Electronic Commerce Research (2019) 19:655–687
https://doi.org/10.1007/s10660-019-09369-0

A survey on anonymous voice over IP communication: attacks and defenses

Ge Zhang · Simone Fischer-Hübner

Published online: 12 August 2019

Abstract Anonymous voice over IP (VoIP) communication is important for many users, in particular, journalists, human rights workers and the military. Recent research work has shown an increasing interest in methods of anonymous VoIP communication. This survey starts by introducing and identifying the major concepts and challenges in this field. Then we review anonymity attacks on VoIP and the existing work done to design defending strategies. We also propose a taxonomy of attacks and defenses. Finally, we discuss possible future work.

Keywords SIP · VoIP · Anonymity · Privacy

1 Introduction

Realtime voice communication has been served over closed circuit-switched network infrastructures since the invention of the telephone. With the increasing popularity and widely deployment of the packet-switched Internet data network in the past two decades, we see a tendency to combine both voice and data networks on an all-IP network basis. Voice over IP (VoIP) is an example.

"On the Internet, nobody knows you're a dog." [30], is a statement indicates the public's perception that the Internet provides a certain level of anonymity in communication. VoIP users have reasons to believe that their identities are well protected as (1) They use pseudonyms for communication; (2) The communication is encrypted; (3) The communication is relayed by a trusted third party (e.g. a service provider or a

G. Zhang (✉)
Karlstad University, Karlstad, Sweden
e-mail: ge.zhang@kau.se

S. Fischer-Hübner
e-mail: simone.fischer-huebner@kau.se

Peer-to-Peer (P2P) overlay network). However, quite to the contrary, recent exposure on the NSA PRISM Surveillance Program [18,51] shows how easily "big brothers" breaks these obstacles on the Internet to monitor their citizens, or even those people who live in other countries. Since the anonymity provided by the Internet is so vulnerable, we believe that adversaries are indeed capable of identifying the VoIP communication partners. As a result, the privacy of VoIP users is at risk.

As VoIP deployment increases, protection against de-anonymization attacks is becoming a necessity. From a business aspect, anonymous VoIP communications can also be a value-added service [6,7,36,49,56]. Citizens may want to report criminal evidence without worrying about revenge by criminals. Companies may want to talk with their business partners without being noticed by their competitors. Private users may not want their VoIP service providers to sell their personal calling records to marketing companies. Recently there has been research on privacy attacks on VoIP and technical solutions for anonymous VoIP communications, but no comprehensive survey exists.

The objective of this paper is twofold: First we provide a comprehensive overview of current anonymous VoIP communication research to new researchers in this field. Second we analyze and compare those works, and then identify their gaps. The survey begins with providing the necessary background, including VoIP protocols, and the taxonomy of VoIP architectures. We conduct a survey on some of the proposed attacks to identify the communication partners. We also review the existing work done to design anonymous VoIP communication services. Finally, we discuss the major open problems in anonymous VoIP communication and possible directions for further research.

2 Background of VoIP

We first briefly describe protocols, architectures and Quality of Service (QoS) requirements of VoIP systems.

2.1 Protocols and architectures

2.1.1 Protocols

Voice over IP (VoIP) enables realtime voice communication over IP networks. A VoIP system has two basic functions: (1) A *signaling* function is designed to establish, modify and terminate a conversation; (2) a *media* transmission function is used to carry voice traffic. For the implementation of the two functions, there exists both standard protocols and proprietary protocols. Here we introduce one example for each.

– Standard protocols [by Internet Engineering Task Force (IETF)]: They are Session Initiation Protocol (SIP) [38] and Realtime Transport Protocol (RTP) [39]. SIP is a text-based protocol with HTTP-similar messages format. A SIP message can take a Session Description Protocol (SDP) [19] message as payload to negotiate the session parameters (e.g. preferred codec) between communication partners.

SIP users are identified using Uniform Resource Identifiers (URI) [4], a universal string with a pair of domain name and a user name registered for this domain (e.g. sip:ge.zhang@kau.se). SIP messages are suggested to be protected using TLS [15], IPSec [22] or S/MIME [34]. RTP protocol defines the format of packets for voice content delivery. Besides voice content, an RTP packet can also carry user button-click events to indicate that the button has been pressed [40]. This enables a user to interact with an Interactive Voice Response (IVR) server. RTP packet payload can be encrypted using SRTP [3] mechanisms.
- Proprietary protocols [by Skype]: Skype [43] is a popular VoIP service provider. To create an account, Skype users can freely select a username which has not been taken by others yet. The details of its signaling and media transmission protocols are not available to the public. On its homepage [44], Skype announces that it employs the Advanced Encryption Standard (AES) algorithms with a maximum 256-bit length key to protect users' communications.

For a given VoIP call, we define a sequence of packets for its signaling function as *a signaling flow* and correspondingly, those for its media transmission function as *a media flow*.

2.1.2 Entities and architectures

A VoIP network consists of different entities, like *Server* and *User Agent (UA)*. A server provides services, like locating users and relaying traffic, etc. A UA is a user's equipment to make or answer calls. It can generate signaling messages on behalf of its owner. During a conversation, a UA encodes its owner's speech signals into media packets and sends them to the communication partner, who will recover speech signals from media packets. However, a prerequisite is that they must have been already negotiated the same codec for encoding/decoding. Two codec properties are related to this survey:

- Silence suppression: It allows discontinuous voice packets transmission [63], which is a capability to recognize the silent periods and to stop producing media packets during these periods. Thus bandwidth can be saved with little performance impact. If silence suppression is not applied, the media packets are generated constantly with a fixed time interval (e.g. 20 ms).
- Coding bit rates: Two types of coding bit rates can be distinguished: *Fixed Bit Rate (FBR)* and *Variable Bit Rate (VBR)*. FBR codec (e.g. G.711) employs a fixed codebook with constant bit rate. Thus the generated media packets are the same packet size. On the other hand, VBR codec (e.g. Speex) can employ an adaptive codebook with variable bit rate. It exploits the fact that some sounds are easier to represent than others. For instance, fricative sounds require lower bit rates than vowels. Thus the fricative sounds need fewer bits to be encoded to save bandwidth. In this way, UAs produce media packets with different packet sizes.

The sequences of media flow packets count and sizes are highly dependent on the audio signal. Figure 1 shows a piece of audio signal with its corresponding sequence of media flow packets count (an FBR codec with silence suppression set on) and its

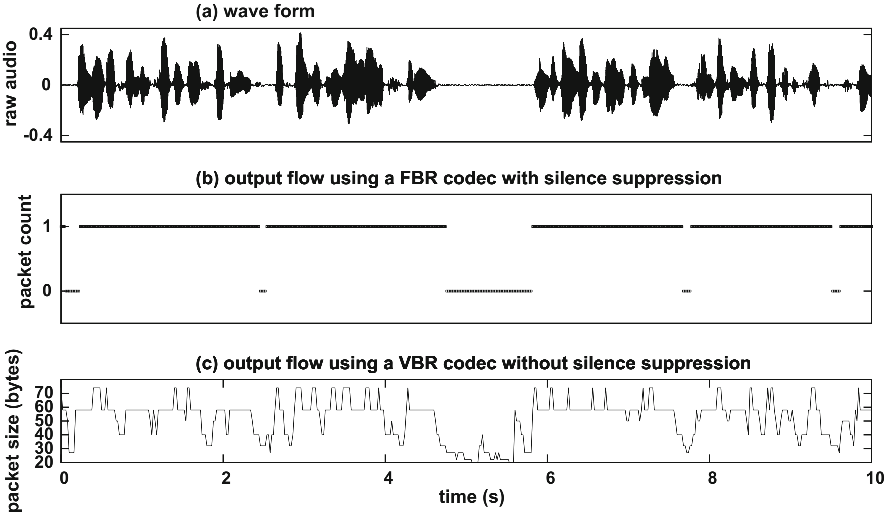


Fig. 1 A piece of audio signal with its corresponding sequence of media flow packets count (a FBR codec with silence suppression on) and its corresponding sequence of media flow packet sizes (a VBR with silence suppression off)

corresponding sequence of media flow packet sizes (a VBR with silence suppression set off). From Fig. 1a, b, we can see that the UA stops generating media packets when the input audio signal is weak. From Fig. 1a, c, we can observe that the stronger the input audio signal, the larger the media packets will be.

With regard to network topology, the architecture of a VoIP system is either Client/Server (C/S) or Peer-to-Peer (P2P), as discussed as follows:

- Client/Server (C/S): In this architecture, there are servers deployed to provide different services (e.g. user location, traffic relay, session management, etc) to users. The users rely on the servers to build conversations. An overview is illustrated in Fig. 2: the signaling flow and media flow between u_1 and u_2 are relayed by the servers.[1]
- Peer-to-Peer (P2P): In a P2P architecture, a user relies on other peer nodes for the services. An example of this architecture is illustrated in Fig. 3: The flow may go through several peers before it arrives at its final destination. The peers are selected according to a route selection algorithm. Here we introduce two basic ones:
  - Shortest route selection: To find the callee, the caller broadcasts a router setup request to all her neighbor peers. The request contains the identity of the callee. A neighbor peer should drop the request if it has been received recently. Otherwise, the peer checks the request to see whether it contains her identity. If it does, the peer is the callee and the request is terminated here. Otherwise the peer continue to broadcast the request. This algorithm ensures the route between the

[1] Media flows can be built end-to-end. However, in many cases, service providers need to relay them because: (1) A relay can help NAT traversing for the users who only have private IP addresses; (2) It is easy to do session management (e.g. billing and QoS management).

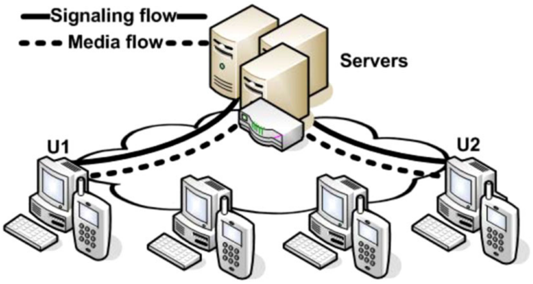


Fig. 2 VoIP C/S architecture overview

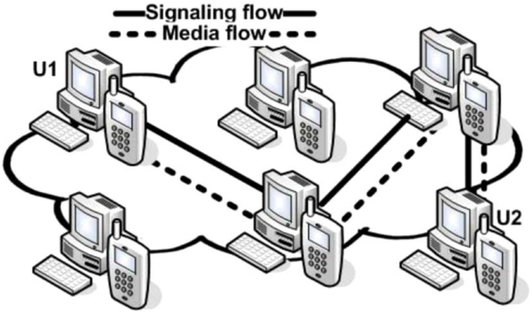


Fig. 3 VoIP P2P architecture overview

caller and the callee is the shortest one to minimize the end-to-end delay. It is good for call performance. Nevertheless, attackers may use their knowledge of network topology to trace the caller [46]. This problem will be described in Sect. 4.4.

