Privacy-aware Presence Management in Instant Messaging Systems

Karsten Loesing, Markus Dorsch, Martin Grote, Knut Hildebrandt,
Maximilian Röglinger, Matthias Sehr, Christian Wilms, and Guido Wirtz

Otto-Friedrich-Universität Bamberg
Distributed and Mobile Systems Group
Feldkirchenstr. 21, 96047 Bamberg, GERMANY
{karsten.loesing, guido.wirtz}@wiai.uni-bamberg.de

Abstract

Information about online presence allows participants of instant messaging (IM) systems to determine whether their prospective communication partners will be able to answer their requests in a timely manner, or not. This makes IM more personal and closer than other forms of communication such as e-mail. On the other hand, revelation of presence constitutes a potential of misuse by untrustworthy entities, e.g. generation of presence logs. We argue that current IM systems do not take reasonable precautions to protect presence information. We propose an IM system designed to be robust against attacks to disclose a user's presence. It stores presence information in a distributed hash table (DHT) in a way that is only detectable and applicable for intended users and even not comprehensible for the DHT nodes. We apply an anonymous communication network to protect the users' physical addresses.

1. Introduction

One of the most important resources of instant messaging (IM) systems is *presence* of participants. Users provide their current presence to other users on their so-called contact list. In addition to the boolean information whether somebody is available or not, a user can add awareness information, e.g. the degree of willingness to communicate, or a reason for being temporarily unavailable. This information helps other participants to get a notion of their prospective communication partner's context before sending a message. Presence information—together with the ability to send messages instantly—makes IM more personal and closer than other electronic communication media.

Though presence information is a valuable resource for legitimate communication partners, it might potentially be *misused* by unauthorized entities. Typically, presence information is stored on a central or predictable node which may be subject to eavesdropping or infiltration attacks or might even be untrustworthy. An adversary (e.g. hacker, corporation, or government) might track a certain user's presence and awareness information to obtain a survey of his online actions. From such a presence log he could infer a business user's working hours and idle times and deduce a private user's leisure behavior. He could also make assumptions resulting from a change in usage behavior, e.g. absence from work or from home in case of a sudden discontinuation of system usage.

Apart from disclosure of a user's own presence information, an adversary could also reveal links between users. Users subscribe to receive each other's presence information. An adversary could either try to access these subscriptions, or intercept requests for a user's presence information by another user. This does not even require IM communication between both users. Revelation of connections between two persons could be precarious, e.g. if one of them is the CEO of a competitor, a headhunter, a dissident, etc.

All these threats apply predominantly to public IM systems which can be accessed from everywhere in the Internet, i.e. in an untrusted network. Corporations might set up their own corporate IM server and restrict usage to intranet users. Users outside the corporate network could access it using a virtual private network (VPN). Though this approach should resist attacks on disclosure of presence information, it has a major drawback: It restricts system access to corporate users and excludes employees of other corporations or other users unless they have access to the VPN. Further, corporate IM systems are not available for private users.

1-4244-0054-6/06/$20.00

In our opinion, public IM systems do not take reasonable precautions to protect presence information and VPN-based corporate IM systems are too restrictive. We propose a presence management system feasible for IM applications which impedes attacks attempting to disclose presence information. It does not require a trusted network, so that it may be applied by both, private and corporate users. Neither the content of presence information nor the store and retrieve operations may provide any hint at a user's presence to any other entity than the authorized contacts. This explicitly excludes the registry of the presence service. To achieve this, we utilize an anonymous communication network in order to obfuscate physical communication which might indirectly disclose presence as well. Additionally, we deploy a distributed hash table (DHT) instead of a central presence server to make attempts of traffic analysis more complicated.

The rest of this paper is organized as follows: First, we formulate in section 2 the requirements for a privacy-aware presence management system from a user perspective and the assumptions to the underlying system model. In section 3 it is investigated whether these requirements are met by existing IM systems. We discuss in section 4 the details of our proposed system as well as possible attacks on it. In section 5 we refer to related work. Section 6 concludes this paper and gives an outlook on future work.

2. Requirements and assumptions

Propagation of presence information: Presence information consists of the binary information whether a user is available or not, and a contact address, e.g. for exchanging awareness information or instant messages. Propagation of presence information is limited to a user-defined set of other users and is assumed to be accomplished in a timely manner.

