Int. J. Nonlinear Anal. Appl. 14 (2023) 1, 2101–2110
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2022.27504.3628

Provide an algorithm for connecting bootstrap-based smart objects to the cloud, using asymmetric encryption

Alireza Zeynali

Department of Electrical and Computer Engineering, Isfahan University Of Technology, Isfahan, Iran

(Communicated by Javad Vahidi)

Abstract

CCloud computing and the Internet of Things have both experienced rapid and independent evolution. These two technologies are very different from each other and most of their features complement each other. These properties, which complement other technologies, are the main reason why researchers propose to integrate these two technologies, which in certain cases can be of great benefit. Therefore, the present study aimed to provide an algorithm for the connection of smart objects based on bootstrapping with the cloud, using asymmetric cryptography. In terms of purpose, this research is in the category of applied research. All relevant documents from written sources, including books, articles published in reputable scientific-research journals, research reports on the subject and academic dissertations, and valid Internet resources, and Related to the subject are studied and used. Using the public key on a resource-limited smart device with CoAP, we learn how to obtain a 3GPP-based public bootstrap architecture to ensure authentication and connectivity across a variety of devices. Then, with the help of standard protocols, including RADIUS and EAP, without the need to install new software on the phone, it is possible to automatically set up a wireless network with content to communicate with the cloud. Improved protocols meet the requirements of key IoT security services such as privacy, publicity, and credibility, and can achieve better performance with lower communication costs. On the other hand, the goal of most methods is to increase the scalability and performance of the system and control access to prevent unauthorized access.

Keywords: Smart objects, Bootstraping, Asymmetric encryption
2020 MSC: 68P25, 74M05

1 Introduction

Because smart objects should be available at relatively low prices, it may not be possible to add additional hardware features to secure credit storage or to secure communications, at an additional cost. In addition, smart objects are often located in small spaces or inaccessible places, so adding new hardware or regular access to install software updates and fix security vulnerabilities is very limited [13]. Another factor that plays an important role in designing security solutions for smart objects is the nature of their limited resources; these devices have a small amount of memory and computing power of limited energy sources [21]. Therefore, in many cases, devices must be asleep for a long time, to save energy, and be able to wake up only for a short time to report sensor reports [12]. Such intelligent objects cannot be exposed to data streams or security protocols for long periods of time [3]. The process is called automatic startup or

Email address: alirezazeynali75@gmail.com (Alireza Zeynali)

Received: February 2022 *Accepted:* May 2022

bootstraping. It is a process in which a node, after going through the validation and authorization processes, gathers any information (not only IP address but also security parameters) to join a network or service. This information can include, for example, accessible encrypted requests, shared keys, certificates, service parameters, etc. [1, 2, 3, 4]. The duration of bootstraping and its encryption algorithm after the smart device is turned on is the main location of intrusive attacks on smart devices [5].

Given that the smart object has limited resources, the sensory information recorded by that object based on a programming logic based on cloud computing should be distributed between the smart object device and an integrated server with better computing capabilities and network, and a server it may provide application logic for storing processing history and analyzing sensor data received from multiple devices, such as resource constraints. These servers are mostly placed in the cloud to achieve lower costs [7]. On the other hand, the growing popularity of web protocols and programming languages, many smart objects with low-power and cheap microcontrollers have covered their weakness with JavaScript and HTML, and given the low cost microcontrollers [6], IP networks [8], cloud computing [9] and Web technologies [11] are the building blocks of the Internet of Things. There are still many challenges to the pervasive IoT that need to be addressed, the most important of which is to ensure the security of smart objects. Recent evidence of spying on the Internet and wireless communications by intelligence agencies has not only highlighted the shortcomings of current technical solutions, but also the awareness of the importance of security and privacy among the general public [12]. Standardization organizations such as the Internet Engineering Working Group (IETF) Partnership Projects and Third Generation Joint Projects (3GPP) have responded to security and privacy concerns, prioritizing security for new protocols and communication architectures, but with respect to Different security protocols, no clear agreement on how to apply them to smart objects connected to the Internet with limited resources available, and at bootstrap [13].

