented by simply marshalling Java Objects over the SSLSocket.

4.4 Random Walks and Query Tokens

In order to optimize the retrieval of random walks for each node, a separate RandomWalkCache class controls the batch requests and caching for random walks. When the node requests for n random walks, s steps away, the RandomWalkCache will batch enough remote requests of length $(s-1)$ to fulfill the incoming request and fill its own cache. These requests are randomly distributed to all of its active network peers with uniform distribution. If any peer fails to return, it is marked as inactive for a fixed amount of time. RandomWalkCache uses locks and condition variables to ensure that there exists at most one thread to fill the cache for each length from s, $(s-1) \ldots 1$. For requests of length 0, RandomWalkCache simply returns the local node's record and a generated query token.

Requests to RandomWalkCache returns a full DHT record and a query token. The DHT record will be further discussed in Section 4.5. Certain public methods in the Whanau DHT protocol, such as lookup, should only be called at the end of a random walk. These methods require some computational work so the query token is passed in as a parameter to limit the number of times they can be called. Query tokens represent the work from the random walk and limit denial of service attacks on a particular node.

RandomWalkCache gains several benefits from this design. By batching requests instead of sending individual requests, network resources are better utilized. Everything is asynchronous and thread-safe. Therefore, malfunctioning or unresponsive nodes do not keep functioning nodes from proceeding. Furthermore, a node only pulls for more random walks if it needs them. For the purposes of this implementation, the size of the cache is just enough to satisfy all of anticipated requests from Whanau DHT setup.

4.5 General Key Value Checkers

This implementation of Whanau DHT is designed to support a number of applications. In order to allow different schemes of self-certifying key/value pairs, applications can pass in different key value checkers by implementing the KeyValueChecker interface. For example, an application may just publish arbitrary values but use the value's hash as the key.

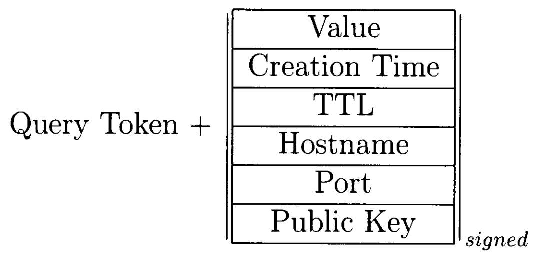


Figure 4-2: Contents of a Published DHT Record

In the WhanauSIP application, a signing key value checker is used. Figure 4-2 lays out the result from a random walk All DHT records contain the published value, creation time, and a time to live. In addition, it will contain the publisher's host, port, and public key. The entire record is then signed by the publisher's private key. The record is verified by verifying that the record is signed by the included public key. Then the public key is hashed and compared with the associated key.

4.6 Synchronization

As mentioned in Section 3.3.1, all nodes in the system must run setup in synchronized steps. Furthermore within a single run of setup, there are setup phases that must also be synchronized. In order to work, all nodes running Whanau DHT must synchronize their system times to an accurate clock.

The implementation uses condition variables to accommodate for timing discrepancies of up to 30 seconds. If a node requests information from a past phase that has not occurred yet, the remote method will block until the information is available. The client node waits up to a fixed limit before it tries again with another node.

In order to synchronize when setups start, nodes are hardcoded with a period value p, in milliseconds. Nodes wait until its coordinated universal time (UTC) aligns with an integer multiple of p from January 1, 1970. Nodes can optionally use NTP to synchronize their clocks.

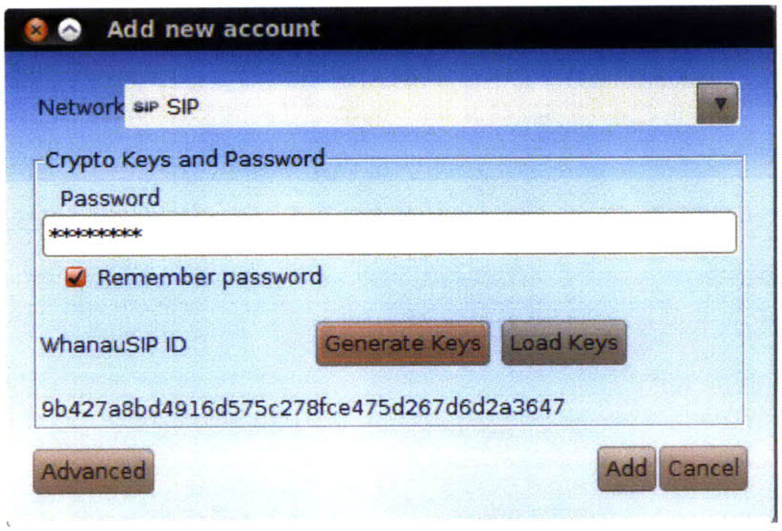
4.7 Applications

Two applications, the WhanauSIP gateway and the desktop client, were implemented with the Whanau DHT library. They each listen on two ports: one port listens for Whanau DHT traffic and the other port responds to incoming SIP traffic. The implementations also use the Java-based JAIN-SIP stack from NIST [4].

When a SIP message is sent to the WhanauSIP gateway, the gateway will parse the 'to' address URI. From the SIP URI `sip:username@domain.com`, the `username` is extracted. If it is a SHA-1 hash, then the gateway will perform a lookup on this hash in the Whanau DHT. The result from the lookup is wrapped in a REDIRECT message and send back to the client. Note that users can store any SIP URI in the DHT.

The desktop client extends the SIP Communicator software package. In order to maximize code reuse, the WhanauSIP protocol plug-in uses the SIP protocol plug-in as a base. Additional functions, such as looking up WhanauSIP URIs are added as features.

Users start by entering a password and either generating or loading a set of keys. Figure 4.7 shows the wizard used to add a new account. Once the account is registered into the client, the buddy list will show a new group, 'WhanauSIP Peers'. Figure 4-4 shows the buddy list and IM window. All network peers should be added as contacts into the "WhanauSIP Peers" group. In order to bootstrap, at least one peer's hostname and port must be added with the contact URI in the form: `whanausip:<public key hash>@host:port`. Notice that SIP URIs can also be added into the contact list, but they cannot be network peers. Also keep in mind that Whanau DHT will not be routed across a network peer until contact information has been mutually exchanged.



