IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol. 8, Issue.11, November 2020 E-ISSN: 2347-2693

An Improved Key Distribution Protocol Using Symmetric Key Cryptography

Yasser Ali Alahmadi[1*], Saleh Noman Alassali[2]

[1,2] Dept. Computer Science, Sheba Region University, Marib, Yemen

**Corresponding Author: yasser_ali8891@yahoo.com, Tel.: +967-713333731*

DOI: https://doi.org/10.26438/ijcse/v8i11.2126 | Available online at: www.ijcseonline.org

Received: 12/Nov/2020, Accepted: 19/Nov/2020, Published: 30/Nov/2020

***Abstract*—** Due to the rapid growth in technology, the development and use of cryptosystems has become plays an important role in networked and distributed applications. To get the benefits of such applications, the principals will cooperate by exchanging information over an open networks such as the internet. A key distribution protocol is an essential component of any cryptosystem to generation and sharing of cryptographic keys between the principals involved in the network securely. In the current days, there are a number of key distribution protocols that have been developed and implemented. However, the most of such protocols were found to be prone to several attacks a long time after deployment. In this paper, the key distribution protocol is designed to improve the Nomaskd protocol. The two protocols are analyzed and verified by a formal verification tool called Scyther, the verification results show that the Nomaskd protocol does not fulfill the strong authentication goals, whereas the improved protocol fulfill these goals.

***Keywords*—**Key distribution, Formal verification, Symmetric key cryptography, Nomaskd protocol, Scyther tool.

I. INTRODUCTION

Over the eras, information has gained great interest, especially nowadays, and due to the need to transfer information from one location to another, there must have been ways to protect it during its transmission. Perhaps the issue of information security and exchange across an open networks is one of the issues that concern not only researchers and specialists, but also international organizations and the world related to them.

There is no doubt, the increased depending upon information and its networks increases the impact of the risks that can be faced, therefore it was necessary to seek to face these risks. Cryptography has an important role for protect the information during its transmission over an open networks. The cryptography is broadly divided into two types[1],[2]: symmetric key cryptography and asymmetric key cryptography. In the former, the same key is used for both encryption and decryption processes whereas in the later, two keys are used, one of them is used for encryption and another for decryption. The encryption key plays a very important role in symmetric key cryptography since its security directly depends on secrecy of this key. In this technique, before enciphering both principals (The principals may be users, hosts or processes) previously should agree on a secret key for encryption/decryption. Compared to asymmetric key cryptography this technique is simple, fast implementations and good for encrypting large amounts of data, but key distribution is a major problem of this technique[3],[4]. This problem is solved using key exchange protocols such as Diffie-Hellman[5] or asymmetric key encryption schemes, e.g., RSA or Elgamal[1].

The use of asymmetric encryption to exchange session keys may not be suitable for some applications, because it requires high computing capabilities[6]. Therefore, many of an authentication protocols have been emerged for generation and sharing of session key between two communicating principals using symmetric key cryptography such as [7], [8],[9],[10],[11].

This paper proposed a revised session key distribution protocol based on the Nomaskd protocol[11]. In this protocol, The Trusted Server is used for generation and sharing of session key between two communicating principals using symmetric key cryptography. The original protocol is first examined, clearly its weaknesses are specified and then some of the changes are suggested to avoid various weaknesses of the same protocol. In order to ensure the reliability of the improved protocol, Scyther tool[12] is implemented and used to analysis the two protocols and show the differentiation between them.

This paper is organized into eight sections, Introduction is in Section I, The notations used in describing protocols are in Section II, Review of related works is in Section III, Weaknesses of the Nomaskd protocol are in Section IV, The improved protocol is in Section V, Formal security analysis using Scyther is in Section VI, Discussion is in Section VII and Conclusion and Future work are in Section VIII.

II. NOTATION

The notations used in describing protocols throughout this paper are listed in Table 1.

Table 1. The Notations Used in Describing Protocols.

