An Authentication Scheme for SIP using Needham Schroeder Authentication Protocol

[1] Natalia Chaudhry, [2] Rabia Sirhindi

[1] Department of Computer Sciences, Kinnaird College for Women
Lahore, 54000, Pakistan

[2] Department of Computer Sciences, Kinnaird College for Women
Lahore, 54000, Pakistan

Abstract - Session Initiation Protocol (SIP) based Voice over Internet Protocol (VoIP) services has attained much attention over the last decade. SIP is an extensively used Internet protocol for real time communication and establishment of media sessions. However, it is vulnerable to several security attacks due to its open architecture and text-based nature of SIP messages. Also, the inherent vulnerabilities of the underlying transport protocols such as TCP, SCTP and UDP renders SIP exposed to some serious security flaws. One of these is the protocol's weak authentication scheme that leads to a number of attacks including registration hijacking, impersonating a server, message tampering, session teardown, Dos etc. This paper discusses various security attacks and their impact on VoIP communication. A novel authentication scheme based on Needham Schroeder authentication protocol is also proposed along with the defenses it provides against various security attacks.

Keywords - *VoIP; session initiation protocol; TCP; UDP; SCTP; Needham Schroeder authentication protocol.*

1. Introduction

Session Initiation Protocol (SIP), proposed by IETF operates at the application layer of the OSI communication model for initiating, maintaining and terminating real time multimedia sessions. It makes use of UDP, TCP and SCTP (Stream Control Transmission Protocol) on the transport layer. It carries session information in combination with other application layer protocols. SIP is based on a client-server model using request and response. SIP is based on a client-server model. It makes use of requests and responses. Its address formation is similar to e-mail, having unique identifier indicated by telephone number and unique domain identifier. SIP endpoints are known as user agents (UA). UAs can either act as clients or servers. The User Agent Client (UAC) initiates a call, and the User Agent Server (UAS) receives a call. Other functional components of SIP are discussed below [1] [2].

- **Proxy server:** A proxy server acts as an intermediary computer between the user's computer and the Internet. Proxy server keeps on forwarding call to other proxy servers till it reaches its destination (UAS).
- **Redirect server:** UAs and proxy servers communicate with redirect server for finding the location of an endpoint (User Agent).
- **Registrar server:** UACs and UASs register their respective location with a registrar server. Registrar server places that location information into its location database.
- **Location server:** This server holds the location database for registered UAs.
- **Back-to-back user agent (B2BUA):** B2BUA behave as a UA server (UAS) and client (UAC) at the same time. It aborts the signaling from the UAC and initiates signaling to the UAS.
- **Presence server:** This server accumulates the presence and subscription information and forwards status notifications [2].

SIP makes use of plain-text messages which helps in troubleshooting. SIP message contains message body and message header [2]. SIP messages can be either a request or response to a request. Following are the request messages defined in SIP: INVITE, BYE, REGISTER, ACK and CANCEL [3]. Status codes of SIP responses with explanation is given in [2] *Trying, ringing* and *OK* response are assigned status codes *100, 180 and 200* respectively.

SIP call flow contains a SIP User Agent Client (UAC) that sends a request to the SIP User Agent Server (UAS) to invite UAS to a session. UAC sends the request to a proxy or redirect server for locating the UAS. Proxy server may forward the request to other proxy servers until it reaches UAS. SIP UAs register themselves with a proxy server or a registrar. Proxy servers will then act as an intermediary developing session setup [1]. SIP URI is a name that can be resolved to an IP address for reaching a specific user.

This is done by making use of DNS lookups and proxy servers.

Fig. 1 explains the scenario in which proxy servers are used for initiating a call session between UAC and UAS. As UAC does not know IP address of UAS, a SIP proxy server is used for setting up session. The procedure starts by DNS lookup of UAS's SIP URI host name (bth.se). This generates IP address of SIP proxy server (kariskrona.bth.se). The *invite* request is sent by UAC to SIP proxy server. The SIP proxy server locates UAS's IP address in its database. The SIP proxy server then forwards the request directly to UAS. Two steps are performed in this process.

1) DNS lookup performed by UAs for finding the IP address of SIP proxy server
2) Database lookup performed by SIP proxy server for locating the IP address of UAs.

In response to *invite* request received, UAS sends *180 ringing* response to SIP proxy server. The SIP proxy server forwards that response to UAC by dispatching the previous address holding header (kariskrona.bth.se). The SIP proxy server also sends *200 OK* message to UAC when UAS accepts a call, by again dispatching the previous header [1] [4]. In SIP registration process, UA sends a SIP *register* request to registrar server. The *register* request consists of SIP URI address and c*ontact* URI depicting IP address of UA. The SIP registrar makes use of this information for routing SIP request and updating it to database of proxies. The *contact* URI is stored and then acknowledged by sending *200 OK* response to UA. The response contains UA's information and registration expiry time that tells about registration validity duration [1]. Fig 2 depicts the registration process of UAC.