- Random selection: The caller sets up a route to the callee by randomly selecting several peers in the network. This way is similar to the one used in Tor [16]. Since the peers are selected at random, less information is leaked to attackers. However, the end-to-end latency is not taken into account for the selected route. Thus the performance over the route might not be acceptable for voice conversation.

2.1.3 Quality of Service (QoS) requirements

The transmission of VoIP flows is QoS sensitive. Three issues are frequently taken as criteria for evaluation:

- End-to-end delay: It is the time interval between encoding a media packet at the sender and decoding it at the recipient. According to [35], users will notice a significant hesitation in their partners' response if the end-to-end delay is above 250 ms.
- Delay jitter: It refers the variation of packet interarrival time. It is caused by network congestion and improper routing during the transmission of media packets. A solution is to buffering received packets to recover the original order. However, waiting for packets that are buffered introduce more end-to-end delay.

- Packet loss: Packets might be accidently dropped in packet switched networks. Fortunately, the loss of a small amount of packets will not prevent users from understanding of the whole conversation. Thus VoIP applications can endure a certain level of packet loss.

3 Terminology and VoIP anonymity

The section introduces the terminology of anonymity. It also defines the anonymity requirements as well as threats. Finally it shows the possible adversary models on C/S and P2P architectures.

3.1 General terminology of anonymity

Pfitzmann et al. [32] defined terms for anonymity, unlinkability, unobservability and pseudonymity. These definitions have been used in much research work and have been continuously updated[2]:

- Anonymity: "*Anonymity* of a subject means that the subject is not identifiable within a set of subjects, the anonymity set." Thus it is necessary to have a set of subjects with the same attributes to achieve the anonymity to hide the subject, and the set is the *anonymity set*. Generally, increasing the size of the anonymity set can help to enhance the degree of anonymity. For instance, to protect an individual's privacy in a database, one can generalize attributes until each row is not identifiable within at least $k-1$ other rows, thus the anonymity set size is increased to k. This property is called k-anonymity [48], which has also been extended to measure privacy of communication and other areas besides database privacy [47].
- Unlinkability: "*Unlinkability* of two or more items of interest (IOIs, e.g. subjects, messages, actions, $\cdots$) from the attacker's perspective means that within the system (comprising these and possibly other items), the attacker cannot sufficiently distinguish whether these IOIs are related or not." Unlinkability means that the ability of attackers to distinguish one IOI from another does not increase after observing the system. *Sender anonymity* means each message is unlinkable to the message sender. *Recipient anonymity* means each message is unlinkable to the message recipient. *Relationship anonymity* means the sender of a message cannot be linked with the message recipient, e.g. one cannot find out who is communicating with whom. Relationship anonymity is a weaker property since it can be met only when either sender anonymity or recipient anonymity is achieved.
- Unobservability: Unobservability in a network means that an attacker cannot observe whether a communication is taking place or a service is being used. For instance, given the fact that a user sends a particular message, an attacker does not know whether anyone has performed the "sending" action.
- Pseudonymity: "*Pseudonymity* is the use of pseudonyms as identifiers". There are different pseudonyms: A *person pseudonym* is a substitute of the owner's real

[2] http://dud.inf.tu-dresden.de/Anon_Terminology.shtml

name for multi-purposes (e.g. a social security number). A *role pseudonym* is only applied to specific purposes (e.g. a club member registration number). A *transaction pseudonym*, like an one-time identity, is used and only valid in one transaction. Transaction pseudonyms cannot be linked with each other and thus enable the strongest degree of anonymity while person pseudonyms provide the weakest degree of anonymity and the highest degree of linkability.

3.2 Techniques and implementations for anonymous communications

For realizing anonymous communication, Chaum [8] introduced the *"mix net"* concept. A mix net consists of a chain of so-called mixes that function as forwarding proxies to hide the relationships between message senders and recipients. The sender encrypts the message with different layers of public key encryptions by using the public keys of each mix in the chain in reverse order (i.e. starting with the encrypting the message with the public key of the last mix and ending with the public key of the first mix in the chain) and then sends the message to the first mix node in the chain. Each layer of encryption with the public key of a mix also includes information about the address of the next mix in the chain or (in case of the last mix) of the recipient. Each mix node along the chain removes replays, decrypts one layer of the encryption with its private key and withholds the message until it receives several messages from other users or mixes. Then the mix node reorders the messages and flushes them together to the next mix node or the recipient. By multi-layer cryptography, the appearance of a message is changed by the decryptions performed by each "mix" node, which prevents that messages can be traced along the path by an adversary by content correlations. In addition, since incoming messages are collected and sent out in a different order by a mix node, tracing of messages along the path between sender and recipient by time correlations are prevented.

Based on the "mix" concept, there are some implementations for different communication applications in the real world. Mixminion [14] is designed for high latency applications, such as e-mail. It waits for several messages arriving at the mix before forwarding them all together in random order. Since e-mail is not a time sensitive application, the latency caused by waiting messages will not interrupt the service. However, it is not the case for low latency applications, like web surfing and telephones. In these applications, delaying messages will annoy users or even interrupt the services. AN.ON [5] and Tor [16] are designed for connection oriented low latency applications like web surfing or FTP, while ISDN-Mixes [33] were proposed for traditional telephone communications.

Dummy messages can be used to achieve sender unobservability, in combination with message broadcast to achieve recipient anonymity. Such a system enforces all its users to constantly send dummy messages to a proxy node with a fixed sending rate. A dummy message is a garbage message with random content. If the users have a meaningful message to send, they send the meaningful messages to replace the dummy ones. The "mix" node knows whether a received message is a meaningful one or a dummy one by trying to decrypt it. It drops dummy messages and forwards meaningful ones. An adversary cannot say who sends the meaningful messages. On the other hand,

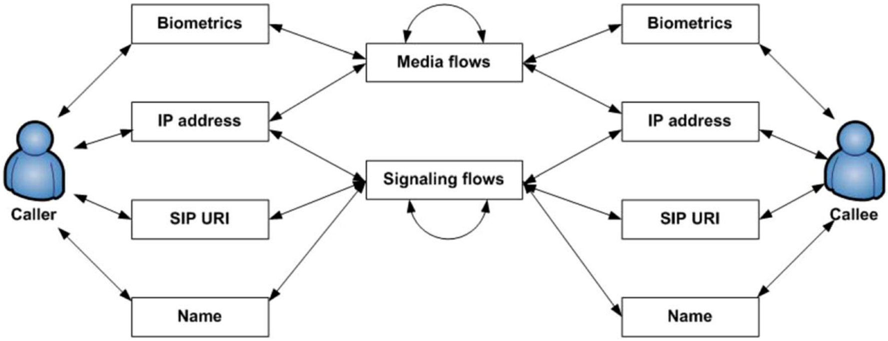


Fig. 4 Linkability of IOIs in VoIP communications

a proxy node can broadcast received messages to all users, but only the actual recipient is able to decrypt and read the message. Thus, an adversary cannot distinguish who is the real recipient. Nevertheless, both broadcasting and dummy messages consume considerable bandwidth, especially, if there are a number of users in the system.

3.3 VoIP linkability and anonymity

We have analyzed the linkability between IOIs in VoIP communications in [57]. Illustrated in Fig. 4, a VoIP user has identity attributes, which are linkable to VoIP traffics. Biometrics (e.g. speech language, pause duration and frequency) are human-specific information that can be profiled from media flow features like packet sizes and time interval. IP address and SIP URI, deemed as role pseudonyms assigned by service providers, appear in packet headers or signaling messages in plain text. In addition, VoIP traffic itself is inter-linkable. For instance, a VoIP flow (a signaling or media flow) may be relayed and separated into several *flow legs* to enhance unlinkability: Given one eavesdropped flow leg, an adversary can at most locate the IP address of either the caller or the callee but not both of them because at least one side of the flow leg is a relay. Nevertheless, later we show that inappropriate design may make flow legs linkable by some flow features.

Since a VoIP user sends packets (e.g. for speaking) and receives packets (e.g. for hearing other's talk), she acts as both sender and recipient. Thus, solely achieving either sender anonymity or recipient anonymity does not guarantee the protection of her identity. Considering this, we define a VoIP user as anonymous if she can achieve both sender anonymity and recipient anonymity.

3.4 Attacks on VoIP anonymity

For wiretapped VoIP traffic in a VoIP system with n users, an adversary's objective is to determine the caller or the callee. Let us assume that the adversary first needs to find the caller, then initially all the n users are the potential candidates, the set of which is *the initial caller anonymity set* S_{init} with the size of n. In this case, each user has an equal probability ($\frac{1}{n}$) to be the caller. Unfortunately, by observing the system with an attacking method, the adversary can gain additional knowledge, which may make the users to appear to be senders for the adversary with unequal probabilities. Thus the adversary could suspect that the user with the highest probability is the caller. To evaluate the accuracy, a *detection rate (accuracy rate or identification rate)* is the most frequently used method to indicate the ratio of the number of successful guesses to the number of attempts. Some works also used statistical classification techniques: A *true positive (tp)* is defined as correctly recognizing the real user's data. A *true negative (tn)* indicates correctly not classifying the imposter's data as the real user's data. A *false positive (fp)* is defined as mistakenly taking the imposter's data as the real user's; and A *false negative (fn)* refers mistakenly classifying the real user's data into the imposter's class. Further concepts based on those classification techniques are calculated as follows,

$$
\begin{aligned}
detection\ rate &= \frac{tp+tn}{tp+tn+fp+fn}\\
true\ positive\ rate &= \frac{tp}{tp+fn}\\
true\ negative\ rate &= \frac{tn}{tn+fp}\\
false\ positive\ rate &= \frac{fp}{tn+fp}\\
false\ negative\ rate &= \frac{fn}{tp+fn}\\
precision &= \frac{tp}{tp+fp}
\end{aligned}
$$

Let us assume that there is a scenario in which the first four calls are made by user a and the last three calls are made by user b, denoted as $\{a, a, a, a, b, b, b\}$. If the adversary guesses the result as $\{a, a, a, b, b, a, a\}$, the detection rate is $\frac{4}{7}$ for user a. The true positive rate, true negative rate, false positive rate, false negative rate and precision of user a are $\frac{3}{4}$, $\frac{1}{3}$, $\frac{2}{3}$, $\frac{1}{4}$and$\frac{3}{5}$ respectively.

