Complete SIP message obfuscation: PrivaSIP over Tor

Georgios Karopoulos
Institute for the Protection and
Security of the Citizen
Joint Research Centre
georgios.karopoulos@jrc.ec.europa.eu

Alexandros Fakis
Dept. of Information and
Communication Systems Engineering
University of the Aegean
alfa@aegean.gr

Georgios Kambourakis
Dept. of Information and
Communication Systems Engineering
University of the Aegean
gkamb@aegean.gr

Abstract

Anonymity on SIP signaling can be achieved either by the construction of a lower level tunnel (via the use of SSL or IPSec protocols) or by employing a custom-tailored solution. Unfortunately, the former category of solutions present significant impediments including the requirement for a PKI and the hop-by-hop fashioned protection, while the latter only concentrate on the application layer, thus neglecting sensitive information leaking from lower layers.

To remediate this problem, in the context of this paper, we employ the well-known Tor anonymity system to achieve complete SIP traffic obfuscation from an attacker's standpoint. Specifically, we capitalize on Tor for preserving anonymity on network links that are considered mostly untrusted, i.e., those among SIP proxies and the one between the last proxy in the chain and the callee. We also, combine this Tor-powered solution with PrivaSIP to achieve an even greater level of protection. By employing PrivaSIP we assure that: (a) the first hop in the path (i.e., between the caller and the outbound proxy) affords anonymity, (b) the callee does not know the real identity of the caller, and (c) no real identities of both the caller and the callee are stored in log files. We also evaluate this scheme in terms of performance and show that even in the worst case, the latency introduced is not so high as it might be expected due to the use of Tor.

1 Introduction

The demand for private communications is high among businesses, activists, journalists, military, and law enforcement [1]. Nowadays, telecommunication providers increasingly shift their business model towards Voice over IP (VoIP) communications which are more flexible and inexpensive, but with more security and privacy issues compared to traditional communications. One of the most prominent protocols supporting multimedia services is the Session Initiation Protocol (SIP) [2], which is an application layer, text-based, signaling protocol responsible for session management. Despite its popularity, SIP still suffers from privacy issues, two of the most notable of which are (a) user identity, and (b) IP address disclosure. Since SIP signaling messages are in plaintext, an eavesdropper can acquire sensitive data such as: communicating parties' names and affiliations, IP addresses and hostnames, and SIP Uniform Resource Identifiers (URIs). The SIP header fields that reveal these details are mainly `From`, `To`, `Contact`, and `Call-ID`. An example of a SIP message header format is presented in Table 1; here some data like `branch` and `tag` values were omitted for readability.

Apart from SIP signaling messages, the architecture of the protocol per se is also another source of problems. That is, SIP can be employed in either client/server or Peer-to-Peer (P2P) architectures; in both cases the participation of intermediary servers complicates the pri-

Table 1: SIP message header format

```
INVITE sip:al@agn.org SIP/2.0
Via:  SIP/2.0/UDP pc8.agn.org;branch=...
Max-Forwards:  70
To:  Al <sip:al@agn.org>
From:  Geo <sip:geo@agn.org>;tag=...
Call-ID: a84b4c76e66710@pc8.agn.org
CSeq:  314159 INVITE
Contact:  <sip:geo@pc8.agn.org>
Content-Type:  application/sdp
Content-Length:  142
```

vacy problem. Previous work on these issues [3, 4] has shown that several header fields can provide private data to eavesdroppers. While for some data privacy protection is straightforward if the user selects not to provide them, other header data cannot be omitted since they are needed for the correct routing of SIP messages to their final destination.

Ideally, any complete anonymity solution for SIP should be designed and provided in a cross-layer manner. That is, while SIP operates at the application layer, several other sensitive information regarding its signaling inevitably leak from lower layers. For example, while it is possible to conceal the IDs of the communicating parties by applying, say, a pseudonymity scheme to `Via` and `From` headers, the IP addresses of both ends are available to an observer by just tracking the headers of the IP packets conveying SIP messages.