Notation	Description
A,B	Names/Identities of principals
S	The Trusted Server
[M]K	Encryption of message M with K
Kas, Kbs	Secret keys of principal A and B shared with S
Kab	Session key shared between A and B
Na, Nb	Nonce values chosen by principal A and B
I	An intruder

III. REVIEW OF RELATED WORKS

There are many of key distribution protocols that have been proposed by various researchers in the past. This section review a few typical protocols from literature. In order to provide the reader with a better understanding, we will review these protocols in a few detail. In such protocols, each principal has a secret key shared with The Trusted Server which in turn generates and distributes a secret session key between two communicating principals.

A. Needham-Schroeder Protocol

In 1978, R.M. Needham and M. Schroeder proposed a first protocol for key distribution and authentication[7]. This protocol uses symmetric key cryptography, The Trusted Server and based on use of nonce to verify freshness of the messages. This protocol involves 5 steps as follows.

1. A → S: A, B, Na
2. S → A: [Na, B, Kab, [Kab, A] Kbs]Kas
3. A → B: [Kab, A]Kbs
4. B → A: [Nb]Kab
5. A → B: [Nb-1]Kab

In this protocol, The principal A (Alice) sends The Trusted Server the unencrypted message 1, telling it she wants to communicate with the principal B (Bob). The Trusted Server generates a new session key Kab, construct the message 2 and send it to principal A. The principal A decrypts the message 2 utilizing the secret key Kas, retrieves the needed session key Kab and checks that received nonce Na is the same as in message 1. if so, she forwards the part of the message encrypted with B's secret key Kbs to the principal B. The principal B decrypts the message 3 utilizing the secret key Kbs, retrieves the needed session key Kab and sends the principal A nonce Nb encrypted under Kab (message 4) to show his knowledge of Kab. The principal A decrypts the message 4 and guarantee that this message is from B, then the principal A performs a simple operation on the nonce, re-encrypts it and sends it back verifying that she is still alive and that she holds the key.

In this protocol, the purpose of the first 3 steps are to complete the distribution of session key Kab whereas the last 2 steps are for the handshake to prevent a certain type of replay attack. Despite the handshake, the protocol is still vulnerable to a some attacks[6]. The attack on Needham-Schroeder protocol in case of compromised session key Kab is demonstrated by[13].

B. Otway-Rees Protocol

In 1987, D. Otway and O. Rees proposed an authentication and key distribution protocol designed to remove a replay attack of Needham-Schroeder protocol[8]. This protocol involves 4 steps as follows:

1. A → B : M,A, B, [Na, M, A, B]Kas
2. B → S : M,A, B, [Na, M, A, B]Kas, [Nb, M, A, B]Kbs
3. S → B : M,[Na, Kab]Kas, [Nb, Kab]Kbs
4. B → A : M,[Na, Kab]Kas

In this protocol, The principal A sends the principal B the encrypted message 1, included the session identifier M, the identities of A and B and A's nonce Na using the key Kas along with the session identifier M and the identities of A and B as a plaintext. The principal B receives the message 1 and creates his own similar message encrypted with his secret key Kbs, then he sends his message along with A's message to S. S receives the message 2, search for the secret keys Kas and Kbs, decrypts these two encrypted message pieces, generates a new session key Kab, construct the message 3 and sends it to the principal B. The principal B receives message 3, removes the last encrypted part with his secret key, decrypts this sub-message with his secret key Kbs, retrieves the session key Kab and checks that received nonce Nb is the same as in message 2. if so, he sends the remaining part of the message to A. In this way, A is also able to retrieves the session key Kab, she decrypts the message 4 using her secret key Kas, and the two principals are able to start communicating.
In this protocol two attacks are founded by [14].