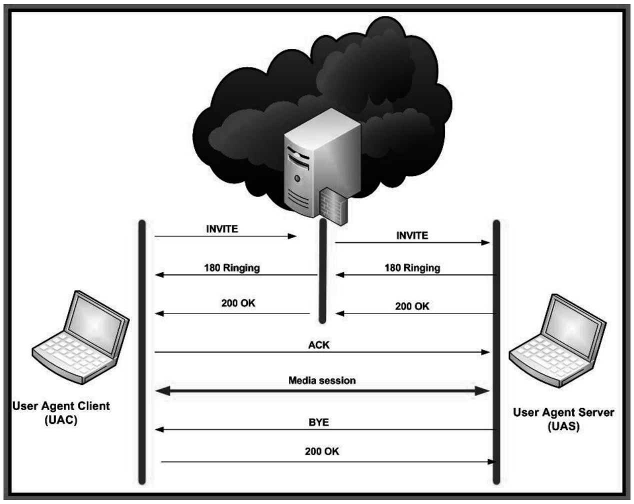

Fig. 1 Call flow of SIP using proxy server [7]

The security of SIP has attained much attention in today's networks environment. Due to its open architecture, text-based nature of messages and utilization of TCP, SCTP and UDP, some serious attacks (registration hijacking, impersonating a server, message tampering, session teardown, DoS(denial of service) attacks can be carried out against SIP. All these attacks exploit SIP's weak authentication scheme. Identification of participating users is the most important consideration in SIP. There is a high need of mutual authentication between two parties that are engaged in a session.

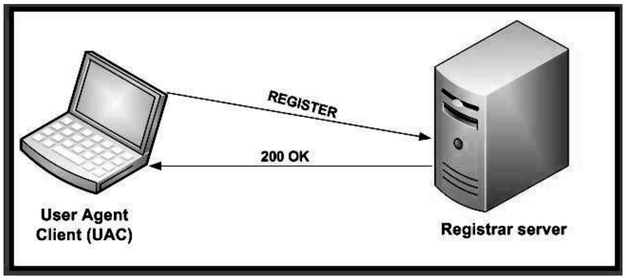

Fig. 2 Registration process of UAC [7]

An attacker may hijack secret information of users through impersonating server if authentication is not done. For achieving confidentiality, encryption is performed. This encrypted message will then can only be decrypted by valid recipient. Various authentication schemes have been proposed against attacks on SIP authentication and will be discussed in the following sections.This paper is organized as follows. Section 2 presents security attacks in SIP followed by literature review of techniques that have been proposed for SIP authentication in section 3. Section 4 discusses a proposed solution based on Needham-Schroeder public-key authentication protocol and possible attack scenarios. Section 5 concludes the paper and Section 6 presents future recommendations.

2. Security Attacks against SIP

All of the entities involved in SIP based VoIP networks i.e., UAs, proxy servers and registrar server are targets for different types of security attacks. For example the security of UAs can be compromised using dictionary based password attacks. Other types of attacks target the communication session between UAs, proxy server and registrar server. These include registration hijacking, session hijacking, impersonation, man-in-the-middle, stolen-verifier and offline password guessing. The following sections describe these in greater detail.

2.1 Registration Hijacking

In SIP, UAs must have to get registered with SIP proxy and registrar server that allows proxy to carry out incoming calls to the UAs. Registration hijacking attack is launched by an attacker by impersonating a valid UA to registrar server and then replacing the legitimate valid UA's registration with its own address. Due to this attack, incoming calls that have to be sent to valid UA are hijacked and sent to attacker impersonating as a UA [4]. Fig. 3 explains this attack.

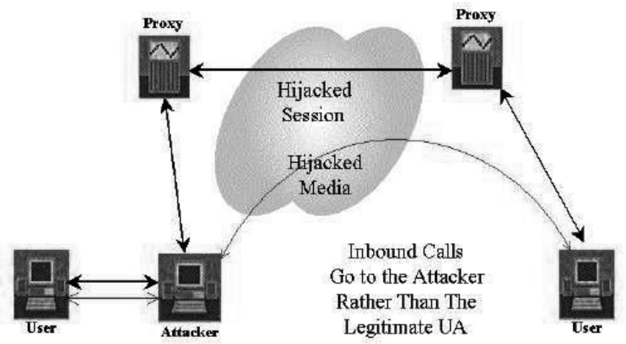

Fig. 3 Registration Hijacking attack [4]

2.2 Impersonating A Server

This attack occurs when an attacker puts himself between proxy or registrar servers and UAs. The *request-URI* usually carries information about domain on which SIP request has to be sent. A proxy or registrar server in that domain can be directly approached by UAs for sending request. There is a much chance for an attacker to impersonate as a remote server and intercepts the UA's request [5]. Fig. 4 explains this attack.

.

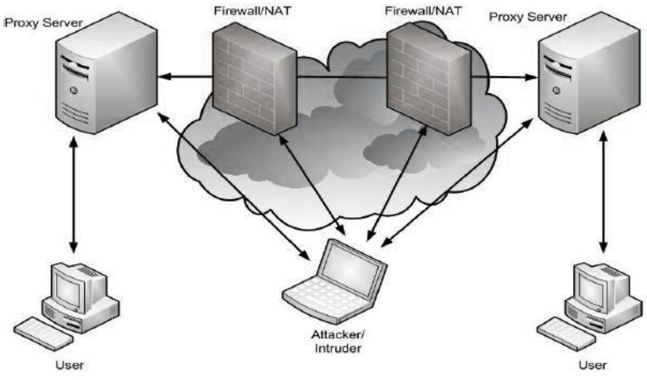

Fig. 4 Server impersonation attack [7]

2.3 Message Tampering

This attack occurs when attackers grabs and improvise/fabricate the SIP messages. Message tampering attacks occur mainly due to registration hijacking, server impersonation and other attacks on honored entities such as firewall. SIP message headers are of great importance.

