A Distributed Security Mechanism for Resource-Constrained IoT Devices

James King[1] and Ali Ismail Awad[1,2]
[1]Department of Computer Science, Electrical and Space Engineering
Luleå University of Technology, Luleå, Sweden
[2]Faculty of Engineering, Al Azhar University, Qena, Egypt
E-mail: {jamyking@gmail.com}, {ali.awad@ltu.se}

Keywords: internet of things (IoT), IoT security, class-0 IoT devices , object data encryption

Received: October 30, 2015

Internet of Things (IoT) devices have developed to comprise embedded systems and sensors with the ability to connect, collect, and transmit data over the Internet. Although solutions to secure IoT systems exist, Class-0 IoT devices with insufficient resources to support such solutions are considered a resource-constrained in terms of secure communication. This paper provides a distributed security mechanism that targets Class-0 IoT devices. The research goal is to secure the entire data path in two segments, device-to-gateway and gateway-to-server data communications. The main concern in the provided solution is that lighter security operations with minimal resource requirements are performed in the IoT device, while heavier tasks are performed in the gateway side. The proposed mechanism utilizes a symmetric encryption for data objects combined with the native wireless security to offer a layered security technique between the device and the gateway. In the offered solution, the IoT gateways provide additional protection by securing data using Transport Layer Security (TLS). Real-time experimental evaluations have demonstrated the applicability of the proposed mechanism pertaining to the security assurance and the consumed resources of the target Class-0 IoT devices.

Povzetek: V članku je analiziran mehanizem za varen prenos podatkov med napravami interneta stvari (IoT).

1 Introduction

Recently, the Internet of Things (IoT), coined as such in 1999, has become an evolving paradigm in wireless communications [1]. IoT is now a hot topic in Information and Communication Technology (ICT) and has drawn the attention of many research institutions [2, 3]. The generic infrastructure of IoT is a network of devices or objects such as embedded computers, controllable and intelligent automated devices, sensors, and Radio Frequency IDentification (RFID) tags, in addition to the IoT gateway and the remote server. IoT devices have the ability to connect and exchange data with other devices and services over a network and over the global Internet [4, 5]. The deployments of IoT core technology encompass home automation, manufacturing, environmental monitoring, and medical and healthcare systems. A future mega-market is anticipated for a broad scope of applications that utilize IoT devices and technology [1].

The constrained IoT devices, Class-0 IoT devices, are devices with limited or constrained resources with respect to CPU processing power, ROM, RAM, and battery life. However, these devices still have the capability of providing their intended functionalities. The constrained IoT devices are often small in size with limited functions, such as sensors and smart devices controlling electrical appliances or services [6]. They are capable of collecting and transmitting data, such as sensor readings, across the Internet for storage and analysis. The collected and transmitted data may be personal, private, and sensitive. Figure 1 demonstrates a general architecture of an IoT system using an example of constrained IoT medical devices.

Due to a wide range of IoT applications, data security has become a major concern in IoT systems in addition to the system's scalability [7]. Information insecurity will directly impact the performance of the entire IoT system [8]. A study states that 70% of the ordinarily used IoT devices face security vulnerabilities such as insufficient authorization, lack of encryption, and insecure web interfaces [9]. In some application domains such as healthcare, data leakage can threaten the life of individuals. Therefore, developing security and privacy protection approaches is an imperative requirement [10, 11, 12]. While solutions exist to secure data from IoT devices, the majority of these solutions require support for Transport Layer Security (TLS) standards. Class-0 IoT devices fall short of the resource requirements to support most of the security approaches offered [13, 14]. Therefore, a particular security mechanism, which is designed for Class-0 IoT devices, is highly demanded.

This paper provides a distributed security mechanism that is appropriate for the Class-0 IoT devices. The philosophy behind the provided solution is that light resource-consuming object encryption is implemented on the IoT

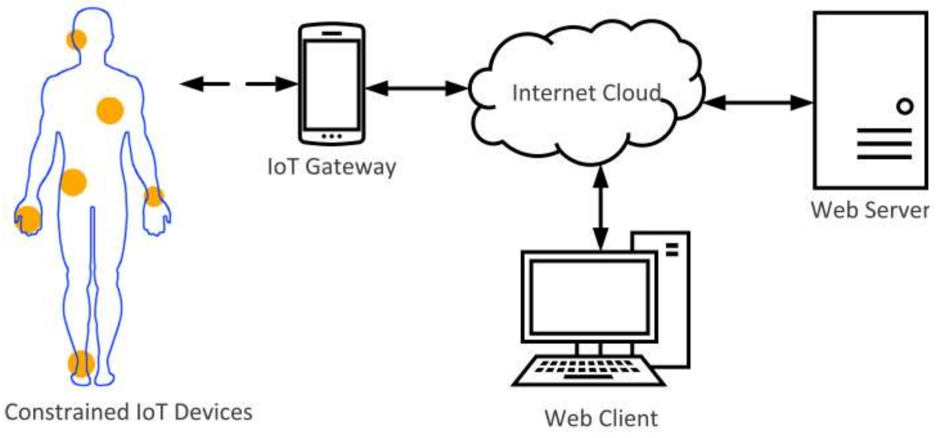


Figure 1: A generic IoT system using the example of resource-constrained IoT medical devices. The IoT system includes a network of devices, a gateway, and a web server. The IoT devices record and communicate data over the Internet.

device side, where object and protocol processing, which consumes resources heavily, is delegated to the gateway. The IoT gateway acts as an intermediary between the IoT device and the Internet [6]. The IoT gateway, shown in Figure 1, can take the form of a microcomputer, router, smart phone, or any device with ample resources to conduct TLS-based secure communication. In this research, a device-to-gateway layered security architecture has been designed and developed by implementing an Advanced Encryption Standard (AES) for the data object within the IoT device [15, 16, 17]. The extra device-to-gateway security layer has been created by employing the standard wireless security mechanism for IoT device authentication.

1.1 Paper contribution

The major contribution of this research is that it offers a complete security mechanism for the resource-constrained IoT devices. The security mechanism spans the IoT device, the IoT gateway, and the remote Internet server. The contribution comprises the design and implementation of a symmetric encryption of data objects at the IoT device over the native wireless security. We are thereby able to create a two-layer device-gateway security architecture. The implementation of a TLS-based security at the gateway works on standardly secure data objects before it travels over the Internet [6, 18]. The server has been configured to accept, process, and extract the data from the IoT gateway in its new format.

