Journal of Network and Computer Applications 33 (2010) 16–28

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A framework for identity privacy in SIP

Giorgos Karopoulos *, Georgios Kambourakis, Stefanos Gritzalis, Elisavet Konstantinou

Info-Sec-Lab: Laboratory of Information and Communication Systems Security, Department of Information and Communication Systems Engineering, University of the Aegean, Samos GR-83200, Greece

ARTICLE INFO

Article history:
Received 15 November 2008
Received in revised form
10 May 2009
Accepted 27 July 2009

Keywords:
Privacy
SIP
Security
Heterogeneous networks
Performance evaluation
Multimedia

ABSTRACT

Secure multimedia delivery in modern and future networks is one of the most challenging problems towards the system integration of fourth generation (4G) networks. This integration means that different service and network providers will have to interoperate in order to offer their services to end users. This multidomain environment poses serious threats to the end user who has contract with, and trusts only a limited number of operators and service providers. One such threat is end users' privacy on which we will focus in this paper. Probably the most promising protocol for multimedia session management is the Session Initiation Protocol (SIP), which is an application layer protocol and thus can operate on top of different lower layer technologies. SIP is quite popular and a lot of research has been conducted; however, it still has some security issues, one of which is related to privacy and more particularly the protection of user identities (IDs). In this paper we comment on the ID privacy issue of SIP and propose a framework called PrivaSIP that can protect either the caller's ID or both the caller's and the callee's IDs in multidomain environments. We present different implementations of our framework based on asymmetric and symmetric cryptography analyzing the pros and cons of each one of them. Furthermore, we provide performance measurements in order to estimate the performance penalty of our framework over standard SIP. The most significant advantage of our method is that it can assure user ID protection even when SIP messages are transmitted through untrusted SIP domains, while our results show that this can be achieved with no perceived delay by the end user.

1. Introduction

Multimedia is an application class with great importance in today's and future networks no matter whether these are wired or wireless. In fact, it is important that multimedia delivery is based on interoperable protocols so that converged (and possibly heterogeneous) networks can offer uninterrupted services. It is expected that the next generation of wireless networks, namely 4G, will be based on IP, realizing an all-IP architecture. It is obvious at this point that such IP-based networks will be fully compliant with wired networks and the Internet with no need for gateways or other traversal means. In such an environment the multimedia deliverance will be possible even when users move or change between networks with different access layer technologies.

One of the most important protocols supporting multimedia services is the Session Initiation Protocol (SIP) (Rosenberg et al., 2002). SIP is an application layer control signaling protocol responsible for the creation, modification and termination of multimedia sessions. One of the facts that show the significance of SIP is that 3GPP consortium (3rd Generation Partnership Project) chose it to be the multimedia management protocol of 3G networks multimedia subsystem (IP Multimedia Subsystem—IMS). Since SIP is an application layer protocol, it can transparently operate over any type of network; furthermore, it also has the ability to support application layer handovers when a lower layer handover occurs (Schulzrinne and Wedlund, 2000). Another advantage of SIP is that it can inherently support multidomain environments and thus it can fully operate in future networks where many different operators and service providers may co-exist.

SIP has been a protocol that has received extensive attention and part of the research has shown that it suffers from security issues (Geneiatakis et al., 2005) some of which have already been solved (Geneiatakis et al., 2005; Geneiatakis et al., 2007). In this paper we focus on privacy and more specifically on the protection of user IDs that normally are publicly available to anyone who has access on the underlying network. SIP utilizes SIP URIs to locate users and appropriately route SIP messages. A SIP URI has the form "user_ID@domain_name", for example smith@minitrue.org. When a SIP user wishes to contact another user both their SIP URIs (and thus their user IDs) will be present in the corresponding SIP messages. In this paper we protect these user IDs from

* Corresponding author. Tel.: +30 22730 82246; fax: +30 22730 82009.
E-mail addresses: gkar@aegean.gr (G. Karopoulos), gkamb@aegean.gr (G. Kambourakis), sgritz@aegean.gr (S. Gritzalis), ekonstantinou@aegean.gr (E. Konstantinou).

1084-8045/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2009.07.004

unauthorized parties; this does not imply the protection of domain names as well since they are necessary for the operation of the protocol. While there are also other solutions for protecting the privacy of end users, these are not adequate in certain occasions compared to our schemes as we will show.

The existence of several overlapping networks in 4G will lead to a plethora of choices between different network providers for the user. Taking into account that multimedia content providers could be other than the network providers it is obvious that each user has to communicate with different administrative domains. These domains will not always be known or trusted beforehand so the users must be very careful when revealing their IDs to such foreign domains. The only viable assumption that can be made in such environments is that only domains that the user has a subscription with can be considered trusted.

In this paper we propose a framework that protects the IDs of communicating users regardless of the number or the level of trust of domains that reside between them. In Karopoulos et al. (2008), we have proposed a method for protecting caller only IDs based on SIP Proxy server certificates and the RSA algorithm. Here, we introduce the utilization of other cryptographic algorithms as well, in order to benefit from the advantages of symmetric and elliptic curve cryptography. Also, we further improve our scheme by offering the option to protect the callee's ID as well. Our framework operates in an ad-hoc manner requiring no prior trust agreements between the user and traversed domains other than the possession of already shared or public keys of the respective SIP Proxy server. We also provide performance analysis of various implementations of our framework through an appropriate testbed and compare our results with standard SIP that provides no ID privacy. Furthermore, we review existing solutions in SIP privacy pointing out their advantages and disadvantages compared to our proposal.

Section 2 starts by presenting the ID privacy issues of SIP in more detail. In this section the problem statement is given followed by a section reviewing related work on solutions that could offer privacy services to SIP users. Section 3 presents our framework, namely PrivaSIP, its two variations and five implementations with different cryptographic algorithms. In Section 4 we provide time delay measurements of our schemes in comparison to standard SIP. In Section 5 there is a discussion about the results of our experiments and comments on choosing the optimum implementation. Section 6 summarizes the contribution of this paper compared to previous work, while Section 7 concludes the paper and gives some directions for further research.

2. SIP identity privacy issues

In this section we present a generalized SIP architecture which spans across many different administrative domains and the identity privacy issues that arise from it. Our analysis is so general that applies to either wired or wireless scenarios or a mix of them. We pay special attention on the applicability of our solution to heterogeneous, in terms of access technology, networks which belong to different administrative domains. This is because the next generation of networks, also known with the term 4G, will probably be composed of interconnected networks that may not be administered by the same provider or by providers that have trust agreements between them. In such a many-to-many fashioned environment where security and/or privacy policies enforcement is not always feasible, care should be taken so that users IDs are protected even when they are roaming through untrusted domains. Multimedia content delivery in 4G can be realized with SIP no matter which link layer technology is being used. Without loss of generality, for the remainder of the paper we employ an example of a Voice-over-IP (VoIP) call between two users. However, the proposed solutions apply as is to other types of multimedia sessions as well. In this example SIP utilizes the standard way of authentication as defined in Rosenberg et al. (2002), which is Digest authentication using pre-shared username and password as credentials.

In Fig. 1, O'Brien uses a fixed terminal residing in *miniluv* domain and Smith uses a mobile terminal. Smith's Home Domain is *minitrue* but at the moment he roams to a different domain, *minipax*, and wants to contact O'Brien. If Smith's terminal is not aware of its Home SIP Proxy's IP address then a possibility is that other Proxies (like Local outbound Proxy) intervene between Smith and *minitrue.org* as well as between *minitrue.org* and *miniluv.org*. Most of the times these SIP Proxies are unknown to Smith and cannot be considered trusted; moreover, Smith has no means to control which Proxies his messages will travel through. Such messages, which are known as SIP messages, contain among other information SIP URIs. A SIP URI is a URI similar to an e-mail address which contains a user ID and a domain name separated by the "@" symbol. Taking *smith@minitrue.org* as an example, "smith" is the user ID and "minitrue.org" the domain name. In our

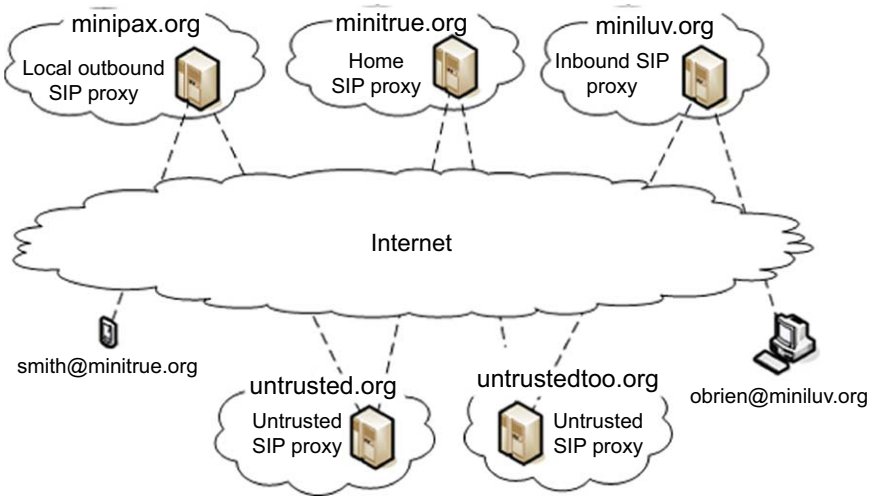

Fig. 1. Multidomain SIP architecture.

example, *minipax* is not Smith's Home Domain, but if Smith is to use its services he must have some kind of agreement with it, and consequently, trust it to some extent. However, the credentials used in this domain can be different than those used in his Home Domain, which is *minitrue.* A probable requirement here could be that Smith wishes that each set of credentials is available only to the corresponding domain and not to anyone else. Considering ID privacy, SIP per se cannot protect users' IDs since they are transmitted in the clear. Other methods that have been proposed so far and are presented in the related work section prove to be inadequate in certain occasions. What is really needed is a solution that selectively makes Smith's ID known only to entitled trusted entities, while hiding it from untrusted ones.