Network SIP

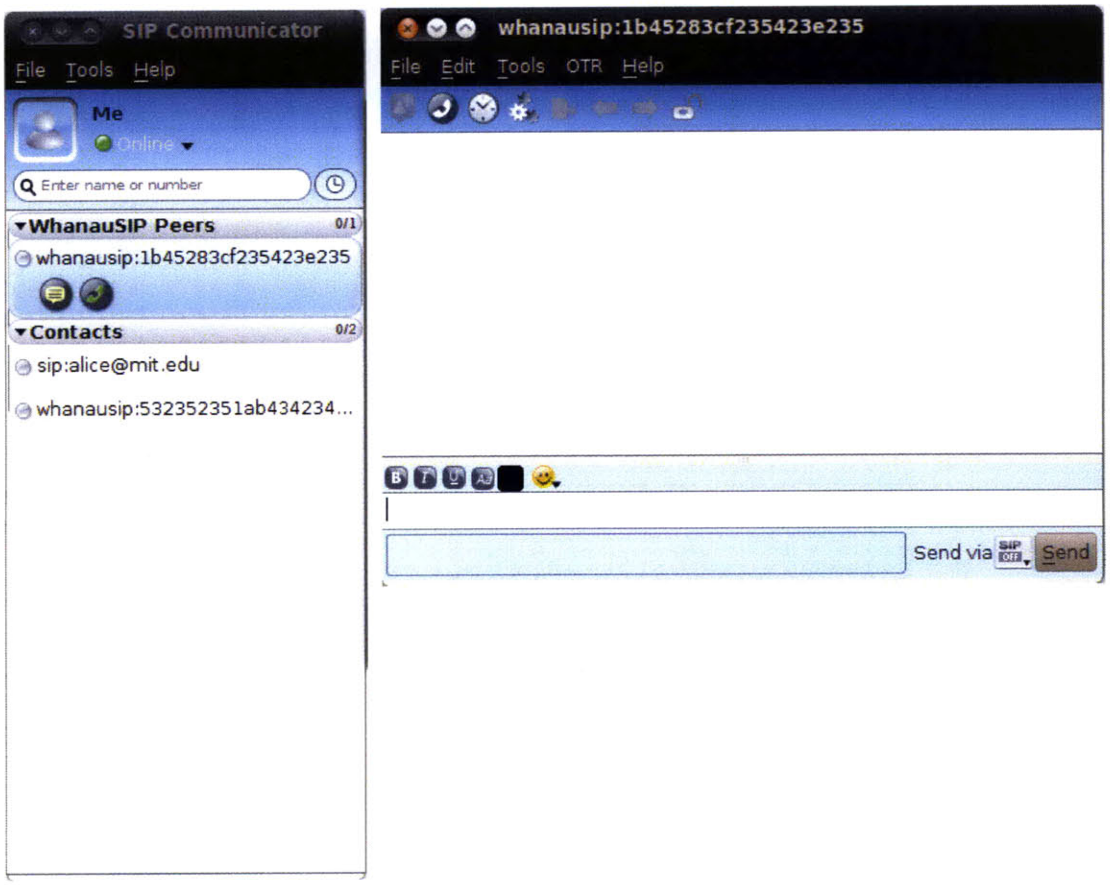
Password *********

Remember password

WhanauSIP ID

9b427a8bd4916d575c278fce475d267d6d2a3647

Figure 4-3: SIP Communicator screenshot: Add a new WhanauSIP account



WhanauSIP Peers 0/1

whanausip:1b45283cf235423e235

Contacts 0/2

sip:alice@mit.edu

whanausip:532352351ab434234...

Figure 4-4: SIP Communicator screenshot: Buddy list and IM window

Chapter 5

Evaluation

The following chapter will evaluate the performance of WhanauSIP across a real network, specifically measuring its ability to deliver the goals mentioned in Chapter 2. The measurements focus on performance and security. In this chapter, it is shown that (1) lookup times across a large WhanauSIP network provide reasonable performance for use with VoIP telephony and (2) WhanauSIP can withstand Sybil key-clustering attacks.

5.1 Test Environment

The WhanauSIP gateways are distributed on the PlanetLab network [5]. Approximately 500 physical machines are used, each running one instance of the software, providing 500 virtual nodes used for testing. Each node stores 50 entries in its routing table, a number low enough such that local routing tables cannot answer most requests but also high enough such that lookups can still succeed. Each node is connected to 5 other random nodes, forming a random graph. For all of the tests, random walks are performed with a length of 7.

Virtual nodes are controlled using RPC calls over SSL using the WhanauRefControl interface, described in Section 4.2. Using the RPC calls, a command and control computer can initialize new virtual nodes, modify the social network, set up the routing table, perform lookups, and query a variety of internal state.

5.2 Performance Evaluation

Whanau DHT theoretically performs lookups in $O(1)$ time. However due to the nature of varying CPU load, network availability, and round-trip network latency, lookups may take varying amounts of time to return. It is expected that with greater amounts of resources performing parallel lookups, the percentage of lookup success would also increase.

5.2.1 Setup Routing Tables

As described in Section 3.3.1, each of the nodes must synchronize their setup in order to set up the network routing table. Within each setup, there are also multiple phases for each DHT layer that must be carried out in lock-step. Therefore, each phase was given a fixed time window to accommodate small discrepancies in clock synchronization. Provided hardcoded windows of 110 seconds for each layer of the DHT and 120 seconds for synchronization, a five layer DHT should be properly set up in at most 670 seconds. The tests showed that each node finished `setup()` around 668 seconds on average.

5.2.2 Lookup Response Times

When a node calls `lookup` on a key, it will spawn a number of threads, r_l. Each thread will perform a random walk on the social graph and call `lookupTry` to the result of the random walk. This method will query a number of entries in its finger table that it suspects is the key's predecessor. If any of the threads returns a valid answer before a timeout, then that answer is reported. Otherwise, the `lookup` failed. The first measurement is performed in very lax time constraints to capture the asymptotic behavior of a lookup, which is then used as a control for the remainder of the measurements.

