A Modified Otway-Rees Protocol to Overcome Triple MAB Vulnerability

Pranav Vyas

***Abstract**: With growth of internet and wireless networks all around the world, information security has taken a central stage to protect commercial and well as personal data. Key exchange algorithms play an important role in information security. In this paper we study the Otway-Rees protocol and its vulnerabilities and propose a modified Otway-Rees protocol to overcome them. We then evaluate performance of the protocol from various points of view such as execution time, encryption speed and power consumption by comparing the original and proposed protocols. We find that although the original protocol is faster, the difference in speed is not significant and the modified protocol protects against vulnerability resulting in the best choice between the two protocols.*

***Keywords** : Applied Cryptography, Key Exchange Protocol, Otway-Rees Protocol, Triple MAB Vulnerability.*

I. INTRODUCTION

We live in an increasingly connected world today where secure communication is especially important. It is possible to achieve secure communication by applying concepts of encryption and decryption. There are mainly two techniques to achieve this: i) Symmetric key encryption and ii) Public key encryption [1].

In symmetric key encryption technique a single key is shared between two parties, the sender can encrypt the message with the shared secret key and the receiver can decrypt the message using the same shared secret key. In this scenario, the message is only secure if the shared key is secure and neither sender nor receiver is compromised by an attack. There is also another problem of coming up with a shared secret key. Techniques such as Diffie-Hellman [2], Andrew Secure RPC [3], Karbaros [4] and Schnorr's protocol [5] were developed to overcome the previously mentioned problem of securely exchanging of secret key.

In public key encryption system, a trusted third party is present which is trusted by both sender and receiver and which works as a mediator in initial exchange of secret key. In public key cryptography each party has two keys, one is public key that is known to all other parties and is freely shared. Another is a private key which is private to the owner and is never shared with anyone. These keys are usually mathematically related to each other in such a way that information encrypted by one key can be decrypted by using the other key. In this technique, the information is usually encrypted by using a public key of receiver and since it can only be decrypted by the private key of receiver and private key is with receiver only, no one but the receiver can decrypt the information. The public key encryption requires higher computational power and is slower than symmetric key encryption technique. Some examples of key exchange algorithms with help of a third party are: Needham-Schroeder protocol [6], Otway-Rees protocol [7], Denning-Sacco protocol [8] and Neuman-Stubblebine protocol [9].

In the real world scenario, a hybrid technique is most frequently used. Here the public key encryption scheme is used to exchange the symmetric key between sender and receiver and once the symmetric key is with both parties, that key is used for further encryption and decryption of the actual messages.

In this paper, we take a look at Otway-Rees protocol [7]. We also propose a solution to the vulnerability noted by Clark and Jacob [10] in their paper.

We also run an experiment to check the efficiency of our proposed modified protocol against the original protocol and compare the results.

This paper is divided into 5 sections. Section 1 is the introduction section. In section 2 we describe the original Otway-Rees protocol and the vulnerability noted by Clark and Jacob in their paper. In section 3 we propose a modified Otway-Rees protocol to overcome this vulnerability. Section 4 describes method used for evaluation of the efficiency of the proposed protocol against the original protocol and the reasoning behind it. This section also contains discussion on results of the experiment. In section 5 we present our conclusion.

Table 1 contains various notations used to describe the original protocol, the attack and the modified protocol.

Table 1. Protocol Notations

Notation	Description
A, B, S, I	Principal
M, Na, Nb	Nonce
Kab	Shared secret key for Alice and Bob
Ksp	Public key of S
Kap	Public key of Alice
Kbp	Public key of Bob

Revised Manuscript Received on May 30, 2020.
***** Correspondence Author

Dr. Pranav Vyas*, Smt. Chandaben Mohanbhai Patel Institute of Computer Applications, Charotar University of Science and Technology, At & PO Changa, Ta. Petlad, Dist. Anand, Gujarat, India. Email: pranavvyas.mca@charusat.ac.in

II. ORIGINAL OTWAY-REES PROTOCOL AND TRIPLE MAB ATTACK

According to the authors of the Otway-Rees key exchange protocol [7], it was designed to overcome vulnerabilities in Needham-Schroeder protocol [6] and Denning-Sacco protocol [8]. The basic idea behind the protocol is for the suspicious party to present the challenge for the other party to solve. In this scenario, both parties are suspicious of each other and, therefore, both must generate a challenge for each other. Once both parties solve each other's challenge, they mutually authenticate each other eliminating the element of suspicion between them. The authors of Otway-Rees protocol further state that the challenge should not be re-used and to eliminate the possibility of reusing of the challenge they propose either storing the previously used challenges, generating challenges from monotonically increasing numbers or sufficiently large random number generation.

