Pr2-P2PSIP: Privacy Preserving P2P Signaling for VoIP and IM

Ali Fessi, Nathan Evans, Heiko Niedermayer, Ralph Holz
Technische Universität München
Boltzmannstrasse 3
Munich, Germany
{fessi|evans|niedermayer|holz}@net.in.tum.de

ABSTRACT

In the last few years, there has been a good deal of effort put into the research and standardization of P2P-based VoIP signaling, commonly called P2PSIP. However, there has been one important issue which has not been dealt with adequately, privacy. Specifically *i)* location privacy, and *ii)* privacy of social interaction in terms of who is communicating with whom. In this paper, we present Pr^2-$P2PSIP$, a Privacy-Preserving P2PSIP signaling protocol for VoIP and IM. Our contribution is primarily a feasibility study tackling the privacy issues inherent in P2PSIP. We leverage standard security protocols as well as concepts and experiences learned from other anonymization networks such as Tor and I2P where applicable. We present the design and on-going implementation of Pr^2-P2PSIP and provide a threat analysis as well as an analysis of the overhead of adding privacy to P2PSIP networks. Particularly we analyze cryptographic overhead, signaling latency and reliability costs.

Categories and Subject Descriptors

C.2 [**Network Architecture and Design**]: Miscellaneous; K.4.1 [**Public Policy Issues**]: Privacy

General Terms

Privacy, anonymization, Peer-to-Peer(P2P), Session Initiation Protocol (SIP)

Keywords

P2P signaling, P2PSIP, location privacy, social interaction privacy, onion routing, reliability costs

1. INTRODUCTION

The Session Initiation Protocol (SIP) [30] is a protocol standardized by the IETF for setting up multimedia sessions, in particular Voice over IP (VoIP) sessions. It can also be used for Instant Messaging (IM) [29]. There has been a lot of effort in research and standardization in the last few years related to P2PSIP [6]. The concept behind P2PSIP is that the location of a SIP User Agent (UA) (IP address and port number) is published not to a SIP Registrar, but in a Distributed Hash Table (DHT). This data is stored at other peers with peer identifiers (IDs) uncorrelated to the SIP UA. These peers, called replica nodes, reply to queries from any other peer looking for the UA. This makes the UA available for incoming VoIP phone calls and chat messages. However, the SIP UA has no control over knowing which peers have asked for its current location. Curious and malicious peers can perform a lookup for the SIP URI of the UA regularly. The IP addresses of the UA could then be mapped to geographic locations [1]. Using this information, attackers could build location profiles of a user. Even worse, attackers could "crawl" the P2PSIP network and harvest location profiles of all participants. This issue has been left out-of-scope in the IETF P2PSIP working group (WG) [2]. On the other hand, location privacy had been thought of early in the GSM standardization process. Thus, it seems to be necessary to consider this privacy issue in P2PSIP networks as well.

Another privacy threat in P2PSIP is that replica peers can observe that communication is established between two SIP UAs and deduce knowledge about the social interaction of the two users.

In this paper, we tackle the two privacy issues illustrated above; the former, *location privacy* and the latter, *social interaction privacy*, by developing a new protocol which we call *Privacy-Preserving P2PSIP* (*Pr*^{2}*-P2PSIP*). The rest of this paper is organized as follows. In Section 2, we present our on-going work on the design and implementation of Pr^2-P2PSIP. Section 3 provides an evaluation of Pr^2-P2PSIP in terms of threat analyses as well as an analysis of the overhead of adding privacy to P2PSIP networks in terms of cryptographic overhead, signaling latency and reliability costs. Section 4 provides an overview of related work and Section 5 concludes our findings in this paper.

2. DESIGN OF PR^2-P2PSIP

In this section, we introduce Pr^2-P2PSIP.

2.1 Model and Notation

First, we introduce the model and notation used in the rest of the paper.

2.1.1 SIP UAs and Public Identities

The SIP UAs provide the means for users to perform their

IPTComm 2010, 2-3 August, 2010 Munich, Germany
Copyright 2010 ACM ...$10.00.

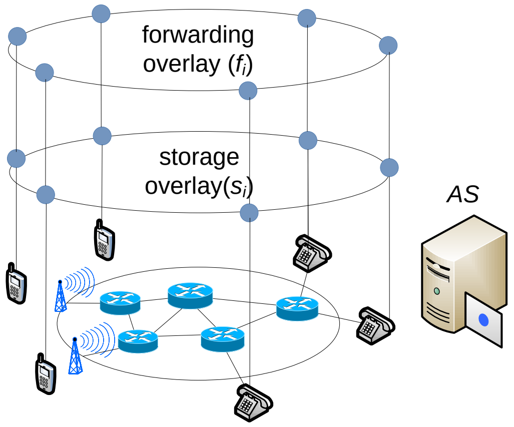

Figure 1: Architecture of Pr^2-P2PSIP

social interactions. They send chat messages and initiate phone conversations on behalf of the users. Let $\mathcal{N}$ be the set of UAs in a P2PSIP network and $n = |\mathcal{N}|$ the number of UAs. In this paper, we use capital letters, e.g., A, B or $A_i, i \in \{1, 2, ...n\}$ to denote interchangeably (unless otherwise explicitly mentioned) a user name, her SIP UA, or her SIP URI.

Note that we use the term "UA" and "peer" interchangeably.

2.1.2 *Authentication Server*

Pr^2-P2PSIP functions with a central authority, which is an *authentication server AS*. The *AS* authenticates a user A using a long-term preshared key, e.g., user password, or a high entropy key stored in the (U)SIM card of the user's smart phone. After successful authentication, the *AS* provides the UA with a certificate that binds the user's public key $+K_A$ to her public identity A. The *AS* is indispensable for Pr^2-P2PSIP as it provides verifiable identities at the application layer. This enables UAs to mutually authenticate each other and establish secure channels for encryption and integrity-protection at the application layer (SIP signaling and multimedia streams). The *AS* provides verifiable identities at the overlay layers as well (Pr^2-P2PSIP includes two different overlays, explained in Section 2.1.3) in order to prevent attacks on the overlays, e.g., Sybil and eclipse attacks. Another attack that would be possible without a central authority would be the so-called *chosen-location attack* where malicious peers choose a convenient peer ID where they could, eclipse (hide) other peers, or eclipse the content they would be responsible for. In the context of privacy, chosen-location attacks would allow malicious peers to choose a strategically "good" position where they could monitor the activities of certain other peers.

2.1.3 *Storage and Forwarding Overlays*

In addition to its public identity, a UA A_i has two *pseudonyms* f_i and s_i which it uses for participating in two different overlays as sketched in Figure 1. $s_i, i = 1, \ldots, n$ is the *storage* overlay. $f_i, i = 1, \ldots, n$ is the *forwarding* overlay.

Storage.

Storage is the common service that DHT's provide. The DHT stores information required to contact other UAs for sending them application layer signaling messages. However, the information stored in the Pr^2-P2PSIP DHT differs from P2PSIP. Specifically, it does not reveal the actual location of UAs. The content of this information is explained in Sections 2.2.2 and 2.3.2.

Table 1: Notation

$+K_e$	Public key of an entity e
$-K_e$	Private key of an entity e
$K_{a,b}$	Shared secret key between entities a and b
$\{m\}_{K_{a,b}}$	Message m encrypted and integrity-protected with the symmetric key $K_{a,b}$ (See Section 2.4).
$\{m\}_{+K_e}$	Message m encrypted with the public key of entity e.
$l(e,t)$	Location (IP address and port number) of the entity e at a certain point of time t
$L(A,t)$	Data stored in P2P network required to reach UA A at a certain point of time t

Forwarding.

Forwarding is an additional function that peers need to perform in Pr^2-P2PSIP. It differs from typical forwarding in DHT algorithms with recursive routing, e.g., Chord or Pastry, given that these DHT algorithms were not designed with privacy in mind. Message forwarding in Pr^2-P2PSIP is explained in 2.2.1.

Overlay Algorithm.

We currently use Kademlia [20] as our DHT overlay algorithm. However, Pr^2-P2PSIP could be used with other DHTs. We do not claim that the choice of the overlay algorithm is orthogonal to the impact of Pr^2-P2PSIP on user privacy. Thus, this design decision requires further investigation in future work. For this paper, we use the Kademlia RPCs FIND_NODE, FIND_VALUE, PING and STORE in the storage overlay. Since the forwarding overlay is used only for finding other peers (i.e., no data stored in the DHT, see Section 2.2.1 for details), the forwarding overlay makes use only of the FIND_NODE and PING RPCs.

