Strong Pseudonymous Communication for Peer-to-Peer Reputation Systems

Michael Kinateder, Ralf Terdic, and Kurt Rothermel

Institute of Parallel and Distributed Systems (IPVS)
Universität Stuttgart
Universitätsstr. 38
70569 Stuttgart, Germany

{kinateder, rothermel}@informatik.uni-stuttgart.de, ralf@terdic.de

ABSTRACT

In this paper we present a novel approach to enable untraceable communication between pseudonyms. Our work provides strong sender and recipient anonymity by eliminating the need to know of each other's address.

We use a variation of Chaum mixes to achieve unlinkability between sender and recipient and introduce a concept called *extended destination routing* (EDR) which relies on routing headers constructed in multiple layers of encryption and published in a *distributed hash table* (DHT). In order to communicate, a sender requests from the DHT the recipient's routing header, which is extended and used for routing the message via a mix cascade to this recipient.

This work was performed in the context of the UniTEC reputation system and describes the functionality of its anonymous communication layer, which is completely independent of the other UniTEC layers. Although trust and reputation systems in general are typical application areas for our contribution, the presented concepts are suitable for various other application areas as well. We have implemented a prototype of UniTEC and present the first results from an ongoing evaluation in our network emulation testbed.

Categories and Subject Descriptors

C.2.4 [**Communication Networks**]: Distributed Systems; E.3 [**Data**]: Data Encryption; K.4 [**Computing Milieux**]: Computers and Society—*Privacy, Electronic Commerce*

General Terms

SECURITY, ALGORITHMS

Keywords

Distributed reputation systems, data protection, pseudonymous communication, mixes, extended destination routing

1. INTRODUCTION

With the advent of the Internet and the ever increasing demand for communication world-wide, more and more information is *available to people* with uncertain quality and at the same time more and more *data about people* is gathered and stored electronically.

The former fact is addressed by research in the trust management and reputation system area that strives to help users estimate the quality of products, services etc. and assess the subjective trustworthiness of other users e.g. as reviewers/recommenders for the aforementioned recommendation targets. However, reputation systems collect and process sensitive user data, which amplifies the latter fact of detailed user profile building and the associated privacy risks. The main danger lies in matching these profiles to real user identities.

Anonymous communication methods are no viable option for reputation systems, since in order to build trust in an entity, a way of representing that entity is necessary. The alternative of using *digital pseudonyms* instead of real identities has raised much interest lately. Some persons would argue, that trust in a pseudonym cannot be built. However, virtual communities do exist nowadays and people are making friends via IRC, Instant Messenger clients or multi player online games, with people that they may have never seen in real life nor do they know, whether these peoples' virtual names, age, gender etc. resemble reality at all. In the light of these facts, *trusting pseudonyms is possible* and done already today.

Friedman and Resnick [7] describe the effect of using pseudonyms in the context of trust and reputation systems. In [10] we propose to use TCPA attestation identities in order to create pseudonyms. Besides providing a way to reveal the real identity of a pseudonym holder in a legal dispute this addresses also the risk of Sybil attacks defined by Douceur in [5]. Seigneur and Jensen argue in [14], how the inherent conflict of privacy protection on the one hand and trust-establishment on the other can be solved by fine-grained negotiation mechanisms.

So trust in pseudonyms is possible and the concept of pseudonyms is already being used in trust and reputation systems. Are we there yet and all problems are solved adequately? We think not. The aforementioned contributions are essential but in themselves not sufficient if the employed pseudonyms can be linked to a globally unique identifier like the IP address. Therefore, the users' *privacy* has to be *pro-*

tected at various layers, ranging from the application down to the communication medium.

Regarding the *communication aspects* in a *client-server scenario*, users can employ web anonymizers like the Java Anonymous Proxy (JAP) project[1] of the Technical University Dresden to protect their privacy. JAP and similar approaches build on the concept of Chaum mixes (discussed in Section 2) that hide the communication between two entities in the communication of large groups of senders and recipients. However, the mix approach is *not directly suitable for peer-to-peer* (P2P) type communication since clients need to know the IP address of the servers in order to communicate. Mixes therefore mainly protect clients but not servers. In spite of the increasing popularity of peer-to-peer based applications, there is still a need for anonymization techniques efficiently supporting this communication paradigm. Our approach meets this need and *provides untraceable P2P communication between pseudonyms* with strong sender and recipient anonymity. This serves as a base for the higher-level communication of the application built on top, for instance the UniTEC recommendation queries, trust assessment queries and the corresponding replies.