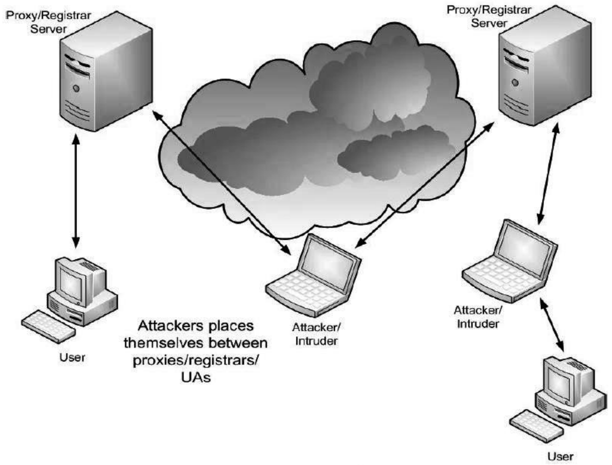

Fig. 5 Message tampering attack [7]

Specifically the *subject* header field carries information about media session [5]. Fig. 5 depicts that attack.

2.4 Session Tear Down

This type of attack occurs when an attacker uses *bye* request to terminate and tear down the session between UAs. To accelerate this attack, attacker must have to learn necessary parameters of session by sniffing or through man-in-middle attacks [4] [6]. Fig. 6 describes this attack.

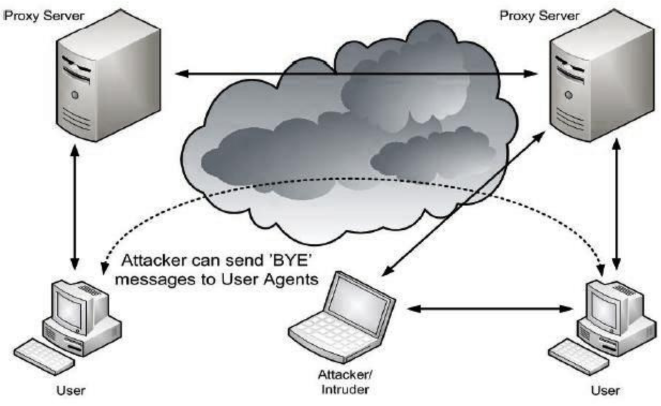

Fig. 6 Session tear down attack [7]

2.5 DoS Attack

This attack is used to make a particular SIP network entity inaccessible. This is done by bombarding a network interface with huge number of packets, i.e. by flooding and overflowing with SIP requests such as *register* and *invite* requests [5]. Fig. 7 shows DoS attack. In it, an attacker is shown launching DoS attack by sending excessive *BYE* messages to victim user thereby terminating the session between two users.

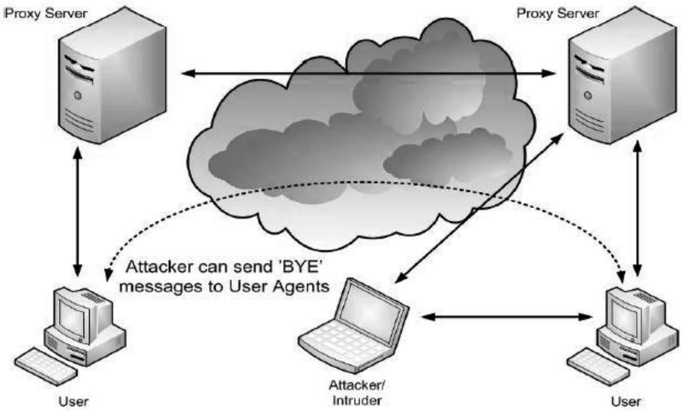

Fig. 7: DoS attack [7]

2.6 Password Guessing Attack

Password guessing attack is an attack in which the attacker interferes in the conversation between UAs. In this way true secret passwords can be guessed and verified on the basis of verifying whether the guess is correct or not [8] [9].

2.7 Replay Attack

Replay attack is an offending attack in which an attacker impersonates and deceives legitimate UAs by capturing session messages and modifying and replaying them to intercept the session [8] [9].

2.8 Man-In-The-Middle Attack

Man-in-the-middle attack is one of the most serious and malicious attack on SIP security, in which an attacker intercepts and track the communication between UAs. An attacker hijacks all messages between UAs [10].As shown in a Fig. 8, an attacker intercepts all messages.