Considering the previous example the information that is revealed to third parties is that a user from *minitrue.org* domain has a conversation with O'Brien from *miniluv.org.* In a second example, a more effective, in terms of privacy, scheme could also protect O'Brien's ID so that the only information available to third parties would be that a user from *minitrue.org* has some sort of communication with a user from *miniluv.org.*

In order to demonstrate the SIP ID privacy issue more clearly we examine the headers of a SIP message used for placing a call, e.g. an INVITE sent from Smith to O'Brien (other SIP messages have similar headers):

```
INVITE sip:obrien@miniluv.org SIP/2.0
Via: SIP/2.0/UDP 195.251.161.144:5060;
branch = h2zbG4b47Kb43
Max-Forwards: 60
From: Smith⟨sip:smith@minitrue.org⟩;
tag = 2exfdf76sl
To: O'Brien⟨sip:obrien@miniluv.org⟩
Call-ID: 22013825198482768810minitrue.org
CSeq: 1 INVITE
Contact:⟨sip:smith@minitrue.org⟩
Content-Type: application/sdp
Content-Length: 151
```

As it can easily be seen in the above message, particular headers reveal private information about the two communicating parties. More specifically, the headers that reveal information about the caller and the callee are:

- The first line of the message contains a Request-URI which reveals the callee's ID
- ⟨Via⟩ reveals the caller's host IP address,
- ⟨From⟩ reveals the caller's SIP URI (which is composed from the user's ID followed by his Home Domain name)
- ⟨To⟩ reveals the callee's SIP URI
- ⟨Call-ID⟩ reveals the domain where the caller belongs (in this case *minitrue.org*) and
- ⟨Contact⟩ reveals where the caller can be contacted so that the two parties can establish a peer-to-peer connection (the value of this field can be either a SIP URI or simply an IP address).

We argue that our aim is to protect only the IDs of the users and not all the information about them like their hosts' IPs and the domains they belong. In fact such information is necessary since the callee must know the caller's IP in order to eventually establish a peer-to-peer media session. Moreover, third-party Proxies must be aware of the caller's and the callee's Home Domains in order to route the messages to the right SIP Proxies. For the above reasons we argue that the protection of such information is out of the scope of this paper; we refer the reader to Gritzalis (2004) where a review of such solutions can be found. For instance, one possible solution towards solving this issue is that the user's SIP Home Proxy could play the role of a Network Address Translation (NAT) server as well.

There are a number of malicious acts associated with the lack of user ID privacy. The first and more obvious one is that everybody can have access to information regarding who is communicating with whom. If this information is systematically gathered then a certain user can be profiled, based on VoIP calls and other multimedia usage. When SIP URIs are made available then a possible attack is also Spam over IP Telephony or SPIT (Rosenberg and Jennings, 2008), which is similar to e-mail spam. Another consideration is that the movement of a specific user can be tracked by observing the transmitted IDs over time. This can happen when a mobile user handovers between different networks and transmits his ID in order to transfer the existing session to the new network. This can also be the case when session mobility is supported and a certain user continues using a session but changes between different devices, either mobile or not.

3. Related work

The issue of privacy protection is not completely ignored in SIP and this is proved by the fact that (Rosenberg et al., 2002) includes certain mechanisms that can assist a user in protecting his privacy. These mechanisms can be separated to cryptography based ones which are S/MIME (Ramsdell, 2004), SIPS URI/TLS and IPsec, and the non-cryptographic solution of "Anonymous" URI. A different approach is the extension of the basic SIP protocol that led to the solution presented in Peterson (2002), which will be referred here as "Privacy Mechanism for SIP". This is in fact a general purpose privacy mechanism which has also been used in Jennings et al. (2002) adapted to the specific needs arose there. What we must note is that the aforementioned mechanisms are not focusing on protecting user ID privacy specifically but since they can achieve such qualities we consider them here as related work. In the following all the above solutions are presented in more detail, while more attention is given on how each solution can be utilized to protect end users' IDs.

3.1. S/MIME

SIP messages consist of two parts: the header and the body. The body part is nothing more than a MIME body so an obvious solution to protect it is by using the standard way which is S/MIME. Although this may seem out of scope, given that our work focuses on protecting specific SIP headers, S/MIME in the context of SIP can be used to cryptographically protect individual SIP headers.

S/MIME protects the confidentiality of SIP headers and bodies using digital certificates. In order to protect the privacy of end users, S/MIME can encapsulate SIP messages into MIME bodies and encrypt them properly. In our case the encapsulated message can contain the real ID of the caller while the "outer" message contains a ⟨From⟩ header of the form: "sip:anonymous@anonymizer.invalid". When the called party receives the message, he decrypts the body to find the ID of the caller. What must be noted here is that the ID of the callee cannot be anonymized using the same mechanism since the intermediate SIP Proxies do not have access to the plain MIME body and an anonymous ⟨To⟩ field would made them unable to route the message to the intended recipient.

Although S/MIME can seem as a promising solution there are some obvious weaknesses. First of all the receiver of messages must somehow be aware of the identity of the sender a priori, in

order to find the appropriate certificate to decrypt the message body. Another privacy weakness is that the receiver knows the ID of the sender, while the receiver's ID is not protected from third parties. Another serious weakness of this scheme is that it cannot support authentication since the ID of the caller is not visible to any of the intermediate SIP Proxies. Finally, a full Public Key Infrastructure (PKI) is required in order to manage certificate for the end users.

3.2. SIPs URI/TLS

It is possible for end users to request that their messages along the whole path to their destination are transported with the use of TLS protocol in order to ensure their privacy protection. This is accomplished with the use of "sips:" instead of "sip:" in a typical SIP URI. While a SIP message having a ⟨To⟩ header of the form: "sip:obrien@miniluv.org" will be visible by anyone, its security enhanced equivalent "sips:obrien@miniluv.org" will request all intermediaries to use TLS in a hop-by-hop manner until the specified domain is reached. After that, the message is handled according to the local security and routing policy.

This approach also presents some worth noting issues. If SIPS URI scheme is selected, then the use of TLS implies the use of TCP as a transport means, while the preferred transport protocol for SIP is UDP. While there is also the solution of DTLS (Rescorla and Modadugu, 2006), which is the equivalent of TLS using UDP as transport mechanism, it is a scheme that was proposed later than SIP so it is not included in (Rosenberg et al., 2002). The main drawback of SIPS URI, however, is that there is no guaranteed end-to-end protection. While TLS can be used in each hop-by-hop connection, it is not possible to dictate or even be informed somehow that it will be used in every intermediate connection. This can result in two possible attacks; the first one is a downgrade attack, where some intermediate proxy just does not use TLS or replaces "sips:" scheme with "sip:". In the second attack the caller uses plain "sip:" scheme and some intermediate proxy modifies it to a SIPS URI so that the recipient of the message believes that their communication is TLS protected. This approach requires the operation of a full PKI to manage digital certificates for end users and intermediate SIP Proxies.

3.3. IPsec

For the purposes of SIP, IPsec can be used in a hop-by-hop fashion protecting the data transmitted between two hosts at the network level. The main difference between IPsec and SIPS URI/TLS in the context of SIP is the transparency offered by IPsec to SIP User Agents (UAs). As it is stated in Rosenberg et al. (2002), IPsec will be more suitable in cases where the communicating hosts have already established a trust relationship with one another as opposed to SIPS URI scheme.

What holds for end-to-end protection in SIPS URI also applies here; it is not guaranteed. This is because there is neither an available mechanism to impose the use of IPsec in all intermediate hosts nor a way for communicating parties to be aware of whether this actually happened or not.

3.4. Anonymous URI

Another approach proposed in Rosenberg et al. (2002) for the protection of caller's ID is the use of an Anonymous URI in the ⟨From⟩ field. This URI has meaningless values and it is of the form: "sip:anonymous@anonymous.invalid". It must be noted here that this Anonymous URI is inserted into the ⟨From⟩ field by the UA itself which means that the SIP Proxy can never have access to the real URI.

The drawback of this solution is that it cannot support UA authentication since no ID is transmitted. A possible workaround could be a UA device shared among many end users. This device will own a specific pair of username and password for authentication purposes which will be the same for all users; however, such a solution creates other important security issues like repudiation of actions.