We structure our paper as follows: In the next section we give an overview of the related work in the area of anonymous communication. This is followed by a brief presentation of the UniTEC reputation system in Section 3. We cover the design goals that our system has to cope and the basic concepts enabling our approach in Section 4, which lead to the discussion of selected protocols in Section 5. Section 6 provides an evaluation of our proposed concepts with respect to security and performance issues and we conclude our work in Section 7.

2. RELATED WORK

In this Section we present a selection of the related work in the area of anonymous communication, namely the basic concepts of onion routing, crowds and DC networks and the related projects Tarzan and Freenet, which partly build on top of these basic concepts. The used terminology for classifying the concepts draws on the work of Pfitzmann and Waidner in [12].

Chaum introduced in [1] the idea of *mix networks* that describes how nodes called *mixes* can be used to enable anonymous email messaging. Later on, Reed, Syverson and Goldschlag have taken on this idea in the concept of *onion routing* described first in [8] for allowing general anonymous communication. Mixes are nodes used by senders as intermediaries to pass on messages either to other mixes (thereby forming a so-called *mix cascade*) or finally to the intended recipient. A mix accumulates a certain number of messages, reorders them and passes them on. If the necessary number of messages is not received within a certain time the mix creates dummy messages to fill up the queue and sends the messages on. Instead of transmitting in plain text over a cascade of mixes $M_1, \ldots, M_x$ the sender wraps multiple layers of encryption around data d thereby forming an onion-like structure.

The approach of mixes provides *unlinkability* between sender and recipient even in the case of a global attacker. This attacker can observe groups of senders communicating with groups of recipients, but it cannot observe which individual from the sender group communicates with which individual from the recipient group. The anonymity property holds even as long as there is just a single non-corrupt mix in the cascade. On the downside, the sender has to know the address of the recipient in order to form the onion, therefore this approach does not provide recipient anonymity. The solution presented in this paper builds on mixes while ensuring the property of recipient anonymity.

Freedman and Morris proposed in [6] the Tarzan system that builds on mix networks and establishes a peer-to-peer anonymizing IP network overlay. It guarantees high resistance against traffic analysis through the use of layered encryption, multi-hop routing, cover traffic and a special mix selection protocol. Tarzan provides a high level of sender and recipient anonymity; however, the sender still has to know the address of the recipient in order to communicate.

A theoretical concept named *DC Networks* was developed by Chaum [2] and enhanced by Waidner and Pfitzmann in [16]. This approach has interesting anonymity properties by providing *sender and recipient anonymity* plus *unlinkability* of sender and recipient. However, implementations are still rare due to the massive communication overhead.

Reiter and Rubin introduced the *Crowds* system (see [13]) that provides users with the possibility to hide their transactions with a specific web server in those transactions of all the other users in the crowd. The Crowds concept does not provide unlinkability between sender and recipient in case of a global attacker. For non-global attackers Crowds provide sender anonymity, since they allow the user to deny having sent a request to the webserver.

A project that is related to the Crowds approach is the Freenet[2] project [3]. Freenet is a peer-to-peer network for distributed data storage that provides sender anonymity against collaborating nodes. Clarke et al. argue that the anonymity properties can again be strengthened by employing mixes for pre-routing messages. We should note that the focus of Freenet lies in anonymous distributed data storage and that it cannot be easily applied to P2P communication.

3. THE UNITEC REPUTATION SYSTEM

This section briefly introduces the UniTEC background as one application area for the proposed approach.

Its main focus is to provide trust management functionality to users and applications alike. This is achieved by placing a *UniTEC agent* on each participating user's computer. We argued in [10] that the UniTEC system is greatly strengthened if the computing platform that the agent resides on is a *trusted platform*, but this is not a prerequisite as such. Each agent consists of several components as described in Fig. 1.

For privacy reasons, as pointed out in the introduction, each user uses multiple digital pseudonyms for interacting in the system. Each pseudonym has an associated public and private key pair and is responsible for a certain part of the trust model, e.g. pseudonym A is responsible for car recommendations whereas pseudonym B handles book recommendations and so on. The *identity management component* (IMC) provides e.g. pseudonym creation, assignment of responsibilities in the trust model and the secure intentional disclosure of the link between pseudonym and real identity for certain highly trusted entities.

