ROSS: Low-Cost Self-Securing VoIP Communication Framework

Alfin Syafalni[1], Azman Samsudin[1], Yazid Jaafar[1] and Mohd. Adib Omar[1]
[1] School of Computer Sciences, Universiti Sains Malaysia
11800 USM, Penang, Malaysia.
[e-mail: as10_com083@student.usm.my, azman@cs.usm.my, mymj10_com116@student.usm.my, adib@cs.usm.my]
*Corresponding author: Azman Samsudin

Received August 15, 2012; revised November 8, 2012; accepted December 4, 2012; published December 27, 2012

Abstract

Reliance on the Internet has introduced Voice over Internet Protocol (VoIP) to various security threats. A reliable security protocol and an authentication scheme are thus required to prevent the aforementioned threats. However, an authentication scheme often demands additional cost and effort. Accordingly, a security framework for known participants in VoIP communication is proposed in this paper. The framework is known as Randomness-Optimized Self-Securing (ROSS), which performs authentication automatically throughout the session by optimizing the uniqueness and randomness of the communication itself. Elliptic Curve Diffie-Hellman (ECDH) key exchange and Salsa20 stream cipher are utilized in the framework correspondingly to secure the key agreement and the communication with low computational cost. Human intelligence supports ROSS authentication process to ensure participant authenticity and communication regularity. The results show that with marginal overhead, the proposed framework is able to secure VoIP communication by performing reliable authentication.

Keywords: VoIP communication, self-authentication, security & privacy protocol, low-cost framework, applied cryptography

This research was supported by a research grant from Universiti Sains Malaysia (USM) [304/PKomp/5312091].

http://dx.doi.org/10.3837/tiis.2012.12.019

1. Introduction

Voice over Internet Protocol (VoIP) has revolutionized voice communication because of its features such as scalability and flexibility. VoIP offers lower service, configuration, and deployment costs compared with the long-established telephone system known as public switched telephone network (PSTN). PSTN is built upon a firm physical infrastructure that reinforces its quality of service (QoS) and security [1]. Unlike PSTN, VoIP relies on IP infrastructure which is presently inadequate to fulfill the QoS requirement of its fast-growing application and technology [2].

Recently, research confirmed that VoIP suffers from high latency, jitter, and packet loss during data transmissions [3][4]. In addition, security issues related to Internet connectivity can jeopardize communication and privacy [5]. Thus, security has been considered as a significant aspect in providing good-quality and trustworthy VoIP service. A secure VoIP communication system has three characteristics [1][6][7], namely, confidentiality, integrity, and authenticity. Confidentiality, also known as privacy, indicates that data communications are concealed throughout the session. Integrity is a decisive factor that ensures the data communications are valid and genuine, and authenticity is crucial in establishing trust between endpoints. Cryptography has been applied to satisfy these characteristics. However, cryptography involves substantial computing that requires additional processing cost. Therefore, providing a secure VoIP system which has a QoS-friendly implementation has become a challenging task.

Fig. 1 presents typical cryptographic approaches used in secure communication systems, such as in VoIP. Encryption and decryption are convenient practices of securing transmitted data over media stream protocol. However, a proper key is required in these approaches. Hence, the aforementioned approach presents another challenge in negotiating a key without risking the disclosure of the key to an unauthorized party. In this regard, a notable key exchange protocol such as Diffie-Hellman (DH) [8] is utilized. However, key exchange lacks authentication, which makes it defenseless against a man-in-the-middle (MITM) attack [7][9].

Public-key infrastructure (PKI) has been introduced as a mechanism that provides authentication by relying trust on other eligible parties known as trusted third party (TTP). In PKI, a TTP endorsement is required to perform authentication which makes PKI costly and time-consuming. Thus, relying trust on TTP for a real-time system such as VoIP is inconvenient. Moreover, each approach in **Fig. 1** introduces different computational delays to the system. Aside from its computation delay, authentication usually involves challenge and response interactions that cause more delays in the system [10].

A framework to secure a VoIP session particularly for the known participants, called Randomness-Optimized Self-Securing (ROSS), is proposed in this paper. ROSS implements

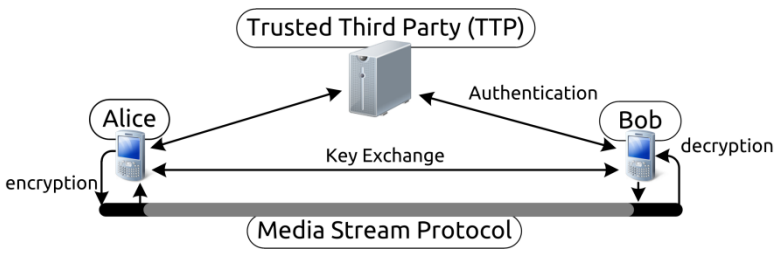

Fig. 1. Typical approaches in securing VoIP system

cryptographic protocols to achieve confidentiality, integrity, and authenticity. ROSS secures the communication automatically during the session without requesting a TTP and tangible responses from the participants. Automatic authentication is the major contribution in this research paper and is achieved by optimizing the uniqueness and randomness of VoIP communication. ROSS utilizes cryptographic protocols such as symmetric-key encryption, hash, and key exchange. Human intelligence plays an essential role in establishing robust authentication by ensuring the authenticity of the participants and communication regularity.

The paper is organized as follows. The next section describes common security threats to privacy in VoIP communication. Section 3 reviews several related works that have been proposed. The proposed methodology is defined in Section 4. Section 5 and 6 show the design and implementation of the proposed framework and its experimental results and analysis. Finally, the conclusion is given in Section 7.

2. Threats on User's Privacy

Common threats such as eavesdropping, impersonation, and session hijacking endanger user privacy within an unsecured communication channel, as shown in **Fig. 2**. The shown actors: Alice and Bob as the actual VoIP participants, Eve as the eavesdropper, Ivan as the impersonator, and Mallory as the MITM.

2.1 Eavesdropping

Eavesdropping, illustrated in **Fig. 2(a)**, is a passive attack in which an unauthorized party listens to the communication between the authorized source and the destination covertly without permission. This attack can be accomplished by monitoring the packet traffic with the use of widely available network tools or packet sniffers [9]. Encryption protocol is a preventive mechanism against eavesdropping. As proven by Zhu-Fu [11], encrypted conversation can be recovered by means of an extensive traffic analysis. Therefore, a highly secure encryption protocol is needed, although the computational cost will increase [3][6].

2.2 Impersonation

Impersonation is an act of intentional bluffing by using someone else's identity, as simplified in **Fig. 2(b)**. In VoIP, impersonation can occur on media stream or signaling protocol. Impersonation on the media stream involves voice forgery and mimicking. Those acts can be easily accomplished when a participant talks to a stranger. However, this attack is difficult and requires cost and skill [12]. Identity spoofing is a form of impersonation used in signaling protocol. This attack replaces a signaling packet such as caller ID with other legitimate ID. The actual solution for identity spoofing is to employ reliable authentication that has the capability to assure the users' identity as well as packet integrity [7]. Additionally, encryption further assists in securing the authentication process [6].

