JCST Journal of Data Acquisition and Processing
ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.7645330

INVESTIGATION ON HYBRID SECURITY TECHNIQUE FOR IOT

Pardeep Kumar
Research Scholar, IKGPTU Jalandhar

Dr. Amit Gupta
Assistant Professor, IKGPTU Jalandhar

Abstract:
It is crucial to create robust strategies for protecting Internet of Things gadgets. There is a risk that cyberattacks could compromise IoT devices. This justifies the need for their safety. They will be safe from harm if they have a solid identity. For instance, the gadget should only talk to the systems it manages, and its information should be kept in a safe place. This information needs to be encrypted the minute it is transmitted and stored. To prevent malicious actors from gaining access to private data, the device must be encrypted by an external entity. None of these things have been standardised yet. In addition, the use of more lightweight cryptographic systems is required because big data sources, such as sensors placed in IoT applications, produce enormous data. Diffie Hellman became an effective method for secure and lightweight data exchange for linked devices in IoT settings after it was combined with Elliptic Curve (EC-DH). Even with these enhancements, there are still security holes in the Internet of Things. We propose a Hybrid Technique that improves security in IoT applications as a solution to this issue. We analysed a real-world example of anomaly detection in the medical field. Protocols like MQTT and XMPP are set up with the Adafruit IO cloud and the Watson IoT platform for the experiments. Several experiments validated the scheme's efficacy, demonstrating its ability to increase security while simultaneously improving key characteristics associated with the Internet of Things (uploading, downloading, encrypting, and decrypting). The proposed approach is useful for real-time IoT applications.

Keywords: Internet of things (IoT), IoT Security, Hybrid Approach. AI, Machine Learning

I. INTRODUCTION

Changes in the dissemination of knowledge have resulted with the advent of the Internet. The greater reliance on technology is directly related to the proliferation of Internet of Things devices. Because of this, there is now a higher risk of being hacked. When compared to more conventional technology, IoT devices are more easier to breach. Some examples of applications are [1, 2] automated buildings, cities, industries, residences, and agriculture. Organizations need to think about IoT Security and how to safeguard connected devices as the number of these devices continues to grow. Firewalls and antivirus programmes are two examples of typical cybersecurity tools that can aid in protecting connected devices, but they may not be the best option. The technology utilised by IoT devices is often not capable of processing the necessary security measures, so these technologies often fail to appropriately safeguard them. As a result, businesses need to create a comprehensive cybersecurity strategy to protect against numerous types of cyberattacks. [3, 4]. Organizations should make protecting IoT devices a key priority. Internet of Things devices need to be protected not just from the outside world but also from each other by being placed behind a firewall. This separation can reduce vulnerability to threats from networked gadgets. An all-encompassing security plan is required for the safety of IoT devices. Perimeter network firewalls prevent malicious actors from penetrating a network and gaining access to a system. When it comes to the Internet of Things, device security is paramount. Notifications of illegal access to sensitive information should be made available to staff members by IoT Security. Multiple security concerns have been raised over the IoT. There are three types of these types of issues: methodological, organisational, and technical. Data flow, distributed system types, physical constraints, resource limits, data security, and an understanding of the IoT's design all fall under the third area, technical [5].

So, businesses need to take preventative measures to safeguard their networks. While certain safeguards are tailored to counteract one particular type of danger, broader approaches may be available as well. In any event, they should create a thorough security strategy to safeguard their IoT devices.

The fact that the devices themselves are not always secure connections between them is a major issue for IoT security. Moreover, malicious actors can intercept communications and steal sensitive information. The good news is that there are several approaches to fixing this issue. Possible answers include various forms of blockchain technology, AI, and CAPTCHAs. Smart devices are the key to ensuring the continued safety of the Internet of Things. In the meanwhile, businesses should take the initiative to set up a solid security system.

Access control is another crucial aspect of IoT safety. The goal of access control is to limit system use to authorised personnel only. Companies can protect themselves from the risks associated with credential breach and the misuse of Internet of Things (IoT) devices by enforcing stringent security measures. The disclosure of credentials is another potential security risk that can be reduced with a strong access management policy. Aside from that, it is crucial to enforce user profiles and log all access.