[1] `http://anon.inf.tu-dresden.de/index_en.html`

[2] `http://freenet.sourceforge.net`

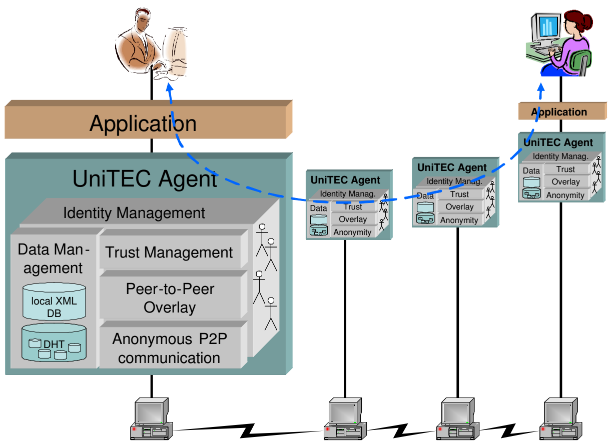

Figure 1: UniTEC System Model: Nodes with communication capabilities host UniTEC agents that provide *pseudonymous* communication capabilities to applications respectively their users.

Recommendations, (e.g. about products or services) or arbitrary trusted data items are digitally signed with the key of the appropriate pseudonym and stored in the XML database of the *data management component* of the pseudonym owner's UniTEC agent.

The *anonymous peer-to-peer communication component* provides basic messaging capabilities between digital pseudonyms to the upper UniTEC layers or other applications while hiding the link between pseudonyms and their owner's real identity. How this can be achieved is the main contribution of this paper and topic of the following sections.

An overlay is constructed by the agents communicating via the basic anonymous messaging capabilities. The *peer-to-peer overlay component* allows the construction of application-specific overlays on top of the generic UniTEC overlay formed by the existence of the agents.

Information queries, e.g. for recommendations or trust statements regarding previously unknown pseudonyms, are sent in the overlay under a certain pseudonym on behalf of its owner. Agents that can satisfy a certain query send the digitally signed response back to the requesting pseudonym. A trust chain is built during this process from the requester to the recommender. When the response is received at the requester, the resulting trust of the requester in the recommender is calculated by evaluating the trust chain. Finally, information and trust are presented to the requesting user. At a later point in time, the user provides feedback to the UniTEC agent regarding the quality of the data of each recommender which leads to an update in the requester's trust in the recommenders. These affairs are the responsibility of the *trust management* component, which keeps track of its user's trust in the known pseudonyms, updates this trust and calculates trust in foreign pseudonyms with which no previous experiences have been made with.

More information on UniTEC can be found in [11].

4. DESIGN GOALS AND DESIGN

In this section we introduce the design goals that the anonymous communication layer has to fulfill and cover the basic functionality offered by UniTEC agents.

4.1 Design Goals

- **Fault tolerance:** The failure or takeover of any node may not lead to the failure of the whole system.
- **Peer-to-Peer:** The system must not depend on centralized concepts in order to be applicable in a peer-to-peer context.
- **Ensure strength of pseudonyms:** The underlying communication layer may not enable an attacker to link the pseudonym to a real identity through the use of a globally unique identifier (e.g. the IP address). Furthermore, the concepts must not allow an attacker to link different pseudonyms of an entity to each other.
- **Unlinkability of sender and recipient:** Even if it may be possible to discern that a sender is communicating and a recipient is receiving information, it should not be possible to notice that they are communicating with *each other*.
- **Sender and recipient anonymity:** The initiator respectively the recipient of the communication should remain hidden from other UniTEC nodes and – most notably – also from each other.

4.2 Notation

Throughout the paper we will use the following notation:

A_S, A_R	*IP address* of the agent, where the sender's pseudonym S or the recipient's pseudonym R is registered
A_{M_X}	*IP address* of the agent M_X that has enabled the mix service
Id_S, Id_R	Identifier of pseudonym S respectively R
PrK_X, PuK_X	*Private key* respectively *public key* of entity X (X might be a pseudonym or mix)
$Enc(PuK_X, D)$	Data D *encrypted* with the public key of entity X
$Hash(D)$	Hash of data D
$\{D_1, D_2\}$	Data item D_2 appended to data item D_1

4.3 Properties of Digital Pseudonyms

The digital pseudonyms are managed by the UniTEC agent's IMC. Upon creation of pseudonym P, the IMC creates a public and private key pair (PuK_P, PrK_P) and an identifier (Id_P). The key pair is used among other things for ensuring confidentiality of the communication with P, whereas Id_P identifies P. We consider two options for identifying the pseudonyms: firstly the public key of the pseudonym and secondly the hash of this public key.