This fact motivated us to think of taking advantage of the services of a generic anonymization system with the aim to apply an holistic anonymity solution for SIP. On the one hand, such a solution seems promising as anonymization systems like Tor [5] are self-reliant, i.e., normally their operation does not hinge on the protocols of upper layers, hence they can be straightforwardly combined with them. Moreover, as discussed further in Section 4.3, such a system is able to protect SIP signaling from a plethora of other type of attacks including timing and collusion ones. However, on the negative side, taking Tor as an example, the problem with SIP is that currently Tor only supports TCP for its transport layer. As a result, although RFC 3261 [2] requires all SIP entities to mandatory implement both UDP and TCP, many real-world VoIP applications rely solely on UDP for latency reasons. So, at least for the time being, this is a serious impediment for VoIP users to enjoy strong anonymity to real-time voice communication. Tunneling of the UDP traffic through Tor does not really solve this issue because the traffic would be encapsulated in TCP. The latency induced by Tor is also known to be quite heavy as the system relays and mixes its traffic via multiple nodes.

In this context, the paper at hand attempts to answer two basic questions: Is SIP over Tor affordable in terms of service time? And if so, to which network hops should be preferably Tor activated in order to achieve a fair balance between the level of anonymity and the time penalty introduced? Also, what is the additional delay if one considers to even anonymize network links that cannot be covered by Tor (think of the first hop between the caller and the outbound SIP proxy or the registrar).

To shed light on the aforementioned questions we implement a proof-of-concept SIP over Tor system and conduct measurements to assess its performance in terms of service times. Also, we combine this system with a purely application layer anonymization solution for SIP to make a decision whether an end-to-end preservation of anonymity is affordable. The results we obtained seem quite promising showing a latency in the vicinity of 2 secs across all scenarios. It should be noted here that our solution, as well as the aforementioned delay, concerns SIP signaling only; thus, media protection should be considered separately.

As discussed in a following section, previous work in the same topic is fragmentary and has only touched upon these issues not considering SIP at all. Therefore, to our knowledge, this is the first work that elucidates on the foregoing issues and provides real results that can be used as a reference towards building truly anonymous VoIP systems.

The rest of this paper is structured as follows. The next section briefly covers PrivaSIP [6, 7], an application layer solution to preserve anonymity in SIP. Section 3 offers a basic background on Tor operation. The joint operation of Tor with PrivaSIP to achieve a high level of user anonymity is addressed in Section 4. Section 5 reports on the experimental testbed, the scenarios, and the results we obtained when running PrivaSIP over Tor. The last section concludes the paper and gives pointers to future work.

2 PrivaSIP

In the search of ways for establishing a system that would protect SIP end-users' privacy, we chose Pri-

vaSIP [6, 7], an identity protection framework for SIP. An alternative method could be [8], which was published after we finished our work and we did not had the time to evaluate it.

PrivaSIP is an application-layer protocol, whose main idea is that each end-user's real 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. In this way, untrusted proxies or malicious users performing traffic analysis, will not be able to eavesdrop or even recover from the corresponding ciphertext the real ID of the user. This way, a basic level of unlinkability (of SIP transactions made by the same user in the course of time) can be retained as well. There are two versions of PrivaSIP one can use, depending on his needs for privacy:

- PrivaSIP-1: providing caller identity privacy
- PrivaSIP-2: providing caller and callee identity privacy.

The first version, PrivaSIP-1, provides only identity privacy for the person making the call. This is achieved by encrypting the username of the caller and using the created ciphertext on every SIP message that will be exchanged between the caller and the callee. To avoid personal information leakage, the display name field on the SIP message is removed. The second version, PrivaSIP-2, provides identity privacy for both the caller and the callee. In PrivaSIP-2 the caller's SIP application encrypts his username and the username of the callee, while removing the display names that appear in all SIP messages. PrivaSIP also varies in the encryption methods that it employs; a user can choose between symmetric or asymmetric cryptography. To highlight this flexibility, different implementations have been previously presented involving symmetric, traditional asymmetric and Elliptic-curve cryptography:

- Symmetric cryptographic algorithm
  - PrivaSIP-1 using AES
- Asymmetric cryptographic algorithms
  - PrivaSIP-1 or -2 using RSA
  - PrivaSIP-1 or -2 using ECIES

In PrivaSIP-AES, 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 the user ID with AES. PrivaSIP-RSA uses the Caller Proxy's and the Callee Proxy's public keys to encrypt the caller user ID and Digest username, and the callee user ID respectively. PrivaSIP-ECIES works in a similar way with the difference of using Elliptic Curve cryptography.