Protecting Internet of Things devices from unauthorised access requires isolating them. This could mean, for instance, that they can't share a network with the rest of the internet. There is potential for harm here. Malware susceptibility is a strong indicator of the device's susceptibility to cyberattack. For the sake of both data and privacy, it is crucial that IoT devices be secured. In both cases, the possibility of a hack or data breach is low. Security and virus protection are paramount for IoT devices.

Internet of Things devices can be protected in a variety of methods. Most IoT devices have a number of features that can be exploited by hackers. Configuring them with robust security is essential to shield them from threats. Disabling unnecessary functionality is one of the best methods to keep these IoT gadgets safe. DNS filtering, meanwhile, can block users from visiting harmful websites. With this method, a single computer can connect securely to a server's network.

In order to keep unwanted hands out of your IoT gadgets, you need keep them separate. Also, they need to be safe from any outside attempts at hacking. Protected and tamper-proof Internet of Things devices are a must. Companies can safeguard their data against theft by restricting the number of Internet of Things devices that can connect to the network at once. They also need to be safe from prying eyes that might steal the information they hold. IoT security measures should additionally check if malware hasn't been installed on the device.

The security of IoT devices has not advanced much, according to [6]. Both its security issues [9–10] and its frameworks [7, 8] addressed multiple facets of IoT security. Traditional RSA is unsuitable for Internet of Things devices because to the complexity and weight of its mechanism. It's defenceless to Replay or MIM attacks. An enhanced ECDH-based security technique for IoT systems is presented here. This is where we pitch in.

First, Hybrid ECDH is a secure cryptographic approach that is immune to man-in-the-middle (MITM) and replay attacks.

IBM Cloud, MQTT, and Adafruit IO are just a few examples of cloud systems that make use of Python.

Third, a performance boost over existing methods is established by experimentation.

This section of the paper is structured as follows. The literature on IoT security is reviewed in Section 2. The fascinating case study and problem statement can be found in Section 3. In Section 4, we suggest a less invasive method of protecting the IoT from harm. The application of the case study is discussed in Section 5. The experimental setup is described in Section 6. In Section 7, we show the results of our end-to-end security testing with an Internet of Things (IoT) use case.

.

II. RELATED WORK

The interconnected nature of IoT devices makes protecting them difficult. A systematic approach to IoT security was proposed by the authors of a recent study [12]. It was found that domains with Internet of Things support have their own norms. There is not one single cause or circumstance that always results in a vulnerability. The lack of security standards and the integration of multiple technologies that make up the IoT increase their susceptibility to attack. The Internet of Things (IoT) is integrated with blockchain-based distributed ledger technologies in [13]. One-to-one communication is crucial in user-specific IoT applications with many connected devices since IoT enables M2M correspondence [14]. Computer networks, mechanical and industrial controls, and transportation networks are all vulnerable to cyberattacks, the severity of which depends on the sophistication of the attack. Terrorists frequently strike against natural gas pipelines and power plants. The transportation timetable of a nation can also be thrown off by a successful cyberattack. Cyberattacks can inflict extensive damage in addition to financial losses. Many examples of cybercrime can be traced back to ideological, political, or even personal grievances. [15-16].

Securing the Internet of Things is difficult in the modern digital environment. As the sophistication of Internet-enabled gadgets increases, so too must the frequency with which security protocols be modernised. Legislation was been passed at the federal level with the express purpose of improving security measures for Internet of Things gadgets. Although the new law is directed solely at government entities, it will certainly have far-reaching effects in the commercial sector. Device manufacturers will need to adopt these guidelines to guarantee their products are up to par with current standards. There are multiple levels of support needed for IoT devices[17-20]. Connecting to a gateway device, such as a smartphone, is a standard part of the Internet of Things' communication architecture for gaining access to the cloud. The local gateway device in this IoT system translates between the device's native protocol, translates data, and ensures the device's safety. When a local gateway is set up to work with an Internet of Things device, it becomes an integral part of the system that can be relied upon. It has the ability to interact with the cloud service and deliver data and applications required by the Internet of Things. [21, 22]. Many low-overhead methods exist for Internet of Things software. Methods like the Constrained Application Protocol (CoAP) are discussed in detail in [23]. Diffie-Hellman and Elliptic Curve Digital Signatures (ECDH) are used for the bandwidth-friendly key exchange. However, they lead to man-in-the-middle assaults. In this research, Hybrid Techniques are used to solve the security problems associated with low-overhead protocols like ECDH.