The public key could be used as an identifier since the likelihood of two pseudonyms having the same public key is infinitesimally small. However, the length of PuK_P (1024 Bit RSA) makes this solution infeasible as an address for communication. Therefore, our approach uses the second option, namely the *SHA-1 hash* (160 Bit) of the public key:

$$Id_P = Hash(PuK_P) \quad (1)$$

The fact that cryptographic hash-functions produce an equipartition of the output space ensures that the probability of two pseudonyms having the same identity is again miniscule. SHA-1 is a *secure hash-function*. Therefore finding a public key that produces a given hash is computationally infeasable. It is important to note, that this connection between identity and public key relieves us from the necessity to employ digital certificates. Obviously, when a key gets compromised, the pseudonym cannot be used any longer. One issue still has to be solved: how can we make these keys available to other pseudonyms without compromising the anonymity.

4.4 Publishing Public Keys

Three different entities could publish the public keys: the agent of the key owner, a centralized or a distributed repository. Storing a public key on the key owner's agent however would enable inquirers to see the link between the IP address of the agent and the identity of the pseudonym, the very thing we need to keep hidden. A centralized solution well-known in the area of public key infrastructures is to employ a directory service. Yet this contradicts the requirement of no centralized components in the UniTEC system.

Therefore we chose the distributed approach, namely we built our solution on Chord (see [15]), a redundant representative of distributed hash tables (DHTs). Chord organizes its participating nodes in a logical ring. Each UniTEC agent participates in this Chord ring and stores part of the content of the whole database. Upon startup, the agent registers all of its managed pseudonyms in the DHT as we explain in more detail in Section 5.2.

4.5 Mixing

As base for anonymity UniTEC agents provide mix functionality in a similar way to David Chaums mix network approach described in the related work section. When enabled, the mixing service creates a public and private key pair that is used for encrypting messages to be sent via this mix. It is the user's choice whether or not to enable the mix service at their agent, however, we argue later that not enabling this service results in weaker anonymity properties for the pseudonyms registered at that agent.

Mixes in traditional onion routing strip off the messages' layers of encryption and pass the messages on. Compared to that, UniTEC mixes perform different operations as described in the following section. All the mixes send *dummy traffic* in order to cope with message sparsity.

5. SELECTED PROTOCOLS

In the following we describe the concept of extended destination routing (EDR) and several selected protocols necessary to enable EDR.

5.1 Extended Destination Routing (EDR)

We first cover a basic version of EDR which still allows two different kinds of attacks which we address in the following. The adaptations to basic EDR which prevent the aforementioned attacks form our suggested approach of full EDR.

5.1.1 Basic EDR

Conventional onion routing assumes the availability of the recipient's address information. In order to ensure recipient anonymity, this information cannot be made available in plain text. To solve this problem, pseudonyms store their *addresses* encrypted as *onions* in a lookup service, in our case the DHT Chord. The resulting onion is obtained by communication initiators and used as a *routing header* for routing the message to the intended recipient. As an example, a pseudonym R chooses a mix cascade containing three mixes M_1, M_2 and M_3. R encrypts its address A_R successively with the public keys of the chosen mixes PuK_{M_1}, PuK_{M_2} and PuK_{M_3}. The resulting routing header RH is:

$$\begin{aligned} RH = \{\ & A_{M_1}, \\ & Enc(\ PuK_{M_1}, \\ & \quad \{\ A_{M_2}, \\ & \quad Enc(\ PuK_{M_2}, \\ & \quad\quad \{\ A_{M_3}, \\ & \quad\quad Enc(\ PuK_{M_3}, \\ & \quad\quad\quad A_R\ )\}\ )\}\ )\} \end{aligned} \tag{2}$$

As the pseudonyms' routing headers and public keys are stored in the DHT, they are available to all potential senders. If a pseudonym S wants to send a data item D to pseudonym R, it encrypts D with R's public key and appends R's routing header after stripping off the address of the first mix A_{M_1}. The resulting message M is sent to M_1:

$$\begin{aligned} M = \{\ & Enc(\ PuK_{M_1}, \\ & \quad \{\ A_{M_2}, \\ & \quad Enc(\ PuK_{M_2}, \\ & \quad\quad \{\ A_{M_3}, \\ & \quad\quad Enc(\ PuK_{M_3}, \\ & \quad\quad\quad A_R\ )\}\ )\}\ ), \\ & Enc(\ PuK_R,\ D\ )\} \end{aligned} \tag{3}$$

The receiving mix decrypts the remaining part of the routing header, strips off the address of the next mix A_{M_2} and uses this address to forward the data. This is recursively repeated until R receives $Enc(PuK_R, D)$.