3.5. Privacy mechanism for SIP

The scheme described in Peterson (2002) is an extension of the basic SIP protocol and defines two ways for the protection of end user's privacy: user and network provided privacy. The end user can choose between these two or utilize both at the same time. When the UA chooses user provided privacy, it populates certain SIP headers with meaningless values, for example ⟨From⟩ field with an Anonymous URI. When network provided privacy is selected an intermediate node is assigned a new logical role for offering anonymization services to UAs while at the same time is responsible for directing messages from and to the anonymous user as a normal SIP Proxy. In order to enable UAs to request such services a new SIP header is introduced, namely "Privacy-hdr", which takes the following values: header, session, user, none and critical. With the use of one or more of these values the users can ask the network to: obscure headers that cannot be altered without the assistance of an intermediate, for example ⟨Via⟩ and ⟨Contact⟩, provide anonymization services for the session initiated by the message, cancel any default privacy preferences or mark the criticality of the request for privacy. The recommended way for the UA to communicate with the privacy service provider is by using network or transport layer security protocols.

This mechanism has also been adapted to fit certain requirements in Jennings et al. (2002). In this version the user sends a SIP message through a trusted set of Proxies revealing his true ID. When the message is about to leave this trusted domain, the last Proxy withholds the true ID of the user. Similarly to the initial scheme the last Proxy must keep state information in order to route back the responses.

A shortcoming of this method is that the node offering privacy services must keep a significant amount of state information in order to complete the proper routing of the messages. Another issue is that this node can potentially be a single point of failure if replication is not used. When user provided privacy alone is chosen then what applies for the "Anonymous URI" solution also applies here. The authors of this method have chosen not to consider any privacy considerations arose by the use of authentication mechanisms like Digest authentication. However, a username used in such a method could possibly reveal private information about the end user. Finally, a limited PKI is necessary with certificates issued only for SIP Proxies so that UAs can contact them securely with the use of TLS.

4. PrivaSIP

Following the privacy issues presented in Section 3 we would like to present our approach of defeating such problems when using SIP. While concealing user IDs could be a sort of protection for both examples of Section 3, it would make SIP non-operable. That is because SIP needs user IDs in order to locate the correspondent users, route the messages appropriately and possibly charge them for the received services. A more convenient solution would be the revealing of user IDs only to absolutely

necessary parties so as to route SIP messages appropriately and authenticate the caller before offering their services.

The proposed solution is an identity protection framework named PrivaSIP. The main idea behind the PrivaSIP framework is that each ID should be individually encrypted in a way that it can be recovered only by entities that need to do so in order for SIP protocol to operate correctly. Hereinafter the term "ID" is used to abbreviate either the user ID part of a SIP URI or other types of user IDs like a Digest authentication username. We define here two different variations of PrivaSIP: in the first one only the caller's IDs are protected while in the second both the caller's and the callee's IDs are protected; we will refer to these two variations as PrivaSIP-1 and PrivaSIP-2, respectively.

Using the previous analysis of a SIP INVITE message we can present here which are the specific header fields that need protection, so that user IDs are protected as well. What we must clear out here is that our solution does not aim at protecting the confidentiality of whole messages or provide message integrity; such requirements should be met by utilizing other mechanisms. The solution we propose is to strip whichever information is not necessary and use encryption for the rest. More specifically:

- we leave ⟨Via⟩ field's value as is, because it only reveals the IP address of the host
- ⟨Contact⟩ field's value is replaced with the IP address of the caller's host. End users' IP addresses usually are not static so eavesdroppers cannot easily relate it with the permanent ID of the user
- the display name in ⟨From⟩ field ("Smith" in our example) is stripped or replaced by the string "Anonymous"
- the user ID part of ⟨From⟩ field (i.e. "smith" in "smith@minitrue.org") is encrypted using either asymmetric or symmetric cryptography. As it is obvious we propose a scheme that rather relies on pseudonymity than anonymity (Pfitzmann and Hansen, 2008). If the same pseudonym is always used then the user can be "profiled" and his movement (in case of a mobile user) can be easily tracked. For this reason a padding scheme should be used so that the resulting pseudonym is different every time
- when callee's ID privacy is also a requirement then the previous encryption procedure applies to the user ID found in Request-URI and ⟨To⟩ field.

The resulting message for the first variation of PrivaSIP, where only the caller ID is protected, is shown below. In this example the hexadecimal representation is used for the encrypted part of the URI and its length depends on the cryptographic algorithm.

```
INVITE sip:obrien@miniluv.org SIP/2.0
Via: SIP/2.0/UDP 195.251.161.144:5060; branch=
bK7z9hb43G44
Max-Forwards: 60
From:⟨sip:0AEE5F83...129F32@minitrue.org⟩;
tag=ced79fslx6
To: O'Brien⟨sip:obrien@miniluv.org⟩
Call-ID: 6293841888221120785@minitrue.org
CSeq: 1 INVITE
Contact: 195.251.161.144
Content-Type: application/sdp
Content-Length: 151
```

If authentication is not required then the most practical and effective solution would be the employment of "Anonymous" URI in ⟨From⟩ header. However, in a real-world environment the most probable case is that the user must be authenticated in order to be charged for the services he receives. If caller ID privacy is also a requirement then the existing schemes presented in the related work section are not adequate. As already stated, in this paper we only consider Digest authentication (Franks et al., 1999), which is the standard way of authenticating users in SIP environments. In the following we will present an example where both the Local outbound SIP Proxy and Home Proxy require Smith to authenticate in order to receive their services. We assume that Smith has a different set of credentials for each of the two domains and he is willing to present each of the two IDs he possesses only to the corresponding domain. Naturally, since Smith has credentials from both domains it means that he has some kind of agreement with each one of them, so he is aware of what kind of private information he presents to each domain. The key point here is that the caller has the choice to present private information only to selected domains minimizing the number of entities that posses this information. Moreover, he reveals to each domain only the private information this domain already know about the user's ID and not IDs that this user may possess from other domains. Caller ID privacy during the authentication process can be assured in a similar way as in the previous example. When the INVITE message is received, the Local outbound Proxy responds with a 407 Proxy Authentication Required message. Smith sends back a new INVITE where he encrypts the username used in ⟨Proxy-Authorization⟩ field with the (public or shared) key of the Local outbound Proxy as shown below. Also, the user ID part of ⟨From⟩ field is encrypted with the key of Home Proxy. It is worth noting that this encryption process does not imply in any way that it supports user authentication; this task is conducted with the utilization of Digest authentication. The different user IDs used here are in accordance with Rosenberg et al. (2002) and reveal each ID only to the intended Proxy.

```
INVITE sip:obrien@miniluv.org SIP/2.0
Via: SIP/2.0/UDP 195.251.161.144:5060;
branch=4bh74Kb43Gz9
Max-Forwards: 60
From:⟨sip:0AEE5F83...129F32@minitrue.org⟩;
tag=xed9f6sc7l
To: O'Brien⟨sip:obrien@miniluv.org⟩
Call-ID: 3204878628218519821@minitrue.org
CSeq: 1 INVITE
Proxy-Authorization: Digest username=''A838F347
...0A19AE98'', algorithm=MD5, realm=''minitrue.
org'', nonce=''1ae384d7...0f45de0a'', qop=
''auth'', opaque=''54a20fde7b987310'', respon-
se=''fa1fe3...86e75e'', nc=00000001, cnon-
ce=''adhefgbci''
Contact: 195.251.161.144
Content-Type: application/sdp
Content-Length: 151
```

The Local outbound Proxy decrypts Smith's username and completes the authentication process and, if it is successful, it forwards the INVITE to Smith's Home Proxy. The Home Proxy also completes authentication in the same manner. After that, the initial INVITE message is forwarded to the Inbound Proxy which sends it to O'Brien. As we can see no untrusted entities involved in the protocol (including O'Brien) are aware of Smith's ID. When O'Brien answers the call he uses the same encrypted headers, and his response travels all the way back to *minitrue.org* where the Proxy deciphers ⟨From⟩ header to discover the recipient of the message.

While the usefulness of our scheme is proven through examples, this does not limit its generality. The same procedure would be followed if, for instance, there were SIP Registrars instead of Proxies and REGISTER messages instead of INVITEs.

We can further improve the aforementioned variation so that it also preserves called party's ID as well. In order to present the inner workings of our second variation we will use the same example. The protection of callee's ID is achieved by a similar mechanism as the caller's ID with the use of asymmetric cryptography; more specifically we encrypt the ID with the public key of callee's Home Domain SIP Proxy. This variation uses only asymmetric cryptography since the caller usually does not have any shared secrets with the callee's Home Proxy. It must be noted here that we also protect caller's ID as shown in Section 3.

As we already presented some SIP headers of an INVITE message must be altered to protect the ID of the caller. Apart from these headers, in this scheme we also protect 〈To〉 field and Request-URI, which exposes callee's ID. The resulting message is shown below:

```
INVITE sip: 3F778AD9...B9E1C0A1@miniluv.org SIP/
2.0
Via: SIP/2.0/UDP 195.251.161.144:5060;
branch = b44bK73G49hz
Max-Forwards: 60
From: 〈sip:0AEE5F83...129F32@minitrue.org〉;tag =
d7ce9ab7s1
To: 〈sip: 8AD73F79...B9E1C0A1@miniluv.org〉
Call-ID: 3848276298220188511@minitrue.org
CSeq: 1 INVITE
Contact: 195.251.161.144
Content-Type: application/sdp
Content-Length: 151
```

What applies for user authentication in our first variation also applies here. The caller can hide both his ID and Digest username, while also the callee's ID is protected from third parties. The procedure that is followed is the same as presented previously except that when an INVITE is forwarded to the Inbound SIP Proxy, the 〈To〉 field is decrypted and subsequently send to O'Brien. When O'Brien responds back he uses the same encrypted headers so that the privacy-enhanced SIP message is routed appropriately.