Pseudonyms in the Storage and Forwarding Overlays.

The pseudonyms f_i and s_i are temporal identities which are unlinkable to the UA's public identity A_i (we use non-capital letters to denote pseudonyms). Pseudonyms f_i and s_i belong to an identifier space $\mathcal{K}$, e.g. $\mathcal{K} = \{0, \ldots, 2^{160} - 1\}$. Each pseudonym is linked to a public key as well: $(f_i, +K_{f_i})$, $(s_i, +K_{s_i})$. As such, a UA uses different public/private key pairs for different purposes.

By "UA A_i", we mean the UA with public identity A_i while "UA f_i" or "UA s_i" is the UA with pseudonym f_i or s_i respectively. Table 1 provides additional notations used throughout this paper.

2.1.4 *Threat Model*

Given a UA $A \in \mathcal{N}$, we assume that an attacker M wants to collect as much information as possible about A, in particular:

1. its current locator $l(A, t)$

2. its location profile: a history of $l(A, t)$

3. a social interaction profile: a history of social interactions $A \rightarrow B$ or $B \rightarrow A$ for any $B \in \mathcal{N}$.

Note that man-in-the-middle, eavesdropping and message forgery attacks on the application data (chat messages and phone conversations) can be successfully countered (unless the AS turns malicious) using the UA's certificates provided by the AS. Note also that the AS guarantees that each UA receives a single pseudonym f_i and a single pseudonym s_i, so Sybil attacks can be excluded and eclipse attacks are difficult (since the overlay routing algorithm provides multiple disjoint paths between two arbitrary peers).

We consider the following attackers in Pr^2-P2PSIP:

1. a single malicious UA participating in the Pr^2-P2PSIP network: $M \in N$. In this case, we assume every UA operates on its own. Different malicious UAs do not exchange information for the sake of breaking other users' privacy. Thus, each UA can observe only the messages it sends and it receives. Additionally, if it forwards a message from one peer to another, it can decrypt only the messages (or message parts) for which it has the appropriate key.

2. a *partial* observer in the network underlay observing that communication is taking place between different IP addresses. The attacker may be able to observe some traffic and deduce some conclusions about the location or social interaction of some UAs.

2.2 Protocol Overview

In this section we describe how Pr^2-P2PSIP handles data storage and message forwarding. Storage and forwarding in the Pr^2-P2PSIP network differ from a "regular" P2PSIP network, because UAs seek to keep their location and social interaction private.

2.2.1 Message Forwarding

An application layer message (e.g., SIP MESSAGE for IM or SIP INVITE for establishing a phone call) from a UA A to a UA B is sent via intermediate forwarding peers using so-called *onion routing* [15]. In onion routing, the sender of a message m chooses intermediate forwarding peers which route the message to B on behalf of A. A orders these peers in series and encrypts m several times recursively. One layer of encryption is removed at each of the forwarding peers, so that the final peer in the tunnel has the original unencrypted message.

In Pr^2-P2PSIP, peers establish *inbound tunnels* and *outbound tunnels* (see Figure 2). The choice of tunnel length has some effects on privacy which are discussed in detail in Section 3. For illustration purposes, we consider a tunnel length of three hops throughout Section 2.

A UA A uses its pseudonym ($f_{O_0} = f_{I_0}$ in Figure 2) to communicate with the first hop of each tunnel. For outbound tunnels, A (sending application layer messages) generates symmetric keys for protected communication (i.e. encrypted and integrity protected) with each of the outbound forwarding peers (f_{O_1}, f_{O_2} and f_{O_3}). For inbound tunnels, A (receiving application layer messages) generates symmetric keys for protected communication with each of the inbound forwarding peers f_{I_1}, f_{I_2} and f_{I_3}. In both cases, A uses the public keys of the forwarding peers to distribute the required symmetric keys which will be used during the tunnel lifetime. Additionally, the forwarding peers establish TLS sessions for hop-by-hop security. Figure 3 sketches the resulting encryption and integrity-protection layers. The layered encryption ensures that the message looks different for each hop.

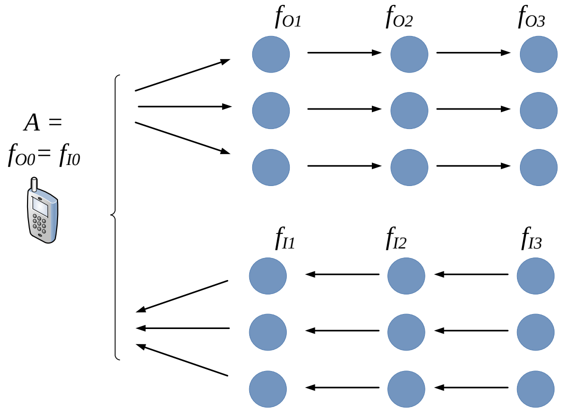

Figure 2: Inbound and outbound tunnels of sender/receiver A

While the end-to-middle symmetric keys are valid only for the tunnel lifetime, a hop-by-hop TLS session may be multiplexed for several inbound and outbound tunnels serving several sender/receiver peers and can be long-lasting. This design decision is borrowed from Tor and should make traffic analysis more difficult. Unlike Tor where all peers are connected in a full mesh and establish TLS tunnels to each other, Pr^2-P2PSIP TLS tunnels are established on demand, since otherwise Pr^2-P2PSIP could not scale to more than few thousand peers.

Forwarding Pool.

To discover forwarding peers, peers query the forwarding overlay. Additionally, each peer keeps a local pool of the forwarding peers it has learned about, and which it can ask to be a part of its tunnels. This pool should be kept up-to-date, so a peer can refresh its inbound or outbound tunnels.

The peer will occasionally learn about other forwarding peers as a side effect of overlay maintenance. However, it is crucial for the privacy goals of Pr^2-P2PSIP to not rely solely on overlay maintenance for re-filling its forwarding pool and not to simply choose peers from its overlay routing table. Instead, a UA A should perform node lookups (a FIND_NODE RPC in Kademlia) for *random* identifiers in the forwarding overlay when it needs to update its forwarding pool, in order to prohibit an attacker M from being able to force A to select her (M) as a forwarding peer in her tunnels (i.e., path selection attack; see Section 3.1).

2.2.2 Contact Data Storage

The contact data of all UAs are stored in a DHT. For each UA A there exists a value stored in the DHT with the contact data of A under the key $h(A)$. The contact data is a tuple $(+K_A, L(A, t))$. $L(A, t)$ does not reveal any information about A's real location $l(A, t)$. Instead, $L(A, t)$ includes information about the entry points of A's inbound tunnels (i.e., the forwarding peers furthest from A in her

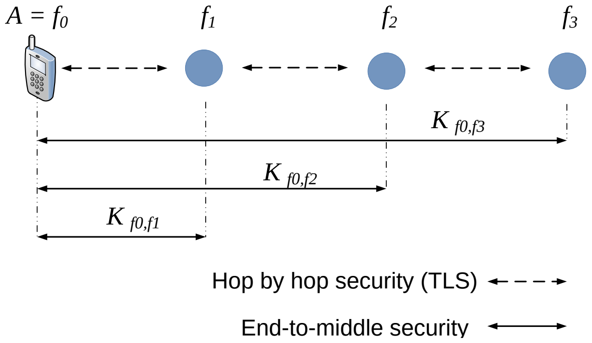

Figure 3: End-to-middle and hop-by-hop encryption and data-integrity layers in Pr^2-P2PSIP

inbound tunnels), which will forward incoming messages towards A. Details on the structure of $L(A, t)$ are provided in Section 2.3.2.

2.3 Protocol Operations

In this section we provide more low level details on the protocol operations of Pr^2-P2PSIP.

2.3.1 *Tunnel Setup*

Up to this point we have differentiated between inbound and outbound tunnels. However, the procedure for setting up both kinds of tunnels and the per-hop state required for them is the same. A forwarding peer can be unaware of the type of tunnel it is participating in. This reduces the complexity of Pr^2-P2PSIP.

