Chapter

Advanced Lightweight Encryption Key Management Algorithms for IoT Networks

Menachem Domb

Abstract

An Internet of Things (IoT) Network is a collection of sensors interconnected through a network that process and exchange data. IoT networks need sufficient resources to cope with the growing security challenges. In most cases, cryptography is implemented by symmetric and asymmetric encryption methods to cope with these security issues. Symmetric cryptography requires transmitting an encryption key to the receiver to decrypt the received encrypted messages. Consequently, secured key distribution techniques are the core for providing security and establishing a secured connection among objects. Encryption keys are frequently changed through key distribution mechanisms. Encrypted key exchange is a protocol that allows two parties who share the same key to communicate over an insecure network. This chapter outlines the challenges and core requirements for a robust key distribution mechanism, beginning with evaluating existing solutions and then detailing three innovative, efficient, and lightweight methods that balance the security level, network performance, and low processing overhead impact.

Keywords: key management/distribution, symmetric/asymmetric encryption, IoT networks, lightweight RSA, probability-based keys sharing

1. Introduction

IoT devices collect and distribute massive transactions and data in real time non-stop, which requires some means to secure this data, such as data encryption.

Figure 1 depicts the IoT's pivotal role in the complete picture of computing and communications. IoT networks comprise a wide range of interconnected devices that collect and analyze environmental data and act using actuators. IoT networks are utilized in various sectors, including smart energy grids, industrial control systems, healthcare, transportation, home appliances, and wearables [1]. The evolving IoT array adds numerous devices to the Internet with poor security resistance, risking the entire community of Internet users.

Symmetric Encryption methods are the best-balanced solution for such an enormous load, as they are reliable and have a minimal performance impact. However,

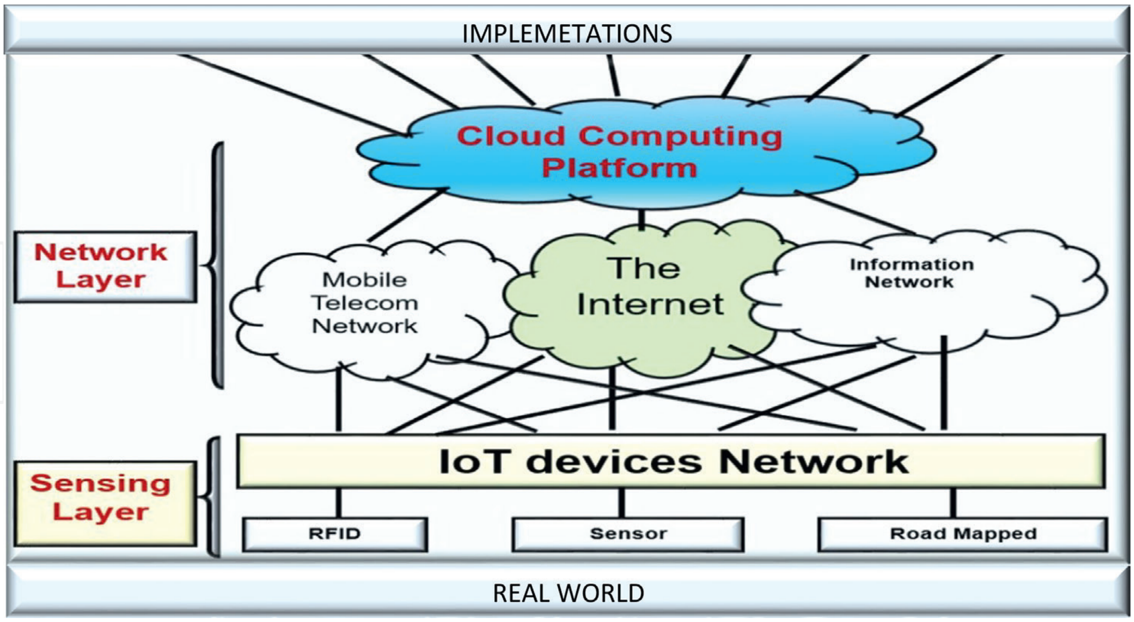

Figure 1.
IoT's role in the complete picture of computing and communications.

advanced processing capabilities reveal encryption keys instantly, forcing frequent key generation and spreading to have a unique encryption key per conversation, including generating, storing, distributing, and backing up the keys. IoT devices suffer from inherent weaknesses due to limited computing and communication resources, which prevent using standard key management systems designed for common networks. Dynamic key management schemes have already been proposed assuming homogeneous network architecture, while IoT networks are heterogeneous with no established standards. In this section, we suggest an ongoing key management process based on a probability analysis providing a key shared between any pair of IoT devices. The key size, randomness, sequence, and the number of alternate keys prevent attacks.

A comprehensive survey by Oraib et al. [2] outlines the most known key distribution methods suitable for IoT. They propose criteria to evaluate any key distribution schemes, which will be used later to compare various techniques in terms of performance and efficiency. The implementation should be scalable, resilient, and connective, while the efficiency concerns the node's communication, computing, and memory storage complexity. Some authors classified the key distribution based on the current proposals into trusted-server schemes and self-enforcing schemes. Hamid et al. [3] present four key distribution schemes, probabilistic, deterministic, hybrid, and group-based. Others classified the symmetric key distribution schemes in IoT into probabilistic, deterministic, and other categories. We propose to classify the secured key distribution into three classes, i.e., lightweight, robust encryption such as RSA, secure distribution by additional means such as internal CA, and key distribution means.

