

A java servlet based transaction broker for internet of things edge device communications

Zainatul Yushaniza Mohamed Yusoff¹, Mohamad Khairi Ishak¹, Lukman AB Rahim²

¹School of Electrical and Electronic Engineering, Universiti Sains Malaysia, Pulau Pinang, Malaysia

²Department of Computer and Information Science, Faculty of Science and IT, Universiti Teknologi Petronas, Perak, Malaysia

Article Info

Article history:

Received Aug 31, 2021

Revised Nov 19, 2021

Accepted Dec 10, 2021

Keywords:

ECC

Internet of things

Java servlet

Security

ABSTRACT

Internet of things (IoT) technology is growing exponentially in almost every sphere of life. IoT offers several innovation capabilities and features, but they are also prone to security vulnerabilities and risks. These vulnerabilities must be studied to protect these technologies from being exploited by others. Cryptography techniques and approaches are commonly used to address and deal with security vulnerabilities. In general, the message queuing telemetry transport (MQTT) is an application layer protocol vulnerable to various known and unknown security issues. One possible solution is to introduce an encryption algorithm into the MQTT communication protocol for secure transmission. This study aims to solve the security problem of IoT traffic by using a secure and lightweight communication proxy. The strategy behind this communication broker acts as a network gateway providing secure transaction keys to all IoT nodes in the network. This task uses a java servlet and elliptic curve cryptography (ECC) algorithm to generate identity encryption keys in a component-based web transaction infrastructure. This approach encrypts the data before it is sent via the MQTT protocol to secure the communication channel and raise the security device and network transactions.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Mohamad Khairi Ishak

School of Electrical and Electronic Engineering, Universiti Sains Malaysia

14300, Nibong Tebal Pulau Pinang, Malaysia

Email: khairiishak@usm.my

1. INTRODUCTION

The internet of things (IoT) is one of the most exciting technologies that connect the physical world with digital communications, interactions, transactions, predictive analytics, enabling governments, and large corporations to make decisions with analysis. In the existing digital world, information security, and cybersecurity schemes are used in a thin-layer approach. The former relates to transaction security, the second being related to peripheral devices and central nodes [1]. However, in today's digital world operating as IoT devices within industry 4.0, a multi-layered security approach is required, developed in this study as a secure middleware framework. It has traditionally assumed that many independent systems and strategies are solved most of these problems independently [2], [3]. For example, it includes secure bank authentication schemes using one-time passwords, cryptographic authentication schemes, two-factor authentication, application-level middleware solutions, and finally, blockchain implementations [4]-[8]. However, all these solutions do not provide built-in security for these IoT devices. The IoT is the physical unit where devices combine wireless sensors to send data to a platform. With advances in IoT processing, devices as botnets increase security risks, which can be dangerous for many people. The most used protocols in IoT are hypertext transfer protocol (HTTP), extended presence and message protocol (XMPP), restricted application

protocol (CoAP), extended message queuing protocol (AMQP), and message queuing protocol (MQTT) messages [9]. MQTT is popular in IoT devices because it operates with low bandwidth and can use as a limited resource for IoT devices. MQTT consists of three primary nodes: editor nodes, mediation nodes, and member nodes [10]-[13]. The MQTT protocol is not secure for communication and is readable by unauthorized persons. Network devices and applications are a stand-alone approach proposed or applied at different levels for the homogeneous dispersion system. The intrinsically associated with heterogeneous, distributed network, the IoT regulatory requirements, as well as centralized and distributed deployment scenarios, should provide a security architecture in the various layer (physical, network, and application) to achieve the following: secure device provisioning (configuration with secure personal identification (SPID) key); secure inter-operability (network management with secure using middleware broker); secure device-to-device & service-to-service interactions (inter-device transactions using secure middleware broker).

Its motivation is to use servlet technology to create a secure SPID system with a private identification key for IoT communication systems. The system generates encryption keys based on the ECC algorithm. In this work, the proposed middleware security broker key for enhanced security and device/network isolation to protect the native devices, networks, and transactions in the IoT network operating at the edge (overlapping region), as shown in Figure 1. In addition, the proposed method can be implemented for various security approaches to enhance security as supported [14]-[18].