In the subsequent sections we present five different implementations of our framework using different encryption algorithms. Our purpose is to find out which category of algorithms or specific algorithm is more efficient when used in our framework. These implementations fall into two categories following the two PrivaSIP variations: in the first one only the caller ID is protected while in the second one both the caller and the callee IDs are protected from third parties. In all these schemes the caller ID is protected and this can be done either with symmetric cryptography using as a key the Digest authentication password shared between the user and his Home Proxy, or with asymmetric cryptography using the public key of the Home Proxy. In the second category of schemes where the callee ID has to be protected as well, the public keys of both the caller's and the callee's Home Domains are used. This category of schemes uses only asymmetric cryptography since the caller usually does not have any shared secrets with the callee's Home Proxy. The specific cryptographic algorithms used in our case are for PrivaSIP-1: the well-known public key algorithm by Rivest, Shamir and Adleman (RSA) (Rivest et al., 1978), the Elliptic Curve Integrated Encryption Scheme (ECIES) (Abdalla et al., 1998), and the symmetric Advanced Encryption Standard (AES) (Daemen and Rijmen, 2002), and for PrivaSIP-2: RSA, and ECIES. Other asymmetric and symmetric algorithms can also be used based on the same principles. The use of asymmetric cryptography implies that some kind of PKI should be in place; in our case a limited PKI is enough. This partial PKI will issue and manage certificates for SIP proxies only, since these certificates are only used for the encryption of user IDs (and Digest usernames when needed).

4.1. PrivaSIP-1: caller identity privacy

According to the scheme we have proposed in Karopoulos et al. (2008), caller ID hiding can be supported even when untrusted SIP Proxies reside between trusted parties, like in the example shown in Fig. 1. In order to fulfill this requirement we used asymmetric cryptography with RSA and encrypted the caller's ID with the Home Proxy's public key so that only this trusted entity can recover it. At the same time, everybody else (including the callee, other users and Proxies) has access only to the encrypted form of the ID.

Here we define three implementations that utilize either asymmetric or symmetric cryptography. The first one, which is the one proposed in Karopoulos et al. (2008), is PrivaSIP-1 with RSA or PrivaSIP-1-RSA for short, and uses the Home Proxy's public key to encrypt the user ID and Digest username when this is necessary. Our next implementation, namely PrivaSIP-1-ECIES, is based on the standard encryption scheme for Elliptic Curve Cryptography (ECC), which is ECIES, also known as DHAES (Abdalla et al., 1998). A different approach is used in the last one, which is PrivaSIP-1-AES, where a symmetric cryptographic algorithm is utilized, more specifically AES, for the encryption of the caller ID. Since the caller and his Home Proxy share a password, which is used for Digest authentication, this password can also be used as a key (or as a key seed or master key) for the encryption of user ID with AES.

We have already pointed out the necessity of a padding scheme so that a user cannot be "profiled" even when his IDs are encrypted. For this reason we utilized Optimal Asymmetric Encryption Padding (OAEP) (Bellare and Rogaway, 1995) for RSA, resulting in a different user pseudonym every time. Describing ECIES in high level, to encrypt an amount of data, a new symmetric key is produced each time from the recipient's public key and data are encrypted with a symmetric algorithm using this derived key. For this reason, a padding scheme is not necessary here since the key is different every time so the ID pseudonym will also be different. A suitable padding scheme for PrivaSIP-1-AES would be the one described in ISO 10126:1991 (1991); this standard specifies that the padding should be done at the end of the last block of data with random bytes, and the padding boundary should be specified by the last byte. This way, every time the ID pseudonym will have a different value. Other padding schemes like adding zeroes at the end of the data block or adding the same string every time would result in the same pseudonym every time rendering our mechanism useless.

A pre-condition for PrivaSIP-1-RSA and PrivaSIP-1-ECIES to work is that the Home Proxy has a public–private key pair and the UA possess the public key. Similarly for PrivaSIP-1-AES there must be a Digest authentication password shared between the UA and the corresponding SIP Proxy.

4.2. PrivaSIP-2: caller/callee identity privacy

In this section we further improve the aforementioned scheme so that it also preserves called party's ID as well. Our purpose is to have an alternative when callee's privacy is also a requirement. In PrivaSIP-2 the caller ID is only recoverable by his Home Proxy while the callee's ID is available only to his own Home Proxy (which is the "Inbound SIP Proxy" in Fig. 1).

In PrivaSIP-2 we define two implementations using asymmetric cryptography. Here an approach based on symmetric cryptography (e.g. AES) cannot be applied to protect callee's ID

since it is difficult for the caller to have a shared secret with every possible callee's Home Proxy. In the first implementation the RSA algorithm is utilized and padding can be implemented with a mechanism like OAEP encoding; the UA of the caller encrypts caller's ID with the public key of his Home Proxy and callee's ID with the public key of the callee's Home Proxy. For the second implementation we have used Elliptic Curve Cryptography with ECIES and its operation is identical to the previous one based on RSA. This way in both schemes the caller's ID is only recoverable by his Home Domain and the callee's ID is only disclosed to his own Home Proxy. In this scheme SIP Proxies must have public–private key pairs and both public keys must be available to the caller.

5. PrivaSIP service time measurements

The performance of the proposed schemes for both the client and the server was evaluated in a properly designed testbed and the results are depicted in this section. It is well known that security or privacy mechanisms come always at a cost. However, apart from the effectiveness and robustness of the proposed mechanism, the key question in every case is if that cost is affordable. So, our intention here is not to evaluate SIP's performance in general but to determine the performance penalty imposed by our methods compared to standard SIP transactions. In the related work section we extensively discussed all known schemes that could be used for providing some sort of privacy in SIP. However, we do not compare the performance of these methods with that of PrivaSIP. The chief reason to do so is that each scheme presents different qualities, and each of them is useful under a specific context irrespective of the performance penalty one might impose. For example, when the user ID must be protected and authentication is also a requirement, then PrivaSIP is the only viable solution; when authentication is not a requirement, then Anonymous URI is the right choice. Also, other solutions either do not provide enough or assured privacy (IPsec, SIPS URI/TLS) or do not protect privacy during authentication (Privacy mechanism) or do not support authentication at all (S/MIME, Anonymous URI).

Our measurements derive from two scenarios, one measuring the delay on the client side and the other on the server side:

1. *Client delay*: In this scenario we compute the time required for a UA to construct an INVITE request; moreover, for comparison purposes, we recorded measurements when our schemes are in use and when they are not. The measured request creation phase constitutes from the preparation of all SIP headers including the encryption of the respective user IDs when PrivaSIP is utilized. We have measured delays using a "low-end client" as well as a "high-end client" so that we could investigate what is the impact of our method on different hardware configurations. This scenario runs only on clients and does not involve any network interaction since we only measure the INVITE preparation delay.
2. *Server delay*: According to this second scenario we measured the time required for a SIP Proxy server to serve a request. We have measured delays while the server is (over)stressed with different queue sizes of requests. The scenario was executed one time for each of our implementations and once using standard SIP using different queue sizes ranging from 100 to 1000 calls. For each queue size the call rate is automatically adjusted by SIPp. The measured time starts when an INVITE is sent and ends when a "180 Ringing" is received by SIPp; this means that the user has been authenticated and his call has reached the intended recipient.

We must note here that we take the worst case scenarios; all SIP URIs and digest usernames are computed each time they are needed and no party stores call state information. The delays included are:

- the parsing of the unauthenticated INVITE by Home Proxy (for our schemes SIP Express Router-SER (SIP Express Router (SER)) decrypts caller's URI),
- the digest response preparation time by the caller's UA (no encryption takes place here; the encrypted values used are hardcoded in SIPp's scenario file),
- the parsing of UA's response (for our schemes this involves the decryptions of UA's ID and Digest username),
- the parsing of INVITE by Inbound Proxy (for PrivaSIP-2 only, this involves the decryptions of callee's URI), and finally
- the respective network delays.

We have already presented some initial results for PrivaSIP-1-RSA in Karopoulos et al. (2008). However, here we present new results for this method's server delay. This is essential since in Karopoulos et al. (2008) we measured only a part of a SIP call while here we measure its overall time, i.e., from its initiation until the ringing phase. Another reason is that part of the testbed used in Karopoulos et al. (2008) has been substantially changed in the current work to be more realistic. The difference between the two SIP call flows (roundtrips) is depicted in Fig. 2. More specifically, the delays we presented in our previous work measure the time from the initiation of the call until the end of the first roundtrip, while here we measure the delay until the second roundtrip. The reason we do this is that in our previous work we only modified our Home SIP Proxy in order to cope with our cryptographically protected messages while all other network elements are based on standard SIP. So there was no need to take the whole call until the second roundtrip into consideration. Here, on the other hand, both SIP Proxies were modified and we were forced to rerun our experiments in order to be able to compare our schemes to each other. For the client's delay on the other hand we have used the same measurements presented in Karopoulos et al. (2008) for PrivaSIP-1-RSA since there is no modification to the client's software or hardware configuration.