We now describe the original Otway-Rees protocol. The process is also described graphically in figure 1.

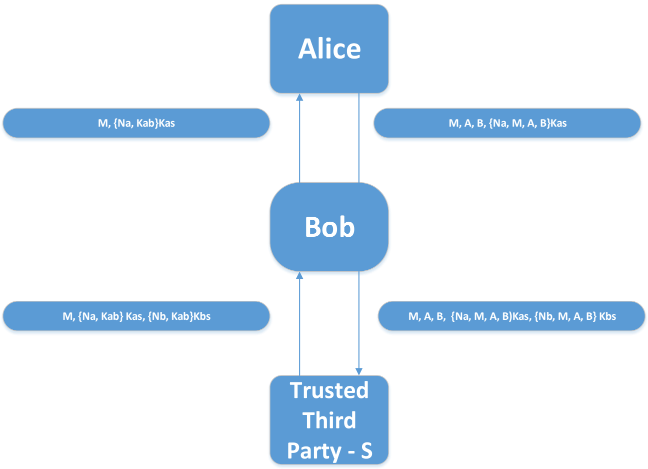
Alice
M, {Na, Kab}Kas
M, A, B, {Na, M, A, B}Kas
Bob
M, {Na, Kab} Kas, {Nb, Kab}Kbs
M, A, B, {Na, M, A, B)Kas, {Nb, M, A, B} Kbs
Trusted Third Party - S

Fig. 1.The original Otway-Rees Protocol

Following is the notation of original Otway-Rees protocol:

A→B: M, A, B, {Na, M, A, B}Kas – (1)

B→S: M, A, B, {Na, M, A, B)Kas, {Nb, M, A, B} Kbs – (2)

S→B: M, {Na, Kab} Kas, {Nb, Kab}Kbs – (3)

B→A: M, {Na, Kab}Kas – (4)

Here, nonce M is used to identify the current session. Nonce M denoting the current session eliminates the need for timestamp to maintain freshness of the message.

The values of Kas and Kbs are derived using symmetric key generation method. The value of Kas is known only to A and S. Similarly the value of Kbs is only known to B and S.

Kab is a shared secret key that is generated by a trusted third party S for communication between A and B. Kab is generated after S receives request from B as described in step 2 of the protocol. After receiving the request from B with necessary information, S compares M, A and B from both messages from A and B. The key is generated only if the MAB pair from A matches MAB pair of B. Na and Nb sent by A and B respectively works as a challenge for mutual authentication. Once the key is generated by S, two copies of the shared secret key are made. One is encrypted with Na for A using Kas which only A can decrypt. The other copy is encrypted along with Nb using Kbs, this can only be decrypted by B. Na and Nb along with the shared secret key guarantees the mutual authentication as these are the same nonce that were sent by A and B respectively as a challenge. The encryption of copies of the shared secret key with Kas and Kbs respectively guarantees that no one other than A and B can decrypt the keys as they are encrypted using public keys of A and B respectively. Both keys are sent from S to B in step 3. In step 4, B sends the part of the message it originally received from S to A that contains Na and Kab encrypted with Kas.

An attack on Otway-Rees protocol is described in their paper by Clark and Jacob [10]. This attack is known as triple MAB attack as the set of the three information MAB is used in making A believe that MAB is actually a key. Following is the description of the claimed attack by Clark and Jacob:

A→B: M, A, B, {Na, M, A, B}Kas – (1)

I (B)→A: M, {Na, M, A, B}Kas – (4)

As the information of M, A and B is initially sent over plaintext, it is public knowledge and available for anyone to use. It is assumed that the attacker already has knowledge of nonce Na.

