Analysis of node performance in lightweight key management protocols in IoT networks

LUIS HINOJOSA, University of Twente, The Netherlands

Recently the Internet of Things (IoT) has become increasingly ubiquitous as the development of smart homes and smart cities has gained traction. In view of this, it is concerning that one of the main weaknesses of IoT networks is their security [10]. These networks face resource constraints [7] that make traditional cybersecurity measures impractical due to their computational demands. Given the sensitive nature of data transmitted by IoT devices, the development of lightweight Key Management Protocols (KMPs) tailored for resource-limited IoT nodes has gathered significant interest [1]. However, the field is still in the preliminary stages of research. This paper offers a comparative analysis of two lightweight KMPs from the literature. First, An implementation of a network environment using the raspberry pi testing node is presented. A performance analysis is run and a respective benchmarking of the different key operations in both protocols is presented. We discovered that within a moderately sparse network, the performance of a hybrid KMP[[18]] doesn't match the efficacy observed with a more commonly used protocol that employs solely public key infrastructure-based encryption[11]]. However, we discuss how this is likely to be subject to the network topology.

Additional Key Words and Phrases: asymmetric encryption,key-management, IoT, wireless sensor network, MQTT,cybersecurity.

1 INTRODUCTION

The internet started properly functioning in the early 1980s [13] and now is one of, if not the most widely used technology in the world[5]. Internet technology is present in virtually every aspect of our lives, this is now referred to as *The Internet of People*. On the other hand, multiple manufacturers are now leveraging the Internet and connecting devices to extend their functionalities, this is known as *The Internet of Things* [20]. Devices connected to the internet include but are not limited to smart TVs [21] and other home appliances such as thermostats; such connections allow for remote control of the appliances as well as monitoring functions. Another subset of devices is the so-called *Wearables*, such as smart-watches; this set of devices often collect vital signs from the user and can provide insight into health levels [19], the progress of fitness goals, and even emotional states (the field of psychophysiology uses measures like skin conductance to get insight into a person's emotional state [3]). Knowing the ubiquitous presence of the Internet of Things, it is unsettling to realize that IoT networks more often than not have poor cyber-security and are susceptible to adversarial attacks [12]. If an IoT network is compromised the consequences can range from appliance malfunction to personal data breach to traffic-light system hijacking [14]. However, IoT-constrained capabilities hinder progress in this regard. An IoT network has a varying number of sensor nodes, with the nodes themselves being comprised of homogenous devices. Furthermore depending on the application, and network topology, there can be thousands of nodes [4]. In order to save power, the system nodes have limited processing and storage capabilities [7], as such, common security protocols are unsuitable since they make use of heavy-weight resource-intensive algorithms. Another issue is the lack of scalability of the key management associated with these protocols, given the resources needed for the key operations (i.e. cryptographic key distribution, key loading, and key replacement, among others) [6] as storing and handling the cryptographic keys of thousands of nodes becomes unfeasible.

To overcome the security issues bound to the constrained resources of IoT, multiple works have been developed, giving special attention to key management [15]. These KMPs focus on providing acceptable levels of security while keeping computational costs low with the use of Lightweight Cryptography (LC). LC achieves this by using smaller cipher block sizes, shorter key lengths, and a compact representation of the code compared to traditional methods. While KMPs are still in the preliminary stages of research, they represent a promising step towards improving the security of IoT networks [9].

However, the industry's lightweight protocol standard is yet to be established. Moreover, developers of the IoT systems may have different needs for the KMPs depending on the type of project that is being carried out. As IoT networks are being adopted faster than their security measures, a reference for practitioners to leverage security with resource consumption and scalability of different protocols is needed.

This paper is structured as follows: section 2 presents the research questions, section 3 briefly discusses related works, section 4 explains the methodology to be used to answer the research questions, section 5 discusses the two protocols present in the literature for analysis, section 6 delves into the implementation of the network environment, section 7 presents the experiments and subsequent analysis and section 8 elaborates on our findings, their real-world implications and recommendations for future works.

2 RESEARCH QUESTIONS

Knowing the state of the field in regard to practical usage, our research aims to take the first steps in providing a practical guide for developers to determine what security protocol to implement for their systems. We plan to do this by answering the following research questions:

(1) What is the effectiveness and efficiency of the emerging lightweight key management protocols (KMPs) presented in the literature?
(2) What is the performance of the two studied protocols, more specifically in key generation, distribution, certificate authentication, and service execution time.