Privacy of presence information: The presence management system has to make sure, that presence information is not disclosed to unauthorized persons inside or outside the system. The set of authorized entities explicitly does not include single trusted entities, because they constitute a target for eavesdroppers and attackers, or might in fact be untrustworthy.

Assumptions to underlying system model: Basically, the system consists of client nodes and a registry which are interconnected by bidirectional communication links. The registry is required for storing and retrieving presence information. It may be realized by a single node or a set of interconnected nodes. The network is not assumed to be secure, i.e. communication links are subject to eavesdropping and manipulation.

3. Threats in common IM systems

Common IM systems promise to restrict access to presence information to authorized users. However, they do not take reasonable precautions to protect it from unauthorized access. Most IM systems store their users' presence information on a central registry server. When logging into the system a user connects to this server and authenticates himself by using password authentication. Then he provides his own presence information and requests his contacts' presence information. Most systems provide encryption of communication between client and server using SSL. Figure 1 shows the nodes and communication channels as well as the propagation of presence information from Alice to Bob in a server-based IM system. Nodes with a gray dot are aware of Alice's presence. The MSN system (`http://messenger.msn.com/`) is an example for a server-based IM system. The web resource `http://www.hypothetic.org/docs/msn/` contains an unofficial, but elaborate protocol specification of the MSN protocol.

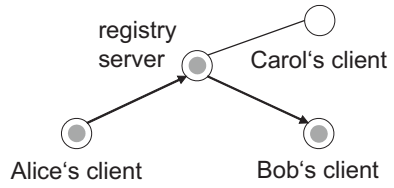

Figure 1. Propagation of presence information in a server-based IM system.

The following attacks are imaginable in a server-based IM system:

Eavesdropping of communication to the server: If the communication between Alice's client and her server is not encrypted, an eavesdropper can infer her presence simply by looking at the package contents. If packets are encrypted, an eavesdropper who is aware of Alice's IP address can still determine her presence by looking at the (unencrypted) packet header.

Infiltration of server: An attacker might break into the server and gain access to its data including Alice's contact lists and presence information.

Untrustworthiness of server: The server administrator might be untrustworthy and might have an agenda concerning presence information of his users. His intentions might range from performing research about his service usage up to passing data to third parties.

In contrast to the server-based IM systems, Jabber (see `http://www.jabber.org/` for the project homepage and the requests for comments [9] and [10] which

describe the closely related XMPP protocol) makes use of several registry servers in a decentralized way. Servers are distributed and interconnected via the Internet. Every client is connected to exactly one server. Figure 2 shows the propagation of presence information in Jabber. Alice's presence information is routed from herself via her server and Bob's server to Bob. All nodes with a gray dot are aware of Alice's presence.

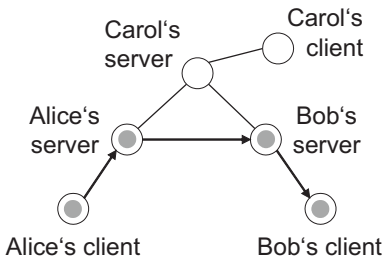

Figure 2. Propagation of presence information in Jabber.

Jabber exhibits the same threats like a server-based IM system with regard to the server to which a client is connected. Further, the following threats are conceivable due to the decentralization of servers:

Traffic analysis of own server: An eavesdropper might intercept packets from and to a certain server. He might exploit the fact that Alice's contacts are (typically) distributed over the network and, thus, her presence information is routed to a specific set of servers. This leads to a user-characteristic pattern of incoming and outgoing packets. An eavesdropper might combine the appearance of such a recurring pattern with external observations and conclude Alice's presence even without knowing her physical address in advance.

Attack on a contact's server: The threats to Alice's own server also apply to Bob's server. Since Alice's presence information is transferred to all of her contacts' servers, an attacker can learn about her presence by attacking one of those servers as well.

4. Proposed system

In this section we first state two basic design principles which should be followed by a privacy-aware presence management system. Then we give a brief overview of our system by describing the overall architecture and a typical system usage. Next we describe the requirements on and details of the anonymous communication network which we use. Afterwards we specify and give reasons for the messages which have to be exchanged in our protocol, both, with the registry and with other users. Finally we discuss possible threats.