C. Nomaskd Protocol

In 2020, Shalini and M. Kushwaha presented protocol derived of the Needham-Schroeder and Otway-Rees protocol[11]. In this protocol, the session key is generated by The Trusted Server that distributes it between two communicating principals using symmetric key cryptography. This protocol involves 5 steps as follows:

1. A → B: A, B, [Na, A, B]Kas
2. B → S: A, B, [Na, A, B]Kas, [Nb, A, B]Kbs
3. S → B: [Na,Nb, Kab]Kas, [Nb, Na, Kab]Kbs
4. B → A: [Na, Nb,Kab]Kas, [Na]Kab
5. A → B: [Nb]Kab

In this protocol, The principal A sends the principal B an encrypted message included the identities of A and B and A's nonce Na using the key Kas along with the identities of A and B as a plaintext. The principal B receives the message 1, he generates his own similar message which include B's nonce Nb, A's identity and B's identity,

integrate his own message with A's message and forward it to S. S receives the message 2, search for the secret keys Kas and Kbs, decrypts these two encrypted message pieces utilizing Kas and Kbs and equate with A and B. On confirmation, it generates a new session key Kab, construct the message 3 and sends it to B. The principal B decrypts a part of message encrypted utilizing his secret key Kbs to retrieves the needed session key Kab and B's nonce and checks nonce Nb to validate that it is for the current session. If so, he encrypts Na utilizing Kab, integrate it with the sub-message intended to principal A and transfer it to principal A. The principal A decrypts the first part of message using Kas to retrieves the needed session key Kab, A's nonce and B's nonce and then principal A uses the session key Kab to decrypts the other part to retrieves Na. Thus, the principal A gets confident that message is dispatched by B and key is for the current session. In order to substantiate her distinctiveness to B, she encrypts Nb via session key Kab, and transfer it to B. Once principal B collects this message and decrypts it using Kab he gets assured that the session key has soundly arrived at principal A.

IV. WEAKNESSES ON NOMASKD PROTOCOL

In this section, we will highlight the weaknesses of Nomaskd protocol and describe how an intruder can exploit these weaknesses to interact with this protocol. We assume that the intruder I is one of the legitimate users of the system, he can interrupt any message in the system and generate new messages encrypted with his own secret key Kis. He can also replay complete encrypted messages or replace part of a message and replay it. The following describes the three weaknesses of Nomaskd protocol:

1. The principal A sends the principal B an encrypted message included the identities of A and B and A's nonce Na using the key Kas along with the identities of A and B as a plaintext.

1. A → B: A, B, [Na, A, B]Kas

That means 2/3 of the plaintext and its ciphertext occur together in the same packet, which in turn compromise the secret (long-term) key Kas in the future[6].

2. In step 2, The principal B sends The Trusted Server the following message:

2. B → S: A, B, [Na, A, B]Kas, [Nb, A, B]Kbs

The Trusted Server receives this message, it uses identities of A and B to search for the secret keys Kas and Kbs, then it decrypts these two encrypted message pieces utilizing Kas and Kbs. Afterwards, it should checks the contents of the message to make sure of its correctness content. Thus, there are two actions, The Trusted Server can do one of them:

A1: S checks that the values in the two decrypted pieces match.

A2: S checks that the values in the plaintext match the values in the two decrypted pieces.

If S performs the checking as in **A1**, the intruder can obtains a session key shared with the principal A and deceives the principal A into establishing a false session with principal B. The intruder plays the role of principal B as shown in Figure 1 and the traces of attack on this protocol are as follows:

1: A → I(B) : A, B, [Na, A, B]Kas
2: I(B) → S : A, I, [Na, A, B]Kas, [Ni, A, B]Kis
3: S → I(B) : [Na, Ni, Kai]Kas, [Ni, Na, Kai]Kis
4: I(B) → A : [Na, Ni, Kai]Kas, [Na]Kai
5: A→I(B): [Ni]Kai

In this attack, The intruder I interrupts the message 1, then he generates his own nonce Ni, creates his own message encrypted with his secret key Kis, integrate it with A's message and sends it to S impersonating as B. This message will be correct from the S's point of view, it will generates a new session key Kai and encrypts it with the intruder's secret key Kis, thus the intruder will obtains a session key shared with principal A and deceives the principal A to believes that the key Kai is shared with principal B, whereas in fact it is shared with I.

