Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 26 (2023), No. 3, pp. 793–806
DOI : 10.47974/JDMSC-1755

Embedded security framework for M2M communication using ULBC and Rubik's Cube encryption algorithm

Mahendra Balkrishna Salunke *
Department of Computer Engineering
Smt. Kashibai Navale College of Engineering
SPPU
Pune
Maharashtra
India

Parikshit Narendra Mahalle †
Department of Artificial Intelligence and Data Science
Vishwakarma Institute of Information Technology
SPPU
Pune
Maharashtra
India

Gitanjali Rahul Shinde §
Department of Computer Engineering
Vishwakarma Institute of Information Technology
SPPU
Pune
Maharashtra
India

Abstract

The potential of a variety of stockholders has increased as a direct result of the development of the Internet of Things and cloud computing. These individuals are now able to communicate and share data in a productive manner. This interaction with devices is regarded as extremely helpful and advantageous by a lot of users. Inadequate configuration of a network system makes it an easy target for potential security breaches. This is why it is

* *E-mail:* msalunke@gmail.com (Corresponding Author)
† *E-mail:* aalborg.pnm@gmail.com
§ *E-mail:* gr83gita@gmail.com

important that a security framework is developed for the Internet of Things (IoT) and cloud computing models. This paper talks about the various frameworks that are used in these two technologies. It also explores the various security methods that are used to protect the devices that are connected to the cloud. Numerous studies have shown that there are various ways that people can manipulate the operations of machine-to-machine (M2M) in the digital world. While analyzing the methodology used in this technology, it is also important to keep in mind the security levels of the cloud and IoT ecosystem. The development of a secure connection between the various devices that are connected to the cloud is also important to ensure that the network is protected from security threats. This paper aims to provide a framework that will allow the integration of the Cloud and IoT and the Cloud Framework for Machine-to-Machine communication. The paper aims to develop a secure framework that will allow the integration of the Cloud and IoT with secure framework for Machine-to-Machine communication using (ultra-lightweight block cipher) ULBC and Rubik's Cube encryption algorithm.

Subject Classification: *94A60 (Cryptography).*

Keywords: *M2M, IoT, AES, ECC, Encryption/ decryption.*

1. Introduction

Machine-to-machine (M2M) communication has grown rapidly with wireless networks. This network lets many devices collect and share data without human intervention. Due to its versatility, it should be cost-effective. The rapid development of machine-to-machine (M2M) technology has opened up many new monitoring and management options for devices and services. Despite its promise, the technology has many technical issues (Irshad 2017).

Software, sensors, and connectivity make up the Internet of Things. This lets them share data. This network is also known as the IoT. M2M communication uses Bluetooth, NFC, and cellular networks to automatically exchange data between machines. IoT and M2M communication technologies could transform healthcare, transportation, agriculture, and manufacturing. They also pose security risks (Barki et. al. 2016; Khanna and Kaur 2020).

Many IoT devices lack security. Many devices lack computing power and security, making them hackable and exploitable. IoT devices often have weak or default passwords, making them easy to hack (Castilho, Godoy, and Salmen 2020).

IoT standardisation is another issue. IoT devices use many protocols and technologies, making ecosystem-wide security difficult to implement. Device manufacturers and users must design and secure their devices to

address these security issues. Strong and unique passwords, updated software and firmware, and secure network configuration are examples.

Governments and industry must also create and enforce IoT and M2M security standards. This can help ensure devices are built with adequate security and that users have the information they need to make informed security decisions.

Security is a major concern when it comes to the growth of M2M (machine-to-machine) communication. M2M systems often deal with sensitive data and control important systems, and ensuring the security of this data and these systems is critical. (Hongsong, Zhongchuan, and Dongyan 2011; Song et. al. 2014). Some of the security issues that can prevent the growth of M2M include:

Device authentication: Ensuring that only authorized devices are able to access the M2M network is a key security challenge. This requires robust authentication mechanisms to be in place.

