An Analysis Of Session Key Exchange Protocols

Pranav Vyas*, Dr. Bhushan Trivedi**

*(Smt.Chandaben Mohanbhai Patel Institute Of Computer Application, Charotar University Of Science And Technology, Changa

** (GLS Institute Of Computer Technlogy, Ahmedabad

ABSTRACT

Information security has been one of the most important aspects in today's technology driven world. By encrypting information we can secure information from unauthorized access, even in case where use has information he/she can not make out meaning of message unless they have a key to decrypt information. We review different techniques to exchange keys between different computers and try to find best suitable technique for mobile computers which have limited processing power and battery capacity while efficiently working on wireless network.

Keywords **– Session Key, Protocols, Mutual Authentication, User Anonymity, Forward Secrecy**

1. INTRODUCTION

As technological advancements are made by humankind, more and more technology is integrated into our lives in different ways. We use technology to communicate, to perform financial transactions, share personal information over public network etc. We need to protect the information we have over the public network so that no one but intended recipient(s) can view the information. This can be done with encrypting information in such a way that information can only be meaningful to recipients who have key to decrypt information.

This paper reviews some of the leading protocols for key exchange between different parties. This key once exchanged is used to decrypt information which is encrypted by sender. We review this paper based on various parameters. This communication whether it is financial, medical or personal takes place on public networks most of the time. These public networks are highly insecure. So, we study if the protocols are good enough to be used on public networks. We also check protocols for any known vulnerabilities and if there are any variants of protocol which addresses vulnerabilities found in original protocol. We also check use of nonce and timestamps as they play important role in identifying validity of message.

We start this paper with introduction to topic and then move to describing protocol s and giving out steps of algorithm to explain how each protocol works. We then move on to describe the parameters we choose to compare protocols on and give a brief idea on importance of each parameter. We compare protocols in next section on each parameter. We

hen present our findings in a tabular format and in last we give conclusion based on our findings after reviewing protocol.

2. PROTOCOL DESCRIPTION

Here we provide brief details on working of various protocols. Following is table of symbols that we have used here:

Symbol	Description
A	Alice's name
B	Bob's name
*E*A	Encryption with a key Trent shares with Alice
*E*B	Encryption with a key Trent shares with Bob
I	Index number
K	A random session key
L	Lifetime
*T*A,*T*B, *T*S	A timestamp
*R*A,*R*B	A nonce, chosen by Alice and Bob respectively
G	Ticket granting server

Table1. Description of symbols used in describing protocols.

2.1 Wide Mouth Frog protocol

Q The Wide-Mouth Frog protocol [1, 2] is probably the simplest symmetric key-management protocol that uses a trusted server. Both Alice and Bob share a secret key with Trent. The keys are just used for key distribution and not to encrypt any actual messages between users. Just by using two messages, Alice transfers a session key to Bob. Following is list of steps on how this protocol works:

2.1.1 Alice concatenates a timestamp, Bob's name, and a random session key and encrypts the whole message with the key she shares with Trent. She sends this to Trent, along with her name: A,EA(TA,B,K)

2.1.2 Trent decrypts the message from Alice. Then he concatenates a new timestamp, Alice's name, and the random session key; he encrypts the whole message with the key he shares with Bob. Trent sends to Bob: EB(TB,A,K)

2.2 Neuman-Stubblebine

This protocol, first presented in [3] and corrected in [4] attempts to counter the suppress-replay attack. It is an enhancement to Yahalom.

2.2.1 Alice concatenates her name and a random number and sends it to Bob. A,RA

2.2.2 Bob concatenates Alice's name, her random number, and a timestamp, and encrypts with the key he shares with Trent. He sends it to Trent along with his name and a new random number.