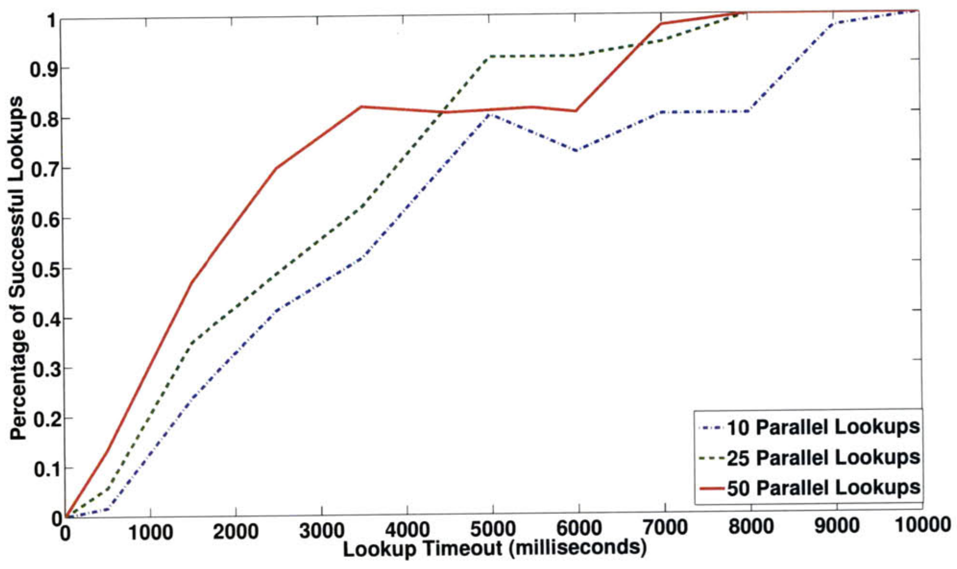


Figure 5-1: Percentage of Lookup Success vs Timeout for 10, 25, and 50 Lookup Threads: Using more lookup threads enables greater percentage of success in shorter time

Results Figure 5-1 shows lookup performance for 3 values of r_l for comparison. For each data point, 100 nodes were asked to perform `lookup` on the same key in parallel. The raw data is plotted in Figure 5-4. As expected, more threads increased the probability that at least one of them will result in success. If 80% lookup success is used as the comparison, $r_l = 10, 25, 50$ reach that level within 5.7, 4.3, and 3.5 seconds respectively. Thus with at least 25 threads, the performance goal for lookup latencies of less than 5 seconds is achieved.

Limitations From the shape of the graph, it is speculated that a subset of the nodes in the PlanetLab slice are under heavy load from a different experiment, causing them to stall for an amount of time before responding. 75% of the nodes respond in a timely fashion, whereas the top of the graph dips unexpectedly. Future experiments should enable maximum pre-existing load restrictions to select only responsive nodes. Furthermore, the amount of time to reliably return a lookup is consistently over several seconds. The most likely cause is the two back-to-back 3-way SSL handshakes that are required for each lookup, one to call `lookupTry` and one to call `query`. See Appendix A for more details. In the experiment, all lookups were executed in parallel. Future experiments could run the lookups in serial, to determine whether the lookups are constrained by CPU resources or network latency. The methodology in the experiment also inherently introduces an amount of error. Instead of measuring the exact time of each lookup, these experiments counted the number of successful lookups within a certain timeout value. While these tests still demonstrate general behavior, future tests should timestamp all lookups and plot a cumulative distribution function (CDF) instead.

5.2.3 VoIP Call Latency

Minimum	5.16 s
Median	5.30 s
Maximum	15.49 s

Sample Size = 10

Figure 5-2: Latency involved in setting up a VoIP session over WhanauSIP

In order to test the practicality of DHT lookups, the latencies in setting up calls between WhanauSIP users was evaluated. When a user requests to call another, two things happen. A lookup is served across the DHT to find the target user's location. Upon success, the users will negotiate a session, agreeing on properties such as the audio codec. Theoretically, the call latency should be identical to lookup performance, with an additional constant to account for session negotiation.

In a lab demonstration, a SIP user registered on MIT's server submits a call request to the WhanauSIP gateway. The gateway will look up the WhanauSIP user and return the value in the REDIRECT message. The SIP user will then proceed to negotiate with the WhanauSIP user directly. The SIP user uses the Sipdroid client on an Android G1 phone. The WhanauSIP user is using the SIP Communicator software package with WhanauSIP plug-in.

Results Figure 5-2 shows the call latency statistics for 10 sample calls. Multiple tests demonstrate that call setup occurs around five seconds consistently. Although this test only measures performance across the PlanetLab network, it is expected that the software will perform similarly when the software is deployed among real users with the same software. Occasionally lookups take substantially longer, on the order of 15 seconds. This is most likely caused by a depletion of the random walk cache and new random walks must be acquired before the lookup can proceed.

5.3 Security Evaluation

All aspects of the Whanau DHT protocol, including setup and lookups, rely on random walks across the social network. Sybil nodes were designed to always return itself when random walks reach it, acting like a black hole. If random walks ever venture into the Sybil region, they never escape. For the majority of remote calls, Sybil nodes will just return null. However, when a request to retrieve the Sybil's ID at a layer, it will perform a key-clustering attack by choosing IDs close to a specified target key.

As a result of this design, a larger number of attack edges would result in a larger representation of Sybil nodes in the routing table. It is expected that as the number of attack edges between honest nodes and Sybils increase, the number of resources required to successfully return a lookup grows exponentially. The key-clustering attack tries to convince honest nodes that Sybils are the predecessors to a target key.

