Detecting Trustworthy Real-Time Communications using a Web-of-Trust

Jan Seedorf, Nico d'Heureuse, Saverio Niccolini
NEC Laboratories Europe
Heidelberg, Germany
Email: lastname@nw.neclab.eu

Marco Cornolti
University of Pisa
Pisa, Italy
Email: cornolti@cli.di.unipi.it

***Abstract*—Voice-over-IP protocols (e.g., SIP) are vulnerable to many types of attacks. One core challenge in preventing VoIP attacks is to assess the trustworthiness of the caller's identity. Further, spoofing attacks must be prevented by verifying that the call has been initiated by the user belonging to the caller's identity. In this paper, we propose to adapt a Web-of-Trust model to real-time communication in order to assess the trustworthiness of incoming VoIP calls based on the social relationships among users. We present the design of a system which is capable of cryptographically verifying trust chains associated with VoIP users in *real-time*, i.e., with minimal overhead during the regular processing of signaling messages. We highlight the benefits of such a system as well as its limitations, discuss open issues, and finally present an evaluation of the proposed approach based on a prototypical implementation. Our results show that indeed *real-time* cryptographic verification of trust chains among users is feasible for VoIP communications.**

I. Introduction

Voice-over-IP (VoIP) signaling protocols, e.g., the Session Initiation Protocol (SIP) [1], are vulnerable to many types of attacks. Examples are interruption of service attacks (i.e., Denial of Service, DoS) and social attacks (e.g., SPam over Internet Telephony, SPIT). In order to prevent these attacks it is necessary to estimate if an incoming signaling message (e.g., SIP `INVITE`) is malicious or not. One core challenge is to assess the trustworthiness of the caller's identity with respect to sending malicious (or unsolicited) messages. Moreover, it is important to verify that the call has been initiated by the user belonging to the caller's identity in a SIP-message (the `SIP-URI` in the `From-Header`). This problem is not easy to solve as VoIP-identities (e.g., the SIP-URI) can be spoofed easily. This makes the detection of social attacks like SPIT and unsolicited communications in general a sophisticated problem. But also DoS attacks are easier to protect against if identity-spoofing can be prevented.

In this paper we address these problems: We propose to adapt a Web-of-Trust (WoT) model to *real-time communications*. Our approach uses the trust relationships between users in order to detect if an incoming signaling message was really send by the user belonging to the caller-identity inherent in the message or not, and further to estimate if this user/identity is trustworthy, i.e., not sending malicious or unsolicited messages[1].

[1]We assume the reader to be familiar with asymmetric cryptography and the general scheme of a Web-of-Trust, as used, e.g., by PGP [2].

To be applicable to real-time communications such as VoIP, the trust relationships between users in a Web-of-Trust have to be known to the callee either prior to the call or must be derived at the time of the call *in real-time*. Otherwise, the downloading of certificates and cryptographic verification of trust paths is likely to delay communications (e.g., a VoIP call) too much to be acceptable to users. There exist algorithms for real-time derivation of WoT trust chains [3]. However, existing usage models of a Web-of-Trust are different from our approach and do not enable *real-time verification* of trust chains. Consider the common usage of the PGP [2] Web-of-Trust in non real-time communications such as email: PGP key-servers (e.g., [4]) merely offer storage of users' certificates. A user which receives a signed email usually has a direct trust relationship with the sender of the email. Otherwise, the user has to retrieve the certificate of the sender and must then verify the trust chain between itself and the sender of the email (which most probably results in fetching more certificates). While there exist services for computing the trust path between sender and receiver [3], such tools are not integrated in email clients and the cryptographic verification of the trust chain can only be done *after* receiving an email. This renders current usage of Web-of-Trust models infeasible for real-time communications.

The main contribution of this paper is to adapt a Web-of-Trust model to real-time communications such as VoIP. We adapt existing algorithms to derive a trust path between a VoIP caller and callee in *real-time* and additionally propose novel techniques to achieve cryptographic pre-verification of the trust chain. In summary, our system enables to combine a decentralized trust model, cryptographic identity assertion, and the assessment of the trustworthiness of VoIP users to be used in *real-time communications*.

The rest of this paper is organised as follows. We discuss existing solutions and their drawbacks in section II. In section III we present the rationale for our approach and describe the proposed system in detail. Subsequentially, we present the results of our prototypical implementation (section IV). Further, we discuss potential limitations of our proposal in section V. Finally, we relate our work to other approaches (section VI) and conclude with a summary (section VII).