1.2 Paper structure

The rest of this paper is structured as follows: Section 2 provides background information on the IoT system, the problem description, and the related work. In section 3, the proposed security mechanism is theoretically explained in terms of the design requirements and interactions between IoT components. Section 4 is dedicated to demonstrating the implementation phase of the solution and the experimental setups. The performance evaluation of the proposed mechanism is documented in Section 5. Conclusions and future works are discussed in Section 6.

2 Preliminaries

IoT technology has developed in recent years to include more and more devices adopting embedded systems and communication interfaces. The future growth of IoT deployments comprises healthcare, education, manufacturing, and transportation. The main concept behind IoT devices is the possibility of collecting and sending information over the Internet [4, 11]. The architecture of the IoT components can be divided into three layers: the perception layer (physical devices), network layer (transmission layer), and application layer [7, 19]. However, each layer has its own security needs. This paper focuses on the perception layer for securing the entire data path. Figure 2 represents the layered architecture of IoT components and the data networks.

Constrained IoT devices can be grouped based on the available resources into three categories: Class-0 (C0), Class-1 (C1), and Class-2 (C2) devices. A comparison of the available resources in every category is shown in Table 1 [6]. It is apparent that the Class-0 devices have much fewer resources in terms of RAM and ROM memories. Furthermore, the available RAM size is not able to handle intensive security mechanisms.

IoT security needs to cover the entire IoT hierarchal architecture. IoT security spans the application layer, network layer, and perception layer. The basic security concerns include data confidentiality, integrity, and availability [7, 19, 20]. Constrained IoT devices have limited resources and therefore are limited to the protocols and standards they can support [21]. Efforts have been made by groups such as the Internet Engineering Task Force (IETF) to develop protocols and standards more suited for constrained environments, such as Datagram Transport Layer Security (DTLS)

Table 1: A comparison of the available resources in the categories of the constrained IoT devices [6].

	RAM (Data size)	ROM (Code size)
Class-0 (C0)	$\ll$ 10KB	$\ll$ 100KB
Class-1 (C1)	$\sim$ 10KB	$\sim$ 100KB
Class-2 (C2)	$\sim$ 50KB	$\sim$ 250KB

and Constrained Application Protocol (CoAP), by increasing the efficiency and minimizing the required computing resources [22, 23].

The security of the transport layer for Class-1 and Class-2 IoT devices can be achieved using DTLS over HTTP or CoAP. DTLS is an adaptation of the TLS protocol and has a heavy resource footprint in addition to existing application code in the device itself [14]. Like HTTP, CoAP as a stand-alone protocol does not contain security features necessary for secure data communication [24]. In order to fix this issue, a variation of TLS was developed to run under CoAP and over UDP called (DTLS) [22]. DTLS contains many features of TLS such as data encryption and authentication, with added features to deal with the unreliability of UDP [23]. Recently, CoAP over DTLS has been termed as CoAPS.

Despite the efforts of the IETF group, there is still a range of devices that fall short of the minimal resources needed to support such technologies on top of existing applications. These devices are known as “Class-0" as they fall short of the minimum threshold (10 KB of RAM and 100 KB of ROM) to support secure communication using TLS-based solutions [6].

The minimal code size and memory consumption for using DTLS were presented by Kumar et al. [14] in the DTLS implementation guide. The memory requirements outlined in the report suggest that the minimum resource requirements for DTLS (3.9 KB of RAM and 15.15 KB of ROM) would not be feasible in most Class-0 devices. It may also perform poorly on some Class-1 devices with connection times as slow as 24 seconds for a secure transmission [25]. As a conclusion, an alternative security solution is required for highly constrained IoT devices, especially for Class-0 IoT devices.

Doukas et al. [18] have attempted to secure data communication from constrained IoT medical devices by deploying IoT security in the gateway as an intermediary between the device and the Internet. This developed security solution secures data communication over the Internet by applying Public Key Encryption (PKI) and Secure Socket Layer (SSL) at the gateway. Although the solution presented in [18] focuses on the communication between the gateway and the Internet, the IoT system is still susceptible to attacks and data interception between the device and the gateway.

A solution offered by Vučinić et al. [26] was designed for more resource-heavy devices (C1 and C2). This offered solution uses DTLS-based security, and it applies a data object encryption inside a data transmission payload. The security of data objects is provided with symmetric encryption by way of an extra layer of protection for data communication. Although object layer security on its own does not offer effective security, it may be possible to add it to other security mechanisms for stronger security solutions.

Existing research addresses different challenges of secure data communication in Class-0 devices, but no single solution can be considered as a comprehensive solution that aims to secure the data path through all the IoT system components represented in Figure 2 Moreover, most of the available solutions do not target Class-0 devices. A security mechanism similar to that developed by Doukas et al. [18] provides a good base for securing IoT devices. However, it does not cover the entire data path, and it leaves a security gap between the device and the gateway. A comparison of some available solutions is presented in Table 2.

Driven by the demand for a comprehensive security solution to Class-0 IoT devices, this paper presents a complete and distributed security mechanism for these devices. Data encryption is one of the security requirements in the perception layer [21]. The novelty of the proposed solution is three-fold: the presented security mechanism focuses on the perception layer and aims to secure the entire data path from Class-0 IoT devices to the Internet; the distribution of the proposed solution over the IoT device, the gateway, and the remote server; and the provided multi-layer security between the IoT device and the gateway. By adding an extra layer of encryption at the object layer, message content can be protected inside the local network and at the gateway until it can be securely transferred over the Internet to its final destination.

3 A distributed security mechanism

This section focuses on the conceptual design of a distributed security mechanism. The design covers three IoT systems components: the IoT device, IoT gateway, and remote web server. Each component is discussed in detail along with a description of how the data are communicated. The design of the proposed security mechanism aims to achieve the requirements for Class-0 devices that are documented in Table 3.

