Volume: 8 Issue: VII Month of publication: July 2020

DOI: https://doi.org/10.22214/ijraset.2020.30754

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VII July 2020- Available at www.ijraset.com

Data Compression over Network using IOT

Puneeth S P[1], Gayathri M[2], Kavya S A[3], Poornima C S[4], Varshitha H[5]
[1, 2, 3, 4, 5]*Department of Information Science and Engineering, BIET Davanagere*

Abstract: Energy is the most important resource in state-of-the-art Internet of Things solutions. There are a lot of concepts and techniques dedicated to save energy, mainly focused to reduce transmission, since the energy used for pre-processing (encoding) is incomparable smaller than energy used for broadcasting. If applications do not require real-time measurements, data compression is one solution to energy saving problem. In this project we develop new coding scheme for data compression that can be used for efficient data compression of temporally correlated data, such as temperature measurements coming from different smart devices. Along with reducing weightage of data we also provide security by encrypting it.
Keywords: correlated data and Encryption.

I. INTRODUCTION

The combination of information technologies and advanced communication and sensing systems, creates a variety of new potential applications under the umbrella of Internet of Things (IOT). IOT represents a worldwide network of uniquely addressable interconnected smart objects or smart devices. In IOT, smart objects have its own processor, memory and module for communication (usually wireless).

The power supply for the devices can be provided from the traditional electric grid, but also, many devices come with battery or distributed renewable power supply, such as solar panel.

Energy saving for wireless devices has been very challenging problem for decades. In IOT context, energy saving is not only important from power supply perspective, but also from network perspective. Namely, new applications are constantly fed with raw data coming from many sensors. Therefore, reducing network traffic is very important in order to avoid saturation and to achieve many devices to work cooperatively within the same data hub.

Sensors are now pervasive and ubiquity of sensors makes IOT applications useful as well as exotic. IOT applications pervade inside human body, inside luxury (even low cost) cars to all the aspects of human life. Such huge amount of sensors generate huge amount of data. Storage, transmission and processing of such high volume data pose potentially destructive performance and scalability risk; tiny sensors require large buffer, transmission of high volume data results in reduction of battery lifetime, higher processing power is needed by the tiny relay node for in-network processing. In order to harvest IOT applications like smart energy management, elderly monitoring, e-health care, data volume is to be reduced such that utility is optimized while maximizing the compression. Extensive research works already proposed various data compression methods particularly for sensor data. Data compression is suitable approach in applications were data are not needed in real time.

Security zone IoT can be broadly categorized as, device security, network security, service security, and the infrastructure security technology. Device security, up to the gateway and high-performance device from ultra-lightweight / low power consumption sensor, security operating system with security technology that specializes in devices with a variety of resources / performance, security SoC, security device platforms such as it is included. Network security, hardware resources, and communication systems, in environments where security structure is connected / linked between different networks, in response to a variety of attacks, is a technique for providing reliable end-to-end communications. Security technology services, smart home, smart healthcare, smart car / transportation, such as smart energy, is a security technology that is specialized to meet the security requirements of a variety of IOT applications services. Other IOT environment to the applicable lightweight password, authentication platform, also fundamental technology, such as privacy protection there.

A. Objectives

The proposed system will accomplish the following objectives:

1) To provide the privacy for the data while transmission using protocol.
2) To compress the data after automatically compressed by protocol by considering the storage medium without loss of actual information.
3) To perform comparative analysis of the data compressed.

B. Proposed System

We propose a coding scheme for delta compression, which can be used for efficient data compression of temporally correlated data, such as temperature measurements coming from different smart devices. The block diagram is shown in figure below