2.2.3 Trent generates a random session key. Then he creates two messages. The first is Bob's name, Alice's random number, a random session key, and the timestamp, all encrypted with the key he shares with Alice. The second is Alice's name, the session key, and the timestamp, all encrypted with the key he shares with Bob. He sends these both to Alice, along with Bob's random number. EA(B,RA,K,TB),EA(A,K,TB),RB

2.2.4 Alice decrypts the message encrypted with her key, extracts K, and confirms that RA has the same value as it did in step (1). Alice sends Bob two messages. The first is the message received from Trent, encrypted with Bob's key. The second is RB, encrypted with the session key. EB(A,K,TB),EK(RB)

2.2.5 Bob decrypts the message encrypted with his key, extracts K, and confirms that TB and RB have the same value they did in step (2.2.2).

2.3 Karbaros Authtentication System[5]

Kerberos is a variant of Needham-Schroeder. In the basic Kerberos Version 5 protocol, Alice and Bob each share keys with Trent. Alice wants to generate a session key for a conversation with Bob. This protocol assumes that everyone's clocks are synchronized with Trent's clock. Synchronization effect is obtained by synchronizing clocks to within a few minutes of a secure time server and detecting replays within the time interval. Following is list of steps how the protocol works:

2.3.1 Alice sends a message to Trent with her identity and Bob's identity. A, B.

2.3.2 Trent generates a message with a timestamp, a lifetime, *L*, a random session key, and Alice's identity. He encrypts this in the key he shares with Bob. Then he takes the timestamp, the lifetime, the session key, and Bob's identity, and encrypts these in the key he shares with Alice. He sends both encrypted messages to Alice**.** EA(T,L,K,B),EB(T,L,K,A)

2.3.3 Trent generates a message with a timestamp, a lifetime, *L*, a random session key, and Alice's identity. He encrypts this in the key he shares with Bob. Then he takes the timestamp, the lifetime, the session key, and Bob's identity, and Alice's identity. He encrypts this in the key he shares with Bob. Then he takes the timestamp, the lifetime, the session key, and Bob's identity, and encrypts these in the key he shares with Alice. He sends both encrypted messages to Alice**.** EA(T,L,K,B),EB(T,L,K,A)

2.3.4 Alice generates a message with her identity and the timestamp, encrypts it in *K*, and sends it to Bob. Alice also sends Bob the message encrypted in Bob's key from Trent. EK(A,T),EB(T,L,K,A)

2.3.5 Bob creates a message consisting of the timestamp plus one, encrypts it in *K*, and sends it to Alice. EK (T + 1).

2.4 Denning Sacco Protocol

This protocol also uses public-key cryptography [6]. Trent keeps a database of everyone's public keys. Following steps explain working of protocol:

2.4.1 Alice sends a message to Trent with her identity and Bob's identity: A, B

2.4.2 Trent sends Alice Bob's public key, *K*B, signed with Trent's private key; *T*. Trent also sends Alice her own public key, *K*A, signed with his private key. ST (B, KB), ST (A, KA)

2.4.3 Alice sends Bob a random session key and a timestamp, signed in her private key and encrypted in Bob's public key, along with both signed public keys. EB(SA(A,B,K,TA)),ST(A,KA),ST(B,KB)

2.4.4 Bob decrypts Alice's message with his private key and then verifies Alice's signature with her public key. He checks to make sure that the timestamp is still valid.

3. COMPARISION CRITERIA

There are many criteria on which key exchange protocols can be compared. In this paper we have compared protocols on following criteria: 1) Vulnerability to attacks 2) Variants available 3) Usage of nonce 4) Mutual Authentication 5) User Anonymity 6) Security of session key. Following is the reason of choosing particular criteria.

3.1 Vulnerability

Vulnerability is one of the criteria on which session key exchange protocols are evaluated. This criterion helps in determining loop holes which can be exploited by attackers to gain control over communication session by obtaining session key. Usually vulnerability of a protocol is found out by using it in various situations and analyzing its response. Examples of vulnerabilities are reply attacks in Wide Mouth Frog Protocol [7] [8], Parallel Session Attacks in Neuman–Stubblebine[9][10] and DOS attacks in Karbaros [11] etc..Analysis of vulnerability helps protocol designers in determining exact behavior that causes protocol to fail to provide secure communication between two parties. This analysis also gives insights on how to prevent these exploits in future.