The cost that comes along with PrivaSIP is negligible concerning the privacy features offered, when symmetric cryptography is used. Nonetheless, as argued in [6, 7], a user may perceive a latency ranging between 0.5 to 2 secs for the first SIP message when asymmetric cryptography is chosen, depending on the server load. In subsequent SIP messages this time penalty can be minimized provided that some caching method is used for the correspondence between real and encrypted IDs. In any case, the aforementioned latency occurs only during the setup phase of the call.

3 Tor operation

The Onion Router (Tor) [5] capitalizes on the onion routing [9] mechanism to deliver a low-latency Internet networking protocol designed to anonymize the data relayed through it. As a result, Tor is also employed as a powerful weapon against Internet censorship carried out by governments or by private organizations on behalf of others. BitTorrent and HTTP are well-known to use Tor services to enhance their level of anonymity. Tor traffic passes through a large number (more than 5,000) of relays that are distributed globally and operated by volunteers. Instead of directly communicating with a network entity (e.g., a web server), a "Torified" application channelizes its traffic through other Tor nodes (also known as circuit), thus hiding the end-user's IP address from externals as well as intermediate Tor nodes. Note that due to the use of encryption along the virtual pathway all the intermediary Tor nodes are only aware of their predecessor and successor in the path, but no other nodes in the circuit. In this respect, Tor's main purpose is to protect users' location and usage data from eavesdropping and/or traffic analysis attacks.

Tor architecture includes three main entities: The first one is called Onion Proxy (OP) and runs locally by the end-user whenever a Tor-enabled connection is desired. An OP is in charge of administering connections

triggered by "Torified" user applications and setting up circuits. On the other hand, the Onion Router (OR) has the role of an intermediary node in the circuit. Therefore, it is responsible for relaying information arriving from other onion proxies or routers. The communication between ORs (as well as between OPs and ORs) is protected by means of a TLS tunnel. OPs and ORs share a session key used to protect the exchanged data. That is, an OP creates a circuit gradually by establishing a symmetric key with each OR in the circuit. Thus, a circuit is built hop-by-hop, always commanding the currently endmost node in the circuit to add one more hop. The third entity is that of the Directory Server (DS) used to support Tor network by providing proxies with a current list of the available routers as well as the short-term onion keys that must be used per router to manage the establishment of circuits.

After an OP retrieves from the DS the list of available ORs it proceeds by randomly selecting a set of ORs forming the circuit between the sender and destination node. Given that each OR owns a public/private key pair (onion key), the OP will establish an ephemeral session key with each OR in the circuit by means of a Diffie-Hellman handshake. After that, the sender is ready to initiate anonymous communications with the receiver. All information entering a circuit is in fixed-size cells (data-packets) of 512 bytes. Tor operation mandates all the cells transmitted from an OP toward the receiver to be repeatedly enciphered under the session keys shared with ORs consisting the circuit. This means that each OR in the path only removes a single layer of encryption by decrypting the incoming cell using the session key shared with the initial sender. As soon as the cell content is being validated it is forwarded to the subsequent OR in the path. This way no individual relay is aware of the complete path a given cell has followed.

The OR lying in the edge of the circuit will eventually remove the last layer of encryption having the cell containing the original application data forwarded to the receiver. Similarly, the reverse path is followed in the opposite direction. Each cell arriving from the previous OR in the path, gets enciphered under the session key shared with the destination (sender) and is transmitted to the next hop in the circuit. The destination needs to repeatedly decrypt the cell to acquire the original data. To increase its performance, Tor is also known to employ the same circuit for sessions that take place within the same 10 mins. This quality is of certain significance in our case and it is discussed further in Section 5.

4 SIP Torification

In this section we explain how the combination of PrivaSIP with Tor can be highly profitable in terms of preserving end-users' privacy.

4.1 Issues with PrivaSIP

PrivaSIP provides a more advanced level of privacy compared to plain SIP; however, due to its application-oriented nature it is not a complete solution. The main issue is that, while real user IDs are concealed, IP addresses of communicating users are still visible since they are needed for the proper SIP message routing. Another issue is that end-users' domains are visible for the same reason, but this is rather minor since domains can also be derived from IP addresses.