Data integrity: M2M communication involves the exchange of large amounts of data, and it is important to ensure that this data has not been tampered with or altered during transmission.

Data privacy: M2M communication often involves sensitive data, such as personal information or industrial secrets. Ensuring the privacy of this data is a critical security concern.

Network security: M2M communication relies on network connectivity, and securing the network against attacks is a key concern. This includes protecting against malware, Denial of Service (DoS) attacks, and unauthorized access.

Device management: Ensuring that M2M devices are properly configured and managed is important for security. This includes keeping devices up to date with the latest software and security patches, as well as monitoring for any unusual behavior.

If these security issues are not properly addressed, it can create risks and vulnerabilities that could discourage organizations from adopting M2M systems.

The security challenges faced by different types of networks, such as those used for WSN and M2M, are similar in both areas. As with other new fields, the efforts in the security of M2M are mainly focused on mapping existing network components to specific security requirements. Unfortunately, the security solutions that are used for wireless sensor networks are not designed for the specific needs of M2M networks. This means that a comprehensive analysis of the security issues faced by

these types of networks is required(Imani et. al. 2019) transporting, and managing information in the Internet of things (IoT).

2. Related Work

S. Sridhar et. al. (Sridhar and Smys 2017) suggest using asymmetric key encryption to protect a system from eavesdropping, DDoS, and quantum attacks. The protocol generates keys and establishes a mutual authentication algorithm between services and devices using a Device ID.

Control network devices used in the Internet of Things (IoT) and Machine-to-Machine (M2M) require a secure, easy-to-use Middleware Security solution, according to Sergio D. Castilho et. al. (2020). This implementation supports CoAP, MQTT, and others. It can communicate with other devices and has a network address translator, IPTable firewall, and encryption software.

Saadia Malik et. al. (2022) create a CloudIoT integration that lets devices communicate securely and efficiently. It will also cover device security methods. Several papers demonstrate how to manipulate the digital M2M ecosystem.

Nenad Gligoric et. al. (2012) propose a hybrid application layer to protect SMS signatures. Secret key, device IMEI, and payload calculate the signature. This protects the message from fake signatures. Android-based mobile devices performed well against a non-secure payload in a live GSM network. Message response times and bandwidth were unaffected.

M. Hossein Ahmadzadegan et. al. (2014) developed a triangular framework for multi-purpose M2M communication. It shows how this platform can develop and deploy apps.

S. Satheesh Kumar et. al. (2021) propose using ECDSA to encrypt key to improve RC4. This process allows XOR operations. SHA-256 secures incoming data. The proposed model achieved 631 ms encryption and 616 ms decryption faster than other methods in experiments. It outperforms other methods in throughput.

Blockchain technology addresses CPS machine communication security. Shiyong Yin et. al. (Yin et. al. 2017) created a blockchain for the M2M communication system to secure and stabilise private and public spaces.

Bayu Anggorojati et. al. (2018) propose an Elliptic Curve cryptography-based capability-based key management system for a local M2M cloud platform. Implementing it in the platform's security manager tests its feasibility. Performance is assessed through experiments.

Sarfraz Alam et. al. (Alam, Chowdhury, and Noll 2011) create a multi-tiered architecture to safely transfer data between layers of the Internet of Things ecosystem. The semantic overlay—rules and ontology—is the most crucial part of this framework. A machine-to-machine communication platform can address security interoperability.

DongBum Seo et. al. (2016) create a context-aware framework to collect and deliver context data to mobile apps. Simulations and experiments can assess system performance. CI-UCEMA lets mobile apps run on cloud platforms. Cloud and ubiquitous computing are powerfully combined.

3. M2M network architecture

A machine-to-machine (M2M) network architecture is a system that enables devices to communicate and exchange data with each other without the need for human intervention. As shown in fig.1, an M2M network architecture typically consists of the following components:

M2M devices: These are the devices that are participating in the M2M communication system. Examples of M2M devices include sensors, actuators and control systems.