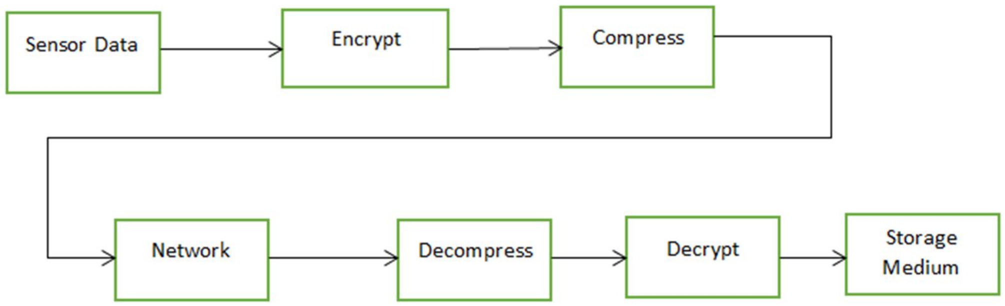


Fig. 1 Block diagram of proposed system

With statistical processing of a big amount of data, we can get more accurate stochastic model of the process. For this purpose, we collected "raw" temperature measurements from IOT devices. Temperature process is a slow changing process and there is a temporal correlation, i.e. temperature value of the next sampling moment depends on the temperature value of previous sampling moment. If the output is 10 bits, up to 1024 states (symbols) can be generated. Each symbol from the output is statistically depended, so the next state depends of the previous states and not only from the probability of the symbol. If calculating Shannon entropy of this source is possible, then with proper coding of these symbols, it would be possible to decrease average length of the code word and to achieve efficient data transfer.

We implement Brotli a compression algorithm. It is developed by Google and serves best for text compression. The reason being, it uses a dictionary of common keywords and phrases on both client and server side and thus gives a better compression ratio. It is supported by all major browsers. Along with this we use data encryption algorithm for preserving data privacy.

II. LITERATURE REVIEW

In 2006, Hadim, S., Mohamed, N made survey which focuses on the current state of research in middleware design and general issues in designing a middleware for WSN. It also examines the various approaches of middleware design, compares and suggests different types of applications where each approach can be used.

In 2014, Riahi, A proposed work in which they analyse the role of each of the mentioned actors in IoT security and their relationships, in order to highlight the research challenges and present their approach to these issues based on a holistic vision of IoT security. The interactions of these four IoT components, person, intelligent object, technological ecosystem, and process, highlight a systemic and cognitive dimension within security of the IoT. The interaction of people with the technological ecosystem requires the protection of their privacy. Similarly, their interaction with control processes requires the guarantee of their safety. Processes must ensure their reliability and realize the objectives for which they are designed.

In 2015, Lee, J., Bagheri, B and Kao H propose a next-generation computational framework for dynamic, data-driven optimisation of production. In keeping with the principles of Industry 4.0, their architecture is easy to reconfigure and connect to various hardware/software devices allowing for quick reconfiguration of factories. It has a live representation of the real world enabling mangers and other users to keep track of activities in the factory. It also provides users with statistics and other abstraction tools to augment decision making capabilities and has the ability to automatically allocate resources thereby allowing for nearly autonomous operation of the factory. They further demonstrate an implementation of the proposed architecture on a model problem developed in conjunction with Foxconn.

In 2011, Hachem, S., Teixeira, T and Issarny give an overview of a service-oriented middle- ware solution that addresses those challenges using semantic technologies to provide interoperability and exibility. They especially focus on modelling a set of ontologies that describe devices and their functionalities and thoroughly model the domain of physics.

In 2009, Hamad, F., Smalov, L and James provide research study which focuses on users' perspectives of m-commerce development in China. The development of m-commerce was measured by the extent of m-commerce businesses adopting business intelligence. A research framework was developed for users' perceptions on m-commerce development, survey questionnaires were used to collect data, and ANOVA was used for data analysis.

In 2008, Mathur, S., Trappe,W and Mandayam. N present a protocol that allows two users to establish a common cryptographic key by exploiting special properties of the wireless channel: the underlying channel response between any two parties is unique and decor relates rapidly in space. The established key can then be used to support security services (such as encryption) between two users. Their algorithm uses level-crossings and quantization to extract bits from correlated stochastic processes.

In 2008, Montenegro**,** G and Castelluccia C describe a method for establishing authenticated channels in a wireless ad-hoc network. The presented protocol is fully self-organized, and it also provides an identification framework. The two main contributions are (1) a fully self-organized protocol that establishes an authenticated communication channel between nodes of a wireless ad-hoc network and (2) a secure identifier framework that is resilient to impersonation. The authenticated channel provided by the protocol can then be used to establish a secret communication channel between nodes.

In 2013, Xiaohui X provides an overview of IoT, IoT architecture, key technologies in IoT and application scenarios of IoT. Various security issues and challenges in the IoT environment are also discussed and presented.

In 2012, Suo H research and provide brief review on the progress of IoT, and pay attention to the security. By means of deeply analyzing the security architecture and features, the security requirements are given. On the basis of these, they discuss the research status of key technologies including encryption mechanism, communication security, protecting sensor data and cryptographic algorithms, and briefly outline the challenges.