The basic EDR described until now permits two attack possibilities, *message tracing* and the *own mix attack* which we both address in the following by the real EDR.

5.1.2 Message Tracing

The encrypted payload $Enc(PuK_R, D)$ remains the same during the communication, allowing message route tracing and thus endangering the unlinkability between S and R. The basic defence against this form of attack is *hop-by-hop encryption*: At each node, before forwarding a message, the payload part of the message is encrypted with the public key of the next hop (e.g. $Enc(PuK_{M_1}, Enc(PuK_R, D))$ for the first mix). After receiving a message, its payload is decrypted and then again encrypted with the public key of the next hop (PuK_{M_2}), and finally the message is sent to this hop. A random-length padding is applied before the encryption in order to vary the payload's length and cope with statistical attacks.

5.1.3 Own Mix Attack

A malicious pseudonym R can manipulate its routing header by choosing itself as the only "mix" of the cascade:

$$\begin{aligned} RH = \{\ & A_R, \\ & Enc(\ PuK_R, \\ & \quad \{\ A_R, \\ & \quad Enc(\ PuK_R, \\ & \quad\quad A_R\ )\}\ )\} \end{aligned} \tag{4}$$

Since routing headers are encrypted, this manipulation cannot be detected. When the sender communicates directly with the first "mix" which is in fact R, its anonymity is compromised.

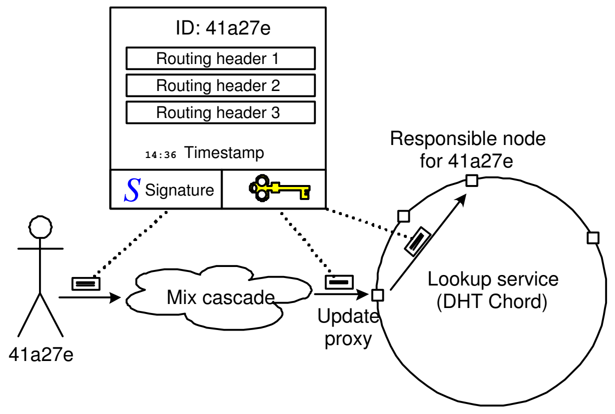

Figure 2: The Routing Update Protocol.

The solution defence against this form of attack is *extended destination routing.* After looking up the routing header, the sender chooses part of the cascade itself. A simple way to accomplish this without major changes to the described approach is to extend the recipient's cascade from the routing header with a sender-controlled part of the cascade. For example, the manipulated header mentioned above could be extended by the sender with a cascade part containing two mixes M_1 and M_2. The extended routing header (ERH) looks as follows:

$$\begin{aligned} ERH = \{\ & A_{M_1}, \\ & Enc(\ PuK_{M_1}, \\ & \quad \{\ A_{M_2}, \\ & \quad Enc(\ PuK_{M_2}, \\ & \quad\quad \{\ A_R, \\ & \quad\quad Enc(\ PuK_R, \\ & \quad\quad\quad \{\ A_R, \\ & \quad\quad\quad Enc(\ PuK_R, \\ & \quad\quad\quad\quad A_R\ )\}\ )\}\ )\}\end{aligned} \tag{5}$$

5.2 How to Store Public Keys and Routing Headers?

As introduced in the previous section we store public keys as well as routing headers in the distributed hash table Chord. However, in order to preserve anonymity, the pseudonym information cannot be stored directly in the lookup service. As illustrated in Fig. 2 the updating pseudonym sends its information through a mix cascade to an *update proxy.* The update proxy can be any UniTEC agent with enabled update service.

The information to be stored consists of a signed package containing several routing headers for increased availability, a timestamp to avoid replay attacks, the pseudonym ID and its public key. The integrity and authenticity of the package can be checked by verifying the signature and comparing the hash of the public key with the ID.