TScIT 39, July 17, 2023, Enschede, The Netherlands

To answer the research questions, we delve deeper into lightweight KMPs. We review the latest protocols discussed in the literature. In addition, we briefly discuss the relevance of performance analysis that includes a physical component in the implementation of these protocols. This is critical to ensure that the proposed solutions are not just theoretically sound, but also practical in real-world applications. Our simulation environment allows for the creation of as many nodes as we need and the submission of our physical device to these conditions.

As part of this discussion, we present an implementation using Raspberry Pi as a sensor node. This includes the installation of various state-of-the-art protocols, with an emphasis on benchmarking their performance and analyzing the results. By doing so, we aim to contribute to the ongoing efforts to secure IoT networks and ensure their continued utility in our increasingly digitalized world.

The rest of this paper is structured as follows: section 2 will present the related work, section 3 will discuss the methodology, section 4 will provide the literature review of the protocols that we will be implemented in further sections, section 5 will discuss our implementation of a simulation environment, section 6 will introduce the experiments and results, and section 7 will discuss the results, and finally, section 8 will present the performance analysis.

3 RELATED WORK

While there are several proposed lightweight solutions for securing IoT systems, several of them are only in a preliminary stage, and need more research. Furthermore, there are very few works comparing many different lightweight protocols, and in the works that do, their implementation misses aspects that would be relevant for their deployment. In other words, the results of the current studies are yet to produce the results that are immediately applicable in practice.

For instance, [15] elaborates a comparison of different KMPs for IoT as part of their research, then runs a simulation comparing three of them. They use Arduinos as devices that transmit the protocol-secured data. However, Such a simulation with only 3 sensor nodes may be enough for demonstrating the essence of the different techniques. However, given that the number of nodes can have a significant impact on the system's performance, this may not illustrate the issues that may arise when scaling up the network for real-world applications of IoT.[8] Presents a blockchain-based protocol, where every node is a separated docker container, and Their simulation contemplates different scenarios for different numbers of nodes. However, only the presented protocol is tested without a comparison with other works. Moreover, the testing device is a personal computer rather than a constrained device, which the real-world network would be comprised of.

4 METHODOLOGY

In this section, the steps to be taken to answer the research questions are discussed.

- **On answering research question 1**
  A literature review will be performed to find lightweight security protocols with promising features in terms of speed, computing cost, security, and power consumption. Each work's characteristics will be discussed in terms of the protocol's context, key management, and security performance. The search will be performed using Google Scholar by applying keywords like key management, lightweight, and cryptosystem; furthermore, the results will be filtered to be at most 5 years old. More results will be derived from the initial hit's bibliographies, by assessing what paradigm the KMPs were used (for example, there are many cryptosystems based on a Chaos-theory paradigm).
- **On answering research question 2**
  A physical-based implementation of an IoT network will be set up. This setup will consist of a raspberry pi as a sensor node and a PC acting as the server. Furthermore, the PC will instantiate many simulated nodes. For this, a custom-made simulator will be built, with a modular architecture to be able to reuse the different protocols to be tested. This network will be subjected to the different KMPs and the performance analysis will compare their impact on the raspberry pi's key operations throughput, latency, computing cost, and power consumption. Since sensor nodes are the resource bottleneck of IoT networks, the performance analysis will be limited to the impact of different protocols on them.

5 LITERATURE REVIEW

Many different paradigms for KMPs have been developed in the literature, reflecting the diverse approaches and strategies employed to address the unique challenges of securing IoT networks. IoT faces specific challenges coming from the characteristic of the devices that can comprise a network. Computational constraints come from the need of having numerous sensor nodes perpetually gathering environmental data. As such, the complexity of the computations must be relatively low to be able to run for long periods of time. Very often these nodes have to be small in order to be placed unobtrusively in their environment; their dimensions drive the use of small low-power batteries, which yield a lower power output and shorter battery life. Another signature challenge of the field is that of network heterogeneity. IoT networks frequently are comprised of different sets of devices, for instance, smart homes may have temperature, humidity, touch, and voice sensors. The different devices have different technical requirements and hence different specifications. In the context of KMP, this is a potential source of complications since a given device may have significantly lower capabilities than other groups in the network, rendering it incompatible with the used KMP.In this section, we define a KMP's ability to support device heterogeneity as KMP versatility.

In recent years, numerous studies have focused on proposing innovative KMPs that strike a balance between security, performance, power usage, and versatility. For each paradigm, a promising instance has been selected in terms of its security robustness, performance metrics, and versatility. In this section, we delve into a comprehensive examination of the chosen protocols, highlighting their distinctive features, strengths, and potential limitations. Through this discussion, we aim to provide a comprehensive overview and contrast of the key management paradigms.