Given this heterogeneity in devices and the variety of scenarios that may arise, this research proposal does not claim to provide a public security solution, but rather we want to design, implement, and evaluate a security architecture for low-consumption smart objects while sleeping. They are routed to a port or proxy to transfer data to the cloud. This will include designing an energy efficient communication model that will work with the Limited Application Protocol (CoAP) [**?**].

2 Research background

Liao and Hsiao proposed the process of validating and controlling access to the IoT perception layer. This procedure uses simple and effective two-way authentication as well as creating a security key based on elliptic curve (ECC) encryption with very low storage and communication overheads. For the access control policy, an ABC-based validation method has been used. Their architectural design is mainly based on the concept of base station (BS), which collects data and controls sensor nodes. Two-way authentication ensures the security of communication between the user and the nodes, which provides a very simple process to solve the problem of limited resources in the IoT perception layer. Access to data can be accompanied by accurate and flexible access control based on user-specific certifications in the access control reference [10].

In 2015, Vučinić et al. examined a new approach to E2E and IoT security based on the concept of object security when downloading applications, and considered confidentiality and independent credibility for domain trusts. Not only does this support the concept of object security of multicast privacy, but it also protects any client from security tampering when uploading to servers. This allows us to rely on secure communication channels with licensed servers that are responsible for managing access to the access key [23].

In 2015, Antikianen et al. proposed several suggestions for authenticating textual or spatial information, or inputs such as voice or motion, to reduce the need for user interaction in the device authentication process. Proposed protocols use common fuzzy confidentiality verification methods. After examining existing methods for creating shared privacy, the researchers used a noisy input and developed a new protocol that is an integrated, time-based package [1].

In 2017, Neto et al. introduced a topic called Object Identification (AoT). This topic provides protocols for authentication and access control for the device lifecycle in the IoT environment. They used an Android smartphone like the LG G4 to test their point, and evaluated all cryptographic initiatives such as implementing a feature-based signature design (ABS) on more limited devices such as the Intel Edison and Arduino Due. The results showed that in devices limited to resources, the implementation of object authentication is more economical than having a device with rich but expensive resources [14].

3 Proposing a cryptographic method

Due to the vulnerabilities of the DTLS protocol method and the weaknesses of the two-step protocol, based on the elliptic curve cryptography, the three-step structure, which is derived from the two-step structure and also eliminates its weaknesses, is proposed and This is because elliptic curve encryption based on implicit certificates provides less overhead for limited networks [15]. It can therefore be used for IoT authentication mechanisms. The main components of the proposed protocol are as follows:

- Certificate Applicant Existence (u): Issues a communication request.

  Authentication Server (AS): Each u entity is first required to prove its identity to this server and receive a password to communicate with the RC.

  Registration Center (RC) server: issues an implicit certificate for each u entity. Server entity (V): Provides a specific service for u entity.

- If the implicit certificate C for the existence and the public key of the RC is also P, if the certificate C belongs to the existence of u; then the public key is also K. This algorithm will have less overhead for devices with limited resources.

3.1 Variables of cryptographic methods

Table 1 lists the variables involved in the cryptographic proposal.

Table 1: Encryption suggestion variables

Private key of existence u	d_u	The first number and represents the field Fq	q
Public key of existence u	Q_u	The coefficients of the equation $Y2 = X3 + aX + b$	$b - a$
Random nonce number generated by entity u	N_u	Base Point with order n	p
Connection key between entity u and v	$K_{v,u}$	The integer of the series produced by the existence of u	r_u
Meeting key to communicate with RC	K_s	The point of the elliptic curve for the certificate request, sent by the entity u	R_u
The secret key of existence u	K_u	Implicit certificate of existence u	$Cert_u$
Public key RC server	K_{RC}	An integer to store the hash value of Certu	e
		An integer to calculate the private key of the applicant's existence	s

3.2 Initialization

As in Figure 1, the u entity has a secret key (Ku) and AS has encrypted a regular pair (U, Ku) with its own key, which is embedded inside the device [16]. This key will be erased from the device memory after performing the authentication steps.