In fact, in both cases communication takes place in both directions, for instance to acknowledge tunnel setup and to tunnel RPC responses backwards to the initiator of a RPC (this is the case for publishing data in the DHT; see Section 2.3.2; and retrieving data from the DHT; see Section 2.3.3).

Forwarding peers need to store state information that is required to process incoming and outgoing messages for each tunnel. Let A be the UA which initiates the tunnel setup for sending or receiving application layer messages. Let f_1, f_2 and f_3 be the forwarding peers chosen by A to build the tunnel (as in Figure 3). A uses its pseudonym f_0 to communicate with the first hop in the tunnel, f_1. The state stored at each forwarding peer $f_i, i = 1, 2, 3$, called the *tunnel binding* in Pr^2-P2PSIP, is a tuple which consists of the following data:

- *tunnel ID*: a tunnel ID α used for multiplexing between different tunnels,
- *successor* and *predecessor*: the pseudonyms, public keys and locations of the successor and the predecessor peers in the tunnel: $(f_{i+1}, +K_{f_{i+1}}, l(f_{i+1}, t))$ and $(f_{i-1}, +K_{f_{i-1}}, l(f_{i-1}, t))$,
- *end-to-middle symmetric key*: K_{f_0,f_i}.

This data is distributed by A during the tunnel setup. Furthermore, f_{i+1} and f_{i-1} are used at each forwarding peer locally to determine whether it has already established TLS sessions with the successor and predecessor peers.

The data for the tunnel binding is sent by A onion-encrypted along the tunnel. For each node $f_i, i = 1, 2, 3$, A sends (indirectly) a message:

$$\begin{aligned} m_i = \quad & (\alpha, \\ & f_{i+1}, +K_{f_{i+1}}, l(f_{i+1}, t), \\ & f_{i-1}, +K_{f_{i-1}}, l(f_{i-1}, t), \\ & K_{f_0,f_i}) \end{aligned} \tag{1}$$

For f_3, the information about the successor is marked with *null* values:

$$\begin{aligned} m_3 = \quad & (\alpha, \\ & null, null, null, \\ & f_2, +K_{f_2}, l(f_2, t), \\ & K_{f_0,f_3}) \end{aligned} \tag{2}$$

Of course, this has the consequence that f_3 can deduce that it is the last hop in the tunnel. The impact of this information available to f_3 will be discussed in Section 3.1. The message flow for setting up the tunnel initiated by A looks as follows.

$$\begin{aligned} f_0 \leftrightarrow f_1 \quad &: \quad TLS \quad handshake \\ f_0 \rightarrow f_1 \quad &: \quad \{m_1, \{m_2, \{m_3\}_{+K_{f_3}}\}_{+K_{f_2}}\}_{+K_{f_1}} \\ f_1 \leftrightarrow f_2 \quad &: \quad TLS \quad handshake \\ f_1 \rightarrow f_2 \quad &: \quad \{m_2, \{m_3\}_{+K_{f_3}}\}_{+K_{f_2}} \\ f_2 \leftrightarrow f_3 \quad &: \quad TLS \quad handshake \\ f_2 \rightarrow f_3 \quad &: \quad \{m_3\}_{+K_{f_3}} \end{aligned} \tag{3}$$

The TLS handshakes take place only if two successive forwarding peers have not yet established a TLS session. After tunnel setup, A (i.e., f_0) can exchange messages with f_3 without revealing her location $l(A, t)$ or her identity (neither the public identity A, nor her pseudonym f_0). f_3 knows only the information about f_2.

A message m from A to f_3 is forwarded as follows:

$$\begin{aligned} f_0 \rightarrow f_1 \quad &: \quad \{\alpha, \{\alpha, \{\alpha, m\}_{K_{f_0,f_3}}\}_{K_{f_0,f_2}}\}_{K_{f_0,f_1}} \\ f_1 \rightarrow f_2 \quad &: \quad \{\alpha, \{\alpha, m\}_{K_{f_0,f_3}}\}_{K_{f_0,f_2}} \\ f_2 \rightarrow f_3 \quad &: \quad \{\alpha, m\}_{K_{f_0,f_3}} \end{aligned} \tag{4}$$

while a message from f_3 to A is forwarded as follows:

$$\begin{aligned} f_3 \rightarrow f_2 \quad &: \quad \alpha, \{m\}_{K_{f_0,f_3}} \\ f_2 \rightarrow f_1 \quad &: \quad \alpha, \{\{m\}_{K_{f_0,f_3}}\}_{K_{f_0,f_2}} \\ f_1 \rightarrow f_2 \quad &: \quad \alpha, \{\{\{m\}_{K_{f_0,f_3}}\}_{K_{f_0,f_2}}\}_{K_{f_0,f_1}} \end{aligned} \tag{5}$$

The tunnel setup (message flow (3)) is acknowledged by the last forwarding peer f_3. Thus, the acknowledgement message is the first message sent from f_3 to A via f_2 and f_1. Note that the acknowledgement of the tunnel setup by f_3 is crucial for the reliability of Pr^2-P2PSIP. This will be discussed in detail in Section 3.2.

2.3.2 *Publishing UA Contact Data*

Publishing the contact data of a UA in the DHT makes use of outbound tunnels and the Kademlia STORE RPC. A UA A publishes its application layer public key $(+K_A)$ as well as the pseudonyms, the public keys and the locations of the entry points of its inbound tunnels. For example, assume A has three parallel inbound tunnels. Then, the value stored

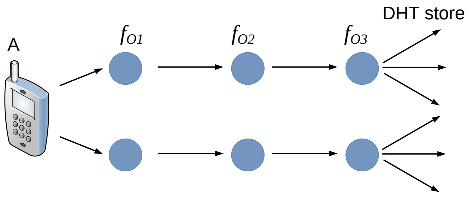

Figure 4: Publishing UA contact data in the DHT

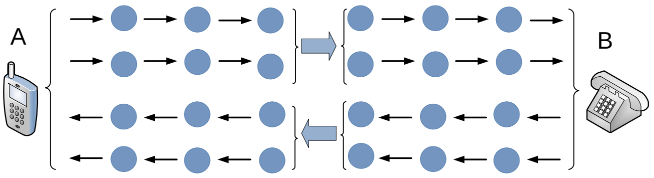

Figure 5: Bidirectional signaling in Pr^2-P2PSIP

in the DHT under the key $h(A)$ is a tuple $(+K_A, L(A,t))$ where

$$\begin{aligned} L(A,t) &= (f_{I_3}, +K_{f_{I_3}}, l(f_{I_3},t), \alpha), \\ &\quad (f'_{I_3}, +K_{f'_{I_3}}, l(f'_{I_3},t), \beta), \\ &\quad (f''_{I_3}, +K_{f''_{I_3}}, l(f''_{I_3},t), \gamma) \end{aligned} \tag{6}$$

where f_{I_3}, f'_{I_3} and f''_{I_3} are the entry points of the different inbound tunnels; and α, β and γ the respective tunnel IDs. The STORE RPC request is sent from A to f_{O_3} using message flow (4). This is depicted in Figure 4. It is crucial that the STORE RPC responses received by f_{O_3} are forwarded back to A (using message flow (5)). The reason for this is that A can not be sure that all peers in the outbound tunnel (f_{O_1}, f_{O_2} and f_{O_3}) are still online since the tunnel has been established or refreshed. If A does not receive a response to her STORE request from f_{O_3}, she needs to re-initiate the RPC using another outbound tunnel. The time interval between two successive RPC requests is a trade off between latency and signaling overhead. In the extreme case, A could send STORE RPCs simultaneously along several outbound tunnels. However, this parallelism may produce a large unnecessary signaling overhead depending on the stability of the network (and thus, the stability of the outbound tunnels). As a trade off, we use an aggressive timeout of $1s$ before the next outbound tunnel is invoked.

2.3.3 *Retrieving Contact Data*

Looking up data in the DHT is quite similar to publishing data in the DHT except the Kademlia RPC used is FIND_VALUE. A uses one of her outbound tunnels and asks the last peer in the tunnel to lookup the data on behalf of her. The same procedure with timeouts is performed if no response is received from an outbound tunnel.

Using the same procedure for publishing and retrieving data in/from the DHT reduces the complexity of the protocol.