The false rates, including false positive rate and false negative rate, usually vary depending on the parameters of an attack. The equal error rate (EER) is the crossover point at which the false positive rate equals the false negative rate. The lower the EER, the better performance for identification. In addition, there is also a trade-off between precision and the true positive rate. *F-measure*, as a harmonic mean of true positive rate and precision, is calculated as

$$
F\text{-}measure = 2 \times \frac{true\ positive\ rate \times precision}{true\ positive\ rate + precision}
$$

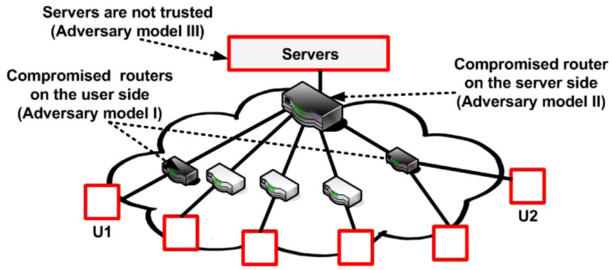


Fig. 5 Attack scenario on VoIP C/S architecture

3.5 Adversary models

We consider the following adversary models:

- Model I: The adversary controls the routers which connect to some UAs. For instance, the adversary is the Internet Service Provider (ISP) of the users.
- Model II: In a C/S VoIP system, the adversary controls the router which connects to the servers. For instance, the adversary is the Internet Service Provider (ISP) of the service provider.
- Model III: In a C/S VoIP system, the server is not trustworthy. For instance, the software on the server may have a backdoor.
- Model IV: In a P2P VoIP system, the adversary controls some UAs in the network. For example, they are “decoy” peer nodes that have been intentionally deployed by the adversary.

When a network entity (e.g. a router or a UA) has been controlled by the adversary, the adversary can do passive attacks or active attacks on the flows going through this compromised entity:

- Passive attacks: The adversary can eavesdrop and read the packets of the flows.
- Active attacks: The adversary can modify, drop, delay, insert or replay the packets of the flows.

In addition, we assume that an adversary does not have the following abilities:

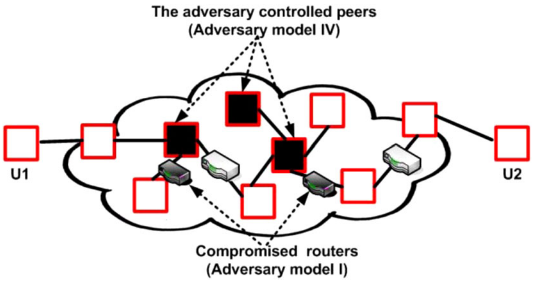
- The adversary cannot break the cipher protections on traffic. Thus, the encrypted traffic content is not available to the adversary in clear text.
- The adversary might control UAs of some users, but the adversary cannot control UAs of all users.

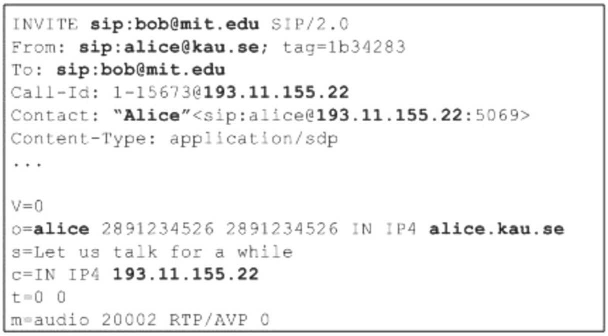
Figures 5 and 6 illustrate examples of the abilities of an adversary on C/S and P2P architecture respectively.

4 Survey of VoIP anonymity attacks

In this section, we provide a survey on possible threats that have been studied earlier by us and others. We classify them into four categories (attacks based on unencrypted sig-

Fig. 6 Attack scenario on VoIP P2P architecture





```
INVITE sip:bob@mit.edu SIP/2.0
From: sip:alice@kau.se; tag=1b34283
To: sip:bob@mit.edu
Call-Id: 1-15673@193.11.155.22
Contact: "Alice"<sip:alice@193.11.155.22:5069>
Content-Type: application/sdp
...

V=0
o=alice 2891234526 2891234526 IN IP4 alice.kau.se
s=Let us talk for a while
c=IN IP4 193.11.155.22
t=0 0
m=audio 20002 RTP/AVP 0
```

Fig. 7 An example SIP message, with bold texts marking identity related information

naling messages, biometrics profiles, flows correlation and topology analysis) according to their fundamental attack principle. For each attack method, we list its prerequisite and leaked identity related information.

4.1 Attacks based on unencrypted signaling messages [31,42]

- Prerequisite: Signaling messages are in plain text.
- Leaked information: IP address, SIP URI, name, etc.

SIP messages are not mandatorily encrypted. Thus an adversary is able to read the content of a SIP message. RFC 3323 [31] presents the threat in which a SIP message leaks identity related information of a user (e.g. IP address, SIP URI, etc). An example is shown in Fig. 7. Firstly, some message header fields such as *to*, *contact* reveal the IP address or SIP URI of a user. Since these header fields are used for session establishing, they cannot easily be withheld or obfuscated. Moreover, a SIP message may sometimes include optional information (e.g. the real name of a caller). Shen and Schulzrinne [42] list these optional headers that may leak identity related information. This attack is straightforward so there are no experiments conducted for confirmation.

It is worth mentioning that this threat is not an issue of Skype since the signaling messages of Skype are mandatorily encrypted.

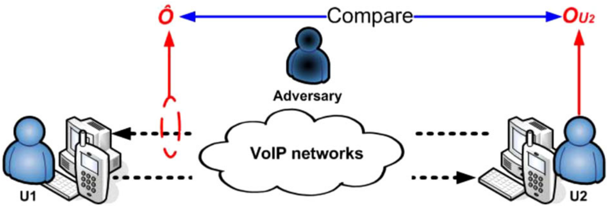


Fig. 8 Human features $\hat{O}$ extracted from flow are linkable to a particular user

4.2 Attacks based on biometric profiles

Users exhibit human-specific features (e.g. speech language, pause duration and frequency, keystroke behavior, etc.) in the conversations in which they participate. These features lead to particular statistical profiles of media flow properties (e.g. the distribution of packet sizes and arrival time). As a result, an adversary is able to profile the human-specific features of the speaker for a given flow and use the features for user identification. We define the problem as follows:

Given a media flow f and a set of n users forming the anonymity set $\{u_1, \cdots, u_n\}$ with their corresponding biometric features $\{O_1, \cdots, O_n\}$, there are three steps for user identification:

1. Extract features $\hat{O}$ from f.
2. Compare $\hat{O}$ to each one in $\{O_1, \cdots, O_n\}$.
3. From $\{O_1, \cdots, O_n\}$, find those O_x which are on the t^{th} shortest distance to $\hat{O}$. Thus they are the suspected users who generated f.

An example for this kind of attacks is illustrated in Fig. 8. The rest of this section summarizes the previous work on this area.

4.2.1 Identify spoken language[54]

- Prerequisite: The UA employs a VBR codec (e.g. Speex).
- Leaked information: The spoken language.

From a given media flow, Wright et al. [54] guess the spoken language from the frequency distribution of packet sizes of that flow. The prerequisite of this attack is that the victim user must use a VBR codec, which adaptively selects the most appropriate bit rate to code the speech signal. For instance, fricative sounds require lower bit rates than vowels. Thus the fricative sounds need fewer bits to be encoded. As a result, the frequency distribution of media packet sizes is highly related to the frequency distribution of different speech signals, which might be a language-dependent feature.

To confirm the hypothesis, Wright et al. encoded speech files from a corpus with 22 languages using Speex. It adaptively selects 9 distinct bit rates for encoding (denoted as $r_1, \cdots, r_9$). Their test shows that the frequency distribution of bit rates is related to spoken language. For instance, the probability of using r_7 for Brazilian Portuguese is around 34 % while that for Hungarian is 30 % or so.

The adversary cannot directly observe which bit rate has been selected. However, it is feasible to guess the selected bit rate from the size of the eavesdropped media packet, since one bit rate uniquely maps to one packet size. Thus, there are nine possible packet sizes in this case, denoted as $s_1, \cdots, s_9$ respectively. They count the 3-g[3] of packet sizes in a given flow. The frequency distribution of 3-g is taken as the feature $\hat{O}$ of the flow. On the other hand, the adversary can learn this feature of all candidate languages. The language with the most similar frequency distributions to $\hat{O}$ is the suspect language spoken by the user.

The authors performed a series of experiments on binary classification, which lets the classifier distinguish between only two languages for a given media flow. The overall detection rate was 75.1 %.

4.2.2 Identify spoken feature by packet size[23,62]

– Prerequisite: The UAs employ a VBR codec (e.g. Speex).
– Leaked information: The speech patterns of a user.

Khan et al. [23] applied a similar method to [54], but for different purposes. Instead of recognizing the spoken language, they tried to recognize the speakers. For a given flow, they modeled $\hat{O}$ by counting its frequency distribution of 3-g of media packet sizes. Then they studied the feature profiles $\{O_1, \cdots, O_n\}$ for all candidate speakers $\{u_1, \cdots, u_n\}$. The distance between $\hat{O}$ and O_i, $1 \le i \le n$ reveals the probability of whether u_i is the speaker.

Their experimental results show that the method can achieve a detection rate of 51.2 % to identify the actual speaker within an anonymity set of 20 potential suspects, with an F-measure value around 72.3 % and a minimum EER of 17 %.

Zhu and Fu [62] applied the same method on Skype with flow traces from 169 candidate users on the Internet. The best detection rate is between 18 and 61 % depending on the setup. The average EER of their tests is around 16 %.

4.2.3 Identify interval and frequency of speech pause [2,61]

– Prerequisite: The UAs employ silence suppression.
– Leaked information: The speech pattern in pause duration and frequency

Backes et al. [2] extract the duration and frequency of speech pauses for user identification. With silence suppression set on, it is easy to observe the durations of speech and pause from media flows, as $[s_1, p_1, s_2, p_2, \cdots, s_k]$, where s_i and p_i indicate the i^{th} speech and pause period respectively. The feature $\hat{O}$ is constructed by the relative frequency of 3-g of durations of adjacent speach-pause-speach, as

$$\hat{O}[(x, y, z)] = \frac{\#\{j | s_j = x, p_j = y, s_{j+1} = z\}}{k - 1}$$

[3] For example, given a flow of packets with sizes of s_1, s_6, s_4 and s_8, the 3-g are (s_1, s_6, s_4) and (s_6, s_4, s_8).

By measuring the distance between $\hat{O}$ and the 3-g distribution of known users, the adversary can find the suspect. In an empirical setup with 20 speakers, their analysis shows an average detection rate of 48 % of all cases.

Similarly, Zhu [61] extracts the intervals of speech and pause of a flow as the feature vector.

$$\begin{pmatrix} s_1 & s_2 & \cdots & s_n \\ p_1 & p_2 & \cdots & p_n \end{pmatrix}$$

where n is the length of a feature vector, s_i and p_j denote the duration of the ith speech and the jth pause respectively. He then models O using a Hidden Markov Model (HMM) based on the feature vectors. The detection rate is up to 30 % with 109 users.

4.2.4 Identify keystroke feature by packet type[58]

- Prerequisite: VoIP Users access an Interactive Voice Response (IVR) system which requires PIN input. Media flow is constructed on RTP.
- Leaked information: The key-click pattern of the flow originator.