III. OUR PROPOSED SOLUTION TO TRIPLE MAB VULNERABILITY OF OTWAY-REES PROTOCOL

The Otway-Rees protocol suffers from triple MAB vulnerability because the value of M, A, B are sent in plain text with no encryption of any type. This enables the adversary to capture this information and use it to exploit the fact that Alice relies on Bob for getting the shared secret key.

In our solution, we propose a modification of the original protocol that can patch the triple MAB vulnerability. We propose to construct 3 encrypted messages when Alice wants to send a message to Bob. The first encrypted message will contain values of M, A, B encrypted with public key of trusted third party S. The second message will contain copies of M, A, B and will be encrypted with public key of Bob. The third message will be for S which will be encrypted with public key of Alice. This message will contain values of M, A, B as well as the value of Na. Upon receiving the messages from Alice, Bob will first decrypt the message that was encrypted with Bob's public key. Bob will get value of M, A, B. Bob will store value of M for later reference. Now, Bob will construct another message that will contain Nb and values of M, A, B originally received from Alice in the message. Bob will then encrypt this message with its own public key. Bob will now send all three messages as a part of single unit of information to the trusted third party S.

After receiving messages from Bob, S will first decrypt the message encrypted by Alice with S's public key. This will give values of M, A, B to S. S will store these values as original values and will compare values with the values it gets from both of the two messages it received from Bob. Once S verifies the values of M, A, B from all the messages and finds it is matching with the original M, A, B values received from Alice, it will proceed to generate the shared secret key for Alice and Bob. Once the key is generated, it will be encrypted into two different copies, each copy will be encrypted with secret keys of both Alice and Bob. The value of session nonce M will be encrypted with Bob's public key. These will be sent to Bob. Upon receiving the message from S, Bob will decrypt the value of M and compare it with the original value it received from Alice, it will proceed further only if these two values are matching.

Bob will then decrypt the message encrypted with its public key and store the shared secret key in the memory. Bob will also forward the other message to Alice. After receiving the message, Alice will also compare value of M received from Bob with the original value. Alice will proceed only if these values of M are matching. Alice will decrypt the message and extract the shared secret key and store in its memory for future encryption of messages when communicating with Bob. Figure 2 depicts the process of proposed modified Otway-Rees protocol.

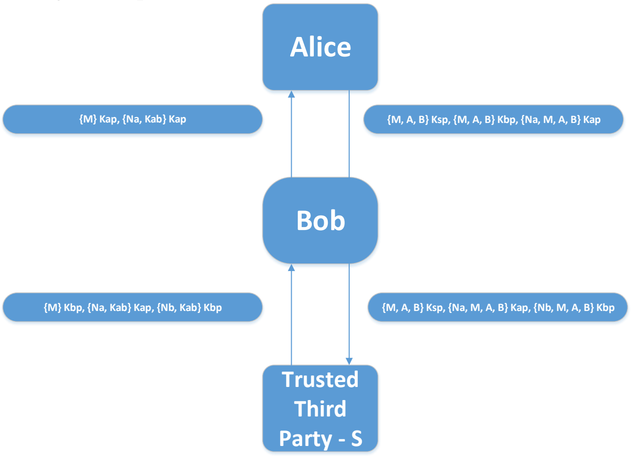
Alice
{M} Kap, {Na, Kab} Kap
{M, A, B} Ksp, {M, A, B} Kbp, {Na, M, A, B} Kap
Bob
{M} Kbp, {Na, Kab} Kap, {Nb, Kab} Kbp
{M, A, B} Ksp, {Na, M, A, B} Kap, {Nb, M, A, B} Kbp
Trusted Third Party - S

Fig. 2. Modified Otway-Rees Protocol

Following is the notation of modified Otway-Rees protocol:
A→B: {M, A, B} Ksp, {M, A, B} Kbp, {Na, M, A, B} Kap – (1)
B→S: {M, A, B} Ksp, {Na, M, A, B} Kap, {Nb, M, A, B} Kbp – (2)
S→B: {M} Kbp, {Na, Kab} Kap, {Nb, Kab} Kbp – (3)
B→A: {M} Kap, {Na, Kab} Kap – (4)

IV. THE EXPERIMENT, EVALUATION METHOD AND RESULTS

The encryption algorithm has a vital role in ensuring information security. Therefore, it is important to verify its performance. Some major parameters in an evaluation of encryption algorithm include security analysis, encryption speed and power consumption to name a few. There is a significant discussion available on cryptanalysis of Otway-Rees algorithm [11] [12] [13], hence we do not include that here.