In order to conduct our experiments we constructed a testbed, which comprises from the following elements:

- One low-end laptop machine, which incorporates an AMD Mobile Athlon 4 CPU at 1.2 GHz and 256 MB of RAM. For the purposes of our experiments, the laptop's CPU was downgraded from 1.2 GHz to 500 MHz with the use of Powersave daemon version 0.10.15, which is part of the machine's Operating System (OS). This enabled us to have similar capabilities as today's handheld and mobile devices. This laptop's network interface was not used since it ran only the client scenario as a "low-end UA". The OS of this machine is SuSE Linux 10.0, kernel version 2.6.13-15-smp, with gcc version 4.0.2 and the software used for measuring client's delay is based on Twinkle SIP softphone version 1.1 (Twinkle SIP softphone).
- One desktop PC with an Intel Pentium 4 Hyper-Threading CPU at 2.6 GHz and 512 MB of RAM, which also does not utilizes its network card since it is the "high-end User Agent (UA)" for measuring client delay. The OS of this machine is SuSE Linux 10.0, kernel version 2.6.13-15-smp, with gcc version 4.0.2 and the software used for measuring client's delay is based on Twinkle SIP softphone version 1.1.
- One desktop with a dual-core Intel Pentium 4 CPU at 3 GHz and 1 GB of RAM which plays the role of "User Agent 1" in Fig. 2. This machine connects to the network through a Broadcom

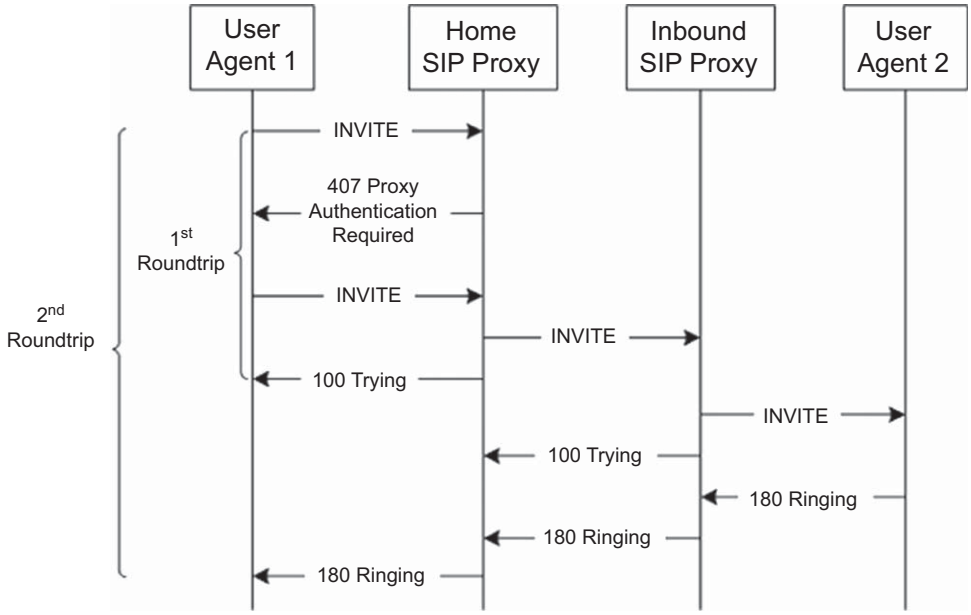

Fig. 2. SIP call flow.

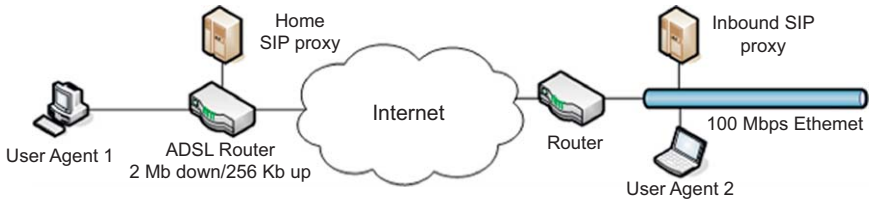

Fig. 3. Testbed network architecture.

NetXtreme Gigabit Ethernet card. Its purpose is to make multiple calls to User Agent 2 through the two Proxies so that we can measure the delay of each request when the Proxies have queue sizes of certain length. This is realized with the use of SIPp 3.0 in client mode which automatically adjusts the call rate so that a stable queue size is maintained. This machine's OS is openSuSE Linux 10.3, kernel version 2.6.22.18-0.2, with gcc version 4.2.1.

- One PC with a dual-core AMD Athlon X2 64 CPU at 1.9 GHz and 2 GB of RAM which plays the role of "Home SIP Proxy" in Fig. 2. This machine connects to the network through a Realtek RTL8102E Fast Ethernet 100 Mbps network card. The SIP proxy software is based on SER version 0.9.6 supported by MySQL version 5.0.45-community during the authentication procedure. This machine's OS is openSuSE Linux 11 (32-bit version), kernel version 2.6.25.16-0.1 with gcc version 4.3.
- One desktop PC with a dual-core Intel Pentium 4 CPU at 2.8 GHz and 1GB of RAM, which connects to the network through a Broadcom NetXtreme Gigabit Ethernet card and is used as the "Inbound SIP Proxy" in Fig. 2. The OS of this PC is openSuSE Linux 11, kernel version 2.6.25.16-0.1 with gcc version 4.3. The SIP proxy software is based on SER version 0.9.6.
- One desktop with a dual-core Intel Pentium 4 CPU at 2.6 GHz and 512MB of RAM which plays the role of "User Agent 2" in Fig. 2. This machine connects to the network through a Broadcom NetXtreme Gigabit Ethernet card. Its purpose is to receive the calls made by User Agent 1 and send back a "180 Ringing" message which is realized with the use of SIPp 3.0 in server mode. The OS of this PC is openSuSE Linux 11, kernel version 2.6.25.16-0.1 with gcc version 4.3.

Two different 1024-bit RSA digital certificates were issued for the Home Proxy and the Inbound Proxy to be used from PrivaSIP-1-RSA and PrivaSIP-2-RSA, and the corresponding public keys have been transferred to the UAs. For ECIES we have used 160-bit keys, and for AES 128-bit keys. Cryptographic operations related to RSA and AES were executed with the help of the open source OpenSSL library version 0.9.8g (OpenSSL), while for ECIES we used Crypto++ library version 5.5.2 (Crypto++). The measurements where conducted on the network architecture shown in Fig. 3. UA 1 and Home SIP Proxy reside in the same 100 Mbps LAN while Inbound SIP Proxy and UA 2 reside in another 100 Mbps LAN. The two subnetworks connect through the Internet over a 2 Mbit ADSL connection with 2048 Mbps maximum downlink and 256 Kbps maximum uplink speed. The average ping time between the two subnetworks is 22 ms but this value can only be considered as an indication.

We have made the following modifications to the initial versions of the open source software used:

- Twinkle. Our modified Twinkle was modified so that it encrypts 〈From〉 field and caller's Digest username in PrivaSIP-1 while for PrivaSIP-2 it encrypts both 〈From〉 and 〈To〉 fields and caller's Digest username. Encryptions involving RSA and AES are performed using OpenSSL while for those involving ECIES we employed Crypto++.
- SER. Our modified SER decrypt the user IDs, processes the request and forwards the message with the original encrypted user IDs. When Digest authentication is used it also decrypts the username of the UA.
- SIPp. SIPp creates SIP messages based on an XML file that describes a scenario. While encrypted SIP URIs are parsed

Table 1
SIP request preparation delay.

Configuration	Delay (ms)			Standard deviation	Confidence interval (95%)
	Mean	Min	Max		
1	0.16	0.14	1.34	0.07	(0.15, 0.16)
2	0.61	0.55	3.01	0.13	(0.6, 0.62)
3	4.88	4.34	8.42	0.48	(4.85, 4.91)
4	0.18	0.17	0.23	0.01	(0.18, 0.19)
5	0.99	0.89	3.29	0.24	(0.97, 1)
6	9.53	8.88	53.13	1.47	(9.44, 9.62)
7	0.38	0.31	6.11	0.20	(0.37, 0.4)
8	1.6	1.36	8.14	0.26	(1.59, 1.61)
9	24.22	22.26	251.26	7.23	(23.78, 24.67)
10	0.47	0.34	2.24	0.12	(0.46, 0.48)
11	2.66	2.33	10.36	0.48	(2.63, 2.69)
12	46.89	44.17	280.76	7.49	(46.43, 47.36)

correctly, we had to modify SIPp in order to parse long usernames (in our case 256 characters). When a 407 Proxy-Authorization request is received, SIPp's response includes the encrypted forms of the user ID and the username used for authentication.

For the client delay scenario we have taken measurements with 12 different configurations; our five implementations plus standard SIP run on a high-end UA as well as on a low-end-client. For each configuration we have measured the delay of the preparation of a single INVITE message 1000 times. These configurations are:

1. High-end UA with standard SIP
2. High-end UA with PrivaSIP-1-RSA
3. High-end UA with PrivaSIP-1-ECIES
4. High-end UA with PrivaSIP-1-AES
5. High-end UA with PrivaSIP-2-RSA
6. High-end UA with PrivaSIP-2-ECIES
7. Low-end UA with standard SIP
8. Low-end UA with PrivaSIP-1-RSA
9. Low-end UA with PrivaSIP-1-ECIES
10. Low-end UA with PrivaSIP-1-AES
11. Low-end UA with PrivaSIP-2-RSA
12. Low-end UA with PrivaSIP-2-ECIES

The measurements for configurations 1, 2, 7 and 8 are those calculated in Karopoulos et al. (2008); results for all other configurations correspond to novel schemes we introduced in this paper. Table 1 shows the results for each of the 12 different configurations. Apart from the mean delay, we have included in the table the minimum and maximum delays, the standard deviation of the taken measurements and the 95% confidence interval.