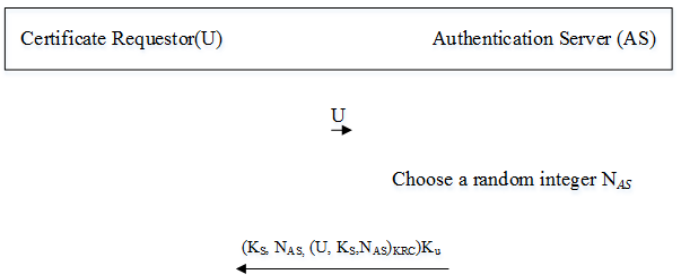

Figure 1: Initial initialization step

It is assumed that the individual network administrator is trusted and the initial network configuration is done securely. Initially, u explicitly sends its user ID, U, explicitly to AS. AS, after authentication, generates the parameters

Ks and KRC (U, KS, NAS) and NAS (random number) for us and sends them after encryption with the ku key [18]. It then erases the regular pair corresponding to u. The entity, using the ku key (of which only it is aware), decrypts the message and extracts KS and the ticket from it. And then erases the Ku key from its memory.

3.3 Authentication

As in Figure 2, in order to establish authenticated communications, entities must have an implicit certificate. At this stage, the connection is between two entities, which according to the contract, the first party is called the client (u) and the second party is called the server (v). The existence of u sends the following parameters to v:

- Cipher suite
- Certu
- U
- (Certu, Nu)

Existence v calculates the public key using the received implicit certificate and decrypts the text with the obtained public key and ensures the accuracy of the received certificate and its freshness with the help of Nance [17]. Then, for the freshness of the Kvu key in each session, it selects a random number (Numr) and finally generates the Kv, u key and a random value (NV), then to confirm the validity of the sent parameter certificate (Certv, Certu, Nv) which With its private key encrypted with Numr parameters, Certv encrypts u with the public key of the entity u and sends the encrypted value along with the other values.

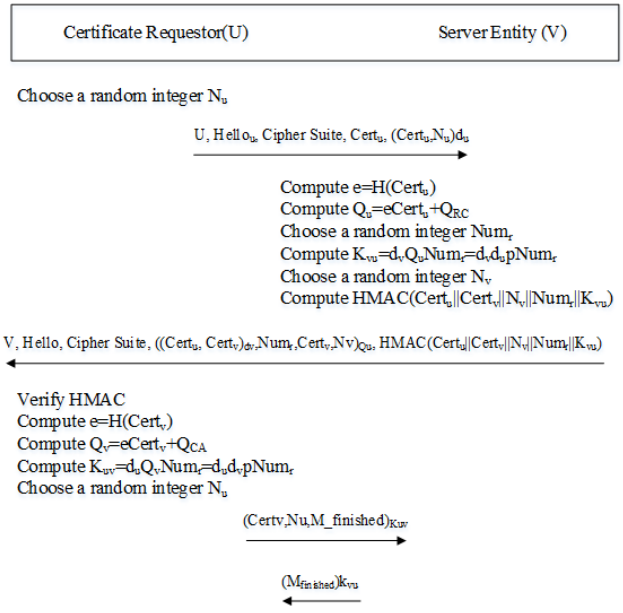

Figure 2: Proposed authentication

Because only u has this private key it can decrypt the text. The public key v is extracted from the implicit v certificate and decrypted by it, and the authenticity of the received certificate and the public key extracted from it is verified, and then the value of the kuv key is calculated and the HMAC value is checked. It then encrypts Certv with the random value (Nu) by kuv to prove its identity for v. It also encrypts and sends the finished message along with the previous two values to complete this step. Since only v knows about the common key, it can decrypt it, and by seeing the certificate it sent to u, our identity is revealed to it. To complete this step, the v entity encrypts the finished message with the kvu key and sends it to u. After completing this step, the two entities u and v are fully assured of the authenticity as well as the freshness of the messages and keys, and can exchange encrypted messages with each other using their shared key [19].