To enable a user to interact with an IVR, IETF has standardized a type of RTP payload to represent the keystroke of users.

We [58] proposed a method to extract keystroke patterns from media flows for user recognition. An adversary can find whether the payload of a media packet is for voice or keystroke by reading the RTP header field (Fig. 9). SRTP is not a countermeasure for this problem since it does not encrypt RTP headers. From the arrival time of the keystroke packets, the adversary then can observe key holding durations (between pressing and releasing a key) and key switch durations (between releasing a key and pressing the next key).

In the experiment, we invited 31 test persons to participate. Each student inputs a 4-digit PIN for 50 repetitions. For each repetition, they construct $\hat{O}$ as

$$\hat{O} = [th_1, ts_1, th_2, ts_2, th_3, ts_3, th_4]$$

where th_i indicates the time holding duration, and ts_i denotes the time interval between releasing key i and pressing key $i + 1$. Employing machine learning algorithms, the method achieved average EER of 10–29 % with identification rate up to 65 %.

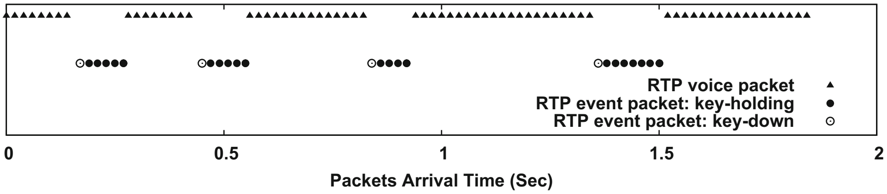


Fig. 9 An example flow showing that type and interarrival time of RTP packets reveal keystroke patterns

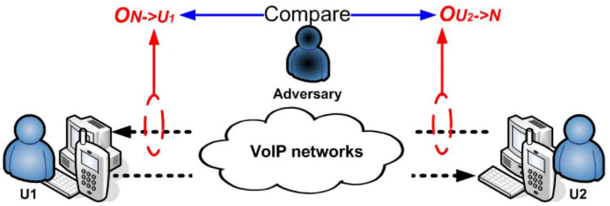


Fig. 10 Flow patterns $O_{u_2 \to N}$ and $O_{N \to u_1}$ are extracted to be compared for flow correlation

4.3 Flows correlation

In the architectures we introduced above, a media flow is not built end-to-end: It goes through a third party, either servers (C/S) or other peer nodes (P2P). In this way, a media flow is separated into several *flow legs*. This actually enhances the anonymity of users: Given one eavesdropped flow leg, an adversary can at most locate the IP address of either the caller or the callee, but not both.

Unfortunately, an adversary can correlate flow legs based on their flow patterns. To explain this, we define the flow leg entering a VoIP network[4] as *ingress flow* and the flow leg leaving a VoIP network as *egress flow*. Theoretically, an egress flow should inherit the flow pattern of its corresponding ingress flow if the pattern has not been changed by the VoIP network. In this way, the observer can say whether an ingress flow is related to an egress flow, and thus find the IP addresses of both users. An example is shown in Fig. 10: The adversary can wiretap the channels from u_1 and u_2 to the VoIP network. When u_1 and u_2 build a conversation, the adversary can eavesdrop the ingress flow $u_2 \to N$ and the egress flow $N \to u_1$ (N indicates the network). Moreover, the adversary extracts patterns from the two flows as $O_{u_2 \to N}$ and $O_{N \to u_1}$. The distance between them suggests the probability of correlation. In this section, we provide an overview of the methods for flow correlation.

4.3.1 Packet count [26]

- Prerequisite: Media flow is built by RTP.
- Leaked information: the IP addresses of caller and callee.

Each RTP packet contains one time stamp field that records the time points when the packet is generated. This value is not encrypted in an SRTP scheme and thus it is available by adversaries. In this way, an adversary can observe a sequence of time stamps for the RTP packets from a media flow. As the value of time stamp is not updated during the transmission, the corresponding ingress and egress flow should have the same sequence of time stamps. Nevertheless, noise might be caused by unexpected packet loss.

Lu and Zhu [26] proposed a method to model this feature using the Fourier transform of packet counter (FPC) vectors. It decomposes the sequence of packet time stamps into components of different frequencies. The reason they choose FPC is that it resists

[4] A VoIP network can use either C/S architecture or P2P architecture.

network noise. They calculate the correlation values of the ingress and egress flows based on FPC. Ideally, the correlation flows should own similar correlation values. Their experiments show that the detection rate can be 75–100 % for 120 different flow traces.

4.3.2 Local-sensitive hash algorithm [11]

- Prerequisite: The UAs apply a VBR codec.
- Leaked information: the IP addresses of caller and callee.

VBR codec will cause a variation of packet sizes in a given flow. Ideally, the ingress and egress flows should have the same sequence of packet sizes. Nevertheless, noise might be caused by packet loss and delay jitter.

Coskun et al. [11] proposed a local-sensitive hash algorithm to correlate media flows for user tracking. The algorithm takes packet sizes and the packet arrival time of a VoIP flow as an input. Given a media flow containing P packets, let T_i indicate the arrival time of the ith packet and let B_i denote the payload size of the ith packet, where $i = 0, 1, \cdots, P-1$. h is the hash digest with L bits and H is a projection array containing L integers. $R_1(), \cdots, R_L()$ are L smooth pseudorandom functions. All elements in H are initialized with 0. For each packet from 1 to $P-1$, the algorithm calculates its size difference from the previous one (as $B_i^\Delta = B_i - B_{i-1}$) and the relative arrival time since the arrival time of the first packet (as $\hat{T}_i = T_i - T_0$). Then, the algorithm projects $\hat{T}_i$ on the smooth pseudorandom functions. The elements in H are updated using the B_i^Δ multiplied by the projecting result. Finally, each bit of h is produced depending on the signs of the corresponding integers in H:

$$h_l = sign(H_l) \begin{cases} 1, & \text{if } H_l \geq 0 \\ 0, & \text{if } H_l < 0 \end{cases} \tag{1}$$

where $l = 1, 2, \cdots, L$. The h_l is the lth bit in H. The detailed algorithm of Coskun hash algorithm is shown below in Algorithm 1.

```
$H \leftarrow [0, 0, 0, \cdots, 0]$ // initialize $h_1, h_2, \cdots, h_L$
for all captured packet $i$ with $i = 0, 1, \cdots, P-1$ do
  if $i = 0$ then
    $flowStart \leftarrow T_i$ // arrival time of the first packet
  else
    $\hat{T}_i \leftarrow T_i - flowStart$ // relative arrival time
    $B_i^\Delta = B_i - B_{i-1}$ // packet size different
    $H \leftarrow H + B_i^\Delta[R_1(\hat{T}_i), \cdots, R_L(\hat{T}_i)]$
  end if
end for
$h = sign(H)$
```

Algorithm 1: The Coskun flow hash algorithm [11]

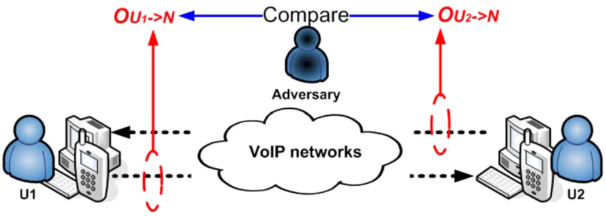


Fig. 11 Flow patterns $O_{u_2 \rightarrow A}$ and $O_{u_1 \rightarrow A}$ are extracted to be compared for flow correlation

Their detection rates are above 90 % with a false alarm rate less than 1 % under the scenario in which the packet loss rate is less than 1 %.

4.3.3 Complementary of flows [52]

- Prerequisite: The UAs apply silence suppression.
- Profile: The IP addresses of caller and callee.

Oliver et al. [52] proposed a method taking advantage of human conversation patterns: When one speaks, the other usually listens. This “alternate in speaking and silence” represents a basic rule of human conversation in a telephone call. In addition, silence suppression enables an attacker to detect silence or speech periods for a given flow. Countrary to previous works, instead of correlating one ingress flow with its egress flow, this method correlates two ingress flows (or 2 egress flows) to see whether they belong to the same conversation. A example is shown in Fig. 11.

Therefore, an attacker can correlate two ingress flows ($u_1 \rightarrow N$ with $u_2 \rightarrow N$) or two egress flows ($N \rightarrow u_1$ with $N \leftarrow u_2$) if the two flow legs belong to a conversation.

Given two flow legs f_i and f_j recorded during time T, a *pairing index value* C for these two flows can be calculated as:

$$C(i, j, T) = \sum_{t=1}^{T} \frac{XOR(f_i[t], f_j[t])}{T}$$

$f_x[t] \in \{0, 1\}$, where 1 indicates that the flow f_x represents speech at time t and 0 indicates silence at time t. Thus, according to the “alternate in speaking and silence” rule, the higher of $C(i, j, T)$, the higher probability that F_i and F_j belong to one conversation. Their experiments demonstrate that the identification rate can reach 97 % for the flows with duration of over 5 minutes.

4.3.4 Watermark attacks [9,41,53]

- Prerequisite: Silence suppression is not used.
- Leaked information: The IP addresses of caller and callee.

Wang et al. [9,53] proposed an active attack to correlate VoIP flows. In this way, an attacker can embed timing watermarks into a VoIP flow by slightly delaying randomly selected packets.

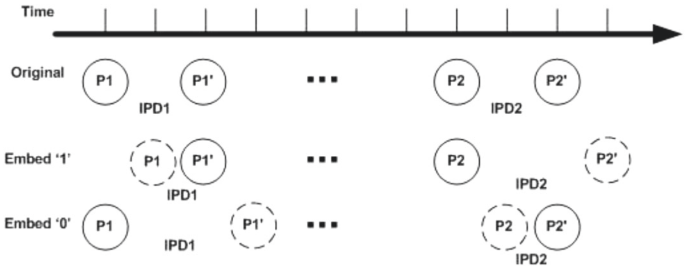


Fig. 12 Embedding a binary bit into a VoIP flow using interpacket delay [9]

An example is illustrated in Fig. 12. Suppose an adversary randomly chooses two packets P_1 and P_2 from a particular voice flow. Then the adversary groups P_1 with P_1' and P_2 with P_2', where P_1' and P_2' are the next packet of P_1 and P_2 in the sequence. Suppose P_1, P_1', P_2 and P_2' arrive with the adversary at t_1, t_1', t_2 and t_2' respectively. Then the adversary can obtain the inter-packet delay (IPD) for these two groups: $IPD_1 = t_1' - t_1$, $IPD_2 = t_2' - t_2$. In this way, the adversary can calculate the normalized difference $IPDD = (IPD_2 - IPD_1)/2$. Since silence suppression is not applied, voice packets should be generated in a constant rate. Thus, the distribution of $IPDD$ should be symmetricly centered around 0. To insert a watermark into the flow, the attacker can slightly delay P_1 and P_2' to embed bit '1' ($IPDD > 0$) or delay P_1' and P_2 to embed bit '0' ($IPDD < 0$). Wang's experiment shows that a delay of 3ms is enough to successfully embed a watermark in Skype flows. By embedding 100 24-bit watermarks in a flow, the true positive rate of the correlation reaches 100 % with only 0.1 % false positive rate.