Throughout this paper and in our prototypical implementation (see section IV) we exemplify our proposed scheme for

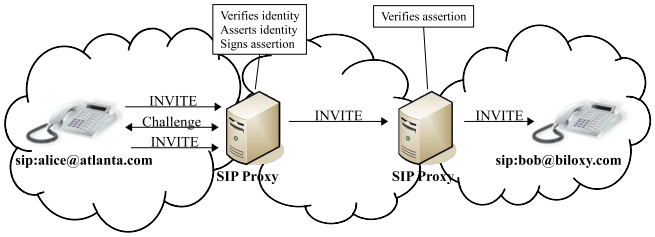

Fig. 1: SIP strong identity approach (RFC 4474)

applying a WoT model to real-time communications with SIP [1] and PGP [2]. Our approach is, however, general in nature and in principle applicable to any kind of signaling protocol for setting up and managing real-time communication session and any WoT infrastructure. Our approach only requires a (cryptographic) WoT among users which is instantiated through protocols and key-servers.

II. Background

The research and standardisation communities have realised the problem of SIP identity spoofing. Many currently proposed mechanisms for estimating if an incoming message was sent by the user belonging to the caller-identity rely on so-called *strong identities* [5]: If an identity is signed by a centralised authority which is trusted by the receiving end, messages received from this identity are believed to be non-malicious. For SIP, the identity of the caller is the `SIP-URI` in the `From-Header` of a SIP `Invite` message, and it has been proposed to have this identity signed by the domain of the caller (RFC 4474 [6]). When the caller places a call, the domain challenges the caller with an authentication request. Only after proper authentication the domain will sign the outgoing message. When receiving a call, the proxy of the callee verifies the signature with the public key of the caller's domain [6]. Figure 1 shows this approach in the context of the classic SIP trapezoid.

However, there are several problems with such an approach:

1) There must exist a mechanism to exchange public keys between domains.
2) A mechanism is necessary to verify the binding of a retrieved public key to the domain of the sender.
3) The cryptographic assertion of an identity is merely technical: the proxy's assertion does not regard the trustworthiness of the signed identity but instead in most cases only assesses the possession of the private part of an asymmetric cryptographic key pair.

The common practice (also suggested in [6]) to solve the first two problems is to use a Public Key Infrastructure (PKI). A hierarchy of Certificate Authorities (CAs) is used to establish a cryptographically verifiable trust chain between any two entities in the system. This implies that a central authority on top of this CA-hierarchy, the Root-CA, is trusted by all entities in the system. The Root-CA is the basic building block for all trust related to assertions of identities in such a system. Thus, trust is centralized.

More importantly, with current approaches there is no trustworthiness associated with the process of identity assertion. It is important to realise that an identity asserted by its domain can still misbehave, e.g., send malicious messages (which are correctly signed by its outgoing SIP-proxy). The reason is that it is not specified what the basis for asserting a SIP-identity is. In other words, what is considered an *authenticated* user may vary between different domains. Some domains might sign identities without proper checking while others require strong identity proofs (e.g., a copy of a passport and a signature send via regular mail) or use cryptographically more secure devices like smartcards for identity assertion. A receiving domain, however, cannot assess the quality of the assertion-policy in foreign (sending) domains. In addition, user terminals might have been compromised by malware (e.g., botnets, worms) which is capable of accessing the user's credentials. In this case, a SIP user agent can perfectly authenticate itself with its domain while still sending malicious messages.

In summary, approaches based on so-called *strong identites* [6] are only as good as the verification/assertion policy of the caller's domain. However, even if strong authentication mechanisms are used by the caller's domain, there is no trustwortiness associated with signed outgoing messages.

To overcome these limitations of existing approaches, this paper proposes a different mechanism for real-time verification of signed identities: a Web-of-Trust combined with advanced preprocessing mechanisms. In short, the advantages of the novel approach are that it does not rely on a central trust authority, exploits social relationships between end-users, and additionally associates trustworthiness with signalling messages by using cryptographic signatures. All this is done in real-time to enable a timely decision on how to process a message.

III. Our Approach: Web-of-Trust for Detecting Solicited VoIP Calls

The main idea behind our approach is to use a Web-of-Trust to verify the binding of a user to an identity and additionally to verify the trustworthiness of this user/identity. The rationale behind this approach is that the social relationships among users in a VoIP network are very beneficial for assessing the trustworthiness of identities. Users can assess the trustworthiness of identities based on the perceived interaction in the past, i.e., through received phone calls. Using a Web-of-Trust adds the necessary cryptographic primitives so that users can express their trust by signing other users' identities.

Compared to using a Public Key Infrastructure (PKI) for retrieving and verifying public keys of the trust chain between caller and callee, our approach has the advantage of *decentralized trust*: In a Web-of-Trust, trust is not centralized but distributed among users. Thus, there is no single root authority which has to be trusted by all participants in the system (as is the case in a PKI, see section II).

A. Assumptions and Definitions

We assume that a Web-of-Trust infrastructure exists (e.g., the publicly available protocols used by PGP [7]) where redundant servers (similar to PGP key-servers [4]) store certificates of users. The WoT key-servers update each other's key-database with a special protocol amongst themselves. Users can upload certificates which have been signed by other users to these servers. Each public/private key pair has a unique *key-ID* in the system. The binding between a user-identity (*user-ID*) and such a key-ID is signed by users in certificates.