The proposed solution secures data communication in Class-0-constrained devices by applying a 128-bit symmetric encryption (AES-128) to data objects, such as sensor readings, before they are transmitted between the device

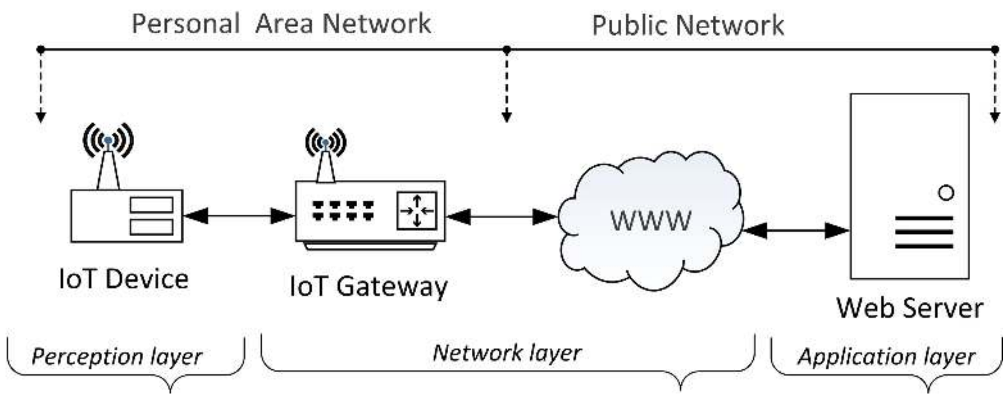


Figure 2: The three main layers of the layered architecture of IoT components. From the networking viewpoint, IoT devices and IoT gateway fall into a Personal Area Network (PAN), whereas the web server falls into a Public Network (PN).

Table 2: A comparison of some available security solutions for IoT devices.

	The concept	The drawback for Class-0 devices
Doukas et al. [18]	Enabling data protection through PKI encryption	Does not secure the device to the gateway
Rescorla et al. [22]	Datagram Transport Layer Security V1.2	Very heavy resource requirements
Vučinić et al. [26]	Object Security Architecture for the IoT	Very heavy resource requirements

Table 3: The design requirements for the proposed distributed security mechanism.

	Requirement
#1	Provide data security between the Class-0 device and the IoT gateway
#2	Secure data transported between the IoT gateway and the Internet
#3	Perform efficiently with minimal resource consumption

and the gateway. The data are formatted in JavaScript Object Notation (JSON) and are sent as a CoAP or HTTP POST to the gateway. The data object is encrypted using a secret key and can only be decrypted by devices with the same key. This key is shared with the destination, in this case, the web server.

Wireless transmissions in the LAN/PAN between the device and the gateway are secured at the Data Link Layer using a wireless interface module. Constrained wireless standards such as IEEE 802.15.4 and protocols such as Low power Wireless Personal Area Network (6LoWPAN) [27] are capable of supporting AES 128-bit symmetric encryption at this layer. By using an offered Pre-Shared Key (PSK) to encrypt wireless transmissions, only authorized devices connected to the network can receive traffic. By encrypting data objects at the device level (perception layer), only the device and the final destination will be able to read the encrypted data. An overview of the proposed security mechanism is represented in Figure 3. Further descriptions of the proposed distributed security mechanism on each IoT system component are provided in the following paragraphs.

3.1 Device-to-Gateway security

From the communication standpoint, another level of security between the IoT device and the gateway can be achieved using hardware-based symmetric encryption of the Data Link Layer (DLL) as part of the wireless protocol (e.g., IEEE 802.15.4, IEEE 802.11n). Wireless transmission can be provided using an IEEE 802.15.4 module such as a ZigBee or 6LoWPAN interface. When connecting to a network, devices are secured with a PSK, which is installed on each authorized device, and it is required for communication initiation between the gateway and the constrained devices in the network. Any unauthorized devices monitoring the traffic will not be able to decrypt data without the correct PSK. However, the built-in wireless security protects data from entities without the PSK, leaves data exposed if someone manages to compromise the wireless security, or capture the PSK from another device or from the gateway.

Confidentiality is assured between the IoT device and the destination by encrypting data at the object level. Object layer security exists at the application layer inside the payload of a transmission packet. Objects in this context