3.2 Variants

Once vulnerability is found in protocol and analyzed, protocol designers address this vulnerability by introducing new technique or feature or extra bit of information that prevents attacker from applying known exploits on this protocol and making protocol secure in turn. Well known variants of session key protocol includes, variants of Wide Mouth Frog described by [20], Neuman–Stubblebine variant described by [9][1] . These variants may not be proof from vulnerability and needs to be checked as original protocols to discover vulnerability if there are any.

3.3 Usage of nonce

A nonce is an identification of party involved in communication. Nonce can be a number or any random string. When Alice send a nonce encrypting it with public key of Trent and Trent replies with same nonce encrypted inside Alice’s private key it confirms to Alice that Trent is actually trusted party and there is no impersonator involved. This is important in establishing identity when two un-trusted parties are about to communicate and want to make sure that they are communicating with each other and there is no impersonator involved. Protocols such as Neuman–Stubblebine, Karbaros and Denning Sacco Protocol make use of this technique for mutually authenticating parties involved in communication.

3.4 Mutual Authentication

Mutual authentication is very important property when communicating on insecure network channels. Network such as Internet where impersonation is easily achieved mutual authentication technique is used to make sure both hosts identify each other before communication can take place. There are many techniques of mutually authenticating such as EAP-IKEv2 [13]’ pseudonym identity [14], trusted third parties, [15] and nonce. Protocols such as Needham-Schroeder and Yahalom provides mutual authentication.

3.5 User Anonymity:

User anonymity refers to be able to communicate without revealing one’s identity. Reveling identity on insecure networks such as Internet can be risky as it can be tracked on Internet. Sharing a session key without revealing their identity is an important property of session key exchange protocols. According to [16] anonymity is said to be achieved when an adversary who are not in possession of secret key cannot learn the identity of signer of signature of secret key. Shoup[17] defines anonymity in the context of the simulation framework for key exchange security, as opposed to the indistinguishability framework of, for example, Canetti-Krawczyk [18], which has now become more commonplace for analyzing key agreement protocols.

There are two forms of anonymity according to [19]. There are protocols such as [20], [21], [22], [23] which aim to provide identity hiding where identity of one party remains hidden. However identity becomes available by the end of protocol to peer. There are protocols that have been suggested by [23], [24] that overcome this problem.

There are also protocols such as [17], [25], [26], [27], [28], [29] aim to give anonymity where even peer of party does not learn its long term identity. This property is very important for practical applications such as TOR [30].

4. PROTOCOL COMPAIRISION

4.1 Wide Mouth Protocol

This protocol was proposed by [1].The wide mouth frog protocol uses a trusted third party to exchange key between two parties involved. The third party Trent is trusted by both Alice and Bob and stores secret keys of both. Because of involvement of trusted third party in the communication no party has to send any key in plain text. A message including key is encrypted from the secret key of either Alice or Bob.

There are number of small problem with this protocol. There is a global clock or synchronization between all three parties is required for this protocol to function. If Trent trusted third party is compromised then this protocol cannot guarantee confidentiality of information. Also, this protocol is stateful. More functionality is required from server such as in situation where Bob is not available.

One of the vulnerability of this protocol is assumption that Alice is competent enough to generate good session keys [31]. So, if Alice/Bob can use method which can generate a secure session key using random number then this protocol is excellent choice to be used in insecure networks. This might not be case always as random numbers are not very easy to generate.

This protocol is vulnerable to reply attacks which are fairly easy to construct as demonstrated by [7][8]. In [7] Mallory grabs a packet from Bob to Trent and sends it again EB(T,A,K), or it can grab packet from Alice EA (T,B,K). If this packet is inside appropriate time window Mallory can make Trent update time stamp of key K. This way he can extend life time of K as he wants where as Alice and Bob will assume that key K has been expired and has been destroyed by Trent.