The chapter is organized as follows. Section 2 details the research related to secured key distribution and management. In sections 3, 4, and 5, we elaborate on the original and unique key distribution methods tailored to cope with sensor constraints, and in Section 6, we provide our conclusions.

2. Literature review

Naoui et al. [4] studied WSN key distribution protocols, mentioning centralized and decentralized methods, and concluded that a few proposals are comprehensive for different IoT applications. N-tier modeling of robust key management and cost-effective security paradigm with a 2-tier model to safeguard cloud data with effective authentication are addressed in [5, 6]. As the introduction explains, many solutions to the key distribution problem are classified into three categories. Below we provide a literature review for each of the methods.

2.1 Key distribution using lightweight encryption methods

A well-known key distribution method encrypts the distributed key using Asymmetric cryptography, such as RSA. However, Asymmetric encryption requires computation resources beyond a typical IoT device's resources. Therefore, a lightweight, fast, low computational cost algorithm is needed. Fadhil and Younis [7] discuss Elliptic Curve Diffie-Hellman (ECDH) key agreement algorithms for IoT. Goyal and Sahula [8] proposed a lightweight encryption algorithm for IoT devices using ECDH and AES encryption. Usman et al. [9] proposed an adaptive symmetric encryption algorithm (SIT) that merges Feistel and SPN with a 64-bit cipher. Nandini and Vanitha [10] analyzed several lightweight cryptography algorithms, such as HISEC, PRINCE, OLBCA, PRESENT, PRINT, TWINE, and KLEIN, and concluded that adding more S-boxes increases security. Some researchers implemented authentication by adding a new device as the authenticator. Ummer Iqbal et al. [11] proposed a lightweight ECC-based key exchange mechanism for fog federation. Their analysis indicates that it is safe from various attacks, with an overhead of 210.66 mJ and a communication overhead of 2144 bits, while conforming to the desired security specifications.

@@Eldefrawy et al. [12] presented a lightweight key distribution protocol for Industrial IoT that requires a single message exchange, handles node addition and cancelation, and fast rekeying. The scheme provides forward/backward privacy and avoids node capture and server takeoff attacks. Lian et al. [13] suited the traditional password-authenticated key exchange (PAKE) method to the IoT, with limited computing capability allowing two parties with a shared password to establish a session key. Their proposed protocol requires only three exponentiations per party while ensuring that the transmitted record and password file will not reveal the identity information. The Diameter protocol scheme provides a secure key agreement protocol that uses the ECDSA and the ECDH key agreement algorithm with less computational efforts suitable for IoT.

2.2 Key distribution assisted by external authentication means

Salman et al. [14] required a device to become a Certificate Authority (CA) server. Shivraj et al. [15] used cloud applications for constructing and distributing OTP. Aman et al. [16] proposed adding a separate device as a central authorization server and used simulations, BAN logic, and the Random Oracle model to assess its security strength simulations were performed for security verification. A detailed comparison of the proposed scheme with LKSE was conducted. Guo et al. [17] proposed a new AP-SGKD protocol using double chains and access polynomials. The new protocol is the AP-SGKD protocol that fulfills basic security properties with optimal storage requirements. Their simulation results showed that the new scheme could be applied to the Zigbee network since it performs well on security, storage, and communication. IoT nodes are connected to the Internet and communicate over a virtual network. No nodes with malicious intent are connected to the web to avoid cyberattacks. Moharana SR et al. [18] proposed a framework for the security over the virtual network for IoT nodes in a cloud system, including a lightweight cryptographic technique involving a key exchange protocol to establish secure end-to-end communication among the IoT nodes. This framework is a unique key exchange protocol between the CSP and the user group with the IoT nodes, which utilizes a balanced incomplete block design (BIBD) model.

Furthermore, it uses two communication channels, the Elliptic Curve Diffie Hellman (ECDH) protocol and the identity information for key exchange and sensor data communication. The ECDH generates the same shared secret key for the participants. Perfect Forward Secrecy (PFS) guarantees the safety of the hidden keys even if the private key is compromised. The secret key is derived using a hash function which is used later as the Key on Advanced Encryption Standard (AES) algorithm. The AES secures the sensor's data transmitted over the network.

2.3 Other key distribution methods

Orieb Abu Alghanam et al. [2] propose H2KD, a hierarchical architecture, and protocol for key distribution in IoT/WSN, which supports mobility, scalability, heterogeneity, and constrained nodes' limited capabilities. The performance was evaluated based on a quantitative measure of several metrics, memory storage, computation cost, scalability, the number of messages exchanged, and resilience required to establish a new session key for a mobile node. The results of their experiments show that the protocols are safe against attacks and reduce communication, computation, and storage costs for constraint nodes. The key agreement scheme uses the elliptic curve algorithm, and the symmetric encryption scheme uses AES and RC4. A quadratic-based wireless sensor key management scheme builds a shared key with a binary t-order symmetric polynomial, introduces a multivariate asymmetric quadratic polynomial, and utilizes the relationship between the quadratic eigenvalues and eigenvectors. It improves the anti-capture property, connectivity, scalability, communication overhead, and storage overhead. It is based on the Diameter protocol and introduces a new Z-Wave application layer protocol to provide end-to-end security. Othman et al. [19] propose a new AP-SGKD protocol using double chains and access polynomials. Their new protocol is the first AP-SGKD protocol that satisfies all basic security properties with an optimal storage requirement. In addition, they propose balancing the session key recovering time for less communication cost. Their simulation results show that their new scheme can be applied to the Zigbee network since it performs well on security, storage, and communication.