Network infrastructure: This is the infrastructure that is used to connect the M2M devices in the network. This can include wired or wireless networks, such as cellular networks, WiFi networks, or Bluetooth networks.

M2M gateway: This is a device that acts as an intermediary between the M2M devices and the network infrastructure. The M2M gateway is responsible for translating the data transmitted between the M2M devices into a format that can be transmitted over the network infrastructure.

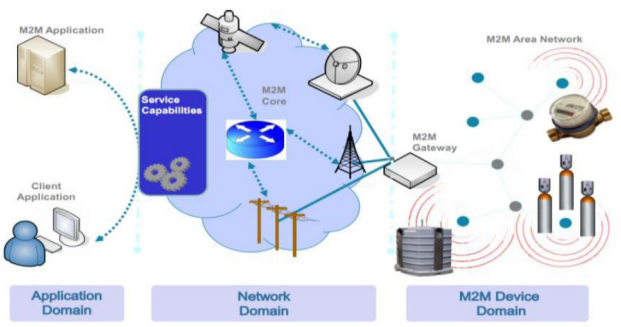


Figure 1

M2M network architecture

M2M server: This is a centralised server that is in charge of storing and processing the information that is passed between the M2M devices. The management of the M2M communication system and the coordination of the activities of the M2M devices may also fall under the purview of the M2M server.

An M2M network architecture is a system that enables M2M devices to communicate and exchange data with each other, enabling them to perform actions and make decisions without human intervention.

3.1 *Importance of secure framework for M2M architecture*

A security framework for M2M communication is important for several reasons:

Confidentiality: M2M communication often involves the transmission of sensitive data, such as financial transactions or personal information. A security framework helps to ensure that this data is kept confidential and is not accessible to unauthorized parties.

Integrity: A security framework helps to ensure that the data transmitted between devices is accurate and has not been tampered with. This is important for maintaining the integrity of the M2M communication system and ensuring that the data being transmitted can be trusted.

Availability: A security framework helps to ensure that the M2M communication system is available to authorized devices when needed. Without a security framework, the system may be vulnerable to attacks or other disruptions that could prevent authorized devices from accessing it.

Trust: A security framework helps to build trust in the M2M communication system. By demonstrating that the system is secure and that the data transmitted between devices is protected, stakeholders are more likely to trust the system and use it for critical communication and data exchange.

Overall, a security framework for M2M communication is important for protecting the confidentiality, integrity, and availability of the system and for building trust in the system among stakeholders.

3.2 *M2M security requirements and vulnerabilities*

The security of M2M communication is important because it involves the exchange of sensitive data and the control of critical systems. M2M communication systems must be designed with security in mind to protect against unauthorized access and data breaches. Some of the key security

requirements and vulnerabilities of M2M communication are described below.

Security requirements:

Confidentiality: M2M communication systems must protect the confidentiality of data that is transmitted and stored. This includes protecting data from unauthorized access or tampering.

Integrity: M2M communication systems must ensure that data is not altered or corrupted during transmission.

Authentication: M2M communication systems must verify the identity of devices and users before allowing access to the network.

Non-repudiation: M2M communication systems must provide proof of the authenticity of data to prevent parties from denying their involvement in a transaction.

Access control: M2M communication systems must implement measures to prevent unauthorized access to the network and to control the level of access granted to different devices and users.

Vulnerabilities:

Unsecured networks: M2M communication systems that are not properly secured may be vulnerable to hacking and other cyber-attacks.

Unpatched software: M2M devices and systems that are not regularly updated with security patches may be vulnerable to exploits and attacks.

Lack of encryption: M2M communication systems that do not use encryption to protect data during transmission may be vulnerable to interception and tampering.