Hybrid Symmetric-Asymmetric key management
[17] make use of Smart Objects (SO) as middle-ware between sender and receiver nodes. Originally they frame their KMP in the context of senders as sensor nodes with messages hopping throughout SOs to arrive at another source node. In their paper and preceding work [16] the authors establish a lightweight Key management protocol that makes use of both symmetric and asymmetric encryption. Sending a message consist of a source node, a destination node, and optionally, intermediate nodes. Each message has a data portion where the actual payload that needs sending is, and the routing portion, which has instructions about the trajectory of nodes that the message needs to hop through. The intermediate nodes encrypt/decrypt the routing part of the message and the destination node decrypts the actual data, which was encrypted by the source node. The authors define SOs as devices with superior resources compared to other nodes in the network; this definition aligns with what is commonly defined as an IoT gateway node. These gateways SOs make use of their superior memory and processing capabilities to process the raw data sent by the source nodes and handle the security operation of the protocol, relieving sensor nodes of much of the computational load.

Key management
Every node has a public-private key pair for securing the data, additionally, symmetric keys are shared between every pair of communicating nodes. Three tables are stored in the devices to support the KMP: the Connection table(CT), Global key table(GT), and local key table (LT). CT stores an entry for each pair of linked nodes, a row contains the node name and another field with the corresponding symmetric key. GT has a row per symmetric key, with one column being the key and another column the actual key value. LT has an entry for every interaction of an external node with a given key.
In the implementation shown by [17], SHA256 was used in a hashing function on the random-generated keys every time a new connection was created in CT. In this experiment, the authors also run a registration function previous to the execution of the KMP algorithm. In this registration function, network nodes share their public keys so that the message exchange can be put forth .
In this algorithm, source node A and destination node B are taken as function parameters together with message m. First, the existence of entry CT_{AB} is checked, if such a connection does not exist then A needs to generate a new key, store it in GT_A, and adds B's ID and public key in its CT table (CT_A). To finalize this step, A uses the hash conversion function on its key and sends it to B. If A's key is received and stored properly on CT_B then B sends an acknowledge message and finishes the key setup.
if the connection exists unidirectionally (only in CT_A), A may use B's public key to encrypt a message containing A's hash key. Then B stores the hash key in its LT and A's id in CT_B, establishing a bidirectional connection. if a bidirectional connection is already in place we can say a secure channel is enabled for message exchange. In the event of the message having intermediary nodes, say A,B through B, then the secure channel A,N,B is built by establishing bidirectional connections A,N and N,B.

Characteristics
This protocol is resistant to external attacks since the attacker would have to guess a global key in other to authenticate itself and gain access. The other target of an external adversary would be to try to intercept symmetric keys, but since they are hashed, such an attack would be unfeasible as the adversary would have to guess the hashing function to be used. As a result, the scheme is particularly resistant to attacks like man-in-the-middle and eavesdropping. The drawbacks are that storing GT and LT needs moderately high space, the time needed for communication can be high for a single source and the system is not protected from an internal (compromised) node.

Public Key Infrastructure with ECQV implicit certificates
This work [11] makes use of lightweight ECQV implicit certificates to securely establish a communication channel and reduce the overhead of authenticating devices. This scheme leverages the Public Key Infrastructure and Ephemeral Diffie Hellman Over Close (EDHOC) key exchange. The Ephemeral in EDHOC stands for keys that are changed for every session. Since EDHOC lacks a method for distributing Keys so an Elliptic Curve Qu vanstone (ECQV) certificate scheme is introduced, with a CBOR encoding. Moreover, a complete EDHOC cipher suite is introduced to provide different levels of security, this will be detailed below.

Key management
The proposed framework uses EDHOC with certificates for authentication and key establishment. In this way, it improves the capabilities of PKI with lighter implicit certificates. First, the initiator generates an ephemeral public key, connection identifier (unique between two nodes), and additional data needed for the receiver to correctly process future messages (the security context), such data include the initiator-supported cipher suites, this is sent in a message to the receiver. In the presence of message 1, the receiver ensures that it supports at least one of the cipher suites, and it generates its own ephemeral key pair, shared secret, and its own security context. The receiver agrees on the encoding to be used, then computes MAC based on the mentioned payload. Next, it derives pseudo-random keys using HMAC from the shared secret and encrypts the new security context. Then, this party generates a COSE_Encrypto object using the agreed algorithm from the cipher suite. This object along with the identification and authentication data is encoded in CBOR and sent as a message to the initiator.
After the initiator receives and decodes message 2, it retrieves the agreement of the cipher suite using the connection identifier and decrypts the rest of the message accordingly.
If the previous step succeeds, then this party generates message 3 in the same fashion as message 2, with the difference of generating the COSE_Encrypto object by using a key derived from the content of message 2. This COSE object is sent together with a connection identifier to the responder as a CBOR-encoded message.
Finally, upon the receiver acquiring message 3, it is proven that only the initiator may calculate symmetric random keys for communication. Since the initiator is not sure that the handshake is successful at this point, it stores the keying information until it