Results Figure 5-3 shows the result of the Sybil attack on lookup performance. For each data point, the number of attack edges was specified in the social network. `Setup` was repeated to update the routing tables. Lookup tests were performed with an increasing number of parallel threads until the performance reached 80% of the asymptotic maximum. As expected, fewer attack edges would require fewer lookup threads. As the number of attack edges increase in the system, the number of lookup threads needed so that at least one would evade a black hole Sybil node and retrieve

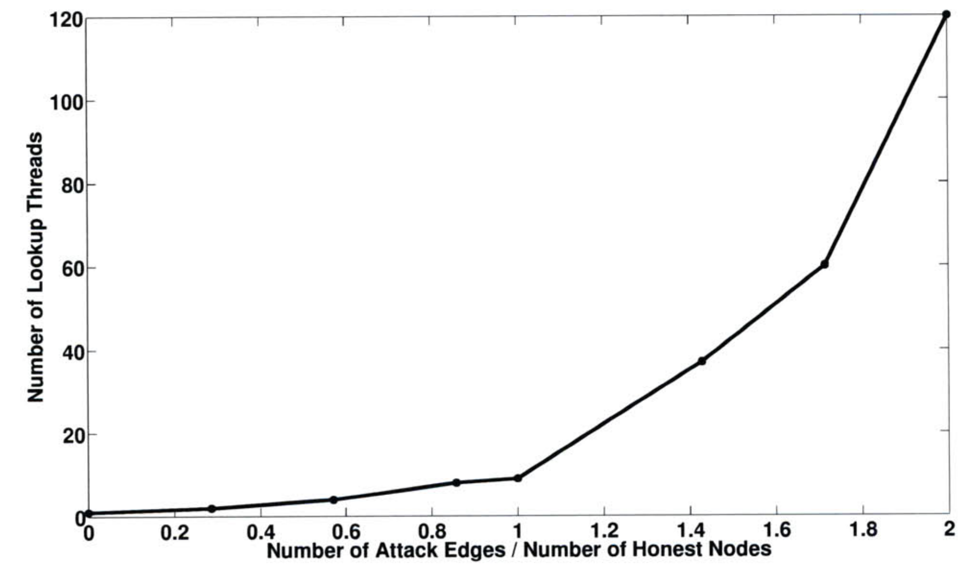


Figure 5-3: Minimum Number of Lookup Threads vs. Attack Edge Ratio: The number of threads needed to return at least one successful value increases exponentially with the number of attack edges

a valid value should exponentially increase. Even with twice as many attack edges as honest nodes, the PlanetLab network was still able to consistently perform lookups, albeit with more resources. The results demonstrate that the DHT is resilient to Sybil key-clustering attacks, achieving the security goal.

Limitations Because lookups were repeated until a performance mark was met, the data introduces a downward bias to the data. This error misrepresents the number of threads that are actually needed to achieve high percentage of success, given the number of attack edges. While this data still demonstrates the behavior of the system under attack, minimizing this measurement error is important to properly set the number of lookup threads in an implementation.

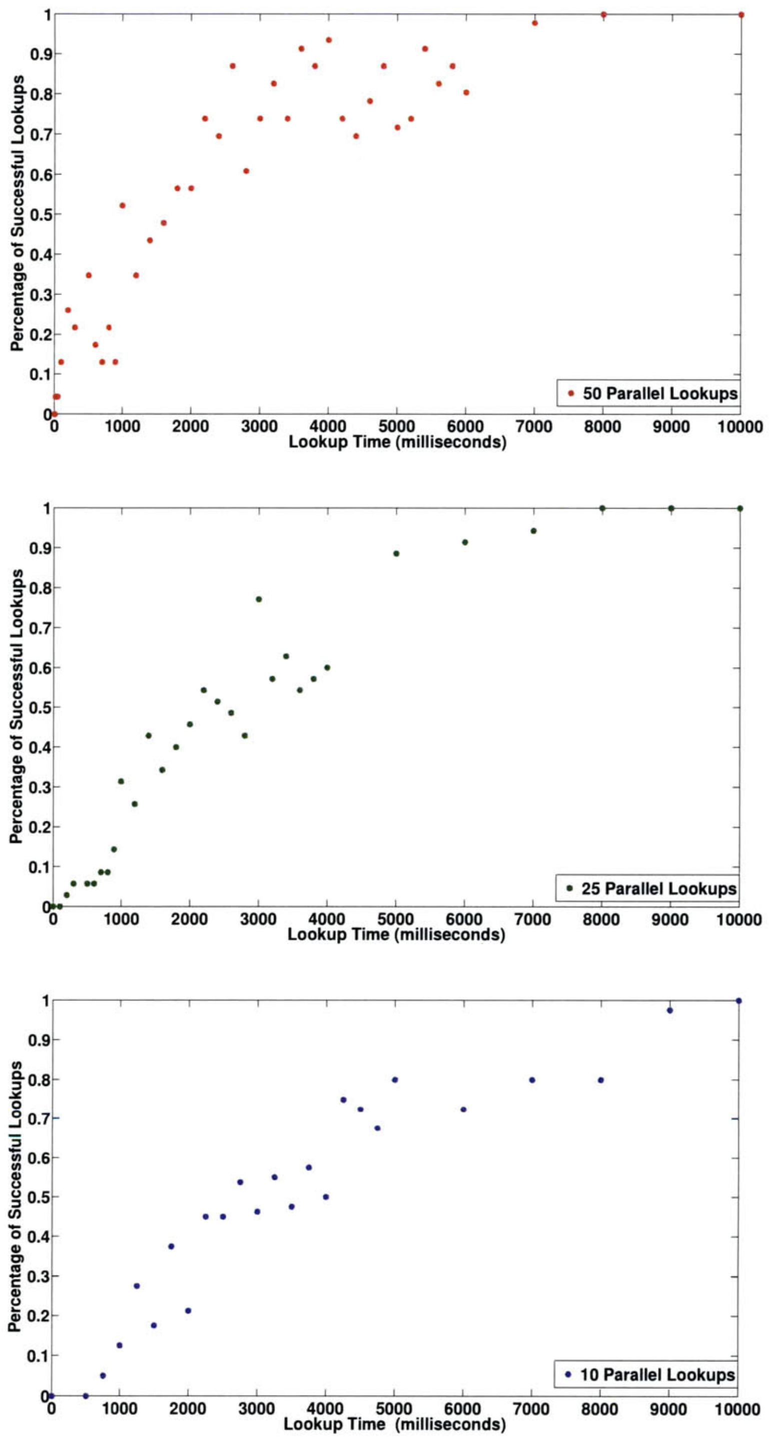


Figure 5-4: Percentage of Lookup Success vs Timeout for 10, 25, and 50 Lookup Threads: Raw Data

Chapter 6

Related Work

A number of established centralized methods exist for establishing instant messaging and voice-over-IP communications. Three of the most popular protocols include XMPP [30], SIP [29], and Skype [9]. Additionally, P2P-SIP is a large research effort to create peer-to-peer location service that integrates with existing SIP networks. A few of the major contributors to P2P-SIP include Columbia University, the College of William and Mary, Cisco, and the University of Hamburg. Current implementations [32, 11, 19, 10, 27, 8] focus on integration with SIP and resolving security problems which will be described later in this chapter. All of these protocols have different notions of security and decentralization. This chapter will attempt to clarify the differences and the unique capabilities they possess.