4.1. Design principles

Having analyzed the threats of disclosure of presence information, we conclude that at least the following two design principles should be followed when conceiving a privacy-aware presence management system:

Concealing communication on the physical layer: Communication on the physical layer might reveal sensitive information to an eavesdropper. It is not sufficient to encrypt the content of messages, but the sender and recipient addresses of messages should be concealed, too. The physical addresses of clients may never be passed to other entities in the system, in order to completely hide the relationship between a user and his IP address. Thus, eavesdropping on the physical layer can only be performed by an adversary who knows the user's IP address from another source but the presence management system.

Keeping secret plain-text presence information: Presence information is not protected, if other entities than the intended communication partners are aware of it. Though other entities have to be involved, they may never learn about any user's presence information in plain-text. Further, the content of communication with (potentially untrusted) entities may not reveal a certain user's presence to an adversary.

4.2. System overview

We propose a presence management system which relies on (1) an anonymous communication network and (2) a DHT as registry. The anonymous communication network is used to conceal all communication on the physical layer as we postulated in our first design principle. The task of the registry is to enable two communication partners to establish a connection using the anonymous communication network. Users store and retrieve so-called *contact information* which constitutes a logical address for anonymous communication to/from the registry. It is *not* used to store *presence and awareness information* directly. The latter is transferred between two users after successful establishment of an anonymous connection link. The main reason for this decoupling is that the registry cannot be trusted due to our second design principle.

The following steps are performed by two users, Alice and Bob, in order to share their respective presence information (see figure 3; nodes with a gray dot are aware of Alice's and Bob's presence):

1) Alice and Bob have to agree on a shared key which is used for storage and retrieval of contact information to/from the registry. Therefore, Alice sends an invitation message to Bob using a communication

channel *outside* of the anonymous communication system, e.g. e-mail. This invitation message has to be sent only once and can be omitted in subsequent sessions.

2) Whenever Bob logs into the system, he publishes his contact information to the registry and tries to retrieve Alice's contact information. He uses the anonymous communication network to obscure his physical address. In this example Bob logs in before Alice, so that he cannot find Alice's contact information.

3) When Alice logs into the system, she performs the same operations as Bob did in step 2. We assume that the registry allows storage of multiple entries with the same key. Alice obtains Bob's recently published contact information.

4) Alice establishes a communication link over the anonymous communication network using Bob's contact information in order to exchange presence information or instant messages.

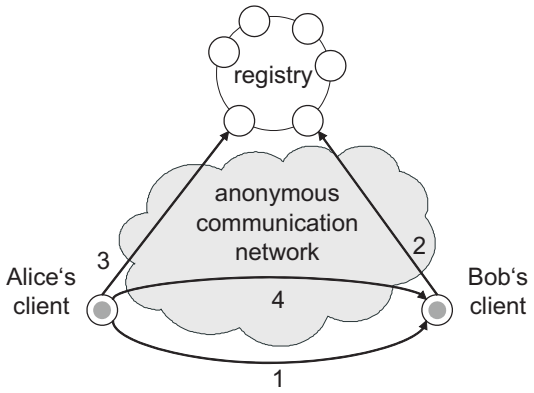

Figure 3. Sharing of presence information.

4.3. Anonymous communication network

We require an anonymous communication network to hide a user's physical sender address when communicating (1) with a publicly known host, e.g. a DHT node, and (2) with another user whose physical address remains unknown. The first requirement is typically fulfilled by all anonymous communication networks. It is the prerequisite for applications like anonymous web surfing. The second requirement implies the possibility to *receive* messages by other participants not only as an answer to previous requests, but unsolicitedly, i.e. to act as a server. Therefore a host must be able to anonymously announce a logical address which clients can specify as recipient address of their requests. This feature is called *responder anonymity* or *location-hidden service*. It is not supported by all anonymous communication networks.

We apply the anonymous communication network Tor [2] to our protocol. Tor is a circuit-based low-latency anonymous communication system. In order to communicate anonymously a user builds a so-called *circuit* to a remote Tor router. Therefore he incrementally negotiates symmetric keys along a path of other Tor routers up to the chosen remote Tor router. Tor provides a list of all running routers on a set of trusted servers. The result is a multi-hop connection in which every node only knows its direct successor and predecessor, but not both, source and destination of the circuit. The user may now anonymously establish connections to public services without being able to be identified as initiator of a request, i.e. circuits provide a user with *sender anonymity*. In contrast to this, the physical recipient address must be known in advance.