3. Probability-based keys sharing

This approach assumes a two-stage process. (i) Constructing a central pool of encryption keys and (ii) distributing sub-pools to the IoT devices. When two IoT devices want to establish communication, they identify a shared key, encrypt the message, and send it. These stages are executed within the IoT network devices.

The key-pool construction process is performed in a distributed and parallel mode. One IoT device is designated the Master, which requests the other devices to generate keys sent back and piled by the Master into one pool. The Master then randomly distributes keys to each IoT device such that any two devices have at least one shared key. **Figure 2** depicts the overall process stages.

We automated scalability and node mobility independent of this process in Python using Raspberry Pi-3 devices as IoT devices network. **Figure 3** illustrates the result of the key-pool set scattered into subsets where each subset has at least one shared key with another subset. Subsets are designated by the letter R and its key references. For example, nodes R3 and R4 share two keys, 8 and 5, and R4 and R5 share one reference key, 7. If there is no overlapping between two subnets, their communication is blocked. The implementation proves the feasibility of our proposed security protocol for IoT networks. The proposed scheme is symmetric or asymmetric and has network scalability and node mobility independent of the cryptography method. Assuming the keys are divided into subsets and distributed to all IoT network members. Each key has an assigned sequence number for security purposes, which will be used for inter-device communications. When a node is about to exchange messages with another node, it sends its list of key references. The receiving node intersects it with its subset, selects one of the intersected keys, and replies with the reference number chosen. The sender encrypts its message using the referenced key and sends it.

Eschenauer and Gligor [1] refer to a different setup, used a random graph, and employed probability-based key sharing. This setup does not contender with the unique IoT constraints. In [20], they used KMS and Asymmetric cryptography for IoT networks. Ciancalepore et al. [21] propose a Key Management Protocol for mobile and industrial IoT systems, with robust key negotiations, lightweight node authentication, fast rekeying, and efficient protection against replay attacks. It leverages ECC constructions, key exchange, and implicit certificates. Its advantage is that it allows suitable integration in a security protocol exchange such as 802.15.4. Roman et al. [22]

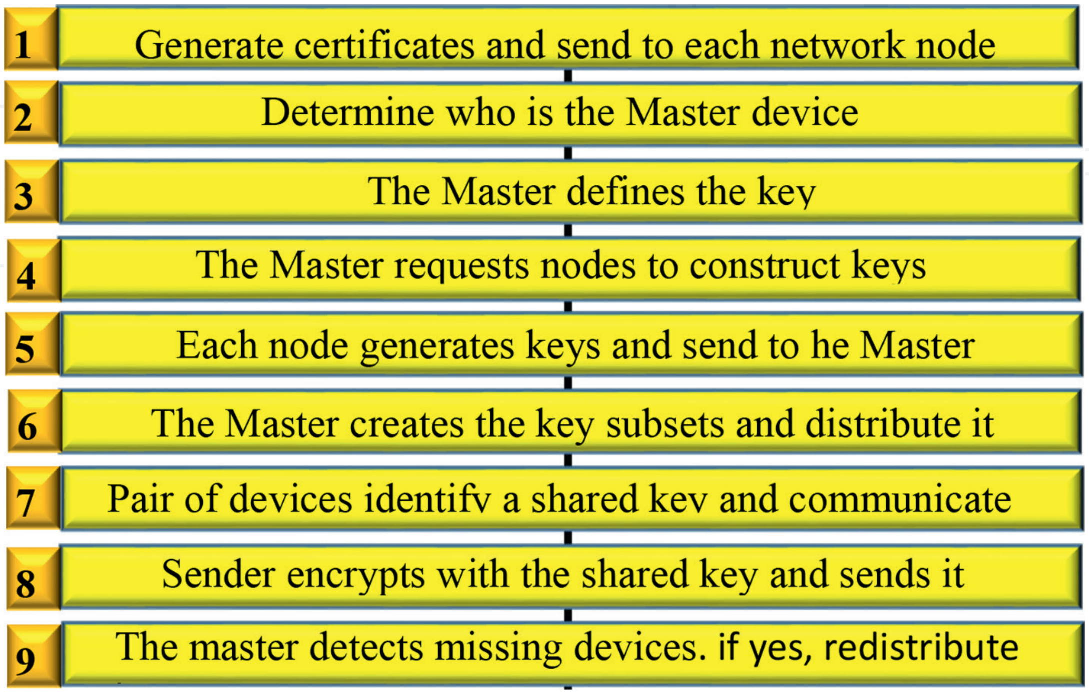
1
Generate certificates and send to each network node
2
Determine who is the Master device
3
The Master defines the key
4
The Master requests nodes to construct keys
5
Each node generates keys and send to he Master
6
The Master creates the key subsets and distribute it
7
Pair of devices identifv a shared kev and communicate
8
Sender encrypts with the shared key and sends it
9
The master detects missing devices. if yes, redistribute

Figure 2.
Probability-based keys sharing stages.