Sengar et al. [41] did similar research: However, contrary to [53], the delay embedded varies depending on time. The delay for a selected packet p_i is $d = f(t_i - t_1)$, where t_i indicated the arrival time t_i for packet p_i. They have two requirements on the function $f()$ to help in reconstructing the curve during the decoding phase.

- $f(t)$ is differentiable (with no discontinuities)
- It should be periodic.

For instance, the adversary can select particular parameter values (A_1, v_1, ϕ_1) and introduce an inter-packet delay by $d = f(t_i - t_1) = A + A\sin(2\pi v(t_i - t_1) + \phi)$.

They use Discrete Fourier Transform (DFT) to decode the watermarks. DFT is a method to analyze the frequencies contained in sampled signals.

$$X(\omega_k) = \sum_{n=0}^{N-1} x(t_n)e^{-j\omega_k t_n}, k = 0, 1, 2, \cdots, N-1$$

where j=$\sqrt{-1}$, $x(t_n)$ is the input signal amplitude at time t_n, and t_n is the nth sampling instant. $X(\omega_k)$ is the spectrum of the input signal x at the kth frequency ω_k. They have not evaluated the accuracy of the attack in [41].

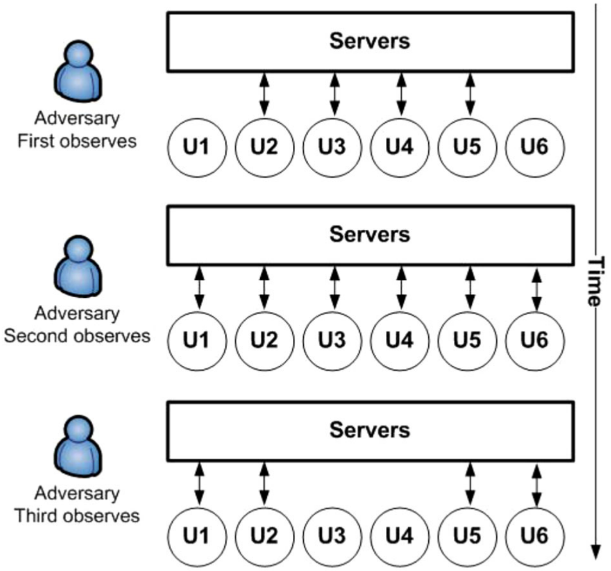


Fig. 13 An example in which six users expose their communications when they start or terminate conversations. The adversay only needs to observe three times

4.3.5 Appear and disappear of flows [59]

- Prerequisite: FBR, without silence suppression
- Leaked information: The IP addresses of caller and callee.

Media flows can be mixed for anonymity. Nevertheless, each conversation has unique starting and ending times, which are not synchronous with other conversations. New flows are generated when a conversation is built and flows are terminated when a conversation is terminated. Thus an adversary can do a timing attack by simply observing the changing states of flows. An example is shown in Fig. 13: Assume that at a certain time point there are four users (u_2, u_3, u_4, u_5) involved in conversations. In this case, the adversary cannot precisely say who is talking with whom. However, at a following time point, u_1 and u_6 start to make a conversation. By observing the new flows, the adversary can say that u_1 is communicating with u_6. Later when u_3 and u_4 terminate their conversation, the adversary can observe the disappearance of the flows and thus knows that u_3 and u_4 have communicated before. This attack is quite straightforward and thus no experiments have been done in [59].

4.4 Topology analysis on P2P VoIP networks [46]

- Prerequisite: The shortest route selection algorithm is used in a P2P network; the attacker partially knows the topology of the network.
- Leaked information: The IP addresses of caller and callee.

To minimize the end-to-end delay, P2P VoIP networks usually employ a shortest route selection algorithm to set up a communication route between the caller and the

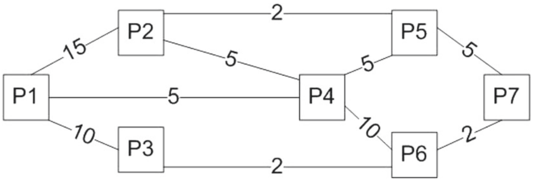


Fig. 14 The topology of a P2P VoIP network. A number in the figure shows the delay between two peer nodes. [46]

callee. The algorithm is introduced in Sect. 2.1.2. In brief, an intermediary peer node only accepts the first arrived instance of a call request and then broadcasts it to all its neighbors. Since only the first arrived instance of a call request will be accepted, thus the established route should be the shortest one between the caller and the callee. For instance, given an example topology illustrated in Fig. 14, the shortest route between P_1 and P_7 is P_1-P_3-P_6-P_7.

Srivatsa et al. [46] found a vulnerability in a P2P VoIP network. The vulnerability is exploitable if (1) a call request contains an unencrypted initiation timestamp t_s;[5] (2) there are malicious peer nodes in the network; and (3) the adversaries partially know the topology of the network. Thus a malicious peer node p can guess its distance to the caller using the timestamp t_s from the request and the arrival time of the request. Additionally, if p received the request firstly from its neighbor q, then the shortest path also must be via q.

The authors then proposed a triangulation attack: Let p be a malicious node that received a request that originated by the caller at time t_s. Say p received it at time t_p, thus p can estimate its distance to the caller: $dist(\widehat{caller}, p) = t_p - t_s$. For all suspected callers s in the network, $|dist(\widehat{caller}, p) - dist(s, p)| < \epsilon$, where ϵ is a detection threshold. The candidate callers to p can be calculated as $score_p(s) = \frac{1}{|dist(\widehat{caller}, p) - dist(s, p)|+1}$. For several colluding malicious nodes $p_1, p_2, \cdots, p_n$, the scores to them are $\frac{\sum_{i=1}^{n} score_{p_i}(s)}{n}$.

They conducted simulations based on the NS-2 topology generator. The malicious nodes were randomly selected and they calculated the scores of each candidate caller of relayed flows using triangulation attacks. The scores are ordered from high to low. Their experiments show that even if only a small fraction (1 %) of the network is malicious, the probability of the real caller appearing in the top-10 on the score list will be around 93 %.

5 Survey of VoIP Anonymity Mechanisms

This section summarizes previous work on countermeasures against VoIP anonymity attacks. These countermeasures are introduced in 4 classes (pseudonym based, padding based, flow correlation resistance and route selection based).

[5] The unencrypted initiation timestamp t_s enables a peer node to discard timeout requests (e.g. those that have been initialized more than 250 ms).

5.1 Pseudonym based defenses

A pseudonym is a substitute of a user's real identity. By using pseudonyms in conversations, users can conceal their real identity-related information (e.g. IP address, SIP URI) while still being accountable. This kind of architectures requires a trusted identity provider to maintain links between pseudonyms and real identities. Users provide their pseudonyms to get services and the identity provider maps pseudonyms to real identities for accountability. The adversaries, without knowing the link, cannot know the holder of a pseudonym. In this way, users can achieve a certain level of anonymity.

5.1.1 User-provided and network-provided pseudonyms [31,42]

RFC3323 [31] proposed two kinds of privacy-enhanced mechanisms: *user-provided privacy* and *network-provided privacy*. The user-provided privacy mechanism is designed for a requirement of low-level anonymity. With this mechanism, optional personal information is removed from SIP messages (For instance, a SIP message can optionally contain a URL pointing to an online photo of the caller. As optional information, this kind of URL should be automatically stripped by a user-provided privacy mechanism). The actual VoIP call is not impacted by the removing of optional information. However, the effect of this mechanism is rather limited: the users' URI and the IP addresses of their equipments still appear in SIP messages: Without them, SIP servers do not know where the responses of these messages should be forwarded. Thus, RFC 3323 suggested the network-provided privacy mechanism, in which a privacy server, working as a trusted third party, converts the user's URI in a SIP message to a randomized pseudonym. In this case, the type of pseudonym is transaction pseudonym which will not link with each other. A privacy server also should keep the mapping state of the user's URI and the pseudonym for the routing purpose. Based on RFC 3323 [31], Shen et al., [42] explained a more detailed architecture for this solution.

The protection is mainly focused on signaling flows, not media flows. Thus the identifiers of media flows, such as IP addresses, are still observable to external adversaries. In addition, this mechanism faces a risk that the privacy server can profile the calling records of all users. Finally, implementation and evaluation has not yet been done.

5.1.2 Enhanced network-provided pseudonyms [29]

RFC 5767 [29] discussed a framework to conceal users' real identity based on Globally Routable User Agent URIs (GRUU) and Traversal Using Relays around NAT (TURN). GRUU [37] works as a temporary globally unique identifier for a specific UA instance. TURN [27] provides a temporary IP address to allow a user to traverse NAT. In this scheme, a user can obtain a temporary SIP URI from a GRUU server and a temporary IP address from a TURN server for a VoIP call. To an external observer, these temporary identifiers are pseudonyms and not directly linkable to the user.

This mechanism is more advanced than [31,42] because it takes both signaling flows and media flows into account. Nevertheless, this solution heavily depends on the infrastructures of GRUU and TURN, which have not been widely deployed.

5.1.3 Network-provided pseudonyms by encryption [20,21]

Karopoulos et al. [20,21] realized the concept of [31,42] in another way: by encryption. In their framework, SIP service providers work as identity providers. When a caller initializes a SIP message, the caller encrypts the identity header fields.[6] The SIP service providers can recover the identities by decryption to forward the message. Either symmetric key cryptography algorithms or asymmetric ones can be used: For the first case, the hashed password shared between a user and a proxy can be used as a secret key for identity encryption. For the second case, identities are encrypted using the provider's public key and decrypted using its private key.

The delay caused by encryption/decryption varies depending on the algorithm. For instance, by using a symmetric algorithm like AES, there is no significant delay introduced. However, an asymmetric algorithm may increase a delay by 1–45 ms for one message operation.

5.2 Padding based defense [55]

As previously mentioned, a VBR codec produces media packets with different packet sizes. Sections 4.2.1 and 4.2.2 discussed the problem that the distribution of packet size discloses information such as a user's spoken language or spoken pattern. A naive countermeasure is to pad packets with some random bits so the real distribution of packet size is not easy to observe. In an extreme case, all packets are padded to the largest possible size. Thus all packets have an equal size.[7] Nevertheless, this naive solution leads the system losing all benefits provided by a VBR codec, as much bandwidth will be wasted. Thus, Wright et al. [55] proposed an optimized way for padding. Let us assume the packet size distribution of a flow is $X = [x_1, x_2, \cdots, x_n]^T$, where x_i is the probability of the ith largest packet size under the source process. Their method is to change X into a target distribution $Y = [y_1, y_2, ..., y_n]^T$, as $Y = AX$, where A is a $n \times n$ matrix. The cost of changing X to Y (the expected number of additional bytes that a user must transmit) is denoted as $f_0(A)$. Thus, selecting a proper A becomes an optimization problem: Service providers want a minimized $f_0(A)$ while still meeting defined constraints.