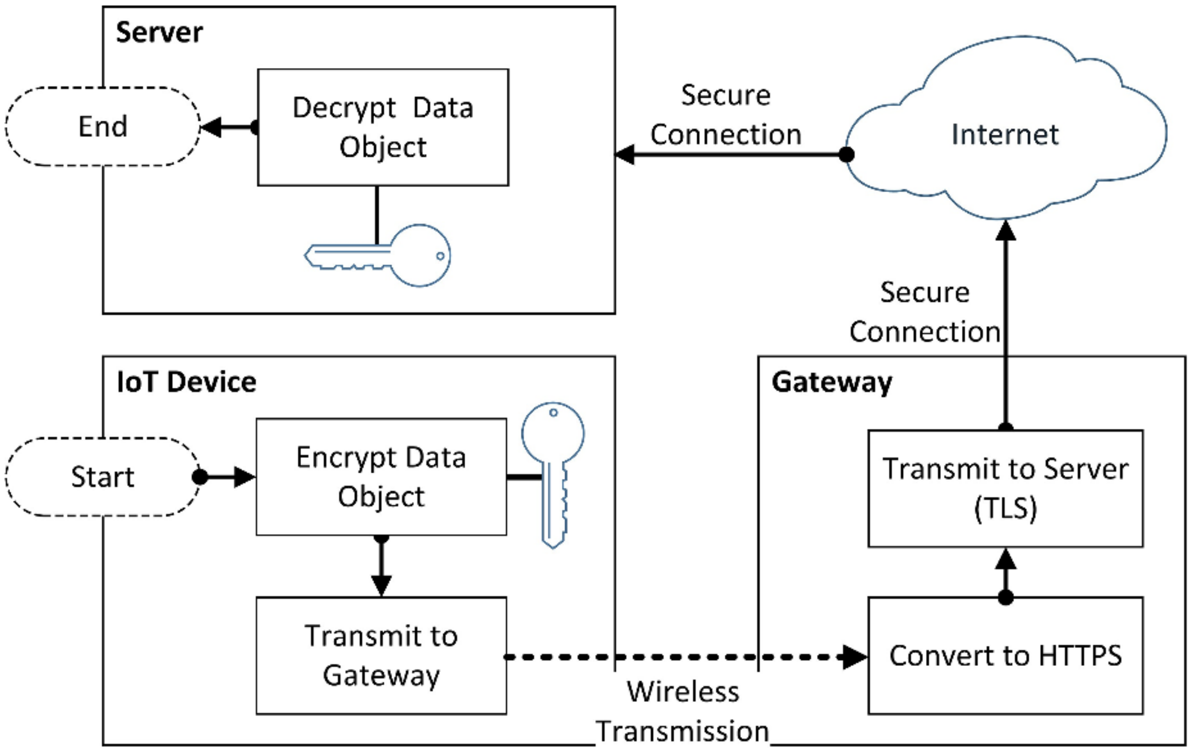


Figure 3: An overview of the proposed security mechanism shows the major processes that run on each component. The figure also represents the data connections and transmissions between the three IoT system components.

refer to a container of information, which has been formatted to be human readable. Different data formats exist for the web, including JSON, XML, and YAML. It is worth noting that object-layer security applies cryptography to a data object, but the header information such as the source and destination addresses remain exposed. The packet format and data encryption are shown in Figure 4. This level of encryption is used as a primary layer of protection, and it can be combined with the offered wireless security for stronger security between the IoT device and the IoT gateway. It works as a second defensive wall in case of a compromised wireless network.

Figure 4 depicts the two layers of security applied to data transmitted from the device to the gateway. Security is applied at the Data Link layer in the form of hardware-based AES encryption secured with a PSK. The second layer of security is applied only to the contents of the data object. Addressing and source information remain unencrypted in this layer. The data object is encrypted with a symmetric key, which has only been shared with the server so that no intermediaries will be able to decrypt the data.

3.2 Gateway-to-Internet security

IoT gateways are computational devices with enough resources to run operating systems and protocols necessary to securely transfer traffic across the Internet. An IoT gateway may take the form of a microcomputer with a Linux-based operating system. The gateway has sufficient resources to apply heavy security and communication protocols that cannot be supported by Class-0 devices. Once data are received by the gateway, they are processed into HTTPS and prepared for transmission to the remote server. The gateway is configured with Secure Socket Layer (SSL) tools, which are used to create a secure HTTPS connection between the gateway and the server. From the gateway point, one can forward secure communications to the server over the Internet using the configured secure socket layer.

The gateway acts as an intermediary with ample resources to support these security measures and secure data before sending it over the Internet. Data sent from the IoT device will be sent to the gateway using protocols such as CoAP and HTTP and sent across the Internet using HTTPS (HTTP over TLS) to the web server. In the proposed security mechanism, the payload of the packets is formatted as a JSON object and encrypted using AES 128-bit or 256-bit symmetric encryption. This data object will exist inside the transmission payload, while the packet header information such as source and destination address remains unencrypted, as demonstrated in Figure 4.

The JSON object is not readable by the gateway or any other intermediary entity other than the intended destination. Similarly, if the server sends a command back to the device, the data object is encrypted using the pre-shared symmetric key and is forwarded to the device for decryption. Security is applied at the Data Link Layer in the form of hardware-based AES encryption secured with a PSK. Only authorized devices should be in possession of

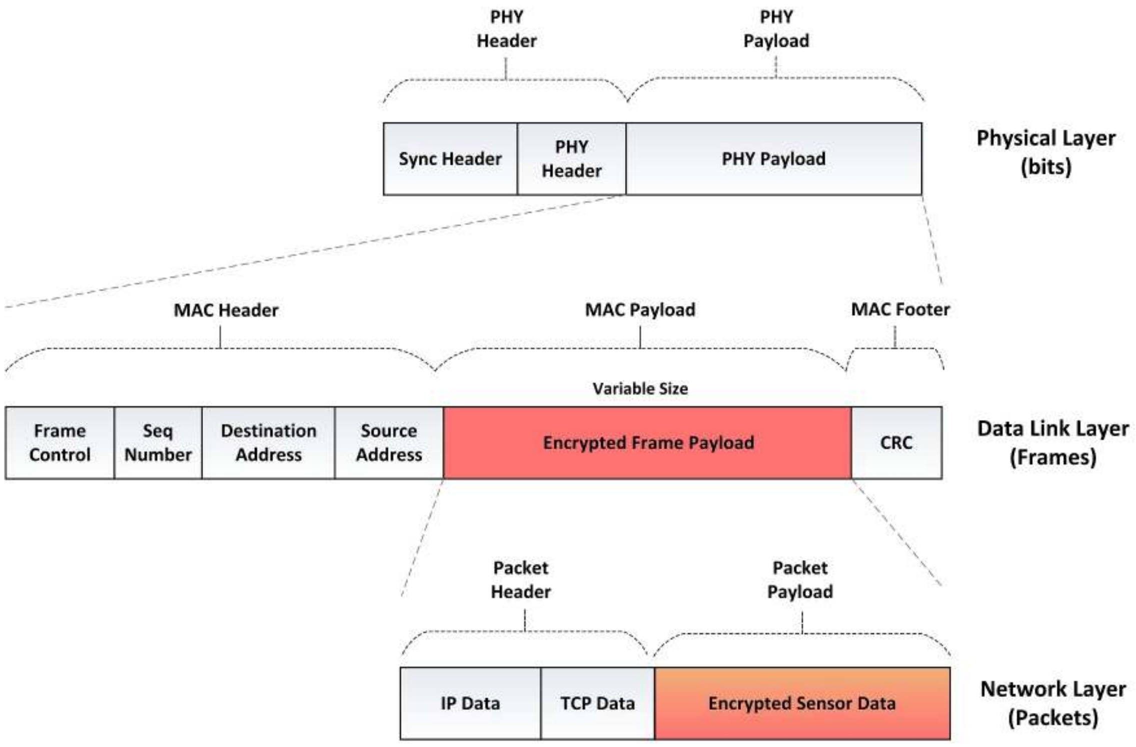


Figure 4: The utilized packet format ,which represents the packet header, the packet payload, and the encrypted part of the packet. The packet formats in the physical layer, the data link layer, and the network layer are represented.

the PSK. The second layer of security is applied only to the contents of the data object. Addressing and source information remain unencrypted in this layer. The data object is encrypted with a symmetric key, which has only been shared with the server, so that no intermediaries will be able to decrypt the data.