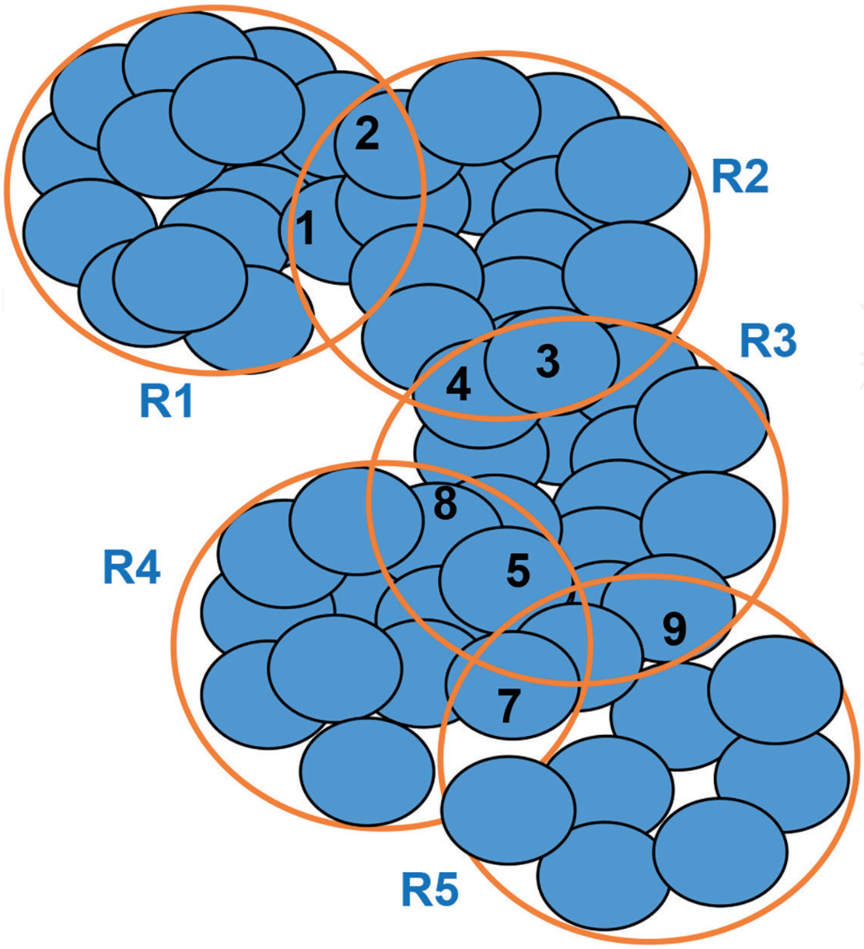

Figure 3.
The key pool after its division into subsets and before distribution to the devices.

propose key management mechanisms enabling two remote devices to negotiate specific security credentials while providing shared keys for sensors in the same network.

Wazid et al. [23] designed a new secure, lightweight three-factor remote user authentication scheme for IoT, using automated validation of Internet security protocols, offering offline sensing node registration and anonymity. Benslimane and BenAhmed [24] propose a lightweight key management protocol that allows the constrained node to transmit captured data to an internet host on a secure channel. Mahmood and Ghafoor [25] propose an Efficient Key Management (EKM) scheme for multiparty communication-based scenarios. The proposed session key management protocol applies a symmetric polynomial for group members. The polynomial generation method uses security credentials and a secure hash function.

3.1 Experiment setup and results

In this section, we demonstrate the operation of the Probability-Based Keys Sharing protocol we developed, as described in Section 3, with the new approaches to dealing with the vulnerabilities of previous protocols. We performed a lab experiment with 3 Raspberry Pi 3 Model devices (#1: 10.0.0.26, #2: 10.0.0.10, #3: 10.0.0.5). We executed the following steps. **Figure 4** outlines the six stages of generating the whole set of keys and dividing it into subsets with at least one overlapping key, as elaborated herein.