In 2012, Haitao, L.I.U.B.C.H.W., Ying, F.U analyses the security issues and challenges and provides a well-defined security architecture as a confidentiality of the user's privacy and security which could result in its wider adoption by masses.

In 2013, Jiafu Wan, Min Chen, Feng Xia, Di Li, and Keliang Zhou first present the correlations among machine-to-machine (M2M), wireless sensor networks (WSNs), CPS and internet of things (IoT), and introduce some research activities in M2M, including M2M architectures and typical applications. Then, they review two CPS platforms and systems that have been proposed recently, including a novel prototype platform for multiple unmanned vehicles with WSNs navigation and cyber transportation systems. Through these reviews, they propose CPS is an evolution of M2M by the introduction of more intelligent and interactive operations, under the architecture of IoT.

III.SYSTEM DESIGN

The data from the sensor will get encrypted using AEN algorithm. Then it is compressed again with Brotil compression technique. Later, the data is sent to the destination address through network layer. Their data packet is again decompressed and decrypted before storage. The figure below will show the architecture of our proposed system. The Advanced Encryption Standard (AES) is a symmetric block cipher chosen by the U.S. government to protect classified information. AES is implemented in software and hardware throughout the world to encrypt sensitive data. It is essential for government computer security, cyber security and electronic data protection. The National Institute of Standards and Technology (NIST) started development of AES in 1997 when it announced the need for an alternative to the Data Encryption Standard (DES), which was starting to become vulnerable to brute-force attacks. AES was created for the U.S. government with additional voluntary, free use in public or private, commercial or non-commercial programs that provide encryption services.

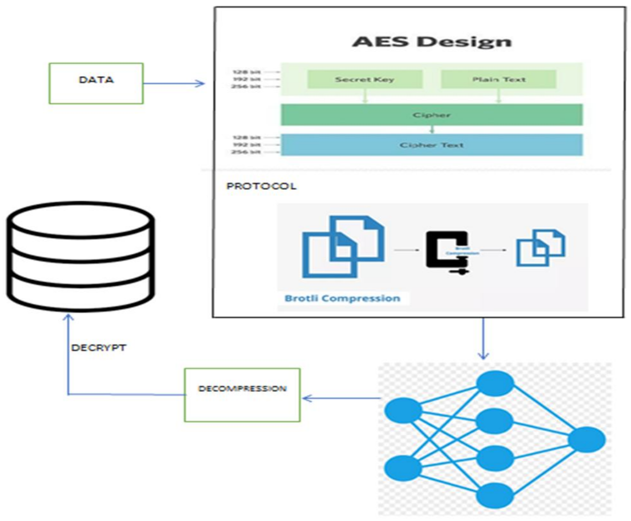


Fig. 2 Architecture of proposed system

Brotli compression is an open source compression algorithm developed by Google to help further reduce the size of files. Google released a different compression algorithm in 2013 called Zopli to perform "very good but slow deflate or Zlib compression". However, based on a compression algorithm study done at Google, Brotli came out significantly faster with a 20-26% increase in compression ratio compared to Zopli.

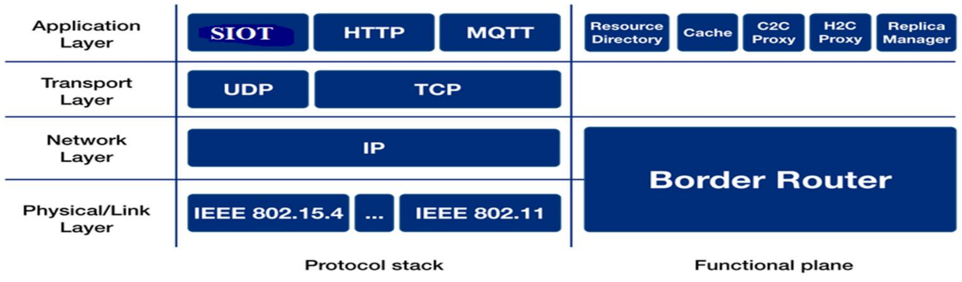


Fig. 3 Steps in Data packet Transmission in different layers

IV. IMPLEMENTATION

A. AES Algorithm

AES includes three block ciphers:

1) AES-128
2) AES-192
3) AES-256.

AES-128 uses a 128-bit key length to encrypt and decrypt a block of messages, while AES-192 uses a 192-bit key length and AES-256 a 256-bit key length to encrypt and decrypt messages. Each cipher encrypts and decrypts data in blocks of 128 bits using cryptographic keys of 128, 192 and 256 bits, respectively.