A key assumption for our system is that users sign other identities in the system only if they have had positive (i.e., non-malicious) interactions with this identity in the past. This means that users can express how trustworthy they regard other users to be (with respect to sending malicious/unsolicited messages) by signing the certificates of these other users in the WoT. Note that in a PGP-like WoT, key-servers are not trusted and any user can store any certificate on these servers. Specifically, the signatures of the certificates are not verified by the key-servers. This task is left to the users. Thus, to achieve real-time computation and verification of a *trust chain* between two users this approach is not feasible per se and needs to be modified (which is a core contribution of our work).

A trust path (or *trust chain*) between two users u_x, u_l in a WoT exists if there is a chain of trust relationships between users in the WoT such that u_x *trusts* u_1 *trusts* u_2 ... u_{l-2} *trusts* u_{l-1} *trusts* u_l. The length l of a trust path is the number of intermediate entities between the two users including u_l. Note that there can be various different trust paths between two users. We define the minimal trust path length n between two users as the minimal length among all trust paths between these two users.

The underlying assumption is that in the average case the shorter the trust path, the less is the probability that the trust chain between two users has been infiltrated by an attacker (e.g., by deluding a legitimate user to trust an attacker). Adopting the definition of a trust path to real-time communications such as VoIP and expressing it cryptographically, we say that there is a trust chain of length n between callee and caller, if in the WoT the minimal trust path is such that $[keyID(callee)]$ *signed* $[keyID_1, userID_1]$ *signed* $[keyID_2, userID_2]$ ... $[keyID_{n-1}, userID_{n-1}]$ *signed* $[keyID(caller), userID_{caller}]$. Since the receiving party (i.e., the callee) wants to know how trustworthy the calling identity is, we are only interesting in trust paths going from callee to caller (and not vice-versa).

In the rest of this paper, we assume that the callee is protected by its upstream SIP proxy which analyzes incoming messages (although in principle also the callee itself could analyze incoming messages, we regard protection by upstream entities to be the more realistic case). To assess the trustworthiness of incoming calls, the callee's proxy can either invoke a trust path calculation from a third entity or such a computation can be performed by the SIP proxy itself. Depending on the found trust path length, the proxy conducts further processing of SIP messages (e.g., forwarding to the callee for very short trust paths, conducting further security tests for medium length trust paths, and potentially directly forwarding the call to the mailbox of the callee for long trust paths). The rationale is that the longer the trust path, the less reliable is the (indirect) signature chain of the caller's identity to the callee. Overall, our approach potentially enables identifying trustworthy identities (*whitelisting*) but not the detection of badly behaving identities. In other words, if no or only a long trust path can be found, our approach cannot assess the trustworthiness of the caller. We therefore assume that our WoT scheme is used in conjunction with a holistic VoIP protection solution as presented in [8]. If a short trust path is found, however, we assume this to be an indication of a non-malicious call (and depending on system settings, e.g., the set trust path threshold, the call might directly be forwarded to the callee, or this fact might influence an overall security score computed by the callee's SIP proxy).

B. Real-time Derivation and Cryptographic Verification of Trust Paths

Our approach relies on the computation of trust paths between callee and caller. In principle, an attacker can fake trust paths by uploading a certificate to a key-server which binds its user-ID to its key-ID, but with an invalid signature apparently from a legitimate user. This results in falsely marking the attacker as trustworthy. To prevent such trust path forging, any trust chain in a WoT must be cryptographically verified. A novel feature of our approach (compared to a web-of-trust used in email systems) is finding the trust path between two identities as well as cryptographically verifying the complete trust chain *in real-time*. To achieve *derivation* and *cryptographic verification* of a trust path in real-time, we propose a specialized WoT key-server as part of the key-server federation which operates normally in the key-server network but additionally does the following:

1) For each certificate it receives either from a user or from another server it verifies the signatures in the certificate. The server can do this because as a key-server it is in possession of the corresponding public keys to the private keys which signed the certificate[2].
2) It periodically publishes a file which contains in a compressed, machine-readable format the trust relationships between all the certificates it stores of which it has verified the signatures. These trust relationships are verified by the server in the sense that all cryptographic signatures responding to the trust relationships inherent in the file have been verified. The file published by the server is structured in such a way that trust paths between any two identities can be computed in *real-*

[2]In case the key-server is missing a public key necessary to verify a newly received certificate, it tries to retrieve this key from other servers in the key-server network. If it cannot obtain the necessary certificate, the corresponding signature is considered as unverified by the key-server.