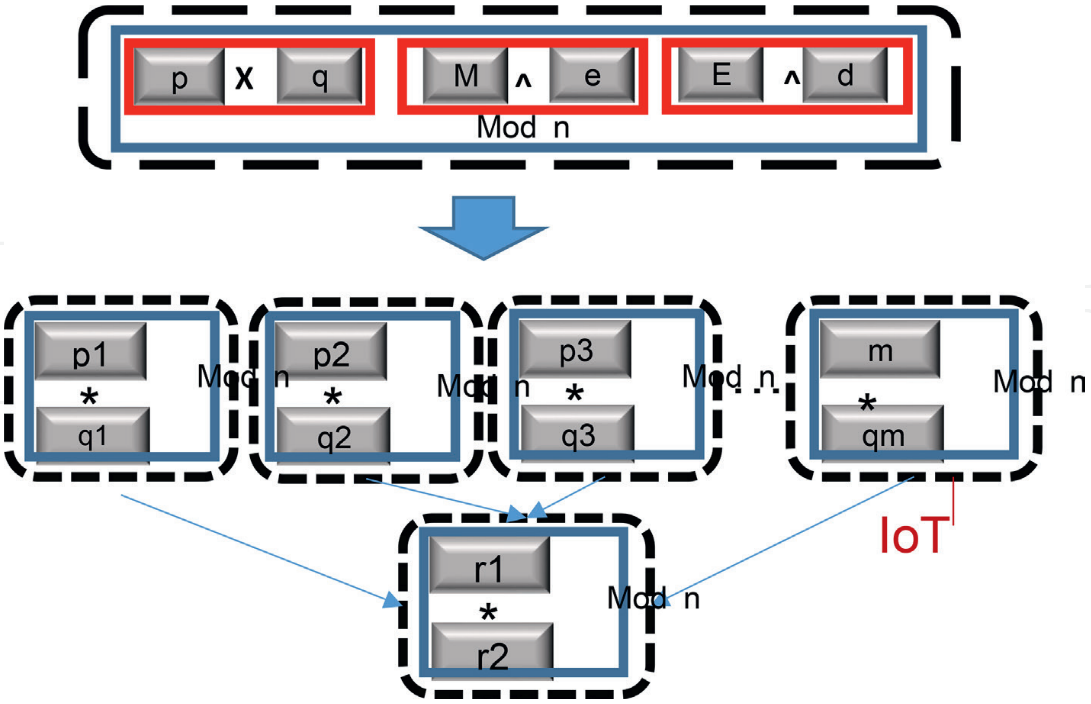

Figure 4.
Splitting and consolidating the modular multiplication into smaller modular multiplication.

Step 1: Preliminary step. The network manager generates a local certificate for each IoT device.

Step 2: Determine who the Primary device is. This operation is done by device #1, which looks for the Primary device on the network, sends a message in broadcast with its certificate to other devices, and declares itself as a master. The other instruments would ignore this message if the device were without a certificate.

Step 3: The Master defines the key-pool size. Device #1 calculates the required pool of keys and the number of keys each device will receive, ensuring an overlap of at least one key between any two devices on this network with a 90% probability. In our case (3 devices), a pool of 152 and 16 keys per device is required.

Step 4: The Master requests the manufacturing of distributed keys. Device #1 (the Master) sends a message in broadcast for all devices to generate keys.

Step 5: Generate distributed keys by each node. Each device generates keys as required. The keys are sent to the Master encrypted with the Master's public key.

Step 6: Distribute the subset keys to each device.

Step 7: Finding a shared key. When a node wants to exchange encrypted messages with another node, it sends a statement with its reference keys. In the experiment, device #2 received a message from device #1 to find a shared key (148).

Step 8: Secure network, Node #2 wants to communicate with node #3 (neither is the Master), and the shared key between them is 42. Node #2 sends a message encrypted by AES using the shared key. Node #3 decrypts it with the same key.

Step 9: Detection of missing devices is employed to discover potential cyberattacks on any network device. The Master (#1) sends each node a ping message every time interval. If there is no answer from a particular device, the current batch of keys is canceled and replaced accordingly.

We executed the experiment step by step, and it went well. We handled several intensive messaging sessions with various key generations and massive messaging.

4. Key sharing using key distribution via a downsized RSA

The most common approach for a secured key distribution is using Asymmetric encryption such as RSA, which executes several modular multiplications for generating the encryption key and for the encryption and decryption stages. A typical IoT device has limited computing resources, which prevents it from executing RSA, compromising key distribution security. In recent related papers, the classic approach replaces RSA with ECC, a similar security level—Saxena and Kawamura [26, 27] proposed parallel processing. Xian-Fu [28] uses GPU achieving a 12% performance improvement. Stergioua [29] execute RSA in Cloud. Goyal [30] recommended ECDH/ECC algorithms. Duy An Ha [5] used ECQV and DTLS, combining authentication and transmission for IoT, and Fadhil and Younis [7] combined multicore CPUs and single-core GPUs.

We propose a downsized RSA implementation in that its results are equivalent to the regular RSA. In this implementation, we split modular multiplications of huge numbers into micro calculations that IOT devices can process. **Figure 4** describes the splitting and consolidating process of a modular N multiplication of two huge numbers, P and Q. Each split component is transmitted to an IoT device to execute it and return the result to the Master device, which consolidates it to provide the required output, encryption key or encrypted/decrypted element.

The processing model comprises the RSA-Distributor and RSA-Observer sub-module.

Figure 5 outlines the RSA-Observer, which is waiting for the RSA-Distributor to send three operads to calculate its modular multiplication, and, when ready, send back its result.

Figure 6 depicts the Distributor's detailed modular multiplication model, from splitting to micro multiplications and distributing it to the available IoT devices for execution. Once all IoT devices' results are accepted, it integrates the detailed results to get the final result, which is then returned as the output of this process.

We conducted an experiment using four connected computers from various manufacturers to prove the model's applicability and assess its effectiveness. The results well support our approach.