6.1 Session Initiation Protocol (SIP)

SIP [29] is an IETF-defined signaling protocol that includes standards for resource location and multimedia communication sessions. First, users register an address, such as `alice@foo.com`, with a registrar server. Each domain, such as `foo.com`, is responsible for hosting a registrar for administering the users in its network. When Alice wants to contact another user in another domain, such as `bob@bar.com`, she will contact `foo.com`'s proxy server, which will in turn contact `bar.com`'s proxy server, which will locate `bob@bar.com`. Standard DNS techniques are used to look up reg-

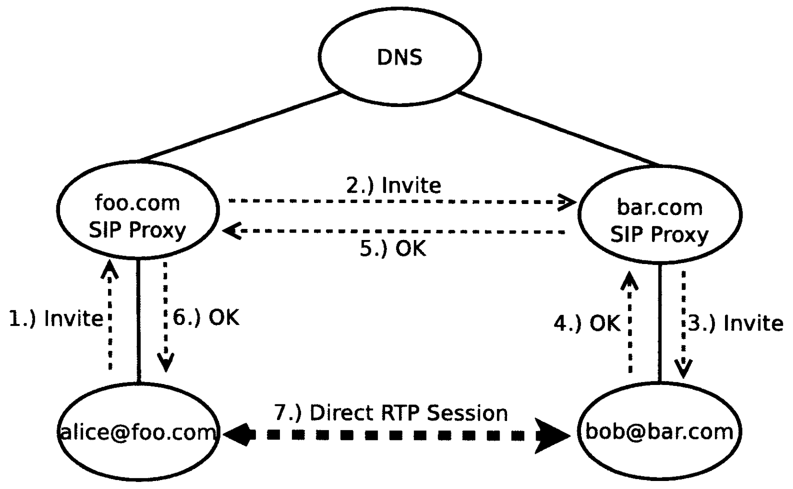


Figure 6-1: Traditional SIP: Sequence to Establish a Connection Between alice@foo.com and bob@bar.com

istrar and proxy servers. Once the proxy servers have located Bob, Alice and Bob initialize a direct session. SIP includes fine-grained negotiation mechanisms for agreeing on properties such as audio codecs [29]. To secure communications and privacy, SIP can be routed over TLS. The proper function of the entire protocol requires that the proxy servers remain alive. Therefore, system administrators can improve reliability using traditional redundancy techniques such as DNS and IP address takeover.

SIP is partially decentralized but not peer-to-peer. It provides mechanisms for anyone to start a SIP domain and anyone in any SIP domain can contact another. In this sense, the entire network is decentralized but there is a central point of failure for each domain.

It is possible to offload resource location to DNS and use SIP just to negotiate a direct session. This method is unattractive for a variety of reasons. DNS has a rigid hierarchy with relatively high startup cost [28]. Furthermore, it makes it difficult for users to update their location if they frequently move around as mobile clients would. This consequence is due to the fact that DNS relies heavily on caching for performance and it often takes many hours for cache entries to expire.

6.2 Extensible Messaging and Presence Protocol (XMPP)

XMPP was started as an open-source project called Jabber in 1999 to provide a decentralized platform for real-time instant messaging as well as presence information [30]. It has since been extended with additions like Jingle to provide rich-media communications, such as VoIP and video chat [22]. XMPP is decentralized but not peer-to-peer, much like SIP.

Similar to SIP, each domain, such as `foo.com`, maintains a central server. Users register an address with the server, such as `alice@foo.com`. When `alice@foo.com` sends a message to `bob@bar.com`, she sends the message to `foo.com`, which forwards it to `bar.com`, who finally sends it to `bob.com`. Notice that unlike SIP where proxies are used for resource location and messages are sent directly, all XMPP messages are routed through a network of servers [30]. Because all data across the network is encoded as XML, the extensions that enable video and voice generally send the data out of band.

XMPP depends on domain servers, much like SIP, and messages can be routed over SSL. Users can send their messages over SSL to an XMPP server, but end-to-end encryption between users is generally not included by default. Services like Google Talk use this property to offer logging of message traffic.

6.3 Skype

Skype is a free application that is deployed on a number of platforms. It uses a proprietary protocol, which makes it difficult to study and develop like open standards. Due to this fact, it does not provide interoperability with other protocols like SIP. The protocol is based on the Kazaa/FastTrack architecture, which contains its own routing overlay. Its API allows application developers to read and write text messages to a Skype application stream [2].

Skype is a hybrid between a centralized and a peer-to-peer system. All users log in through Skype's central login server. Joining nodes are assigned to a supernode by the Global Index Server. Supernodes perform a similar function to SIP proxies, as they maintain availability information about connected nodes and perform lookups by querying other supernodes. It is believed that this lookup uses a variant of flooding as Kazaa did, which is much less efficient than current DHT lookups [9].

6.4 Peer-to-Peer Session Initiation Protocol (P2P-SIP)

P2P-SIP replaces the SIP proxy and registrar with a distributed hash table while maintaining interoperability with traditional SIP networks. P2P-SIP tries to expand on previous SIP work in two main ways. By moving to a peer-to-peer system, the protocol would no longer require SIP proxies, which require maintenance and configuration costs. Instead, users lookup other users using a distributed hash table as a routing overlay, much like the case of WhanauSIP. The DHT must be able to perform all functions that SIP servers would normally provide, namely discovery, registration, and lookup [32].