receives message 4 from the receiver.

characteristics

This scheme leverages a Public Key Infrastructure (PKI) with the use of implicit certificates for authentication. The authors propose that the implicit certificates are installed by the certificate authority in the safe premises of the manufacturer, this is known as pre-enrollment. Special nodes called border routers possess richer capabilities and mediate the IOT devices with the internet.

One of the major security strengths of the L-ECQV protocol is its robustness against various possible attacks. It provides mutual authentication, perfect forward secrecy, and identity protection. Moreover, even if an attacker physically captures the IoT device, they cannot compute the public key from the certificate, thus securing the session key. In addition, the scheme ensures confidentiality, integrity, and non-repudiation in communication, increasing its overall security.

However, even though the paper's proposal is very robust, it does not fully solve the problem of storing, distributing revoking, and overall managing certificates in networks with large numbers of nodes.

The various Key Management Protocols (KMPs) strive to address the unique challenges posed by IoT networks, balancing security, performance, power usage, and versatility. The hybrid symmetric-asymmetric protocol by [17] leverages smart objects to optimize computational load and resist external attacks, yet it demands moderate storage space and lacks protection against internal attacks. While the Public Key Infrastructure with ECQV implicit certificates by [11] reduces overhead in device authentication with a lightweight certificate scheme. It is evident that no KMP is perfect, but asking for a silver bullet is not a realistic request for any type of system, instead by clearly exposing the different characteristics of each protocol, developers should thoughtfully choose which one is better for the system they are intending to protect.

6 IMPLEMENTATION

In order to perform the subsequent performance analysis, a system implementation is carried out. As discussed earlier, a fair amount of performance analyses have been done in the lightweight KMP literature. [2] uses a table of computation costs from well-known cryptographic operations, these specific values are widely referenced and used throughout the KMP literature, with most of the papers presenting their algorithm computation in terms of the different operations added together. Execution time is not widely presented in the literature, and the preferred way of testing implementations is by simulating a network environment via software with tools like NSE3 and Cooja. However, as evidenced by the work of [17], these types of simulators have limitations in credibility. Our implementation is hardware-based, installing the code for the algorithms in a constrained device, namely the raspberry pi. Since we are focusing on the perspective of the node, the subsequent analysis examines the performance of the physical device. The remaining nodes in the network are simulated and instantiated from the server; every node runs a script to manufacture sensor data that will be sent over messages.

For our implementation, a custom simulator is implemented, based on the work done by [18]. The simulator is written in Python 3.9 and it uses the networkX library to create a graph, where every node is an IoTNode and the edges represent the connection between nodes. This functionality is embodied in the NetworkGenerator class. The Dijkstra class is closely related to NetworkGenerator and IoTNode, this class finds the most efficient route between two graph nodes; routing information is used as the 'control' part of the messages. The nodes connect to one another via MQTT, the Paho-MQTT library is used to implement this communication, with the MQTT broker used being Mosquitto. The broker was set up in a personal computer, with the configuration parameter *allow_anonymous* set to *true* to allow communication within devices connected to the same wifi network to communicate with the broker. Two topics are used for the communication, namely SEND and RECEIVE. Upon registration, every node saves the MQTT topic `SERVER_IOD/IoTNode_ID/SEND` and `SERVER_IOD/IoTNode_ID/RECEIVE` in memory; subscribing and publishing to the respective topics are handled at run-time. In this way every two nodes that want to communicate subscribe to the SEND topic with the receiver's node id. While the Receive topic is used to transfer the corresponding acknowledgment messages.