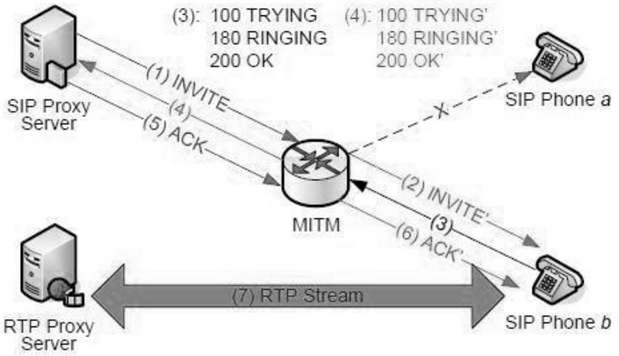

Fig. 8 Man-in-the-middle attack [10]

2.9 Stolen-Verifier Attack

Stolen-verifier attack is an attack in which an attacker by using stolen password can directly impersonate as a legitimate UA [9].

2.10 Denning-Sacco Attack

Denning-Sacco attack occur when an old session key is compromised and an attacker attempts in finding other session keys. By this secret password of UAs may get compromised and through this an attacker can impersonate as SIP UAs [11].

2.11 Offline Password Guessing Attack

In this attack, an attacker has an access to password hashes. These attacks are different from online password guessing attack in which an attacker do not has access to password hashes [12].

3. Existing Authentication Schemes for SIP

Various approaches have been proposed to guard SIP against security attacks. Http based digest authentication has been proposed [13]. HTTP authentication contains two categories: HTTP basic authentication and HTTP digest authentication. HTTP basic authentication is based on username and password. Passwords of users are sent through encoding, without encryption. That’s why it is not secure authentication protocol. HTTP digest is more secure than http basic authentication because it communicates user credentials using encryption, by applying hash function [14]. Http digest authentication scheme makes use of challenge and response message to authenticate communicating parties. When a UAC wishes to establish a session with UAS, then the server (proxy, registrar etc) sends unauthorized response and challenges the identity of UAC. The UA then initiate a request with credentials. This scheme is unable to handle server spoofing and offline password guessing attack. To overcome this limitation Diffie-Hellman scheme of key exchange has been proposed [15].

In Diffie-Hellman, two parties independently compute the same value for session key to be used. This is accomplished through computation of key factors as exponents of large prime numbers, resulting in a unique pair of values. Public number in computed unique pair is shared publically and other one is kept secret. This scheme is vulnerable to man in the middle attack. Also, it involves exponential operations that are expensive making it unsuitable for devices having low computation power. In addition to this, computation time is quite large. However it is secure against passive attacks. To overcome these challenges, Elliptic Curve Cryptography (ECC) having key size of 160 bits has been proposed that is based on Elliptic Curve Discrete Logarithm (ECDL) problem [15]. It turned out to be comparatively faster than Diffie-Hellman regarding memory usage and execution time, providing same level of security as RSA with 1024 bits key [15].

However, ECC is vulnerable to man in the middle attack. In 2008, another solution based on ECC was proposed by Wu et al. [16]. This solution is not completely secure against off-line password guessing attacks. In 2009, a new scheme based on ECDLP was proposed by Yoon and Yoo. It is also efficient as compared to previously proposed schemes. However, it is vulnerable to password guessing, replay attack and stolen verifier attack. In 2011, another efficient scheme based on ECDLP was proposed by R. Arshad et al. [17].This scheme turned out to be faster than other schemes but it is still vulnerable to off-line password guessing attack. In 2012, a new scheme was proposed by Tang et al. [18]. This scheme is also based on ECDLP, but it is still not secure against offline-password guessing attack. In 2012, Sadat et al proposed a new scheme based on ECC. They show that this scheme is highly efficient and is secure against several attacks [19]. In 2008, Tsai proposed an authentication scheme based on random nonce [20]. This scheme has low computational cost and is suitable for low computation equipment. But this scheme is still not safe against offline password guessing attacks and stolen-verifier attack. In 2006, Ring et al proposed a scheme based on identity-based cryptography. In ID authentication mechanism a private key is generated by user's public key and safely given to user using trusted third party (TTP). Hash value of user's identity is used as public key of user without the need of having certificates [21]. This scheme has significant advantage of not suffering from security problems related to passwords. Also, it avoids the difficulties involved in PKI certificate. This scheme has high computation involved. In 2009, a new scheme was proposed by H.H Kilinc which integrates Elliptic Curve Digital Signature Algorithm (ECDSA) and ID based authentication scheme [22]. This scheme turned out to be faster and is also secure against spoofing and password guessing attacks. However, it has limitation in terms of large signature size. In 2012, signature based secure authentication scheme was proposed by Rongwei Tu et al.[23] This scheme can guard against several attacks and computational cost is low. In 2008, a scheme based on HTTP Digest authentication was proposed [24]. A new header *Integrity-Auth* header is involved which prevents signaling attacks. However, this scheme is vulnerable to offline password guessing and stolen-verifier attack. In 2009, a new scheme was proposed by Guillet and et al. [25] based on HMAC one time password (HOTP) authentication. It lessens the handshakes to one. It is very efficient and fast. However, it has limitation in terms of not securing against MITM attack. In 2009, a new scheme using self-certified public keys (SCPKs) was proposed by Yi-Pin Liao and Wang [26].This scheme gives mutual authentication and is secure against several attacks. It also has low computational cost.