After receiving this package, the update proxy determines the nodes responsible for storing information for this pseudonym based on the pseudonym ID and forwards the package to these nodes, which store it in memory. There is no need to store this data on persistent media since the information contained in the routing headers is based on the availability of the used mixes and updated at regular intervals.

One might ask why the updating pseudonym does not send the data directly to the responsible nodes instead of going via an update proxy. At the time being, none of the existing DHT implementations allow the determination of the responsible nodes while preserving anonymity for the querying entity. Furthermore, using an update proxy permits a separation between the DHT and the anonymity layer, allowing us to replace the DHT implementation if necessary.

5.3 Mix Discovery

A prerequisite for the presented approach is the availability of mix information, i.e. addresses and public keys, as they are used to create routing headers and to forward messages. For this purpose, every agent manages a list containing information about known available mixes.

Due to the use of Chord the participating agents are organized in a logical ring topology. When activating the mix service on an agent, the local mix information is pushed to the first n successors in the ring, which store it in their list. Periodically, every agent starts a mix discovery, asking a random agent for known mixes, thus pulling the information contained in that agent's list. The pulled information is merged with the local list, and eventually the resulting list is pruned of entries containing non-responsive mixes.

6. EVALUATION

6.1 Security – Strengths and Weaknesses

The general approach described in this paper provides sender and recipient anonymity through the use of extended destination routing, especially through the creation of routing headers and their storage in the DHT. By applying the mix concept we ensure unlinkability of sender and recipient as in the original onion routing.

On the downside, the usage of hop-by-hop encryption might motivate an attack on the mixes, since the payload is decrypted at each mix (but still encrypted with the recipient's public key). This leads to two relevant attack scenarios: *cooperation of corrupt mixes* and the *last mix attack.*

6.1.1 Cooperation of Corrupt Mixes

Corrupt mixes could exchange information about the forwarded payload, thereby being able to trace at least part of the message route. It is unlikely that, while using extended destination routing with a high number of available mixes, the resulting cascades entirely consist of corrupt mixes. Nevertheless, cooperating mixes can diminish the anonymity degree provided by a cascade.

This effect can be reduced (though not eliminated) by using a different approach for the cascade extension. Instead of just extending the routing header, the sender can use traditional onion routing for the sender-controlled part of the cascade. For example, using

$$\begin{aligned} RH = \{\ & A_{M_1}, \\ & Enc(\ PuK_{M_1}, \\ & \quad \{\ A_{M_2}, \\ & \quad Enc(\ PuK_{M_2}, \\ & \quad\quad A_R\ )\}\ )\}\end{aligned} \tag{6}$$

as published routing header, and

$$Enc(PuK_R, D) \tag{7}$$

as encrypted data to be sent to recipient R, the sender extends the recipient cascade M_1, M_2 with two more mixes

M_3, M_4, generating the following message M:

$$\begin{array}{l} M = Enc(\ PuK_{M_3}, \\ \qquad\qquad \{\ A_{M_4}, \\ \qquad\qquad\quad Enc(\ PuK_{M_4}, \\ \qquad\qquad\qquad\quad \{\ A_{M_1}, \\ \qquad\qquad\qquad\qquad Enc(\ PuK_{M_1}, \\ \qquad\qquad\qquad\qquad\qquad \{\ Enc(\ PuK_{M_1}, \\ \qquad\qquad\qquad\qquad\qquad\qquad \{\ A_{M_2}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\quad Enc(\ PuK_{M_2}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad A_R\ )\}\ ), \\ \qquad\qquad\qquad\qquad\qquad Enc(PuK_R,\ D\ )\})\})\}) \end{array} \tag{8}$$

Obviously no mix in the sender cascade can see the payload, as it resides safely inside the onion. This still does not protect from attacks on mixes in the recipient's part where the sender has no control over the cascade. However, one can argue that the recipient is responsible for selecting the mixes carefully. Dingledine et al. [4] describe an approach that uses reputation for selecting "good" mixes, which might be an option worth investigating, since UniTEC inherently provides reputation mechanisms.

6.1.2 *Last Mix Attack*

If the sender of a message happens to be the last mix in the cascade and it is aware of this by recognizing the payload and by analyzing the routing header – which is empty after stripping off the address of the last hop – then the recipient's address is exposed to the sender.