The following parts of this section describe the design of the experiment to evaluate the encryption speed and power consumption of the proposed protocol.

From a brief review of the literature, we can deduct that the speed of encryption of the protocol and the energy requirements not only depend on protocol algorithm but also on the size of the block and the size of the key used for encryption [14] [15] [16] [17]. Therefore, we adjust the block size and the key size to 64-bit each.

A. Encryption Speed Evaluation

As the public key encryption scheme is more computationally intensive compared to symmetric key encryption scheme, the speed of encryption is considered an important indicator of the performance of the protocol. The encryption speed is the number of bytes that the protocol is able to encrypt in one time unit. As the information is measured in bytes, we will consider bytes that can be encrypted per second to measure the encryption speed.

In case of our proposed protocol, encryption time is time it takes for different parties to encrypt given message into cipher text using public keys of different parties. We consider the encryption speed for Alice, Bob and S differently.

B. Power Consumption Evaluation

Power consumption during the encryption process is another important performance indicator of the protocol. This is especially true in case of wireless devices that have limited battery capacity. Significant references are found in the literature regarding power consumption. Here, Kreiser presents a scheme with low energy overhead for key exchange by automation systems [18]. Lee argues about an energy efficient protocol for key exchange in sensor network environment [19]. Here the authors present study on Diffie-Hellman key exchange protocol's energy consumption in wireless sensor network scenario [20].

In our experiment we execute the encryption operation repeatedly for 10 million times on a laptop computer without external power source. We calculate battery consumption based on observing remaining battery percentage after the encryption process.

C. Experiment Results

To compare the performance of both protocols, both were implemented using common language and tested on a common platform for evaluation. The protocols were executed in Python under Windows 10 operating system. The platform was a laptop computer with 8GB RAM, Intel core i7 processor with 1.7 Ghz clock speed.

We also note that as per the findings from the review of the literature, the packet size also affects the communication efficiency. It is shown in paper by Prasithsangaree et. al that the average packet size is between 64-128 bytes [21]. Therefore, we also randomly generate packets with size of 128 bytes for this experiment. The experiment results are shown in table 2.

The table shows results of runtime, encryption speed and remaining battery for all the parties. The first column shows the details of protocols we are comparing. Second column is about the number of cycles in millions. Third column is about runtimes showing number of seconds the program takes to complete a cycle. The forth column is showing the speed of encryption, this is denoted by the number of bytes being encrypted per second by the algorithm. This is derived from total bytes encrypted divided by number of seconds it takes for execution given in runtime. The last column shows remaining battery percentage after the algorithm has executed its given number of cycles.

Based on the figures 3 and 4, we can deduct that the original Otway-Rees protocol is both faster at the encryption and consumes less power when compared to proposed modified Otway-Rees protocol. Figure 5 shows remaining battery power for both protocols. However, the difference of performance in speed and power consumption between the two protocols is not significant if enhancement of security is also taken into the consideration as the proposed modified Otway-Rees is more secure than the original Otway-Rees protocol.

V. CONCLUSION

The key exchange algorithm is vital for exchange of shared secret keys over insecure networks. This results in information privacy and security. In this paper we studied Otway-Rees algorithm and its vulnerability. We also proposed a modified Otway-Rees algorithm that is not vulnerable to triple M, A, B attack. We then evaluated the performance of both the original and proposed protocols based on encryption speed and power consumption. We see that the original Otway-Rees protocol is slightly better as it requires less number of encryption operations overall when compared to proposed modified protocol. However, our proposed protocol is more secure than the original protocol. This is a tradeoff between performance and security where security will be preferred over performance as the difference in performance between the two protocols is not significant.

In the future research we plan to modify and adjust the Otway-Rees protocol for Internet-Of-Things (IOT) applications over wireless communication.