If S performs the checking as in **A2**, an attack above also may be performed. In this case, the intruder must be able to replace B's identity with his own identity in the ciphertext ([Na, A, B]Kas) of the message 2. In the fact, the intruder can do that, due to, in many situations, there are similarity between the identities of users related to the same Local Area Network LAN. The words user-1, user-2, user-3,…, user-n or IP numbers such as 192.168.1.2, 192.168.1.3, ... etc. Sometimes are used to represent the identities of the users. Such identities make impersonation easier by exploiting some design flaws and type of encryption used in the system. In such identities, the difference between the identity of any user and another is only one number, and because the intruder I is one of the legitimate users of the system, therefore the identity of the intruder will be close to the identity of any user of the system. This makes the intruder able to replaces an identity of any user with his own identity by only flipping some of the values of the final bits in the ciphertext.

3. In step 3, when the message 3 is created by S, the authentication semantic of the message need to clearly understand by the principals whom S intends to make the key know, but the information provided is insufficient to prevent impersonation attacks.

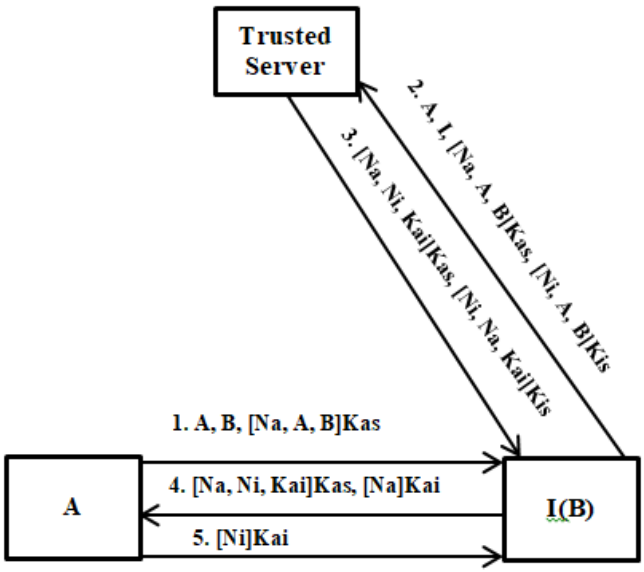

Figure 1: An Attack on Nomaskd Protocol

V. THE IMPROVED PROTOCOL

This section presents the improved protocol to remedy the security weaknesses of the Nomaskd protocol. The improved protocol involves 5 steps as shown in figure 2.

V.I Protocol description

1. A → B: A, [Na, B]Kas
2. B → S: A, [Na, B]Kas, [Nb, A]Kbs
3. S → B: [B, Kab, Na]Kas, [A, Kab, Na, Nb]Kbs
4. B → A: [B, Kab, Na]Kas, [Nb,Na]Kab
5. A → B: [Nb]Kab

In step 1: The principal A sends the principal B an encrypted message included the identity of B and A's nonce Na using the key Kas along with the identity of A as a plaintext.