In the Design phase, the choice is made to assign different classes to handle the different responsibilities of the nodes. A Crypto class contains the Key generation, Storage, loading, and deletion functions besides encrypting/decrypting, hashing, and padding functions; cryptography operations are handled with PYCAS's cryptography library. MqttMessaging handles the topic subscription/publishing needed for exchanging messages between nodes, respective callback functions, and the connections to the broker. The central part of the simulator is the OrchNode class. This sub-class of IoTNode Orchestrates the instantiation of the simulated Network, MQTT broker, and nodes. Furthermore, it runs the corresponding kMP as an algorithm. Additionally, there is an ID handler class; all objects in this environment use it to generate traceable IDs consisting of a combination of class names and a counter.

7 EXPERIMENTS AND RESULTS

7.1 Challenges

As proposed in the research question, a power-consumption analysis was intended to be performed. However, after acquiring and using the device, only a negligible amount of power was detected, so the analysis was discarded. however. It is unclear if this was due to a lack of precision in the device's measurements or a physical problem with the raspberry pi's USB ports.

Also, a problem, with the Paho-Mqtt library was encountered, where when executing the provided callback functions for subscribing to topics, resulted in a node object being erroneously passed on the message parameter. This resulted in some messages being lost, which probably impacted the accuracy of the performance measurements.

For the L-ECQV, a library supporting implicit certificates in Python could not be found, so X.509 certificates with CBOR encoding were used instead.

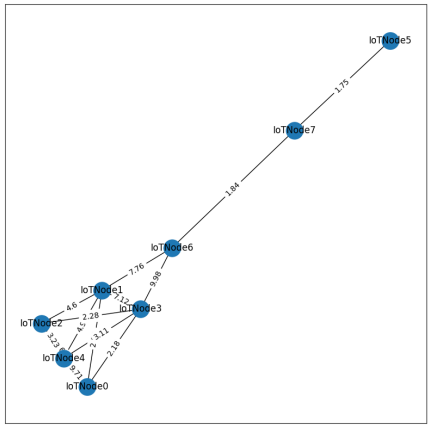

Fig. 1. Sparse incomplete network with 8 nodes

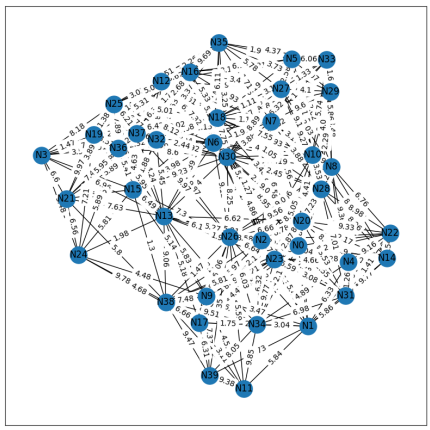

Fig. 2. instance of sparse intermediate-sized network

7.2 Performance analysis

The work presented by [18] presents a hybrid implementation with simulated data and 3 physical devices, which play the role of gateway nodes. We follow this simulation and extend it to be able to digitally add more nodes to the network. This characteristic allows us to have insight into the scalability of the protocol, which is a relevant aspect that can determine which measures to implement in a smaller set-up like a smart home or a larger industrial setting. With the simulation environment in place, children's classes of the original objects are implemented to suit the KMP.
The programs are installed in a raspberry pi 3 Model B Rev 1.4 with Linux 10 the device is. The raspberry pi has a modified version of

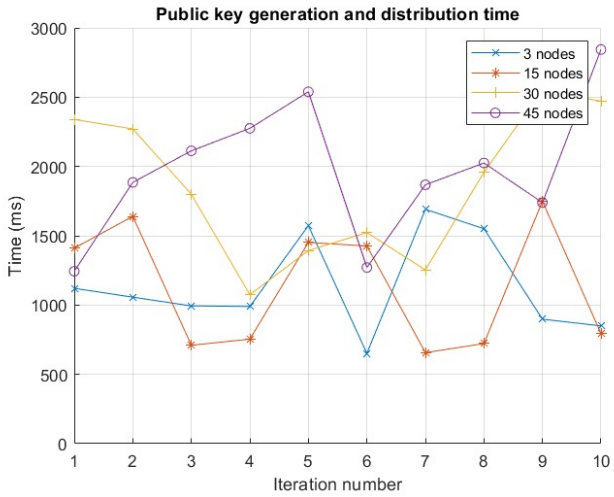

Fig. 3. Measure of latency for key generation and distribution in hybrid protocol.

the classes, so that is running an infinite loop, waiting for incoming messages on the MQTT topic of a fixed node id.
The simulation was run on a laptop with an Intel core i7 9th gen CPU with 6 cores, and 16 GB RAM. the raspberry pi and computer were connected through WLAN with a receive and transmit rate of 72 Mbps and a band of 2.4 GHz. It is worth noting that all the nodes including the raspberry pi communicate with each other exclusively through MQTT.