The observation of the table reveals that when our schemes are in use the INVITE preparation delay is in some cases significantly higher compared to standard SIP and this is obviously due to cryptographic operations involved. The highest delays are observed when ECIES is in use. However, all delays measured are in ms with an overall maximum of 280.76 ms, meaning that actually there is no perceived delay by the end user. Also standard deviation of all values remains low, showing that their majority is spread near the mean delay. This observation is further supported by the calculated confidence intervals.

Fig. 4 shows the impact of hardware configuration on INVITE request preparation delay for the different implementations of PrivaSIP-1, and Fig. 5 the corresponding delays for PrivaSIP-2. Here we depict the mean preparation delay values presented in Table 1 adding the confidence intervals for each mean value as error bars on the graph. The *X*-axis represents the scheme used, while *Y*-axis shows the INVITE preparation delay in ms.

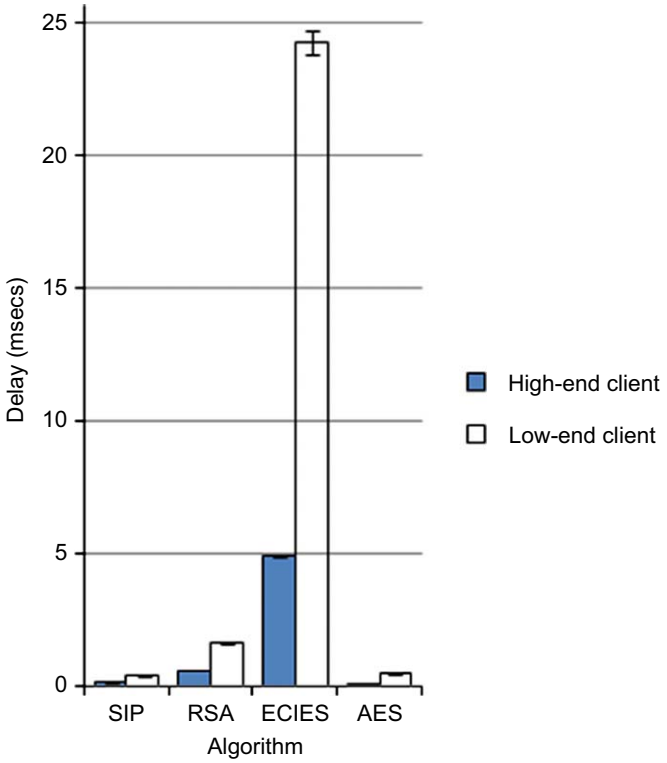

Fig. 4. Mean INVITE preparation delay for PrivaSIP-1.

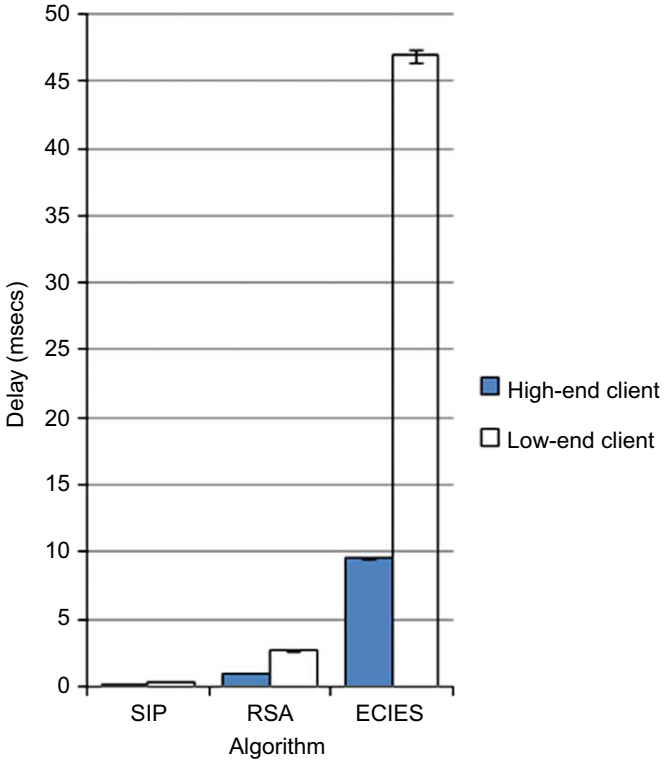

Fig. 5. Mean INVITE preparation delay for PrivaSIP-2.

During the execution of the second scenario we measured the mean server response times for different queue sizes for each implementation. These queue sizes range from 100 to 1000 calls and for each one of them we computed the mean response time of 1000 authenticated calls. For each different implementation examined, server's queue is populated with similar requests, i.e., standard SIP messages for measuring standard SIP's response delays, PrivaSIP-1-RSA messages for measuring PrivaSIP-1-RSA, etc. Server's queue population was realized with the SIPp tool, which can create multiple calls with automatically adjusted call rate, so as to keep server's queue at a predefined stable length.

Tables 2–7 show the results for the second scenario. These tables demonstrate the mean server response delays from the moment the user initiates a call until he gets back a "180 Ringing" message; for each implementation we also include the standard deviation of each mean value and the 95% confidence interval. From these results we infer that there is an overhead in our proposals in comparison to standard SIP regarding the response delays. However, these results are based on the assumption that in the first case we only have standard SIP requests while in the second case only our modified requests. In a more realistic scenario (where probably privacy will be offered with some additional cost) the requests will be mixed at all SIP proxies involved and the performance penalty will be decreased. Furthermore, as we have already explained, here we consider a worst case scenario regarding the number of cryptographic operations; keeping state information in some SIP Proxies and reusing encrypted URIs will improve the performance of our schemes.

Taking PrivaSIP-2-RSA as an example, in a full roundtrip as shown in Fig. 2, 6 decryptions take place; 4 in Home Proxy (first INVITE's ⟨From⟩ decryption, second INVITE's ⟨From⟩ and Digest username decryption, 180 Ringing ⟨From⟩ decryption) and 2 in Inbound Proxy (INVITE's ⟨To⟩ decryption, 180 Ringing ⟨To⟩ decryption). These decryptions could be limited to 2 if: (a) the client uses the same encrypted URI for all messages of a session,

Table 2
Mean server response delays for SIP.

Server queue size (calls)	Mean delay (ms)	Standard deviation	Confidence interval (95%)
100	483.2	757.93	(441.02, 525.39)
200	601.72	1019.41	(543.92, 659.52)
300	693.21	1183.6	(623.74, 762.69)
400	738.24	1243.67	(665.66, 810.81)
500	773.7	1306.31	(696.61, 850.79)
600	919.89	1464.04	(832.61, 1007.16)
700	861.13	1364.84	(779.42, 942.83)
800	934.92	1454.72	(847.26, 1022.58)
900	873.53	1361.27	(791.31, 955.75)
1000	1049.62	1461.18	(959.55, 1139.69)

Table 3
Mean server response delays for PrivaSIP-1-RSA.

Server queue size (calls)	Mean delay (ms)	Standard deviation	Confidence interval (95%)
100	755.77	992.59	(699.13, 812.4)
200	1030.39	1392.59	(941.46, 1119.32)
300	1145.02	1472.4	(1047.01, 1243.03)
400	1205.5	1536.38	(1100.34, 1310.65)
500	1149.63	1434.47	(1051.21, 1248.05)
600	1155.68	1460.5	(1055.96, 1255.4)
700	1213.87	1543.15	(1108.12, 1319.62)
800	1177.49	1515.59	(1072.72, 1282.25)
900	1279.31	1629.86	(1163.13, 1395.49)
1000	1209.43	1514.79	(1106.75, 1312.11)

Table 4
Mean server response delays for PrivaSIP-1-ECIES.

Server queue size (calls)	Mean delay (ms)	Standard deviation	Confidence interval (95%)
100	627.03	877.59	(577.83, 676.24)
200	828.42	1223.53	(756.79, 900.04)
300	993.72	1431.08	(907.41, 1080.03)
400	999.01	1404.05	(912.5, 1085.51)
500	951.42	1348.1	(868.73, 1034.11)
600	973.94	1369.37	(889.11, 1058.77)
700	1000.87	1385.58	(915.21, 1086.54)
800	991.05	1401.81	(904.08, 1078.03)
900	1065.5	1494.8	(972.76, 1158.24)
1000	1019.43	1409.61	(933.09, 1105.77)

Table 5
Mean server response delays for PrivaSIP-1-AES.

Server queue size (calls)	Mean delay (ms)	Standard deviation	Confidence interval (95%)
100	494.43	806.16	(449.32, 539.53)
200	586.44	969.07	(531.12, 641.75)
300	716.1	1182.66	(656.95, 775.24)
400	716.35	1174.95	(647.1, 785.59)
500	731.04	1218.94	(658.81, 803.27)
600	776.28	1295.45	(699.65, 852.9)
700	762.36	1219.36	(689.81, 834.91)
800	834.99	1346.2	(754.29, 915.69)
900	860.8	1356.03	(778.94, 942.67)
1000	936.5	1433.47	(848.75, 1024.26)

Table 6
Mean server response delays for PrivaSIP-2-RSA.