4. Proposed Solution: Using Needham-Schroeder Authentication Protocol

An authentication protocol ensures that a message came from legitimate user. Needham-Schroeder protocol is an authentication protocol that can be used for providing mutual authentication through use of nonce and confidentiality using encryption techniques. It is used for catering against several attacks to SIP security. The individuals communicating to each other and having a SIP session authenticates each other, resisting against several attacks. This protocol guards against many attacks and also provides security against password guessing attack also [31]. The protocol comes in two versions: (i) using symmetric keys and (ii) using a-symmetric keys.

4.1 Needham-Schroeder Symmetric Key Protocol

The Needham Schroeder's protocol involves the following entities in two way communication.

- User Agent Client= (UAC)
- User Agent Server= (UAS)
- Trusted third party=(T)
- Nonce sent by UAC =N (UAC)
- Nonce sent by UAS =N (UAS)
- Secret session key=KS
- y{x}= encryption of x using y
- P_{UAC}= public key of UAC
- P_{UAS}=public key of UAS

The protocol follows the following steps for authentication of both parties. UAs commonly cannot communicate to each other directly. They do this through proxy servers. Proxy server will act as a relay server in a communication between UAs. It is responsible for taking call and session parameters from UAC and forwards it to other proxy servers till it finds a UAS.

1) UAC →T : UAC, UAS, N(UAC)
2) T →UAC : PU_{UAC} { KS, N(UAC), UAS, PU_{UAS}{KS,UAC} }
3) UAC→ UAS : PU_{UAS} { KS, UAC}
4) UAS→ UAC : KS {N(UAS)}
5) UAC→ UAS : KS {N(UAS)-1}

In step 1, message is transmitted to trusted party (as indicated by arrow head). UAC sends to trusted party his and UAS's identity and a nonce. In step 2, trusted party generates a secret key KS and sends a message in encrypted form to UAC that ensures that message is fresh due to nonce. In step 3, UAC sends a message to UAS, which includes a secret session key KS and identity of A encrypted using the key shared between trusted party and UAS. In step 4, UAS sends a message to UAC, including a nonce that is encrypted using secret key KS. Then finally at step 5, UAC sends a message to UAS. This message includes in encrypted form, a nonce with operation performed on it. Needham-Schroeder Symmetric Key Protocol's architecture is shown in Fig. 8.

This protocol does not provide security against replay attack. If an attacker is able to record one round of this protocol then he will get to know the secret key KS being used He can simply replay the message containing in encrypted form secret key and identity of A. Because key is not fresh, UAS will accept it. A solution to this problem is the usage of timestamps in this protocol that will prevent replay attack [27].

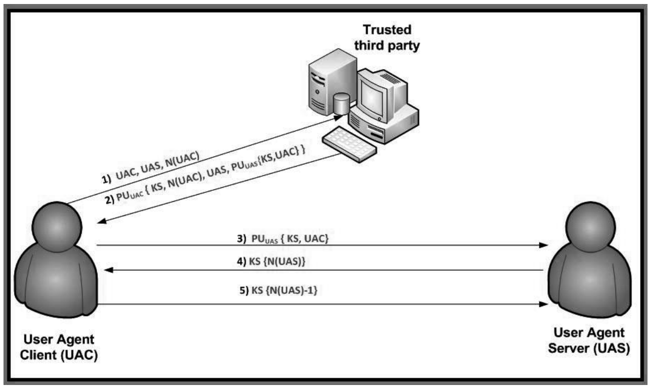

Fig. 8 Needham-Schroeder Symmetric Key Protocol's architecture

Following are the steps for Needham Schroeder symmetric key protocol which involves usage of timestamps for preventing replay attack:

1) UAC →T : UAC, UAS, N(UAC), $\mathbf{T_{UAC}}$
2) T →UAC : PU_{UAC} { KS, N(UAC), UAS, PU_{UAS}{KS,UAC}, $\mathbf{T_{UAC}}$ }
3) UAC→ UAS : PU_{UAS} { KS, UAC, $\mathbf{T_{UAC}}$}
4) UAS→ UAC : KS {N(UAS), $\mathbf{T_{UAC}}$, $\mathbf{T_{UAS}}$}
5) UAC→ UAS : KS {N(UAS)-1, $\mathbf{T_{UAS}}$}

$\mathbf{T_{UAC}}$ is the timestamp generated by UAC for assuring that the reply to the message is fresh. Similarly, $\mathbf{T_{UAS}}$ is the timestamp generated by UAS. Through use of timestamps both parties will get to know whether the message is fresh or not, thus preventing replay attack.

4.1.1 Possible Attacks

This section proves the security of this protocol against some attacks.

4.1.1.1 Scenario 1

In this scenario, an attacker launches impersonation attack and impersonates as a UAC to trusted party T.

1) UAC→T: UAC, UAS,N(UAC)
2) T→**Attacker**(UAC) : PU_{UAC} { KS, N(UAC), UAS, PU_{UAS}{KS,UAC} }

In the steps shown above, attacker will not be able to perform decryption of message to UAC. Attacker forwards it without changing anything and cannot do any harm.

4.1.1.2 Scenario 2

In this scenario, an attacker launches message tampering attack. Attacker tampers a message that is to be transferred from UAC to trusted party T and impersonates as a UAS. Attacker changes the identity of UAS with its own identity and sends this to T.