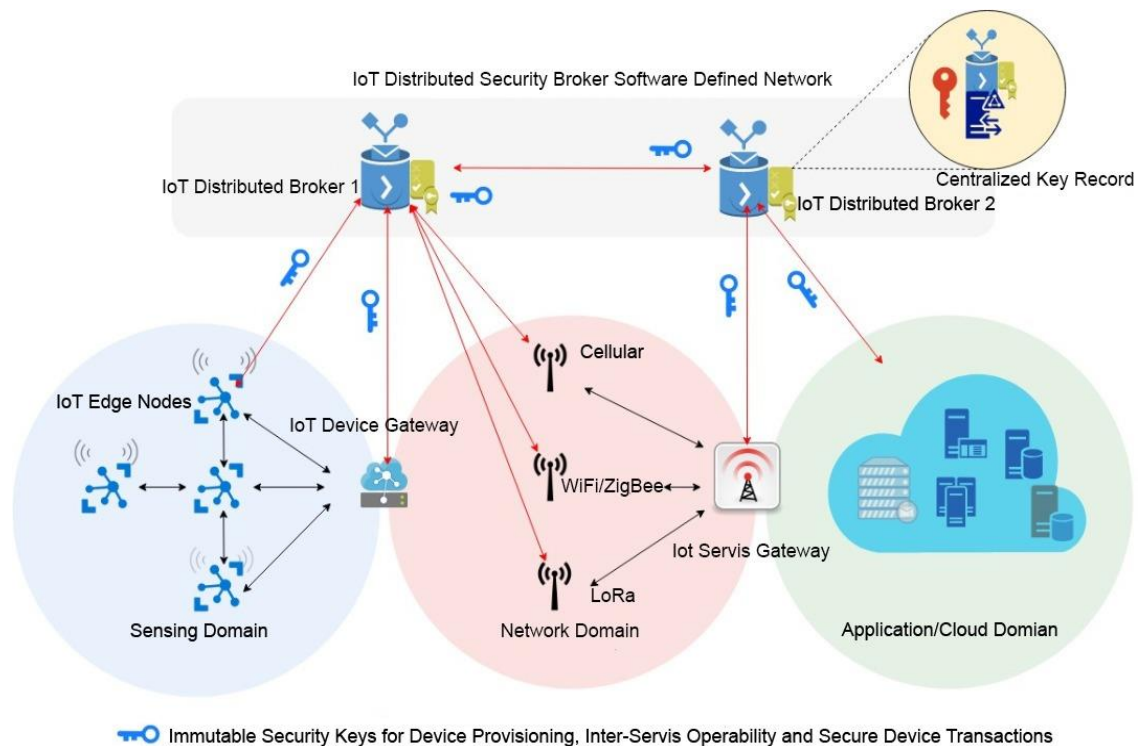


Figure 1. Proposed middleware security broker key for enhanced security and device/network isolation

The security framework provides a brokerage middleware key solution that runs in parallel and distributed deployment. Security roles and concerns can be divided into the three-tier model (collection domain, network domain, and application domain), as shown in Figure 1. A key-based approach to data encryption identification keys is the ECC algorithm. ECC uses a public/private key pair to decrypt and encrypt the web traffic. A practical approach to encoding is ECC, an alternative to rivest-shamir-adleman (RSA). It uses an elliptical curve to create essential protection for public-key cryptography. From a security perspective, the difference between RSA and ECC encryption keys is significant [19]. Simply put, a 384-bit elliptic curve encryption key provides the same level of security as a 7680-bit RSA Size is an essential characteristic of elliptic curve coding. It is due to the increased performance of small mobile devices. ECC is faster and offers greater security with shorter keys than RSA in a world where mobile devices with lower processing power can perform more encryption [20].

2. THE PROPOSED METHOD

According to the international telecommunication union (ITU), governments, and businesses are increasing their investments in cyber security [21]. However, these statistics have been collected from mobile operators, internet service providers, and traditional IP networks considered consistent and homogeneous. The task is difficult because of limited device capabilities, heterogeneity, and limited human intervention on the internet of things. Cyber security strategies are managed and implemented in traditional IP-based systems through rigorous security testing, such as vulnerability assessment and penetration testing. The limitations of device capabilities, multiple communication technologies, and lack of standardization in IoT-based systems expose security vulnerabilities that cannot quickly address this technology area due to the direct impact of penetration testing [22], [23]. Mapping existing IP-based cyber security models to IoT-based systems requires understanding current security practices outlined in the confidentiality, integrity, and availability (CIA) model. Existing cyber security networks and their implementations loosely coupled with an excellent triangular CIA model provides a flexible framework for data availability, integrity, and confidentiality, as shown in Figure 2. The main purpose of this model is to provide a point. Secure the IT systems so that data is always sent to reliable equipment, always available, and always stored in a location where interventions can track, analysed, and logged [24]-[30]. By 2025, more than 75 billion devices are expected to be connected to the internet [31]. The gateway supports multiple IP networks and an extensive network address translation (NAT) storage database. Another aspect to consider is a static root, static NAT translation, usual NAT translation, and dynamic NAT translation (amount of memory saved for time-consuming prompts). Furthermore, new devices are continually added and removed from the internet. As a result, existing IP addressing schemes cannot wholly solve the problem of scale correctly. Many security solutions are currently being proposed at various levels to address the challenges in a network system.