To avoid this, it is necessary to prevent mixes from identifying empty routing headers. This can be realized by including not only the address into the core of the recipient routing header, but also random size padding encrypted with R's public key. This prevents the last mix to notice that it actually sends to the recipient's agent since there might be hidden still more mixes of the recipient's part of the cascade in the padding, whose contents the last mix cannot access. This is the reason why we recommend to activate the mix service at each node, so that the last mix cannot discern from a non-activated recipient mix service that it communicates directly with the recipient.

6.2 Performance

The UniTEC prototype was developed using Java 2 SDK Standard Edition Version 1.4.1, Bouncy Castle Crypto API[3] as JCE provider and XML-RPC as communication protocol. The implementation is being evaluated via two evaluation scenarios on our 64 node network emulation cluster NET. More information on NET can be found in [9].

Figure 3 presents the first performance results of our first evaluation scenario. Here, we execute the UniTEC agent on 21 cluster nodes, 20 of which with enabled mix service in addition to one dedicated sender. The sender sends messages while recording the *round-trip-time* (RTT) until an acknowledgement for each message is received. We vary the message payload size and total mix cascade length while keeping the mix queue delay (0 ms) and RSA key size (1024 Bits) fixed. This allows us to focus mainly on the cryptographic operations and lookups in the DHT. The total mix cascade length is the sum of mixes in the sender and recipient cascades for message and acknowledgement. Each point in the figure represents the average RTT of 20 sent messages.

The second performance evaluation scenario was chosen to take into account the effects of background load. We

[3]`http://www.bouncycastle.org`

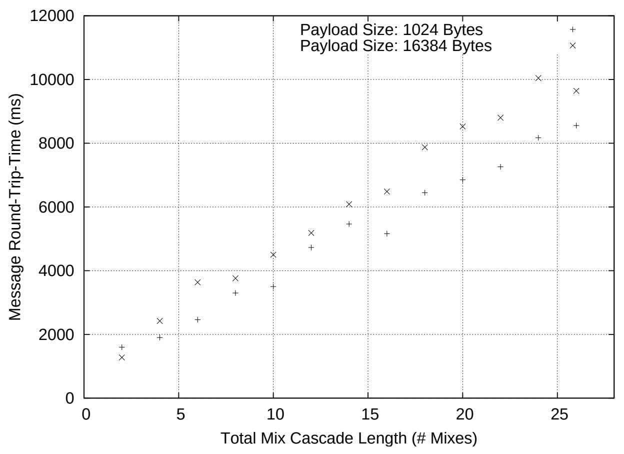

Figure 3: Message round-trip-time with varying payload size and cascade length in case of a single sender and 20 mixes.

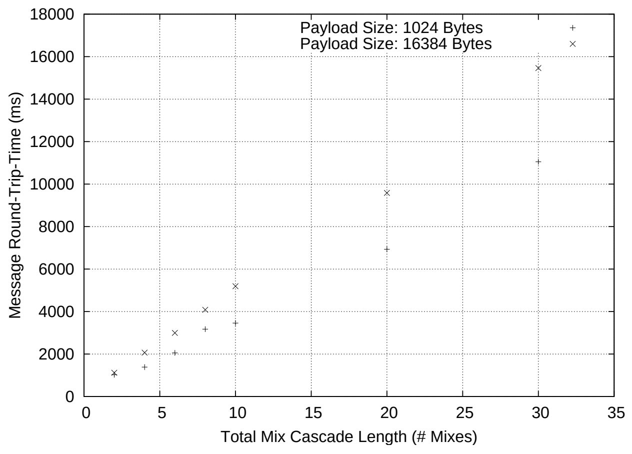

Figure 4: Message round-trip-time in case of 15 mixes, 16 non-mixes and background load.

perform this evaluation by executing the UniTEC agent on 31 cluster nodes with the mix service being enabled on 15 agents. 30 agents are generating load by each sending a 2048 Byte message every 5 seconds over a cascade of total length 10. The single agent sends messages with variable size and mix cascade length while measuring the RTT. The results of the evaluation can be seen in Fig. 4. Again, each point represents the average RTT of 20 messages.

When compared to Fig. 3, the difference of the background load in Fig. 4 seems almost negligible. In both figures, the RTT depends linearly on the length of the mix cascade. This is reasonable since for every additional hop, a fixed amount of time is necessary to provide the decryption and re-encryption of the message. We can see further that the gradient depends on the message size due to the fact that the time for performing the aforementioned actions increases linearly with the message size.