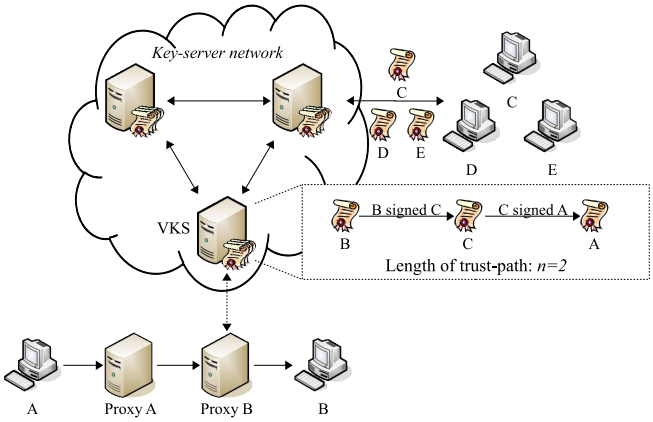

Fig. 2: System Architecture of Proposed Approach

time using this file[3].

This special *Verifying Key-Server (VKS)* is trusted by the callee's SIP-proxy. However, in general trust is decentralized in our system: The signatures in certificates are not based on a centralized CA hierarchy. The VKS fetches publicly available certificates from other key-servers and verifies the signatures. The callee's proxy merely trusts that the VKS performs this signature verification correctly. The authorities which signed certificates (i.e., the users) are distributed in our system.

C. System Architecture

The general procedure of our approach is as follows: A user signs a signaling message (e.g., SIP *INVITE*) with its private key and appends (e.g., via S/MIME) a self-signed certificate containing the corresponding public key[4]. The proxy of the callee receives the message with the attached self-signed certificate of the caller. To check that the message was really sent by a user who is in possession of the corresponding private key, the callee's proxy verifies the message-signature using the public key from the attached certificate. To further check that the message was sent by the user belonging to the identity inherent in the message and to check that the identity of the caller is trustworthy, the proxy checks that the SIP-URI in the *From-header* of the message corresponds to the SIP-URI in the attached certificate and invokes an algorithm which delivers a trust path between the caller and the callee. The input to this algorithm is the certificate attached to the received SIP-message as well as the receiver's certificate. Since the certificate attached to the message binds the caller's user-ID (e.g., his/her SIP-URI) to its key-ID, attacker's cannot spoof SIP identities as long as they are not in possession of the corresponding private key. Depending on the length of the trust path between caller and callee, the proxy may invoke further steps to check the trustworthiness of the message, e.g., by applying other tests on the message [8]. The computation of trust paths is either done by the callee's proxy or by a third party.

Figure 2 shows the general architecture of the proposed solution. As described previously, the VKS in the figure is a specialized key-server which not only stores keys/certificates but additionally verifies the signatures. Also, it periodically computes a trust-relationship file which contains the trust relationships between all verified certificates by the server. Using this file, it can offer the service of computing a trust chain between two identities. As an alternative, it may also publish this trust-relationship file to be used by others. In the example, however, a proxy which receives a message from A to B uses the services offered by the VKS to find out the trust path length n between B (callee) and A (caller). The server detects that B trusts an identity C which in turn trusts identity A. Since all signatures in the corresponding certificates of A, B, and C have been verified by the VKS a priori during their upload, the trust path is not only computed but verified as well.

D. Detailed Scheme and Message Flow

We now describe the cryptographic procedures and message flows of our proposed scheme in detail. Assume a caller (with SIP-URI s) is trying to establish a SIP-based [1] VoIP call with a callee (with SIP-URI r). Assume further that the callee is protected by its proxy P_r which uses a special, trusted WoT key-server VKS which offers real-time derivation of pre-verified trust paths. Then the following steps are executed:

1) P_r receives a SIP *INVITE* message m which is signed by s with its private key $k_{pri}(s)$, attached is a self-signed certificate from s:
   $s \rightarrow P_r: \ \{m\}_{sign(k_{pri}(s))}, \{k_{pub}(s), s\}_{sign(k_{pri}(s))}$
2) To protect r, P_r needs to find out if the certificate attached to m really belongs to the SIP-URI in the From-header of m (i.e., s). Therefore P_r verifies the signature using the public key from the certificate:
   $P_r: \ \{m\}_{verify(k_{pub}(s))}$
   If the signature is valid, P_r knows that whoever sent m was in possession of the private key $k_{pri}(s)$. Otherwise (i.e., if the signature is not valid), P_r rejects m.
3) Additionally, P_r wants to know how trustworthy the identity s is. Thus, P_r computes the key-ID for s, $k_{ID}(s)$, by hashing the certificate of s with a specified hash function h, and then sends this key-ID as well as the SIP-URI of the callee, r, to VKS:
   $P_r: \ k_{ID}(s) = h\left(\{k_{pub}(s), s\}_{sign(k_{pri}(s))}\right)$
   $P_r \rightarrow VKS: \ \{k_{ID}(s)\}, \{r\}$
4) VKS computes the length of the minimal trust path in the WoT between r and s, n. If there is no trust path, n is 0. VKS returns n to P_r:
   $VKS \rightarrow P_r: \ n$
5) Depending on n, P_r conducts further processing of m.

[3]Note that there already exists a file format for storing trust paths in a compressed way which is created by certain PGP key servers (.wot-file format [9]). However, regular PGP key servers do not verify the signatures before creating this trust path file. Thus, existing .wot files contain *unverified* trust paths, rendering them not useful for real-time communications.