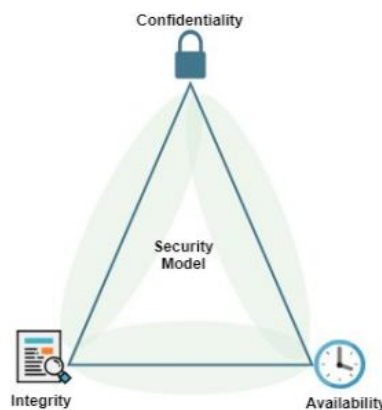


Figure 2. CIA security model

However, there is still a lack of the overall security, trust, and compliance model needed to manage such large-scale networks capable of executing billions of transactions daily. Our approach is based on these solutions, deploying a secure data layer API to monitor devices, logs, service contracts, and transactions between them. The architecture of the IoT, applications in all disciplines, and all related ecosystems are heterogeneous. This diversity needs to be determined; the proposed security agent architecture should consider proper IoT and industrial IoT (IIoT) implementation at all levels to provide unique security and support. It is also essential to distinguish between traditional IoT devices and counterfeit devices. The limited computing resources of standard IoT devices make it difficult to implement traditional security systems (application or hardware-based) such as firewalls. It is also essential to identify which IoT devices can use through imitation devices. In this situation, the proposed middleware broker key is generated for all IoT devices and network gateways. A device or physical layer is provided by generating a key for each device in the IoT architecture. A unique security key can identify billions of existing scalable IoT devices and IP traffic [32]-[36]. This study proposes a brokerage middleware key solution that runs in parallel and distributed deployment. Each IoT device has a unique security agent ID. These keys are part of the broker, so only devices with the appropriate information can join or remove the network and exchange information.

2.1. Implement of key broker using java servlet and elliptic curve cryptography system

Figure 3 shows the working principle associated with the presented middleware security broker key. As shown in the figure, the entered URL/data is first encrypted with ECC technology. At the same time, the record key is identified by a SPID, which is an essential feature of the broker by using java servlet.

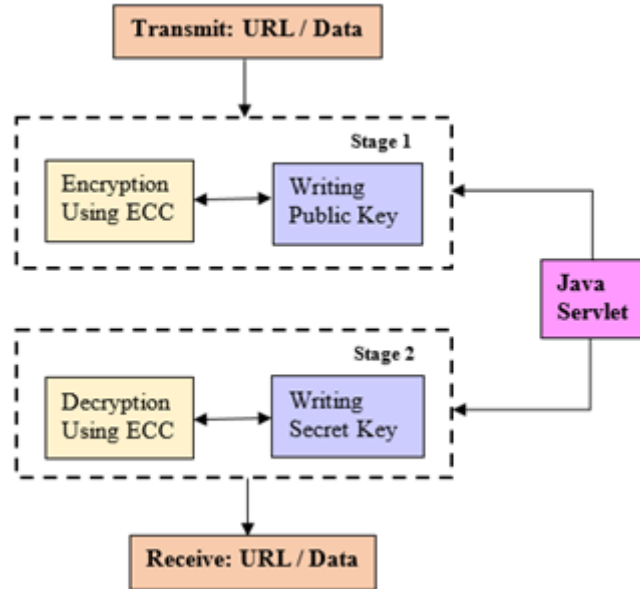


Figure 3. Overall process of proposed middleware security broker key

2.1.1. Elliptic curve cryptography technique

Elliptic curves are related to ECC, a type of public-key cryptography, primarily working with public/private key pairs. Today's public key cryptography does not require interaction between two parties before encrypting a message. Pre-shared keys can decrypt before the exchange begins and form the basis for more secure communication. ECC is known as the escape function. That is, it is easy to calculate in one direction and difficult to calculate in the other. Deciphering a message is easy and fast when working in one direction (private key, the other, and private key) knowing part of the equation. What makes ECC unique is the way the numbers used for public and private key pairs are generated. Today's public key cryptography system, RSA, multiplies two large primes, and uses products with them to secure communications. If ECC does not know the starting point (private key) that generated the curve, it uses the projection properties of the elliptic curve to pick a randomly visible point on the graph. Therefore, the curve is represented by the expression specified as in (1).

$$y^2 = x^3 + ax + b \pmod{p} \quad (1)$$

Where a and b mean integers and p defines mod values. The prime number is specified in ECC as k , and the private key is specified as $S.K$. So, the ECC operation is as in (2).

$$dum = ((x_{cube}).add(a.multiply(x))).add(b) \quad (2)$$

It specified as mode p . Compare against y_{square} with dum equal to 0. One of the main advantages of ECC is that it consists of a small key size, small storage space and transfer conditions. If the condition $x=y$ is met, the best point for the eclipse curve $E.C$ is determined.

2.1.2. Generation of keys

Two keys apply here. Express the private key in the form as in (3).

$$sk = cs.generateKey() \quad (3)$$

On the other hand, the public key expression is in (4).

$$pk=sk.getPublic() \quad (4)$$

The metric required scaled according to the total task, namely $PRIMESECURITY=500$.

- Encryption process

It is one of the most popular data protection methods for encrypting confidential messages so that the recipient can access the data. According to this proposal, releases are block-by-block and are encrypted using the ECC method. The number of blocks is denoted by (i, j) , where i and j define the row and column for each standard block. The expression is in (5).

$$cipher=cs.encrypt(buffer, top, key) \quad (5)$$

- Decryption process

It is the reverse process of an encryption mechanism based on transforming encrypted data in a simple context. This method uses the private key sk to retrieve the expression as in (6).

$$buffer=cs.decrypt(encrypt, key) \quad (6)$$

2.2. Java servlet technique

Servlets are server-free Java programs and platforms that can process data between clients and web servers by displaying or modifying data interactively using dynamic web page revenue technology. The servlet runs on the server and does not require a graphical user interface. A client program can be a browser or other program that can connect to the internet and access and make requests to web servers. A servlet can respond to a client's request by dynamically generating a response returned to the client each client request represented by a servlet request object (servlet request type). The response sent to the client is represented by a servlet response object of type servlet response. Figure 4 shows the fundamental behavior of the servlet. A servlet can call multiple times to handle requests from multiple clients. A servlet can handle multiple requests at the same time and synchronize them. It can also redirect requests to another server or servlet. To access the servlet, you need to run a URL command that points to the servlet's location as if it were in the directory or computer running the webserver. Servlets written in Java is ideal for implementing complex business application logic that allows clients to access relational databases through dynamic web pages.