2.3.4 *Bidirectional Signaling*

Once A has found the entry points of the inbound tunnels of B, she can use her outbound tunnels to send application layer messages to B. A may include her real location $l(A,t)$ (encrypted with $+K_B$) in the first signaling message to B or $L(A,t)$ if she does not want to reveal her location to B. The same holds for the response of B to A. Every SIP message is acknowledged end-to-end, i.e., if B receives a message from A through one of his inbound tunnels, he sends an acknowledgement through one of his outbound tunnels.

The same procedure with timeouts applies here as well: if A sends a SIP message to B and the acknowledgement does not reach A within $1s$ another end-to-end path, i.e., another combination of an outbound tunnel of A and an inbound tunnel of B is used. At this point, it is worth it mentioning that Pr^2-P2PSIP is designed with signaling in mind and is not optimized for real-time communication. The main problem with real-time communication is the accumulated one-way-delay in both directions between A and B, given that there are four to six hops between A and B (depending on the tunnel length).

2.4 Cryptographic Primitives

In this section, we provide implementation details on the cryptographic primitives used in Pr^2-P2PSIP.

Symmetric Cryptography.

As mentioned in Table 1, $\{m\}_{K_{a,b}}$ is a message m encrypted and integrity-protected with the shared key $K_{a,b}$. This is used to provide end-to-middle security in the inbound and outbound tunnels (see Figure 3). However, it is well known that different keys should be used for different purposes and for each direction [13]. Thus, four symmetric keys are derived from $K_{a,b}$ on both sides using a cryptographic key expansion function. These keys are derived at the tunnel setup and used during the tunnel lifetime.

Public Key Cryptography.

Given that Pr^2-P2PSIP makes extensive use of public key encryption, in particular for inbound and outbound tunnel setup, it is crucial to optimize the use of the public key cryptographic primitives. We use two solutions for this purpose:

- a message m from a to b encrypted with the public key $+K_b$ is actually encrypted with a temporary symmetric key $K_{a,b}$ generated by a. Then, $\{m\}_{K_{a,b}}$ is sent together with the temporary key $K_{a,b}$ encrypted with $+K_b$. Thus, $\{m\}_{+K_b}$ is actually implemented as $(\{K_{a,b}\}_{+K_b}, \{m\}_{K_{a,b}})$.

- an important design decision in Pr^2-P2PSIP is to use Elliptic Curve Cryptography (ECC) [31] instead of RSA for public key encryption. The reason is the convenient key length without necessarily sacrificing performance. An ECC key length of 194 bits provides comparable entropy to a 2054 bit RSA key[1].

The impact of the design decisions on the cryptographic primitives are further discussed in Section 3.3.

2.4.1 *Pitfalls*

In this section, we explain a few details that need to be taken into account when implementing Pr^2-P2PSIP. These details were skipped in the previous sections for the sake of simplicity.

[1] The choice of the private key for RSA is limited by the choice of prime numbers, while any random number can be used as a private key for ECC.

Outbound tunnels used by A for publishing $L(A,t)$ should not be used for other purposes, e.g., retrieving contact data of another UA B. The last hop in the outbound tunnel of A, f_{O_3} sees only the hash value of A when the data is stored in the DHT. However, if f_{O_3} has a list of user names, it can determine whether A is one of them. If the same outbound tunnel is used for retrieving the contact data of B, f_{O_3} can deduce that A is about to send a SIP message to B. Thus, the social interaction privacy of A would be broken.

In the description of the tunnel setup in Section 2.3.1, the tunnel ID α remains constant along the tunnel. However, this raises a privacy threat especially for inbound tunnels. Intermediate hops (f_{I_1}, f_{I_2} and f_{I_3}) are all aware of the tunnel ID α published in the contact data of A in the DHT: $L(A,t)$. Thus, by crawling the DHT, f_{I_1} can discover which UA A has published its contact data $L(A,t)$ with α as tunnel ID, and can deduce the public identity of A. Since f_{I_1} has direct IP communication with A, the location privacy of A is broken. In order to defeat this attack, the tunnel ID has to be changed at each hop. Thus, each forwarding peer has two different tunnel IDs, one shared with the predecessor and another one shared with the successor. Since A needs to know the final tunnel ID at f_{I_3} in order to publish its contact data $L(A,t)$ in the DHT, f_{I_3} informs A about the tunnel ID to be published when it confirms the tunnel setup to A. Since f_{I_3} and A use end-to-middle encryption to secure their communication, f_{I_1} and f_{I_2} can not deduce which tunnel ID is published in the DHT.

3. EVALUATING PR^2-P2PSIP

3.1 Threat Analysis

In this section, we evaluate whether Pr^2-P2PSIP fulfills its goals, i.e., whether it can thwart attacks on location privacy and social interaction privacy. Additionally, based on an extensive threat analysis, we deduce appropriate recommendations for the tunnel length.

The threat analysis of Pr^2-P2PSIP benefits from attacks on anonymization networks that have been described in the literature. Therefore, we provide an overview of those attacks that are relevant to Pr^2-P2PSIP first. We then evaluate whether these attacks can be applied to Pr^2-P2PSIP and if Pr^2-P2PSIP introduces new attack vectors.

3.1.1 Attacks on Anonymization Networks

Attacks on anonymization networks can be classified into *passive* and *active* attacks. Passive attacks are attacks where the attacker monitors communication between other peers. For this purpose, the attacker may try to become part of one of the victims tunnels. However, in passive attacks, attackers do not alter the data they observe or forward. In contrast to passive attacks, active attacks involve a participant actively altering or injecting data in the network. Nevertheless, an attacker may combine passive and active attacks in order to reach his malicious goals. As with all privacy preserving networks, a trade off exists between usability and security.

Traffic Analysis.

Traffic analysis is a general term referring to monitoring data as it passes through a network to glean useful information. In an onion routing network over the Internet this typically means monitoring underlying network communications or data handled by a participant in the network overlay. A subset of traffic analysis called *timing analysis* measures *when* data enters or exits the network or nodes in the network. All of the attacks described herein utilize some form of traffic analysis. As discussed in [3, 33] an attacker that is able to observe both ends of a tunnel may be able to correlate that two peers (identified by IP addresses) are communicating by analyzing inbound and outbound packet counts between every two peers. This attack is depicted in Figure 6. However, the attacker can not be sure that the two peers are communicating, since they could simply be forwarding data for other peers.

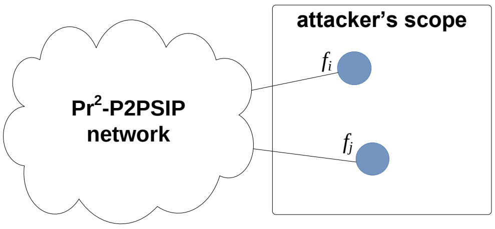

Figure 6: Passive attacks on Pr^2-P2PSIP

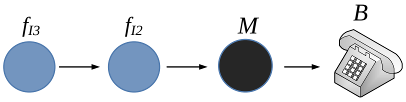

Figure 7: Path selection attacks on Pr^2-P2PSIP

Path Selection Attacks.

Another type of passive attack is the path selection attack [5]. The attacker forces particular peers to be chosen for a tunnel, preferably controlled by the attacker. Since we assume peers do not collude in Pr^2-P2PSIP, this attack is useful only if the attacker is on an end of the tunnel directly connected to the victim as in Figure 7. Given that peers choose forwarding peers using *random* identifiers in the forwarding overlay, the probability of a successful path selection attack when a peer builds its inbound tunnels is inversely proportional to the size of the network. However, given that a peer occasionally has to change the peers in its inbound tunnel, the probability of a successful path selection attack grows over time.

Most other passive attacks [3, 9] require a *global* passive adversary, outside of the threat model for our work.

Congestion Attacks.