III. PROBLEM DEFINITION AND SETUP

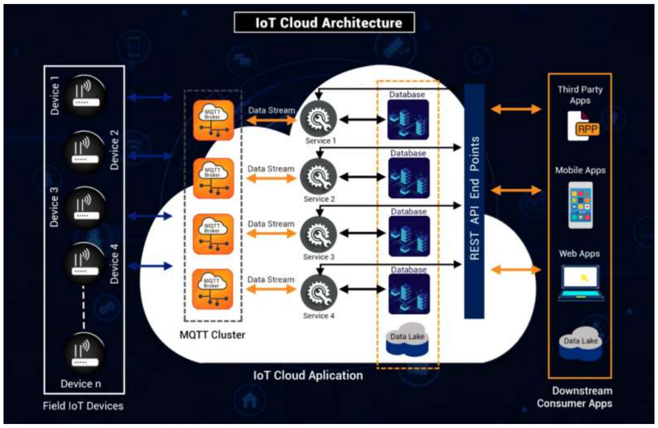

Fig. 1 Architecture of IoT application

As can be seen in Fig. 1, the IoT device can talk to an IoT platform like Adafruit IO using protocols like MQTT. Because of its low resource requirements, the MQTT protocol is well suited for use in Internet of Things (IoT) messaging applications involving low-powered devices. The protocol is also made for communication systems with high latency and jitter. For communications between machines (M2M) that don't involve a human, this is an excellent choice. In a real-world, out-of-the-way context, M2M scenarios are indispensable. In spite of its low bandwidth and resource requirements, MQTT delivers satisfactory performance. Other Cloud Platforms, such as IBM's cloud platform, Google Cloud, and Microsoft Azure, provide access to all the required infrastructure and computing resources on demand. Big data analytics, Big data, relational storage, and objects are all supported, as is the Internet of Things (IoT) middleware.

Despite its usefulness, IoT technology has spawned a host of security concerns. Threats to data integrity can come in many forms, including attacks on devices, application services, networks, and online interfaces. Device assaults are carried out to compromise Internet of Things devices. Application service attacks compromise IoT-related system applications. Threats to the network can compromise the capacity of IoT devices to communicate with one another.

Attacks on the web interface can take several forms, such as cross-site scripting, SQL injection, and cross-site reference forgery. When an adversary makes unauthorised changes to data, known as a data integrity assault [15], the data's original state is no longer guaranteed. These attacks need to be considered, and swift security measures must be put in place. The following data are included in this paper:
Superior to more traditional Internet of Things security methods is a hybrid approach.

3.1 RSAAlgorithm

Based on the fact that it is mathematically difficult to determine the factors of a huge composite number, RSA is a public-key cryptography algorithm. One method of encrypting information to keep it safe from hackers is the Rivest-Shamir-Adleman algorithm. You can think of it as a "key pair generator" as well. In this article, we'll take a quick look at the philosophy behind it all. Both a public and private key are required to use the RSA cryptographic algorithm. The public key is used to encrypt messages and is accessible to everyone. Only the owner knows the private key's secret code, and no one else can read it. Two integers, n and e, stand in for the private key, while d and e stand in for the public. There is a common misconception that the public key is the private one, however it could actually be a portion of the private key. Both the public and inverse RSA functions rely on very large random prime numbers, making them challenging to calculate. [21-22]

3.2 Diffie–Hellman

Key exchanges are made easier and lighter thanks to the Diffie-Hellman method, which was developed in response to the complexity and heaviness of Public Key Infrastructure (PKI). Instead of exchanging keys, the two parties just exchange an automatically created secret key. The Diffie-Hellman protocol is a means through which a common key can be created. It relies on passing around public keys to communicate. If you are the lucky recipient of someone else's public key, you can increase the value of their private key. Then, the same number, 'z,' is calculated and transmitted to Alice. What could not be better for use in an encryption scheme as a key. To reduce the burden of the security plan, the DH system is used, as shown in Figure 2. As a result, the communicating parties are able to calculate the secret key. While promising, the plan does have its restrictions. It is less effective to use a DH-based asymmetric key. In addition, it is not a valid option for use in digital signatures.