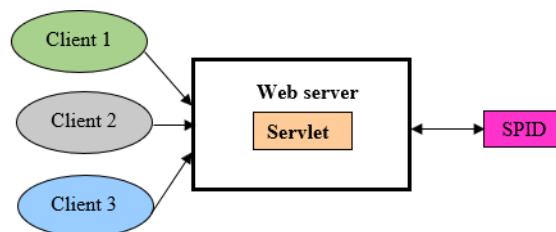


Figure 4. Basic servlet function

3. RESEARCH METHOD

Figure 5 shows the flowchart implementation of SPID essential broker functions. It consists of:

- Device ID as a level of discovery primarily includes access to a physical node, device, or sensor. Most technologies such as RFID, WSN, and GPS collect data sent to higher layers for analysis and intelligent processing. This layer has several unique properties that make it vulnerable to vulnerabilities and attacks, such as inconsistency, computing power, remote deployment, physical damage, and environmental changes.
- Network IDs, such as the network layer or, more generally, the transport layer, are helpful primarily in the application layer to provide ubiquitous access to the knowledge and communication layers. The transport layer is complex, including many communication technologies such as (LoRa, WiFi, 3G, 4G, ZigBee, Bluetooth, and others) and various protocols. Communication networks are secure, but the openness and heterogeneity of IoT devices, their use of information exchange methods, protocols, technology stacks, and multiple communication technologies can make them vulnerable to attack.
- Application ID, most application tiers are considered the most secure as they operate in a cloud environment that can provide high analytical and computational skills. However, the application layer has

a lot to do with the client as it provides the necessary data or services. One of the biggest challenges on the IoT is the large amount of information received by touch nodes. Personal (user behaviors displayed via intelligent device or social network integration), critical (medical device), work (smart office), and building), industrial and government or military applications. This information is highly confidential, providing a very complex threat interface with data breaches and the primary threat vectors.

- The transaction key for IoT device-to-device communication acts as an authentication transaction key. After detecting the device ID, network ID, and application ID, the ECC algorithm generates the ID transaction key before sending it to the gateway receiver. The public and private keys follow a 256-bit elliptical curve, so each ID transaction key generated is unique. Finally, the transaction ID key is sent to the destination requested by the user.

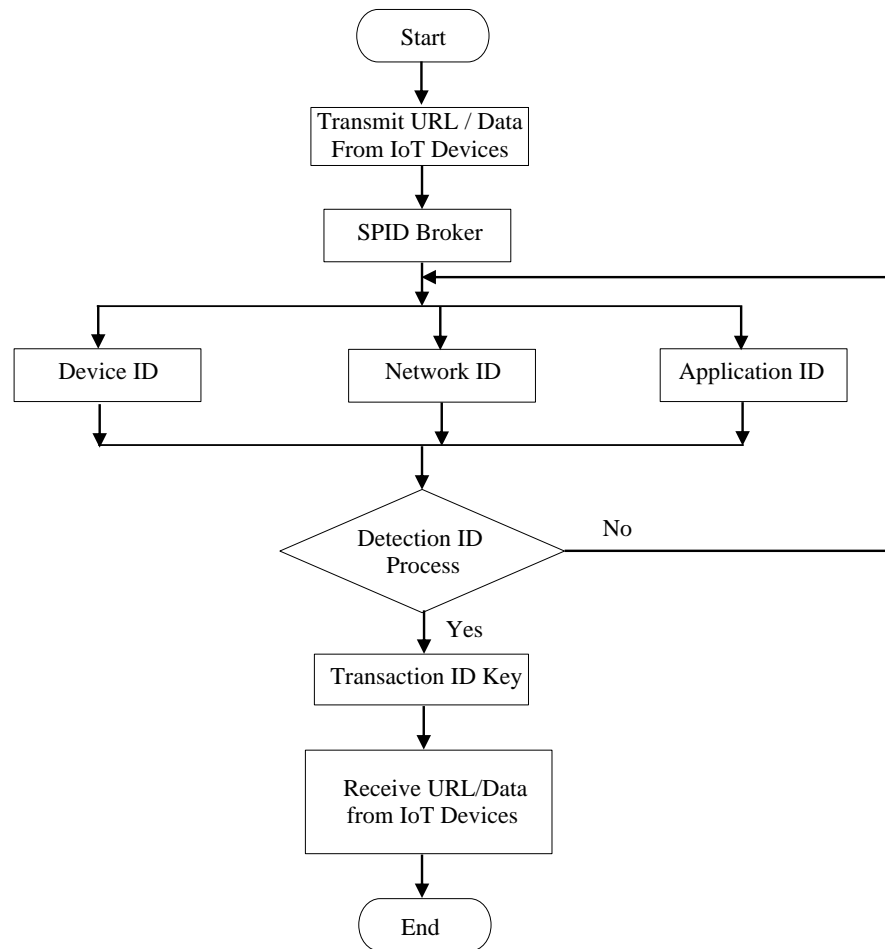


Figure 5. Flowchart of secure personal identification (SPID) essential broker functions