[4]The concrete way of signing a message is orthogonal to our method as long as the signature is unique for every message to prevent replay attacks (RFC 4474 [6] specifies such signatures for SIP messages).

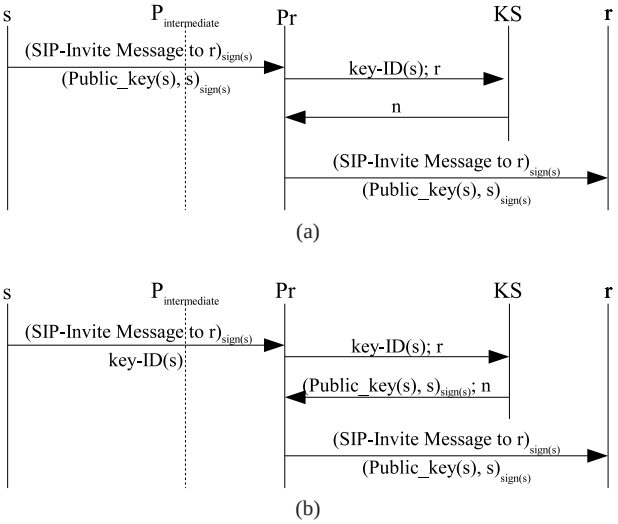

Fig. 3: Message Flow in Proposed Scheme: Sender attaching its certificate (a), Sender attaching its Key-ID (b)

Figure 3a shows an example for a possible message flow using the proposed scheme for establishing a VoIP call. Again, the callee's SIP proxy server P_r uses an external service provided by a special WoT VKS. It is also possible that the proxy does the trust path computation itself (in this case P_r and KS could either be located in the same device/entity or P_r would frequently fetch a .wot file published by the VKS). Furthermore, although also not shown in the figure, it is also possible that instead of forwarding n to the callee the proxy may take different actions on m depending on the length of the trust chain.

As an alternative (shown in figure 3b), the caller may only append its $key - ID$ (instead of its self-signed certificate) to the message. In this case, the proxy of the callee passes the key-ID on to the key-server which returns not only the length of the minimal trust path but also the certificate of the caller[5] (see figure 3b). This variation has the advantage that the SIP *INVITE* message does not increase much compared to a regular SIP message (except for the key-ID in a new SIP-header). Appending the certificate with each SIP *INVITE* message would increase each such message by the size of a certificate (e.g., up to 4kb with 4096-bit RSA keys). On the other hand, as a disadvantage in this case (i.e., only appending the key-ID), it is not possible for the proxy to verify the signature of the message instantly (i.e., before passing on information to the key-server). Instead, the proxy can only verify the signature *after* having received n and the certificate of s from the key-server. This makes the proxy more susceptible to Denial-of-Service (DoS) attacks: an attacker could send bogus message with invalid signatures in order to stress the computational power of KS. In the scheme depicted in figure 3a, a cookie mechanism similar to the one used in IPSec [10] could protect P_r against similar DoS attacks with invalid signatures. Note that replay attacks are not possible in either case since the signature is unique for every SIP message.

[5]Delivering the corresponding certificate for a certain key-ID is the basic primitive provided by any regular WoT key server.

IV. Evaluation

We implemented our approach in a prototype which uses the existing PGP [2] WoT and the corresponding infrastructure (i.e., key-servers [4], protocols [7], etc.). Using the PGP WoT allows us to use the largest publicly available and cryptographically secured WoT for our experiments. Currently the PGP WoT is used to bind email addresses as identities to public keys. In principle, however, this infrastructure would also allow to bind SIP-URIs (which are very similar to email addresses) to public keys. To evaluate our approach with a real WoT, we treat the identities in the existing PGP WoT as SIP-URIs instead of email addresses and we regard the corresponding signatures in the WoT as according to our scheme (i.e., not only binding an identity to a public key but also expressing trust with respect to that identity sending malicious messages).

Our prototype sends signed SIP *INVITE* messages to a modified SIP proxy which then verifies the signatures and calculates the trust path length. For signing the messages we follow the procedure depicted in figure 3b (see section III-D). We compute the signature for each message as specified in RFC 4474 [6] and also use the SIP *Identity* header for the signature and the SIP *Identity-info* header for the key-ID as defined in [6]. Note, however, that we only use the syntax from [6] and that our approach is different: instead of transmitting the identity of a PKI certificate authority we convey a Web-of-Trust key-ID in the *Identity-info* header.

The OpenCDK library [11] is used for all PGP related functions. All experiments were performed on an *Athlon 2800 XP* system with 2GB of RAM running Linux 2.6. All software libraries and algorithms were written in C/C++ in order to assess a fast and realistic implementation.