Symmetric, also known as secret key, ciphers use the same key for encrypting and decrypting, so the sender and the receiver must both know -- and use -- the same secret key. The government classifies information in three categories: Confidential, Secret or Top Secret. All key lengths can be used to protect the Confidential and Secret level. Top Secret information requires either 192- or 256-bit key lengths.

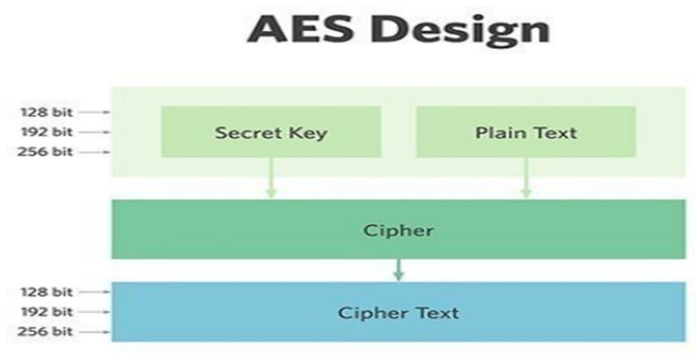


Fig. 4 the relationships between secret key, plaintext, cipher and cipher text

There are 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. A round consists of several processing steps that include substitution, transposition and mixing of the input plaintext to transform it into the final output of cipher text. The AES encryption algorithm defines numerous transformations that are to be performed on data stored in an array. The first step of the cipher is to put the data into an array -- after which, the cipher transformations are repeated over multiple encryption rounds.

The first transformation in the AES encryption cipher is substitution of data using a substitution table; the second transformation shifts data rows, and the third mixes columns. The last transformation is performed on each column using a different part of the encryption key. Longer keys need more rounds to complete.

NIST specified the new AES algorithm must be a block cipher capable of handling 128-bit blocks, using keys sized at 128, 192 and 256 bits. Other criteria for being chosen as the next AES algorithm included the following:

Security. Competing algorithms were to be judged on their ability to resist attack -- as compared to other submitted ciphers. Security strength was to be considered the most important factor in the competition.

Cost. Intended to be released on a global, nonexclusive and royalty-free basis, the candidate algorithms were to be evaluated on computational and memory efficiency.

Implementation. Factors to be considered included the algorithm's flexibility, suitability for hardware or software implementation, and overall simplicity.

Difference between AES-128 and AES-256

Overall, security experts consider AES safe against brute-force attacks, in which all possible key combinations are checked until the correct key is found. However, the key size employed for encryption needs to be large enough so that it cannot be cracked by modern computers, even considering advancements in processor speeds based on Moore's law.

A 256-bit encryption key is significantly more difficult for brute-force attacks to guess than a 128-bit key; however, because the latter takes so long to guess, even with a huge amount of computing power, it is unlikely to be an issue for the foreseeable future, as a hacker would need to use quantum computing to generate the necessary brute force.

Still, 256-bit keys also require more processing power and can take longer to execute. When power is an issue -- particularly on small devices -- or where latency is likely to be a concern, 128-bit keys are likely to be a better option.

When hackers want to access a system, they will aim for the weakest point, which is typically not the encryption, regardless of whether it's a 128-bit key or a 256-bit key. Users should make sure the software under consideration does what they want it to do, that it protects user data in the way it's expected to and that the overall process has no weak points.

Additionally, there should be no gray areas or uncertainty about data storage and handling. For example, if data resides in the cloud, users should know the location of the cloud. Most importantly, the security software that has been selected should be easy to use to ensure that users do not need to perform unsecure workarounds to do their jobs.

AES is used widely for protecting data at rest. Applications for AES include self-encrypting disk drives, database encryption and storage encryption. On the other hand, the RSA (Rivest-Shamir-Adleman) algorithm is often used in web browsers to connect to websites, in virtual private network (VPN) connections and in many other applications.

Unlike AES, which employs symmetric encryption, RSA is the base of asymmetric cryptography. Symmetric encryption involves converting plaintext to ciphertext using the same key, or secret key, to encrypt and decrypt it. On the other hand, the term asymmetric comes from the fact that there are two related keys used for encryption: a public and a private key. If encryption is performed with the public key, decryption can only happen with the related private key and vice versa. Typically, RSA keys are employed when there are two separate endpoints.