Weak passwords: Devices that use weak passwords or do not require password authentication may be vulnerable to unauthorized access.

Insufficient access controls: M2M communication systems that do not implement proper access controls may be vulnerable to unauthorized access or manipulation.

To secure M2M communication systems, it is important to implement measures such as encryption, secure authentication, and access controls. It is also important to regularly update software and devices with security patches and to educate users on the importance of security in M2M communication.

4. Proposed framework

Proposed design of framework for M2M communication that uses an ultra-lightweight block cipher(Salunke 2021) and the Rubik's cube encryption algorithm(Salunke, Mahalle, and Shinde 2022). A block cipher is a type of encryption algorithm that operates on fixed-size blocks of data and uses a secret key to transform the data into an encrypted form. An ultra-lightweight block cipher is a type of block cipher that is designed to be efficient in terms of computation and memory requirements, making it suitable for use in resource-constrained M2M devices.

The Rubik's cube encryption algorithm is a novel approach to encryption that is based on the concept of rearranging the faces of a Rubik's cube to create a unique key. The algorithm uses the position and orientation of the cube's faces as the key, which can be changed and updated dynamically to provide ongoing security.

Figure 2 illustrates the comprehensive flow of the proposed method. It visually represents the sequence and steps involved in the methodology. To implement this framework, the M2M devices would need to be equipped with the necessary hardware and software to perform the ultra-lightweight block cipher and Rubik's cube encryption. The devices would then use the block cipher to encrypt data that is transmitted over the network, and the Rubik's cube encryption algorithm to generate and update the key for the

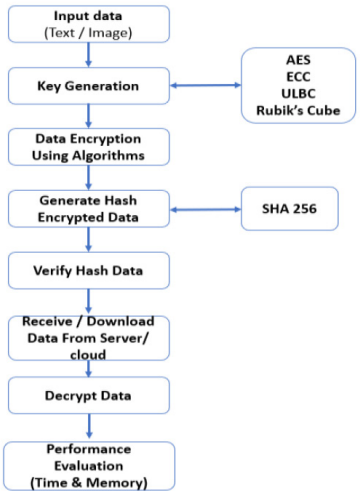


Figure 2

Proposed method flow

block cipher. This framework could secure M2M communication while conserving resources. In M2M communication systems, key management using the Rubik's cube encryption algorithm may be novel. However, the efficacy and security of any encryption framework depend on its implementation and algorithm strength.

SHA-256, a cryptographic hash function, generates a fixed-size, unique representation of a message or data. It is a Secure Hash Algorithm (SHA) defined in FIPS 180-4, which is widely used for secure data communication and data integrity. SHA-256 generates fixed-size hash values for messages and data. Since any data change changes the hash value, it can be used to verify data integrity. SHA-256 is secure because it is hard to generate a collision (two messages with the same hash value) or reverse the hash value back to the original message.

AES, a popular symmetric encryption algorithm, uses a fixed-length key to encrypt and decrypt data. It replaced the weak and attackable Data Encryption Standard (DES). Block cyphers like AES process data in 128-bit blocks. It performs substitution, permutation, and XOR on data. The key controls data transformation.

AES is fast and efficient, making it ideal for data storage and secure communication. VPNs, disc encryption, and SSL/TLS protocols use it. AES has never been cracked. It protects sensitive data for many governments and militaries.