DHTs have a simple put/get interface. Users can put key/value pairs into the hash table, and the routing layer is responsible for finding the proper node on the network to store the data. Techniques such as consistent hashing allow the network to delegate responsibility and determine consistent rendezvous points for key ranges. These structured P2P systems provide guarantees that unstructured protocols cannot. Unstructured protocols generally involve inefficient flooding algorithms which cannot reliably retrieve all data in bounded time. On the other hand, DHTs like Chord can look up key/value pairs in $O(\log(n))$ time, where n is the number of users in the system [33, 7].

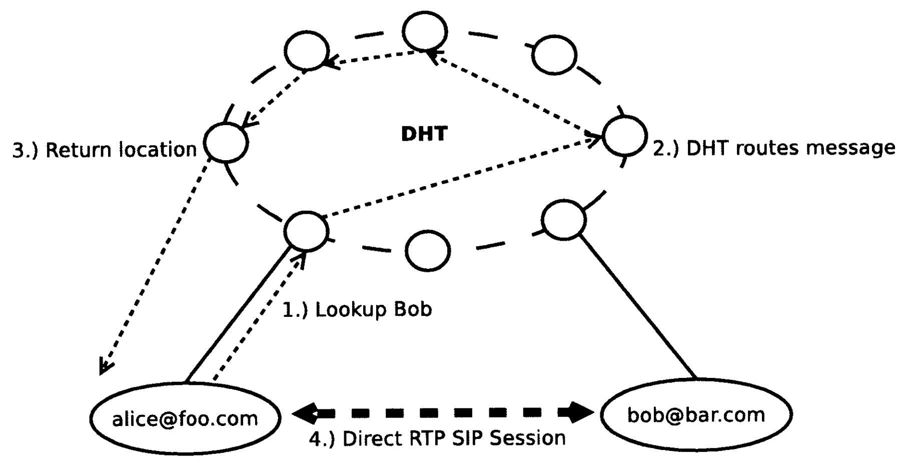


Figure 6-2: P2P-SIP: Sequence to Establish a Connection Between alice@foo.com and bob@bar.com

The following sections will describe some of the common elements as well as some of the unique design decisions between different P2P-SIP implementations. In particular, this proposal will give an overview of the security features of RELOAD from the IETF, OpenVoIP from Columbia University, P2PNS from the Institute of Telematics, and SIPDHT2 [19, 10, 8, 27].

6.4.1 General P2P-SIP Design

The DHT can be formed using all of the users in the system. However as a performance optimization, existing P2P-SIP implementations make distinctions similar to Skype supernodes. Users with higher bandwidth and no NAT traversal become the servers that form the DHT. Other users, such as mobile clients, simply perform lookup on one of these DHT nodes. Lookups are still performed in $\log(n)$ time but the DHT becomes more stable with less maintenance traffic in dynamic joins, leaves, and state transfer [32].

When a node joins the system, it performs a SIP REGISTER command on any DHT node. This operation effectively stores a binding between a name such as `alice@foo.com` with a physical location (IP). Similarly when a node leaves the network, it removes this binding with another REGISTER message.

When a client wants to perform an instant message or call setup, it will send either the MESSAGE or INVITE message respectively. The DHT node that receives this message will act as a SIP proxy and perform a lookup for the user's location. This location is then used to redirect the client to the desired node, where SIP will then initialize a session directly between the two nodes. This functionality is described more in depth in the SIP RFC 3261 and other P2P-SIP papers [29, 32].

Due to the nature of a DHT, P2P-SIP can provide much more advanced services. For example, by storing other data types in the DHT, it can provide storage of offline messages if a user is not available.

6.4.2 RELOAD

This working group at IETF 72 is the culmination of years of work from Columbia University, the College of William and Mary, Cisco, SIPeerior Technologies and Network Resonance. The 130-page draft is an unpolished glimpse into a future RFC for peer-to-peer resource location. RELOAD works as a layer between applications like SIP and a DHT. An implementation can choose any overlay routing scheme or distributed hash table written as a topology plug-in in RELOAD. RELOAD specifies exactly the structure of messages, error messages, and requirements that are sent over this overlay. Specifically, RELOAD requires that the topology plug-in implement join, leave, update, and lookup queries. After a resource such as a user, is found in this network, an ATTACH message is sent. This operation allows the two peers to communicate directly, where traditional SIP takes over to initialize a session [19].

Each overlay network has a unique domain name, such as '`mit.edu`'. In the network, each user generates a private/public key pair which it will use to secure communications. A user chooses its SIP address of record (AoR) by registering its keys with a central certificate authority (CA) [13]. This protocol does not require a global PKI, but it does require PKI on an organizational basis. The CA will issue a certificate to the user that contains the assigned SIP AoR (`alice@mit.edu`) and a Node-ID (hash of the user's public key), all signed by the trusted entity. All messages sent over the network are signed and sent over TLS links using these public keys which identify the users. Communications are secured using TLS-PSK or TLS-SRP. The RELOAD draft explicitly states that the only way to call P2P-SIP members in a different domain is to attempt to join their overlay which requires registration with a central certificate authority [19, 12].

6.4.3 OpenVoIP

OpenVoIP is an ongoing implementation developed at Columbia University. A few of these developers were also part of the RELOAD working group so many of the design elements are shared between these two. The current release of the program supports Chord, Kademlia, and Bamboo. It also integrates into the OpenWengo client and contains a test base on PlanetLab [8].

6.4.4 P2PNS

In contrast to RELOAD, which depends on a central certificate authority, the P2PNS project from the Institute of Telematics in Germany focuses their energy in making a purely peer-to-peer system. However, P2PNS design is very similar to RELOAD. Each overlay represents one network under one domain name. The DHT stores a binding from the AoR to the Node-ID as well as a binding from the Node-ID to the IP address [10].