While RSA encryption works well for protecting the transfer of data across geographic boundaries, its performance is poor. The solution is to combine RSA encryption with AES encryption in order to benefit from the security of RSA with the performance of AES. This can be accomplished by generating a temporary AES key and protecting it with RSA encryption.

B. Brotli Compression

The Brotli compression algorithm is a noteworthy advancement since by reducing file sizes it will help make the web a faster and more enjoyable place for users - especially true for mobile users. As Zoltan Szabadka, a software engineer on Google's compression team said,

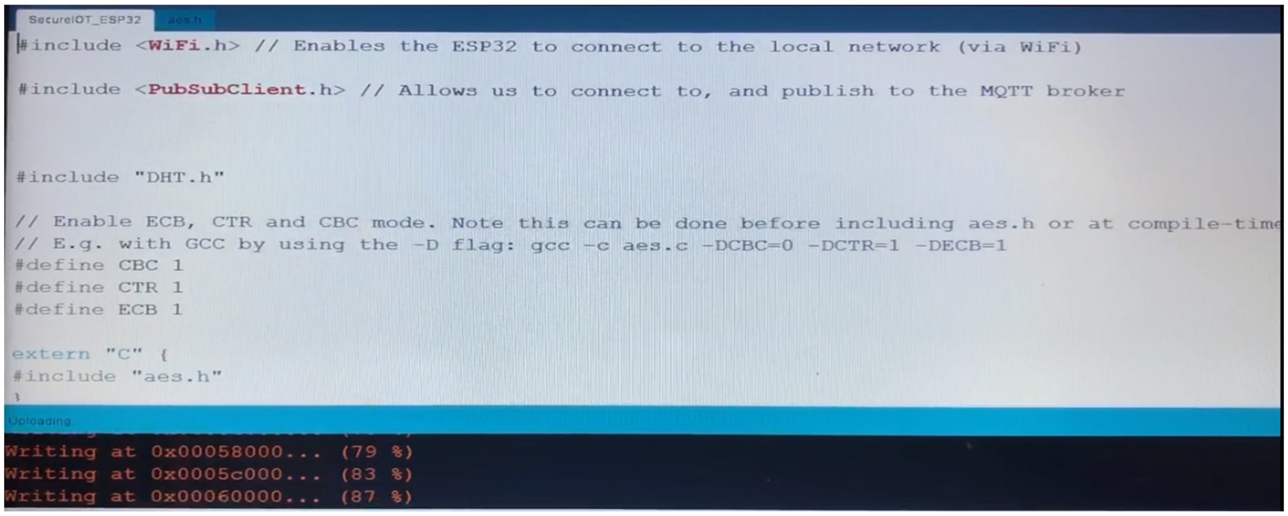
Compression has come a long way in the past couple of years and Brotli is now at the forefront in that category. Here are a few of the highlights that make Brotli a leader in compression:

1) Brotli is independent of CPU type, operating system, file system, and character set.
2) It can produce a compression ratio that is comparable to the best compression methods currently available and most importantly is considerably better than Gzip.
3) It decompresses much faster than current LZMA implementation.

ECB (Electronic Codebook) is essentially the first generation of the AES. It is the most basic form of block cipher encryption. CBC (Cipher Blocker Chaining) is an advanced form of block cipher encryption. With CBC mode encryption, each cipher text block is dependent on all plaintext blocks processed up to that point. This adds an extra level of complexity to the encrypted data.

AES-CTR mode has many properties that make it an attractive encryption algorithm for in high-speed networking. AES-CTR uses the AES block cipher to create a stream cipher. Data is encrypted and decrypted by XOR with the key stream produced by AES encrypting sequential counter block values. AES-CTR is easy to implement, and AES-CTR can be pipelined and parallelized. AES-CTR also supports key stream precomputation.