3.3 ECDHAlgorithm

ECDH is an Elliptic Curve Security Algorithm that is used in secure messaging and cryptography. This type of encryption is based on a secret key that is shared between two parties. It can be used to secure data exchange over a non-secure channel. ECDH is superior to other similar algorithms, such as public key peers, and is also very simple to use. Moreover, it is a very effective way to exchange private and public keys.

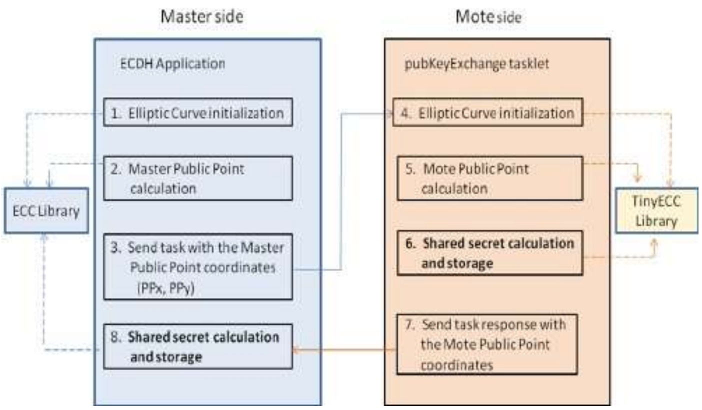
Master side
Mote side
ECDH Application
1. Elliptic Curve initialization
2. Master Public Point calculation
3. Send task with the Master Public Point coordinates (PPx, PPy)
8. Shared secret calculation and storage
ECC Library
pubKeyExchange tasklet
4. Elliptic Curve initialization
5. Mote Public Point calculation
6. Shared secret calculation and storage
7. Send task response with the Mote Public Point coordinates
TinyECC Library

Fig. 2 Shows key exchange with DH scheme

It is theoretically capable of withstanding a Man In The Middle (MITM) assault. However, it remains vulnerable to MITM attacks because to the non-dynamic distribution of public keys.

3.4 Malware Node Classification

It is the classification of malware nodes that best defines the malicious behaviour of a file system. In-degree, centrality, and cohesiveness are indicators of a network's degree of fragmentation. Malware can be categorised based on its system call graph utilising these network characteristics. To quantify malware's in-degree, we divide the number of system calls with an in-degree of one by the number of calls with an in-degree of one to one.

Size-based categories define the ingress and egress of malware samples. The malware's in-degree is its most prominent trait, whereas the out-degree is less prominent. More weight is given to the in-degree than the out-degree. The effectiveness of a malware node's classification is significantly impacted by its mean in-degree centrality and the weight of its in-degree. The classifier of a malicious node is highly influenced by these characteristics.

Malware classification relies on the other characteristics as well. Both in- and out-degrees are crucial when describing a given sample. Formally, degrees in an in-degree system are equivalent. When compared to an in-degree, a weighted in-degree is same. In determining the category of a malware node, these are the two most crucial aspects to consider. Malware classification relies on these three characteristics, however they cannot be used alone to identify a sample's origin.

3.5 Switching Off of Malware Node

The need of protecting edge devices from malicious software is growing. Existing software-based malware detectors can do in-depth malware analysis, but they are not portable to edge devices. Malware detectors that use static signatures to identify fresh samples do so by comparing them to patterns stored in already-existing databases. These might be easily prevented with well-written malware code. Dynamic malware detectors, on the other hand, watch for changes in software and hardware settings while the infection is running. Although this method eliminates the problems caused by static alternatives, it often comes with hefty implementation costs. [37].

We improve upon a previously described method for inferring applications in order to develop a detector for recognising possibly harmful applications. As a next step, we deployed our malware detector model to cloud servers for examination. The rapid growth of AI has resulted in a plethora of new uses, one of which is in the identification of viruses. Significant progress is being made in the area of using machine learning to model malware based on its features and even interpret its behaviour. Using software and hardware abstraction layers, malicious behaviour can be characterised. The microarchitectural components generated by system software include, among other things, branch misses and cache hits [3]. Other examples include system calls, application programming interfaces (APIs), and network activity within software. [4]. Side-channel leakage caused by electricity and electromagnetic emissions communicates the behaviour of a programme at the instruction level granularity (EM). In [9], for instance, the authors showed how EM-emissions can be used to characterise application behaviour and to reveal hidden execution paths in programmes.