In another attack can happen on this protocol if the duplicate packet reaches Bob from Trent described by [8], In this case Bob believe that it has two different connections with Alice but Alice believes that there is only one active connection. This attack however actually fails against complete specification of protocol.

There is a variant of this protocol which overcomes vulnerability of reply attacks by [8]. This variation uses nonce for mutual authentication between Alice and Bob. Here Bob can authenticate Alice by sending message (RB)K. Alice can reply to this message by ((RB)RA)K

Timestamps are used in this protocol to check freshness of message.

This protocol does not make use of nonce to mutually authenticate parties. This is just a key exchange protocol, authentication of the Alice or Bob is out of scope of this protocol. In this case since mutual authentication is not possible, identity theft is possible.

User anonymity is possible as none of the parties give their identity at the time of exchanging messages.

4.2 Neuman-Stubbleine Protocol

According to [32] clocks can get out of synchronization due to number of reasons. When sender’s time is ahead then receiver’s time message can be intercepted by Mallory and can be transmitted later when timestamp current at receiver’s side. To address this problem of suppress-reply attack Neuman-Stubblebine protocol is used.

It was initially developed by [33]. In this protocol synchronized clocks are not required as the timestamps carries time of only Bob’s clock that he generated himself.

Another advantage of this protocol is Alice and Bob do not need Trent to verify their identity to each other using Trent in case they had communicated within some predetermined time limit.

However, this protocol is vulnerable to parallel session attack using one principle as oracle who will generate key as demonstrated in [10][9].

In this protocol there is no role for Trent in providing mutual authentication. Alice only uses message sent to her by Trent to initiate process of mutual authentication by sending message (A,K,TB)KB, R'A to Bob. Bob replies to this message with new nonce and encrypting Alice's nonce with session key sent by Alice in following message R'B, (R'A)K. Once this message reaches Alice and it verifies nonce it originally sent to Bob it authenticates Bob's identity. Alice then can let Bob authenticate herself by sending message (R'B)K which when decrypted by Bob authenticates Alice's identity.

There are number of simple attacks documented on this protocol. Two such attacks are documented in [9].

In first attack authors uses simplified version of Yahalom for the first 4 messages where B accepts the nonce RA has fresh shared key K. In this attack Mallory poses as Alice and communicates with Bob. It first sends message to identify itself as Alice with message A,RA. In next step Bob sends following message to Trent B, (A, RA, TB)KB, RB. Trent generated session key and sends it in separate messages to both Bob and Alice. Here, real Alice ignores the message since I has not previous communication record with Trent for communication on this session. Next, Mallory posing as Alice messages to Bob {A, RA, TB}K, {RB}KA. Mallory also sends second message to Bob R'A, (A, RA, TB)KB. This message is for mutual authentication and in reply to this message Bob sends message R'B, (R'A)RA and thus authenticate himself to Mallory posing as Alice. Mallory then successfully authenticate himself as Alice by sending message (R'B)RA. There is variant described in [9] as well as correction by authors in [4] which solves this problem.

The second attack concerns with repeated authentication part assuming that K has been recorded in previous legitimate run of protocol. This attack on protocol starts form step 5 of the original protocol. In this scenario Mallory is posing as Alice for Bob. Mallory sends message R'A, (A, K, TB) KB to Bob as part of authentication sequence. Bob replies to this message with R'B, (R'A)K. Now as part of repeatedly authenticating Bob, Mallory send message R'B, (A, K, TB)KB. Bob replies to this message with R''B, (R'B)K. To this message Mallory replies (R'B)K. Thus, repeatedly authenticating Bob as long as it would like to and possibly creating DOS type of attack.