In Fig. 4 we see e.g. that a 16KB message takes on average 5s through a total cascade of length 10. The two main reasons for this high cost are firstly the usage of XML RPC and secondly the modest encryption performance offered by

Java. This is especially influent due to the fact that three asymmetric operations have to be performed at every mix: two for decrypting message and routing header and one for the message re-encryption.

Nevertheless this performance is still acceptable for the intended application scenario of a combined recommendation and reputation system and especially in the light of the offered anonymity and the fact that we evaluated a first prototype that can still be specifically optimized for performance.

7. CONCLUSION

In this paper we introduced the novel concept of extended destination routing which is a general approach for enabling communication between pseudonyms while at the same time preserving unlinkability between sender and recipient as well as strong sender and recipient anonymity. In addition to these anonymity properties, we achieve fault tolerance through a completely decentralized design.

Although the UniTEC system as a whole provides many additional functions like trust management, the P2P communication and data management layers work transparently to the upper layers and can be applied to other reputation systems and even other application areas as well. Our preliminary evaluation results show that the performance of the prototype is sufficient for the reputation system scenario and can still be improved with optimizations to address the needs of further application scenarios.

In our view, this area of anonymity preserving technologies for P2P applications has not drawn sufficient attention yet. With this paper, we made a contribution to this underresearched topic and we hope many more researchers are taking on the challenge to ensure users' privacy with this communication paradigm.

8. ACKNOWLEDGEMENTS

We thank Jochen Widmaier for interesting discussions during the conceptualization phase and contributions to the identity management functionality of the UniTEC prototype. Furthermore our thanks go to Daniel Herrscher, Timo Heiber, and Arno Wacker for good comments and helpful pointers on the further UniTEC evaluation.

9. REFERENCES

[1] D. Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–90, Feb. 1981.

[2] D. Chaum. The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.

[3] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Proceedings of the ICSI International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 46–66, Berkeley, CA, USA, June 2000. Springer-Verlag.

[4] R. Dingledine, M. J. Freedman, D. Hopwood, and D. Molnar. A Reputation System to Increase MIX-net Reliability. In I. S. Moskowitz, editor, *Proceedings of the 4th International Information Hiding Workshop*, volume 2137 of *LNCS*, pages 126–141, Pittsburg, PA, USA, Apr. 2001. Springer-Verlag.

[5] J. Douceur. The Sybil Attack. In *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS '02)*, pages 251–260. Springer-Verlag, 2002.

[6] M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 193–206. ACM Press, Nov. 2002.

[7] E. Friedman and P. Resnick. The Social Cost of Cheap Pseudonyms. *Journal of Economics and Management Strategy*, 10(2):173–199, 2001.

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

[9] D. Herrscher and K. Rothermel:. AA Dynamic Network Scenario Emulation Tool. In *Proceedings of the 11th International Conference on Computer Communications and Networks (ICCCN 2002)*, pages 262–267, Miami, USA, Oct. 2002.

[10] M. Kinateder and S. Pearson. A Privacy-Enhanced Peer-to-Peer Reputation System. In K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, editors, *Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies (EC-Web 2003)*, volume 2738 of *LNCS*, pages 206–215, Prague, Czech Republic, Sept. 2003. Springer-Verlag.

[11] M. Kinateder and K. Rothermel. Architecture and Algorithms for a Distributed Reputation System. In P. Nixon and S. Terzis, editors, *Proceedings of the First International Conference on Trust Management*, volume 2692 of *LNCS*, pages 1–16, Crete, Greece, May 2003. Springer-Verlag.

[12] A. Pfitzmann and M. Waidner. Networks without User Observability. *Computers and Security*, 6(2):158–166, Apr. 1987.

[13] M. K. Reiter and A. D. Rubin. Anonymity loves company: Anonymous Web transactions with Crowds. *Communications of the ACM*, 42(2):32–38, Feb. 1999.

[14] J.-M. Seigneur and C. D. Jensen. Trading Privacy for Trust. In *Proceedings of the Second International Conference on Trust Management*, volume 2995 of *LNCS*, pages 93–107, Oxford, UK, Mar. 2004. Springer-Verlag.

[15] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149–160, San Diego, CA, USA, Aug. 2001.

[16] M. Waidner and B. Pfitzmann. The dining cryptographers in the disco: Unconditional sender and recipient untraceability with computationally secure serviceability. In J.-J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology EUROCRYPT 89*, volume 434 of *LNCS*, pages 206–215, Houthalen, Belgium, Apr. 1989. Springer-Verlag.