3.3 Web server security

The messages being transmitted to the server are encrypted with the server's public key, which is installed in the gateway. Only the server can decrypt messages using its corresponding private key. The private key is located on the server and is not shared with any other devices. The detailed flowchart of the proposed security mechanism with all sequential processes that are mapped to the three IoT system components is shown in Figure 5.

Once the HTTPS packets are received by the server, they are decrypted using the private key. The encrypted data object can then be decrypted using the symmetric secret key from the originating device, in this case, our class-0 IoT device. If the key is only present on one IoT device and the server, it can be used to authenticate data received from either party. If the key is shared with multiple devices, the devices are authenticated as part of a group. This scenario maintains the confidentiality of IoT data whenever it passes over a public network.

3.4 Advanced encryption standard

Advanced Encryption Standard (AES) is one such symmetric standard, which operates at fast speeds and requires fewer resources than DTLS, making it very suitable for Class-0 devices [14]. AES can be easily implemented and optimized on hardware. AES inputs data as 16-byte (128-bit) blocks that are then encrypted using a cryptographic key that is either 128 bits, 192 bits, or 256 bits in size [28]. The larger the key size, the greater the security and resource requirement for the device to encrypt and decrypt. Symmetric encryption can be applied at different layers of the communication stack such as the data link layer (e.g., wireless transmissions) and to specific objects of data within a message such as sensor readings. AES is suitable for the needs of Class-0 IoT devices in terms of the encryption speed and the required resources.

Symmetric cryptography involves encrypting data with a single encryption key, which is shared between multiple devices. Any device that possesses the key can decrypt data that have been encrypted with the same key. When the key is shared with other devices, there is a higher risk that it may fall into the wrong hands, and therefore, it must be kept safe.

Currently, in the proposed solution, the IoT data are encrypted in the IoT device using a symmetric key. The symmetric key is static and is installed only on the IoT device and the server. Thus, the gateway is not able to decrypt the packet payload. Messages being transmitted from the

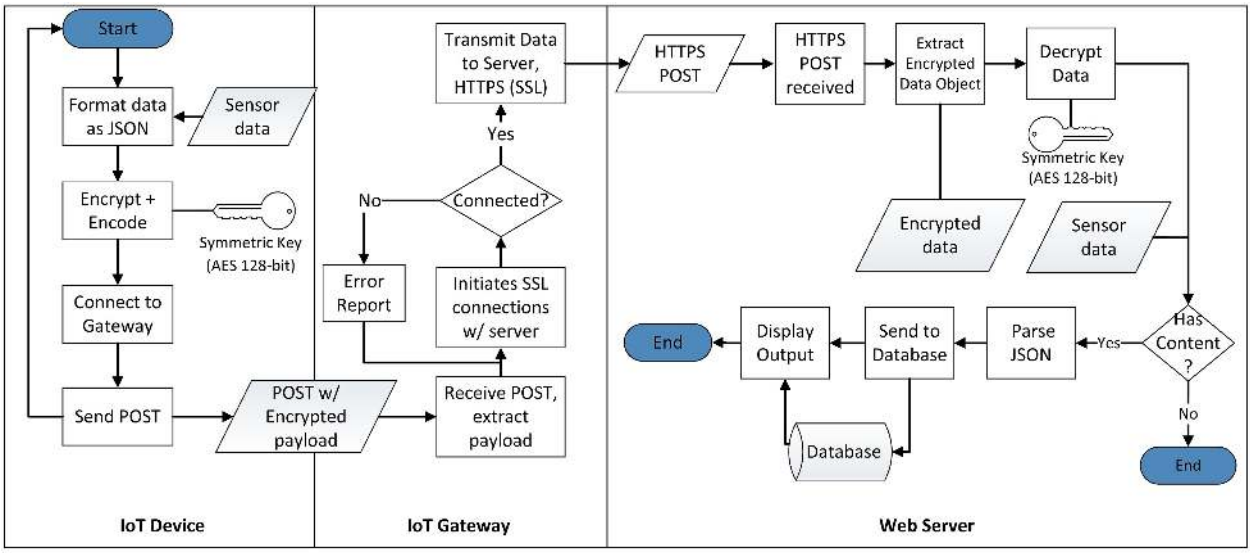


Figure 5: A full flowchart of the proposed security solution across the three IoT system components.

gateway to the server are encrypted with the server public key, which is installed in the gateway. Only the server can decrypt messages using its corresponding private key. The private key is located on the server and is not shared with any other device. An asymmetric key cryptography approach is used between the gateway and the server due to the plethora of computing capabilities.

4 Implementation flow

The distributed security mechanism has been implemented using real-time hardware configurations. This section describes the implementation, hardware specifications and configurations of the three IoT system components.

4.1 IoT device setup

For the hardware underlying the IoT device, an Arduino Uno microcontroller was used with an additional Ethernet shield added for connectivity. A wireless shield has been used as an alternative, but for the proof of concept, the Arduino wa connected directly to a wireless router via an Ethernet cable. A “DHT11” temperature and humidity sensor was connected to the Arduino. The Arduino hardware set up is shown in Figure 6 (a). The Arduino connects to and reads data from the sensor and then parses the data into the JSON format before encryption. The device automatically begins the sensor reading process when the device is connected to a power source, and it continues to repeat the process until the power is disconnected.

The temperature data are parsed as JSON and padded to 16 bytes, as this is the required block size for AES. The data are then encrypted using an AES-128 encryption library. The encrypted output may contain special characters, which are not web-friendly or human readable; therefore, it is encoded using the Base64 [29] character set so that it is easier to transmit to the remote server.