4.2 PrivaSIP over Tor

The aforementioned issues of PrivaSIP led us to employ Tor to alleviate them. In the rest of the text when we use the term "PrivaSIP", we refer to PrivaSIP-2 which obfuscates both the caller and the callee's real IDs. The advantages of using PrivaSIP over Tor are briefly the following (we will further elaborate on them later in this section):

- third parties cannot mount traffic analysis attacks
- real user IDs are not leaked to intermediate SIP proxies
- log files in intermediate SIP proxies do not contain real user IDs
- the real caller ID is unknown to the callee
- the real caller ID is unknown to the callee's proxy
- the real callee ID is unknown to the caller's proxy
- user authentication and accountability (e.g., for billing purposes) are still supported

A prototype architecture of our proposed scheme is shown in Figure 1. Here we assume that Client A resides in a corporate network and his SIP proxy is placed in the same local network as well. Thus, it is not necessary to protect this communication with Tor; we do protect, however, the real IDs of the end-users from other corporate users by employing PrivaSIP. The traffic

data (a) between Proxy A and Proxy B, and (b) between Proxy B and Client B, are protected from third parties by Tor. To avoid the Tor exit node eavesdropping issue, it is required that Proxy B and Client B act as Tor relays, so that no plaintext SIP messages are transmitted through the Internet.

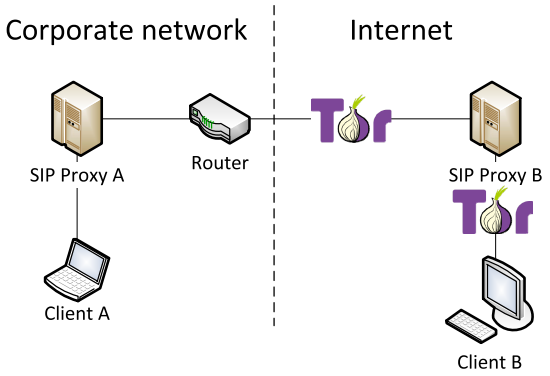

Figure 1: Prototype PrivaSIP over Tor architecture

One major question related to our choice of protocols is why someone would use PrivaSIP over Tor and not plain SIP. The short answer is because PrivaSIP offers a more advanced level of privacy. First of all, since the first hop in the path (i.e., between the caller and the proxy in the corporate network) is not protected by Tor, PrivaSIP assures that the real IDs of the communicating parties are not revealed to the rest of the corporate users. Tor protects the traffic data from third parties; with PrivaSIP we also hide the real ID of the caller from the callee. Last but not least, SIP messages are delivered as plaintext to intermediate SIP proxies; by employing PrivaSIP we allow their proper routing while protecting the real IDs of the communicating parties at the same time. This also has an implication on proxies' log files, where no real user IDs are stored. This works in favor of unlinkability as well, i.e., certain SIP transactions cannot be correlated in such a way that can be traced back to the same user.

Apart from the aforementioned advantages, the main benefit of our proposal is the protection of traffic data offered by Tor. Essentially, Tor solves the inherent issues of PrivaSIP as presented in the previous section. Thus, it prevents traffic analysis attacks by obfuscating traffic data like SIP end-users' IP addresses and domain names. A more detailed analysis of Tor's benefits can be found in the next section.

Another point of discussion is related to the deployment of the proposed solution and its compatibility with existing SIP infrastructures. Regarding PrivaSIP, some modifications are needed in SIP proxies and user clients in order to properly encrypt/decrypt obfuscated user IDs; more information on this point can be found in [6, 7]. Tor, on the other hand, acts as a proxy so it can be transparently utilized by end-users and servers; it is, however, needed to be installed and configured so that the callee's SIP proxy and client act as Tor relays.

4.3 Privacy analysis

As discussed in the previous section, the qualities of Tor can be of great value in SIP. More specifically, enabling SIP over Tor communications can lead to a robust cross-layer privacy preserving system capable of dealing with a variety of major privacy attacks [5, 10, 11]. First off, message size types of attack are avoided. This is because Tor mandates the use of fixed-length cells. As a result, an observer is unable to infer any usable information when examining a cell's length. Moreover, due to the use of encryption, no one is in position to change a cell's coding when in transit through the Tor network. This particular quality also works in favor of protecting from packet context oriented attacks. Simply put, the IP address, application port, etc., included in the TCP header remain well-hidden. In any case, all connections in Tor use TLS link encryption based on ephemeral keys. This way, connections between entities enjoy perfect forward secrecy, preventing medication of data while in transit. For the same reason, attacks aiming on masquerading an OR are also considered unpractical.