The congestion [23] or circuit clogging [21] attack combines typical traffic and timing analysis with an active denial or reduction of service attack. The basic layout of this attack is depicted in Figure 8. In this type of attack, a malicious peer initiates a “legitimate” communication with the victim. Using this communication, she alternates between periods of sending data and being silent on the tunnel. She concurrently builds tunnels between all (or some subset of) possible other peers in the network and sends probe traffic down each. If she can correlate the sending periods on the legitimate tunnel with traffic on the probe tunnels she has discovered that some peers on the probe tunnel are also part

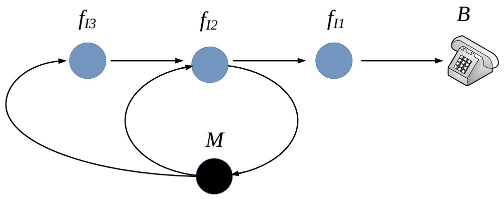

Figure 8: Congestion attacks on Pr^2-P2PSIP

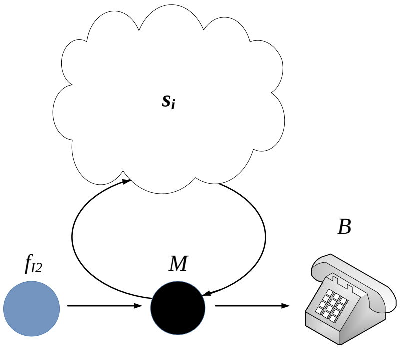

Figure 9: Attacks on two-hop inbound tunnels

of the legitimate one. This method works if forwarding peers have to split resources equally between their tunnels; utilizing one tunnel therefore alters the latency properties of the other tunnels. By building repeated probe tunnels through different sets of possible peers she can eventually determine exactly which peers are being used. Provided that the peers on the legitimate tunnel are rotated over time (as is the case in Pr^2-P2PSIP) and the victim will be the only peer which will be always part of the tunnels, the attacker could discover the actual IP address of the victim.

3.1.2 Attacks on Pr^2-P2PSIP

In this section, we provide a security threat analysis of Pr^2-P2PSIP on inbound and outbound tunnels for different tunnel lengths.

Attacks on One-hop Inbound Tunnels.

Using one-hop inbound tunnels, the only inbound forwarding peer and potentially malicious peer $M = f_{I_1}$ is directly connected to the victim B. The contact data of B published in the DHT points to f_{I_1}. Thus, by crawling the DHT (i.e. the storage overlay), f_{I_1} can find out which UAs have published their contact information with f_{I1} as a tunnel entry point. f_{I1} might be the tunnel entry point for several peers, let' say B, B' and B''. After collecting the data $\{L(B,t), L(B',t), L(B'',t)\}$ from the DHT, f_{I1} can correlate the tunnel IDs in $L(B,t)$, $L(B',t)$ and $L(B'',t)$ with the tunnel bindings it has previously setup and can unambiguously deduce the location of B, B' and B''.

Attacks on Two-hop Inbound Tunnels.

Using two-hop inbound tunnels, as shown in Figure 9, a similar attack remains possible. A malicious peer M can trivially recognize from communication with the successor and the predecessor in the tunnel that she is not the entry point of the tunnel. Thus, M can deduce its position in the tunnel and that its predecessor is the initiator of the tunnel (B) and its successor is the entry point of the tunnel (f_{I_2} in Figure 9). By crawling the content of the DHT, M can find out which UAs have published their contact information with f_{I_2} as a tunnel entry point, again let' say B, B' and B''. The difference to the one-hop case is that M can not necessarily identify which one of these peers is the initiator of the tunnel she is part of. This is because the tunnel ID is not constant along the tunnel. Nevertheless, M could significantly reduce the number of possible public identity of the tunnel initiator, potentially to one. This would lead to an unambiguous link between the public identity of B and his current location $l(B,t)$.

Depending on the size of the network, B may have changed its inbound tunnels while M is still crawling the DHT, and the data M is looking for in the DHT may become unavailable. However, we can not rely on this assumption, if M has sufficient resources.

One possible approach to reduce the probability of this attack could be the concept of *entry guards* [25], which were suggested for thwarting attacks on discovering the origin of *hidden services* in Tor. These attacks are based on path selection attacks. The concept of entry guards is as follows; instead of choosing uniformly at random from the set of all peers for the crucial hop (the nearest to the hidden server in Tor, the nearest to the UA in the inbound tunnel in Pr^2-P2PSIP, i.e., f_{I_1})), a small set of peers are chosen initially and one of these is always utilized in that position. Choosing forwarding peers uniformly at random gives a patient attacker the chance to be chosen as the crucial hop with a high probability if B rotates his tunnels regularly, whereas the probability of choosing the attacker with "final guardians" is only g/n where g is the total number of guardian nodes used (and n the overall number of peers as mentioned in Section 2.1).

Nonetheless, since malicious peers have the chance here to discover the public identity of B and its location with an effort estimated by $O(n)$ (crawling the DHT), we consider the attack on one-hop and two-hop inbound tunnels as a real threat to Pr^2-P2PSIP.

Attacks on Three-hop Inbound Tunnels.

Using three-hop inbound tunnels, a possible attack scenario is a variant of the circuit clogging attack, where the participants of a tunnel can be deduced. In this scenario the attacker M initiates a communication with the victim B (Figure 8). M wants to discover the IP address of B. To do so, she actively builds tunnels through many peers which she uses to send a steady stream of data to herself. She then sends a certain pattern to B (for example, via chat), which can be detected on the tunnels that she is monitoring because of interference [21, 23, 28]. Since M may not necessarily obey to the agreed inbound tunnel length in the network, she could conceivably connect to every peer with a one hop tunnel back to herself and send the pattern to B (via his legitimate inbound tunnel). If the pattern is detected, this reveals either B or a part of his tunnel. By repeating the same procedure for each of B's multiple inbound tunnels, M can eliminate B's tunneling peers, because B will be the only peer present on *each* of the inbound tunnels used.

This attack becomes more difficult as the number of peers in the network increases, because the attacker needs to monitor them all for the pattern she is sending. False positives or false negatives may occur due to other traffic in the network at the same time as the attacker's probe or pattern traffic. The attack may also take a prohibitively long amount of time to mount; if the attacker cannot monitor all nodes in the network at once, she will need to perform this attack by monitoring only some subset of the network at a time.

General Attacks on Outbound Tunnels.

No matter how long the outbound tunnel is, the last hop in the tunnel (furthest from A) which is used for publishing the contact data of A in the DHT should not be used for other purposes as mentioned in Section 2.4.1. Otherwise, the social interaction privacy of A would be broken.

Attacks on One-hop Outbound Tunnels.

If the outbound tunnel of a UA A consists of one hop only, when A publishes her contact data in the DHT, the outbound forwarding peer f_{O_1} receives the STORE RPC from A directly, and thus, can trivially discover the public identity of A and correlate it with her IP address. This would break the location privacy of A.

Attacks on Two-hop Outbound Tunnels.

Attacks on two-hop outbound tunnels become more difficult. The last peer in the outbound tunnel f_{O_2} may misuse the property of Pr^2-P2PSIP that communication in both inbound and outbound tunnels takes place in both directions, and send certain traffic patterns to f_{O_1} which are forwarded to A and thus may be the basis for a congestion attack.

Conclusions.

Given the threat analysis above, we conclude that:

- Passive attacks are of limited use because while they may reveal that two peers are participating in the network and connected, this does not indicate whether the peers are forwarding data for other peers or actually communicating.

- Path selection attacks require that the attacker be chosen as the victim nodes final inbound hop. The probability of the success of such an attack is inversely proportional to the size of the network. Though it increases over the time by changing the tunnel. Unless entry guards are chosen as crucial hop.

- Congestion attacks may be feasible, but at high cost, take a long time and are susceptible to false positives and false negatives.

- A tunnel length of *three hops* for *inbound tunnels* and *two hops* for *outbound tunnels* provide location and social interaction privacy at a high and satisfactory degree.

3.2 Reliability Cost Analysis

In this section, we provide a model of Pr^2-P2PSIP based on reliability theory [26]. This model will then be used for estimating the overhead generated by adding privacy to P2PSIP. First, we start with some basic knowledge in reliability theory from [26] which is required to understand the model.

3.2.1 Reliability Theory

Reliability theory provides tools for estimating the reliability of a whole system by estimating the reliability of the single units/components of the system. Let T be the *time to failure* of a unit, i.e., the time elapsed between when the unit is put into operation until it fails for the first time. T can be assumed to be continuously distributed with a density function $f(t)$ and distribution function:

$$F(t) = Pr(T \leq t) = \int_0^t f(u)\,du \qquad (7)$$

The reliability $R(t)$ is the probability that the unit will be still operating at time t:

$$R(t) = 1 - F(t) = Pr(T \geq t) \qquad (8)$$

A structure of units is *series* if the operation of the structure depends on the operation of all units in this structure. A *parallel* structure is a structure which operation requires at least one of the units operating.