3.4 Security check of the proposed encryption protocol

3.4.1 Bilateral authentication

In the initialization phase, since only AS and u know the secret key of u's existence, then only AS can issue a ticket for us and encrypt with this key, and on the other hand, only u can decrypt it. In the registration phase, the RC extracts the real identity and the K key from inside the ticket [20]. The identity of the RC is also proved for the existence of u, because only the RC could decrypt the ticket and extract K from it and decrypt and respond to the other messages. In the authentication phase, the two entities u and v send their certificate encrypted with their private key and use each other to calculate the public key and obtain the $Kuv = Kvu$ session key. As a result, according to the mentioned cases, the proposed protocol provides mutual authentication.

3.4.2 Confidentiality

In the initialization phase, the exchanged message is encrypted with the secret u key and this key is also erased from memory. In the registration phase, all messages exchanged between us and the RC are encrypted. Also for the authentication phase, the Kuv = Kvu session key is used to encrypt the messages between the two entities u and v based on the discrete logarithm; Is provided and the attacker can never access this (Kvu) and therefore can not decrypt messages encrypted by this key [21]. As a result, the proposed protocol has the feature of confidentiality.

3.4.3 Recently sat down key

In this protocol, because a random number numr is involved in each session to generate the session key, the session key is fresh and updated.

3.4.4 Style calculations

For symmetric encryption, 128-bit AES-CTR algorithm is used, for one-way encryption, SHA-1 algorithm is used, and for asymmetric encryption, elliptic curve (ECC) encryption with 160-bit elliptic curve parameters is used [22]. This structure produces an implicit certificate of 44 bytes, unlike DTLS, which has a key size of 2048 bits and a certificate of 1 KB. The advantage of using ECC is that by providing proper security, it also brings less overhead. Also, to ensure the accuracy of the exchanged message, use HMAC, which is one of the fastest methods in the world and its implementation at the hardware level is much lower than other methods.

3.4.5 Session key security

This means that the key to the meeting at the end of the key agreement process is unknown to anyone but the two parties. In the proposed structure, the session key is not known to any entity other than the legitimate entity, because Kuv = Kvu = duQvNumrmodp = dvQuNumrmodp is not obtained by an attacker who does not know du or dv and Numr. Therefore, the proposed method provides session key security [23].

3.4.6 Stealth forward

That is, if an attacker gains access to the server's secret key or entity password, he or she will not be able to calculate previous session keys. In the proposed scheme, the session key is calculated using the formula Kuv = Kvu = duQvNumrmodp = dvQuNumrmodp. And due to the difficult ECDLP problem the attacker can not get du through Qu = dup or dv through Qv = dvp. As a result, the proposed method provides this feature [24].

3.4.7 Known key secrecy

This means that even if the attacker gets the key to one of the old sessions; can not access other session keys from this key and the key must be unique. In the proposed method, this feature also applies due to the novelty of the session key property [25].

3.5 Investigate vulnerabilities to various attacks of the proposed encryption protocol

3.5.1 Stolen confirmation attack

In a group of authentication methods, servers have a table that stores entity authentication information, but if this important information is stored without encryption, an attacker could easily access this sensitive information by

accessing this table. And the security of the protocol is compromised [26]. In the initialization phase, the ordered pair (U, Ku) is encrypted by the public key AS. Which prevents the attacker from accessing this information if he has access to AS. On the other hand, assuming the initial deployment of the network is secure, Ku will be deleted from the u-memory after receiving the ticket. Also, in the proposed RC protocol, there is no information about the entities. Therefore, the proposed method is safe against this type of attack.

3.5.2 Middle person attack

In this attack, the attacker communicates between the two sides to eavesdrop on information and change the information exchanged. In our proposed protocol for communication with RC, the ticket (U, Ks, NAS) receives kRC from AS and this ticket is also encrypted with KRC, so only RC can see the contents of this ticket and on the other hand, along with the ticket, the key. It is also sent between RC and u, which is encrypted with the secret key u, and only u, which is aware of this key, can access these parameters. Also, all messages and parameters of the registration phase are sent by password with this key. In the authentication phase, the information is encrypted by a common key agreed upon by the Diffie Hillman method. And since this method is vulnerable to attack by the middle person; in the proposed protocol to prevent this type of attack, an implicit certificate of entities is sent. By sending certificates and extracting the public key from this certificate, the identities u and v are verified for each other. Due to the confidentiality of the data, the attacker does not access the content of the exchanged information by listening to the communication channel, and as mentioned, the proposed protocol can provide mutual authentication between the two communication parties.