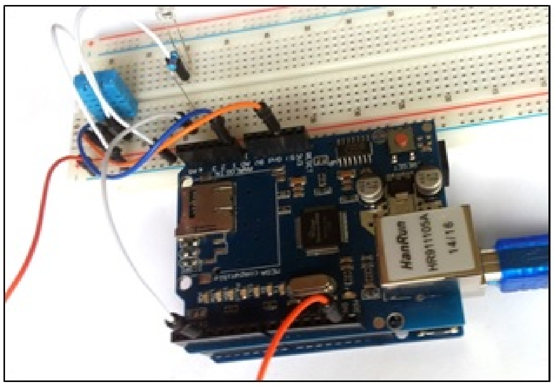
A web client has been prepared and installed on the Arduino in order to establish a connection to the IoT gateway. Once a connection is established, the Arduino uses a POST method to send data to the gateway via HTTP. The encrypted data are added to the contents of the HTTP POST before being sent. As soon as the POST message is sent, the Arduino receives a response back from the gateway confirming that the POST was received, waits for a period of time, and then restarts the processes from the beginning. Naturally, the confirmation back from the gateway to Arduino improves the reliability of the connection between the two terminals.

Through the formulation of the POST in the Arduino code, the header information is coded with the destination IP address and web service address “index.php". The encrypted sensor data are added to the contents of the post through the variable “dataEncoded”. If an error is received while attempting to connect to the gateway, the response is read when the POST reaches the gateway. If no errors are received, the connection is established and the packet is sent.

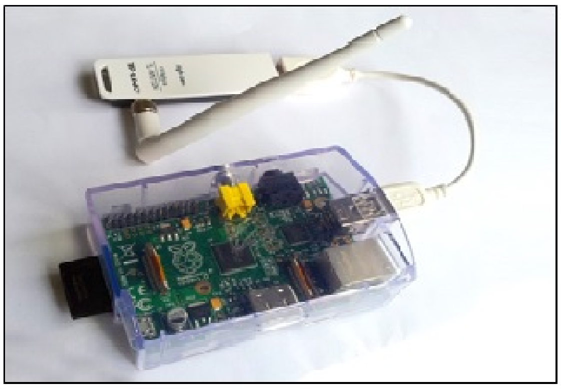
Figure 7 represents a captured TCP packet after it has been transmitted from the sensor (IoT device) and reassembled by the gateway. The packet includes the header information such as the destination and source address. It also includes the encrypted sensor data in the POST contents (i.e., “ZGioFzoApFk9CfV9XFQhxQ==").

4.2 IoT gateway setup

The IoT gateway was built on a Raspberry Pi (RPi) model B. The gateway hardware is shown in Figure 6 (b). The RPi is a microcomputer with ample resources to perform the heavier security processes that are too resource-intensive for the IoT device. The RPi contains a 700-MHz CPU, 512



(a)



(b)

Figure 6: The hardware set up for the implementation of the proposed security mechanism. (a)– The setup of an IoT device using Arduino hardware and (b)– The setup of an IoT gateway with a wireless antenna (Raspberry Pi setup).

MB, and a SD card reader that acts as its storage memory. In this case, an 8GB SD card was used for storage. The RPi can be configured with a range of Linux-based operating systems. The RPi connects to the wireless router through a wireless USB adapter. The RPi was installed with a PSK to access the wireless network that is secured with AES 256-bit symmetric encryption.

A web application running on an Apache web server was installed on the RPi to receive and process data from the IoT device. When a POST is received from the Arduino, the encrypted payload (sensor data) is stripped. The gateway does not contain the symmetric key to decrypt data from the Arduino; however, it forwards it to the server over a secure connection.

The RPi (IoT gateway) connects to the server using a SSL connection and posts the data to the server in an HTTPS POST. For testing purposes, the security certificate was not signed by a certificate authority, and therefore, when the IoT gateway attempted to connect to the server, the verification of the certificate with a trusted third party was disabled in the code (VERIFYPEER and VERIFYHOST). In a real environment, this would be unsafe, and by disabling the verification, the gateway would not be able to ensure that the connection has not been tampered with.

4.3 Web server setup

For testing purposes, the server was set up on a laptop within the local area network. This server represents the online server to which data would be transmitted. An Apache web server was installed and configured on the laptop. A security certificate was created, and the server was set up to receive HTTPS connections using SSL/TLS. As soon as the connection is established by the gateway, a HTTPS POST will be sent to the remote server carrying the encrypted data.

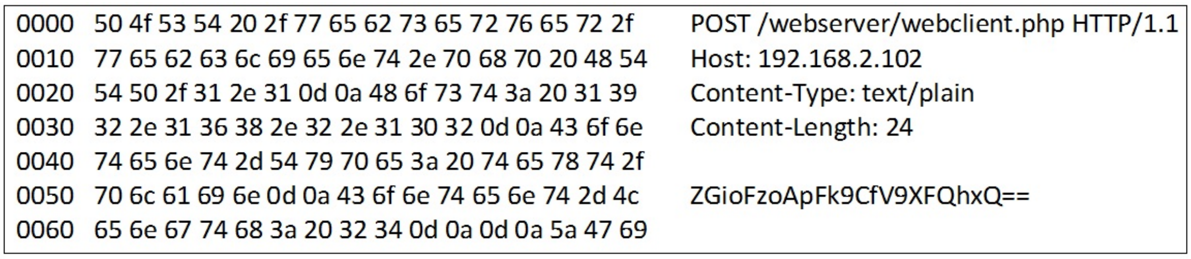
On the server side, a web service that handles the decryption process was installed. The cipher text and the symmetric key are passed to the service. The cipher text is decoded from base64 [29] into its original encrypted form. It is then processed using a "rijndael-128" cipher, which is another reference for AES-128. The final stage of the process is to parse the decrypted output and upload it to a database along with the date and the original encrypted message for reference. A web page was also created to demonstrate the working solution. The web page allows the user to view the latest sensor results, which are stored in the database.

5 Performance Evaluation

The proposed security mechanism has been evaluated based on its performance and ability to meet the outlined requirements in Section 3 In addition, the proposed security solution should perform in a timely manner and not be subject to an unacceptable amount of packet loss or failure. The solution is designed to support Class-0 devices with respect to resource consumption and processing time.

With AES, the data are passed to the algorithm in 16-byte blocks. If the input is larger than 16 bytes, it is divided into subsequent blocks. If a subsequent block falls short of the 16 bytes, padding is applied to increase the size of the data to 16 bytes. During the test, a small single-line JSON string was created with a temperature reading from the sensor. The result was 12 bytes in size, and 3 bytes of padding were added to the string before it was encrypted and then encoded using Base64 encoding scheme.