Wright et al. applied this method as a defense against the attacks proposed in Sect. 4.2.1: By this optimized packets padding, the attack accuracy is reduced from 71 to 50 %. With an attack accuracy of 50 %, the attack is equivalent to a random guess. Thus the attack is not reliable anymore. On the downside, this countermeasure introduces 15.4–42.2 % bandwidth overhead according to their experiments.

5.3 Flow correlation resistance

Flows correlation attacks aim to correlate an ingress flow and its corresponding egress flow. The correlation is usually based on the similar features of flow legs. The counter-

[6] The caller can encrypt caller's identity or both caller's and callee's identities.

[7] It is equivalent to using a FBR codec.

measures are mainly designed in two ways: (1) Modify an egress flow so that an ingress flow and its corresponding egress flow have different features; or (2) Let many flow legs have the same features. Thus it is difficult to use the features for flow correlation.

5.3.1 Silent packets dropping [60]

In [60] we proposed a mitigation solution by modifying egress flows to defend against both watermark attacks [53] and complementary attacks [52]. As introduced in Sects. 4.3.4 and 4.3.3, a watermark attack correlates flows by exploiting normal distribution of packet interarrival time while a complementary attack takes advantage of an on-off flow pattern caused by silence suppression. Thus, there is a dilemma: If users apply silence suppression, they are vulnerable to complementary attacks [52]; If users do not apply silence suppression, they are vulnerable to watermark attacks [53].

If silence suppression is applied, the packet interarrival time for a flow is not constant. It is difficult to insert a watermark in a flow. Nevertheless, it discloses the speech on-off behavior so that a complementary attack is easy to mount. On the other hand, without silence suppression, a complementary attack does not work at all. However, with constant packet inter-arrival time, a watermark attack is rather easy. Taking the dilemma into account, we proposed a solution based on the “defensive dropping” concept [24]. The UAs do not apply silence suppression but only Voice Activity Detection (VAD): A UA can detect silence period, but itself does not drop silence packets. Instead, the UA can instruct the VoIP network to drop some of the silence packets according to a dropping rate dr. This can be achieved by putting one bit (‘0’ for keeping and ‘1’ for dropping) inside the encryption layer. Since the selected packets for dropping are silence packets, it introduces less negative impact on the performance of a VoIP conversation.

Theoretically, this solution can decrease the detection rate of the two attacks. Firstly, it weakens the linkability between ingress flows and egress flows. All ingress flows have constant packet inter-arrival time, but all egress flows have varying time characteristics. Furthermore, not all silence packets will be dropped so that the on-off behavior on flows is still unclear, which means a complementary attack is still difficult to launch. Nevertheless, the amount of dropped packets depends on the dropping rate (dr). In extreme cases, either $dr = 0\,\%$ (no silence packets should be dropped) or $dr = 100\,\%$ (all silence packets should be dropped) violates our design. The dr with a value around 10 % is the optimal according to our tests. A performance analysis of this countermeasure has not yet been conducted.

5.3.2 Achieving k-anonymity

As introduced in Sect. 3.1, the concept of k-anonymity is borrowed from the area of database privacy. The k-anonymity indicates the requirement on the sizes of an anonymity set ($\geq k$). The anonymity set is a set of subjects with the same attributes so that it is difficult for an adversary to distinguish one from the set. To prevent flow correlation, a flow leg is said to satisfy k-anonymity if there are at least other $k - 1$ flow legs that share the same observable patterns. Here the observable patterns include starting time, ending time, the distribution of packet size and interarrival time. There are two approaches focused on achieving k-anonymity for C/S and P2P architectures respectively.

k-anonymity on a C/S architecture [59]: To address the problem in Sect. 4.3.5, we proposed a scheme to process k VoIP conversations as a batch on a C/S architecture [59]. The scheme requires media flow to be generated using an FBR codec with silence suppression off. Thus the media flows will have the constant packet sizes and interarrival time. In addition, the scheme also requires the same starting and ending time of the k flows. To do so, a server will not process call/terminate requests until it receives k call/terminate requests. Here is an example with $k = 2$. Let us say that u_1 wants to call u_2. The u_1 first initializes a call request to the server. However, the server does not forward to u_2 immediately. Instead, the server waits until the second call request is received. At this moment, u_3 makes a call to u_4. Then, the server takes these two requests as a batch and flushes them to u_2 and u_4 respectively. The batch scheme is applied to terminate requests, too. With this scheme applied, the attackers at most know that u_1 , u_2 , u_3 , and u_4 are involved in 2 conversations. However, they cannot distinguish who called whom in detail. For instance, u_1 might call u_2 or u_4. The complexity increases with the k. The major side effect of this method is that the time of establishing and terminating a call is beyond the user's control and dependant on other users (whether there are other users join the batch or not).

k-anonymity on a P2P architecture [45,47]: In a C/S architecture, a media flow can be easily mixed at the server side with other flows. However, this is not the case in a P2P architecture, where users can select their own routes. Due to the topological complexity of P2P networks, it is unlikely that the flows are mixed at a single peer node. To solve this problem, Srivatsa et al. [45,47] proposed an anonymity-aware route selection algorithm. Recall that only the first received call request will be accepted by a peer node in the shortest route algorithm described in Sect. 2.1.2. Differently, in an anonymity-aware route selection algorithm, each peer node knows the number of flows that it currently relays (denoted as $in(p)$). When a peer node p accepts a call request r, the node marks r's updated anonymity degree ($r_k = in(p) + 1$) and broadcasts it to all neighbors. Let us assume a peer node q receives the request twice at two time instances t_1 and t_2 ($t_1 < t_2$) with anonymity degree r_{k_1} and r_{k_2} respectively. Note that the second received request will be accepted instead of the first one if $r_{k_2} > r_{k_1}$. Figs. 15 and 16 illustrate the comparison: Assume there are three routes (P_1, P_2, P_3) and four users ($u_1, \cdots, u_4$) in the network. With shortest path selection, the routers at most relay 1 flow. Thus no k-anonymity with $k > 1$

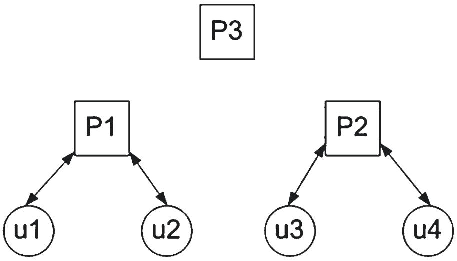


Fig. 15 The route is setup with $k = 1$ [47]

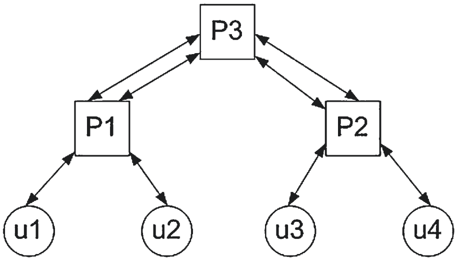


Fig. 16 The route is setup with $k = 2$ [47]

can be achieved and adversary can easily identify the communicating users. With the anonymity-aware route selection algorithm, both the flows will have a round trip to P_3 (of course, if the caused delay is allowed). Thus, from the adversary view, all the routers relay two flows. Then it is hard to judge who u_1 is communicating with. Thus k-anonymity with $k = 2$ can be achieved. Taking k-anonymity into the account, the end-to-end latency will increase since a flow needs to be routed other than the shortest route. By employing an optimization method, the algorithm searches a route with at least k-anonymity. However, one restrictive requirement is that the selected route must satisfy performance conditions (e.g. the end-to-end latency must be less than 250 ms).

5.3.3 Cover traffic based defenses [28]

In this kind of system, all users are required to continuously generate traffic with the same volume and bit rate, no matter whether they are really in a conversation or not. If a user is not in a conversation, its UA generates a dummy flow with randomized information. Otherwise, the dummy flow is replaced by encrypted media flow. With this kind of defenses, an attacker neither knows who is communicating with whom nor whether a user is communicating or not. Thus, it achieves unobservability. Nevertheless, dummy traffic consumes network bandwidth in the network, and in turn, may impact the quality of voice conversation. Thus this kind of defenses is only suitable for systems with a high requirement of anonymity and a small amount of users (e.g. embassies, military communications).

Melchor [28] discussed three types of defenses and evaluated their theoretical bandwidth consumptions. All the defensive architectures are C/S based as follows.

- Trusted third party server: A prerequisite is that each user has exchanged a secret key with the server. If a user is not in a conversation, the user sends a dummy flow to the server. The server should be unable to decrypt the flow and in this way it also sends a dummy flow to the user. Otherwise, the user sends an encrypted media flow to the server instead of the dummy flow. The server successfully decrypts it and finds that it is meaningful. In this way, the server re-encrypts and forwards it to the destination. All users pose the same flow pattern to an external observer and thus the observer cannot say who called whom or who is in a real conversation. However, all users have to trust the server, which gets full knowledge of traffic.
- Broadcast-based server: Instead of sharing a key with the server, each user shares a secret key with her conversation partner. Similar to the architecture above, a user

either sends a dummy flow or an encrypted media flow at any time. The server thus broadcasts received traffic to all users in the network. Only the conversation partner has the right key to decrypt the media flow. Thus the users do not need to trust the server, However, the bandwidth wasting is too high due to broadcasting dummy traffic.

– Private Information Retrieval (PIR)-based server: PIR is a technique that allows a user to retrieve data from a server without revealing to anybody (including the server) what data is being retrieved. This architecture uses PIR on VoIP anonymity. Different to the broadcast-based server, the server will not broadcast flows. Instead, the users privately choose one flow by sending a PIR query every few seconds. For instance, if user u_1 is talking with user u_2, u_1 will choose the flow generated by u_2 with PIR queries. Otherwise, the user will send PIR queries for a random flow. It might be an optimal solution since there is no need to place either trust or broadcasting on the server. Unfortunately, PIR is usually not easily implemented in an efficient manner.

With enforced global dummy flows, the above three types of defenses can achieve unobservability of a caller or a callee in the system. However, none of the above three types of defenses has been implemented. The authors merely conducted theoretical evaluations and predicted that the defenses are suitable to closed circuit networks with hundreds of users.

5.4 Router selection based defenses

These types of defenses are especially designed for P2P VoIP networks. They enable users to optimize their router selection in case of being traced by attackers.

5.4.1 Random walker router selection [46]

It was shown in Sect. 4.4 that the shortest route selection algorithm enables an attacker to trace back the caller. One way to prevent this is to introduce randomness to the route selection. For instance, instead of broadcasting a call request to all neighbors, a peer node p can randomly choose only one neighbor and send the request to it. This route selection algorithm is called *random walk*. Since the route is not the shortest, attackers cannot easily trace back the caller by their knowledge of network topology.