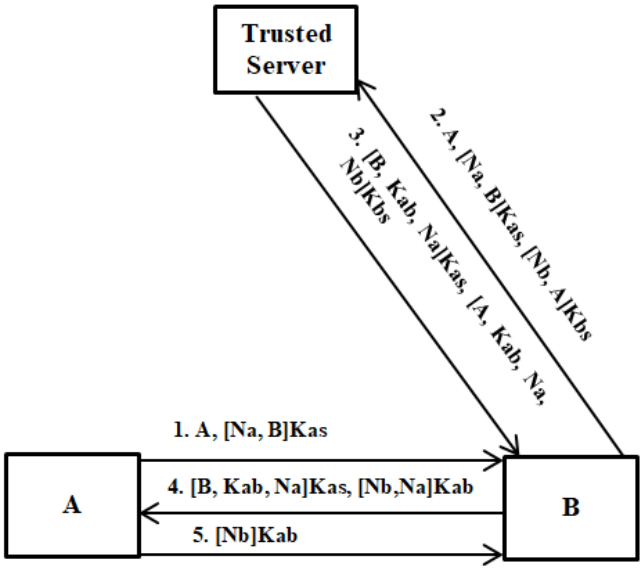

Figure 2: The Description of Improved Protocol

In step 2: B receives the message 1 and generates a similar message encrypted with his secret key Kbs, integrate it with A's message and forward it to S. **In step 3**: S receives the message 2, use A's identity to search for the secret key Kas, then it decrypts the first encrypted message piece utilizing Kas to obtains B's identity. Afterward it use B's identity to search for key Kbs, then it decrypts another encrypted message piece utilizing Kbs. Finally it generates a new session key Kab, constructs the message 3 and sends it to B. **In step 4**: B decrypts a part of message encrypted utilizing his secret key Kbs to retrieves the needed session key Kab, B's nonce and A's identity. Then he checks Nb and identity flag to validates if they are as in the message 2. On confirmation, he obtains confident that the session key Kab is for the current session and shared with principal A. Afterward he encrypts Nb and Na utilizing Kab, integrate this with the sub-message intended to principal A and transfer it to principal A. **In step 5**: The principal A decrypts the first part of message using Kas to retrieves the needed session key Kab, then she uses the session key Kab to decrypts the other part to retrieves Na and Nb. Then she checks Na and identity flag to validates if they are as in the message 1. On confirmation, she obtains confident that principal B has the correct session key for the current session. In order to substantiate her distinctiveness to B, she encrypts Nb via a session key Kab, and transfer it to B. Once principal B collects this message and decrypts it using Kab he gets assured that the session key has correctly arrived at principal A and the principal A still is alive.

V.II Features of The Improved Protocol

Some features of the improved protocol are as follows:

1. In steps 1 and 2, encrypting and hiding of identity of principal B make it difficult to carry out impersonation attacks and also limits the amount of information available to the cryptanalyst.

   1. A → B: A, [Na, B]Kas
   2. B → S: A, [Na, B]Kas, [Nb, A]Kbs

   Furthermore, when the message 2 is received, The Trusted Server does not need to check the identities match, and consequently the computations on the server is reduced.

2. In step 3, the identities of A and B are added, hence the principals A and B would ascertains the identities of each other.

   3. S → B: [B, Kab, Na]Kas, [A, Kab, Na, Nb]Kbs

3. Steps 4 and 5 guarantee necessary messages exchange between A and B, which makes the session key shared by both sides known explicitly and also prevent replay attackers.

   4. B → A: [B, Kab, Na]Kas, [Nb,Na]Kab
   5. A → B: [Nb]Kab

VI. FORMAL SECURITY ANALYSIS USING SCYTHER

This section provides the formal security verification of the original and the improved protocol using the formal security verification tool called Scyther developed by Cremers[12]. Scyther is based on the development algorithm that provides the representation of traces, analyses the security protocols automatically and verifies the entire possible behaviours of a protocol against most of the potential attacks. The adversary model used by Scyther is predefined and based on the Dolev-Yao model[15], It assumes perfect cryptographic conditions with unbreakable encryption, meaning that an intruder learns nothing from an encrypted message without his knowledge of the right key. The language used to writing security protocols in Scyther is Security Protocol Description Language (SPDL) [12],[16]. Scyther takes as input a role-based description of a protocol in which the intended security goals are specified using claim events [17].

The two protocols are implemented in Scyther and verified with security claims: Alive, Secret, SKR, Weakagree, Niagree and Nisynch. The aim of using Secret and SKR claims are to ensuring secrecy, whereas Alive, Weakagree, Niagree and Nisynch claims are to ensuring authentication, the more details about these claims are given in [12]. Table 2 and Table 3 shows the verification results of Nomaskd and improved protocol respectively.