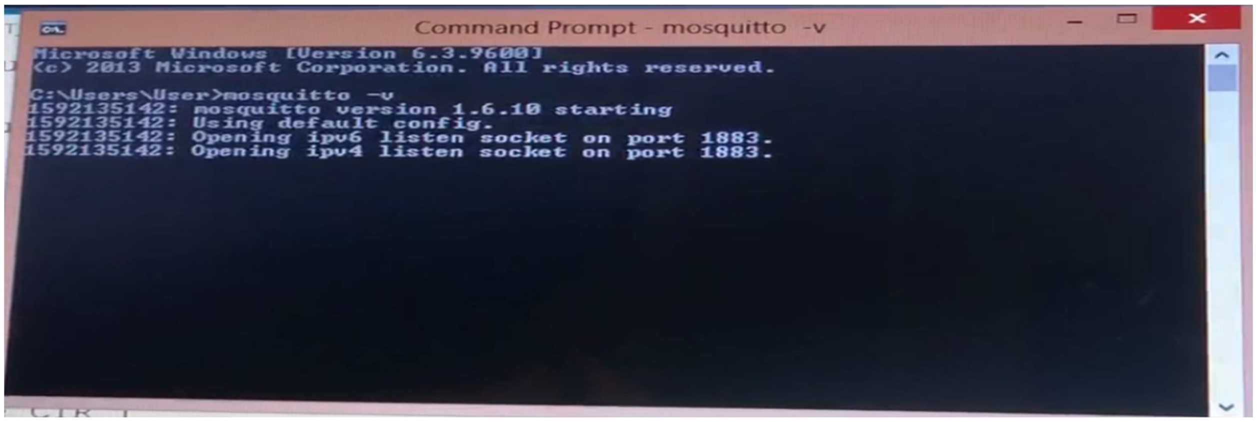
V. RESULTS AND SNAPSHOTS



```
#include <WiFi.h> // Enables the ESP32 to connect to the local network (via WiFi)

#include <PubSubClient.h> // Allows us to connect to, and publish to the MQTT broker


#include "DHT.h"

// Enable ECB, CTR and CBC mode. Note this can be done before including aes.h or at compile-time
// E.g. with GCC by using the -D flag: gcc -c aes.c -DCBC=0 -DCTR=1 -DECB=1
#define CBC 1
#define CTR 1
#define ECB 1

extern "C" {
#include "aes.h"
}
```

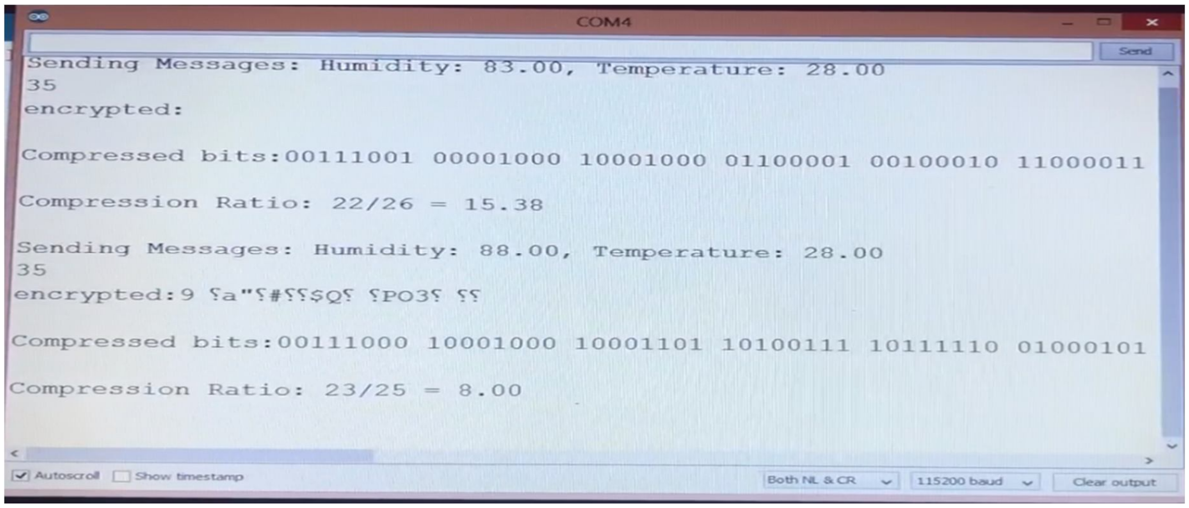
```
Writing at 0x00058000... (79 %)
Writing at 0x0005c000... (83 %)
Writing at 0x00060000... (87 %)
```