Nevertheless, a pure random walker route selection algorithm does not take performance into account. The end-to-end delay over the selected route may be too large for VoIP conversations. A tradeoff is to use a hybrid path selection algorithm:

– Controlled router setup: It is a combination of the random walk algorithm and the shortest route selection algorithm. Use a parameter y to limit the chance of the random walk. When a node p receives a request, it has y probability to choose random walk while $1 - y$ probability to choose the broadcast to forward requests. Clearly, a small y ensures that the latency of the path is near optimal.
– Multi-agent random walk: The caller sends out w requests and the callee accepts the first received request. Thus, as w increases, the route latency tends to the optimal latency.

Srivatsa et al. [46] shows a simulation that measures the end-to-end latency scales from 64 to 8,200 ms depending on y or w. By selecting proper parameters, for example, letting $y < 0.8$ or $w > 10$, the latency (≤ 250 milliseconds) is acceptable.

5.4.2 Route selection relying on friends [13]

Danezis et al. [13] presented Drac, with the trust model around a friend-of-friend architecture. Given n users within a P2P VoIP network, each user has a set of friends who are trusted and can be used to relay communications. The relationships with friends are public to guarantee the anonymity of the actual calls. The protocol of Drac can be summarized as follows:

1. Heartbeat traffic: A user builds bi-directional heartbeat connections with all friends upon connection to the network. Signaling packets can be embedded in the heartbeat traffic so that attackers cannot differentiate between dummy heartbeat packets and signaling packets.
2. Entry points establishment: Each user needs to have a entry point for indirectly building communications. To do so, each user establishes an encrypted circuit of depth D to her entry points. For instance, the user u_a selects at random one of her friends, saying u_c, as the first hop on the circuit. Then, u_a requests u_c to randomly choose a friend, u_f, and to extend this circuit. It will be iterated for D times. In this case, u_f is the entry point of u_a if $D = 2$.
3. Communication establishment: When a user u_i with entry point E_i wants to call u_j with entry point E_j, she requests to extend the circuit to E_j. Note that E_i and E_j do not have to be friends.

In Drac the attacker is not certain that a given user is communicating. So it achieves unobserveability. In the above example, when u_a communicates with u_c, the attacker cannot distinguish whether u_c is the conversation partner or merely a relay. Their simulation shows that the accuracy rate of the attack is almost equal to a pure random guess when $D = 5$.

5.4.3 Performance evaluation [25]

This work is focused on performance evaluations. In a P2P architecture, there are a number of nodes which can be chosen to relay media flows. Assuming the relays are selected at random, the performance of nodes can be unpredictable. Liberatore et al. [25] did empirical tests on the Internet to see the amount of performance loss. The tests were done over PlanetLab, which is a P2P networking testbed with a number of distributed computers on the Internet. They implemented a testing tool and deployed it over PlanetLab with the following features:

1. Form a path over n selected hosts from PlanetLab network. ($2 \leq n \leq 5$). They collected three sets of PlanetLab hosts with regards to their locations (Asia, Europe and Americas). There are at least 40 active hosts in each set. The host selection is subject to four scenarios:
   - Asia scenario: Randomly select n hosts from the Asia set.

   - Europe scenario: Randomly select n hosts from the Europe set.
   - Americas scenario: Randomly select n hosts from the Americas set.
   - Intercontinental scenario: They firstly choose a set from the Asia, the Europe or the Americas at random, and then choose one host from the selected set. The above process will be be repeated n times.
2. Perform ten pings consisting of UDP packets relayed across the path. Each host along the path records the arrival and sending time of each packet.
3. Perform a test of one-way streaming data across the path, with an packet interarrival time of 20 ms and an effective bitrate of 16 kbps bitrate. The parameters of packet interarrival time and bitrate are representative of the general VoIP codecs.

They recorded performance results for different scenarios and number of hops. The average end-to-end delays are 200, 80 and 90 ms for the Asia, Europe and Americas set respectively. The average packet loss is around 0.1 % for all three scenarios. The performance significantly varies depending on the number of hops for the Intercontinental scenario, in particular the average end-to-end delay can be up to 300 ms. From their work, it can be observed that the performance significantly depends on the physical distance between the selected relays. Their work also proves that the performance can be acceptable if we implemented a P2P anonymity service on the Internet for VoIP.

6 Conclusions

Deploying VoIP services offers lower costs, higher flexibility and more features than traditional telephony infrastructures. However, as we place more and more of our conversations on the Internet, our personal privacy is increasingly at risk. Academic researchers have helped to advance the state of the art in anonymous VoIP communications threats and defenses in the recent years. In this article, we have presented a comprehensive survey of these works. The threats can be categorized as follows: (1) an adversary may directly read identity information from unencrypted signaling messages; (2) an adversary may profile human-specific features (e.g. spoken language, speech speed, etc.) from traffic and then identity the users by the features; (3) an adversary may correlate flows that belong to the same conversation and thus find out the IP addresses of both the caller and callee; and (4) for a P2P VoIP network, an adversary may take advantage of the network topology information to guess the most possible caller for a given calling request. These attacks can be combined to detect users in a more accurate manner.

On the other hand, proposed defensive solutions include: (1) users employing one-time pseudonyms instead of using their real identities for VoIP calls; (2) human-specific features extracted from a flow can be adaptively randomized or eliminated; (3) collaborate schemes to mix VoIP flows to achieve k-anonymity. (4) optimal router selection algorithms for P2P VoIP networks; and (5) cover traffic based defenses to withhold who is communicating with whom and even whether a user is communicating or not.

Nevertheless, none of the above defenses is a comprehensive solution that provides highly useable, efficient and practical anonymity for all VoIP users. Future work in this direction is suggested, keeping several key observations in mind:

- Different usage scenarios: VoIP is a delay-sensitive application. Anonymity services typically result in more delay caused by traffic relay and cryptographic operations. How to trade off to meet requirements of different usage scenarios is a valuable research direction. For instance, the military may have high requirements for protection against traffic flow analysis and may have more bandwidth resources than regular citizens.
- Usability tests: Improving user interfaces for anonymous communication is a critical research direction. Lessons show that it is somewhat cumbersome to install and use anonymous solutions for web applications [10]. Unfriendly user interfaces may limit adoption by the general public and in turn decrease the user anonymity set. Thus a bad interface may lead to a weak anonymity. Future design and implementation on anonymous VoIP should pay attention on the user interfaces.
- Privacy by Design: Like many other current Internet applications, VoIP is designed by focusing on functionalities first. After being deployed, people began to seek new solutions for VoIP privacy and anonymity. Privacy by Design is required explicitly by the new proposed EU-regulation [17]. It requires that privacy and data protection should be taken into account throughout the whole process, from the design to implementation and deployment. More work needs to be done to apply privacy by design into VoIP.
- Long-term traffic analysis resistant: None of the above research work discussed the long-term traffic analysis attacks on anonymous VoIP communications. In a system, the probability of a conversation between any two users does not follow a uniform distribution. This is because each user has a unique set of contacts. Thus if u_1 called u_2, u_1 probably will call u_2 again. Previous work shows that an attacker can de-anonymize users by statistic information [1,12]. This problem also exists in anonymous VoIP communications and should be considered.
- Implementation and evaluation: Besides research by academia, there are some open-source implementations of anonymous VoIP also available in the Internet community. For instance, Torfone [50] uses Tor networks [16] to enable VoIP conversations. Based on the implementation, researchers can evaluate it for its performance issues and further vulnerabilities. Also, an effective quantitative metrics will be helpful to design a tuning scheme to trade off anonymity and performance for the implementations.
- An experiment for comparison. Currently there is no such a unified standard to evaluate the accuracy of attacks. For example, some works use accuracy rate while some works take EER. In addition, the evaluations have been done with different environment, e.g. different number of candidate users in different work. Thus it is rather difficult to compare the performance of attacks and countermeasures directly by using the data obtained in these papers. A testbed platform can be created to run different attacks to allow for comparison between studies.

References

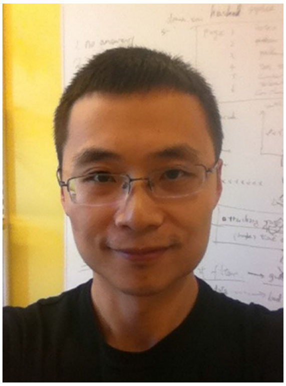
1. Agrawal, D., & Kesdogan, D. (2003). Measuring anonymity: The disclosure attack. *IEEE Security and Privacy*, *1*, 27–34.

2. Backes, M., Doychev, G., Dürmuth, M., & Köpf, B. (2010). Speaker recognition in encrypted voice streams. *ESORICS '10: Proceedings of the 15th European Symposium on Research in Computer Security, LNCS*. New York: Springer.
3. Baugher, M., McGrew, D., Naslund, M., Carrara, E., & Norrman, K. (2004). The Secure Real-Time Transport Protocol (SRTP). RFC 3711.
4. Berners-Lee, T., Fielding, R., & Masinter, L. (2005). Uniform Resource Identifier (URI): Generic Syntax. RFC 3986.
5. Berthold, O., Federrath, H., & Köpsell, S. (2001). Web mixes: A system for anonymous and unobservable internet access. In *International Workshop on Designing Privacy Enhancing Technologies* (pp. 115–129). New York, NY: Springer
6. Buccafurri, F., & Lax, G. (2011). Implementing disposable credit card numbers by mobile phones. *Electronic Commerce Research*, *11*, 271–296.
7. Chang, H. (2013). The security service rating design for it convergence services. *Electronic Commerce Research*, *1*, 1–12.
8. Chaum, D. L. (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, *24*, 84–90.
9. Chen, S., Wang, X., & Jajodia, S. (2006). On the anonymity and traceability of peer-to-peer voip calls. *IEEE Network*, *20*, 32–37.
10. Clark, J., van Oorschot, P., & Adams, C. (2007). Usability of anonymous web browsing: an examination of tor interfaces and deployability. In *Proceedings of the 3rd Symposium on Usable Privacy and Security, SOUPS '07* (pp. 41–51). New York, NY: ACM
11. Coskun, B., & Memon, N. (2010). Tracking encrypted voip calls via robust hashing of network flows. In *ICASSP '10: Proceedings of the IEEE 2010 International Conference on Acoustics, Speech, and Signal Processing* (pp. 1818–1821). IEEE
12. Danezis, G. (2003). Statistical disclosure attacks. In *Proceedings of the IFIP TC11 18th International Conference on Information Security (SEC '03)* (pp. 421–426). Athens: Kluwer
13. Danezis, G., Diaz, C., Troncoso, C., & Laurie, B. (2010). Drac: An architecture for anonymous low-volume communications. In *PETS '10: Proceedings of the 10th international conference on Privacy enhancing technologies* (pp. 202–219). Berlin: Springer
14. Danezis, G., Dingledine, R., & Mathewson, N. (2003). Mixminion: Design of a type III anonymous remailer protocol. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy* (p. 2). Washington, DC: IEEE Computer Society
15. Dierks, T., & Rescorla, E. (2008). The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246
16. Dingledine, R., Mathewson, N., & Syverson, P. (2004). Tor: The second-generation onion router. In *SSYM'04: Proceedings of the 13th Conference on USENIX Security Symposium* (pp. 21–21). Berkeley, CA: USENIX Association
17. European Commission. (2012). Proposal for a Regulation of the European Parliament and of the Council on the Protection of Indivuduals with regard to the Processinf of Personal Data and on the Free Movement of Such Data (General Data Protection Regulation). COM(2012) 11 final, Brussels
18. Google, Facebook, Dropbox, Yahoo, Microsoft, Paltalk, AOL And Apple Deny Participation In NSA PRISM Surveillance Program. (2013). Retrived at 18 June 2013 from http://techcrunch.com/2013/06/06/google-facebook-apple-deny-participation-in-nsa-prism-program/
19. Handley, M., & Jacobson, V. (1998). SDP: Session description protocol. RFC 2327.
20. Karopoulos, G., Kambourakis, G., & Gritzalis, S. (2011). PrivaSIP: Ad-hoc identity privacy in SIP. *Computer Standards & Interfaces*, *33*, 301–314.
21. Karopoulos, G., Kambourakis, G., Gritzalis, S., & Konstantinou, E. (2010). A framework for identity privacy in SIP. *Journal of Network and Computer Applications*, *33*, 16–28.
22. Kent, S., & Seo, K. (2005). Security architecture for the internet protocol. RFC 4301.
23. Khan, L., Baig, M., & Youssef, A. M. (2010). Speaker recognition from encrypted voip communications. *Digital Investigation*, *7*, 65–73.
24. Levine, B. N., Reiter, M. K., Wang, C., & Wright, M. (2004). Timing attacks in low-latency mix systems (extended abstract). In *FC '04: Proceedings of the 8th International Conference on Financial Cryptography* (pp. 251–265). Berlin: Springer
25. Liberatore, M., Gurung, B., Levine, B. N., & Wright, M. (2011). Empirical tests of anonymous voice over IP. *Journal of Network and Computer Applications*, *34*, 341–350.