Table 2. The Verfication Result of Nomaskd Protocol

Scyther results : verify

Claim				Status		Comments	Patterns
NoMask	A	NoMask,A1	Secret Na	Ok		No attacks within bounds.	
		NoMask,A2	SKR Kab	Ok		No attacks within bounds.	
		NoMask,A3	Secret Nb	Ok		No attacks within bounds.	
		NoMask,A4	Alive	Ok		No attacks within bounds.	
		NoMask,A5	Weakagree	Ok		No attacks within bounds.	
		NoMask,A6	Niagree	Fail	Falsified	At least 1 attack.	1 attack
		NoMask,A7	Nisynch	Fail	Falsified	At least 1 attack.	1 attack
	B	NoMask,B1	Secret Nb	Ok		No attacks within bounds.	
		NoMask,B2	Secret Na	Ok		No attacks within bounds.	
		NoMask,B3	SKR Kab	Ok		No attacks within bounds.	
		NoMask,B4	Alive	Ok		No attacks within bounds.	
		NoMask,B5	Weakagree	Ok		No attacks within bounds.	
		NoMask,B6	Niagree	Fail	Falsified	At least 1 attack.	1 attack
		NoMask,B7	Nisynch	Fail	Falsified	At least 1 attack.	1 attack
	S	NoMask,S1	SKR Kab	Ok		No attacks within bounds.	
		NoMask,S2	Secret Nb	Ok		No attacks within bounds.	
		NoMask,S3	Secret Na	Ok		No attacks within bounds.	

Done.

Table 3. The Verfication Result of Improved Protocol

Scyther results : verify

Claim				Status		Comments
Improved	A	Improved,A1	Secret Na	Ok		No attacks within bounds.
		Improved,A2	SKR Kab	Ok		No attacks within bounds.
		Improved,A3	Secret Nb	Ok		No attacks within bounds.
		Improved,A4	Alive	Ok	Verified	No attacks.
		Improved,A5	Weakagree	Ok	Verified	No attacks.
		Improved,A6	Niagree	Ok		No attacks within bounds.
		Improved,A7	Nisynch	Ok		No attacks within bounds.
	B	Improved,B1	Secret Nb	Ok		No attacks within bounds.
		Improved,B2	Secret Na	Ok		No attacks within bounds.
		Improved,B3	SKR Kab	Ok		No attacks within bounds.
		Improved,B4	Alive	Ok	Verified	No attacks.
		Improved,B5	Weakagree	Ok	Verified	No attacks.
		Improved,B6	Niagree	Ok		No attacks within bounds.
		Improved,B7	Nisynch	Ok		No attacks within bounds.
	S	Improved,S1	SKR Kab	Ok		No attacks within bounds.
		Improved,S2	Secret Nb	Ok		No attacks within bounds.
		Improved,S3	Secret Na	Ok		No attacks within bounds.

Done.

VII. DISSCOUTION

From the verification results shown in Table 2 and 3 above we can observe that the Nomaskd protocol does not fulfil the strong authentication claims: Niagree and Nisynch, hence many potential attacks can be performed. Whereas the improved protocol fulfill all security claims within bounds which the Scyther verified. This means the improved protocol compeletly fulfill the strong authentication goals and hence no potential attacks can be performed.

VIII. CONCLUSION AND FUTURE WORK

Authentication is a major property of any security protocol that should be held truly. To ensure that the protocol hold such property, the security protocols should be analysed using formal analysis tools such as Scyther. In this paper, the key distribution protocol is proposed to improve the authentication goals of the Nomaskd protocol. The Nomaskd protocol is first examined and analysed by the Scyther tool, clearly its weaknesses are specified as shown in Table 2 lines 6,7,13 and 14. Afterward the improvements are introduced in an improved protocol to avoid these weaknesses. The improved protocol is analysed by the Scyther tool. The analysis results indicate that the improved protocol satisfies the authentication goals to some extent as shown in Table 3.

In the future the researchers will use the improved protocol to suggest key management security system.

REFERENCES

[1] A.J. Menezes, P.C. Van Oorschot and S.A. Vanstone, "Handbook of Applied Cryptography", 5th Edition, CRC Press, Inc, **United States**, **2001**.

[2] A. Aasarmya and S. Agarwal, "*Improving Security for Data Migration in Cloud Computing using Randomized Encryption Technique*", International Journal of Computer Sciences and Engineering,Vol.**7**, Issue.**8**, pp.**39-43**, **2019**.