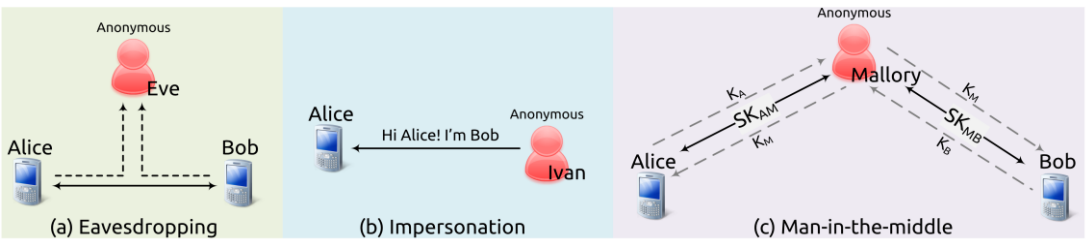

Fig. 2. Privacy threats on VoIP communication

2.3 Session Hijacking

Session hijacking is a harmful threat to a user's privacy. This threat includes eavesdropping and impersonation to support the attack. Session hijacking infiltrates the middle of the communication to gain control over the transmitted data [13]. MITM attack is a renowned act of session hijacking [7][14]. **Fig. 2(c)** shows the MITM scenario that compromises the secure channel by intercepting the key exchange process. Initially, when Alice intends to send her key (K_A) to Bob, Mallory intercepts it and sends her key (K_M) instead. A similar operation is performed while Bob sends his key (K_B). Eventually, Alice and Bob produced shared keys (SK) by pairing their private keys with K_M. Consequently, Mallory will be able to decipher every bit transmitted between Alice and Bob by using the corresponding key (SK_{AM} or SK_{MB}) without Alice and Bob noticing Mallory's presence in the middle of the secure channel.

Engaging a notable key exchange protocol is inadequate against this threat [9]. In order to prevent such an attack, a strong authentication scheme is required [7]. Public-key infrastructure (PKI) has been entrusted to establish trustworthy authentication by using digital certificate. The certificate is issued and verified by a TTP where in centralized trust is known as certificates authority (CA) [15]. CA provides a reliable authentication which makes MITM difficult to forge digital certificates, though the attack is still conceivable to mount. Carelessness in trusting the certificates can lead to a successful MITM attack. In addition, CA suppresses impersonators because only a genuine signature can pass the verification process.

Most CAs are commercial and costly service. As an alternative, a low-cost PKI that applies the concept of decentralized trust has been introduced such as Pretty Good Privacy (PGP) [14][16]. The trust requires numerous TTPs to perform a reliable authentication. The decentralized trust implements a self-signed certificate which is a fragile point for MITM attack. Both concepts of centralized and decentralized trust involve secure interactions between the users and the TTP to succeed the authentication process. Therefore, such approaches are less opportune for a system that has a short session lifetime with diverse endpoints such as VoIP.

3. Existing Security Approaches in VoIP

This section reviews cryptographic protocols that have been proposed to form a secure VoIP system, as summarized in **Table 1**. In a VoIP system, QoS has to be cautiously considered before applying the necessary security features to avoid excessive overhead that can impair communication quality and ease of use. Certain trade-offs are made when resolving the security and privacy issues which are reflected in the complexity of its implementation [4].

Table 1. Existing security approaches in VoIP system

	ZRTP	Enayah-Samsudin	VIPSec	Palmeiri-Fiore	Wang-Liu
Media Protection	SRTP	RC4 (Stream)	Blowfish (Block)	Twofish (Block)	AES (Block)
Key Exchange	DH	DH	-	DH	ECDH
PKC	-	RSA	DH	-	-
Authentication	SAS	Speaker Voice	USO	PKI	PKI
Digital Certificate	-	-	-	X.509	X.509
Eavesdropping	Very Hard	Hard	Very Hard	Very Hard	Very Hard
Impersonation	Knowledge-based	Knowledge-based	Knowledge-based	Very Hard	Very Hard
MITM Attack	Very Hard	Easy	Hard	Medium	Medium

3.1 Privacy in VoIP Communication

In VoIP communication, the media data such as voice and video are transmitted over IP-based network using a protocol, standardized as Real-time Transport Protocol (RTP). Privacy in VoIP communication is achieved by encrypting the RTP packet payload. In cryptography, there are two protocols that have the capability to encrypt information, symmetric-key and public-key cryptography (PKC). Symmetric-key encrypts and decrypts data by using an identical key while PKC uses two different keys, one for encryption and another one for decryption. A notable PKC algorithm such as Rivest-Shamir-Adleman (RSA) [17] has been applied to provide highly secure encryption. However, PKC is inappropriate for intense data streams because it performs heavy computations [18].

The approaches shown in **Table 1** prevent eavesdropping by employing a symmetric-key protocol. Several block ciphers are optimized in stream manner, such as Advanced Encryption Standard (AES), Blowfish, and Twofish. AES, as the current encryption standard, is used in secure RTP (SRTP) standard as the default encryption. Native stream ciphers perform faster operations than block cipher [19], yet many of them are considered broken due to various attacks, especially on the keystream [20][21]. The eSTREAM project has been conducted to propose a standard algorithm for the stream cipher [22]. Nevertheless, block cipher and stream cipher have their own advantages and disadvantages [19]. For instance, block cipher improves ciphertext randomness in cipher-block chaining (CBC) mode whereas stream cipher alleviates the security overhead so that it does not cause a substantial delay during the media transmission.

3.2 Key Exchange

The Diffie-Hellman (DH) key exchange protocol has been widely used for the key agreement on most cryptosystems [7][12][23][24]. However, DH requires large prime numbers for its operation, thereby leading to a bigger key size which consumes more bandwidth during the exchange process [18]. Elliptic curve has been implemented to resolve the key size issue on numerous cryptography protocols including key exchange [25]. Elliptic curve cryptography (ECC) performs faster key generation that produces smaller size of key with comparable strength compared with DH and RSA [26]. Therefore, a combination of elliptic curve and DH, termed as ECDH, enhances the efficiency of key agreement in numerous cryptosystems [18].

3.3 Authentication Scheme

The authenticity of the VoIP session and participants cannot be resolved by relying solely on encryption and key exchange protocols. Various techniques have been proposed, such as Palmeiri-Fiore [7] and Wang-Liu [18] that implement PKI authentication scheme using X.509 digital certificate. On the other hand, Enayah-Samsudin [24] proposed a technique to generate a public key from the participant's voice and then securely exchange a public key using RSA protocol. However, this technique does not provide authentication because the user's voice can be recorded and reused to generate the public key for the next session, which leads to a high potential for an MITM attack. PKI has a better authentication scheme because it binds the user's identity tightly to the respective public key in the digital certificate. This scheme is utilized in most authentications for online transactions such as e-banking and e-shopping.

ZRTP (developed by Phil Zimmermann) [23] is known as a pioneer in authentication over media stream for VoIP communication. ZRTP optimizes human intelligence to authenticate the key agreement through a live conversation by ensuring the shared session key (*SSK*) is not altered by another party [14]. ZRTP users are required to match the four digits of the hashed

SSK called short authentication string (SAS) before securing the communication. The presence of MITM can be detected when SAS between participants does not match. Voice Interactive Personalized Security (VIPSec) [12] uses a similar authentication method which applies PKC variance of DH rather than key exchange. VIPSec matches the user's selected object (USO) that is akin to SAS. In ZRTP and VIPSec, the highest success rate of authentication is achieved when the caller (the participant who initiated the call) has a high sensitivity to the callee (the participant who received the call), especially the voice [14]. Otherwise, it allows someone to bluff during the authentication process.

Video-enabled communication provides the participants with considerable knowledge which increases the sensitivity to the authentication, thereby eliminating impersonation. Unlike in PKI, the authentication scheme over a media stream is performed in real-time through a live conversation without relying on a TTP. This authentication scheme is believed to be an effective and low-cost method of preventing an MITM attack. On the other hand, PKI is an optimal solution to prevent impersonation because only a genuine signature is acceptable in the verification process. PKI features several procedures that present other possible security risks [27]. Finally, PKI, ZRTP, and VIPSec request some physical responses from the users other than voice for authentication, which makes them inapplicable for low-end telecommunication devices [14].