Also, observers are blocked from spying on which circuit a given cell is intended for. Periodical and independent rekeying of TLS ephemeral keys imposed by Tor reduces the impact of a potential key leak. Attacks based on collusion of nodes are also considered highly unpractical. This stands true as all information entering the Tor network is routed through a private network pathway of ORs (circuit) where each relay only knows the previous and the next relay. Lastly, Tor is known to generally defeat privacy attacks based on message timing. In fact, to our knowledge, this issue has been already investigated in [12, 13, 14, 15]. Specifically, the work in [12] argues that a timing attack in Tor is feasible under the global attacker model. However, to be profitable, this attack requires the attacker to be able to eavesdrop on all network nodes. Moreover, the authors

in [13, 14] bring into the foreground some viable attacks under the weaker threat model (no global adversary).

These attacks however are known to be repelled if using supportive protection schemes such as adaptive padding or the insertion of cover traffic in a way that the network cannot perceive between the different network streams [13, 14, 15]. Last, but not least, Tor gives the opportunity to the end-user to also join Tor and become an OR by itself (either a relay or a bridge). This case presents two significant advantages. First, it is anticipated to gradually reduce the latency perceived by all the nodes, and also results to a safer network. Secondly, it would end up in a situation in which a malicious user would not be able to distinguish which connection is initiated as a user and which as an OR.

A last remark here, also succinctly pointed out in the previous sections, is that the boundaries of the system do not enjoy protection by Tor. That is, the endmost communication link between the last OR and the callee cannot be protected, and thus an ill motivated entity could eavesdrop on the packet content. In our case, this problem is tackled by requiring the callee's SIP proxy and client to act as Tor relays so that the packet content is not revealed to third parties. For the other end of the communication path, i.e., between the sender and the OP, it is argued that no protection is required as an OP runs locally in the caller's machine. Overall, we can say that the synergistic operation of PrivaSIP with Tor assembles a powerful solution that achieves the protection of end-users' privacy in a cross-layer fashion. Of course, PrivaSIP could be easily co-work with other privacy solutions for TCP/IP layer including MorphMix [16] and Tarzan [17] ones.

5 Performance evaluation

Since Tor is known to cause long delays, our biggest concern is whether the latency in the proposed system is affordable from the user viewpoint. In the following sections we present the architecture used in our experiments and show that the delay perceived by the end-users is relatively low, within the range of 1.8 to 2 seconds in the worst case. This time penalty is of course in absolute relation with the protection of users' privacy and more specifically the preservation of their anonymity. As a reference, in previous works [6, 7] the delays were approximately 0.5 second for plain SIP and 1.2 second for PrivaSIP-2-RSA. Those measurements, however, were taken with a different hardware setup and server traffic and cannot be directly compared with the present results.

5.1 Architecture

The architecture used for the experiments is a simplified interpretation of a real case, described in the following and depicted in Figure 1. Alice (Client A) wants to call Bob (Client B). Note that Alice is registered to a domain served by SIP Proxy A which is situated within a corporate network. Since Alice and SIP Proxy A belong to the same network, which is considered trusted in the normal case, we avoid using Tor on this link. However, Tor is employed outside the corporate network, that is, both between the two SIP proxies, and SIP Proxy B and Bob.

First, we describe the underlying hardware used in our testbed. All servers and clients were hosted on Virtual Machines. To host those VMs we chose Okeanos[1], a cloud service provided for the Greek Research and Academic Community. With Okeanos, we had the ability to create VMs with dual core processors, 4 GB of RAM, and 60 GB storage. Okeanos provides high speed Internet connection to its users which can reach up to 520 Mbps. There is also the possibility to choose between numerous operating systems; for our scenarios we chose CentOS 6 for the servers, and Ubuntu 12.04 for the clients.

Both SIP proxies are based on SER 0.9.6[2], and modified accordingly as in [6, 7] to support PrivaSIP. For traffic generation on the caller side, SIPp[3] was used, along with sipsak[4]. On the callee side, a modified User Agent (UA) that supports PrivaSIP was used, based on Twinkle software phone[5].

5.2 Results

In this section we describe the procedures followed and the metrics used during the experiments, as well as the results obtained. First off, Bob's UA acting as the caller sends an INVITE message using PrivaSIP-2 to conceal both the caller's and the callee's IDs using RSA. As a consequence, the caller first receives back a 100 (TRYING) message followed by a 180 (RINGING) signifying that Alice's phone is ringing. In this context,

[1] https://okeanos.grnet.gr
[2] http://www.iptel.org/ser
[3] http://sipp.sourceforge.net
[4] http://sourceforge.net/projects/sipsak.berlios
[5] http://www.twinklephone.com

we measure the time starting from when the INVITE was send until the RINGING message is being received back on the caller side. This time includes the operations needed for PrivaSIP, as well as delays imposed by Tor. We should note here that we take the worst case scenario for Tor delay. Since Tor utilizes the same circuit for sessions that take place within the same 10 mins or so, we force each new call to be placed over a new circuit.