Fig. 5 uploading code to ESP32 IOT device



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\User>mosquitto -v
1592135142: mosquitto version 1.6.10 starting
1592135142: Using default config.
1592135142: Opening ipv6 listen socket on port 1883.
1592135142: Opening ipv4 listen socket on port 1883.
```

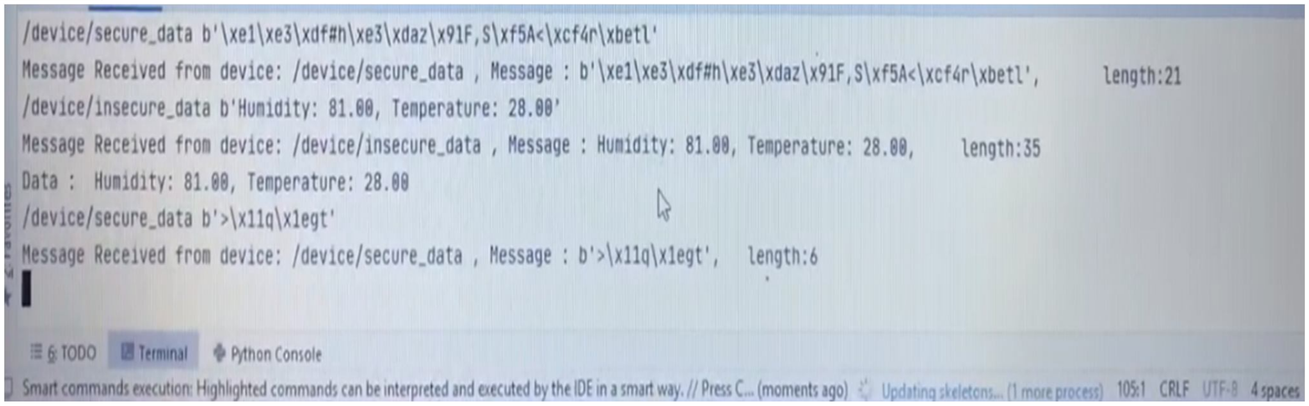
Fig. 6 Starting MQTT broker/server



```
Sending Messages: Humidity: 83.00, Temperature: 28.00
35
encrypted:

Compressed bits:00111001 00001000 10001000 01100001 00100010 11000011

Compression Ratio: 22/26 = 15.38

Sending Messages: Humidity: 88.00, Temperature: 28.00
35
encrypted:9 ʕa"ʕ#ʕʕ$Qʕ ʕPO3ʕ ʕʕ

Compressed bits:00111000 10001000 10001101 10100111 10111110 01000101

Compression Ratio: 23/25 = 8.00
```

Fig. 7 sending message through network

Fig.7 shows the temperature data which is collected in real time through DHT sensor. It also contains compressed bits, compression ratio and size of the data.