Models for machine learning: A. Verification and Creation The machine learning technique originally developed for app inference is applied to the problem of finding malicious software. The DVFS status time series of the processor is related to the application or applications scheduled to run on the core. Therefore, DVFS state transitions can tell the difference between safe and harmful software.

IV. EXPERIMENTAL SETUP

A. The dataset consists of a set of packaged Android malware applications as well as a set of benign apps.

1) The benign dataset, as mentioned in Section III, comprises of Android benchmark applications. There are a total of 14 Android applications. The Drebin dataset was used to collect malware samples [16]. There are a total of 179 malware families. While harmless programmes are used as benchmarks, the approach may be used to any Android application. To establish a roughly balanced sample, we picked just 16 Android malware programmes, each belonging to a unique family.

2) Malware programmes have both benign and malicious components. The training does not include a detrimental component; rather, the classifier sees both benign and malicious code. As a result, the complete DVFS state time series has been classified as malware. As noted in [38], different malwares employ a variety of activation mechanisms and payloads. While it is widely acknowledged that the power consumption of these applications varies, we are interested in the runtime interactions of android programmes with the processor's DVFS states. The collection includes 14 non-malicious benchmark programmes and 16 malicious programmes. During training, each android application in the dataset is run for ten seconds in the presence of the system's default applications. As previously said, training is not restricted in any way by one's underlying circumstances..

Additionally, numerous instances of DVFS state traces are collected for each benign and malicious software in the dataset. This ensures that differences arising from varying background circumstances are accounted for. We set the number of training instances per application to 75 in our experiments.

V. IoT Application

A real-world Internet of Things architecture is used to show secure connectivity and data analytics. An IoT device simulator is developed after enrolling on the IBM Watson IoT platform. In order to connect the IoT platform to the IoT, MQTT is utilised. MQTT is an ISO and OASIS standard. Mobile sensor data, such as gyroscope and accelerometer sensors used to measure patient movement, are routinely sent via MQTT. The protocol is based on the TCP/IP connection-oriented protocol suite. Cloudant NoSQL database of your choice is used to store historical data. Watson Studio extracts and prepares data from databases for analysis. A Python script analyses the data and reports any finds or abnormalities. These discrepancies are indicated for end users. Users can make informed judgments based on historical sensor data irregularities. It makes a connection to the CloudantNoSQL database, which stores data from IoT devices connected to the Watson IoT platform through the Gyroscope and Accelerometer sensors. The last line of code in Listing 1 is critical, as the account being accessed is not funded. It can only handle a specific amount of queries per second, and surpassing that limit results in a "error message."

5.1 Experimentation

Because IoT applications and sensor devices create large volumes of data, PySpark1 is used. With the IoT, the sensor network is the source of huge amounts of data (big data). IoT apps are built using the IBM Watson IoT platformf2. Sensor data is stored in Cloudant NoSQL DB3. Watson Studio is used to examine IoT data. The cloud platform's base is IBM cloud4. The protocol used is MQTT5.