For the sake of comparison, we measured call delays for two scenarios: (a) plain SIP over Tor, and (b) PrivaSIP over Tor. These two scenarios do not have the same level of privacy protection, so they cannot be directly compared. The reason we chose them is to identify the sources of delays since more than one protocols are involved. In each scenario we followed the aforementioned procedure and 100 calls were sequentially produced with the help of SIPp tool. Using Wireshark on the caller's side we were able to compute all call delays. The derived values were rounded to one decimal digit and the frequency per value was counted.

For the plain SIP scenario, the results are summarized in Table 2 and graphically presented in Figure 2. As it is easily observed from the table, the majority of the delays span between 0.9 and 1.1 seconds. On the other hand, for the PrivaSIP scenario, the results are presented in Table 3 and Figure 3. In this case, the majority of the delays are between 1.8 and 2 seconds, leading to the conclusion that the perceived delay from the end-users while waiting for the call to be established is relatively low. What these results show us is that PrivaSIP together with SIP operations adds a delay of approximately 1 sec to the whole scheme, and the rest is caused by Tor. Even if the plain SIP scenario has better performance, the one employing PrivaSIP is preferred, since it comes with more advanced privacy preserving features. In any case, both of these methods can be be offered to the end-users in an opt-in fashion.

Table 2: Range of call delays for plain SIP over Tor

Delay (sec)	**Frequency**
0.6	3
0.7	5
0.8	2
0.9	27
1	43
1.1	18
1.2	1
1.3	0
1.4	1

Table 3: Range of call delays for PrivaSIP over Tor

Delay (sec)	**Frequency**
1.5	1
1.6	0
1.7	3
1.8	26
1.9	37
2	27
2.1	0
2.2	1
2.3	0
2.4	2
2.5	2
2.6	0
2.7	0
2.8	0
2.9	0
3	1

6 Related work

There is one group of mechanisms based on PGPfone and its successor, Zfone. Zfone[6] is a software for secure VoIP communications, using the ZRTP protocol. Zfone works on top of existing SIP and RTP programs and the security it offers is limited to encrypting and decrypting voice packets only. Moreover, it seems that since 2009 its development has stopped.

[6] http://zfoneproject.com

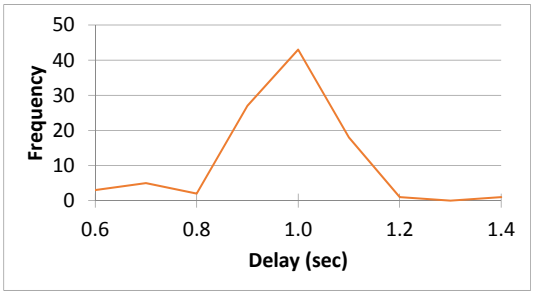

Figure 2: Frequency of call delays for plain SIP over Tor

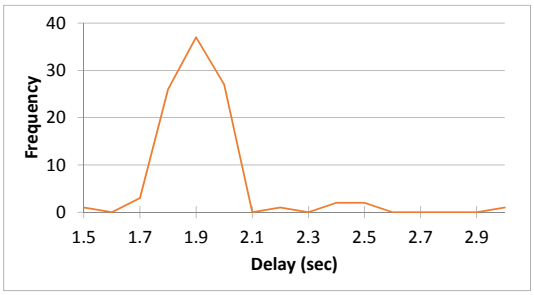

Figure 3: Frequency of call delays for PrivaSIP over Tor

The system that is closer to our proposal is TORFone[7], which is a product that is independent from the Tor Project. It is based on Tor and PGPfone and since the connection between users passes through six transit nodes located anywhere in the world the voice latency can reach 4-5 secs. Apart from that, the main issue with TORFone is that it is based on outdated security software (PGPfone is unmaintained since 1999) and it does not use standard protocols, like SIP. Silent Circle[8] is a company offering secure communications to its subscribers. The connection between the caller and the Silent Network, as well as the connection between the Silent Network and the callee, are both encrypted provided that both users are subscribers of the service. If one of the two users is not a subscriber then the respective connection is not encrypted. Its operation is based on the ZRTP protocol and it mostly protects from Man-in-the-Middle (MiTM) attacks without taking care of end-users' identity privacy.