4. ROSS Framework

The proposed framework, ROSS, is described in this section. ROSS employs ECDH key exchange to establish a *shared session key* (*SSK*) which is used to create a secure channel for the communication. ROSS adopts an authentication scheme over media stream to avoid a costly authentication service such as in PKI. As the main contribution, ROSS eliminates the need for the participant's physical response in the authentication process, such as typing SAS, as in ZRTP, or selecting an object, as in VIPSec. ROSS utilizes data randomness and session uniqueness of VoIP communication to achieve automation in the authentication. The session is automatically authenticated with the assistance from the participants to confirm that they are talking to the intended person.

The framework of ROSS consists of three sequential stages, as shown in **Fig. 3**. The first stage immediately begins after the call is established between the users. ROSS checks the call validity by exchanging public information as the initial signal-of-act. Once the call is validated, the next stage will be run throughout the agreed duration to authenticate the actual owner of the exchanged information. Finally, the last stage completes the key agreement and then creates a secure session until the call is terminated.

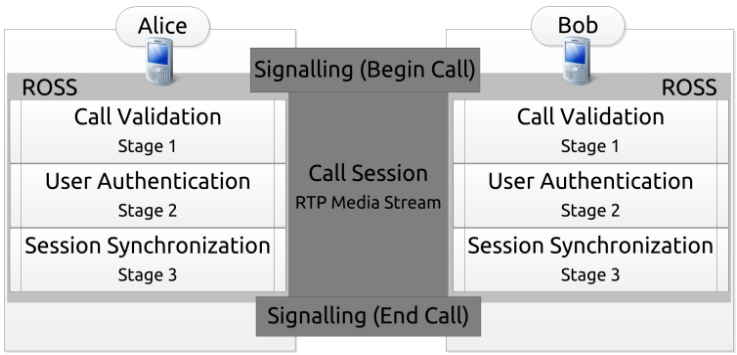

Fig. 3. Overall ROSS framework

4.1 Stage 1: Call Validation

ROSS modifies regular key exchange process. As shown in **Fig. 4**, ROSS exchanges public information known as *initial cue* (*IC*) instead of a plain public key. *IC* holds two values: *key checker* (*C*) and *encrypted public key* (*eU*). Both values are important in authentication and synchronization process in the following stages. Call validation (**Fig. 4**) starts with the caller (example Alice) randomly generating two values: *secret key* (K_A) and an integer (n_A) for *private key* ($R_A = n_A$). Alice's *public key* (U_A) is calculated as $U_A = R_A \times G$, where G is the elliptic curve base point. Subsequently, U_A is encrypted with K_A by using encryption algorithm E, $eU_A = E_{K_A}(U_A)$ and K_A is hashed by using hashing algorithm H to produce *key checker*, $C_A = hash_H(K_A)$. Finally, Alice sends the value of eU_A and C_A as her *initial cue* (IC_A) to the callee (example Bob). Concurrently, Alice is waiting for Bob's *initial cue* (IC_B) and checks the validity of the call based upon IC_B arrival as follows:

$$V_C = \begin{cases} 1 & if \quad IC_B = 0 \quad \rightarrow \text{call is valid} \\ 0 & if \quad IC_B \neq 0 \quad \rightarrow \text{call is invalid} \end{cases} \tag{1}$$

The call is valid if IC_B does not contain null value ($eU_B \wedge C_B \neq 0$). Otherwise if $eU_B \wedge C_B = 0$, a false alarm will be triggered instead. In case *IC* is not received within the allowed time frame, an anomaly event will be detected and the call is terminated.

4.2 Stage 2: Participant Authentication

Participant authentication (**Fig. 5**) aims to authenticate the participant by acquiring a *secret key* (*K*) which is the decryption key of the received *eU*. A valid call ($V_C = 1$) is expected and the same period of authentication must be agreed prior to the execution of this second stage. ROSS operates two core engines: Random Pattern Scrambler (RPS) and Random Pattern Discoverer (RPD) to deliver an authenticated key exchange process, as revealed in Section 5.

As shown in **Fig. 5**, Alice received eU_B, but eU_B is meaningless without K_B. Hence, Bob uses RPS to send K_B to Alice through the *media stream* (MS_B), while Alice uses RPD to retrieve K_B from incoming MS_B. Briefly, to filter K_B as well as to check the integrity of *eU*, a *key checker* (C_B) is incorporated as an input to RPD. As the agreed time for authentication has passed, Alice searches for a potential K_B and automatically authenticates Bob by verifying the output of RPD as follows:

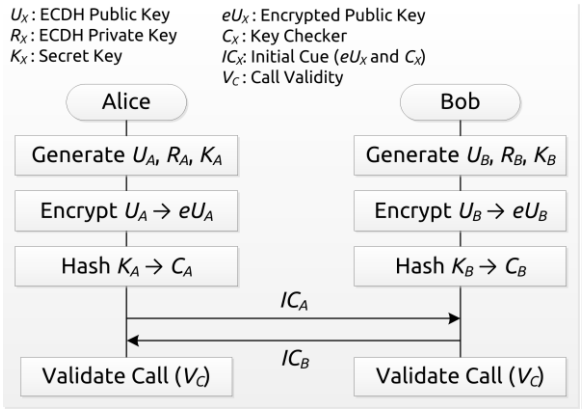

Fig. 4. Stage 1 – Call validation

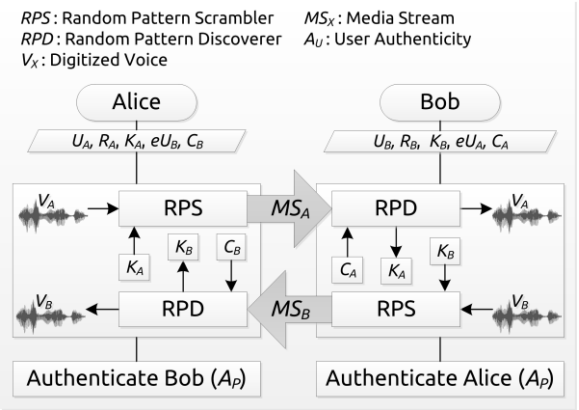

Fig. 5. Stage 2 – Participant authentication

$$A_P = \begin{cases} 1 & if \quad K_B \neq 0 \quad \rightarrow \text{ participant is authentic} \\ 0 & if \quad K_B = 0 \quad \rightarrow \text{ participant is unauthentic} \end{cases} \tag{2}$$

Bob is authentic when Alice successfully discovers K_B. Otherwise the scenario indicates that either K_B is not found or multiple potential K_B are found, which eventually flags the state as a possible attack. The similar process is delivered when K_A is negotiated from Alice to Bob. Confirming the other participant over a live conversation is compulsory before the next stage, or else the protocol is open for attacks, especially impersonation. Moreover, communication irregularity must be avoided during this period in order to achieve a reliable authentication. The criteria of regular communication are listed as follows:

1. **Responsive**: The communication has proper response time interval (absence of unreasonable delay).
2. **Unique**: The current session is distinguishable from the former sessions (no pre-recorded conversations are injected).
3. **Correct**: The participant is giving a correct and rational reply.
4. **Clear**: The media stream does not contain major disturbance.