Table 2. Performance of the protocols

Protocols	Cycles (Mln)	Runtime (Sec)			Speed (Bytes/Sec)			Remaining Battery (%)		
		A	B	S	A	B	S	A	B	S
Original OR	10	176.25	207.37	257.61	1426 1647	1097 6352	0818 4276	92	88	84
	20	297.46	412.70	508.24	1347 0968	1099 2251	0820 7147	86	81	71
Modified OR	10	182.13	219.38	274.93	1238 3667	1014 3959	0803 4257	89	81	83
	20	376.24	443.77	521.46	1251 2174	1028 1453	0827 6174	79	76	72

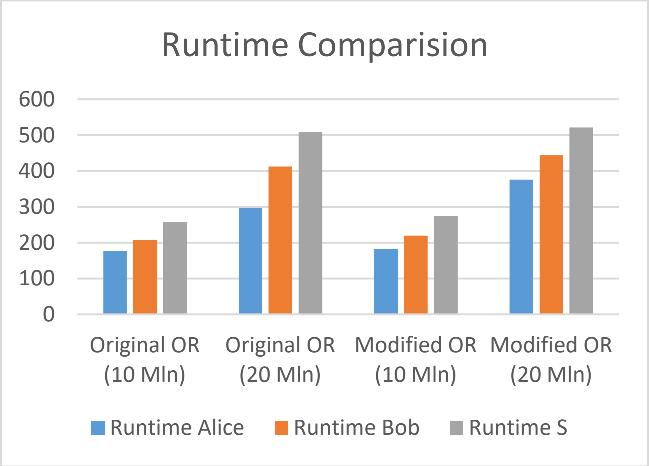

Figure 3. Runtime Comparison for 10 million and 20 million cycles of both protocols