Let a structure consisting of k units with independent failures[2] and equal reliabilities $R_i(t) = R(t)$ for all units $i = 1, \ldots, k$. If the structure is series, the reliability of the structure is

$$R_\wedge(t) = R_1(t)R_2(t)\ldots R_k(t) = R^k(t) \qquad (9)$$

If the structure is parallel, the reliability of the structure is

$$\begin{aligned} R_\vee(t) &= 1 - (1 - R_1(t))(1 - R_2(t))\ldots(1 - R_k(t)) \\ &= 1 - (1 - R(t))^k \qquad (10) \end{aligned}$$

3.2.2 Modeling Pr^2-P2PSIP Networks with Reliability Theory

A Pr^2-P2PSIP (or P2PSIP) network is a system which consists of multiple units, which are the peers. The time to failure of a peer is the time interval between the time when the peer goes online until it leaves the network, i.e., T is the peer lifetime. Different studies of P2P networks for file sharing, in particular KAD [35] and for VoIP, in particular Skype [16] have shown that the peer lifetime is heavy-tailed distributed. Since it is difficult to estimate appropriate parameters for a P2PSIP network, we focus on a generic analytical model first. Note that Skype is not necessarily a good representative since Skype clients are mainly installed on PCs/laptops. Skype shows a high number of peers during working days and middays, while peers in a P2PSIP network could be running, e.g., on some fixed hardphones which are permanently online, or on mobile smart phones, which may change their IP addresses more frequently than laptops. Nevertheless, Skype is the most similar application to P2PSIP and Pr^2-P2PSIP and the study in [16] will help us to interpret the results of our reliability costs analysis as shown below.

Reliability Model of Pr^2-P2PSIP.

A UA B refreshes its contact data in the DHT as well as its inbound tunnels periodically with a refreshing period e.g., $\tau = 20mn$, in order to make sure it remains reachable in the Pr^2-P2PSIP network with high probability. This high probability is a target reliability, e.g., $\bar{R} = 1 - 10^{-5}$.

When B performs a refresh operation at $t = k\tau, k \in \mathbb{N}$, it receives acknowledgement messages for both the storage

[2] which is a dominant assumption in reliability theory

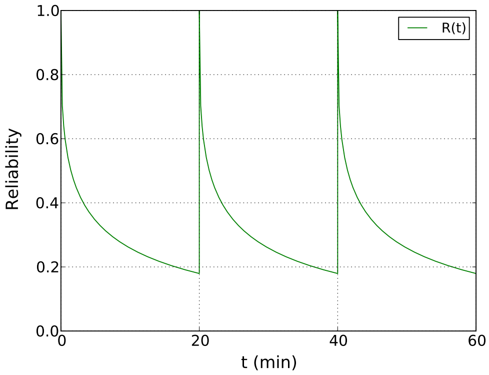

Figure 10: Example reliability of a single storage unit s_i or inbound forwarding unit f_i with periodic refreshes. $\tau = 20mn$.

and the tunnel refresh/setup (as described in Sections 2.3.1 and 2.3.2). Thus, we assume the probability that a peer/unit, either involved in the storage of the contacts of B or involved in one of the inbound tunnels for B, is online at $t = k\tau$ is 1. Then, this probability decreases over the time to the minimum value. An example of this behavior is shown in Figure 10. We denote by μ the minimum reliability of a peer at the end of each refreshing period.

$$\mu = \liminf_{t \to (k+1)\tau} R(t) \quad k \in \mathbb{N} \tag{11}$$

μ could be estimated autonomously by B through measurements. It is the probability that if another UA is observed online at t, the UA will remain online until $(t + \tau)$. μ can be considered as a metric for the churn in the network. If the measured value for μ is too low, then the UA may have to decrease τ, and thus increasing μ.

Furthermore, the following assumptions are required for our reliability analysis:

- We assume that all peers are cooperative, i.e., as long as a peer is online, it will perform requests from other peers to create tunnels, forward messages and store data.
- We assume that peers/UAs leave and join the network independently. A UA which leaves the network deletes all contact data and tunnel bindings of other peers.
- We assume a DHT model like in KAD [35] where peers which publish data are responsible for refreshing this data themselves, i.e., replica nodes do not re-publish data among each other, in particular when some of them leave the network, or new nodes close to the key of the data enter the network.
- We assume that routing in the DHT always succeeds. In particular if A is looking for the contact data of B and there is at least one replica node s_i storing this data, then A will be able to reach s_i and find the contact data of B.

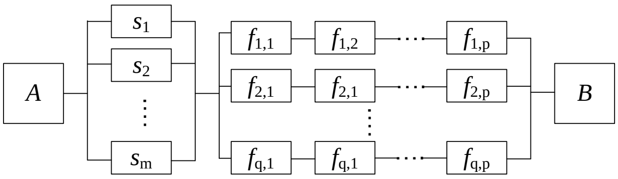

Figure 11: Reliability model of Pr^2-P2PSIP

Figure 11 shows the resulting reliability model under these assumptions. A UA A calling B needs to reach at least one of the storage peers s_i which have stored the contact data of B. Then, A needs to find at least one inbound tunnel to B where all peers which build the tunnel are still online. As shown in Figure 11, let m be the number of storage peers, p the length of B's inbound tunnels and q the number of parallel inbound tunnel.

Estimating the Overhead of Privacy.

If $p = 0$, then we have a regular P2PSIP network. Let m_0 the number of required parallel storage peers, then it follows from equation (10):

$$1 - (1 - \mu)^{m_0} \geq \bar{R} \tag{12}$$

Thus, the number of required storage peers for an inbound tunnel length $p = 0$ can be estimated by:

$$m_0 \geq \frac{ln(1 - \bar{R})}{ln(1 - \mu)} \tag{13}$$

If $p \geq 1$, then the reliability of the storage part at the end of each refreshing period can be estimated as:

$$(1 - (1 - \mu)^m) \tag{14}$$

and the reliability of the inbound forwarding part:

$$(1 - (1 - \mu^p)^q) \tag{15}$$

Let $\bar{R}_s$ the target reliablity of the storage part and $\bar{R}_f$ the target reliability of the inbound forwarding part. Thus, m and q can be estimated as follows:

$$m \geq \frac{ln(1 - \bar{R}_s)}{ln(1 - \mu)} \tag{16}$$

$$q \geq \frac{ln(1 - \bar{R}_f)}{ln(1 - \mu^p)} \tag{17}$$

and the reliability of the whole system:

$$(1 - (1 - \mu)^m).(1 - (1 - \mu^p)^q) \geq \bar{R}_s \bar{R}_f = \bar{R} \tag{18}$$

As it can be seen in Figure 11, the overall number of peers required for each UA in order to be reachable is $(m + pq)$.

By varying the ratio ${}^{\bar{R}_s}/_{\bar{R}_f}$ for a constant system target reliability $\bar{R} = 1 - 10^{-5}$ we obtain different values for $(m + pq)$ which are slightly better than equal target reliabilities for both parts, i.e., ${}^{\bar{R}_s}/_{\bar{R}_f} = 1$. Thus, we determine numerically the optimum value of $(m+pq)$ by varying ${}^{\bar{R}_s}/_{\bar{R}_f}$ for different values $p \in \{0, 1, 2, 3\}$ and $\mu \in (0, 1]$ and $\bar{R} = 1-10^{-5}$ (values of μ are chosen stepwise with steps of 0.01). Figure 12 shows the result. The number of peers required for a UA to be reachable for incoming SIP message increases to infinity if $\mu \to 0$ (i.e., average peer lifetime is $\varepsilon \to 0$) and converges to $(p + 1)$ for $\mu \to 1$ (i.e. a static network with peers never leaving).