VI. Experimental Results

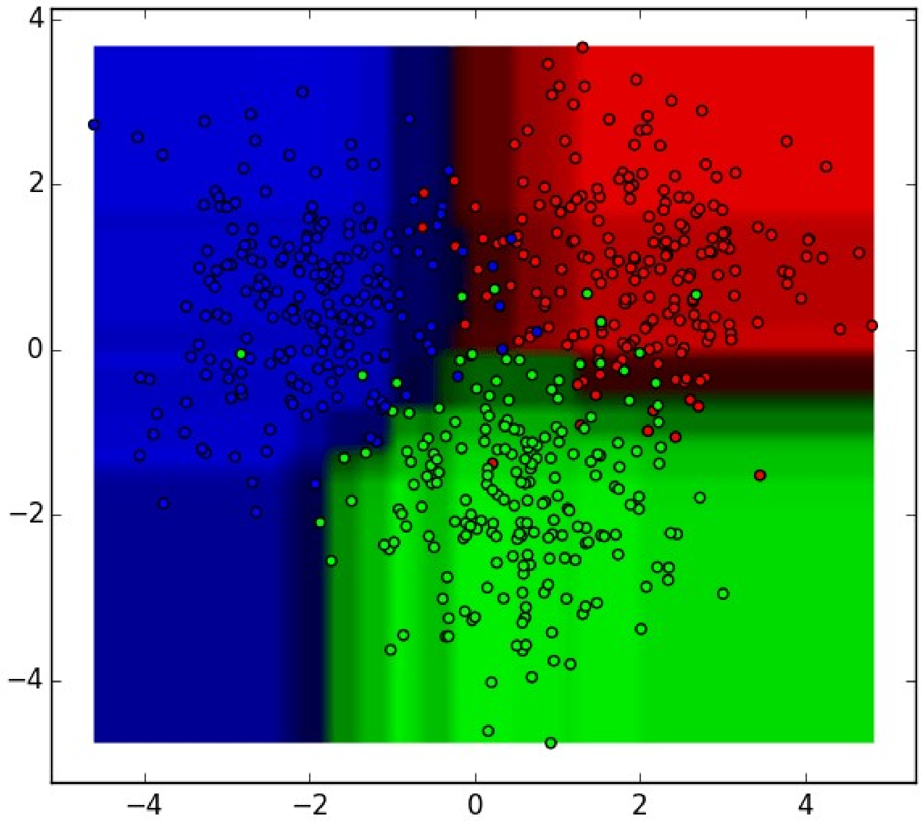

Fig. 4 Decision Tree Classifier Results

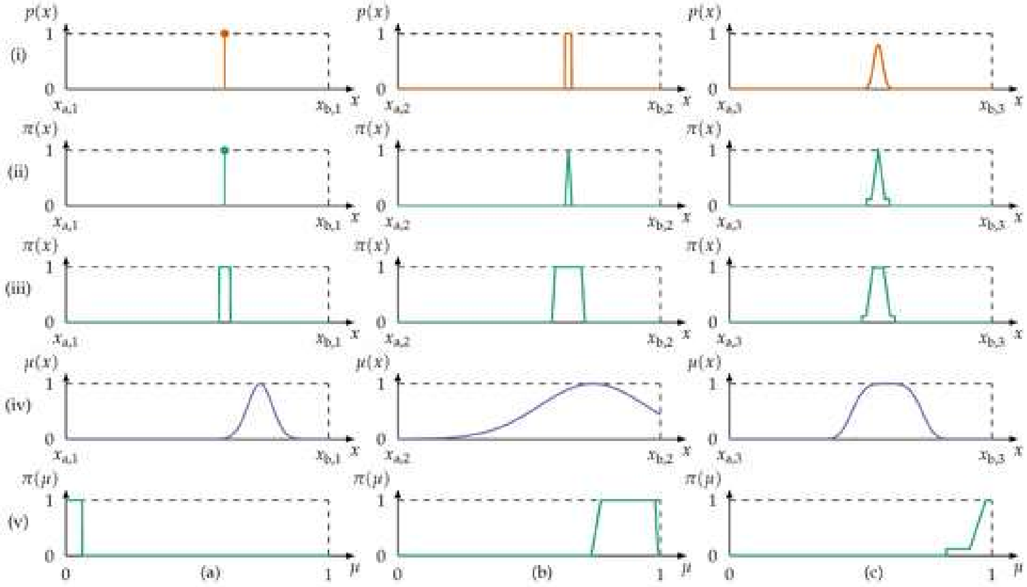

Fig. 5 Data Smoothing steps performed on i to v signals as probability distributions p(x)—as (a) single functions, (b) UPDF, and (c) GPDF. Here information regarding unkown variable or measurands are given by respective items (X1=[xa,1,xb,1], X2=[xa,2,xb,2], X3=[xa,3,xb,3])

Execution Time Comparison

Benchmarks	Device 1 (Raspberry Pi 4)		Device 2 (Intel Atom Broxton M-T5700)		Device 3 (ATMega328 P)		Throughput (Mbps)	Packet Loss Rate (Data Rate = 30 Kbps)
	Execution Time (s)	Performance	Execution Time (s)	Performance	Execution Time (s)	Performance		
AES	0.95	0.45	0.98	0.45	0.83	0.42	59.22	0.35
DES	0.81	0.4	0.88	0.55	0.83	0.54	36.4	0.24
RC5	0.72	0.64	0.78	0.93	0.75	0.91	47.89	0.39
ECC	1.06	0.58	1.02	0.66	1.02	0.66	12.98	0.18
SHA-256	0.73	0.68	0.74	0.92	0.78	0.9	116.85	0.55
ECDH	0.82	0.4	0.83	0.92	0.81	0.92	117.8	0.7
Hybrid ECDH	0.88	0.39	0.89	0.93	0.87	0.93	120.6	0.7