```
/device/secure_data b'\xe1\xe3\xdf#h\xe3\xdaz\x91F,S\xf5A<\xcf4r\xbetl'
Message Received from device: /device/secure_data , Message : b'\xe1\xe3\xdf#h\xe3\xdaz\x91F,S\xf5A<\xcf4r\xbetl',      length:21
/device/insecure_data b'Humidity: 81.00, Temperature: 28.00'
Message Received from device: /device/insecure_data , Message : Humidity: 81.00, Temperature: 28.00,     length:35
Data :  Humidity: 81.00, Temperature: 28.00
/device/secure_data b'>\x11q\x1egt'
Message Received from device: /device/secure_data , Message : b'>\x11q\x1egt',   length:6
```

6: TODO
Terminal
Python Console
Smart commands execution: Highlighted commands can be interpreted and executed by the IDE in a smart way. // Press C... (moments ago)
Updating skeletons... (1 more process)
105:1
CRLF
UTF-8
4 spaces

Fig. 8 Information of data after received from the network

Fig.8 shows the two types of data secure data and insecure data. Secure data is the data received after compression and encryption. Insecure data is the data which is original data received through network from source. Above figure also contains length of data received.

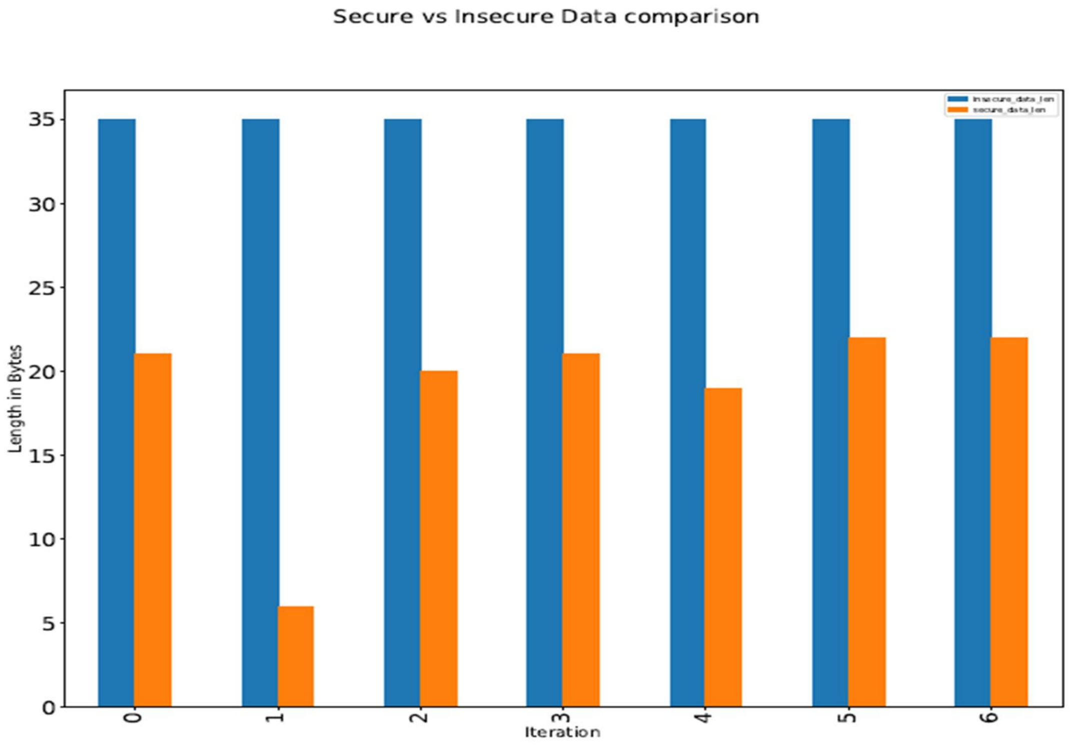


Fig.9 Comparison of secure and insecure data size

Fig.9 shows the comparison graph of secure and insecure data which is received from temperature sensor in real time for every 2 seconds. Above figure shows that our proposed system receives less size of data from original data without information loss.

VI. CONCLUSION

In this era of IOT and omnipresence of sensor devices, the storage, transmission, processing of high volume of sensor data poses potentially destructive performance and scalability risk. Arbitrary (conventional) compression may result in significant intelligence loss. In this project, we proposed a novel sensor data compression scheme that is effective to multitude of sensor data types and yield considerable compression gain. Our proposed system also encrypts the original data so that provides high security for data. We also provides the analysis of secure and insecure data through which we can get the information of size of data is compressed and received.

REFERENCES

[1] Hadim, S., Mohamed, N.:Middleware: middleware challenges and approaches forwireless sensor networks. IEEE distrib. Syst. online. 7(3), 1–10 (2006).
[2] Riahi, A.: A systemic and cognitive approach for IoT security. In: Proceedings of International Conference Computing, Networking, and Communication (2014).
[3] Lee, J., Bagheri, B., Kao, H.-A.: A cyber-physical systems architecture for industry 4.0-based manufacturing systems. Manuf. Lett. 3, 18–23 (2015).
[4] Hachem, S., Teixeira, T., Issarny, V.: Ontologies for the internet of things. In: Proceedings of the 8th Middleware Doctoral Symposium. ACM (2011).
[5] Hamad, F., Smalov, L., James, A.: Energy-aware security in Mcommerce and the internet of things. IETE Tech. Rev. 26(5), 357– 362 (2009).
[6] Mathur, S., Trappe,W., Mandayam, N., Ye, C., Reznik, A.: Radiotelepathy: extracting a secret key from an unauthenticated Wireless channel. In: Proceedings of Mobi-Com, pp. 128–139 (2008).
[7] Montenegro, G., Castelluccia, C.: Crypto-based identifiers (CBIDs): concepts and applications. ACM Trans. Info. Syst. Sec. 7(1), 97–127 (2004).
[8] Xiaohui, X.: Study on security problems and key technologies of the internet of things. In: 2013 Fifth International Conference on Computational and Information Sciences (ICCIS). IEEE (2013).
[9] Suo, H., et al.: Security in the internet of things: a review. In: 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE), Vol. 3. IEEE (2012).
[10] Haitao, L.I.U.B.C.H.W., Ying, F.U.: Security analysis and security model research on IOT. Comput. Digit. Eng. 11(6), 56–61 2012).
[11] WAN, J., et al.: From machine-to-machine communications towards cyber-physical systems. Comput. Sci. Inf. Syst. 10(3), 1105–1128 (2013).