In order to verify a signature, the proxy has to be in possession of the sender's public key. If this key does not exist in the proxy's local cache, the key is downloaded from a PGP key server. Our experiments showed that our system needs in average $0.6ms$ to check if a key is already present in the local key cache. If the key does not exist in the cache, it is downloaded within ~$100ms$. After the download, the signature verification is performed in $3.2ms$. The exact measurement results are shown in table I.

To evaluate the path search performance, we implemented a double sided breadth-search-first (BSF) algorithm [3] in our SIP proxy. As input graph for the search algorithm we use snapshots of the PGP WoT in the *.wot* file format[6] [9]. These files contain all Key-IDs and the trust relationships (who signed whom) between the PGP users of the largest cluster – the so called "strong set". The restriction to the strong set implies that between any two identities a trust path

[6]We assume that for each trust path in the .wot file the signatures have been pre-verified. Specifically, we treat .wot files downloaded from real PGP key-servers as if they were published from a trusted VKS as described in section III.

	Cache check	Key retrieval	Signature verification
$\mu[ms]$	0,62	98,39	3,19
σ	0,08	47,09	0,40

TABLE I: PGP measurements (15,000 repetitions): average time μ, standard deviation σ

Name	# identities	# signatures	Date of Snapshot
WOT25k	25,487	230,445	25-Feb-2005
WOT33k	33,050	328,912	01-Jun-2006
WOT40k	40,480	405,289	15-Nov-2008

TABLE II: WoT snapshots used for analysis

always exists. To analyze the influence of different WoT sizes on the path search performance we used different snapshots: the oldest .wot file we found contains approximately 25k identities (230k signatures); the newest one (at the time of the evaluation) contains 40k identities (400k signatures). Finally we chose a third *.wot* file containing 33k identities (328k signatures). The different WoT snapshots are summarized in table II.

For each WoT we randomly selected 1,200,000 source-destination key pairs and executed the path search algorithm for each of those pairs. Figure 4 shows the distribution of the path lengths found. The distribution is very similar for the different WoT sizes. The average trust path length is nearly identical for the three WoTs (5.99, 6.01, and $5,97$ for WOT25k, WOT33k and WOT40k, respectively). Furthermore, independent of the WoT size, 90% of all paths found in our experiments have a legth of 8 hops or less; 99% of the paths consit of at most 12 hops.

Figure 5 shows the time our implementation within the SIP proxy needs to find a path of a certain length as well as the 95% confidence intervall. As one expects for a BSF algorithm, the time increases exponentially with the path length. At the same time, the results suggest that (at least for the WoT sizes we evaluated) the search time increases linearly with the number of signatures in the WoT.

In a real world application the trust between two identities decreases with the trust path length. I.e., from the user's point of view, a long trust path does not imply much

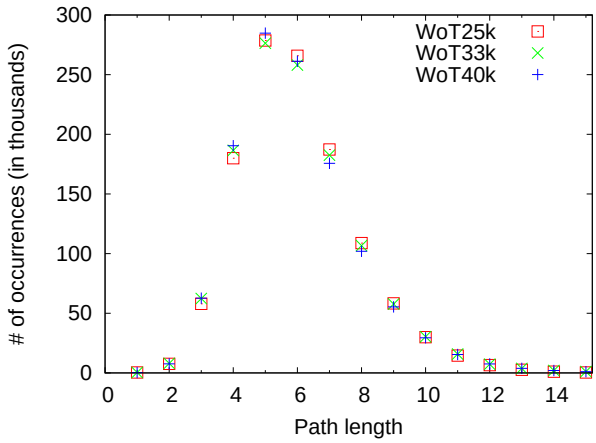

Fig. 4: Occurrences of trust path lengths when randomly selecting *(Source,Destination)*-pairs

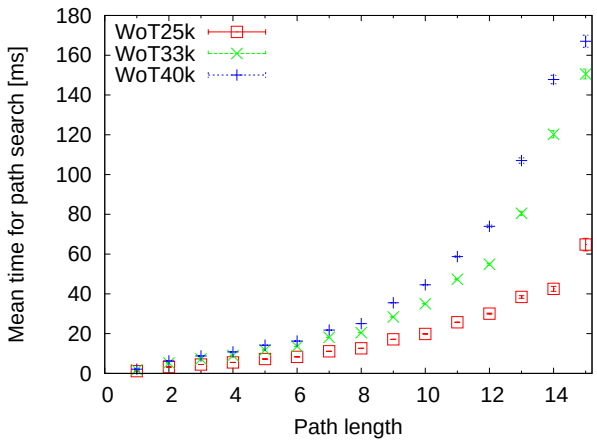

Fig. 5: Duration for path finding depending on path lengths

trustworthiness for messages from the corresponding identity. Thus, it is reasonable to have the path search algorithm only examine trust paths up to a certain length[7].