4.3 Stage 3: Session Synchronization

Once participants are authenticated ($A_P = 1$) with their respective *secret key* (K), Stage 3 (**Fig. 6**) will create *SSK* on each participant to establish a synchronized secure session between them. When K_B is obtained, Alice decrypts eU_B by using algorithm E to retrieve Bob's *public key*, $U_B = E_{K_B}^{-1}(eU_B)$. Alice calculates SSK_A by pairing R_A with U_B, $SSK_A = R_A \times U_B$, whereby Bob calculates, $SSK_B = R_B \times U_A$. Subsequently, Alice engages SSK_A to encrypt her *digitized voice* (V_A) and decrypt the incoming MS_B. At last, Alice synchronizes the session based on the decryption product of MS_B as follows:

$$S_S = \begin{cases} 1 & if \quad V_B \cong MS_B \oplus SSK_A \quad \rightarrow \text{ session is synchronized} \\ 0 & if \quad V_B \neq MS_B \oplus SSK_A \quad \rightarrow \text{ session is not synchronized} \end{cases} \tag{3}$$

The session between Alice and Bob is synchronized if their respective *SSK* decrypts the incoming *MS* properly which only can be achieved if the users have an identical value of *SSK* ($SSK_A = SSK_B$). Once $S_S = 1$ is attained, the communication is secured.

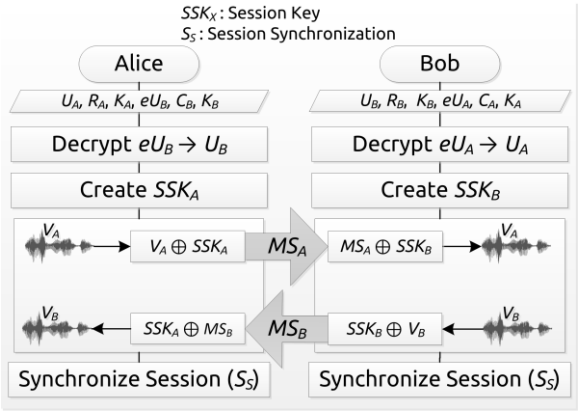

Fig. 6. Stage 3 – Session synchronization

5. Design and Implementation

The ROSS design and implementation are presented in this section. ROSS is designed based on the premises of secure communication as shown in **Table 2**. The design is implemented based on two assumptions of common situations in VoIP communication. First, VoIP sessions are established between two participants who already know each other. Second, the private conversation does not occur at the beginning of the session because it usually starts with salutation and verification whether or not the participant on the other end of the line is the right person. Although the assumptions are not fully applied to all conditions, the users are required to follow the correct procedures to use the framework properly. Furthermore, additional attributes in ROSS are featured as follows:

- **Self-Securing**: Communication is secured without requesting the participant to interact with any response interface.
- **Absolute User Trust**: TTP and digital certificate are not required (voice-based trust).
- **Temporary Session Key**: Session key is unpredictable in the current and the future session because the key is always regenerated on each session.
- **Real-time Authentication**: Real-time authentication is done per session.
- **Inviolable Key Exchange**: Interference in key exchange process is improbable.
- **Independent Platform**: Applicability regardless of the platform, architecture, or protocol used in the VoIP system (end-to-end security).
- **High Feasibility**: High feasibility to work with high-end and low-end telecommunication devices.

ROSS implements two essential protocols which are required in providing VoIP service: signaling protocol and media stream protocol. Signaling protocol handles the call agreement, whereas media stream protocol transports the media such as voice and video over the network. ROSS implements Session Initiation Protocol (SIP) due to its simplistic operation toward IP-based network. ROSS implementation is not based on existing VoIP security standards such as SRTP [28] or IPSec [29]. SRTP merely handles the process of encrypting RTP payload, whereas SRTP employs other standard protocols for the purpose of the key exchange such as Multimedia Internet Keying (MIKEY) [30] or Datagram Transport Layer Security (DTLS) [31]. SRTP uses the key derivation for generating the session key which reduces security overload. However, current and future communications are easily compromised when the attacker can deduce the master key [28]. Unlike SRTP, IPSec handles the entire securing process including both the encryption and authentication. IPSec is a very complex protocol and makes achieving the desirable security level is highly difficult [32].

In most cases, the challenge for key exchange is the MITM attack. MIKEY consists of various approaches which include key exchange and PKI. Nevertheless, some approaches are prone to MITM attack. On the other hand, DTLS-SRTP authenticates the key exchange by using digital certificate. This kind of approach is having difficulties with managing the certificates and can also be vulnerable to MITM attack when the certificates are self-signed

Table 2. Secure communication characteristics

Scope		Premises
Confidentiality	User	Conversation is protected.
	Session	Key is cryptic.
Integrity	User	Media stream cannot be tampered.
	Session	Key is irreversible.
Authenticity	User	Users are authenticated.
	Session	Key is authentic.

and self-verified by the participants [7]. As an alternative, ZRTP was proposed as an extension of SRTP which verbally authenticates DH key exchange [23].

Most of the existing VoIP standards have made modifications to the RTP standard. Thus, the RTP packet contains more crucial information to support additional features. Such modifications also introduce additional operations to the base protocol. Although ROSS does not follow any existing standard, ROSS is constructed based on renowned cryptographic protocols with proven security as described in Section 5.2. Furthermore, ROSS is designed with modest overhead without affecting the original RTP standard in order to provide robust security with low complexity.

5.1 ROSS Core Engines: RPS AND RPD

RPS and RPD are the two fundamental engines in providing automatic authentication in ROSS key agreement. These engines aim to prevent a secret from being altered during its transmission by embedding the secret in the voice stream, as shown in **Fig. 7**. RPS chains a secret with voice stream since it is unique and authenticated persistently by the participant. A legitimate secret is obtained from the stream through filtering which is carried out by RPD. RPD is highly sensitive in discovering the secret such that if the secret does not meet the expected quality or the buffer holds multiple potential secrets, RPD detects these events as malicious and terminates the security process.

Fixed parameters are predefined: number of repetitions (r), authentication period (t), and actual secret size (s_{size}). This impression means a *secret* (S) with actual size of s_{size} is expected to appear r times within t seconds during authentication. The parameters must agree between the two participants or the core will incorrectly progress. As shown in **Fig. 7(a)**, RPS scrambles the input S which is shown in **Fig. 5** as the *secret key* (K) into few equally sized fragments (F) and randomly puts F on the participant's *digitized voice* (V) repeatedly for r times to form a pattern. As a consequence, the participants will hear insignificant disturbances within their conversation. S is scrambled into equally sized fragments based on a randomly generated integer n where

$$\begin{aligned} &S = \{s_i \in \{0,1\}^* \| [s_1, s_2, \cdots, s_{size}]\} \\ &n \in Z^+ \left\| s_{size}/n \right. \le 32 \wedge s_{size} \bmod n = 0 \quad \rightarrow \quad n \text{ usable} \end{aligned} \tag{4}$$

The value of n will be continuously generated until a usable value is obtained. A placement template (T) as the order and position of F on V will be randomly constructed based on **Table 3**. Randomization serves as the decision maker in constructing T. In addition, the construction