3.1. Experimental setup

Testbed for evaluating the performance of the SPID essential broker protocols describe in Table 1. It implemented SPID with our proposed scheme and described in [37] based on the java servlet platform. This same proposed approach is used to implement the transaction key for communication between the devices of each IoT device. It is safe to assume that each node's private key and the initiator's public key is the data source. Deriving the key is the responsibility of all responding nodes. Each node can use its secret point on the elliptical curve [38], [39]. Using a lightweight ECC scheme, it uses the encryption parameters described in Table 2 to visualize bilinear operations with keys of various sizes [40], [41].

Table 1. System details

Equipment	Specification
Hardware	Intel(R) Core I i7-3520M CPU @ 2.90GHz 2.90 GHz
Primary Memory Capacity	4GB
Operating System	Windows 10 Enterprise, 64-bit operating system, x64-based processor
Java Servlet Version	4.0
Java Version	11.0

Table 2. Elliptic curve cryptography parameter

Parameter	Expression	Output
curve	$y^2 = x^3 + ax + b \pmod{p}$	$y^2 = x^3 + 1x + 6 \pmod{11}$
Generator	generator = new ECPPoint(this, ecp.generatorX(), ecp.generatorY());	48439561293906451759052585252797914 20276294952604174799584408071708240 4635286, 36134250956749795798585127919587881 95661110667298501507187719825356841 4405109
Order	order = ecp.order();	11579208921035624876269744694940757 35299969552241357603424222590610685 12044369
a	a = new BigInteger("FFFFFFFF"+"00000001"+"00000000"+"00000000"+"0 0000000"+"FFFFFFFF"+"FFFFFFFF"+"FFFFFFFF", 16);	11579208921035624876269744694940757 35300861434152903141955336313088670 97853948
b	b = new BigInteger("5AC635D8"+"AA3A93E7"+"B3EBBD55"+"769886BC "+"651D06B0"+"CC53B0F6"+"3BCE3C3E"+"27D2604B", 16);	41058363725152142129326129780047268 40911444101599372555483525631403946 7401291
p	p = new BigInteger("FFFFFFFF"+"00000001"+"00000000"+"00000000"+"0 0000000"+"FFFFFFFF"+"FFFFFFFF"+"FFFFFFFF", 16);	11579208921035624876269744694940757 35300861434152903141955336313088670 97853951
gx	gx = new BigInteger("6B17D1F2"+"E12C4247"+"F8BCE6E5"+"63A440F2"+" "77037D81"+"2DEB33A0"+"F4A13945"+"D898C296", 16);	48439561293906451759052585252797914 20276294952604174799584408071708240 4635286
gy	gy = new BigInteger("4FE342E2"+"FE1A7F9B"+"8EE7EB4A"+"7C0F9E16" +"2BCE3357"+"6B315ECE"+"CBB64068"+"37BF51F5", 16);	36134250956749795798585127919587881 95661110667298501507187719825356841 4405109
n	n = new BigInteger("FFFFFFFF"+"00000000"+"FFFFFFFF"+"FFFFFFFF"+" "BCE6FAAD"+"A7179E84"+"F3B9CAC2"+"FC632551", 16);	11579208921035624876269744694940757 35299969552241357603424222590610685 12044369

4. RESULTS AND DISCUSSION

The SPID broker evaluated the transaction key using the ECC algorithm. When the SPID broker receives the URL/data, the device, network, and application identity discover the information via cellular, edge, gateway, or relay. The system is built using the scapy python program and runs in a java servlet program package. Scapy is a library written in Python using a command-line interpreter (CLI) to create, modify, send, and capture network packets. It can be run interactively through the command-line interface or imported into Python and used as a library. The main advantage of scapy is that, unlike other low-level tools, you can modify network packets to use existing network protocols and define parameters as needed [42]-[44]. After the device, network, and application IDs get the output shown in Figures 6-8, respectively, the transaction key system checks the overall risk severity. The application features related to authentication and management are appropriate. Make sure it is implemented in and does not allow attackers. It compromises passwords, keys, or session tokens or exploits other implementation flaws to take over another user's identity temporarily or permanently before the ECC algorithm generates the key.

Figure 6 shows the device ID system output, and it runs in java servlet. It detects the MAC address and name of devices. The transaction key system knows the data to encrypt and decrypt come from what devices. Network ID system output is display in Figure 7 and it also runs in java servlet. The transaction key system knows the data from the network ID system because it detects the IP address location.

```
User=SPID Device=Found 3 devices Device= 1C:3E:84:3D:C5:20 - KJ
Device= 89:94:68:31:FF:B9 - Tango neo 7 Device= 41:42:19:BD:DC:6D - Galaxy Buds
```

Figure 6. Device ID output

IP Address	MAC Address
10.0.2.2	52:54:00:12:35:02
10.0.2.3	52:54:00:12:35:03
10.0.2.4	52:54:00:12:35:04