When comparing the times needed for key download ($\sim 100ms$), signature verification ($\sim 3ms$) and path search (5-$10ms$ for short path lengths), the time for key download is the largest one by at least one order of magnitude. Consequently, we conclude that the usage of WoT based signatures for real-time communication is only feasible if the required public keys for signature verification have been pre-fetched. Our evaluation clearly demonstrates that if the keys first have to be downloaded, the overall time until the completion of the signature verification is too long for most real-time communication applications. Thus, the public key has either to be included in the signaling message (as shown in figure 3a) – which would increase the overall bandwidth required – or the key verification cannot be done by regular signaling entities, but must instead be performed by a specialized service which has the keys pre-fetched.

Our results show that even a simple BSF path search algorithm is fast enough to calculate the trust path length in a reasonable amount of time for small WoTs and short trust path lengths. We could not perform experiments on larger real world WoTs since, to our knowledge, the PGP WoT we used is the largest one available. On the other hand, we regard the limitation to short trust paths to be acceptable since the trust between two users decreases with the path length, rendering long trust paths less useful.

V. Discussion

We propose to adopt a Web-of-Trust model to real-time communication where users can express their trust in identities cryptographically. To work in practice, our approach relies on users behaving correctly. If users are not careful in signing other identities, the overall system becomes less useful because it can be infiltrated by attackers who trick careless users into signing their identities. In general, we

[7]Determining the actual threshold for trust path computation as well as deriving a meaningful trust value from the trust path length is one of our current research projects.

envision two options for users to express that an identity from which they received a call is trustworthy: Either explicitly, e.g., by pressing a special button on the callee's phone, or implicitly, e.g., automatically based on statistics like the combination of call duration and call frequency.

Our system uses pre-verification of trust chains. The disadvantage of such pre-computation is, however, that the key database will always be slightly out of date. This implies that recently uploaded certificates will not necessarily be considered for assessing incoming calls. We regard this not to be a significant problem because it can potentially only result in further message processing by the callee's proxy but not in any kind of attack.

Related to this timeliness of .wot-files is signature revocation. Revocation of certificates and signatures is a challenge in any large system which relies on asymmetric cryptography. For our WoT approach, we assume that users are able to revoke signatures. Further, we assume that such revocations of signatures can be uploaded to key-servers and are taken into account by a VKS when computing trust paths (i.e., if a signature has been revoked the VKS does not consider it anymore in computing trust paths). Thus, if a formerly trustworthy identity suddenly starts to send malicious messages (e.g., because a VoIP terminal has been infiltrated by malicious software), users which have signed this identity are assumed to revoke their signatures for this identity as soon as they realize that the identity is not trustworthy anymore. But as long as not all signatures for such an infiltrated identity have been removed, this identity is still connected to the WoT and may have a short trust path to certain callees.

However, experience with email-worms shows that such malware usually tries to spread using the address book of infected identities. Here our approach clearly has advantages: presumably exactly the identities in the address book of a user are the ones which have trust relationships and can thus revoke certificates. In addition, short-lived signatures could limit the effect of infected hosts. Furthermore, we believe that other protection mechanisms (such a blacklisting or a holistic approach as suggested in [8]) should be used in conjunction with our proposed WoT approach in order to protect against sophisticated threats as botnets.

We showed that a simple BSF path search algorithm performs well for real-time trust path computation. We expect this to be true due to the small world properties of the PGP WoT networks we used. However, we only considered relatively small networks (with respect to real-world VoIP networks) with medium path lengths in our experiments as these are the largest WoT networks publicly available today. The scalability of the approach to very large WoTs still has to be investigated. A lot of research has been done in the area of (heuristic) trust path computations [12]. Although we did not discuss any advanced path search algorithm or heuristic, such research is likely to help in increasing the maximum number of identities and signatures which can be handled by a single server. Additionally, we believe that analyzing load balancing and data distribution of certificates is interesting future research regarding the overall scalability of our approach.

VI. Related Work

Many works consider the security of VoIP and prevention of unsolicited communications (e.g., [13], [14]). PGP certificates had been envisioned for signing messages in the original SIP specification [15] (now deprecated in RFC 3261 [1]). However, PGP was only envisioned as the certificate format for SIP and not for using a distributed WoT model. Zimmermann proposed ZRTP [16] as a decentralized solution for user authentication and key exchange over RTP streams. In contrary to our work, ZRTP does not consider trust paths for signaling messages but only direct authentication of audio streams between caller and callee after the call has been established.

In addition, researchers have analyzed the existing PGP Web-of-Trust (in the context of email communications). Capkun et al. [17] as well as Penning [18] provide a graph analysis of the PGP network. Bidder et al. propose a new method for synchronization among key-servers in a WoT [19].

To the best of our knowledge, our work is the first investigation and implementation of adapting a Web-of-Trust model for securing real-time communications. As such, our work is very related to other work in the area of VoIP security but novel as it exploits the social relationships among users and as it proposes a decentralized cryptographic scheme for assessing trustworthiness of signaling messages. At the same time, our work is related to the field of Web-of-Trust research but novel since it explores *real-time* verification of message trustworthiness which is not possible with existing solutions.