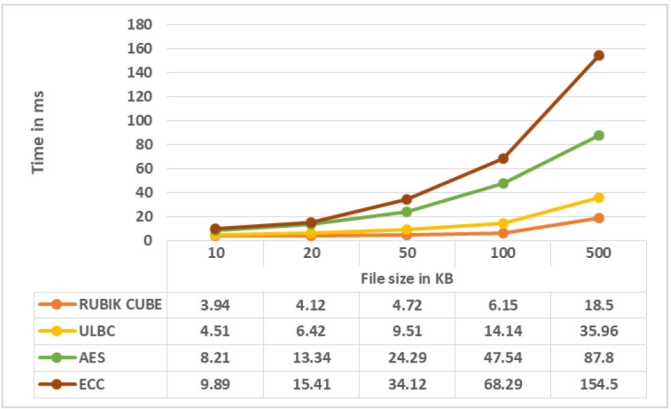
Elliptic curves over finite fields form the basis of ECC, a public-key cryptography. It encrypts, decrypts, and verifies digital signatures. ECC uses elliptic curve keys for encryption and decryption. The private key is hidden, while the other is public (the public key). Encrypting a message with the recipient's public key requires the recipient's private key to decrypt.

ECC offers strong security with short key lengths. It is ideal for mobile devices and other resource-constrained environments where key storage or transmission is a concern. Secure communication, authentication, and data protection all use ECC. SSL/TLS protocols, smart cards, and mobile phones use it too.

5. Performance Evaluation

5.1 *Encryption Time Analysis*

The time it takes to encrypt and decrypt data using a particular encryption algorithm can vary depending on a number of factors,



	10	20	50	100	500
RUBIK CUBE	3.94	4.12	4.72	6.15	18.5
ULBC	4.51	6.42	9.51	14.14	35.96
AES	8.21	13.34	24.29	47.54	87.8
ECC	9.89	15.41	34.12	68.29	154.5

Figure 3

Data Encryption Time comparison

including the type of algorithm being used, the size of the data being encrypted, the processing power of the device or computer being used, and the implementation of the algorithm. In general, symmetric encryption algorithms, such as AES, tend to be faster than asymmetric algorithms, as compared to ECC, because they use the same key for both encryption and decryption; here algorithm based on Rubik's cube outperform all other methods. The size of the data being encrypted can also affect the encryption time as shown in Figure 3. Larger amounts of data will typically take longer to encrypt than smaller amounts. The processing power of the device or computer being used can also play a role, as more powerful devices may be able to encrypt and decrypt data more quickly.

5.2 *Decryption Time Analysis*

Decryption time analysis refers to the process of measuring the amount of time it takes to decrypt encrypted data. This analysis can be used to evaluate the effectiveness of different encryption algorithms or to assess the performance of a particular decryption system. Decryption time analysis can be useful in a variety of contexts, such as when selecting an encryption algorithm for a particular application or when designing a decryption system for use in a high-security environment as shown in Figure 4. In general, it is important to carefully consider decryption time when selecting and implementing encryption technologies, as faster decryption times may be more desirable in some situations, while slower decryption times may be more secure.

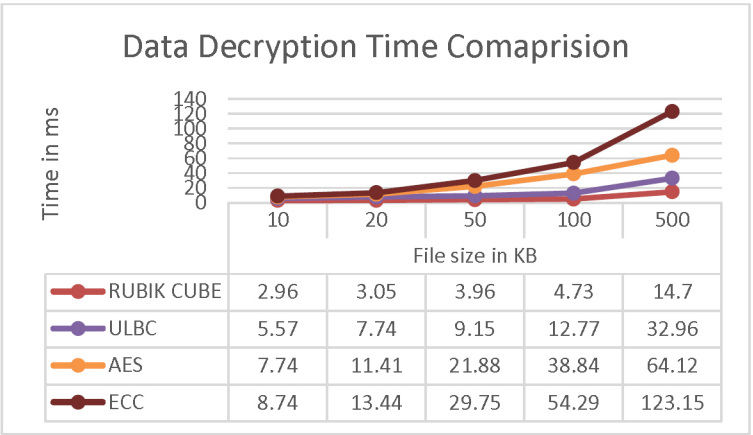


Figure 4

Data Decryption Time Comparison

5.3 *Key Generation Time*

Key generation time refers to the amount of time it takes to generate a key for use in an encryption or decryption process. The key generation process involves creating a series of random numbers or characters that will be used to encode or decode the data as shown in Figure 5. Key generation time is an important consideration when selecting an encryption algorithm or designing a system that uses encryption. In general, faster key generation times may be more desirable for applications that require frequent key generation, such as in real-time communication systems.