Using cryptography requires additional resources from the device. The performance measurements are recorded in Table 4; they satisfy the design requirements in Section 3. The requirements may change depending on the application of the device and the nature of its constraints. It is assumed that a resource overhead of 0.5 KB of RAM and 0.47 KB of ROM with an additional processing time of 0.46 seconds is an acceptable burden on most Class-0 systems. When the key size was increased to 256 bits, there was a 25% increase in processing time to 0.57 seconds and a 1% increase in ROM usage to 0.63 KB. For most applications,



```
0000  50 4f 53 54 20 2f 77 65 62 73 65 72 76 65 72 2f   POST /webserver/webclient.php HTTP/1.1
0010  77 65 62 63 6c 69 65 6e 74 2e 70 68 70 20 48 54   Host: 192.168.2.102
0020  54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 31 39   Content-Type: text/plain
0030  32 2e 31 36 38 2e 32 2e 31 30 32 0d 0a 43 6f 6e   Content-Length: 24
0040  74 65 6e 74 2d 54 79 70 65 3a 20 74 65 78 74 2f
0050  70 6c 61 69 6e 0d 0a 43 6f 6e 74 65 6e 74 2d 4c   ZGioFzoApFk9CfV9XFQhxQ==
0060  65 6e 67 74 68 3a 20 32 34 0d 0a 0d 0a 5a 47 69
```

Figure 7: A captured packet sent from the IoT sensor. The captured packet clarifies the encrypted and not-encrypted data from the sensor. The capturing probe is installed at a point between the device and the gateway.

the increases would be acceptable and provide stronger encryption as a result.

The implemented layered security provides strong circumvention against any external attack to the IoT system. An attacker would first need to gain access to the network either through direct access to the gateway or with a PSK for the network to be able to capture the data. With the additional encryption applied to data objects, even if the attacker had access to the network or the gateway, the attacker would not be able to read the data without the cipher key.

The memories overhead with respect to RAM and ROM in Table 4 are very low compared to the solutions offered in the literature. According to the implementations of two security solutions in [24], in particular, the encryption in the Host Identity Protocol (HIP) consumes 1.7 KB of ROM, and the encryption in the DTLS imposes an overhead of 3.3 KB of ROM and 1.5 KB of RAM. This confirms the applicability of our proposed solution for the Class-0 IoT devices.

At the gateway, the data are processed into HTTPS using RSA 2048-bit and a session key and securely forwarded to the web server. Using the network protocol analyzer tool, we can analyze the traffic exchange between the server and gateway. The received data are encrypted using a session key, and hence, they are not readable by any unauthorized user eavesdropping on traffic via active or passive traffic collection mechanisms [30, 31].

The processing times in Table 4 were recorded on the device (encryption time), on the gateway (object processing time), and on the server (decryption time). Due to the constrained processing power, the encryption time varies from AES-128 to AES-256. However, the processing time is constant on the gateway because the gateway is blind to the message contents. The gateway translates a message from HTTP to HTTPS and forwards it to the server.

The reported processing times are faster than what is reported in the literature. For example, Doukas et al. [18] achieved an 0.8-second overhead on the gateway compared to 0.18 seconds for our security solution. While the processing times in [18] are slightly slower than ours, it is worth noticing that they used a larger data size of "Less than 100 KB", whereas the message size used in this research is limited to 24 bytes.

6 Conclusions

Internet of Things (IoT) is a promising paradigm in wireless communications that offers a capability to connect, collect, and send data over the Internet. IoT keeps expanding with broad deployment demands in many fields such as home appliance, marketing, and healthcare. Despite the research attentions that IoT has received, the security, and hence, privacy issue in Class-0 devices is still a gap. This research has presented a distributed security mechanism for constrained Class-0 IoT devices. The design principle behind the proposed solution is to delegate the low resource consuming operations to the IoT device, and keep the high resource consuming processes at the IoT gateway side. In addition to the native wireless security, a layered security scheme has been offered by performing a asymmetric encryption to the data objects at the device level. The implementation of the distributed security mechanism has included the IoT device, the IoT gateway, and the server side. A complete laboratory setup for IoT infrastructure has been developed for the implementation and the evaluation purposes. Our experimental works have proven the security level of the solution, the suitability of the security mechanism to the Class-0 devices. In the worst case, with AES-256, the encryption process consumes memory overhead of 0.5 KB of RAM, 0.63 KB of ROM, 0.57 second encryption time on the device, and 0.18 second on the gateway. The future work focuses on the distribution and management of the encryption key, bring into attention additional security aspects such as data integrity and availability for improving the overall system's performance and circumvention.

References

[1] Kramp, T., van Kranenburg, R., Lange, S.: Introduction to the internet of things. In: Bassi, A., Bauer, M., Fiedler, M., Kramp, T., van Kranenburg, R., Lange, S., Meissner, S. (eds.) Enabling Things to Talk, pp. 1–10. Springer Berlin Heidelberg (2013)

[2] Medaglia, C.M., Serbanati, A.: An overview of privacy and security issues in the internet of things. In: Giusto, D., Iera, A., Morabito, G., Atzori, L. (eds.) The Internet of Things, pp. 389–395. Springer New York (2010)

Table 4: The memory overhead (RAM and ROM) and the processing times for the proposed distributed security mechanism.

	Memory consumption (KB)		Processing time (Second)		
	RAM (Device)	ROM (Device)	IoT (Device)	IoT (Gateway)	Server
No encryption *	16	0.5	–	–	–
AES-128 **	0.5	0.47	0.46	0.18	0.000205
AES-256 **	0.5	0.63	0.57	0.18	0.000404

*Base memory is considered. ** Overhead memory is considered.