There is one more documented attack by [34] where Mallory can get as many ciphers (A, K, TB)KB as he likes in order to get KB. This attack begins from step 2 of protocol. Here Mallory is poses as Bob and send this message to Trent, B, (A, K0, TB)KB, RB. Trent replies to this message with (B, RB, K1, TB)KA, (A, K1, TB)KB, RB. Mallory replies to this message by B, (A, K1, TB) KB, RB again requesting new key K2 from Trent. This cycle goes on infinitely as long as Mallory would like to continue.

Timestamps and nonce are used for maintaining freshness of keys and authentication of both principles. This protocol also belongs to symmetric key protocol family.

4.3 Karbaros Authentication System

Karbaros is a variant of Needdham-Schroeder protocol implemented by MIT for its Athena project. In this system Alice and Bob shares their private keys with Trent. This protocol assumes that everyone's clock is synchronized [31].

This protocol has several key requirements in order to operate effectively. The protocol must guarantee secrecy of secret keys KA and KB. Trent and Alice or Bob must agree on value and duration of timestamp T's Validity. Alice and Bob must agree on values of timestamp T2. Ticket granting server G and Alice or Bob must agree on duration of timestamp T2.

It is possible to use DOS attack on karbaros system as described in [11]. There are many different versions of Karbaros systems available each version addresses problem reported in previous versions. Complete list is available on [5]. Karbaros uses timestamps to check freshness of message. It also uses nonces to confirm host's authenticity. With usage of timestamps and nonce it overcomes flaws of Denning-Sacco protocol.

4.4 Denning-Sacco Protocol

Denning Sacco protocol was proposed by [6]. It is modified version of Needham-Schroeder protocol. It addresses key freshness problem of Needham-Schroeder. The nonce used in Needham-Schroeder is replaced by timestamps. The message that bob receives is (K, A, T) KB. Here, Bob can check timestamp and verify timeliness message. Since Bob knows message is coming from Alice it can authenticate Alice to Bob with message (B, K, T, (K, A, T)) KA. Since this message is encrypted with secret key of Alice which only Trent and Alice knows, Alice can trust this message as un altered.

However clock drift and network delays needs to be considered when taking into account timestamps [31]. As it's an improvement over Needham-Schroeder protocol it belongs to same family of symmetric key protocols.

This protocol is subject to multiplicity attack described in [8]. In this attack after step (III) where Alice sends Bob the original packet (K, A, T) KB, this packet is intercepted by Mallory which keeps sending this packet repeatedly. Bob has no way to know if this packet is being repeated. Here Mallory can pose as Alice and Bob will think he is communicating with Alice but actually he is communicating with Bob.

This attack can be overcome by use of nonce handshake. Here, after step (III) Bob sends a message to Alice with a nonce (RB)K . To this message Alice decrypts the message and again encrypts the decrypted message with shared key K and sends back to Bob.

4. CONCLUSION

We described various protocols used for key exchange. We also compared on various parameters important for key exchange protocol.

We found that one of the most difficult problems in implementation of key exchange protocol is freshness of message. i.e. clocks may get out of synchronization or there is no timestamp to check when message was sent. It should be made sure that message always contains sufficient information for checking freshness of message bound together in protocol execution.

In nonce based protocols, there is nothing that can help the first time receiver determine its freshness unless something is provided earlier. Attaching a timestamp with the original message can solve this problem. If timestamps are used then also it will be difficult for recipient to determine if the message is fresh and authenticate in case if the two messages containing same type of information are received in a time frame.