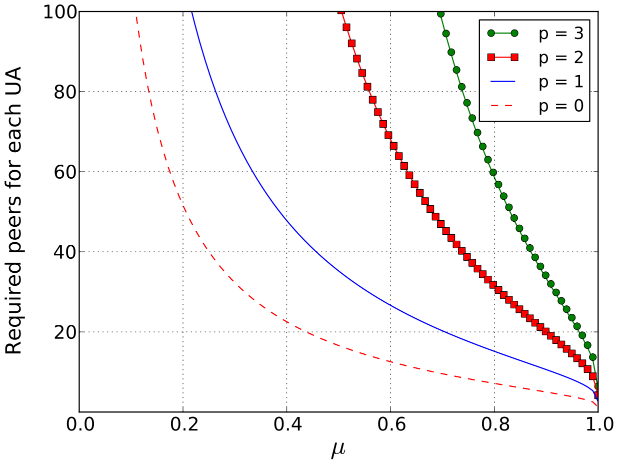

Figure 12: Number of peers required to keep a UA reachable in a Pr^2-P2PSIP network with target reliability $\bar{R} = 1 - 10^{-5}$

Interpretation based on Skype Traces. Using the Skype network as an example, according to [16], around 87% of the Skype super-peers have a peer lifetime more than $30mn$ and 78% more than $1h$. We interpolated these values to estimate the privacy overhead for $p = 3$ with different refreshing periods. The result is shown in Table 2. E.g., assuming a refreshing period of $20mn$ in Pr^2-P2PSIP, then around 33 peers would be required to keep a UA reachable for incoming calls. However, taking only Skype *super-peers* into consideration means that in Pr^2-P2PSIP only stable peers should be used for storage and inbound tunnels.

Table 2: Estimation of the privacy overhead based on Skype traces

Refreshing period (τ)	μ	Number of storage peers (m)	Number of inbound tunnels (q)	Total number of peers ($m+pq$)
10.0 mn	0.95	5	7	26
20.0 mn	0.91	6	9	33
30.0 mn	0.87	6	12	42
40.0 mn	0.84	7	14	49
50.0 mn	0.81	8	17	59
60.0 mn	0.78	9	19	66

Note that if a UA needs around 33 peers for storage and inbound tunnels, this means also that each UA will receive on average 33 requests within $20mn$ from other peers to store data or be a part of an inbound tunnel. Additional signaling is required for the outbound tunnels, overlay maintenance and DHT lookups.

Conclusions. The reliability analysis above provides an estimation of the impact of adding privacy to P2PSIP. The signaling overhead generated by Pr^2-P2PSIP to keep a target reliability of $(1-10^{-5})$ should not be underestimated. Further, the overhead is sensitive to the stability of the storage and forwarding peers. This may have different consequences depending on the types of devices used for the UAs. Processing a few requests per minute for storage, tunnels, DHT lookups and overlay maintenance may not be a problem for fixed hardphones, but would mean a large resource consumption for mobile devices, in particular if they are constantly awoken from standby mode (at least, this is a problem today). Given that the signaling overhead is sensitive to the stability of the storage and forwarding overlay networks, it is crucial for Pr^2-P2PSIP to exclude peers with a short lifetime from these overlays.

3.3 Cryptographic Overhead

Given the design decisions described in Section 2.4, the overhead of the public key encryption of a message m sent from a to b using a 194 bit ECC key $+K_b$ and a 128 bit temporary symmetric key $K_{a,b}$ for AES encryption in CBC mode consists of:

- the length of $\{K_{a,b}\}_{+K_b}$, which results in an ECC block size of 194 bits,
- the length of the initialization vector used for the symmetric encryption in CBC mode: 128 bits,
- and a maximum padding of 128 bits for the symmetric encryption,

which results in an overall overhead between 322 and 450 bits, i.e,. approximately between 40 and 56 bytes. Thus, even if a message is onion-encrypted with three layers the overhead in terms of message length remains acceptable.

However, the cryptographic overhead of Pr^2-P2PSIP in terms of the number of public key operations increases linearly with the number of tunnels per UA and the number of peers per tunnel. Thus, the same conclusions hold here as in Section 3.2.

3.4 End-to-end Signaling Latency

The signaling latency from UA A to UA B is affected by:

1. the processing overhead at each forwarding peer,
2. the tunnel length, or the number of forwarding peers used for inbound and outbound tunnels,
3. the accumulated one-way-delay along the full path between A and B,
4. the probability that all forwarding peers in a path are online since they were last.

As mentioned in Section 2.4, once a tunnel is setup, only symmetric cryptography is used. Thus, the cryptographic processing is certainly not a bottleneck. As for the tunnel length and the accumulated delay, we believe that Pr^2-P2PSIP deployed with the recommended tunnels lengths in Section 3.1 does not necessarily involve more signaling hops than server-based SIP networks used in practice today, in particular, where quite a few components are involved in the signaling for different purposes, e.g., lawful interception, billing, etc.

As for the probability that all forwarding peers in a path are online, as mentioned in Section 2.3.4, A tries another end-to-end path, i.e., another combination of outbound tunnel of A and inbound tunnel of B if it does not receive an acknowledgement to a SIP message within $1s$.

Thus, the maximum overall signaling latency is expected to be within a few seconds. If peers in the forwarding overlay are stable, it becomes more likely that the tunnels are available and the signaling succeeds at the first attempt, thus reducing the latency by an order of magnitude. If Pr^2-P2PSIP is used for chat, the same tunnels should be used for subsequent chat messages, since once tunnels have been successfully used, they are likely to remain available for the next chat messages, assuming a heavy-tailed distribution of the peer lifetime.

4. RELATED WORK

Location privacy was not a main concern when the Internet was conceived, because hosts were fixed. However, it was considered early on in GSM standardization. In GSM and UMTS networks, each mobile devices has a unique identifier called the International Mobile Subscriber Identity (IMSI). However, temporary pseudonyms called Temporary Mobile Subscriber Identities (TMSI) are usually used for communication with base stations. Nevertheless, both GMS and UMTS authentication protocols allow an attacker to impersonate a base station and request the User Equipment (UE) to send its IMSI for authentication.

P2PSIP was suggested initially by [7] and [34] and raised much interest and follow up work. Seedorf [32] discusses the security issues inherent in P2PSIP and mentions privacy briefly. In [4], the authors investigate a game theoretical approach for the security threats of P2PSIP such as SPIT and attacks on overlay routing. However, privacy is not addressed.

RELOAD [18], the base protocol for P2PSIP allows for different overlay algorithms to be plugged in. The IETF P2PSIP WG charter [2] does not preclude the deployment of anonymization networks. However, it can not be assumed that any general purpose anonymization network could be used. The Internet draft [17] describes SIP usage for RELOAD and mentions explicitly that “all RELOAD SIP registration data is public. Methods of providing location and identity privacy are still being studied”. Thus, Pr^2-P2PSIP is right on target to address this issue.

Reliability theory has been used in [35] for modeling P2P networks in the context of the KAD file sharing network. In [19], the authors investigate self-tuning behavior of DHTs in order to optimize the reliability costs in the context of Pastry. However, they consider only the reliability of overlay routing. In [38], the authors investigate the costs of maintenance and lookup in DHTs with different ratios of super peers. Their work considers regular DHT functionality without privacy. Nonetheless, our work can be enhanced in the future with a similar analysis in order to provide better insight on the signaling overhead of Pr^2-P2PSIP with different ratios of fixed and mobile devices with different resources. In [8,36] the authors demonstrate how the end points of P2P VoIP streams, e.g. Skype streams, can be identified. Thus, they demonstrate how one could break location and social interaction privacy. However, Skype peers do not consider each other as potentially malicious.

There are many anonymization networks which utilize onion routing [15] or a derivative, notably Tor [10], JAP [12], MorphMix [28] and I2P [11]. They all share characteristics and sometimes differ only in subtle ways. Our intention is not to invent a new anonymization network or new anonymization techniques, but to leverage existing techniques, particularly onion routing and inbound and outbound tunnels to address the privacy issues of P2PSIP. Nevertheless, Pr^2-P2PSIP can still be clearly differentiated from existing anonymization networks in several aspects. Approaches for anonymization networks can be classified into centralized and P2P approaches. Pr^2-P2PSIP is a P2P approach. Centralized approaches, e.g., Tor [10], Crowds [27] and MorphMix [28] rely on centralized databases (although eventually redundant as in the Tor case) to get a list of relay nodes. Pr^2-P2PSIP relies on a forwarding overlay. Likewise, Tor hidden services, which can be compared to Pr^2-P2PSIP inbound tunnels, are accessed via service descriptors stored in a central database. In Pr^2-P2PSIP, peers get the contact data from the DHT before they contact the inbound tunnel entry points.