1) UAC →**Attacker**(T) : UAC, UAS, N(UAC)
2) **Attacker**(UAC)→T: UAC, **Attacker**, N(UAC)
3) T→ UAC: PU_{UAC} {KS,N(UAC), **Attacker**, PU_{UAS} {KS, UAC}}

UAC will get to know that the message contains identity of an attacker instead of UAS, so attack will not be successful.

4.1.1.3 Scenario 3

In this scenario, an attacker launches message tampering attack. Attacker tampers a message that is to be transferred from UAC to trusted party T and impersonates as a UAC. Attacker changes the identity of UAC with its own identity and sends this to T. As T got attacker's identity, so it generates public key based on attacker's identity.

1) UAC →**Attacker**(T) : UAC, UAS, N(UAC)
2) **Attacker**→T: **Attacker**, UAS, N(UAC)
3) T →**Attacker** : $PU_{\textbf{Attacker}}$ { KS, N(UAC), UAS, PU_{UAS}{KS, **Attacker**} }
4) **Attacker**(T) →UAC : $PU_{\textbf{Attacker}}$ { KS, N(UAC), UAS, PU_{UAS}{KS, **Attacker**} }

This attack will also not be successful because UAC will not be able to decrypt the message.

4.1.1.4 Scenario 4

This scenario is the same as scenario 3. If in step 4 of the scenario 3, attacker somehow succeeds then in the next step (as indicated by step 3 of scenario 4) attacker sends secret key and his identity all encrypted using public key of UAS.

1) **Attacker** →T : **Attacker**, UAS, N(UAC)
2) T → **Attacker**: $PU_{\textbf{Attacker}}$ { KS, N(UAC), UAS, PU_{UAS}{KS, **Attacker** } }
3) **Attacker**(UAC) → UAS : PU_{UAS} { KS, **Attacker** }

This attack will not be successful because UAS will get to know that attacker's identity of impersonated UAC is not the same as the one indicated by trusted party [28].

4.2 Needham-Schroeder Asymmetric Key Protocol

The Needham Schroeder's asymmetric key protocol involves the following entities in two way communication.

- Private session key=PR_k
- PU_{UAC}= public key of UAC
- PU_{UAS}=public key of UAS

Proxy server(s) will also be involved in a communication between UAs as described in a previous section. Following are the seven steps involved in this protocol.

1) UAC→T: UAC,UAS
2) T→UAC: PR_K {PU_{UAS},UAS}
3) UAC→UAS: PU_{UAS} {N(UAC),UAC}
4) UAS→T: UAS, UAC
5) T→UAS: PR_K {PU_{UAC}, UAC}
6) UAS→UAC: PU_{UAC} {N(UAC), N(UAS)}
7) UAC→UAS: PU_{UAS} {N(UAS)}

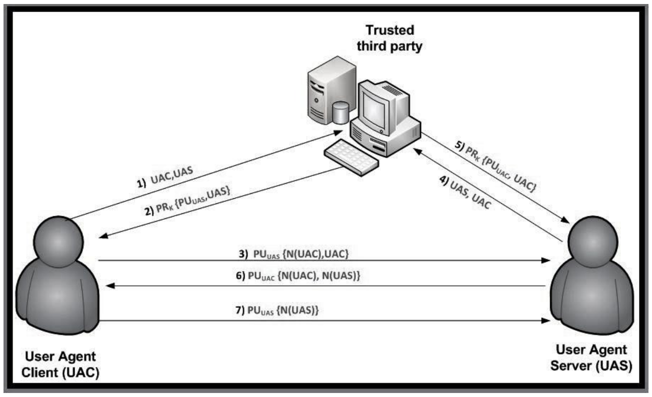
Trusted
third party
1) UAC,UAS
2) PR_K {PU_{UAS},UAS}
5) PR_K {PU_{UAC}, UAC}
4) UAS, UAC
3) PU_{UAS} {N(UAC),UAC}
6) PU_{UAC} {N(UAC), N(UAS)}
7) PU_{UAS} {N(UAS)}
User Agent
Client (UAC)
User Agent
Server (UAS)

Fig. 9 Needham-Schroeder Asymmetric/Public Key Protocol's architecture

In step 1, UAC sends his/her identity along with UAS's identity. In step 2, trusted party sends a reply to UAC containing UAS's key and UAS's identity all encrypted using secret key PR_K. In step 3. A message is sent to UAS consisting of UAC's nonce and identity encrypted through UAS's key. In step 4, UAS sends a message to trusted party containing identity of UAC and UAS. In step 5, trusted party sends a message to UAS that contains key of UAC and his/her identity encrypted using secret key PR_K. In step 6, UAS sends to UAC, nonce of UAC and UAS, all encrypted using key of UAC. In the final step, UAC sends a message to UAS, a nonce generated by UAS and this is encrypted through UAS's key. Steps 1, 2, 4 and 5 get public keys and in steps 3, 6 and 7 UAC and UAS authenticate each other. Needham-Schroeder Asymmetric/Public Key Protocol's architecture is shown in Fig. 9.One issue in this protocol is that there is no guarantee of fresh public keys used in this protocol. Replay attacks can occur. To guard against it time stamps must be used [29].