For the quantitative analysis, we contrast the impact on the performance of the key operations in relationship with the number of nodes in the network. For key generation and distribution, we use 3 different network setups.
For the hybrid symmetric-asymmetric protocol the 4 Setups were tested, A has 3 nodes, resembling the physical implementation presented in [18]. The execution time consists of the initial setup, where the nodes generate, store and exchange their public keys via an insecure channel, directly through MQTT. This step allows the nodes, to freely use the symmetric encryption pathway while protecting their payload from any node other than the source. After running the experiment, we observe that there is a greater variability of latency as the number of nodes goes up, as seen in 3. This could be due to the number of nodes that lack direct connections to other nodes, resulting in messages having to jump through a larger number of nodes to reach their destination. For our testing, we set the probability of lacking a connection edge to 0.30, resembling a moderately sparse network.

For the L-ECQV protocol, the X.509 certificate authentication would be an analogous operation to the public key exchange. The execution time consists of public key extraction, CSR creation, certificate distribution, and authentication. The certificate transmits latency and signing is not taken into account as the role of Border Router is taken over by the computer running the simulation rather than any of the nodes. This is similar to the measurement done by

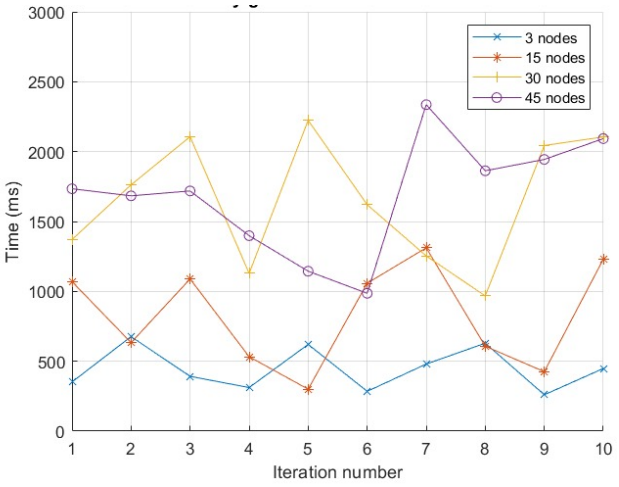

Fig. 4. measure of latency for certificate validation

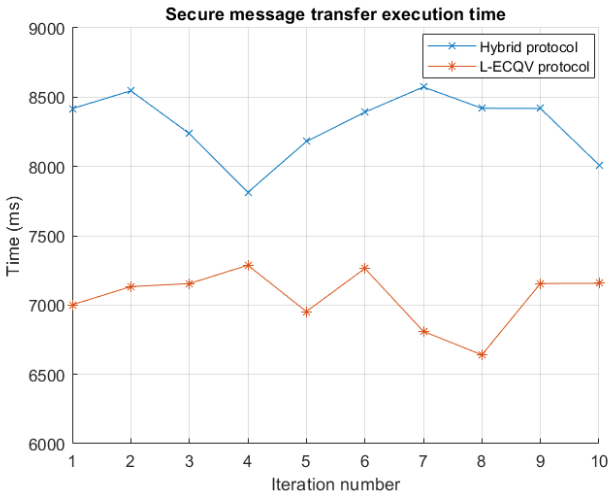

Fig. 5. Comparison of execution time for secure message transfer

[11] referred to as 'certificate validation time', however, it differs in that we are measuring the amount of time that it takes for all the nodes in the network to execute this step. In 4 we can see that overall the latency remains more stable in this protocol, however, this can be influenced by the fact that within most clusters of nodes, virtually all had a connection with the Border Router, which could serve as a mediating node for any pair of source and receiver devices.

For both protocols, We can see a considerable variability of latency. This would depend on network latency and the load of the raspberry pi.

For both protocols, we measured the time it would take to transfer a secure message, in other words, once the initial setup stage is completed and the nodes in the network are ready to transfer data, they need to engage in different operations to establish a secure channel, encrypt the messages, transmit them, and decrypt them on the other side. for this measurement, we used a fixed message length of 40 bytes. we can see the results in **??**

We can see that the L-ECQV protocol performed noticeably better in this regard. This is expected as the hybrid protocol besides using asymmetric encryption on the source has to generate symmetric hashkeys in between if it has not established a secure connection with the receiver node. the measurement was done by placing timestamps right before the message was decrypted and after getting an acknowledgment that the message was decrypted.