3.5.3 Repeat attack

In a repeat attack, the attacker falsifies the identity of one of the communication parties by resending one of the obtained authentication messages. At all stages, a random amount of Nance is sent in the amount of HMAC and encrypted messages. Also, due to the time and date of sending the request by the entity u to RC, which is encrypted with the key, the message to RC is recently updated. In the authentication phase, for randomness of the message, the random value of Nance is used, and for the novelty of the key, the random number, Numr, is used on both sides of the communication. As a result, the proposed protocol is safe against repeat attacks.

3.5.4 Service blocking attack

An attacker can prevent a server or entity from responding to incoming messages by generating heavy traffic or by temporarily disconnecting. In this type of attack, similar to the repetitive attack by sending the time and date of request and also sending a random amount of nance protocol is also resistant to this attack.

3.5.5 Message change attack

In this type of attack, the attacker tries to change the authentication messages in an unauthorized way. In the proposed protocol, because only AS has a regular pair (U, Ku), it can issue a ticket for this entity. Also, since this ticket is encrypted with the public key RC, only RC can decrypt it and access ks and issue an implicit certificate for this entity. On the other hand, only u, which is aware of the ks key, can decrypt the message containing the implicit certificate and receive the certificate. The ticket is also issued in the name, which means that the identity of the ticket applicant is explicitly included in the ticket. In this way, the possibility of someone with the highest level of license being able to purchase a ticket for an unlicensed entity, and be issued a certificate for this entity is eliminated. Also, the authentication phase depends on the certificate of the entities. As a result, according to the mentioned cases, the proposed structure is not vulnerable to this type of attack.

4 Analysis of findings

4.1 Estimation by program method

We have implemented the proposed cryptographic algorithms of Pourambagh et al. In C ++, the code of which is given in Appendices A and B. And then in a computer whose specifications are listed in Table 3 and compiled and executed with Visual Studio 2017. Figures 3 and 4 are shown.

According to the results of Table 4, the average time of execution of Purambg et al cryptographic protocol is 44.747744 ms and the proposed cryptographic protocol is 28.394183 ms, which shows a decrease of 36.55%. Joules

Table 2: Specifications of the computer running the algorithms

Intel(R) Core(TM) i3-2100 CPU @ 3.10 GHz	CPU
4 GiB DIMM DDR3 Synchronous 1333 MHz	RAM
windows 7 home premium 64 bit	O.S.

Table 3: Execution time and energy consumption of the proposed cryptographic protocol and Pourambag et al. based on computer program

	Protocol of Pourambagh et al		Proposed encryption protocol	
Performance	Execution time Millisecond	Energy consumption Microgel	Execution time Millisecond	Energy consumption Microgel
1	39.695271	314.5752	27.997135	6.774902
2	49.269812	311.4624	28.012500	5.584717
3	39.142058	308.9905	28.524019	7.415771
4	45.892009	315.7654	28.951866	7.415771
5	38.773785	311.4624	29.291832	4.943848
6	48.853489	313.9343	28.062786	4.943848
7	39.299551	313.9343	27.743472	6.774902
8	41.948121	311.4624	28.428942	7.415771
9	46.099647	313.9343	27.472838	4.943848
10	46.726752	313.9343	27.777485	4.943848
11	46.520161	312.1033	28.189758	5.584717
12	47.393846	315.7654	29.953782	5.584717
13	39.419678	313.9343	28.350951	6.774902
14	48.452041	311.4624	28.246679	4.943848
15	42.934851	312.1033	27.930228	4.943848
16	45.814485	314.5752	28.328602	5.584717
17	46.373495	313.9343	28.708958	4.943848
18	46.284932	313.9343	27.692907	4.943848
19	49.778397	313.2935	28.324412	4.943848
20	46.282492	309.6313	29.894514	4.943848
Medium	44.747744	313.009640	28.394183	5.717468
Maximum	49.778397	315.765400	29.953782	7.415771
Minimum	38.773785	308.990500	27.472838	4.943848
Standard deviation	3.632262737	1.801961574	0.661992185	0.950330721