Fig. 6 Experimental Results showing Performace

The experiments use an IoT-integrated programme that assures a secure connection. The software uses gyroscope and accelerometer sensors along with an IoT device to collect data. The data is securely published to Watson IoT. The outcomes of data science analysis are offered here. Data transfer and receive times, computational time, encryption / decryption timings, and privacy given by multiple mechanisms are all noted.

The horizontal axis indicated the time succession of observations (Fig. 4). The vertical axis indicates the z-score.

The lower and upper bounds assist identify data spikes. The data spikes are outliers or unusual numbers. The findings showed discrepancies inside the data sensed. Anomalies are statistics that are both across each threshold. The duration of inspection and the z-score values are related.

The empirical investigation included three sensors, as shown in Fig. 5. The letters oa, ob, and og denote them. These sensors have density graphs. The sensor's density distribution is provided. The graphs show the data's various patterns.

As shown in Figure 6, the effectiveness of security systems, along with the new Hybrid-ECDH, is measured in seconds. The RSA, DH, ECDH, and Hybrid-ECDH methods are compared. The data sizes are ten, fifty, one hundred, and five hundred megabytes. The findings show that the proposed method executes faster owing to its compact calculations.

The Hybrid-ECDH protocol specifies upload time in seconds. Concerning comments In terms of total upload time, RSA, DH, ECDH, and Hybrid-ECDH are compared. The data sizes are ten, fifty, one hundred, and five hundred megabytes. The findings show that the recommended approach uploads data faster than the current standard.

The download time of security systems, including the proposed Hybrid-ECDH, has been researched. The total download time for RSA, DH, ECDH, and Hybrid-ECDH is computed. The data sizes are ten, fifty, one hundred, and five hundred megabytes. The results show that the recommended solution takes less overall download time than the current standard.

We use DES as a benchmark for key size. This technique clearly beats rivals when comparing the same amount of bits for a given key size. Due to its smaller key size, E-ECDH reveals the most bits. So E-ECDH beats RSA, DH, AES, and ECDH. The findings in this section bolster those in Sect. 7.

This section evaluates the IoT case study application shown in Fig. 1 for data collecting, secure connectivity, and data analytics. The IBM Watson IoT platform is used for the tests. MQTT is the protocol used to assure lightweight messaging. The database is Cloudant, a NoSQL database. The database is online. Prioritize security and data analytics. The spikes in Fig. 4 show data abnormalities. Various data sizes were used to test the proposed scheme's performance. Thus, the prototype is beneficial by combining required features and security improvements. With time, it will become a more sophisticated out-of-box solution.

VII. CONCLUSION AND FUTURE WORK

There is a novel technique called Hybrid-ECDH that gives a higher level of security than the current state-of-the-art. Because IoT devices and other devices are involved, secure end-to-end connectivity is necessary. Apps for IoT must be secure. ECDH is being replaced with a new algorithm called Hybrid-ECDH. Surpassing ECDH's security issues protects critical digital infrastructure for IoT applications. It is used in an IoT case study application that uses data analytics to detect unusual behaviour. The results are assessed in terms of security strength, execution time, and time necessary to download and upload various sized workloads. The IoT Cloud platform is integrated with a healthcare application in an empirical research. The application helps detect a patient's movements. Then, utilising cloud resources, the data is handled. End-to-end secure connections are utilised to guarantee data flow and analytics are secure. The suggested technique outperforms the current state of the art. We intend to keep increasing IoT security by reviewing apps across domains.

REFERENCES

1. Baranwal, Nitika, T., &Pateriya, P. K. (2016). Development of IoT based smart security and monitor- ing devices for agriculture. In 2016 6th international conference—cloud system and big data engineer- ing (Confluence) (pp. 1–6).

