INTRA-OPTIMISED LIGHTWEIGHT ENCIPIERING ALGORITHM BASED ON MQTT PROTOCOL FOR INTERNET OF THINGS SECURE APPLICATION

Ronald Chiwariro¹, S. Rajendran²,
Information Security and Cyber Forensics,
Department of Information Technology,
SRM Institute of Science and Technology,
Kattankulathur 603 203, Chennai
Tamil Nadu, INDIA
chiwariro@gmail.com
rajendran.s@ktr.srmuniv.ac.in

July 9, 2018

Abstract

Recent advances in technology have led to rapid growth of Internet of Things (IoT) systems which incorporate numerous miniaturized low powered devices with large numbers of sensors and actuators collecting and exchanging data autonomously over the internet generating enormous amounts of data that needs to be secured. Traditional encryption algorithms are not suitable due to great complexity and numerous rounds for encryption and decryption operations. There is however a rising need for elaborate lightweight encryption algorithms with less complexity for optimum security in resource constrained communication networks. In this paper, a
lightweight encryption algorithm called Intra-Optimized Lightweight Enciphering (ILE) Algorithm is proposed. The proposed scheme is complemented by watchdogs who are deployed in the clusters to achieve optimum security for the overall generated clusters at less cost and is simulated on Message Queue Telemetry Transport (MQTT) protocol using Mosquito broker in Cooja simulator and the performance was evaluated. Results from simulations show that the proposed algorithm offers significant security, improved performance and power drain without compromising the quality of service and further a comparison was made with existing lightweight algorithms.

**Keywords**: ILE, MQTT, watchdogs, cluster, lightweight cryptography, constrained devices.

1 WIRELESS SENSOR FOSTERED INTERNET OF THINGS

Internet of things (IoT), also known to as ‘Smart Object’ Networks [1] refers to the internetworking of any physical devices, homes, vehicles or any other thing embedded with sensors, actuators, electronics and network connectivity. IoT devices are basically connected to the internet, can be sensed and therefore can be remote controlled. These intelligent devices have caused a paradigm shift to the way the environment interacts with technology. As rapidly evolving as it is, IoT has infiltrated many research areas, ranging from home automation, the healthcare sector, industrial automation, logistics, the mining sector, security and many others. The increase in general accessibility of broadband internet at lower costs has led to increased connectivity which is resulting in huge amounts of data being generated autonomously every second. Some of the data requires some varying levels of security as we see the application of IoT across all verticals.

The limitations surrounding the IoT environment include unstable network state, limited computation capability and low battery power [2] among others, leading to applications with limited security or non-secure applications which are prone many attacks. The other major problems with IoT objects is that they
run autonomously in the field without constant supervision hence they are susceptible to tempering as well as their wireless nature which makes eavesdropping possible. In order to mitigate these constrains within an IoT ecosystem at the application layer, various data transfer protocols have been proposed in the recent years, which includes Message Queuing Telemetry Transport (MQTT and Advanced Message Queueing Protocol (AMQP) that use the queuing theory in publish/subscribe pattern. Furthermore, the Extensible Messaging and Presence Protocol (XMPP) allows exchange of structured extensible messages and the Constrained Application Protocol (CoAP) uses a lightweight request/response model. These data transfer protocols suffer bottle necks inherited from the constrained environments in which they operate hence security implementations remain a challenge in trying to keep them lightweight; which is why they were proposed in the first place.

2 RELATED WORK

There are numerous researches for data security and cryptographic primitives’ optimization in IoT with various implementations as well. A number of security challenges are associated with systems which the protocols embodies default functionalities and settings. Additionally, many lightweight encryption algorithms have been proposed, with a large number of them having appealing features. However, there is still need for more research to further optimize these algorithms in order to make them suitable for implementation in low power and resource constrained IoT networks.

2.1 Lightweight Cryptography

The need for security in resource-constrained IoT networks gave rise to research in lightweight cryptographic primitives. The primitives include both symmetric and asymmetric block ciphers and stream ciphers, a variety of lightweight hash functions and message authentication codes (MACs), which aim to offer improved performance over conventional cryptographic standards. The main difference between conventional algorithms and these lightweight primitives is that lightweight primitives are intended
for limited applications, and they assume similar power limitations to the attacker. However, this fact should not be confused to mean that the lightweight algorithms are less secure but, contemporary research should use improvements that result in designs that balance between security, improved efficiency and resource usage [3].