In addition to sender anonymity, Tor provides a means for *responder anonymity*. Any user can provide a so-called *location-hidden service* to a (potentially limited) set of clients without revealing his physical address. The protocol of Alice connecting to a hidden service provided by Bob is as follows (see figure 4):

1) During initialization, Bob builds three kinds of circuits: (a) some exit circuits for communication with the Tor directory servers and ordinary web servers, (b) some circuits used as introduction points for the hidden service, and (c) some internal circuits for answering requests to the hidden service. He prompts his introduction points to wait for incoming requests.

2) Bob publishes a so-called rendezvous service descriptor to the Tor directory servers. It contains the physical addresses of his introduction points which are required to access his hidden service.

3) Alice performs initialization by also building three kinds of circuits: (a) some exit circuits, (b) some internal circuits to set up the connection to hidden services, and (c) some circuits which may be used as rendezvous points later on.

4) Alice retrieves the rendezvous service descriptor with the addresses of Bob's introduction points by consulting the Tor directory service.

5) Alice extends one of her internal circuits to connect to one of Bob's introduction points. She requests establishment of a connection using one of her rendezvous points. If required by Bob, she includes an authorization cookie to her request which allows Bob to respond only to authorized requests. This connection is used for just one initialization message.

6) If Bob decides to answer the request (he could also simply drop Alice's last message), he will extend one of his internal circuits to Alice's rendezvous point. Subsequently, messages may be exchanged in both directions, relayed by Alice's rendezvous point.

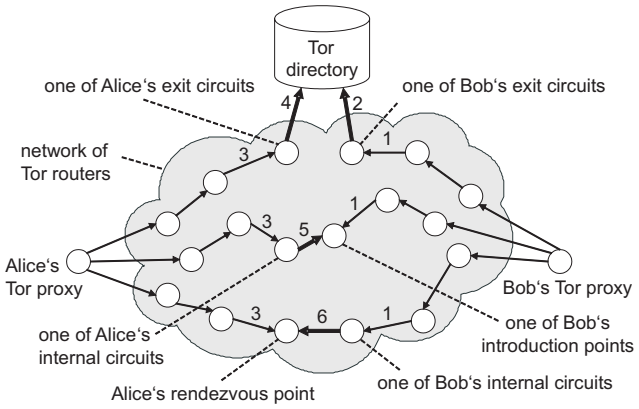

Figure 4. Establishment of a communication link using a location-hidden service in Tor.

After this description one might ask, why Tor is not sufficient to conceal presence of its users by itself. In step 6 Bob can decide whether to answer Alice's request and establish a connection to her rendezvous point, or not. One might argue that if he did not send a message, he would not appear to be present to anybody.

However, the drawback in using the original protocol for a presence management system aimed at concealing presence information resides in steps 2 and 4. In step 2 Bob publishes the rendezvous service descriptor for his hidden service in the Tor directory using the hash value of his public key as publication key. Beyond initial publication he periodically refreshes his entry in the directory, e.g. due to changes concerning his introduction points. In step 4 Alice and everybody else may retrieve Bob's rendezvous service descriptor using the hash value of his public key. Everybody shall be able to make at least an attempt to connect to Bob's hidden service, whether it will be accepted or not. This enables Mallory who is interested in Bob's presence to periodically query the directory for Bob's rendezvous service descriptor. If an up-to-date descriptor is found, Mallory has a hint that Bob is online or has been online recently. If not, he can expect Bob to be offline.

Our protocol makes use of Tor location-hidden services for protecting physical addresses. However, we modify the means of storing and requesting rendezvous service descriptors in our registry as described in the following.

4.4. Specification of protocol messages

Our protocol consists of three types of interactions (cf. figure 3): (1) Alice sends an *invitation message* to Bob (or the other way around), (2) Alice and Bob publish and request their respective *contact information* to/from the registry, and (3) either of them establishes a communication link, so that they can exchange *presence information and instant messages*. First, we specify the operations to store and retrieve contact information, because they justify the invitation message which we describe then. At last we discuss how to exchange presence information and instant messages.

Contact information: Whenever Bob logs into the system, he publishes his contact information to the registry and queries the registry for contact information from Alice. In contrast to the registries in common IM systems, the registry in our protocol cannot perform user authentication. The problem is that Alice's authentication to a potentially untrustworthy registry might unintentionally reveal her presence. In addition to that, Bob cannot rely on the registry to restrict access on his contact information to authorized users. We utilize the registry as a global storage for contact information, but have to perform access control by encrypting contact information only for authorized users.