and the proposed cryptographic protocol is 5.717468 microJoules. May shows a decrease of 98.17%. Also, deviation from the proposed cryptographic protocol standard in both execution and energy consumption indicates a less scatter of this method than the Pourambag et al. figures 4 and 5 show a comparison between the average execution time and the energy consumption of the two protocols, respectively.

5 Conclusion

In this study, a secure and efficient authentication protocol was presented to the IoT protocol communication parties. This protocol, unlike the DTLS method, is based on an implicit certificate. Due to the small size of certificates, they consume less memory per IoT device. In the proposed structure, it is not necessary for the connection to be permanently open between the two parties, so network resources are not wasted. The proposed protocol also addresses the weaknesses and vulnerabilities of the Purambeg et al Protocol, thus establishing the features of two-way authentication, confidentiality, session key novelty, lightweight computing, session key security, forward secrecy, and known key secrecy. This structure is protected against theft authentication attack, middle person attack, duplicate attack, service block attack and message change attack. Due to the fact that IoT objects have limited resources, and according to the above, it can be seen that this structure can be implemented on a variety of devices even with the least resources. Also, considering the security evaluation and performance of the proposed protocol, this protocol can be a good alternative to the DTLS protocol.

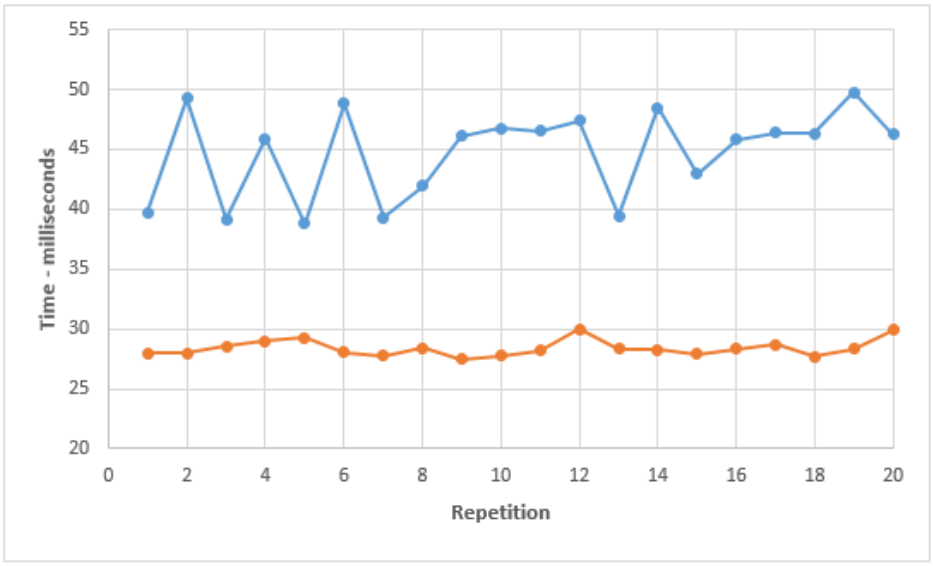

Figure 3: Execution time of the proposed cryptographic protocol and Pourambagh et al. based on a computer program