We can conclude that this property is difficult to achieve. Selecting a protocol that fits best for a mobile computer to exchange key is a trade off. The selection is dependent on the network in which the mobile computer will be operated. If the computer is to be operated on in largely insecure public network such as Wi-FI hot spots then it is necessary to have mutual authentication feature. However if the computer is to be used for communication over internet via mobile service operator it is suggested to make use of protocol that provides user anonymity as the communication between the mobile service provider is between two trusted parties thus removing overhead and making communication faster

Parameters/Protocols	**Used on Insecure N/W**	**Vulnurability**	**Varients**	**Mutual Authentication**	**User Anonymity**	**Forward Secrecy**
Wide Mouth Frog	Yes	Reply Attack	Yes	No	Yes	Yes
Neuman-Stubbleine Protocol	Yes	Reply Attack	Yes	Yes	No	No
Karbaros Authentication System	Yes	N/A	Yes	Yes	No	Yes
Denning-Sacco Protocol	Yes	Multiplicity Attack	Yes	Yes	No	No

Table 2: Protocol Comparison

REFERENCES

[1] M. Burrows, M. Abadi, and R. Needham, "A Logic of Authentication," *Research Report 39, Digital Equipment Corp. Systems Research Center*, Feb 1989.

[2] M. Burrows, M. Abadi, and R. Needham, "A Logic of Authentication," *ACM Transactions on Computer Systems, v. 8, n. 1*, Feb 1990, pp. 18-36.

[3] A. Kehne, J. Schonwalder, and H. Langendorfer, "A Nonce-Based Protocol for Multiple Authentications," *Operating Systems Review, v. 26, n. 4*, Oct 1992, pp. 84-89.

[4] B.C. Neuman and S. Stubblebine, "A Note on the Use of Timestamps as Nonces," *Operating Systems Review, v. 27, n. 2*, Apr 1993, pp. 10-14.

[5] Kerberos Authentication and Authorization System official documentation, Oct 1988 by the Massachusetts Institute of Technology.

[6] D.E. Denning and G.M. Sacco, "Timestamps in Key Distribution Protocols," *Communications of the ACM, v. 24, n. 8*, Aug 1981, pp. 533-536.

[7] R. Anderson and R. Needham. *Programming Satan's computer*,)Cambridge University Computer Laboratory
Pembroke Street, Cambridge, England CB2 3QG 1995).

[8] Gavin Lowe. A family of attacks upon authentication protocols. Technical Report 1997/5, Department of Mathematics and Computer Science, University of Leicester, 1997.

[9] Tzonelih Hwang, Narn-Yoh Lee, Chuang-Ming Li, Ming-Yung Ko,and Yung-Hsiang Chen. Two Attacks on Neuman- Stubblebine Authentication Protocols. *Information Processing Letters, 3*:103–107, 1995.

[10] Ulf Carlsen. Cryptographic Protocol Flaws. In Proceedings *7th IEEE Computer Security Foundations Workshop*, pages 192–200. IEEE Computer Society, 1994.

[11] web.mit.edu/kerberos/advisories/MITKRB5-SA-2011- 002.txt

[12] B.C. Neuman and S. Stubblebine, "A Note on the Use of Timestamps as Nonces," *Operating Systems Review, v. 27, n. 2*, Apr 1993, pp. 10-14.

[13] RFC-5106: The Extensible Authentication Protocol-Internet Key Exchange Protocol version 2 (EAP-IKEv2) Method.

[14] Yixin Jiang, Chuang Lin, Xuemin (Sherman) Shen, Mutual Authentication and Key Exchange Protocols for Roaming Services in Wireless Mobile Networks, *IEEE Transactions on Wireless Communications*, Vol.:5, 9 pp: 2569 – 2577, September-2006.

[15] Markus Jakobsson and David Pointcheval Mutual Authentication for Low-Power Mobile DevicesP. Syverson (Ed.*): FC 2001, LNCS 2339, pp. 178–195, 2002.Springer-Verlag* Berlin Heidelberg 2002.

[16] Jesse walker, Jiangtao Li, Key Exchange with Anonymous Authentication using DAA-SIGMA Protocol, *IACR 2010/454*.

[17] Victor Shoup, On Formal Model for Secure key exchange, *IBM Research Report RZ3120*, 1999.