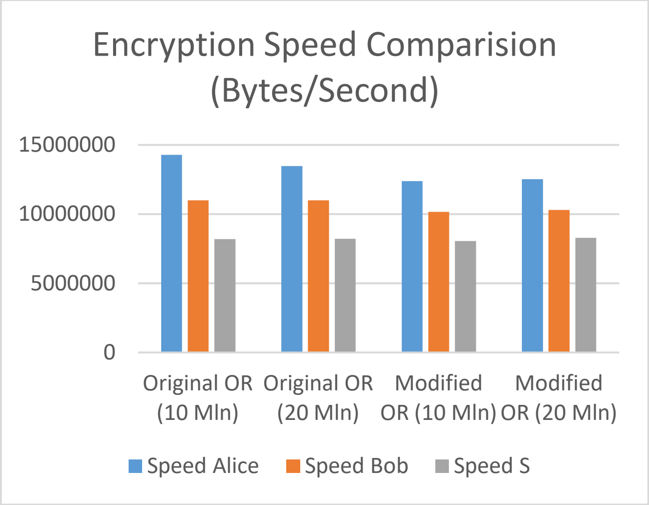

Figure 4. Speed Comparison

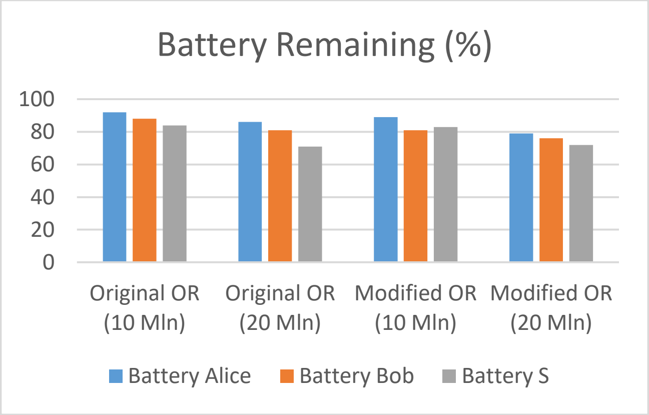

Figure 5. Battery Remaining

REFERENCES

1. B. Schneier, Applied Cryptography: Protocols, Algorithms and Source Code in C, John Wiley & Sons, 2017.
2. D. Whitman and M. Hellman, "New Directions in Cryptography," IEEE Transactions on Information Theory, vol. 22, no. 6, pp. 644-654, 1976.
3. M. Satyanarayanan, "Integrating security in a large distributed system," ACM Transactions on Computer Systems, vol. 7, no. 3, pp. 247-280, 1989.
4. J. G. Steiner, C. B. Neuman and J. I. Schiler, "Kerberos: An Authentication Service for Open Network Systems," Usenix Winter, pp. 191-202, 1988.
5. R. Cramer, I. Damgård and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in Cryptology Conference (CRYPTO'94), Santa Barbara (California, USA), 1994.
6. R. M. Needham and M. D. Schroeder, "Using encryption for authentication in large networks of computers," Communications of the ACM, vol. 21, no. 12, pp. 993-999, 1978.
7. D. Otway and O. Rees, "Efficient and timely mutual authentication," Operating Systems Review, vol. 21, no. 1, pp. 8-10, 1987.
8. D. E. Denning and G. M. Sacco, "Timestamps in key distributed protocols," Communication of the ACM, vol. 24, no. 8, pp. 533-535, 1981.

9. B. C. Neuman and S. G. Stubblebine, "A note on the use of timestamps as nonces," Operating Systems Review, vol. 27, no. 2, pp. 10-14, 1993.
10. J. A. Clark and J. L. Jacob, "A survey of authentication protocol literature : Version 1.0," University of York, York, 1997.
11. B. Groza and D. Petrica, "Cryptanalysis of an authentication protocol," in Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 2005.
12. C. Boyd, A. Mathuria and D. Stebilla, "Introduction to Authentication and Key Establishment," in Protocols for Authentication and Key Establishment, Springer, 2020, pp. 1-52.
13. Santos-González, A. Rivero-García, M. Burmester, J. Munilla and P. Caballero-Gil, "Secure lightweight password authenticated key exchange for heterogeneous wireless sensor networks," Information Systems, vol. 88, no. 1, 2020.
14. S. Koteshwara, A. Das and K. K. Parhi, "Architecture Optimization and Performance Comparison of Nonce-Misuse-Resistant Authenticated Encryption Algorithms," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 5, pp. 1053-1066, 2019.
15. M. F. Mushtaq, J. Abdulkadir, H. Disina, Z. A. Pindar, S. A. Shakir and M. M. Deris, "A survey on the cryptographic encryption algorithms," International Journal of Advanced Computer Science and Applications, vol. 8, no. 11, pp. 333-344, 2017.
16. G. Yadav and A. Majare, "A comparative study of performance analysis of various encryption algorithms," in International Conference On Emanations in Modern Technology and Engineering, 2017.
17. Afolabi and O. Atanda, "Comparative Analysis of Some Selected Cryptographic Algorithms," Computing, Information Systems, Development Informatics & Allied Research Journal, vol. 7, no. 2, pp. 41-52, 2016.
18. D. Kreiser, Z. Dyka, I. Kabin and P. Langendoerfer, "Low-energy key exchange for automation systems," in 3th International Conference on Design & Technology of Integrated Systems In Nanoscale Era, 2018.
19. J. Lee, A. Lee and M. Jun, "Sensor Authentication and Key Exchange Protocol for Energy Efficiency in Sensor Network Environment.," Advanced Science Letters, vol. 22, no. 9, pp. 2475-2479, 2016.
20. Q. A. Al-Haija, H. Enshasy and A. Smadi, "Estimating energy consumption of diffie hellman encrypted key exchange (DH-EKE) for wireless sensor network," in IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing, 2017.
21. P. Prasithsangaree and P. Krishnamurthy, "Analysis of energy consumption of RC4 and AES algorithms in wireless LANs," in IEEE Global Telecommunications Conference, 2003.
22. J. Daemen and V. Rijmen, "AES Proposal: Rijndael," National Institute of Standards and Technology., 2003.
23. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall and N. Ferguson, "Twofish: A 128-Bit Block Cipher," 15 June 1998. [Online]. Available: https://www.schneier.com/academic/archives/1998/06/twofish_a_128-bit_bl.html. [Accessed 07 04 2020].
24. W. Diffie and M. E. Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard," Computer, vol. 10, no. 6, pp. 74-84, 1977.

AUTHOR'S PROFILE

Pranav Vyas, Ph.D. is currently working as an Assistant Professor at Smt. Chandaben MMohabha Patel Institute of Computer Applications, Charotar University of Science and Technology. He has more than 10 years of academic experience in graduate courses, primarily MCA (Master of Computer Applications). His research interests include applied cryptography and Information Security.