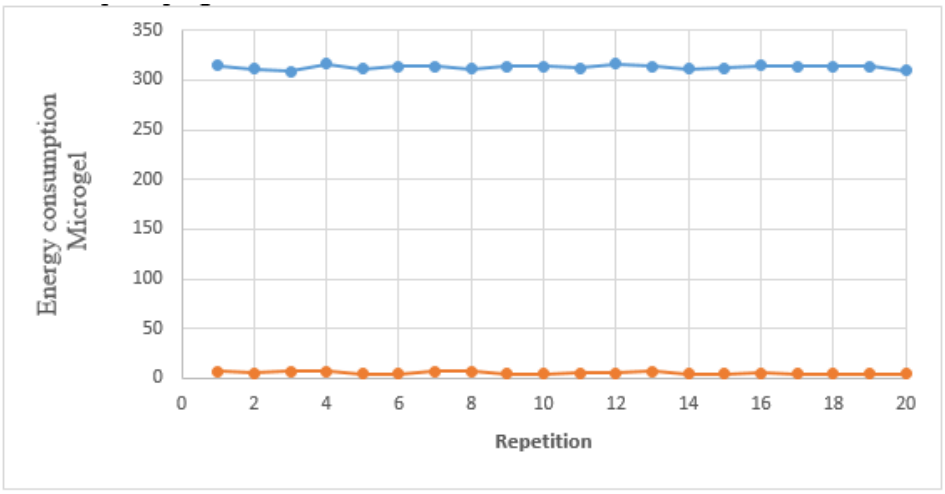

Figure 4: Energy consumption of the proposed cryptographic protocol and Pourambagh et al. Based on a computer program

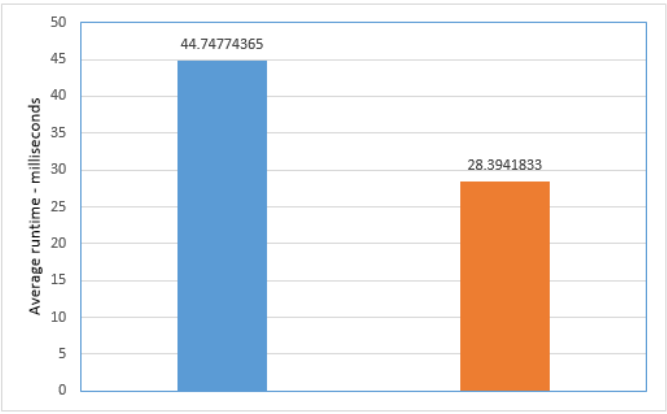

Figure 5: Comparison of execution time of the proposed encryption protocol and Pourambagh et al. based on computer program

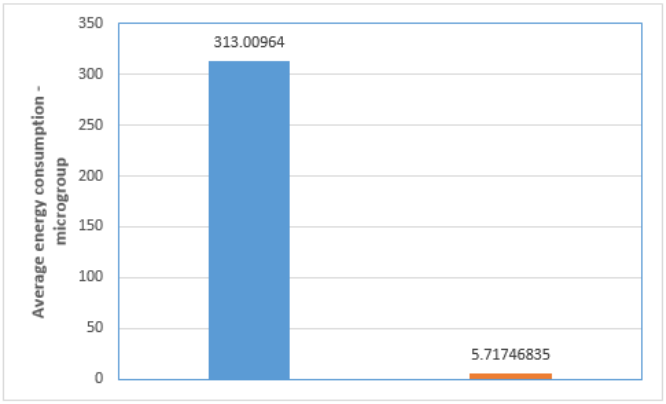

Figure 6: Comparison of energy consumption of the proposed cryptographic protocol and Pourambagh et al. based on computer program

References

[1] M. Antikainen, M. Sethi, S. Matetic and T. Aura, *Commitment-based device-pairing protocol with synchronized drawings and comparison metrics*, Pervasive Mobile Comput. **16** (2015), 205–219.

[2] M. Chen, S. Gonzalez, Q. Zhang and M.V.C. Leung, *Software agent-based intelligence for code-centric RFID systems*, IEEE Intell. Syst. **25** (2010), 12–19.

[3] U. Gasser, R. Faris and R. Heacock, *Internet monitor 2013: Reflections on the digital world*, Berkman Center Res. Pub. **27** (2013).

[4] G. Giaretta, J. Kempf and V. Devarapalli, *Mobile IPv6 bootstrapping in split scenario (No. RFC 5026)*, Network Working Group, 2007.