[3] Lee, G.M., Crespi, N., Choi, J., Boussard, M.: Internet of things. In: Bertin, E., Crespi, N., Magedanz, T. (eds.) Evolution of Telecommunication Services, Lecture Notes in Computer Science, Vol. 7768, pp. 257–282. Springer Berlin Heidelberg (2013)

[4] Khan, R., Khan, S., Zaheer, R., Khan, S.: Future internet: The internet of things architecture, possible applications and key challenges. In: 10th International Conference on Frontiers of Information Technology (FIT). pp. 257–260. IEEE (2012)

[5] Höller, J., Tsiatsis, V., Mulligan, C., Karnouskos, S., Avesand, S., Boyle, D.: From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence. Elsevier, 1st edn. (2014)

[6] Bormann, C., Ersue, M., Keranen, A.: Terminology for constrained-node networks, RFC 7228, (2014), `http://www.rfc-editor.org/info/rfc7228`, last access 29.01.2016

[7] Zhao, K., Ge, L.: A survey on the internet of things security. In: 9th International Conference on Computational Intelligence and Security (CIS). pp. 663–667. IEEE (2013)

[8] Jing, Q., Vasilakos, A., Wan, J., Lu, J., Qiu, D.: Security of the internet of things: perspectives and challenges. Wireless Networks 20(8), 2481–2501 (2014)

[9] Lack of security in internet of things devices. Network Security 2014(8), 2 – (2014)

[10] Weber, R.H.: Internet of things – New security and privacy challenges. Computer Law & Security Review 26(1), 23–30 (2010)

[11] Santos, A., Macedo, J., Costa, A., Nicolau, M.J.: Internet of things and smart objects for M-health Monitoring and control. Procedia Technology 16(0), 1351–1360 (2014)

[12] Storey, A.: There's nothing 'smart' about insecure connected devices. Network Security 2014(7), 9–12 (2014)

[13] Raza, S., Trabalza, D., Voigt, T.: 6LoWPAN compressed DTLS for CoAP. In: The 8th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS). pp. 287–289. IEEE (2012)

[14] Kumar, S., Keoh, S., Tschofenig, H.: A hitchhiker's guide to the (datagram) transport layer security protocol for smart objects and constrained node networks (2013), `https://tools.ietf.org/html/draft-ietf-lwig-tls-minimal-00`, last access 29.01.2016

[15] Stallings, W.: Cryptography and Network Security: Principles and Practice. Pearson Education, NJ, USA (2002)

[16] Paar, C., Pelzl, J.: The advanced encryption standard (AES). In: Understanding Cryptography, pp. 87–121. Springer Berlin Heidelberg (2010)

[17] Fathy, A., Tarrad, I., Hamed, H., Awad, A.I.: Advanced encryption standard algorithm: Issues and implementation aspects. In: Hassanien, A.E., Salem, A.B.h., Ramadan, R., Kim, T.h. (eds.) Advanced Machine Learning Technologies and Applications, Communications in Computer and Information Science, Vol. 322, pp. 516–523. Springer Berlin Heidelberg (2012)

[18] Doukas, C., Maglogiannis, I., Koufi, V., Malamateniou, F., Vassilacopoulos, G.: Enabling data protection through PKI encryption in IoT m-health devices. In: The 12th IEEE International Conference on Bioinformatics Bioengineering (BIBE). pp. 25–29. IEEE (2012)

[19] Sun, X., Wang, C.: The research of security technology in the Internet of Things. In: Jin, D., Lin, S. (eds.) Advances in Computer Science, Intelligent System and Environment, Advances in Intelligent and Soft Computing, Vol. 105, pp. 113–119. Springer Berlin Heidelberg (2011)

[20] Yang, X., Li, Z., Geng, Z., Zhang, H.: A multi-layer security model for internet of things. In: Wang, Y.,

Zhang, X. (eds.) Internet of Things, Communications in Computer and Information Science, Vol. 312, pp. 388–393. Springer Berlin Heidelberg (2012)

[21] Suo, H., Wan, J., Zou, C., Liu, J.: Security in the internet of things: A review. In: IEEE 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE). Vol. 3, pp. 648–651. IEEE (2012)

[22] Rescorla, E., Modadugu, N.: Datagram transport layer security version 1.2, RFC 6347, (2012), https://tools.ietf.org/html/rfc6347, last access 29.01.2016

[23] Shelby, Z., Hartke, K., Bormann, C., Frank, B.: The constrained application protocol (CoAP), RFC7252, (2014), https://tools.ietf.org/html/rfc7252, last access 29.01.2016

[24] Garcia-Morchon, O., Keoh, S.L., Kumar, S., Moreno-Sanchez, P., Vidal-Meca, F., Ziegeldorf, J.H.: Securing the IP-based internet of things with HIP and DTLS. In: Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks. pp. 119–124. WiSec '13, ACM (2013)

[25] Keoh, S., Kumar, S., Garcia-Morchon, O.: Securing the ip-based internet of things with DTLS (2013), https://tools.ietf.org/html/draft-keoh-lwig-dtls-iot-02, last access 29.01.2016

[26] Vučinić, M., Tourancheau, B., Rousseau, F., Duda, A., Damon, L., Guizzetti, R.: OSCAR: Object security architecture for the Internet of Things. Ad Hoc Networks 32(0), 3–16 (2015), Internet of Things security and privacy: Design methods and optimization

[27] Kolahi, S., Li, P., Argawe, M., Safdari, M.: WPA2 security-bandwith trade-off in 802.11n peer-peer WLAN for IPv4 and IPv6 using Windows XP and Windows 7 operating systems. In: The 7th IEEE Symposium on Computers and Communications (ISCC). pp. 575–579. IEEE (2012)

[28] Elfatah, A.F.A., Tarrad, I.F., Awad, A.I., Hamed, H.F.A.: Optimized hardware implementation of the advanced encryption standard algorithm. In: 8th International Conference on Computer Engineering Systems (ICCES). pp. 197–201. IEEE (2013)

[29] Coles, M., Landrum, R.: SQL CLR Cryptography. In: Expert SQL Server 2008 Encryption, pp. 167–184. Apress (2009)

[30] Rubio-Loyola, J., Sala, D., Ali, A.: Maximizing packet loss monitoring accuracy for reliable trace collections. In: 16th IEEE Workshop on Local and Metropolitan Area Networks, LANMAN 2008. pp. 61–66. IEEE (2008)

[31] Rubio-Loyola, J., Sala, D., Ali, A.: Accurate real-time monitoring of bottlenecks and performance of packet trace collection. In: 33rd IEEE Conference on Local Computer Networks, LCN 2008. pp. 884–891. IEEE (2008)