7.3 discussion and limitations

as discussed above, we can see that overall the L-ECQV method performed better in our network environment. Is worth noting that this is affected by the fact that the L-ECQV relies on pre-installing certificates, which is reflected by the node classes computing their certificates in the pre-set-up stage before any measurement was done, whereas the hybrid method executes at least an additional symmetric encryption in the every communication scenario. Another relevant factor is the network setup, with only a moderate degree of sparsity, the hybrid protocol does not use often the faster symmetric paradigm more times than the asymmetric paradigm. In a network where most nodes have a connection with each other, this protocol is very likely to perform worse.

Unfortunately, it was not possible to run an experiment where the sparsity of the network increased as the code for this setup could not be stabilized to run reliably due to time constraints. It is noticeable that the latency of the operations is considerably higher than in their original work. This could be due to different factors: first, the main KMP algorithm did not run using threads, which limited the number of nodes engaging in the protocol at any given time. Second, the network speed was found to be relatively slow. This is due to the fact that for development purposes, the connection was established through a mobile hotspot, to ensure that all devices connect to the same IP address. Sadly this was realized after the measurements were done and a new setup could not be established due to time constraints.

8 CONCLUSIONS

This research presented a comparative performance analysis of two lightweight Key Management Protocols (KMPs) for Internet of Things (IoT) networks: a hybrid symmetric-asymmetric protocol and the L-ECQV protocol. Our investigations were conducted under a fixed sparsity and varying network sizes. Our results demonstrate distinct performance characteristics for the two protocols. The hybrid protocol displayed increased variability in latency as the number of nodes escalated, likely due to the higher number of nodes lacking direct connections to other nodes in our moderately sparse network setup. In contrast, the L-ECQV protocol maintained more stable latency even as the network size increased, likely due to the prevalent connection between the CA and most of the nodes. The real-world implications of these findings are significant. The choice between these two KMPs or more generally, paradigms of KMP needs to be informed by the specific requirements and conditions of each individual IoT network. In IoT networks where stability and predictability of latency are critical, such as healthcare applications, traffic management systems, or industrial automation, the L-ECQV

protocol may be a more suitable choice. On the other hand, in applications where network sparsity is high and direct connections between nodes are limited, a hybrid symmetric-asymmetric protocol might offer more flexibility, despite its increased variability in latency. Furthermore, our results highlight the importance of taking network topology into consideration when choosing a KMP for an IoT network. Depending on the specific network configuration and the nature of IoT devices in use, different KMPs could yield different performance results.//// It is worth mentioning that, due to the challenges and limitations of this research, the findings, although insightful are not completely reliable. In conclusion, our study gives insight into the importance of the choice of KMP in IoT network performance and emphasizes the need for practical-orientated insight in the field, focusing on real-world IoT use cases and varying network conditions.

8.1 recommendations

The research questions of this paper bring significant value to the field. While this paper brings a comparative analysis that reflects the real qualities of the different protocols, the measurements are not completely reliable. But given the importance of the tackled problem, it is recommended that in future works the limitations of the research are addressed. Namely working with threads or docker containers to more accurately simulate the behavior of nodes simultaneously engaging in the given protocol. implementing networks with high sparsity, to be able to study the protocols in the varying scenarios that IoT networks operate in. Once these aspects are in place, studying a greater number of protocols would make more meaningful research.

REFERENCES

[1] Aishah Abdullah, Reem Hamad, Mada Abdulrahman, Hanan Moala, and Salim Elkhediri. 2019. CyberSecurity: A Review of Internet of Things (IoT) Security Issues, Challenges and Techniques. In 2019 2nd International Conference on Computer Applications Information Security (ICCAIS). 1–6. https://doi.org/10.1109/CAIS.2019.8769560

[2] Mohammad Abdussami, Ruhul Amin, and Satyanarayana Vollala. 2022. LASSI: a lightweight authenticated key agreement protocol for fog-enabled IoT deployment. International Journal of Information Security 21, 6 (2022), 1373–1387. https://doi.org/10.1007/s10207-022-00619-1

[3] W. Boucsein. 2001. Electrodermal Activity (2nd. ed.). Springer Science and Business Media, New York, NY. https://doi.org/10.1007/978-1-4614-1126-0

[4] Danco Davcev, Kosta Mitreski, Stefan Trajkovic, Viktor Nikolovski, and Nikola Koteli. 2018. IoT agriculture system based on LoRaWAN. In 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS). 1–4. https://doi.org/10.1109/WFCS.2018.8402368

[5] Tarik Eltaeib. 2015. TCP/IP PROTOCOL LAYERING. 3 (01 2015), 415–417.