In P2P anonymization networks, such as I2P [11], Salsa [24], Cashmere [37], Tarzan [14] and AP3 [22], there is no central authority as in Pr^2-P2PSIP, which makes them vulnerable to Sybil attacks. Further, peers select forwarding peers from their P2P routing tables. This makes them vulnerable to attacks where malicious peers attempt to dominate the routing tables of other peers. Pr^2-P2PSIP uses a separate overlay for forwarding and chooses forwarding peers randomly.

Pr^2-P2PSIP allows anonymous routing only within the network. Other anonymity networks such as JAP [12], Cashmere [37], Tarzan [14], MorphMix [28] and Crowds [27] are designed to allow communication with normal servers in the Internet. Thus, they need to support outbound connections. On the other hand, the clients do not have to be reachable for incoming communication as in Pr^2-P2PSIP.

In summary, Pr^2-P2PSIP benefits from the design of Tor and other anonymization networks and experience learned from them, while it has been designed exclusively to provide the P2P-based SIP user registration and session establishment, while preserving the privacy of the network participants. To the best of our knowledge, there has been no work which provides a dedicated solution to the privacy needs of P2PSIP with such an extensive analysis of the implications.

5. CONCLUSIONS

Our conclusions are as follows: Pr^2-P2PSIP provides location and social interaction privacy with a tunnel length of three for inbound tunnels and two for outbound tunnels. Cryptographic overhead is not a hindrance for Pr^2-P2PSIP, in particular if ECC is deployed. Signaling latency improves as the forwarding overlay becomes more stable. The signaling overhead to keep a target reliability of $(1-10^{-5})$ should not be underestimated. Further, the signaling overhead is sensitive to the stability of the forwarding overlay. Thus, it is crucial for a successful deployment of Pr^2-P2PSIP that stable peers, i.e., those with a long lifetime, are preferentially chosen for building tunnels.

6. ACKNOWLEDGEMENT

The authors would like to thank Christian Grothoff, Georg Carle and the anonymous reviewers for their valuable feedback and support for this paper. This work is partially funded by the EU project ResumeNet (FP7-224619) and by Deutsche Forschungs Gemeinschaft (DFG) under ENP GR 3688/1-1.

7. REFERENCES

[1] Geo ip tool - view my ip information. http://www.geoiptool.com/. Last checked on Feb. 10th 2010.

[2] IETF P2PSIP working group charter. http://www.ietf.org/dyn/wg/charter/p2psip-charter.html. Last checked on Feb. 10th 2010.

[3] A. Back, U. Möller, and A. Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In I. S. Moskowitz, editor, *Proceedings of Information Hiding Workshop (IH 2001)*, pages 245–257. Springer-Verlag, LNCS 2137, April 2001.

[4] S. Becker, R. State, and T. Engel. Using game theory to configure P2P SIP. In *Proceedings of IPTComm '09, Atlanta, Georgia*, pages 1–9. ACM, 2009.

[5] N. Borisov, G. Danezis, P. Mittal, and P. Tabriz. Denial of service or denial of security? How attacks on reliability can compromise anonymity. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 92–102, New York, NY, USA, October 2007. ACM.

[6] D. A. Bryan and T. Broadband. P2P SIP. http://www.p2psip.org. Last checked on Feb. 10th 2010; last updated Jul. 2009.

[7] D. A. Bryan, B. B. Lowekamp, and C. Jennings. Sosimple: A serverless, standards-based, p2p sip communication system. In *AAA-IDEA '05: Proceedings of the First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*, pages 42–49, Washington, DC, USA, 2005. IEEE Computer Society.

[8] S. Chen, X. Wang, and S. Jajodia. On the anonymity and traceability of peer-to-peer voip calls. *IEEE Network*, 20(5):32–37, 2006.

[9] G. Danezis. Statistical disclosure attacks: Traffic confirmation in open environments. In Gritzalis, Vimercati, Samarati, and Katsikas, editors, *Proceedings of Security and Privacy in the Age of Uncertainty, (SEC2003)*, pages 421–426, Athens, May 2003. IFIP TC11, Kluwer.

[10] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[11] I. P. M. et. al. I2p tech intro.

[12] H. Federrath. Jap: Anonymity and privacy. http://anon.inf.tu-dresden.de, 2000-2006.

[13] N. Ferguson and B. Schneier. *Practical Cryptography*. John Wiley and Sons (1st edition), 2003.

[14] M. J. Freedman, E. Sit, J. Cates, and R. Morris. Introducing tarzan, a peer-to-peer anonymizing network layer. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 121–129, London, UK, 2002. Springer-Verlag.

[15] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding Routing Information. In R. Anderson, editor, *Proceedings of Information Hiding: First International Workshop*, pages 137–150. Springer-Verlag, LNCS 1174, May 1996.

[16] S. Guha, N. Daswani, and R. Jain. An experimental study of the skype peer-to-peer voip system. In *IPTPS'06: The 5th International Workshop on Peer-to-Peer Systems*, 2006.

[17] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. A SIP Usage for RELOAD. draft-ietf-p2psip-sip-04, Internet Draft, Work in Progress, 2010.

[18] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, and H. Schulzrinne. REsource LOcation And Discovery (RELOAD). draft-ietf-p2psip-base-08, Internet Draft, Work in Progress, 2010.

[19] R. Mahajan, M. Castro, and A. Rowstron. Controlling the cost of reliability in peer-to-peer overlays. In *In IPTPS'03*, 2003.

[20] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *Peer-To-Peer Systems: First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002*, pages 53–65, 2002.

[21] J. McLachlan and N. Hopper. Don't clog the queue! circuit clogging and mitigation in p2p anonymity schemes. In *Financial Cryptography*, pages 31–46, 2008.

[22] A. Mislove, G. Oberoi, A. Post, C. Reis, P. Druschel, and D. S. Wallach. Ap3: cooperative, decentralized anonymous communication. In *EW 11: Proceedings of the 11th workshop on ACM SIGOPS European workshop*, page 30, New York, NY, USA, 2004. ACM.

[23] S. J. Murdoch and G. Danezis. Low-cost traffic analysis of Tor. In *SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 183–195, Washington, DC, USA, May 2005. IEEE Computer Society.

[24] A. Nambiar and M. Wright. Salsa: A structured approach to large-scale anonymity. In *Proceedings of CCS 2006*, October 2006.

[25] L. Øverlier and P. Syverson. Locating hidden servers. In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 100–114, Washington, DC, USA, May 2006. IEEE Computer Society.

[26] M. Rausand and A. Hoyland. *System Reliability Theory; Models, Statistical Methods, and Applications.* Addison-Wesley Publishing Company (2nd Edition), Reading, Massachusetts, 2004.

[27] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1), June 1998.

[28] M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In *WPES '02: Proceedings of the 2002 ACM workshop on Privacy in the Electronic Society*, pages 91–102, New York, NY, USA, November 2002. ACM.

[29] J. Rosenberg. A Presence Event Package for the Session Initiation Protocol (SIP). RFC 3856 (Proposed Standard), Aug. 2004.

[30] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630.

[31] M. Rosing. *Implementing elliptic curve cryptography.* Manning Publications Co., Greenwich, CT, USA, 1999.

[32] J. Seedorf. Security challenges for peer-to-peer sip. *IEEE Network*, 20(5):38–45, 2006.

[33] A. Serjantov and P. Sewell. Passive attack analysis for connection-based anonymity systems. In *Proceedings of ESORICS 2003*, October 2003.

[34] K. Singh and H. Schulzrinne. Peer-to-peer internet telephony using sip. Technical report, Columbia University CUCS-044-04, 2004.

[35] M. Steiner, T. En Najjary, and E. W. Biersack. A global view of KAD. In *IMC 2007, ACM SIGCOMM Internet Measurement Conference, October 23-26, 2007, San Diego, USA*, 10 2007.

[36] X. Wang, S. Chen, and S. Jajodia. Tracking anonymous peer-to-peer voip calls on the internet. In *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, pages 81–91, New York, NY, USA, 2005. ACM.

[37] L. Zhuang, F. Zhou, U. C. Berkeley, B. Y. Zhao, and A. Rowstron. Cashmere: Resilient anonymous routing. In *In Proc. of NSDI*. ACM/USENIX, 2005.

[38] S. Zoels, Z. Despotovic, and W. Kellerer. Cost-based analysis of hierarchical dht design. In *P2P '06: Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing*, pages 233–239, Washington, DC, USA, 2006. IEEE Computer Society.