We deploy a DHT as registry which *guarantees* to find a previously stored entry by providing its key. Our working group has already employed these guarantees in the context of distributed service discovery [5]. Though our security properties do not rely on this distribution, it makes life of an eavesdropper or attacker significantly harder, as there exists no longer a single target. Distribution of nodes among different authorities, regions, and jurisdictions further reduces likelihood of a conspiracy of node operators. The clients using our presence system should not take part in the DHT, because the correlation of being present either in both systems or in none of them might disclose presence of a user, too. Despite of its distributed nature we consider the registry to be a single unit logically.

Registry entries contain Tor rendezvous service descriptors which may be used to establish a connection to their publisher. Using this connection, two users can exchange presence information and send instant messages. Instead of creating one registry entry for *all* contacts of a user, one separate registry entry is generated for *each* contact of a user. The reason for this is that the registry cannot restrict access on registry entries to authorized users. Though other approaches might be imaginable, contact-wise entries allow straightforward revocation of contact list subscriptions. Another advantage is the ability of a user to precisely control propagation of presence information to particular users.

When publishing a registry entry, Bob sends a put request to the registry:

$$B \rightarrow R : X_{AB}, Y_B \qquad (1)$$

X_{AB} is the *registry key* and is known to both, Alice and Bob. Y_B is the *registry value* and can only be created by Bob. Since the put request is sent to a possible untrustworthy DHT node, we have to take into account, that neither key nor value can be kept confidential. As we argued above, we have to assure that neither third parties nor the registry itself may conclude Bob's presence from this put request or the registry entry. Therefore the registry key is based on a shared secret between Alice and Bob, and the registry value is encrypted for Alice.

After publication of his own contact information, Bob sends a get request to the registry, demanding all registry values stored under his and Alice's registry key:

$$B \rightarrow R : X_{AB} \tag{2}$$
$$R \rightarrow B : Y_B \tag{3}$$

In this example, the registry will only return Bob's previously published registry value and no registry value of Alice. Thereby Bob can expect that Alice is not online.

When Alice goes online, she also sends a put request to the registry and demands all registry values for her shared registry key with Bob:

$$A \rightarrow R : X_{AB}, Y_A \tag{4}$$
$$A \rightarrow R : X_{AB} \tag{5}$$
$$R \rightarrow A : Y_B, Y_A \tag{6}$$

Alice receives Bob's contact information Y_B along with her own previously published registry value. Using this entry, Alice may establish an anonymous connection to Bob. We decided to use the same registry key for both registry values to obtain a synchronization point for Alice and Bob and to avoid race conditions. We expect the underlying DHT to allow storage of multiple registry values using the same registry key.

A *registry key* consists of three components which are concatenated and to which a secure hash function is applied:

$$X_{AB} := H\left(K_{AB}, D, I\right) \tag{7}$$

The three components are a shared secret key K_{AB}, a dynamic component D, and a key index I.

Since D and I are publicly available, K_{AB} is the only component which makes X_{AB} unguessable to third parties. It is exchanged between two users in the invitation message.

At a particular time the mere existence of an anonymous registry entry does not reveal presence of the publisher. But an attacker might observe store and retrieve operations of a certain entry over time. He could associate these observations with external studies and relate registry entries to specific users. In this case, presence of the user is disclosed irreversibly. Though the likelihood of such an attack may be low, it increases continuously. Thus, we periodically change the key by using the hour component of time since 1970 of a globally fixed time zone as D. This assumes the local clock to be reasonably accurate and might require periodical synchronization with a global clock. Entries have to be republished whenever the dynamic component has expired. Additionally, during a short transition period two entries should exist in parallel in order to make message latencies or clock divergences transparent.

We cannot rely on the fact that all nodes in the DHT are trustworthy. Thus, it is possible that the registry drops or scrambles entries by random. Without correct registry entries, the presence service is unable to operate correctly. Therefore, we introduce application-based replication by generating multiple replicas of an entry with consecutive *key indices* I. After applying the hash function, the replicas of one registry entry are very likely to be stored on different nodes of the DHT.

We apply a secure hash function to the concatenation of all components rather than encrypting the other two components with this shared key, because (1) Alice and Bob both must be able to exactly reproduce a registry key, and (2) the result length of the hash function matches the length of valid DHT registry keys.