[6] M. Eltoweissy, M. Moharrum, and R. Mukkamala. 2006. Dynamic key management in sensor networks. IEEE Communications Magazine 44, 4 (2006), 122–130. https://doi.org/10.1109/MCOM.2006.1632659

[7] Asma Haroon, Munam Ali Shah, Yousra Asim, Wajeeha Naeem, Muhammad Kamran, and Qaisar Javaid. 2016. Constraints in the IoT: the world in 2020 and beyond. International Journal of Advanced Computer Science and Applications 7, 11 (2016).

[8] Mohamed Ali Kandi, Djamel Eddine Kouicem, Hicham Lakhlef, Abdelmadjid Bouabdallah, and Yacine Challal. 2020. A Blockchain-based Key Management Protocol for Secure Device-to-Device Communication in the Internet of Things. In 19th IEEE International Conference On Trust, Security and Privacy In Computing And Communications (TrustCom 2020). Guangzhou, China, 1868–1873. https://doi.org/10.1109/TrustCom50675.2020.00255

[9] Dae-Hwi Lee and Im-Yeong Lee. 2020. A Lightweight Authentication and Key Agreement Schemes for IoT Environments. Sensors 20, 18 (2020), 5350. https://doi.org/10.3390/s20185350

[10] Rajesh S M and Prabha R. 2023. Lightweight Cryptographic Approach to Address the Security Issues in Intelligent Applications: A Survey. In 2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT). 122–128. https://doi.org/10.1109/IDCIoT56793.2023.10053412

[11] Manisha Malik, Kamaldeep, Maitreyee Dutta, and Jorge Granjal. 2023. L-ECQV: Lightweight ECQV Implicit Certificates for Authentication in the Internet of Things. IEEE Access 11 (2023), 35517–35540. https://doi.org/10.1109/ACCESS.2023.3261666

[12] Mukrimah Nawir, Amiza Amir, Naimah Yaakob, and Ong Bi Lynn. 2016. Internet of Things (IoT): Taxonomy of security attacks. In 2016 3rd International Conference on Electronic Design (ICED). 321–326. https://doi.org/10.1109/ICED.2016.7804660

[13] University of Georgia Regent Board. ". A Brief History of the Internet. Journal Name ", " ("), 415–417. https://doi.org/''

[14] 2020. Dutch Hackers Found a Simple Way to Mess With Traffic Lights. 39 (August 2020).

[15] Tamanna Tabassum, SK Alamgir Hossain, and Anisur Rahman. 2020. An Efficient Key Management Technique for the Internet of Things. MDPI Sensors (April 2020). https://doi.org/10.3390/s20072049

[16] Tamanna Tabassum, SK Alamgir Hossain, and Md. Anisur Rahman. 2019. A Secure Key Management Technique Through Distributed Middleware for the Internet of Things. In Intelligent Computing, Kohei Arai, Supriya Kapoor, and Rahul Bhatia (Eds.). Springer International Publishing, Cham, 1128–1139.

[17] Tamanna Tabassum, Sheikh Kashem Azad Hossain, Md Abdur Rahman, Mohammad Fairuz Alhamid, and Mohammad Alamgir Hossain. 2020. An Efficient Key Management Technique for the Internet of Things. Sensors (Basel, Switzerland) 20, 7 (2020), 2049. https://doi.org/10.3390/s20072049

[18] Tasnia Tabassum, Sk Md Mizanur Hossain, Md Atiqur Rahman, Mohammad Fahad Alhamid, and Md Anwar Hossain. 2020. An Efficient Key Management Technique for the Internet of Things. Sensors (Basel, Switzerland) 20, 7 (2020), 2049. https://doi.org/10.3390/s20072049

[19] J. Wei. 2014. How Wearables Intersect with the Cloud and the Internet of Things: Considerations for the developers of wearables. IEEE Consumer Electronics Magazine 3, 3 (2014), 53.

[20] Andrew Whitmore, Anand Agarwal, and Li Da Xu. 2015. The Internet of Things–A survey of topics and trends. Information Systems Frontiers 17, 2 (2015), 261–274. https://doi.org/10.1007/s10796-014-9489-2

[21] Ryan Williams, Emma McMahon, Sagar Samtani, Mark Patton, and Hsinchun Chen. 2017. Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach. In 2017 IEEE International Conference on Intelligence and Security Informatics (ISI). 179–181. https://doi.org/10.1109/ISI.2017.8004904