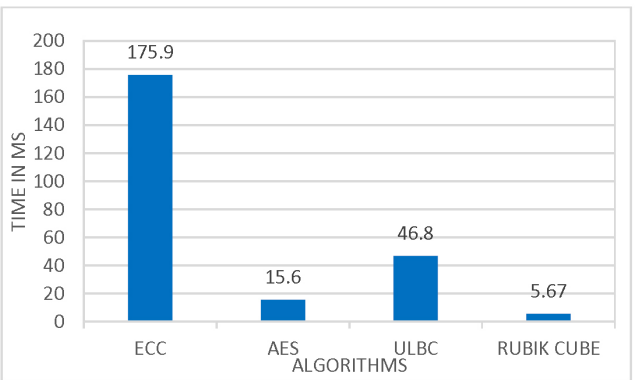


Figure 5

Key Generation Time

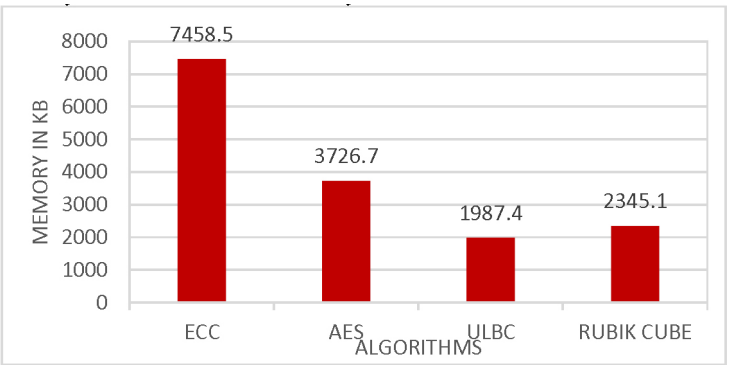


Figure 6

Memory Utilization Comparison

5.4 *Memory Utilization*

Memory utilization refers to the amount of memory that is used by a system or process. In the context of encryption and decryption, memory utilization may be an important consideration when selecting an encryption algorithm or designing a system that uses encryption. It is important to carefully consider memory utilization when selecting and implementing encryption technologies, as high memory utilization may limit the performance of the system or cause it to crash as shown in Figure 6. In general, algorithms that are designed to be memory-efficient may be more suitable for systems with limited memory resources.

6. Conclusion

Based on the information provided, it seems that the conclusion of the research project is that the Rubik's Cube encryption algorithm outperformed the other three encryption models (AES, ECC, and ultra-lightweight block cipher) in the embedded security framework for M2M communication. This conclusion is likely based on the results of various performance and security evaluations that are conducted as part of the paper. It is important to note that the suitability of an encryption algorithm for a particular application depends on a variety of factors, such as the security requirements of the system, the performance requirements, and the available resources (e.g., computational power, memory). Therefore, the conclusion that Rubik's Cube encryption is the best option for this specific M2M communication system may not necessarily hold true for all M2M communication systems or other types of applications.

References

[1] Alam, Sarfraz, Mohammad M. R. Chowdhury, and Josef Noll. "Interoperability of Security-Enabled Internet of Things." Wireless Personal Communications 61(3) : 567–86 (2011).

[2] Anggorojati, Bayu, Neeli Rashmi Prasad, and Ramjee Prasad. "Capability-Based Access Control with ECC Key Management for the M2M Local Cloud Platform." Wireless Personal Communications 100(2) : 519–38 (2018).

[3] Barki, Amira, Abdelmadjid Bouabdallah, Said Gharout, and Jacques Traore. "M2M Security: Challenges and Solutions." IEEE Communications Surveys and Tutorials 18(2) : 1241–54 (2016).