4.2.1 Possible Attacks

4.2.1.1 Impersonation Attack

This protocol is vulnerable to impersonation attack. An attacker can pretend to be a UAS and can establish a session with legitimate UAC. Consider the following scenario:

1) UAC→T: UAC,UAS
2) T→UAC: PR_k {PU_{UAS},UAS}
3) UAC→**Attacker:** $\textbf{PU}_{\textbf{Attacker}}$ {N(UAC),UAC}
4) UAS→T: UAS, UAC
5) T→UAS: PR_k {PU_{UAC}, UAC}
6) UAS→ **Attacker(UAC)** : PU_{UAC} {N(UAC), N(UAS)}
Attacker→UAC: PU_{UAC} {N (UAC), N (UAS)}
7) UAC→ **Attacker**: K(**Attacker**) {N(UAS)}
Attacker (UAC) → UAS: PU_{UAS} {N (UAS)}

This illustrates how an attacker impersonates as a UAS and successfully creates a false session with UAC. To prevent this attack, step 6 of this protocol's procedure needs amendment as follows.

6) UAS→ UAC: PU_{UAC} {N(UAC), N(UAS), UAS}

Through this, an attacker would not be able to impersonate as a UAS because in step 6, identity of UAS is being included and attacker cannot replay a message [30].

5. Analysis of Proposed Scheme

Table 1 provides a comparative analysis of the security of original SIP protocol and the proposed scheme. Since the authentication scheme involves use of encryption and exchange of nonce values, it provides better security against various attacks discussed in the previous sections. The authentication scheme involving asymmetric keys proves secure against all types of attacks; however, it is computationally slow due to using public key infrastructure. Also, since the proxy servers are involved at each step of call establishment, they authentication scheme causes extra overhead in terms of exchanging authentication messages through all proxy servers before the actual communication can take place. There could be two solutions to this problem.

1. The communicating UAC and UAS can authenticate each other before the actual call establishment phase starts. This still has a disadvantage: authentication will require the UAC to determine the location of the UAS and therefore, proxy servers will be involved during this pre-call-establishment phase, adding extra delay.
2. The communicating UAC and UAS can obtain the respective public keys from the TTP and later use them to authenticate each other along with the call establishment messages requests sent through the proxy servers. Thus, steps 3, 6 and 7 of the asymmetric Needham-Schroeder's protocol can be integrated in SIP call establishment phase.

6. Conclusion

SIP is vulnerable to several attacks due to weak authentication mechanism of SIP. Previously proposed solutions are vulnerable to some attacks along with password guessing attack. In this paper, a secure authentication scheme is proposed which is based on Needham Schroeder protocol. Various attacks on this protocol and their solutions are illustrated. This protocol guards against many attacks including replay attack, message tampering attack and impersonation attacks as described in this paper. Also it provides security against password guessing attack too.

Table 1: Security Analysis of Proposed Scheme

Attacks	*Original SIP*	*Proposed Authentication Scheme Using Symmetric Key Protocol*	*Proposed Authentication Scheme Using Asymmetric Key Protocol*
Impersonation Attack	insecure	secure	secure
Man-in-the-Middle Attack	insecure	secure	secure
Message Tampering Attack	insecure	secure	secure
Password Guessing Attacks	insecure	secure	secure
Replay Attack	insecure	secure	secure
DoS Attack	insecure	insecure	insecure

7. Future Recommendations

One drawback in proposed protocol is the inefficiency involved due to usage of nonce and many complex cryptographic operations. Also because of involvement of trusted third party it may be difficult and time consuming to get the public key of that trusted party. Future research should focus on making amendments in the proposed protocol to make it practical and feasible.

References

[1] Alan B. Johnston, SIP:Understanding the Session Initiation Protocol (Second Edition), ISBN-9781580536561, Artech House Inc, 2003.

[2] D. Donohue, D. Mallory, K. Salhoff, Session Initiation Protocol, Cisco Press, 2006. http://www.ciscopress.com/articles/article.asp?p=664148

[3] The SIP Servlet Tutorial, Sun Microsystems, Network Circle Santa Clara, CA 95054, U.S.A, 2009. http://docs.oracle.com/cd/E19355-01/820-3007/gfnfb/index.html

[4] J. Rosenberg, et al., SIP: Session Initiation Protocol, RFC 3261, 2002. http://tools.ietf.org/html/rfc3261

[5] M. Collier, "VoIP Vulnerabilities", SecureLogix Corporation, 2005.

[6] S. El Sawda and P. Urien, "SIP Security Attacks and Solutions: A state-of- the-art review", in IEEE, 2006, Vol. 2, pp. 3187 – 3191.

[7] A. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, 2002. http://tools.ietf.org/html/rfc3265

[8] G. Asghar and Q. Jawed Azmi, "Security issues of SIP", M.S. Thesis no: MEE10:74, Department Of Telecommunication Systems, Blekinge Institute Of Technology School Of Engineering, 2010.

[9] H. Belaoud, J. El Abbadi, A. Habban, “Survey Of Sip Authentication Mechanisms”, Journal of Theoretical and Applied Information Technology, Vol. 58. No. 2, 2013.