Server queue size (calls)	Mean delay (ms)	Standard deviation	Confidence interval (95%)
100	1244.53	1254.74	(1152.69, 1336.37)
200	1522.85	1592.64	(1382.97, 1662.73)
300	1651.57	1662.91	(1497.93, 1805.21)
400	1634.69	1644.03	(1477.84, 1791.55)
500	1741.45	1744	(1578.31, 1904.59)
600	1576.77	1611.76	(1417.22, 1736.32)
700	1721.93	1701.06	(1562.63, 1881.24)
800	1800.18	1853.96	(1627.34, 1973.02)
900	1858.92	1821.25	(1685.77, 2032.07)
1000	1749.49	1718.94	(1587.58, 1911.39)

Table 7
Mean server response delays for PrivaSIP-2-ECIES.

Server queue size (calls)	Mean delay (ms)	Standard deviation	Confidence interval (95%)
100	886.16	1018.81	(822.8, 949.53)
200	1150.4	1346.81	(1063.47, 1237.33)
300	1363.94	1571.47	(1256.25, 1471.63)
400	1403.3	1575.52	(1294.6, 1512.01)
500	1497.01	1726.1	(1376.34, 1617.68)
600	1523.73	1755.12	(1400.64, 1646.83)
700	1449.43	1623.98	(1336.04, 1562.82)
800	1433.99	1672.32	(1315.63, 1552.34)
900	1392.1	1591.52	(1272.04, 1512.17)
1000	1378.73	1571.97	(1258.98, 1498.47)

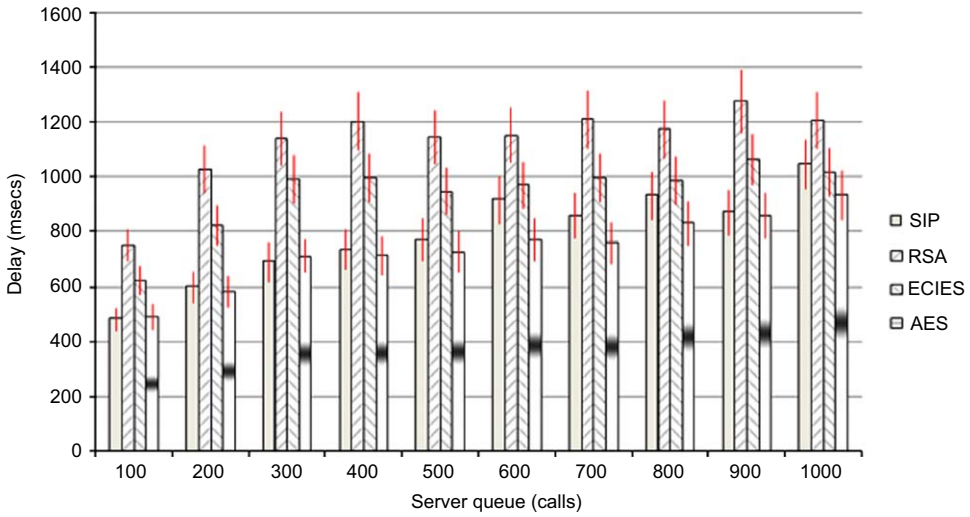

Fig. 6. Server response delays for PrivaSIP-1.

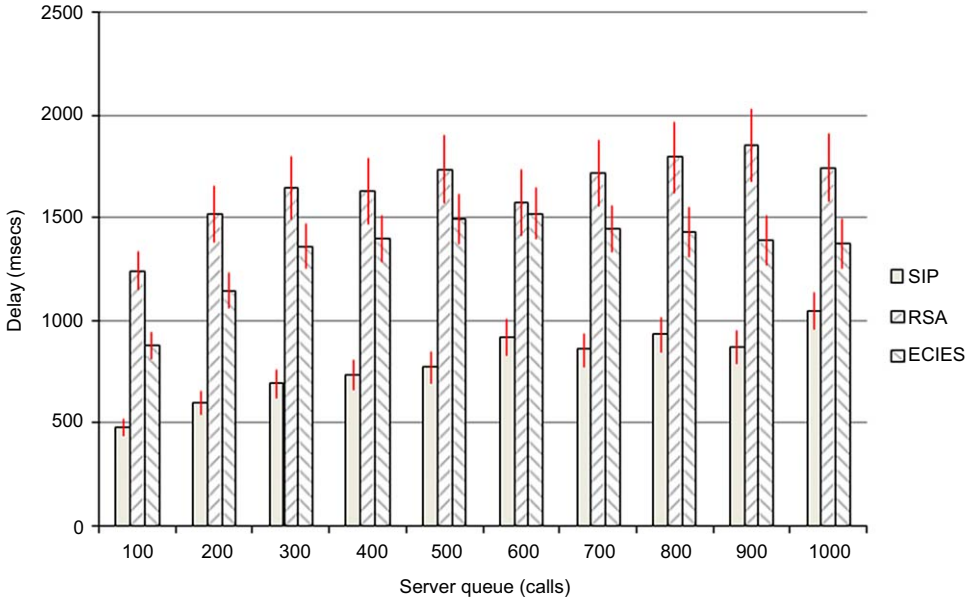

Fig. 7. Server response delays for PrivaSIP-2.

(b) the server stores a correspondence of the encrypted URI and its decrypted value, and (c) Digest username is the same with ⟨From⟩ user ID. To show the performance improvement that can be achieved we take the delays for server queue sizes of 1000 calls. The difference between PrivaSIP-2-RSA and standard SIP, that is 1749.49−1049.62 = 699.87 ms, is mainly due to cryptographic operations. So for each cryptographic operation we have a mean delay of 699.87/6 = 116.65 ms. Following the above optimizations we will have 2 cryptographic operations adding to the delay of standard SIP, i.e., 1049.62+2 × 116.65 = 1282.92 ms, which is a lot better than 1749.49 ms that we measured without any optimization. Of course this is not an accurate value but an estimation, which, however, shows how much faster our methods can be. It is up to the system administrator to decide and make the proper tradeoff between speed and storage needed for keeping state information.

Figs. 6 and 7 depict the mean server response delays for all our implementations for different server queue sizes. The *X*-axis represents the size of the queue, while *Y*-axis shows the mean response delay computed for each queue size in ms. In each point we have also included the corresponding confidence interval as error bars.

6. Discussion

In this section we discuss some interesting points regarding the use of PrivaSIP; we will first comment on the results of comparing related work with PrivaSIP and from the observation of the performance measurements. The first issue has to do with ID hiding. In some occasions it is desirable from the caller not to reveal his ID to the callee. This ID hiding type is supported by our schemes and by two other schemes as well, namely "Anonymous URI" and "Privacy mechanism for SIP". The difference here is that only the two PrivaSIP variations can support this feature while simultaneously protecting the Digest username during the authentication process.

As the most important advantage of our methods we consider their ability to maintain their privacy-protecting features while operating through untrusted domains even when these domains are placed between the caller and his Home Domain. While S/MIME can also protect the user's ID, it cannot protect his username during Digest authentication. Furthermore, it cannot offer caller's ID hiding from the callee.

Another consideration is that only "Anonymous URI" can protect the Home Domain name of the caller; however, this method is less practical since it cannot support authentication. Regarding the IP addresses of the communicating parties it is evident that no method can effectively protect them from eavesdroppers; a possible exception to this would be the employment of IPsec in tunnel mode. Another possible solution would be the use of a NAT server which could be co-located with the SIP Home proxy; however, this needs further investigation and it is considered future work. This solution presumes that the employed scheme should be stateful so all stateful schemes can be utilized; PrivaSIP can operate in either stateful or stateless mode so it is included in these schemes. In general, considering all schemes without the NAT extension, while both domain names and IP addresses are considered private information they should remain publicly available so that the two parties can communicate with each other during, as well as after, the session establishment.

Another remark concerning our schemes is the acquisition of Proxy certificates. Throughout this paper we assume that the UAs have in their possession the digital certificates of the Proxies they need. This is a logical assumption concerning the Home Proxy certificate of each user; however, the same cannot be straightforwardly asserted for other Proxies. Thus, when PrivaSIP-2 is utilized the caller's UA should first acquire and check the certificate of the callee's Home Proxy and then proceed to the protection of the messages. This, however, happens usually once and stands for multiple sessions, i.e., until the certificate of the corresponding foreign SIP Proxy that contains the public key expires. When symmetric cryptography is utilized we assume that the user shares a Digest password with his Home Proxy, which is of course a viable assumption.

As we have already stated, each PrivaSIP variation serves different purposes. From our experiments we can draw some conclusions as to which implementation is more efficient and easy to deploy. For PrivaSIP-1 it seems that the implementation with AES is the best choice since it is the most efficient one and the easiest in deployment; it only reuses the already shared Digest password between the caller and his Home Proxy and does not mandate any sort of PKI. For PrivaSIP-2 the most efficient implementation is the one that uses ECIES with the difficulties in deployment being the same with RSA since a PKI is needed to manage certificates for the Proxies. While ECIES presents higher delays on the client side, it also presents lower delays than RSA on the server side; in our case this is a desired effect since the imposed client delays are not perceived by the end user.