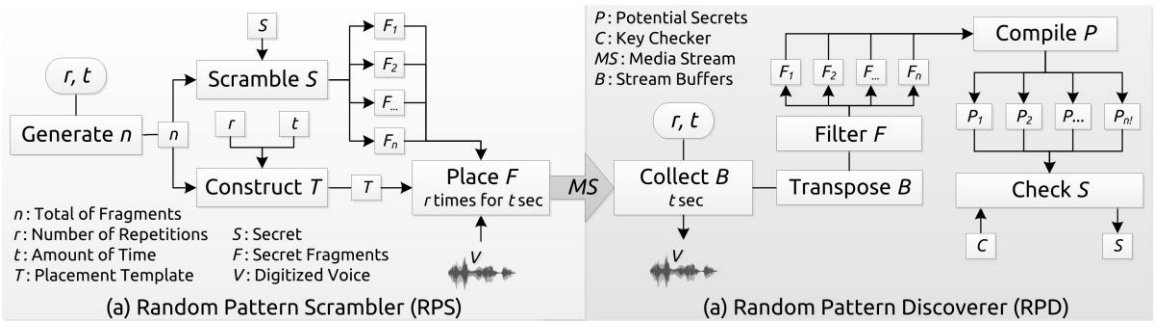

Fig. 7. ROSS core engines

Table 3. RPS and RPD procedures

Subject	Procedures
RPS: Construct *T*	• Vertical orientation (90 °) • No overlapping • No same aligned
RPD: Filter *F*	• $F_{size} > 16$ bits • $r \geq$ total of $F_i \geq r \times 80\%$ (threshold) • $s_{size} \leq K_{size}$ • Randomness at least 50% • Not a subset

of *T* is influenced by specified *r*, *n*, and *t* values. Each *F* will be placed on *V* based on *T* within *t* seconds of users' conversation. Concurrently, RPD collects voice streams (*MS*) into the buffer (*B*) for *t* seconds, as demonstrated in **Fig. 7(b)**. When time *t* is up, collected *B* will be transposed in vertical orientation (90 °).

Subsequently, RPD will retrieve *F* from *B* and filter any *F* that does not satisfy the procedures in **Table 3**. In this step, algorithms such as suffix array [33], quicksort [34] and Boyer-Moore Sunday [35] are employed. Suffix array and quicksort are utilized to find the longest common prefix (LCP) scores [36] to identify any possible *F* pattern that existed in *B*. Furthermore, a fast string matching process is assisted by the Boyer-Moore Sunday algorithm as the final filter of *F* patterns.

After *F* has been filtered, *potential secrets* (*P*) is compiled from all possibilities ($P_{total} = n!$) of *F* combinations. RPD will immediately detect a malicious event if the compiled *P* has different size with the actual size of agreed secret (s_{size}). Finally, an *S* is retrieved among *P* by checking the $hash_H(P)$ values with the received *key checker* (*C*) where the condition $C = hash_H(P)$ must be satisfied. Hence, an eligible *S* is the only one that can recover the valid *public key* (*U*) from the encrypted form, e*U*, as shown in **Fig. 6**.

5.2 ROSS Cryptographic Protocols

ROSS assembles several renowned cryptographic protocols and algorithms as the framework building blocks. The ECDH key exchange serves as the main frame that is modified to provide an authenticated key exchange. ROSS uses pseudorandom number generator (PRNG) to make various decisions, especially on RPS. The PRNG used is based on the SHA1 cryptographic hash algorithm. Moreover, SHA1 is used to generate *key checker* (*C*) which served as the authenticator and the integrity checker of *secret* (*S*) in RPD.

ROSS implements two symmetric-key algorithms: the AES block cipher and Salsa20 stream cipher. The AES-128 is used in CBC mode to encrypt and decrypt *public key* (*U*). Salsa20 [37] (one of the eSTREAM finalists) is employed to encrypt and decrypt *voice stream* (*MS*) through XOR operations with the keystream generated from *SSK*. As benchmarked in [38], Salsa20 is around five times faster than AES so that it can reduce encryption overhead within the communication. *SSK* is obtained from the ECDH key exchange by using prime192 curve, one of the recommended curves that produce a 192-bit shared secret. The SHA-256 hash algorithm is used to convert the shared secret into 256-bit to satisfy the Salsa20 key size requirement.

These cryptographic protocols are requisite to the ROSS framework, but the implementation is not limited to the exact algorithms which allow the users or developers to use other protocol variances following the conditions and personal preferences.

6. Security and Performance Analysis

This section verifies the security of the proposed framework and measures its effectiveness in preventing security threats, especially against communications privacy. The experimental performance analysis focuses on the framework time consumptions, several audio codecs tests, and the impact of the proposed framework toward the QoS.

6.1 Analysis against Communication Privacy Threats

ROSS encrypts data communication to prevent eavesdropping after an authenticated session key is built from the three main stages of the framework. Similar with VIPSec and ZRTP, ROSS lets eavesdropper listen at the beginning of the session until an agreement to secure the channel is reached. ROSS relies on human authentication in preventing impersonation. ROSS removes the possibility that an attacker will reuse a pre-recorded conversation because a fresh and unique conversation is required for every session. Moreover, brute forcing, key guessing, and key stealing cannot break ROSS because the keys used are not permanent and follow the current standard security requirements.

Table 4 shows the summary of ROSS attributes compared with the existing frameworks discussed in Section 3. The Palmeiri-Fiore and Wang-Liu protocols are based on PKI which requires a TTP and digital certificates. PKI can provide automaticity if the key used to sign and verify the certificate is stored inside the device rather than manually kept by the user. However, storing the key inside the device can lead to key stealing which can be done by malicious software such as the Trojan horse. This attack can be mounted on ZRTP as well because the concept of SRTP key derivation is also applied to ZRTP. In PKI, TTP is essential for authentication because a self-signed and self-verified certificate can allow MITM to manipulate certification. Unlike ZRTP and VIPSec, ROSS encrypts the public key during key exchange. Hence, the only way to learn the public key is by discovering its decryption key.

ROSS is highly sensitive against the MITM attack. In order to sneak in the middle of a secure communication, the MITM must intercept the key exchange and falsify the user's public key by sending an encrypted public key along with its decryption key. Falsifying the encrypted public key is relatively easy. However, sending its decryption key must be done through RPS and RPD, which is randomized and time-bound. In addition, randomly placing the key with vertical orientation makes the learning of the key is very difficult before receiving the whole voice stream that contains the key fragments throughout the authentication period. Thus, the MITM forces his decryption key in without being able to remove the original participant's key from the voice stream. Eventually, a malicious event notification is triggered because of multiple keys appearance is detected on the buffer.

The caller's knowledge about the callee is crucial in the authentication scheme over media stream because of the MITM attack can succeed the attack through impersonation. As the last resort, the MITM intercepts the whole voice stream. However, doing so will significantly

Table 4. A comparison with the existing security approaches

	TTP Necessity	MITM Sensitivity	Automatic	Key Stealing
ZRTP	No	High	No	Yes
Enayah-Samsudin	No	None	Yes	No
VIPSec	No	High	No	No
Palmeiri-Fiore	Yes	Moderate	No | Yes	No | Yes
Wang-Liu	Yes	Moderate	No | Yes	No | Yes
ROSS	No	High	Yes	No

interrupt the communication that makes VoIP participants easily notice the irregularity on their communication which poses a possible threat to their privacy.

6.2 Experimental Results

ROSS framework prototype has been tested on a SIP-based softphone which is developed in Java environment. The prototype is executed on two Windows 7 computers that communicate using the standard RTP over simple local area network (LAN). A computer with 2.6 GHz Intel® Core™2 Duo processor and 2GB of RAM is used to generate the experimental results.