A *registry value* consists of the tuple of rendezvous service descriptor R_B and a timestamp T_B and is encrypted using the shared secret key of Alice and Bob K_{AB} (alternatively signed with Bob's private key and encrypted with Alice's public key):

$$Y_B := E_{AB}\left(R_B, T_B\right) \tag{8}$$

The timestamp is necessary in case that a user has to change his introduction points and consequently his rendezvous service descriptor.

Invitation message: Alice and Bob need to agree on a shared secret key to determine a secret registry key where they put and get their respective contact information. Therefore one of them, in this case Alice, sends an invitation message to the other one using a communication channel outside of the system, e.g. e-mail. Alice signs the shared key with her private key and encrypts the result with Bob's public key:

$$A \rightarrow B : E_B\left(S_A\left(K_{AB}\right)\right) \tag{9}$$

This invitation message is indispensable for our protocol. It is not possible to establish anonymous communication based on publicly known information. One might argue that Bob could store one half of a Diffie-Hellman handshake under a publicly known key in

the DHT. Alice could then request this information and complete the Diffie-Hellman handshake to obtain a shared secret key with Bob. The drawback is that Bob would have to refresh his DHT entry periodically to ensure its persistence, and that every put request would be a sign of his presence. Thus, such an approach would be conflicting with our absolute requirement to conceal presence information.

The invitation message should not be confused with an authentication message commonly used in other IM systems. Bob can revoke Alice's authorization to learn about his presence at any time by stopping to publish contact information for her.

Presence information and instant messages: After establishment of an anonymous communication link Alice and Bob can exchange presence information and instant messages. All these messages are encrypted using their shared secret key K_{AB} (alternatively with a previously agreed session key):

$$A \rightarrow B : E_{AB}\left(PI_A\right) \tag{10}$$

4.5. Possible threats

Though we have designed our protocol aiming to resist possible threats, there are some dangerous situations in which our system either sustains denial of service, or provides only limited protection against disclosure of presence:

Sparse usage of the presence system: The fewer the number of participants in our system is, the higher is the probability of disclosure of their presence. If only a few entries were stored in the registry, an attacker could make a guess to whom they belong and deduce the publisher's presence. To prevent this we might store and request dummy entries. This requires an analysis of typical storage and retrieval patterns to make dummy entries indistinguishable from real ones.

Infrequent usage of the anonymous communication system: The same phenomenon applies to the number of users in the anonymous communication network. An eavesdropper who is able to observe a large number of nodes in an infrequently used network might conclude a user's presence as well as the actual recipients of messages. Therefore, we rely on a separate anonymous communication system which is used for other applications at the same time.

Disclosure at the edge of the anonymous communication system: Communication between a client and the first node in a circuit of the anonymous communication network reveals the client's IP address. An eavesdropper who is aware of this IP address may infer the user's presence whenever he observes packets from or to this IP address. Unfortunately, it is impossible to hide communication on the physical layer. As a preventive measure we hide the connection between a certain user and his IP address, so that an attacker will never learn about a certain user's IP address from the presence management system.

Attack on the anonymous communication system: Attackers may attempt to corrupt nodes of the anonymous communication system in order to reveal the actual sender and receiver of a message. However, users choose the nodes for creating a circuit dynamically. Thus, an attacker is required to corrupt a large number of nodes for a successful attack on a complete circuit. For an extensive threat analysis for Tor see [2].

Infiltration of the registry: An attacker might join the DHT which we use as registry and become responsible for a part of its entries. A more powerful attacker might even perform a Sybil attack as described by Douceur in [4] in order to become responsible for a significant fraction of the stored entries. Then he could drop or modify entries, delay storage requests, give false results to retrieve operations, etc. However, the illegibility of entries makes it impossible for an attacker to perform a directed attack. In the worst case, this attack leads to refusal of service, because users do not obtain correct contact information. But replication of entries on application level makes a successful attack very unlikely. Further, clients may switch between access points of the DHT to decrease the probability of communicating with a corrupt node.

Denial of service attack on the registry: Access to the DHT is not limited, because we cannot perform the authentication which would be necessary. An attacker could perform a denial of service attack by putting a huge number of arbitrary registry entries into the DHT. Possible counteractive measures are extending the resources of the DHT or making store operations computationally expensive for the clients.

5. Related work

There do exist deployed approaches to secure IM communication. However, most approaches focus on encryption of instant messages or authentication of users and not on confidentiality of presence information.