Figure 7. Network ID output

Application ID system detects the download data from any URL. From the output shown in Figure 8, the transaction key knows the type of data and the content of the data transfer length. The device, network, and application ID information is collected before the transaction key does the encrypt and decrypt process, as shown in Figure 9. All the information is essential to ensure the URL/data is secure.

```
Content-Type = application/octet-stream
Content-Disposition = null
Content-Length = 32
fileName = latest.csv
File downloaded
BUILD SUCCESSFUL (total time: 1 second)
```

Figure 8. Application ID output

```
Writing Secret key
Writing Public key
Encrypting: AJK.docx -> AJK.docx.enc ...OK
Decrypting: AJK.docx.enc -> AJK.docx.dec ...OK
Reading Public key
sk is:ecc.rsa.RSAKey@2ef614f8
Reading Secret key
sk is:ecc.rsa.RSAKey@3839e5b1
```

Figure 9. Display output of the keys

5. CONCLUSION

In this article, we proposed a secret identification key SPID system for IoT communication systems. The system generates an encryption key based on the ECC algorithm. This task secures native devices, networks, and transactions in a running IoT network using a middleware security broker key to improve device/network isolation and security. This approach allows to optimize your security model's level to compete with the security of device and network transactions while maintaining scalable and manageable security features. In the future, the SPID system will be extended to include an IoT application system in order to assess the proposed system's efficiency.

ACKNOWLEDGEMENTS

The authors would like to thank Universiti Sains Malaysia (USM), particularly Ministry of Higher Education Malaysia for Fundamental Research Grant Scheme with Project Code: FRGS/1/2020/TK0/USM/02/1 and MARA for providing financial support to carry out this research.

REFERENCES




- [1] insidetelecom, "Malaysia - striving to become the Premier Regional IoT development hub," IoT, 2020. [Online]. Available: <https://www.insidetelecom.com/malaysia-striving-to-become-the-premier-regional-iot-development-hub/>.
- [2] M. A. H. M. Yunus, "Regulatory Challenges of Internet of Things (IoT)," *Communication and Multimedia Commission: M.C.M.C. Official Website*, pp. 1-16, 2018.
- [3] N. Zaman, M. Humayun, and S. Almuayqil, "Cyber Security and Privacy Issues in Industrial Internet of Things," *Computer Systems Science and Engineering*, vol. 37, no. 3, pp. 361-380, 2021, doi: 10.32604/csse.2021.015206.
- [4] M. Alwabel and Y. Kwon, "Blockchain Consistency Check Protocol for Improved Reliability," *Computer Systems Science and Engineering*, vol. 36 no. 2, pp. 281-292, 2021, doi: 10.32604/csse.2021.014630.
- [5] M. M. Hamdi *et al.*, "A review on various security attacks in vehicular ad hoc networks," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 5, pp. 2627-2635, Oktober 2021, doi: 10.11591/eei.v10i5.3127.
- [6] A. A. Najib, R. Munadi, and N. B. A. Karna, "Security system with RFID control using EKTP and internet of things," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 3, pp. 1436-1445, June 2021, doi: 10.11591/eei.v10i3.2834.
- [7] M. A. Naagas, E. L. Mique Jr, T. D. Palaoag, and J. D. Cruz, "Defense-through-Deception Network Security Model: Securing University Campus Network from DOS/DDOS Attack," *Bulletin of Electrical Engineering and Informatics*, vol. 7, no. 4, pp. 593-600, December 2018, doi: 10.11591/eei.v7i4.1349.
- [8] P. T. Tin, D. H. Ha, M. Tran, and T. T. Trang, "Physical security layer with friendly jammer in half-duplex relaying networks over rayleigh fading channel: Intercept probability analysis," *Bulletin of Electrical Engineering and Informatics*, vol. 9, no. 4, pp. 1694-1700, August 2020, doi: 10.11591/eei.v9i4.2249.
- [9] S. Rajashree, K. S. Soman, and P. G. Shah, "Security with IP Address Assignment and Spoofing for Smart IOT Devices," *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2018, pp. 1914-1918, doi:

- 10.1109/ICACCI.2018.8554660.
- [10] M. A. Naagas, A. R. Malicdem, and T. D. Palaoag, "DEH-DoSV6: A defendable security model against IPv6 extension headers denial of service attack," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 274-282, February 2021, doi: 10.11591/eei.v10i1.2670.
 - [11] Z. Weng, T. Chen, T. Zhu, H. Dong, D. Zhou, and O. Alfarrarj, "TLSSmell: Direct Identification on Malicious HTTPs Encryption Traffic with Simple Connection-Specific Indicators," *Computer Systems Science and Engineering*, vol. 37, no. 1, pp. 105-119, 2021.
 - [12] S. Görmüş and A. F. Yavuz, "A protocol for Internet of Things: IETF 6TISCH," *2017 25th Signal Processing and Communications Applications Conference (SIU)*, 2017, pp. 1-4, doi: 10.1109/SIU.2017.7960606.
 - [13] A. El Hajjar, "Securing the Internet of Things Devices Using Pre-Distributed Keys," *2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW)*, 2016, pp. 198-200, doi: 10.1109/IC2EW.2016.22.
 - [14] D. Kao and S. Hsiao, "The dynamic analysis of WannaCry ransomware," *2018 20th International Conference on Advanced Communication Technology (ICACT)*, 2018, pp. 159-166, doi: 10.23919/ICACT.2018.8323682.
 - [15] C. T. Poomagal, S. G. A. Kumar, and D. Mehta, "Multi Level Key Exchange and Encryption Protocol for Internet of Things (IoT)," *Computer Systems Science and Engineering*, vol. 35, no. 1, pp. 51-63, 2020, doi: 10.32604/csse.2020.35.051.
 - [16] Y. Zhang, S. Wang, H. Xia, and J. Ge, "A Novel SVPWM Modulation Scheme," *2009 Twenty-Fourth Annual IEEE Applied Power Electronics Conference and Exposition*, 2009, pp. 128-131, doi: 10.1109/APEC.2009.4802644.
 - [17] O. Kupreev, J. Strohschneider, and A. Khalimonenko, "KasperSky DDOS Intelligence report for Q3 2016," *Kaspersky Labs*, p. 14, 2016.
 - [18] F. Chen and C. Yin, "A Novel Method to Resolve the Dynamic IP Address for the Server-Side Gateway over the Internet of Things," *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*, 2012, pp. 1-3, doi: 10.1109/WiCOM.2012.6478338.
 - [19] L. S. Alotaibi and S. Alshamrani, "Smart Contract: Security and Privacy," *Computer System Science Eng.*, vol. 38, no. 1, pp. 93-101, January 2021, doi: 10.32604/csse.2021.015547.
 - [20] B. S. Adiga, M. A. Rajan, R. Shastry, V. L. Shivraj, and P. Balamuralidhar, "Lightweight IBE scheme for Wireless Sensor nodes," *2013 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 2013, pp. 1-6, doi: 10.1109/ANTS.2013.6802866.
 - [21] S. Thomson, T. Narten, and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," *RFC 4862*, 2007. [Online]. Available: <https://tools.ietf.org/html/rfc4862>. [cited 2019 01 March].
 - [22] H. Zhang and T. Zhang, "Short Paper: 'a peer to peer security protocol for the internet of things': Secure communication for the sensible things platform," *18th International Conference on Intelligence in Next Generation Networks (2015)*, 2015, pp. 154-156, doi: 10.1109/ICIN.2015.7073825.
 - [23] C. N. I. Group, "DHCPv6 Based IPv6 Access Service," *Cisco, Editor, Cisco: Cisco White Papers*, p. 43, 2011. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipaddr_dhcp/configuration/xs-3se/3850/dhcp-xe-3se-3850-book/config-dhcp-server.html
 - [24] T. Porter and M. Gough, "Chapter 3 - Architectures," in *How to Cheat at VoIP Security*, 2007, pp. 45-110. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/B9781597491693500049>.
 - [25] M. Ambrosin, M. Conti, A. Ibrahim, A. -R. Sadeghi, and M. Schunter, "Sciot: A secure and scalable end-to-end management framework for iot devices," in *European Symposium on Research in Computer Security, Springer*, pp. 595-617, 2018, doi: 10.1007/978-3-319-99073-6_29.
 - [26] Y. J. Jia *et al.*, "Contextlot: Towards providing contextual integrity to applied iot platforms," in *NDSS Symposium 2017*, vol. 2, no. 2, p. 2, 2017, doi: 10.14722/ndss.2017.23051.
 - [27] Z. B. Celik, G. Tan, and P. Mcdaniel, "IoTGuard: Dynamic Enforcement of Security and Safety Policy in Commodity IoT," *NDSS*, Feb. 2019, doi: 10.14722/ndss.2019.23326.
 - [28] Statista, "Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in Billions)," *IoT 2018* [Online]. Available: <https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/>. [cited 2019 01 March].
 - [29] Y. Jia *et al.*, "Burglars' IoT Paradise: Understanding and Mitigating Security Risks of General Messaging Protocols on IoT Clouds," *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, pp. 465-481, doi: 10.1109/SP40000.2020.00051.
 - [30] F. Chen, Y. Huo, J. Zhu, and D. Fan, "A Review on the Study on MQTT Security Challenge," *2020 IEEE International Conference on Smart Cloud (SmartCloud)*, 2020, pp. 128-133, doi: 10.1109/SmartCloud49737.2020.00032.
 - [31] S. N. Mohanty *et al.*, "An efficient Lightweight, integrated Blockchain (ELIB) model for IoT security and privacy," *Future Generation Computer Systems*, vol. 102, pp. 1027-1037, Jan. 2020, doi: 10.1016/j.future.2019.09.050.
 - [32] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT systems: Design challenges and opportunities," *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 417-423, doi: 10.1109/ICCAD.2014.7001385.
 - [33] E. D. Buenrostro, D. Cyrus, T. Le, and V. Emamian, "Security of IoT Devices," *Journal of Cyber Security Technology*, vol. 2, pp. 1-13, 2018.
 - [34] J. Oh, W. H. Ahn, and T. Kim, "Automatic Extraction Techniques for Component Collaboration With 3blit Filters and Wrappers in Java Web Apps," *Journal of the Society of Information Processing: Software and Data Engineering*, vol. 6, no. 7, pp. 329-336, 2017, doi: 10.3745/KTSDE.2017.6.7.329.
 - [35] L. D. Singh and K. M. Singh, "Implementation of Text Encryption using Elliptic Curve Cryptography," *Procedia Computer Science*, vol. 54, pp. 73-82, 2015, doi: 10.1016/j.procs.2015.06.009.
 - [36] M. Jaiswal and K. Lata, "Hardware Implementation of Text Encryption using Elliptic Curve Cryptography over 192 bit Prime Field," *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2018, pp. 343-349, doi: 10.1109/ICACCI.2018.8554410.
 - [37] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of Attribute-Based Encryption: Toward data privacy in the IoT," *2014 IEEE International Conference on Communications (ICC)*, 2014, pp. 725-730, doi: 10.1109/ICC.2014.6883405.
 - [38] S. Preethi and R. S. Kumaran, "Secure Key Establishment Scheme for Wireless Sensor Network in Distributed IoT Applications," *Secure Key Establishment Scheme for Wireless Sensor Network in Distributed IoT Applications*, vol. 3, no. 8, pp. 90-93, 2016.
 - [39] U. Mustafa and N. Philip, "Group-Based Key Exchange for Medical IoT Device-to-Device Communication (D2D) Combining Secret Sharing and Physical Layer Key Exchange," *2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3)*, 2019, pp. 1-7, doi: 10.1109/ICGS3.2019.8688022.
 - [40] X. Qin, Y. Huang, and X. Li, "An ECC-based access control scheme with lightweight decryption and conditional authentication for data sharing in vehicular networks," *Soft Computing*, vol. 24, no. 24, pp. 18881-18891, doi: 10.1007/s00500-020-05117-x.