The records containing the IP address are secured using self-certifying techniques [24, 36]. The IP address is signed using a public key and in this case the Node-ID is the hash of the public key. Similarly to RELOAD, communications are signed and encrypted over TLS to avoid Eclipse attacks where messages are modified [34]. In order to reduce the effects of a Sybil attack, it uses crypto puzzles to slow down the generation of Node-IDs at the cost of reducing the Node-ID space. Users are allowed to choose arbitrary AoRs on a first come first serve basis. In order to reduce the effects of an attacker claiming all desirable AoR addresses, it also uses crypto puzzles and per-node user limits to slow down these registrations. These protection schemes serve to slow down an attacker but they do not prevent them [10].

6.4.5 SIPDHT2

SIPDHT2 is yet another implementation of P2P-SIP, inspired by the IETF P2P-SIP working drafts. Therefore, it also contains many similarities with RELOAD. Its main notable difference is that it uses Passive Content Addressable Network (PCAN). This DHT requires that peers are invited, which adds some security benefits [27].

Chapter 7

Conclusion

This thesis introduces WhanauSIP, a secure P2P communications platform. The implementation comes in the form of a general Whanau DHT library, a WhanauSIP gateway, and a desktop client. Each package was built with the focus on ease of adoption, enabling WhanauSIP to become a viable alternative to other popular communication protocols. Tests showed the network's capability to scale properly while maintaining reasonable performance. Furthermore, these tests demonstrate the network's ability to continue serving requests even in the midst of attacks.

While this first prototype of the software has demonstrated basic functionality, more work is needed before the software is ready for release. The desktop client provides IM and VoIP functionality, but video and presence data must be implemented. Further work must also be performed to optimize network joins, lookups, and reliability.

Privacy, censorship, and cyber attacks are a growing concern in a world where Internet communications are becoming more pervasive. While popular protocols today offer convenience and partial security, there is no solution that provides full security in a complete P2P fashion. WhanauSIP offers a step in that direction by combining the security guarantees of Whanau DHT with the features of a SIP stack in order to provide users with the ability to communicate in an environment where monitoring and censorship of the entire network is simply not possible.

Appendix A

Whanau DHT Protocol Pseudocode

A.1 Setup Routing Tables

```
SETUP (stored-records(·), neighbors(·); $w, r_d, r_f, r_s, \ell$)
1  for each node $u$
2        do $db(u) \leftarrow$ SAMPLE-RECORDS($u, r_d$)
3  for $i \leftarrow 0$ to $\ell - 1$
4        do for each node $u$
5                 do $ids(u, i)$        $\leftarrow$ CHOOSE-ID($u, i$)
6                    $fingers(u, i)$    $\leftarrow$ FINGERS($u, i, r_f$)
7                    $succ(u, i)$       $\leftarrow$ SUCCESSORS($u, i, r_s$)
8  return $fingers, succ$
```

```
SAMPLE-RECORDS($u, r_d$)
1  for $j \leftarrow 1$ to $r_d$
2        do $v_j \leftarrow$ RANDOM-WALK($u$)
3           $(key_j, value_j) \leftarrow$ SAMPLE-RECORD($v_j$)
4  return $\{(key_1, value_1), \ldots, (key_{r_d}, value_{r_d})\}$
```

```
SAMPLE-RECORD($u$)
1  $(key, value) \leftarrow$ RANDOM-CHOICE(stored-records($u$))
2  return $(key, value)$

RANDOM-WALK($u_0$)
1  for $j \leftarrow 1$ to $w$
2      do $u_j \leftarrow$ RANDOM-CHOICE(neighbors($u_{j-1}$))
3  return $u_w$

CHOOSE-ID($u, i$)
1  if $i = 0$
2     then $(key, value) \leftarrow$ RANDOM-CHOICE($db(u)$)
3          return $key$
4     else $(x, f) \leftarrow$ RANDOM-CHOICE($fingers(u, i-1)$)
5          return $x$

FINGERS($u, i, r_f$)
1  for $j \leftarrow 1$ to $r_f$
2      do $v_j \leftarrow$ RANDOM-WALK($u$)
3         $x_j \leftarrow ids(v_j, i)$
4  return $\{(x_1, v_1), \ldots, (x_{r_f}, v_{r_f})\}$

SUCCESSORS($u, i, r_s$)
1  for $j \leftarrow 1$ to $r_s$
2      do $v_j \leftarrow$ RANDOM-WALK($u$)
3         $R_j \leftarrow$ SUCCESSORS-SAMPLE($v_j, ids(u, i)$)
4  return $R_1 \cup \cdots \cup R_{r_s}$

SUCCESSORS-SAMPLE($u, x_0$)
1  $\{(key_1, value_1), \ldots, (key_{r_d}, value_{r_d})\} \leftarrow db(u)$
   (sorted so that $x_0 \preceq key_1 \preceq \cdots \preceq key_{r_d} \prec x_0$)
2  return $\{(key_1, value_1), \ldots, (key_t, value_t)\}$ (for small $t$)
```

A.2 Lookup Pseudocode

```
LOOKUP($u$, $key$)
1  $v \leftarrow u$
2  repeat  $value \leftarrow$ TRY($v$, $key$)
3          $v \leftarrow$ RANDOM-WALK($u$)
4     until TRY found valid $value$, or hit retry limit
5  return $value$
```

```
TRY($u$, $key$)
1  $\{(x_1, f_1), \ldots, (x_{r_f}, f_{r_f})\} \leftarrow fingers(u, 0)$
   (sorted so $key \preceq x_1 \preceq \cdots \preceq x_{r_f} \prec key$)
2  $j \leftarrow r_f$
3  repeat  $(f, i) \leftarrow$ CHOOSE-FINGER($u$, $x_j$, $key$)
4          $value \leftarrow$ QUERY($f$, $i$, $key$)
5          $j \leftarrow j - 1$
6     until QUERY found valid $value$, or hit retry limit
7  return $value$
```

```
CHOOSE-FINGER($u$, $x_0$, $key$)
1  for $i \leftarrow 0$ to $\ell - 1$
2       do $F_i \leftarrow \{\ (x, f) \in fingers(u, i) \mid x_0 \preceq x \preceq key\ \}$
3  $i \leftarrow$ RANDOM-CHOICE($\{i \in \{0, \ldots, \ell-1\} \mid F_i$ non-empty$\}$)
4  $(x, f) \leftarrow$ RANDOM-CHOICE($F_i$)
5  return $(f, i)$
```

```
QUERY($u$, $i$, $key$)
1  if $(key, value) \in succ(u, i)$ for some $value$
2     then return $value$
3     else error "not found"
```

Bibliography

[1] Stored wire and electronic communications and transactional records access. U.S. Code: Title 18: Part 1: Chapter 121, 1986.