VII. Conclusion

We applied a Web-of-Trust model to real-time communications in order to secure applications such as VoIP. Our approach exploits the social relationships between end-users for detecting *solicited* real-time communications. Further, it can prevent identity spoofing attacks using a decentralized trust model.

We evaluated our proposal using real world WoT graphs. Our results demonstrate that, for short path lengths, a trivial BSF path search algorithm integrated into a SIP proxy performs well enough for deriving trust paths in a real-time communication scenario. However, we showed that the standard certificate downloading behavior of PGP is insufficient in two aspects: First, it is too slow to support real-time communications. Second, signatures in uploaded certificates are not verified. Thus, we introduce a specialized *Verifying Key Server (VKS)* which pre-verifies WoT trust paths. To prevent time-consuming downloading of certificates for key-IDs, such a VKS should either be integrated with a SIP proxy or, as an alternative, the caller must append his public key in a self-signed certificate to signaling messages.

In future work, we intend to investigate path search algorithms and heuristics which scale for operator grade

networks (e.g., with more than 1,000,000 users). Further, we consider developing more sophisticated metrics than the trust path length (e.g., a combination of the number of paths from callee to caller with the individual path lengths) interesting future research.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261 (Proposed Standard), Jun. 2002, updated by RFCs 3265, 3853, 4320, 4916. [Online]. Available: http://tools.ietf.org/html/rfc3261

[2] P. R. Zimmermann, *The official PGP user's guide*. Cambridge, MA, USA: MIT Press, 1995.

[3] H. P. Penning, "Computing shortest paths in WOTs," Aug. 2004. [Online]. Available: http://pgp.cs.uu.nl/doc/shortest-paths-in-wots.php

[4] DFN-Cert, "PGP-Keyserver." [Online]. Available: http://wwwkeys.de.pgp.net/

[5] H. Tschofenig, H. Schulzrinne, D. Wing, J. Rosenberg, and D. Schwartz, "A Framework to tackle Spam and Unwanted Communication for Internet Telephony (work in progress)," Internet Engineering Task Force, 2008. [Online]. Available: http://tools.ietf.org/html/draft-tschofenig-sipping-framework-spit-reduction

[6] J. Peterson and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)," RFC 4474 (Proposed Standard), Aug. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4474.txt

[7] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer, "OpenPGP Message Format," RFC 4880 (Proposed Standard), Nov. 2007.

[8] J. Quittek, S. Niccolini, S. Tartarelli, and R. Schlegel, "On Spam over Internet Telephony (SPIT) Prevention," *IEEE Communications Magazine*, vol. 22, no. 5, 2008, in press.

[9] J. Cederlöf, "Web of trust statistics and pathfinder (WOTSAP)." [Online]. Available: http://www.lysator.liu.se/ jc/wotsap/

[10] S. Kent and K. Seo, "RFC 4301: Security Architecture for the Internet Protocol (Proposed Standard)," Internet Engineering Task Force, 2005. [Online]. Available: http://tools.ietf.org/html/rfc4301

[11] "Open Crypto Development Kit (OpenCDK)." [Online]. Available: http://www.gnu.org/software/gnutls/reference/gnutls-opencdk.html

[12] U. Brandes, "On variants of shortest-path betweenness centrality and their generic computation," *Social Networks*, vol. 30, no. 2, pp. 136–145, May 2008. [Online]. Available: http://dx.doi.org/10.1016/j.socnet.2007.11.001

[13] V. over IP Security Alliance, "VoIP Security and Privacy Threat Taxonomy, Public Release 1.0," Oct. 2005.

[14] V. A. Balasubramaniyan, M. Ahamad, and H. Park, "CallRank: Combating SPIT Using Call Duration, Social Networks and Global Reputation," in *CEAS 2007 Fourth Conference on Email and AntiSpam*, 2007.

[15] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: Session Initiation Protocol," RFC 2543 (Proposed Standard), Mar. 1999, obsoleted by RFCs 3261, 3262, 3263, 3264, 3265. [Online]. Available: http://tools.ietf.org/html/rfc2543

[16] P. Zimmermann, A. Johnston, and J. Callas, "ZRTP: Media Path Key Agreement for Secure RTP, draft-zimmermann-avt-zrtp (work in progress)," Internet Engineering Task Force, Mar. 2009. [Online]. Available: http://tools.ietf.org/html/draft-zimmermann-avt-zrtp

[17] S. Čapkun, L. Buttyán, and J.-P. Hubaux, "Small Worlds in Security Systems: an Analysis of the PGP Certificate Graph," in *In Proceedings of The ACM New Security Paradigms Workshop*. ACM Press, 2002, pp. 28–35.

[18] H. P. Penning, "analysis of the strong set in the PGP web of trust," Mar. 2009. [Online]. Available: http://pgp.cs.uu.nl/plot/

[19] *Key Exchange (KX) - A Next Generation Protocol to Synchronise PGP Keyservers*. Washington, DC, USA: IEEE Computer Society, 2003.