[18] Ran Canetti and Hugo Krawczyk. Security analysis of IKE's signature based key-exchange protocol. In Moti Yung, editor, *Advances in Cryptology Proc*. CRYPTO 2002

[19] Ian Goldberg, Douglas Stebila, and Berkant Ustaoglu, Anonymity and one-way authentication in key exchange protocols *Technical Report CACR* 2011-11, University of Waterloo Centre for Applied Cryptographic Research, 2011

[20] Ran Canetti, Hugo Krawczyk, Analysi fo Key-Exchange Protocols and their use for building secure channels, *Advances in Cryptology- EUROCRYPT'01, Vol2045 LNCS*, PP. 453-474. Springer 2001

[21] Zhaohui Cheng , Liqun Chen, Richard Comley , Qiang Tang, Identity-based key agreement with unilateral identity privacy using pairings, Proceeding *ISPEC'06 Proceedings of the Second international conference on Information Security Practice and Experience* Pages 202-213, 2006.

[22] Hung-Yu Chien. ID-based key agreement with anonymity for ad hoc networks. In Tei-Wei Huo, Edwin Sha, Minyi Guo, Laurence Yang, and Zili Shao, editors, Proc. *Embedded and Ubiquitous Computing (EUC)* 2007, LNCS, volume 4808, pp. 333{345. Springer, 2007.

[23] Ran Canetti and Hugo Krawczyk. Security analysis of IKE's signature based key-exchange protocol. In Moti Yung, editor, *Advances in Cryptology Proc*. CRYPTO 2002,

[24] Alfred J. Menezes and Berkant Ustaoglu. Comparing the pre- and post-specified peer models for key agreement. *International Journal of Applied Cryptography, 1(3):*236-250, 2009.

[25] Ian Goldberg. On the security of the Tor authentication protocol. In George Danezis and Philippe Golle, editors, *Privacy Enhancing Technologies (PET) 2006, LNCS, 4258,* pp. 316-331. Springer, 2006.

[26] Lasse verlier and Paul Syverson. Improving efficiency and simplicity of tor circuit establishment and hidden services. In *Privacy Enhancing Technologies, LNCS, volume 4776*, pp. 134{152. Springer, 2007.

[27] Aniket Kate, Greg M. Zaverucha, and Ian Goldberg. Pairing-based onion routing with improved forward secrecy. *ACM Transactions on Information and System Security, 13(4)*:29, 2010.

[28] Sk. Md. Mizanur Rahman, Atsuo Inomata, Takeshi Okamoto, Masahiro Mambo, and Eiji Okamoto. Anonymous secure communication in wireless mobile ad-hoc networks. In Frank Stajano, Hyoung Joong Kim, Jong-Suk Chae, and Seong-Dong Kim, editors, Proc. *International Converence on Ubiquitous Convergence Technology (ICUCT)* 2006, LNCS, volume 4412, pp. 140{149. Springer, 2007.

[29] Sherman S. M. Chow and Kim-Kwang Raymond Choo. Strongly-secure identity-based key agreement and anonymous extension. In Juan Garay, Arjen Lenstra, Masahiro Mambo, and Ren e Peralta, editors, Proc. *10th International Conference on Information Security Conference (ISC)* 2007, LNCS, volume 4779, pp. 203-220. Springer, 2007.

[30] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In Proc. *13th USENIX Security Symposium*. The USENIX Association, 2004.

[31] Bruce Schneier, *Applied cryptography* 2nd Edition, Willy Publishing Company 1998.

[32] L. Gong, "A Security Risk of Depending on Synchronized Clocks,*" Operating Systems Review, v. 26, n. 1*, Jan 1992, pp. 49-53.

[33] A. Kehne, J. Schonwalder, and H. Langendorfer, "A Nonce-Based Protocol for Multiple Authentications," *Operating Systems Review, v. 26, n. 4*, Oct 1992, pp. 84-89.

[34] ChristophWeidenbach. Towards an automatic analysis of security protocols. In Harald Ganzinger, editor, Proceedings of the *16th International Conference on Automated Deduction, volume 1632* of LNAI, pages 378–382. Springer, 1999