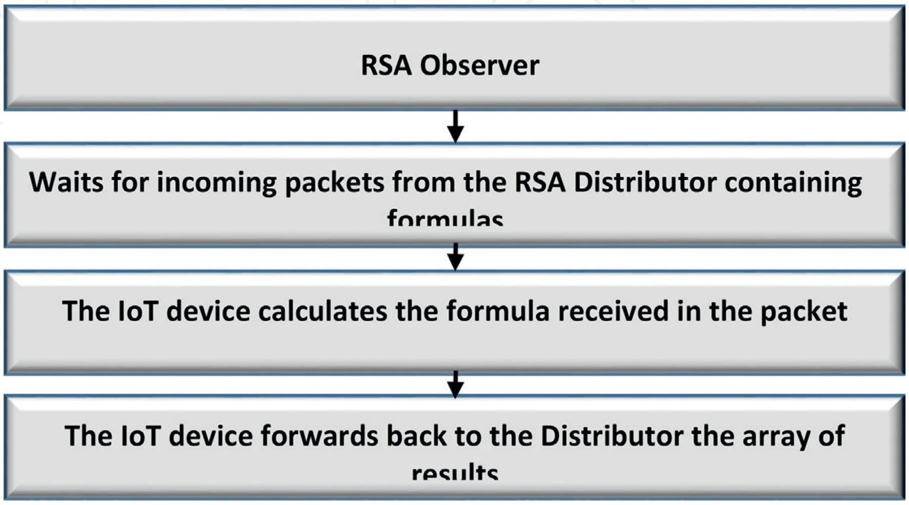
RSA Observer
Waits for incoming packets from the RSA Distributor containing formulas
The IoT device calculates the formula received in the packet
The IoT device forwards back to the Distributor the array of results

Figure 5.
The RSA-observer.

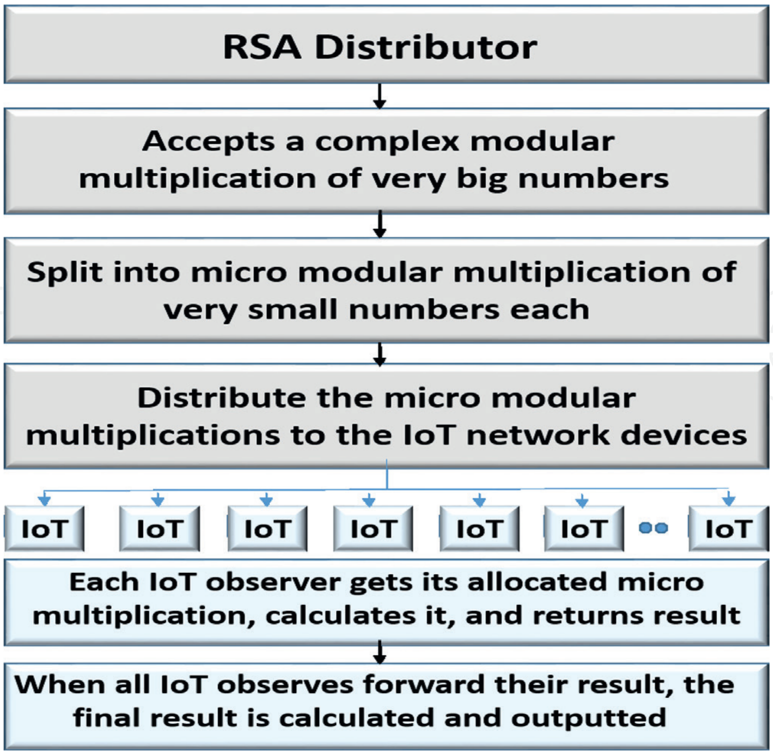
RSA Distributor
Accepts a complex modular multiplication of very big numbers
Split into micro modular multiplication of very small numbers each
Distribute the micro modular multiplications to the IoT network devices
IoT
IoT
IoT
IoT
IoT
IoT
IoT
Each IoT observer gets its allocated micro multiplication, calculates it, and returns result
When all IoT observes forward their result, the final result is calculated and outputted

Figure 6.
The RSA-distributor.

5. Key distribution using a reliable internal CA

The increased interconnectivity and interoperability of previously isolated systems have created new attack paths for hidden adversaries. Integrating IoT technologies has led to new attack opportunities for remote adversaries, mainly due to its poor resistance to cyberattacks. For example, the Parking lot attack occurs when the attacker joins the network and accesses hosts in the internal network. In our case, the first stage begins with determining the Master device to serve as the system controller. The Master is selected before executing the key distribution process. The selection of the Master node is exposed to three attacks: a parking lot attack, exposure of the keys dictionary, and a physical attack. A "parking lot attack" is where a malicious device declares itself as the Master and accordingly controls the keys dictionary and the distribution of keys to all devices. We present a complementary solution to this risk by employing existing methods and technologies to protect the distribution protocol against such attacks. We introduce an internal Certification Authority that issues certificates for each IoT device before joining the network. All keys are distributed by the Master to each device using the Unix OS "password" mechanism. If a device "disappears," all encryption keys are immediately replaced.

Eschenauer and Gligor [1] present a selective distribution scheme and revocation of keys to sensor nodes using probabilistic key sharing among the nodes of a random graph. Alagheband and Aref [20] assess KMS for IoT. Sciancalepore [21] focused on avoiding replay attacks and light authentication using ECC.

Our proposal focuses on a local Certificate Authority, a local keywords Dictionary, and a Detector of Missing Devices.