- [41] F. Mallouli, A. Hellal, N. S. Saeed, and F. A. Alzahrani, "A Survey on Cryptography: comparative Study between RSA vs ECC Algorithms, and RSA vs El-Gamal Algorithms," *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/ 2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, 2019, pp. 173-176, doi: 10.1109/CSCloud/EdgeCom.2019.00022.
- [42] M. Moharir, K. B. Adyathimar, Shobha G., and V. Soni, "Scapy Scripting to Automate Testing of Networking Middleboxes," *Advances in Science, Technology and Engineering Systems Journal*, vol. 5, no. 2, pp. 293-298, 2020, doi: 10.25046/aj050238.
- [43] G. M. Kumar and A. R. Vasudevan, "D-SCAP: DDoS Attack Traffic Generation Using Scapy Framework," *Advances in intelligent systems and computing*, vol. 750, pp. 207-213, 2019, doi: 10.1007/978-981-13-1882-5_19.
- [44] M. M. Zaman, B. N. Bristy, and T. M. Mukur, "Internal Security Monitoring of an Organisation by Scapy & Kali Linux," *Advances in intelligent systems and computing*, 2018.

BIOGRAPHIES OF AUTHORS






Zainatul Yushaniza Mohamed Yusoff    received the B.Eng. degree in electrical, electronic engineering (communication) from Universiti Selangor (Unisel), Malaysia, in 2011, and the M.Sc. degree in wireless communication engineering from the University Putra Malaysia (UPM), Malaysia, in 2018. She is currently pursuing a Ph.D degree with Universiti Sains Malaysia (USM), Malaysia. She can be contacted at email: zainiza75@student.usm.my.



Mohamad Khairi Ishak    received the B.Eng degree in Electrical and Electronics Engineering from the International Islamic University Malaysia (IIUM), Malaysia, the M.Sc. in Embedded Systems, from the University of Essex, the United Kingdom and Ph.D from the University of Bristol, United Kingdom. He is a member of IEEE and a registered graduate engineer with the Board of Engineers Malaysia (BEM). Currently, he is a Senior Lecturer in Mechatronics Engineering at the School of Electrical and Electronic Engineering, Universiti Sains Malaysia (USM). His research interests are Embedded systems, Real-Time Control Communications and the Internet of Things (IoT). Emphasis is given towards the development of theoretical and practical methods which can be practically validated. Recently, significant research has been directed towards important industrial issues of embedded networked control systems and IoT. He can be contacted at email: khairiishak@usm.my.



Lukman AB Rahim    received the Ph.D. degree from Lancaster University, with a project verifying model transformations using model checking. He is currently a Core Researcher at the High-Performance Cloud Computing Center and a Senior Lecturer in Computer and Information Sciences at Universiti Teknologi PETRONAS (UTP). His current research interests are in formal verification, software and system modelling, and software architecture. His current research is focused on adopting model-driven engineering and formal verification in cloud computing. Some of the projects he is presently working on are using architecture-driven modernization and model-driven engineering in deploying engineering simulation software as a cloud service; formal verification of cloud security mechanisms using model checking; and domain-specific modeling languages for educational games. Apart from these projects, he is also involved in research projects related to system engineering and big data, i.e., real-time cloud platforms, the correlation between scheduling and job workload and energy consumption, and secure data transmission for big data applications. He is also working on consultancy projects for corrosion monitoring using acoustic technology and pipeline integrity system, playing the role as project leader and system engineer. He can be contacted at email: lukmanrahim@utp.edu.my.