6.2.1 Authentication Performance Test

The G.723 audio codec with 8 kHz clock rate was utilized for the authentication performance tests. Based on the trials, RTP with G.723 8 kHz has transmission speed approximately 16-17 packets per second with 48 byte payload. Fluctuations in several results were shown due to the randomizing processes in ROSS core engines. Thus, the results were analyzed by plotting the best scores that represent the major appearances of 30 test cycles. As previously shown in **Fig. 7**, RPS and RPD have three fixed parameters that must agree between the users: number of repetition (r), authentication period (t), and secret size (s_{size}). The first two parameters (r and t) are independent variables to specifically observe the time performance in the RPS template (T) generation and RPD secret fragments (F) filtration as shown in **Fig. 8-11**. The results were gathered by varying one of the independent variables. The r variation was increased by 1 to 20 times, whereas t was increased by 5 seconds from ±5 s (80 packets) to ±20 s (336 packets). Computation in RPD prefers a shorter key since it performs pattern searching. The computation can significantly be reduced by using 128-bit (16 byte) of an AES key instead of a 600-bit public key of ECDH prime192. Therefore, a secret key (S) with s_{size} = 16 byte was chosen as the RPS input and the value was maintained static for the experiment. S is randomly scrambled into symmetrical pieces of fragments (F) based on the usable values of integer n that is generated according to **Eq. 4**. In this case, the possible values of n are 1, 2, and 4 that will correspondingly produce F with sizes of 16 byte, 8 byte, and 4 byte.

Fig. 8 and **9** show the time consumed in the T generation (in ms). In **Fig. 8**, t is varied and r is fixed at 10 times, whereas r is varied and t fixed at 10 s in **Fig. 9**. Both **Fig. 8** and **9** have an increasing trend either by changing t or r. The variations in the result are due to the procedures in **Table 3** in which the unsatisfied conditions cause more iteration in the algorithm. **Fig. 10** and **11** show the time consumed in filtering F (in ms). **Fig. 10** shows that increasing t with

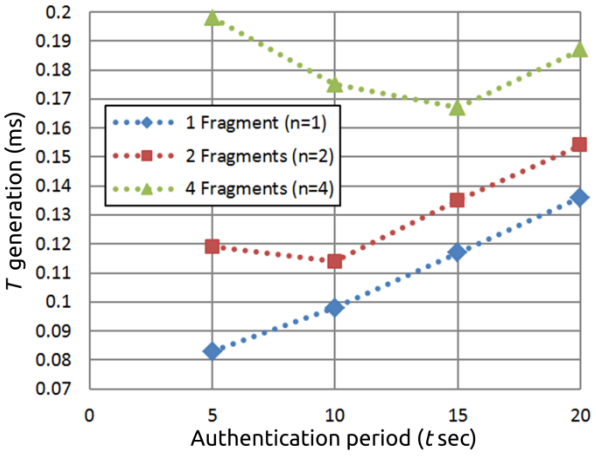

Fig. 8. Time of template generation with fixed r

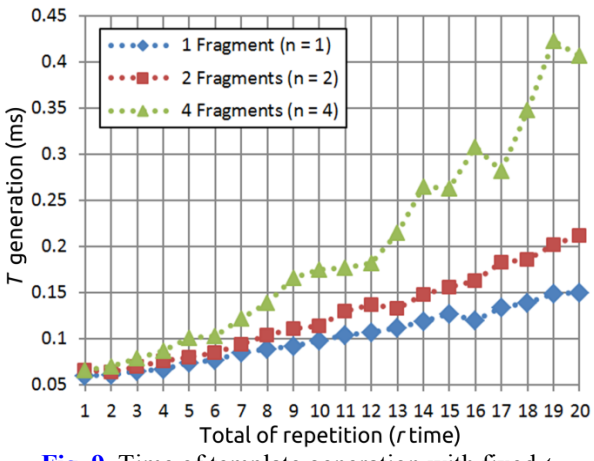

Fig. 9. Time of template generation with fixed t

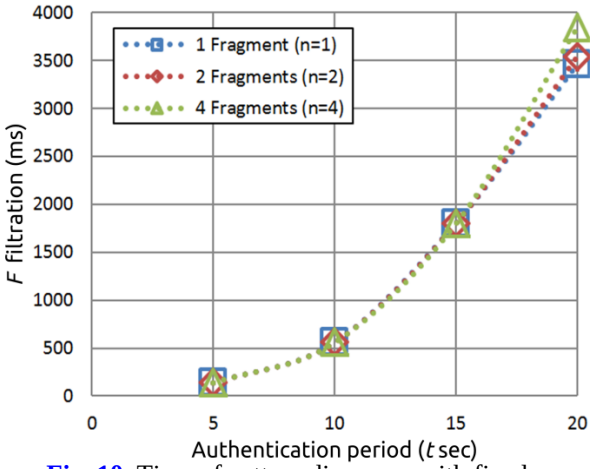

Fig. 10. Time of pattern discovery with fixed *r*

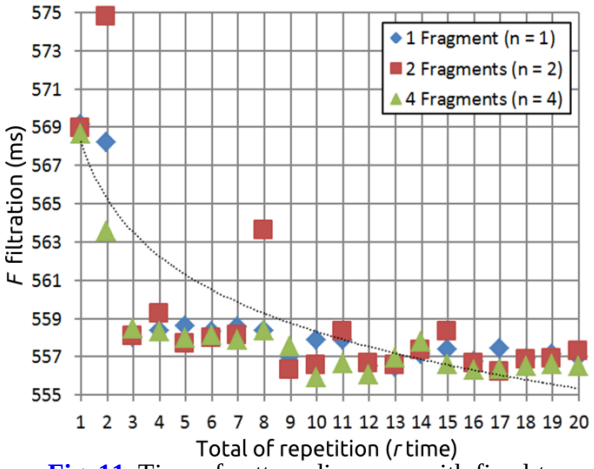

Fig. 11. Time of pattern discovery with fixed *t*

fixed r = 10 times results in significant exponential growth to the time consumed. This result was mainly caused by the fact that RPD filtering processes have numerous nested loops. As a result, RPD runs at an asymptotic growth. In **Fig. 11** where r was varied with fixed t = 10 s, an insignificant decreasing trend is observed since it becomes easier to parse F from the buffer (B) that contains more repeated patterns (bigger n and r).

ROSS's approximate time consumed to authenticate the session is the total duration of the authentication period (t) plus the time taken in filtering F due to the results of r and n variances are negligible ($\leq$ 0.575 seconds). Hence, the estimated total time consumed in performing authentication for t = 5 s, t = 10 s, t = 15 s, and t = 20 s are ±5.137 s, ±10.556 s, ±16.797 s, and ± 23.612 s, respectively. Overall results show that the authentication period (t) within the range 5 s to 15 s is an optimum choice to achieve reasonable and reliable authentication with any proper number of F repetitions (r) and total fragments of F (n). Practically, ROSS spends the waiting time in the authentication for the participants to recognize each other first while it concurrently runs the algorithm in the background. Compared with other authentication schemes, most schemes have unfixed time (usually longer) in building trust due to the involvement of TTP and the requirement of user's physical interaction with the device.