26. Lu, Y., & Zhu, Y. (2010). Correlation-based traffic analysis on encrypted voip traffic. In *NSWCTC '10: Proceedings of the 2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing* (pp. 45–48). Washington, DC: IEEE Computer Society
27. Mahy, R., Matthews, P., & Rosenberg, J. (2010). Traversal using relays around nat (turn): Relay extensions to session traversal utilities for nat (stun). RFC 5766.
28. Melchor, C. A., Deswarte, Y., & Iguchi-Cartigny, J. (2007). Closed-circuit unobservable voice over IP. In *ACSAC '07: Proceedings of the 23rd Computer Security Applications Conference* (pp. 119–128). IEEE
29. Munakata, M., Schubert, S., & Ohba, T. (2010). User-agent-driven privacy mechanism for sip. RFC 5767.
30. Steiner, P. (1993). On the Internet, nobody knows you're a dog., The New Yorker (p. 61)
31. Peterson, J. (2002). A privacy mechanism for the session initiation protocol (SIP). RFC 3323.
32. Pfitzmann, A., & Hansen, M. (2010). A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, v0.34
33. Pfitzmann, A., Pfitzmann, B., & Waidner, M. (1991). ISDN-MIXes: Untraceable communication with small bandwidth overhead. Kommunikation in Verteilten Systemen, Grundlagen, Anwendungen, Betrieb, GI/ITG-Fachtagung, pp. 451–463. London: Springer
34. Ramsdell, B., & Turner, S. (2010). Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751.
35. Recommendation G.114: One-way Transmission Time. (2013). Retrived at 21 July, 2013 from http://www.itu.int/itudoc/itu-t/aap/sg12aap/history/g.114/index.html
36. Rennhard, M., Rafaeli, S., Mathy, L., Plattner, B., & Hutchison, D. (2004). Towards pseudonymous e-commerce. *Electronic Commerce Research*, *4*, 83–111.
37. Rosenberg, J. (2009). Obtaining and using globally routable user agent uris (gruus) in the session initiation protocol (sip). RFC 5627.
38. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., & Schooler, E. (2002). SIP: Session initiation protocol. RFC 3261.
39. Schulzrinne, H., Casner, S., Frederick, R., & Jacobson, V. (2003). RTP: A transport protocol for real-time applications. RFC 3550.
40. Schulzrinne, H., & Taylor, T. (2006). RTP payload for DTMF digits, telephony tones, and telephony signals. RFC 4733.
41. Sengar, H., Ren, Z., Wang, H., Wijesekera, D., & Jajodia, S. (2010). Tracking skype voip calls over the internet. In *INFOCOM '10: Proceedings of the 30th IEEE Conference on Computer Communications* (pp. 1–5). Washington, DC: IEEE Computer Society
42. Shen, C., & Schulzrinne, H. (2006). A VoIP privacy mechanism and its application in VoIP peering for voice service provider topology and identity hiding. *ICC*, *57*, 3844–3849.
43. Skype. (2013). Retrived at 11 June, 2013 from http://www.Skype.com
44. Skype Security, Skype Homepage. (2013). Retrived 21 July, 2013 from https://support.skype.com/en-us/faq/FA31/Does-Skype-use-encryption
45. Srivatsa, M., Iyengar, A., Liu, L., & Jiang, H. (2011). Privacy in voip networks: Flow analysis attacks and defense. *IEEE Transactions on Parallel and Distributed Systems*, *22*, 621–633.
46. Srivatsa, M., Liu, L., & Iyengar, A. (2008). Preserving caller anonymity in voice-over-ip networks. In *SP '08: Proceedings of the 29th IEEE Symposium on Security and Privacy* (pp. 50–63). Washington, DC: IEEE Computer Society
47. Srivatsa, M., Liu, L., & Iyengar, A. (2009). Privacy in voip networks: A k-anonymity approach. In *INFOCOM '09: Proceedings of the 29th IEEE Conference on Computer Communications*. Washington, DC: IEEE Computer Society
48. Sweeney, L. (2002). k-Anonymity: A model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, *5*, 557–570.
49. Taylor, D., Davis, D., & Jillapalli, R. (2009). Privacy concern and online personalization: The moderating effects of information control and compensation. *Electronic Commerce Research*, *9*, 203–223.
50. TORFone. (2013). Retrivd 18 June, 2013 http://torfone.org/
51. US: No Plans to End Broad Surveillance Program. (2013). Retrived at 18 June, 2013 from http://thedailyreview.com/news/us-no-plans-to-end-broad-surveillance-program-1.1503405
52. Verscheure, O., Vlachos, M., Anagnostopoulos, A., Frossard, P., Bouillet, E., & Yu, P. S. (2006). Finding "who is talking to whom" in voip networks via progressive stream clustering. In *ICDM '06:*

Proceedings of the 6th International Conference on Data Mining (pp. 667–677). Washington, DC: IEEE Computer Society

53. Wang, X., Chen, S., & Jajodia, S. (2005). Tracking anonymous peer-to-peer voip calls on the internet. In *CCS '05: Proceedings of the 12th ACM Conference on Computer and Communications Security* (pp. 81–91). New York, NY: ACM
54. Wright, C. V., Ballard, L., Monrose, F., & Masson, G. M. (2007). Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium, SS'07* (pp. 1–12). Berkeley, CA: USENIX Association
55. Wright, C. V., Coull, S. E., & Monrose, F. (2009). Traffic morphing: An efficient defense against statistical traffic analysis. In *Proceedings of the 16th Annual Network & Distributed System Security Symposium, NDSS '09, ISOC*
56. Xu, F., Michael, K., & Chen, X. (2013). Factors affecting privacy disclosure on social network sites: An integrated model. *Electronic Commerce Research*, *13*, 151–168.
57. Zhang, G. (2010). An analysis for anonymity and unlinkability for a voip conversation. In *Proceings of the 5th IFIP Privacy and Identity Summer School* (pp. 198–212). Berlin: Springer
58. Zhang, G. (2011). Analyzing keystroke patterns of pin code input for recognizing voip users. In: *IFIP Future Challenges in Security and Privacy for Academia and Industry, SEC '11*. New York, NY: Springer IFIP
59. Zhang, G., & Berthold, S. (2010). Hidden voip calling records from networking intermediaries. In *Principles, Systems and Applications of IP Telecommunications, IPTComm '10* (pp. 12–21). New York, NY: ACM
60. Zhang, G., & Fischer-Hübner, S. (2010). Peer-to-peer VoIP communications using anonymisation overlay networks. In *Proceedings of the 11th IFIP TC6, TC11 International Conference on Communications and Multimedia Security, CMS '10* (pp. 130–141). LNCS 6109. New York: Springer
61. Zhu, Y. (2010). On privacy leakage through silence suppression. In *Proceedings of the 13th Information Security Conference, ISC '10* (pp. 276–282). New York: Springer LNCS
62. Zhu, Y., & H, Fu. (2011). Traffic analysis attacks on skype VoIP calls. *Computer Communications*, *34*(10), 1202–1212.
63. Zopf, R. (2002). Real-time transport protocol (RTP) payload for comfort noise (CN). RFC 3389.



Ge Zhang received his undergraduate degree in computer science from Anhui University of Tech. in July, 2003, and his Master's degree in computer science from Blekinge Institute of Tech. in May, 2007. He got Ph.D. degree in the Computer Science Department at Karlstad University in September 2012.Before coming to academia he was working for Wiscom System co., ltd as a programmer. He also had an early "academic life" at Fraunhofer FOKUS and Hasso Plattner Institute. His research interests include VoIP systems, network security, optimization techniques and web security. Now Ge Zhang is working at Evry AB as a security analyst.

Simone Fischer-Hübner has been a Full Professor at the Computer Science Department of Karlstad University, Sweden, since June 2000, where she is the head of the PriSec (Privacy & Security) research group. She received a Diploma Degree in Computer Science with a minor in Law (1988), and Doctoral (1992) and Habilitation (1999) Degrees in Computer Science from Hamburg University. Her research interests include IT-security, privacy and privacy-enhancing technologies. She was a research assistant/assistant professor at Hamburg University (1988–2000) and a Guest Professor at the Copenhagen Business School (1994–1995) and at Stockholm University/Royal Institute of Technology (1998–1999). She is the chairperson of IFIP (International Federation for Information Processing) Working Group 11.6 on Identity Management and the Swedish IFIP TC11 representative. Besides, she is member of the NordSec conference steering committee, steering committee member of STINT (the Swedish Foundation for International Cooperation in Research and Higher Education), coordinator of the Swedish IT Secure Network for PhD students. She is currently participating in the EU FP7 projects A4Cloud and SmartSociety.