RedPhone[9] uses a signaling protocol that is custom to RedPhone and the voice traffic is encrypted using ZRTP. Its signaling protocol is similar to HTTP and protected by TLS. However, since TLS is used between clients and relay servers, the communicating parties are identifiable by their IP addresses. Jitsi[10], formerly known as SIP Communicator, supports many protocols including SIP, Jabber/XMPP (GoogleTalk and Facebook), MSN, ICQ etc., and offers call encryption to SIP and those protocols that are based on Extensible Messaging and Presence Protocol (XMPP) through ZRTP. For SIP, secure signaling with the use of TLS is also supported. Nevertheless, the same problem still stands; the communicating end-users are identifiable by their IP addresses. The Open {Secure, Source, Standards} Telephony Network (OSTN) project[11] is foreseen to provide an end-to-end secure VoIP service based on open standards. SIP signaling is protected with SSL, while voice traffic is secured by ZRTP. As with other solutions, the issue with this system is that end-users are not protected by traffic analysis attacks.

There is also a group of solutions that are based on proprietary protocols and/or are unsupported, like Speak Freely[12] (since 2002) and the "I Hear U" (IHU) project[13] (since 2008). GSMK CryptoPhone[14] provides a range of hardware and software products including mobile, landline and satellite phones, softphones, and even a crypto PBX. It is a proprietary solution and the source code is offered for download; however, it seems outdated since the last version is from 2003, and no safe claims can be made about its secure operation. Mumble[15] is an open source, voice chat software mainly intended for use while gaming. Communication to and from the server is protected through encryption which is mandatory and cannot be disabled. The voice as well as the control channel, which transports chat messages and other non-time critical information, are both encrypted. The latter is protected by TLS making IP address dis-

[7] http://torfone.org/index.html
[8] https://silentcircle.com
[9] https://github.com/WhisperSystems/RedPhone/wiki
[10] https://jitsi.org
[11] https://ostel.me
[12] http://www.speakfreely.org
[13] http://ihu.sourceforge.net
[14] http://www.cryptophone.de
[15] http://mumble.sourceforge.net

covery of end-users possible.

Nautilus Secure Phone[16] protects only voice traffic and does not deal at all with traffic analysis attacks. Web Real-Time Communication (WebRTC)[17] is an API definition from the W3C Consortium to enable browser-to-browser applications including voice calling and video chat. It does not explicitly tackle with IP privacy since in "WebRTC Security Architecture"[18] it is stated that it is out of scope.

7 Conclusions

Few will argue that the preservation of user anonymity is an important issue which pertains to almost any protocol or technology deployed in the wired or wireless internet. When it comes to VoIP, it is for sure that not only a broad category of users would highly appreciate anonymous communications, but also several providers would value such a service towards expanding their market share. Unfortunately, all works proposed so far in the literature - either standardization efforts or custom-made solutions - tackle this problem in an unsatisfactory way. Some of them focus solely on the application layer, thus neglecting sensitive data leaking from lower layers, while others propose inflexible or difficult to deploy mechanisms that either require external infrastructures or are in direct contrast with user accountability. On the other hand, secure tunneling of VoIP traffic by means of, say, an SSL connection is considered mostly unpractical.

Compelled by this fact we came with the idea of taking advantage of the well-known Tor anonymization system to achieve complete SIP message privacy. While this may be seem straightforward it is quite tricky because the majority of SIP apps are designed to operate over UDP for increased performance. Tor on the other hand works solidly over TCP and introduces additional delays due to the use of public-key cryptography, and complex segmentation, encryption, and routing of its messages via a random set of nodes. Therefore, the main impediment here is performance in terms of service times. In this context, our experiments employing different setups showed that SIP over Tor is quite affordable as it adds a time penalty that fluctuates between 1.8 and 2 secs in the vast majority of cases. Overall, we think that the extra security and privacy gains that Tor brings along compensate for this penalization (which after all concerns the establishment of the call and not the multimedia session itself).

This work can be used as a reference towards building anonymization solutions that consider privacy in a cross-layer fashion. This is not only bound to VoIP applications but also for other protocols as that in [18], where its authors have already identified this need and provide the necessary background towards a full-fledged solution.