6.2.2 Performance Test on Different Codecs

Encryption on the media stream certainly increases delay in the VoIP communication which probably affects the QoS. This subsection evaluates the performance of three audio codecs: G.723, GSM, and G.711 (u-law), which are available from the Java Media Framework (JMF) library. The codecs used have 8 kHz sampling rate, which reproduce a monophonic sound. Each codec transforms voice data into the respective payload size through an encoding process

Table 5. Comparison of payload size on the tested audio codecs

Audio Codec (8 kHz, Mono)	Payload Size
G.711 (u-law) 8-bit	480
GSM	99
G.723	48

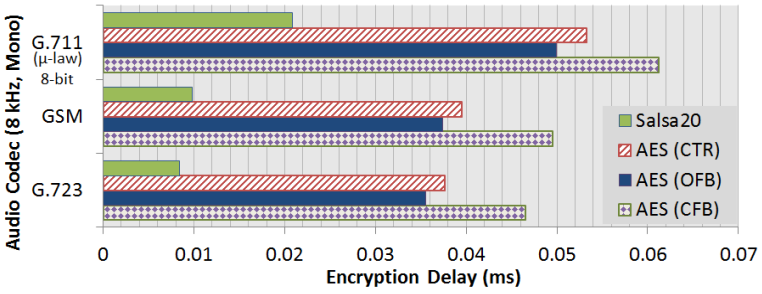

Fig. 12. Encryption delay of tested stream ciphers in different codecs

for RTP packet transmission. Based on **Table 5**, G.723 has the smallest payload size compared to the others. The results, which presented in **Fig. 12**, were obtained by calculating the mean of encryption processing times of 300 RTP payload packets with 3 run cycles using the same audio data on each codec. **Fig. 12** shows the encryption delay in different codecs. 256-bit key length was used for all the tested encryption algorithms.

The results have proven that Salsa20 can ease the encryption overload greater than AES (block cipher), even when AES is running on stream modes such as counter (CTR) mode, output feedback (OFB) mode, or cipher feedback (CFB) mode. From the result, it is also safe to conclude that the overhead incurred by having an audio codec with stream cipher encryption is relatively low. However, considering the unreliability of the IP network, minimizing delays as much as possible locally in the device is a good practice to maintain the acceptability in the communication. According to International Telecommunication Union (ITU) recommendation for G.114, the communication quality is considered as unacceptable if the communication delay exceeds 400 ms per packet transmission [3].

6.2.3 Impact on QoS

In real-time communication systems, time is considered one of the valuable assets in QoS. Most authentication schemes require substantial time and human assistance, especially the authentication that involves challenge and response. ROSS removes this obstacle by offering a new self-authentication scheme that relies on randomness and uniqueness of VoIP communication. ROSS saves time and reduces user's physical involvement in performing authentication. In addition, ROSS maintains the low-cost service in providing VoIP since ROSS does not require outsiders' service such as authentication server or TTP to be involved.

Packet loss becomes another important aspect in QoS. Mostly in the IP network, this issue is intense due to unreliable networks and thus emerges as another challenge. For ROSS, packet dropping is a harmful threat that can damage the pattern and preclude the secret key from being discovered. Hence, placing the fragments in an aligned position is discouraged (**Table 3**) because more than one fragment can be damaged at the same time by a single packet drop. **Fig. 13** shows the ROSS tolerance toward packet dropping. The experiment was simulated by randomly dropping the packets with a consistent increase of 1% drop rate on each of the 20 runs. The random dropping of packets causes the results to fluctuate. The trend shows that having bigger fragments (smaller n) increases the chance of the pattern to be damaged. On the

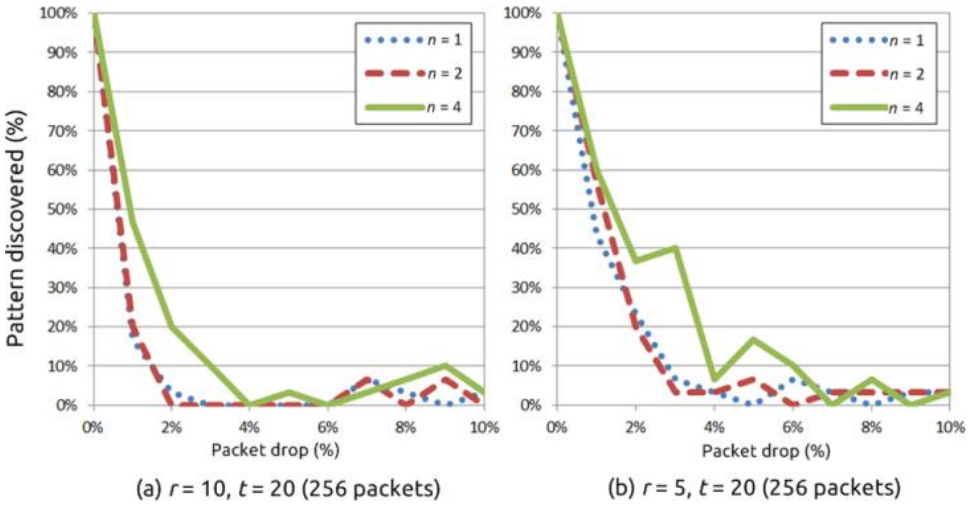

Fig. 13. Percentage of pattern discovery against packet drop

other hand, increasing the total fragments (bigger *n*) helps the pattern survive the dropping of a few packets because of a higher possibility of fragment dispersion. In addition, as shown in **Fig. 13(a)** and **(b)**, a longer authentication period (*t*) is more tolerant toward packet loss although the authentication process takes a longer period. Another current solution to resolve this issue is by employing RTP control protocol (RTCP) that promises packet delivery.

7. Conclusion

The security framework proposed, called ROSS, provides QoS friendly design and implementation. It is achieved by using the known light cryptographic protocols such as ECDH key exchange and Salsa20 stream cipher. ROSS removes unnecessary processes during authentication and eliminates the need of TTP. The user is compulsory to confirm the other participant identity and the communication regularity. ROSS is equipped with two core engines that utilize randomness and uniqueness of communication pattern in VoIP, which are RPS and RPD. The authentication is done automatically and the confidentiality of the public key is guaranteed through the aforementioned core engines. The ROSS core is time-oriented thus a longer authentication period certainly requires more computations. A period between 5 seconds to 15 seconds of authentication is recommended which is also considered acceptable for an authentication process. Moreover, intense packet loss during the authentication can thwart the authentication process. This issue is resolvable by increasing the fragments or implementing RTCP during the authentication. According to experimental analysis, with marginal overhead, ROSS provides authentication with high MITM sensitivity and prevents threats to user communication privacy.

The ROSS concept can be applied to any real-time system that has intense, random, and unique communication pattern which can be persistently verified by human intelligence. ROSS is highly feasible for a wide range of telecommunication devices including low-end devices because of the voice-based trust. Real-time communication systems such as PSTN, GSM cellular, and radio broadcast can also benefited from ROSS.

References

[1] D. C. Sicker and T. Lookabaugh, "VoIP security: Not an afterthought," *Queue*, vol. 2, p. 56, 2004. Article (CrossRef Link)

[2] J. Chandrashekar, Z. L. Zhang, Z. Duan, and Y. T. Hou, "Towards a service oriented internet," *IEICE Trans. Commun.*, vol. 89, pp. 2292-2299, 2006. Article (CrossRef Link)

[3] S. Karapantazis and F.-N. Pavlidou, "VoIP: A comprehensive survey on a promising technology," *Computer Networks*, vol. 53, pp. 2050-2090, 2009. Article (CrossRef Link)

[4] T. J. Walsh and D. R. Kuhn, "Challenges in securing voice over IP," *IEEE Security & Privacy*, vol. 3, pp. 44-49, 2005. Article (CrossRef Link)

[5] D. Schneider, "The state of network security," *Network Security*, vol. 2012, pp. 14-20, 2012. Article (CrossRef Link)

[6] R. Dantu, S. Fahmy, H. Schulzrinne, and J. Cangussu, "Issues and challenges in securing VoIP," *Computers & Security*, vol. 28, pp. 743-753, 2009. Article (CrossRef Link)

[7] F. Palmieri and U. Fiore, "Providing true end-to-end security in converged voice over IP infrastructures," *Computers & Security*, vol. 28, pp. 433-449, 2009. Article (CrossRef Link)

[8] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644-654, 1976. Article (CrossRef Link)