Cerulean Studios (`http://www.ceruleanstudios.com/`) have developed SecureIM, an encryption protocol for instant messages on top of AIM and ICQ. It is based on the Diffie-Hellman key agreement in combination with the Blowfish symmetric encryption algorithm (details are described by Murphy in [7]). SecureIM provides message integrity and confidentiality,

but neither authentication of communication partners nor confidentiality of presence information.

VeriSign offers a personal certificate for signing and encrypting text messages within the AIM network (`http://www.verisign.com/stellent/groups/public/documents/white_paper/005324.pdf`). It provides authentication, integrity, and confidentiality of IM, but no privacy of presence information.

The wija project (`http://www.media-art-online.org/wija/`) is an IM application based on the Jabber protocol. It allows users to encrypt and sign instant messages and to sign presence information. Albeit signing of presence information prevents from impersonating another user, it does not conceal presence at all.

The VoIP protocol Skype (`http://www.skype.com/`) uses a peer-to-peer network to store and retrieve their user's presence information, too. Baset and Schulzrinne [1] have performed a study of Skype network traffic revealing the architecture of Skype. However, they employ a central login server for authentication which faces the same threats concerning revelation of presence as any other centralized IM system.

Literature discusses security and privacy issues from multiple perspectives:

Patil and Kobsa [8] study the tension between awareness and privacy in the context of collaborative work. They investigate and compare mechanisms to protect privacy in current IM clients. Their research can be considered rather on a socio-technical than on an engineering level, as they do not consider security on the network layer and of the IM server.

Mannan and van Oorschot [6] analyze security threats like insecure connections, denial of service attacks, or impersonation, and discuss in how far the well-known messaging systems fail to provide adequate solutions. However, they do not consider disclosure of presence information.

To sum up, neither deployed systems nor literature focus on the problem of disclosure of presence information. Confidentiality of messages and authentication of users are certainly an important building block in the attempt to make IM more secure. But we argue that hiding presence from unauthorized entities is also an essential part of making IM a privacy-aware communication medium and therefore must be considered.

6. Conclusion

We proposed an IM system designed to be robust against attacks disclosing a user's presence. In contrast to existing systems, it stores presence information in a registry in a way that is only detectable and applicable for intended users and even not comprehensible for the registry itself. We use a DHT as registry and apply an anonymous communication network to protect the physical addresses of both, senders and receivers.

We have implemented our protocol as a proof of concept and published the protocol details of our prototype as technical report [3]. We refer to this reference for technical details of our prototype.

As next step we continue the work on the prototype of our system to study its performance and typical communication patterns. We will use measured data to improve our protocol and create dummy entries which conceal the users' presence information even in a sparsely populated network.

Apart from that we aim to formalize our protocol and perform a formal protocol verification on it.

Finally, we will investigate further applications of our presence management system, as IM is only one possible application of presence information.

References

[1] S. A. Baset and H. Schulzrinne. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. Technical report, Department of Computer Science, Columbia University, New York, NY, Dec. 2004.

[2] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, pages 303–320, 2004.

[3] M. Dorsch, M. Grote, K. Hildebrandt, M. Röglinger, M. Sehr, C. Wilms, K. Loesing, and G. Wirtz. Concealing presence information in Instant Messaging Systems—Protocol Specification. Technical report, Otto-Friedrich-Universität Bamberg, Feb. 2005.

[4] J. R. Douceur. The Sybil Attack. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS '02), MIT Faculty Club, Cambridge, MA, USA*, March 2002.

[5] S. Kaffille, K. Loesing, and G. Wirtz. Distributed Service Discovery with Guarantees in Peer-to-Peer Networks using Distributed Hashtables. In *Proceedings of PDPTA '05*, volume II, pages 578–584, June 2005.

[6] M. Mannan and P. C. van Oorschot. Secure public Instant Messaging: A survey. In *Proceedings of the 2nd Annual Conference on Privacy, Security and Trust (PST04)*, pages 69–77, Oct. 2004.

[7] M. D. Murphy. Instant Message Security—Analysis of Cerulean Studios' Trillian Application. Technical report, SANS Institute, June 2003.

[8] S. Patil and A. Kobsa. Preserving Privacy in Awareness Systems. In *Wissen in Aktion*, pages 119–130, 2004.

[9] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Core. RFC 3920, Oct. 2004.

[10] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence. RFC 3921, Oct. 2004.