At this point we would like to clarify why there is no PrivaSIP-3 scheme protecting the callee ID only; e.g. when a user calls some special hotline. Considering anonymity from the caller's point of view, when using PrivaSIP-1 the only available information to others is that "some user from domain D1 is calling hotline H (which belongs to domain D2)". When using PrivaSIP-2, the available information is that "some user from domain D1 is calling some user from domain D2". In a hypothetical PrivaSIP-3 scheme the available information would be that "user U is calling some user from domain D2". However, since domains include (usually similar kind of) services administered by the same entity (this can be a person or corporation, etc.) this could be easily translated into "user U is calling a hotline from domain D2". So, the only protection in PrivaSIP-3 would be that others do not know exactly which hotline the user called. Moreover, everyone is able to access the real ID of the user that attempts to call a hotline. Naturally, this does not improve user's privacy in any way.

Another point needing clarification is how an ID hiding scheme like PrivaSIP can protect end users from spammers; moreover, such protection should be combined with the requirement that providers should log communication information in certain countries. The answer to both questions lies in the fact that both PrivaSIP schemes do not provide anonymity but pseudonymity. Regarding the spam issue, pseudonymity pre-supposes that a real ID should be used and this would probably discourage spammers. Furthermore, our methods support user authentication thus making massive anonymous requests difficult for spammers. Pseudonymity also means that all protected IDs are recoverable by the corresponding entities that are the Home servers of both users. More specifically, the caller's Home proxy can recover the caller's real ID and the callee's Home proxy can recover the callee's real ID. Under this context, data retention policies do not need to change; service providers log connection information and recover user IDs upon request.

The next comment relates with a set of protocols becoming popular among the SIP world, namely–the E.164 Number Mapping (ENUM) protocols (Mealling, 2002a–d). This suite of protocols is used to unify the telephone numbering system (E.164) with the addressing system used in the Internet by using the Domain Name System (DNS). Even when using ENUM, user ID hiding can be realized in the same way; the choice is on the user himself. The user can choose to place either his real SIP URI or a pseudonym (as described in PrivaSIP) in the DNS.

7. Contribution

In this section we would like to summarize and clear out the contribution of this paper compared to previous work. We present two SIP privacy-preserving protocols; the first one, namely PrivaSIP-1, has been introduced in Karopoulos et al. (2008), while the second one, PrivaSIP-2, is a novel one. In Karopoulos et al. (2008) we use RSA for PrivaSIP-1 while here we introduce two more implementations based on ECIES and AES for evaluation and comparison purposes. Regarding the testbed experiments, as it has been presented in Section 5, we have client and server side scenarios. In Karopoulos et al. (2008), we have measured the SIP INVITE preparation delay for the client and here we reuse the same measurements for SIP and PrivaSIP-1-RSA. In the current paper we present new results for PrivaSIP-1-ECIES, PrivaSIP-1-AES and all the implementations of PrivaSIP-2. For the server side scenarios, in Karopoulos et al. (2008) we have presented time measurements for SIP and PrivaSIP-1-RSA only. However, apart from new schemes evaluation, here we had to rerun all our past experiments for reasons explained in Section 5. Thus, all our server side measurements are considered new. Section 3 reviews related work on SIP privacy-preserving methods and comment on their advantages and disadvantages regarding SIP ID privacy; this is also an extended version of the related work presented in Karopoulos et al. (2008).

8. Conclusions

Security is perceived as a sine qua non-condition in public communication infrastructure, whether fixed or mobile. It has a dominant role in ensuring that citizens can trust the capabilities and the service framework that such an environment can offer within the modern Information Society. In the near future it is

envisioned that SIP will co-exist or even supersede classic telephony systems like PSTN. So, before this becomes reality certain security issues must be solved. While SIP is a simple and easy to deploy protocol, it turns out that some of the security problems related to it are hard to solve. One such problem is privacy since SIP messages cannot be cryptographically protected as a whole.

As we already showed SIP has a number of security and especially privacy-protecting mechanisms; however, some privacy issues are still open. Here we analyze a privacy issue concerning the user IDs of SIP users and concentrate on the protection of communicating parties IDs in an easy to deploy manner. We also review existing solutions focusing on how each method can protect user IDs and comment on them to show their advantages and disadvantages.

We argued that our methods can protect user IDs more effectively and in cases where existing methods fail to satisfy users' privacy needs. This is especially true when a fair balancing between privacy and performance is terminus. Our quantitative analysis through extensive testbed experimentation showed that for the client side the delay is negligible, while our methods cost on SIP Proxies turns out to be highly dependent on the chosen implementation ranging from comparable to standard SIP to quite expensive in terms of time delay. We have already discussed which implementations are appropriate under certain circumstances based on efficiency and easiness of deployment; our future work will concentrate in finding ways to further improve the privacy of SIP.

Acknowledgements

The authors would like to thank Mrs. Evangelia Papanagiotou for her assistance in statistical calculations.

This paper is part of the 03ED375 research project, implemented within the framework of the "Reinforcement Programme of Human Research Manpower" (PENED) and co-financed by National and Community Funds (20% from the Greek Ministry of Development-General Secretariat of Research and Technology and 80% from E.U.-European Social Fund).

Also, we would like to thank the anonymous reviewers for their valuable comments that helped us improve the quality and presentation of our work.

References

3rd Generation Partnership Project (3GPP) Consortium, 〈http://www.3gpp.org〉.

Abdalla M, Bellare M, Rogaway P. DHAES: an encryption scheme based on the Diffie–Hellman problem. Submission to IEEE P1363a. 〈http://grouper.ieee.org/groups/1363/P1363a/Encryption.html〉, 1998.

Bellare M, Rogaway P. Optimal asymmetric encryption—how to encrypt with RSA. In: De Santis A, editor. Extended abstract in advances in cryptology—Eurocrypt '94 proceedings, LNCS, vol. 950. Berlin: Springer-Verlag; 1995.

Crypto++, free C++ class library of cryptographic schemes, available at 〈http://www.cryptopp.com〉.

Daemen J, Rijmen V. The design of Rijndael: AES—the advanced encryption standard. Springer-Verlag; 2002.

Franks J, Hallam-Baker P, Hostetler J, Lawrence S, Leach P, Luotonen A, et al. HTTP authentication: basic and digest access authentication, RFC 2617, June 1999.

Geneiatakis D, Kambourakis G, Dagiuklas T, Lambrinoudakis C, Gritzalis S. SIP security mechanisms: a state-of-the-art review. In: Proceedings of the fifth international network conference (INC 2005), Samos, Greece, July 2005.

Geneiatakis D, Kambourakis G, Lambrinoudakis C, Dagiuklas T, Gritzalis S. A framework for protecting a SIP-based infrastructure against malformed message attacks. Computer Networks 2007;51(10):2580–93.

Gritzalis S. Enhancing web privacy and anonymity in the digital era. Information Management and Computer Security 2004;12(3):255–88.

ISO 10126:1991. Banking—procedures for message encipherment (wholesale)—Part 1: General principles, Part 2: DEA algorithm, 1991.

Jennings C, Peterson J, Watson M. Private extensions to the Session Initiation Protocol (SIP) for asserted identity within trusted networks, RFC 3325, November 2002.

Karopoulos G, Kambourakis G, Gritzalis S. Caller identity privacy in SIP heterogeneous realms: A practical solution. In: Zanella A, et al., editors. Third workshop on multimedia applications over wireless networks (MediaWin 2008). Marakkech, Morocco: IEEE Computer Society Press; July 2008.

Mealling M. Dynamic Delegation Discovery System (DDDS), Part one: The comprehensive DDDS, RFC 3401, October 2002a.

Mealling M. Dynamic Delegation Discovery System (DDDS), Part two: The algorithm, RFC 3402, October 2002b.

Mealling M. Dynamic Delegation Discovery System (DDDS), Part three: The Domain Name System (DNS) database, RFC 3403, October 2002c.

Mealling M. Dynamic Delegation Discovery System (DDDS), Part four: The Uniform Resource Identifiers (URI) resolution application, RFC 3404, October 2002d.

OpenSSL, Open source SSL/TLS library. Available at 〈http://www.openssl.org/〉.

Peterson J. A privacy mechanism for the Session Initiation Protocol (SIP), RFC 3323, November 2002.

Pfitzmann A, Hansen M. Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management—a consolidated proposal for terminology, version v0.31, 15 Februry 2008, available at 〈http://dud.inf.tu-dresden.de/Anon_Terminology.shtml〉.

Ramsdell B. Secure/Multipurpose Internet Mail Extensions (S/MIME) version 3.1 message specification, RFC 3851, July 2004.

Rescorla E, Modadugu N. Datagram transport layer security, RFC 4347, April 2006.

Rivest R, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 1978;21(2):120–6.

Rosenberg J, Schulzrinne H, Camarillo G, Johnston A, Peterson J, Sparks R, et al. SIP: Session Initiation Protocol, RFC 3261, June 2002.

Rosenberg J, Jennings C. The Session Initiation Protocol (SIP) and spam, RFC 5039, January 2008.

Schulzrinne H, Wedlund E. Application-layer mobility using SIP. Mobile Computing and Communications Review 2000;4(3):47–57 (ACM SIGMOBILE).

SIP Express Router (SER), Free, open source SIP server, available at 〈http://www.iptel.org/ser〉.

SIPp, Open source performance testing tool for SIP, available at 〈http://sipp.sourceforge.net〉.

Twinkle SIP softphone, Open source, available at 〈http://www.twinklephone.com〉.