[5] J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, *Internet of things (IoT): A vision, architectural elements, and future directions*, Future Gen. Comput. Syst. **29** (2013), no. 7, 1645–1660.

[6] S. Hartman and J. Howlett, *A GSS-API mechanism for the extensible authentication protocol*, JANET(UK), 2013.

[7] T. Hannes, *Enriching bootstrapping with authorization information*, IETF **2005** (2005), 1-26.

[8] I. Ishaq, D. Carels, G. K. Teklemariam, J. Hoebeke, F.V.D. Abeele, E.D. Poorter and P. Demeester, *IETF standardization in the field of the internet of things (IoT): Asurvey*, J. Sensor Actuator Networks **2** (2013), no. 2, 235–287.

[9] V. Karagiannis, P. Chatzimisios, F. Vazquez-Gallego and J. Alonso-Zarate, *A survey on application layer protocols for the internet of things*, Trans. IoT Cloud Comput. 3 (2015), no. 1, 11–17.

[10] Y. Liao and Ch. Hsiao, *A secure ECC-based RFID authentication scheme integrated with ID-verifier transfer protocol*, Ad Hoc Networks. **18** (2014), 133–146.

[11] G. Marsh, A.P. Sampat, S. Potluri and D.K. Panda, *Scaling advanced message queuing protocol (AMQP) architecture with broker federation and infiniband*, Ohio State University, Tech. Rep. **38** (2008).

[12] P. Mell and T. Grance, *The NIST definition of cloud computing*, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology Gaithersburg, 2011.

[13] M. Nakhjiri, *AAA and network security for mobile access: Radius, diameter, EAP, PKI and IP mobilit*, John Wiley and Sons, Hoboken, NJ, USA, 2005.

[14] A. Neto, H. Patil, L. Oliveira, A. Souza, I. Cunha and M. Nogueira, *Aot: Authentication and access control for the entire iot device life-cycle*, Proc. 14th ACM Conf. Embedded Network Sensor Syst. CD-ROM, 2016, pp. 1–15.

[15] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby and C. Gomez, *IPv6 over bluetooth(r) low energy*, IETF, RFC7668, 2015.

[16] K. Rose, S. Eldridge and L. Chapin, *The internet of things: An overview*, Internet Soc. **80** (2015), 1–50.

[17] H. Shuai and X. Jianchuan, *Ensuring data storage security through a novel third party auditor scheme in cloud-computing*, IEEE Int. Conf. Cloud Comput. Intell. Syst. IEEE, 2011, pp. 264–268.

[18] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded internet*, Vol. 43, John Wiley and Sons, 2011.

[19] G. Suciu, A. Vulpe, S. Halunga, O. Fratu, G. Todoran and V. Suciu, *Smart cities built on resilient cloud computing and secure internet of things*, 19Th Int. Conf. Control Syst. Comput. Sci., 2013.

[20] D. Thaler, H. Tschofenig and M. Barnes, *Architectural considerations in smart object networking*, Tech. RFC **7452** (2015).

[21] J.R. Vacca, *Computer and information security handbook*, 2nd ed., Morgan Kaufmann: San Francisco, CA, USA, 2013.

[22] K. Vamsee and R. Sriram, *Data security in cloud computing*, J. Comput. Math. Sci. **2** (2011), 1–169.

[23] M. Vučinić, B. Tourancheau, F. Rousseau, A. Duda, L. Damon and R. Guizzetti, *OSCAR: Object security architecture for the internet of things*, Ad Hoc Networks **32** (2015), 3–16.

[24] R.H. Weber, *Internet of Things–New security and privacy challenges*, Comput. Law Security Rev. **26** (2010), no. 1, 23–30.

[25] R. Westerholt and B. Resch, *Asynchronous Geospatial Processing: An Event-Driven Push-Based Architecture for the OGC Web Processing Service*, Trans. GIS **19** (2015), no. 3, 455–479.

[26] Q. Zhang, L. Cheng and R. Boutaba, *Cloud computing: state-of-the-art and research challenges*, J. Internet Serv. Appl. **1** (2010), no. 1, 7–18.