Our intention is to extend this work by conducting further experiments approximating real SIP traffic and finding ways to lessen this ~ 2 secs overhead through proper optimizations. Future work could also consider more complex VoIP infrastructures and scenarios with more users, services and intermediate SIP elements.

Acknowledgment

This paper is part of the 5179 (SCYPE) research project, implemented within the context of the Greek Ministry of Development-General Secretariat of Research and Technology (GSRT) funded program "Excellence II/ Aristeia II", co-financed by the European Union/European Social Fund - Operational program "Education and Life-long Learning" and National funds.

References

[1] G. Kambourakis, "Anonymity and closely related terms in the cyberspace: An analysis by example," *Journal of Information Security and Applications*, 2014.

[2] 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), June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630, 5922, 5954, 6026, 6141, 6665, 6878.

[3] J. Peterson, "A Privacy Mechanism for the Session Initiation Protocol (SIP)." RFC 3323 (Proposed Standard), Nov. 2002.

[4] C. Shen and H. Schulzrinne, "A VoIP privacy mechanism and its application in VoIP peering for

[16] http://nautilus.berlios.de

[17] http://www.webrtc.org

[18] http://tools.ietf.org/html/draft-ietf-rtcweb-security-arch-08

voice service provider topology and identity hiding," arXiv e-print 0807.1169, July 2008.

[5] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-generation Onion Router," in *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, (Berkeley, CA, USA), pp. 21–21, USENIX Association, 2004.

[6] G. Karopoulos, G. Kambourakis, S. Gritzalis, and E. Konstantinou, "A framework for identity privacy in SIP," *Journal of Network and Computer Applications*, vol. 33, pp. 16–28, Jan. 2010.

[7] G. Karopoulos, G. Kambourakis, and S. Gritzalis, "PrivaSIP: ad-hoc identity privacy in SIP," *Computer Standards & Interfaces*, vol. 33, pp. 301–314, Mar. 2011.

[8] N. Vrakas, D. Geneiatakis, and C. Lambrinoudakis, "Obscuring users' identity in voip/ims environments," *Computers & Security*, vol. 43, no. 0, pp. 145 – 158, 2014.

[9] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr, "Towards an analysis of onion routing security," in *Designing Privacy Enhancing Technologies* (H. Federrath, ed.), vol. 2009 of *Lecture Notes in Computer Science*, pp. 96–114, Springer Berlin Heidelberg, 2001.

[10] N. Hopper, E. Y. Vasserman, and E. Chan-tin, "How much anonymity does network latency leak?," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, pp. 82–91, ACM, 2007.

[11] O. Berthold, H. Federrath, and M. Köhntopp, "Project "anonymity and unobservability in the internet"," in *Proceedings of the Tenth Conference on Computers, Freedom and Privacy: Challenging the Assumptions*, CFP '00, (New York, NY, USA), pp. 57–65, ACM, 2000.

[12] G. Danezis, "The traffic analysis of continuous-time mixes," in *Privacy Enhancing Technologies*, pp. 35–50, 2004.

[13] V. Shmatikov and M.-H. Wang, "Timing analysis in low-latency mix networks: Attacks and defenses," in *ESORICS*, pp. 18–33, 2006.

[14] R. Wiangsripanawan, W. Susilo, and R. Safavi-Naini, "Design principles for low latency anonymous network systems secure against timing attacks," in *ACSW Frontiers*, pp. 183–191, 2007.

[15] J. Feigenbaum, A. Johnson, and P. F. Syverson, "Preventing active timing attacks in low-latency anonymous communication," in *Privacy Enhancing Technologies*, pp. 166–183, 2010.

[16] M. Rennhard and B. Plattner, "Practical anonymity for the masses with morphmix," in *Financial Cryptography*, pp. 233–250, 2004.

[17] M. J. Freedman and R. Morris, "Tarzan: a peer-to-peer anonymizing network layer," in *ACM Conference on Computer and Communications Security*, pp. 193–206, 2002.

[18] F. Pereñiguez-Garcia, R. Marin-Lopez, G. Kambourakis, A. Ruiz-Martinez, S. Gritzalis, and A. Skarmeta-Gomez, "Kamu: providing advanced user privacy in kerberos multi-domain scenarios," *International Journal of Information Security*, pp. 1–21, 2013.