[4] Castilho, Sergio D., Eduardo P. Godoy, and Fadir Salmen. "Implementing Security and Trust in IoT/M2M Using Middleware." International Conference on Information Networking 2020-Janua : 726–31 (2020).

[5] Gligoric, Nenad, Tomislav Dimcic, Dejan Drajic, Srdjan Krco, and Nhon Chu. "Application-Layer Security Mechanism for M2M Communication over SMS." 2012 20th Telecommunications Forum, TELFOR 2012 - Proceedings 7:5–8 (2012).

[6] Hongsong, Chen, Fu Zhongchuan, and Zhang Dongyan. "Security and Trust Research in M2M System." Proceedings of 2011 IEEE International Conference on Vehicular Electronics and Safety, ICVES 2011 (90818016) : 286–90 (2011).

[7] Hossein Ahmadzadegan, M., and Tapio Fabritius. "A Multi-Purpose Triangular Framework for M2M Communication Security." 2014 World Congress on Computer Applications and Information Systems, WCCAIS 2014 (2014).

[8] Imani, Amirhosein, Alireza Keshavarz-Haddad, Mohsen Eslami, and Javad Haghighat. "Security Challenges and Attacks in M2M Communications." 9th International Symposium on Telecommunication: With Emphasis on Information and Communication Technology, IST 2018 264–69 (2019).

[9] Irshad, Mohammad. "A Systematic Review of Information Security Frameworks in the Internet of Things (IoT)." Proceedings - 18th IEEE International Conference on High Performance Computing and Communications, 14th IEEE International Conference on Smart City

and 2nd IEEE International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2016 1270–75 (2019).

[10] Khanna, Abhishek, and Sanmeet Kaur. Internet of Things (IoT), Applications and Challenges: A Comprehensive Review. Vol. 114. Springer US (2020).

[11] Kumar, S. Satheesh, and Manjula Sanjay Koti. "An Hybrid Security Framework Using Internet of Things for Healthcare System." Network Modeling Analysis in Health Informatics and Bioinformatics 10(1) (2021).

[12] Malik, Saadia, Nadia Tabassum, Muhammad Saleem, Tahir Alyas, Muhammad Hamid, and Umer Farooq. "Cloud-Iot Integration: Cloud Service Framework for M2m Communication." *Intelligent Automation and Soft Computing* 31(1) : 471–80 (2022).

[13] Salunke, Mahendra Balkrishna. "Ultra-Lightweight Block Cipher in Medical Internet of Things for Secure Machine-to-Machine Communication Using FPGA." Revista Gestão Inovação e Tecnologias 11(4) : 236–51 (2021).

[14] Salunke, Mahendra Balkrishna, Parikshit Narendra Mahalle, and Gitanjali Rahul Shinde. "Intelligent Systems and Applications In Engineering Rubik ' s Cube Encryption Algorithm-Based Technique for Information Hiding During Data Transmission in Sensor-Based Networks." 10 : 429–39 (2022).

[15] Seo, Dong Bum, You Boo Jeon, Song Hee Lee, and Keun Ho Lee. "Cloud Computing for Ubiquitous Computing on M2M and IoT Environment Mobile Application." Cluster Computing 19(2) : 1001–13 (2016).

[16] Song, Jaeseung, Andreas Kunz, Mischa Schmidt, and Piotr Szczytowski. "Connecting and Managing M2M Devices in the Future Internet." Mobile Networks and Applications 19(1) : 4–17 (2014).

[17] Sridhar, S., and S. Smys. "Intelligent Security Framework for IoT Devices." International Conference on Inventive Systems and Control (ICISC-2017) Intelligent 1–5 (2017).

[18] Yin, Shiyong, Jinsong Bao, Yiming Zhang, and Xiaodi Huang. "M2M Security Technology of CPS Based on Blockchains." Symmetry 9(9) (2017).