[3] S. Verma, R. Choubey and R. Soni, "*An Efficient Developed New Symmetric Key Cryptography Algorithm for Information Security*", International Journal of Emerging Technology and Advanced Engineering, Vol.**2**, No.**7**, pp.**18-21**, **2012**.

[4] N. Srilatha, M. Deepthi and I.R. Reddy, "*Robust Quantum Key Distribution Based on Two Level QDNA Technique to Generate Encrypted Key*", International Journal of Computer Sciences and Engineering,Vol.**5**, Issue.**2**, pp.**15-19**, **2017**.

[5] W. Diffie and M. Hellman, "*New directions in cryptography*", IEEE Transaction on Information Theory, Vol.**22**, Issue.**6**, pp.**644-654**, **1976**.

[6] W. Stallings, "Cryptography and Network Security", principles and practices, 7th Edition, Pearson Prentice Hall, **2017**.

[7] R.M. Needham and M.D. Schroeder, "*Using encryption for authentication in large networks of computers*", Communications of the ACM, Vol.**21**, Issue.**12**, pp.**993-999**, **1978**.

[8] D. Otway and O. Rees, "*Efficient and timely mutual authentication*", ACM SIGOPS Operating Systems Review, Vol.**21**, Issue.**1**, pp.**8-10**, **1987**.

[9] J. G. Steiner, B. C. Neuman, and J. I. Schiller, "*Kerberos: An Authentication Service for Open Network Systems*", USENIX Winter, pp.**191-202**, **1988**.

[10] M. Burrows, M. Abadi and R. Needham, "A Logic of Authentication", Proceedings of the Royal Society of London Mathematical, Physical and Engineering Sciences, Vol.**426**, No.**1871**, pp.**233-271**, **1989**.

[11] Shalini and M. Kushwaha, "*Mutual Authentication and Secure Key Distribution in Distributed Computing Environment*", International Journal of Advanced Research in Engineering and Technology (IJARET), Vol.**11**, Issue.**5**, pp.**378-390**, **2020**.

[12] C. Cremers and S. Mauw, "Operational semantics and verification of security protocols", Springer Science & Business Media, **2012**.

[13] D.E. Denning and G.M. Sacco,"*Timestamps in key distribution protocols*", Communications of the ACM, Vol.**24**, No.**8**, pp.**533–536**, **1981**.

[14] K. Liu, J. Ye and Y. Wang, "The Security Analysis on Otway-Rees Protocol Based on BAN Logic", IEEE 4th International Conference on Computational and Information Sciences (ICCIS), Chongqing, China, pp.**341-344**, **2012**.

[15] D. Dolev and A. Yao, "On the security of public key Protocols", IEEE Transactions on Information Theory, Vol. **29**, No.**12**, pp.**198-208**, **1983**.

[16] N. Dalal, J. Shah, K. Hisaria and D. Jinwala, "*A Comparative Analysis of Tools for Verification of Security Protocols*", Int. J. Communications, Network and System Sciences (IJCNS), Vol.**3**, Issue.**10**, pp.**779-787**, **2010**.

[17] N. Kahya, N. Ghoualmi and P. Lafourcade, "*Secure Key Management Protocol In Wimax*", International Journal of Network Security & Its Application, Vol.**4**, No.**6**, **2012**.

AUTHORS PROFILE

Yasser Ali Alahmadi, received a Bachelor of Computer Science from Sana'a University, Yemen. He is currently pursuing Master of Computer Science, Department of Computer Science of Sheba Region University, Marib, Yemen. His areas of interest are Information Security, C# Programming.

Mr. Saleh Noman Abdullah Alassali, he got Bachelor of computer engineering from KSU University, soudia arabia in 1988, he got master degree in Computer Science from Pune University, India in 2000, he got the Ph.D. in Infromation Securty, SRTMU, India, in 2005. In 2007 he became the head department of Computer Science Marib College, Sana'a University, Yemen. Currently he works as Associate Prof. in Computer Science department Sheba University, Marib, Yemen. He has published more than 8 reseach papers in reputed international journals. His main research work focuses on Cryptography algorithms and random number generators. WhatsAPP: 770317665, Yemen.