[2] Skype developer zone. https://developer.skype.com, 2008.

[3] Facebook developers. http://developers.facebook.com, 2010.

[4] jain-sip: Project home. https://jain-sip.dev.java.net, 2010.

[5] Planetlab - an open platform for developing, deploying, and accessing planetary-scale services. http://www.planet-lab.org, 2010.

[6] Sip communicator: the java voip and instant messaging client. https://www.sip-communicator.org, 2010.

[7] Hari Balakrishnan, Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Looking up data in p2p systems. *Communications of the ACM*, 46(2):6, February 2003.

[8] Salman Baset, Gaurav Gupta, and Henning Schulzrinne. Openvoip: An open peer-to-peer voip and im system. *SIGCOMM*, page 1, August 2008.

[9] Salman Baset and Henning Schulzrinne. An analysis of the skype peer-to-peer internet telephony protocol. *IEEE Infocom*, page 12, September 2004.

[10] Ingmar Baumgart. P2pns: A secure distributed name service for p2psip. In *Proceedings of the Sixth Annual IEEE International Conference on Pervasive Computing and Communications*, Hong Kong, China, 2008. IEEE.

[11] David Bryan, Bruce Lowekamp, and Cullen Jennings. Sosimple: A serverless, standards-based p2p sip communication system. *International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*, page 8, June 2005.

[12] David Bryan, Marcia Zangrilli, and Bruce Lowekamp. Challenges of dht design for a public communications system. Technical Report 3, College of William and Mary, Computer Science Department, Williamsburg, VA, June 2006.

[13] Feng Cao, David Bryan, and Bruce Lowekamp. Providing secure services in peer-to-peer communications networks with central security servers. *The International Conference on Internet and Web Applications and Services*, page 6, February 2006.

[14] N. Carlson. How mark zuckerberg hacked into rival connectu in 2004. *The Business Insider*, March 2010.

[15] D. Carvajal. Governments using filters to censor internet, survey finds. *The New York Times*, May 2007.

[16] Global Internet Freedom Consortium. New technologies battle and defeat internet censorship. page 28, September 2007.

[17] T. Dierks and C. Allen. The tls protocol. *RFC*, 2246:1–79, January 1999.

[18] Tomas Isdal, Michael Piatek, Arvind Krishnamurthy, and Thomas Anderson. Privacy-preserving p2p data sharing with oneswarm. *SIGCOMM*, page 20, 2010.

[19] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. Resource location and discovery (reload). IETF Internet Draft, July, January 2008.

[20] Chris Lesniewski-Laas and Frans Kaashoek. Whanau: A sybil-proof distributed hash table. *7th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, page 16, April 2010.

[21] Ninghui Li, John Mitchell, and Derrick Tong. Securing java rmi-based distributed applications. *Computer Security Applications Conference*, 20:262–271, December 2004.

[22] S. Ludwig, J. Beda, P. Saint-Andre, R. McQueen, S. Egan, and J. Hildebrand. Xep-0166: Jingle. *XMPP Draft Standard*, December 2009.

[23] J. Markoff. Surveillance of skype messages found in china. *The New York Times*, October 2008.

[24] David Mazieres, Michael Kaminsky, Frans Kaashoek, and Emmett Witchel. Separating key management from file system security. *Operating Systems Review*, 34(5):124–139, December 1999.

[25] Pekka Nikander, Jukka Ylitalo, and Jorma Wall. Integrating security, mobility, and multi-homing in a hip way. In *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, 2003.

[26] The Associated Press. 10 nations tell google of privacy concern on buzz. *The New York Times*, April 2010.

[27] Swapnil Pundkar. Peer to peer communication over the internet. Thesis project, Indian Institute of Technology, Telecom Italia, Turin, Italy, May-July 2007.

[28] Venugopalan Ramasubramanian and Emin Sirer. The design and implementation of a next generation name service for the internet. *SIGCOMM*, page 12, August 2004.

[29] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. Sip: Session initiation protocol. *RFC*, 3261:268, June 2002.

[30] P. Saint-Andre. Extensible messaging and presence protocol (xmpp): Core. *RFC*, 3920:89, October 2004.

[31] Jan Seedorf. Using cryptographically generated sip-uris to protect the integrity of content in p2p-sip. *VSW*, page 9, 2006.

[32] Kundan Singh and Henning Schulzrinne. Peer-to-peer internet telephony using sip. Technical report, Columbia University, Department of Computer Science, New York, New York, October 2004.

[33] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *SIGCOMM*, page 12, August 2001.

[34] Guido Urdaneta, Guillaume Pierre, and Maarten van Steen. A survey of DHT security techniques. *ACM Computing Surveys*, 2009.

[35] N. Villeneuve. Breaching trust: An analysis of surveillance and security practices on china's tom-skype platform. Wishful Research Result 7, Fanstord University, Computer Science Department, Fanstord, California, October 1988. This is a full TECHREPORT entry.

[36] Michael Walfish, Hari Balakrishnan, and Scott Shenker. Untangling the web from dns. In *Proceedings of the 1st USENIX Symposium on Networked Systems Design and Implementation*, San Francisco, CA, March 2004. NSDI.

[37] S. Wolchok, O. Hofmann, N. Heninger, E. Felten, A. Halderman, C. Rossbach, B. Waters, and E. Witchel. Defeating vanish with low-cost sybil attacks against large dhts. *Proc 17th Network and Distributed System Security Symposium (NDSS)*, page 15, February 2010.

[38] H. Yu, M. Kaminsky, P. Gibbons, and A. Flaxman. Sybilguard: Defending against sybil attacks via social networks. *SIGCOMM*, page 12, September 2006.

[39] P. Zimmermann, A. Johnston, and J. Callas. Zrtp: Media path key agreement for secure rtp. *IETF Internet Draft*, page 108, January 2010.