2. Ahmed, I., & Kannan, G. (2018). A review on present state-of-the-art on internet of things. Journal of
Advanced Research in Dynamical and Control Systems (12), 352–358.
3. Reijo, M., &Savola, P. (2015). Risk-driven security metrics development for an eHealth IoT applica- tion. In IEEE (pp. 1–6).
4. Kannan, G., & Mohamed Thameez, R. (2015). Design and implementation of smart sensor inter- face for herbal monitoring in IoT environment. International Journal of Engineering Research, 3(2),
469–475.
5. Duc, A., Jabangwe, R., Paul, P., &Abrahamsson, P. (2017). Security challenges in IoT development: a software engineering perspective. Proceedings of XP2017 Scientific Workshops, ACM, 1–5.
6. Datta, S. K., & Bonnet, C. (2016). Easing IoT application development through data tweet framework.
In 2016 IEEE 3rd world forum on internet of things (WF-IoT) (pp. 1–6).
7. Ammar, M., Russello, G., &Crispo, B. (2018). Internet of things: A survey on the security of IoT
frameworks. Journal of Information Security and Applications, 38, 8–27.
8. Kang, Y.-M., Han, M.-R., Han, K.-S., & Kim, J.-B. (2015). A study on the internet of things (IoT)
applications. International Journal of Software Engineering and Its Applications, 9(9), 117–126.
9. Celik, Z. B., Fernandes, E., Pauley, E., Tan, G., & McDaniel, P. D. (2018). Program analysis of com- modity IoT applications for security and privacy: Challenges and opportunities. ACM Computing Sur- veys, 52(4), 1–39.
10. Ahmed, M. I., & Kannan, G. (2020). Overcoming privacy and security challenges of internet of things applications. International Journal of Future Generation Communication and Networking, 13(1),
1550–1556.
11. Jonsson, F., &Tornkvist, M. (2017). RSA authentication in internet of things. http://www.diva-portal. org/smash/get/diva2:1112039/FULLTEXT01.pdf.
12. Riahi, A ., Challal, Y., Natalizio, E., Chtourou, Z., &Bouabdallah, A. (2013) A systemic approach for
IoT security. DCOSS, Boston, United States (pp. 351–355).
13. Pustišek, M., & Kos, A. (2018). Approaches to front-end IoT application development for the Ethere- umblockchain. Procedia Computer Science, 129, 410–419.
14. Datta, S. K., Gyrard, A., Bonnet, C., &Boudaoud, K. (2015). oneM2M architecture based user cen- triciot application development. In 2015 3rd international conference on future internet of things and cloud (pp. 1–8).
15. Tweneboah-Koduah, S., Skouby, K. E., &Tadayoni, R. (2017). Cyber security threats to IoT applica- tions and service domains. Wireless Personal Communications, 95(1), 169–185.

16. Ahamed, J., &Rajan, A. V. (2016). Internet of things (IoT): Application systems and security vul- nerabilities. In 2016 5th International conference on electronic devices, systems and applications (ICEDSA) (pp. 1–5).
17. Liu, Z., & Yan, T. (2013). Study on multi-view video based on IOT and its application in intelligent security system. In Proceedings 2013 international conference on mechatronic sciences, electric engi- neering and computer (MEC) (pp 1–4).
18. DíazLópez, D., Blanco Uribe, M., Santiago Cely, C., TarquinoMurgueitio, D., Garcia, E., Nespoli, P.,
& Gómez Mármol, F. (2018). Developing secure IoT services: A security-oriented review of IoT plat- forms. Symmetry, 10(12), 1–34.
19. Diffie, W., & Hellman, M. (1976). New directions in cryptography. IEEE Transactions on Information
Theory, 22(6), 644–654.
20. AL-mawee, W. (2012). Privacy and security issues in IoT healthcare applications for the disabled users a survey, 1–57.
21. Nguyen, X. T., Tran, H. T., Baraki, H., &Geihs, K. (2015). FRASAD: A framework for model-driven
IoT Application Development. In 2015 IEEE 2nd world forum on internet of things (WF-IoT) (pp.
1–6).
22. Jonsson, F., &Tornkvist, M. (2017). RSA authentication in internet of things. http://www.diva-portal. org/smash/get/diva2:1112039/FULLTEXT01.pdf.
23. Ukil, A., Bandyopadhyay, S., Bhattacharyya, A., Pal, A., & Bose, T. (2014). Lightweight security scheme for IoT applications using CoAP. International Journal of Pervasive Computing and Commu- nications, 10(4), 372–392.