[10] E.J. Yoon, E.K. Ryu, K.Y. Yoo, “Attacks and Solutions of Yang et al.’s Protected Password Changing Scheme”, Informatica, , Vol. 16 No.2, pp. 285–294, 2005.

[11] R. Zhang, X. Wang, R. Farley, X.Yang, X. Jiang, “On the Feasibility of Launching the Man-In-The-Middle Attacks on VoIP from Remote Attackers”, in ASIACCS '09 Proceedings of the 4th International Symposium on Information, Computer, and Communications Security , 2009, pp. 61-69.

[12] E.J. Yoon, K.Y. Yoo, C. Kim, Y.S. Hong, M. Jo, “A secure and efficient SIP authentication scheme for converged VoIP networks”, Computer Communications, Vol. 33, 2010, pp. 1674-1681.

[13] R. Hande, “Password Cracking– Online vs. Offline Password Cracking”,Scribd, 2012.

[14] J. Franks, HTTP Authentication basic and digest access authentication, IETF RFC2617, 1999.

[15] Niemi, A., et al., “Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)”, RFC 3310, 2002. http://www.hjp.at/doc/rfc/rfc3310.html

[16] C. C. Yang, R. C. Wang, and W. T. Liu, "Secure authentication scheme for session initiation protocol”, in Computers & Security, 2005, Vol. 24, pp. 381-386.

[17] A. Durlanik and I. Sogukpinar, “SIP authentication scheme using ECDH”, International Journal of Computer, Information, Mechatronics, Systems Science and Engineering, Vol. 1, No.8, 2007.

[18] L. Wu, Y. Zhang, F. Wang, "A new provably secure authentication and key agreement protocol for SIP using ECC", Computer Standards & Interfaces, Vol. 31, 2009, pp. 286–291.

[19] R. Arshad, N. Ikram, "A novel mutual authentication scheme for session initiation protocol based on elliptic curve cryptography”, in 13th International Conference on Advanced Communication Technology (ICACT), 2011, Vol. 13, pp. 705-710.

[20] H. Tang, X. Liu, "Cryptanalysis of Arshad et al.’s ECC-based mutual authentication scheme for session initiation protocol", Multimedia Tools and Applications, Vol. 65, 2013, pp. 321-333.

[21] S.S. Mousavi-nik, M.H. Yaghmaee-moghaddam and M.B. Ghaznavi-ghoushch, “Proposed SecureSIP Authentication Scheme based on Elliptic Curve Cryptography”, International Journal of Computer Applications, Vol. 58, No.8, 2012, pp. 25-30.

[22] J. L. Tsai, “Efficient nonce-based authentication scheme for session initiation protocol”, International Journal of Network Security, Vol. 8, No. 3, 2009, pp. 312- 316.

[23] J. Ring, K.K. R. Cho, E. Foo, M.H. Looi, “A new authentication mechanism and key agreement protocol for SIP using identitybased cryptography”, in Information Technology Security Conference, 2006, pp. 61–72.

[24] H.H. Kilinc, Y. Allaberdiyev, T. Yanik, “Performance Evaluation of ID Based Authentication Methods in the SIP Protocol”, in proceeding of Application of Information and Communication Technologies (AICT), 2009, pp. 1-6.

[25] R. Yu, J. Yuan, G. Du, P. Li, "An identity-based mechanism for enhancing SIP security", in IEEE 3rd International Conference on Software Engineering and Service Science (ICSESS), 2012, pp. 447-451.

[26] D. Geneiatakis, C. Lambrinoudakis, “A lightweight protection mechanism against signaling attacks in a SIP-Based VoIP environment”, Telecommunication Systems, Vol. 36, 2008, pp. 153–159.

[27] T. Guillet, R. Moalla, A. Serhrouchni, A. Obaid, "SIP authentication based on HOTP”, in International Conference on Information, Communications and Signal Processing (ICICS), 2009, pp. 1-4, 8-10.

[28] Y.P. Liao, S.S Wang, “A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves”, Computer Communications, Vol. 33, Issue 3, 2009, pp. 372-380.

[29] M. Kumar, A. Tuli, R. Tuli, “Secure Communication Using Needham-Schroeder Protocol”, CPMR-IJT, Vol. 1, No. 1, 2011.

[30] Introduction to Networks and Security, Lecture 29, CSE331, 2006. http://www.cis.upenn.edu/~cse331/lectures/CSE331-29.pdf

[31] G. Lowe, “An attack on the Needham-Schroeder public-key authentication protocol”, Information Processing Letters archive, Vol. 56, Issue 3, 1995, pp. 131 - 133.

[32] M. Abadi and R. Needham, “Prudent Engineering Practice for Cryptographic Protocols”, IEEE Transactions on Software Engineering Vol.22, No.1, 1996, pp. 6-15.

[33] W.H. Yang, J.C. Shen, S.P. Shie, “Designing authentication protocols against guessing attacks”, Technical Report 2(3), Institute of Information & Computing Machinery, Taiwan, 1999.

Natalia Chaudhary is a student of BS(Hons) in Computer Science at Kinnaird College for Women University Lahore.

Rabia Sirhindi has received a BS(Hons) in Computer Science from University of Punjab and MS in Information Security from National University of Sciences and Technology. She is presently serving as Lecturer at Kinnaird College for Women University Lahore.