5.1 Local certificate authority method

For security purposes and to neutralize the Parking lot attack, each device holds a certificate on behalf of the local network manager. For this purpose, we use the "Python Own Certificate Authority (OwnCA)" [10], where OwnCA handles the certificates for hosts, servers, or clients. More CAs, such as Certauth 1.3.0, can be found at https://pypi.org/project/certauth. Benslimane and BenAhmed [24] describe the improved protocol:

1. A preliminary step was added to the protocol, where the local network administrator provides a local certificate for each device connected to the network.

2. All messages exchanged between devices include their certificate to prevent a malicious device from being added to the network.

5.2 Own keywords dictionary method

An IoT device in a local network may be attacked physically, leading to key discovery and network hackery. Therefore, the Primary device sends a mechanism to secure the keys to prevent such incidents. The system operates similarly to the agent that ensures the password and shadow files in the UNIX system. The *keyword* file contains the following data: Device name, Key index number – encrypted data (SHA256), Keys – Encrypted data (SHA256), Last key change time – visible data, and Key Expiration Time – Visible data.

5.3 Detecting missing devices

A daemon is activated in the Master device that checks all active devices to prevent a malicious opponent from "snatching" a device and extracting the information required to hack the network. The Master pings each device to reveal those who do not reply within seconds. After three unanswered tries, the Master eliminates these unanswered devices and changes the keys for all remaining devices in the network.

6. Conclusions

This chapter deals with security issues explicitly raised in IoT networks due to their limited resources and capacity. The conventional way to deal with security issues related to network inter-node messaging is to encrypt the data passing through the network using Symmetric encryption, which requires frequent key replacement and a distribution system. We presented various key distribution methods from the literature and described their operation principles. We elaborated on three advanced techniques our team designed, developed, and experimented them: (i) probability-based Key Sharing exploiting the available IoT computing capabilities to generate keys transmitted to the Master and redistribute them to the nodes ensuring the existence of at least one shared key between any two nodes that may need to interconnect. (ii) Lightweight RSA key delivery utilizing free IoT capacity to execute low-scale modular multiplications required for RSA-secured key distribution. (iii) Internal CA key generation and local distribution. These three methods follow the concept of having the IoT network members internally handle the entire security measurements without needing external servers. Methods (i) and (ii) use the excess computing capacity of the devices in the local network resulting in a win-win situation. Method (iii) requires adding a dedicated machine as the internal CA. Although we presented various solutions to the security challenges in networks characterized by low capabilities, it also applies to any network. It may be a better solution in cases with similar attributes since security issues in IoT and other networks are still amidst cyber security interest and research. To conclude this chapter, we may say that security challenges are still ahead of us; we should first strive for solutions that exploit existing capacity without forcing the embedding of new technologies foreign to the existing setup.

Author details

Menachem Domb
Ashkelon Academy College [AAC], Ashkelon, Israel

*Address all correspondence to: dombmnc@edu.aac.ac.il

References

[1] Eschenauer L, Gligor VD. A key-management scheme for distributed sensor networksnetworks. In: Proceedings of the 9th ACM Conference on Computer and Communications security (CCS '02). USA, NY, Washington DC: ACM; 11 2002. pp. 41-47. DOI: 10.1145/586110.586117

[2] AbuAlghanam O, Qatawneh M, Almobaideen W, Saadeh M. A layered architecture and protocol for key distribution in the context of IoT in smart cities. Journal of Information Security and Applications. 2022;**67**:103173. DOI: 10.1016/j.jisa.2022.103173

[3] Hamid MA, Wadud MA-A, Hassan MM. A key distribution scheme for secure communication in acoustic sensor networks. Future Generation Computer Systems. 2018;**86**:1209-1217

[4] Naoui S et al. Security analysis of existing IoT key management protocols. In: 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA). NY, USA: IEEE; 2016

[5] Metan J, Murthy KNN. N-tier modelling of robust key management for secure data aggregation in wireless sensor network. International Journal of Electrical and Computer Engineering (IJECE). 2019;**9**:2682-2690

[6] Veena RS et al. A cost-effective 2-tier security paradigm to safeguard cloud data with faster authentication. International Journal of Electrical and Computer Engineering (IJECE). 2019;**9**:3833-3842

[7] Fadhil HM, Younis MI. Parallelizing RSA algorithm on multicore CPU and GPU. International Journal of Computer Applications (0975-8887). (Berlin, Germany: Researchgate). 2014;**87**(6):1-8

[8] Goyal TK, Sahula V. Lightweight security algorithm for low power IoT devices. In: International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, India. Piscataway, New Jersey, United States: IEEExplore. 2016. pp. 1725-1729. DOI: 10.1109/ICACCI.2016.7732296

[9] Usman M, Ahmed I, Imran Aslam M, Khan S, Usman UASM. A lightweight encryption algorithm for securing IoT devices. International Journal of Advanced Computer Science and Applications. (The Science and Information (SAI) Organization, Queens, New York. Operated from offices in the United States, the United Kingdom, and India). 2017;**8**(1):1-10

[10] Nandhini P, Vanitha V. A study of lightweight cryptographic algorithms for IoT. International Journal of Innovations & Advancement in Computer Science. (SEMANTIC SCHOLAR, North Northlake Way, Suite 110, Seattle, WA 98103). 2017;**6**(1):1-7