[9] P. Gupta and V. Shmatikov, "Security Analysis of Voice-over-IP Protocols," *20th IEEE in Computer Security Foundations Symposium*, 2007. CSF '07, 2007, pp. 49-63. Article (CrossRef

Link)
[10] W. Liang and W. Wang, "On performance analysis of challenge/response based authentication in wireless networks," *Computer Networks*, vol. 48, pp. 267-288, 2005. Article (CrossRef Link)
[11] Y. Zhu and H. Fu, "Traffic analysis attacks on Skype VoIP calls," *Computer Communications*, vol. 34, pp. 1202-1212, 2010. Article (CrossRef Link)
[12] D. Zisiadis, S. Kopsidas, and L. Tassiulas, "VIPSec defined," *Computer Networks*, vol. 52, pp. 2518-2528, 2008. Article (CrossRef Link)
[13] J. C. Pelaez, E. B. Fernandez, and M. M. Larrondo-Petrie, "Misuse patterns in VoIP," *Security and Communication Networks*, vol. 2, pp. 635-653, 2009. Article (CrossRef Link)
[14] M. Petraschek, T. Hoeher, O. Jung, H. Hlavacs, and W. Gansterer, "Security and usability aspects of Man-in-the-Middle attacks on ZRTP," *Journal of Universal Computer Science*, vol. 14, pp. 673-692, 2008. Article (CrossRef Link)
[15] R. Hunt, "Technological infrastructure for PKI and digital certification," *Computer Communications*, vol. 24, pp. 1460-1471, 2001. Article (CrossRef Link)
[16] M. Blaze, J. Feigenbaum, and J. Lacy, "Decentralized trust management," *Proceedings of IEEE Symposium on Security and Privacy*, 1996, pp. 164-173. Article (CrossRef Link)
[17] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, pp. 120-126, 1978. Article (CrossRef Link)
[18] C.-H. Wang and Y.-S. Liu, "A dependable privacy protection for end-to-end VoIP via Elliptic-Curve Diffie-Hellman and dynamic key changes," *Journal of Network and Computer Applications*, vol. 34, pp. 1545-1556, 2010. Article (CrossRef Link)
[19] A. D. Elbayoumy and S. J. Shepherd, "Stream or block cipher for securing VoIP?," *International Journal of Network Security*, vol. 5, pp. 128-133, 2007.
[20] I. Mantin, "Predicting and Distinguishing Attacks on RC4 Keystream Generator," *Advances in Cryptology* – EUROCRYPT 2005, vol. 3494, R. Cramer, Ed., ed: Springer Berlin / Heidelberg, 2005, pp. 551-551. Article (CrossRef Link)
[21] R. Anderson, "Faster attack on certain stream ciphers," *Electronics Letters*, vol. 29, pp. 1322-1323, 1993. Article (CrossRef Link)
[22] eSTREAM. (27 February). eSTREAM, the ECRYPT Stream Cipher Project. Available: http://www.ecrypt.eu.org/stream/index.html
[23] P. Zimmermann, A. Johnston, and J. Callas, "ZRTP: Media Path Key Agreement for Unicast Secure RTP," Internet Engineering Task Force (IETF) 2070-1721, 2011.
[24] M. R. Enayah and A. Samsudin, "Securing Telecommunication Based On Speaker Voice As The Public Key," *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 7, pp. 201-209, 2007.
[25] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, pp. 203-209, 1987. Article (CrossRef Link)
[26] R. Schroeppel, H. Orman, S. O'Malley, and O. Spatscheck, "Fast key exchange with elliptic curve systems," *Advances in Cryptology*—CRYPT0'95, pp. 43-56, 1995. Article (CrossRef Link)
[27] C. Ellison and B. Schneier, "Ten risks of PKI: What you're not being told about public key infrastructure," *Comput Secur J,* vol. 16, pp. 1-7, 2000. Article (CrossRef Link)
[28] M. Baugher, D. McGrew, M. Naslund, E. Carrara, and K. Norrman, "The secure real-time transport protocol (SRTP)," Internet Engineering Task Force (IETF) 2004.
[29] S. Kent and K. Seo, "Security Architecture for the Internet Protocol," Internet Engineering Task Force (IETF) 2005.
[30] J. Arkko, E. Carrara, F. Lindholm, K. Norrman, and M. Naslund, "MIKEY: Multimedia Internet KEYing," Internet Engineering Task Force (IETF) 2004.
[31] D. McGrew and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)," Internet Engineering Task Force (IETF) 2010.
[32] N. Ferguson and B. Schneier, "A cryptographic evaluation of IPsec," *Counterpane Internet Security, Inc*, vol. 3031, 2000.

[33] U. Manber and G. Myers, "Suffix Arrays: A New Method for On-Line String Searches," *SIAM Journal on Computing,* vol. 22, pp. 935-948, 1993. Article (CrossRef Link)
[34] C. A. R. Hoare, "Quicksort," The Computer Journal, vol. 5, pp. 10-16, January 1, 1962 1962. Article (CrossRef Link)
[35] D. M. Sunday, "A very fast substring search algorithm," *Communications of the ACM*, vol. 33, pp. 132-142, 1990. Article (CrossRef Link)
[36] T. Kasai, G. Lee, H. Arimura, S. Arikawa, and K. Park, "Linear-Time Longest-Common-Prefix Computation in Suffix Arrays and Its Applications," in *Combinatorial Pattern Matching*. vol. 2089, A. Amir and G. M. Landau, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 181-192. Article (CrossRef Link)
[37] D. Bernstein, "The Salsa20 family of stream ciphers," New Stream Cipher Designs, vol. 4986, M. Robshaw and O. Billet, Eds., ed: Springer Berlin / Heidelberg, 2008, pp. 84-97, 2008. Article (CrossRef Link)
[38] Crypto++. (10 October). *Crypto++ Library, Free C++ class library of Cryptographic Schemes*. Available: http://www.cryptopp.com/benchmarks.html

Alfin Syafalni received the B.Sc. degree in Computer Science from Universiti Sains Malaysia in 2010 and currently pursuing a M.Sc. degree at the same university. In 2009, he underwent the internship program with the Malaysian Institute of Microelectronics Systems (MIMOS) working on IMS implementation and its application. His research interests are on Networking, Multimedia, and Information Security.

Azman Samsudin is an Associate Professor at the School of Computer Sciences, Universiti Sains Malaysia. He earned his B.Sc. in Computer Science from University of Rochester, New York, USA, in 1989. Later, he received his M.Sc. in Computer Sciences and his Ph.D. in Computer Science, in 1993 and 1998, respectively, both from the University of Denver, Colorado, USA. He has been with Universiti Sains Malaysia since 1998. He has published articles in various professional journals and conference proceedings and has held a series of grants in the fields of Cryptography, Switching Networks, and Parallel Computing.

Yazid Jaafar is a Research Officer at the School of Computer Sciences, Universiti Sains Malaysia (USM), Malaysia. He earned his B.Sc. in Computer Science from USM in 2010 and currently pursuing his M.Sc. degree at the same university. His research interest includes Applied Cryptography, Network Security and Internet Protocol.

Mohd. Adib Omar completed his B.Sc. (Artificial Intelligence) and M.Sc. (Computer Networks) in Computer Science from American University, Washington DC, USA in 1996 and 1997 respectively. He received his Ph.D. in Collaborative Computing from Universiti Sains Malaysia (USM), in 2009. He is currently a senior lecturer at School of Computer Sciences, USM. His research interests include Wireless Networks, Collaborative and Service Computing, Distributed and Parallel Computing, and Information Security.