[11] Iqbal U, Bhola J, Jayasudha M, Ahmad MW, Netware R, Yadav AR, et al. ECC-based authenticated key exchange protocol for fog-based IoT networks. Security and Communication Networks. 2022, 2022:15. DOI: 10.1155/2022/7264803

[12] Eldefrawy MH, Pereira N, Gidlund M. Key distribution protocol for industrial internet of things without implicit certificates. IEEE Internet of Things Journal. 2019;**6**(1):906-917. DOI: 10.1109/JIOT.2018.2865212

[13] Lian H, Yang Y, Zhao Y. Efficient and strong symmetric password authenticated key exchange with identity privacy for IoT. In: IEEE Internet of Things Journal. NY, USA: IEEE; 2022. DOI: 10.1109/JIOT.2022.3219524

[14] Salman O, Abdallah S, Elhaji IH, Chehab A, Kayssi A. Identity-based authentication scheme for internet of Things. In: IEEE Symposium on Computers and Communication (ISCC). Messina, Italy, NY, USA: IEEE; 2016. pp. 1109-1111. DOI: 10.1109/ISCC.2016.7543884

[15] Shivraj VL, Rajan MA, Singh M, Balamuralidhar P. One time password authentication scheme based on elliptic curves for internet of things (IoT). In: The 5th IEEE National Symposium on Information Technology: Towards New Smart World. NY, USA: IEEE; 2016. 2015. pp. 1-6

[16] Aman MN, Taneja S, Sikdar B, Chua KC, Alioto M. Token-based security for iot with dynamic energy-quality tradeoff. IEEE Internet of Things, Journal. (NY, USA: IEEE: 2016). 2018

[17] Guo H, Zheng Y, Li X, Li Z, Xia C. Self-healing group key distribution protocol in wireless sensor networks for secure IoT communications. Future Generation Computer Systems. 2018;**89**:713-721. DOI: 10.1016/j.future.2018.07.009

[18] Moharana SR, Jha VK, Satpathy A, Addya SK, Turuk AK, Majhi B. Secure key distribution in IoT cloud networks. In: 2017 Third International Conference on Sensing, Signal Processing and Security (ICS). NY, USA: IEEE; 4 May 2017. pp. 197-202

[19] Othman W, Fuyou M, Xue K, Hawbani A. Physically secure lightweight and privacy-preserving message authentication protocol for VANET in Smart City. IEEE Transactions on Vehicular Technology. 2021;**70**(12):12902-12917. DOI: 10.1109/TVT.2021.3121449

[20] Alagheband, Mahdi R, Aref MR. Dynamic and Secure Key Management Model for Hierarchical Heterogeneous Sensor Networks. UK, Londom: IET Information Security 6.4; 2012. pp. 271-280. DOI: 10.1049/ietifs.2012.0144

[21] Sciancalepore, Piro G, Boggia G, Bianchi G, Capossele A. Key management protocol with implicit certificates for IoT systems. In: Savio Proceedings of the Workshop on IoT Challenges in Mobile and Industrial Systems. Florence, Italy, NY, USA: ACM; 2015. pp. 37-42. ISBN: 9781450335027

[22] Roman R, Alcaraz C, Lopez J, Sklavos N. Key management systems for sensor networks in the context of the internet of things. Computers & Electrical Engineering. 2011;**37**(2):147-159

[23] Wazid M, Das AK, Odelu V. Design of secure user authenticated key management protocol for generic IoT networks. IEEE Internet of Things Journal. 2017, 2018;**5**(1):269-282

[24] Benslimane KBA. Efficient end-to-end secure key management protocol for IoT. International Journal of Electrical and Computer Engineering (IJECE). 2017;**7**(6):3622-3631

[25] Mahmood Z, Ning H, Ghafoor A. A polynomial subset-based efficient multiparty key management system for lightweight device networks. Sensors. 2017;**17**(4):670. DOI: 10.3390/s17040670

[26] Saxena S, Kapoor B. State of the art parallel approaches for RSA public key base cryptosystem. International

Journal on Computational Sciences & Applications (IJCSA). (USA: Cornell University). 2015;**5**(1):81-88. DOI: 10.48550/arXiv.1503.03593

[27] Kawamura S, Koike M, Sano F, Shimbo A. Parallel computation of the generating keys for RSA cryptosystems. Electronics Letters;**32**(15):1365-1366, IEEE Explorer

[28] Xian-FuWong B-MG, Lee W-K, Phan RC-W. Performance evaluation of RSA and NTRU over GPU with Maxwell and Pascal architecture. Journal of Software Networking. 2017;**2017**(1):201-220

[29] Christos Stergioua, Kostas E. Psannisa, Byung-GyuKimb, Brij Guptac, Secure integration of IoT and cloud computing, Elsevier, Future Generation Computer Systems, 78, Part 3, 2018, 964-975

[30] Goyal TK, Sahula V. Lightweight security algorithm for low power IoT devices. In: 11-2016 Conference on Advances in Computing, Communications, and Informatics (ICACCI). NY, USA: IEEE; 2026. DOI: 10.1109/ICACCI.2016.7732296