2.2 Lightweight Block Ciphers

Advanced Encryption Standard (AES) [4], particularly AES-128 has been simplified in some implementations to improve their efficiency. One of the early lightweight block cipher designs [5] targeted constrained hardware environments. Some designs come in suites such as SIMON and SPECK [6], which are lightweight block ciphers aiming for simplicity, flexibility and performance to the underlying hardware and software. Some algorithms that have been fairly implemented in constrained environments due to their simple round structure are RC5 [7], TEA [8], XTEA [9] and HIGHT [10]. Other lightweight block ciphers are listed in [11]. Particularly the performance benefits in all these algorithms are realized through the design choices that include smaller block and key sizes, simpler rounds and key schedules among others.

2.3 Lightweight Stream Ciphers

Mainly Lightweight Stream ciphers were restricted to resource constrained hardware applications. One such stream cipher is Grain [12] due to its implementation flexibility and authentication support. Another design that was well analysed is Trivium [13] which makes use 80-bit keys. However, Mickey [14] provides limited implementation flexibility when analysed against Grain and Trivium as well as being susceptible to scheduling and power analysis caused by irregular clocking.

2.4 Lightweight Hash Functions

Similarly, the constraints in constrained environments apply to hash functions. This is mainly contributed by the internal state sizes and levels of power consumptions. Lightweight hash
functions, such as PHOTON [15], Quark [16], SPONGENT [17], and Lesamnta-LW [18] were as a result of trying to balance between power consumption and efficiency. The general usage of lightweight hash functions differs from that of conventional hash functions in a variety of ways [19].

2.5 Lightweight Message Authentication Codes

Message authentication code (MAC) uses the combination of a message and a secret key to generate a tag that is used for authentication and integrity checks on messages. The minimum recommended Tag sizes are to be at least 64 bits. Some examples of lightweight MAC algorithms are Chaskey [20], TuLP [21], and LightMAC [22].

3 INTRA-OPTIMIZED LIGHTWEIGHT ENCIIPHERING ALGORITHM (ILE)

The primary objective is to improve the security of IoT systems in particular those using the publish/subscribe pattern and to offer a security mechanism for the resource constrained devices whilst limiting the complexities for encryption and decryption without compromise on the underlying security of the algorithm. The proposed algorithm is intended to offer high security in IoT domain in particular to use cases that involve highly constrained devices where network connections are erratic and power management is of essence. The Intra-Optimized Lightweight Enciphering (ILE) algorithm is a symmetric key block cipher incorporating an 80-bit key and 80-bit plain-text following both the Substitution and Permutation Network (SPN) and feistel structure in order to achieve Shannons requirements for diffusion and confusion as well balance with computational complexity for the encryption and decryption operations by making maximum use of the same registers for both operations. However, in symmetric key algorithm the encryption process comprises of encryption rounds, each round depends on some computational
capacities to create disarray and an increase in number of rounds guarantees improved security. Cryptographic algorithms [23] are normally optimised at the range of 10 to 20 rounds on average to keep the encryption procedure sufficiently rigid to attain higher level of diffusion and confusion. The proposed algorithm is limited to just six rounds, hence to improve the energy efficiency, individual round of encryption incorporates mathematical operation that operate on data blocks of 4 bits. To complement the algorithm, the scheme should employ watchdogs for detection of intrusions. The algorithm goes through two fundamental stages; Key expansion and Encryption on the publisher side followed by Key expansion

3.1 Key Expansion

The key expansion process is the first part as the keys are used for subsequent encryption rounds. The secret key is used to generate round keys for subsequent encryption or decryption rounds. It is important to note that the whole security of the algorithm is dependent upon the key. In this regard, essential measures must be considered to make the disclosure of the key on a need to know basis. The proposed scheme is an 80-bit cipher, which requires an 80-bit key for every round of encryption and decryption. A cipher key of 80-bits is taken as input key by the cipher block. The Block, after operations creates diffusion as well as confusion and in the process creates six special keys from this initial key. These keys should be utilized as a part of the encrypting/ decrypting process and are sufficiently rigid to remain unclear amid attacks. The key expansion process includes the following:

- Initially, the 80-bit cipher key is divided into multiple blocks of 4-bits.
- The bits for each function are obtained by carrying out initial substitutions of the segments of cipher key as given in the equation:

\[ K_{bi} = \bigoplus_{j=1}^{6} Kc(j-1) + 1 \]  

(1)

Where \( i = 1 \) to 5 for first 5 round keys as depicted in Fig in 2.
• Matrix transformation are carried out to obtain round keys, K1, K2, K3, K4, K5, and K6 into six arrays of 16 bits round keys (Kr). The arrangement of these bits is shown in equations (2), (3), (5), (6) and (7).

\[ K_1 = i_1 \oplus i_2 \oplus i_3 \oplus i_4 \oplus i_5 \oplus i_6 \oplus i_7 \oplus i_8 \oplus i_9 \oplus i_{10} \oplus i_{11} \oplus i_{12} \oplus i_{13} \oplus i_{14} \oplus i_{15} \oplus i_{16} \]  
\[ K_2 = j_1 \oplus j_2 \oplus j_3 \oplus j_4 \oplus j_5 \oplus j_6 \oplus j_7 \oplus j_8 \oplus j_9 \oplus j_{10} \oplus j_{11} \oplus j_{12} \oplus j_{13} \oplus j_{14} \oplus j_{15} \oplus j_{16} \oplus j_{17} \oplus j_{18} \oplus j_{19} \]  
\[ K_3 = k_1 \oplus k_2 \oplus k_3 \oplus k_4 \oplus k_5 \oplus k_6 \oplus k_7 \oplus k_8 \oplus k_9 \oplus k_{10} \oplus k_{11} \oplus k_{12} \oplus k_{13} \oplus k_{14} \oplus k_{15} \oplus k_{16} \oplus k_{17} \oplus k_{18} \oplus k_{19} \oplus k_{20} \]  
\[ K_4 = l_1 \oplus l_2 \oplus l_3 \oplus l_4 \oplus l_5 \oplus l_6 \oplus l_7 \oplus l_8 \oplus l_9 \oplus l_{10} \oplus l_{11} \oplus l_{12} \oplus l_{13} \oplus l_{14} \oplus l_{15} \oplus l_{16} \oplus l_{17} \oplus l_{18} \oplus l_{19} \oplus l_{20} \]  
\[ K_5 = m_1 \oplus m_2 \oplus m_3 \oplus m_4 \oplus m_5 \oplus m_6 \oplus m_7 \oplus m_8 \oplus m_9 \oplus m_{10} \oplus m_{11} \oplus m_{12} \oplus m_{13} \oplus m_{14} \oplus m_{15} \oplus m_{16} \oplus m_{17} \oplus m_{18} \oplus m_{19} \oplus m_{20} \]  
\[ K_6 = K_1 \oplus K_2 \oplus K_3 \oplus K_4 \oplus K_5 \]  

An XOR operation is carried out among the six round keys to obtain the seventh round key as shown in equation

\[ K_6 = K_1 \oplus K_2 \oplus K_3 \oplus K_4 \oplus K_5 \]  

3.2 Encryption

As soon as round keys are generated we can begin the encryption process; we create disarray by some logical operations which are swapping, left shifting and substitution. The plain text is divided into blocks which will be encrypted by the round keys in turn. The blocks are Px0-15, Px16-31, Px32-47, Px48-63, PX64-79. Bitwise XNOR Process is carried out amid the individual round keys (K1 K5) which were generated earlier during the key expansion phase. This is done to the blocks to obtain Px0-15 to Px64-79 cipher text blocks respectively. The blocks are then finally XORed together with the last round key to obtain the final cipher text.

3.3 S-box Configuration

The Substitution box (S-Box) constitutes an important module of symmetric-key algorithms. To create a disarray of the plain text and coded text (cipher text) to make it difficult to decode. It is however necessary to choose carefully to avoid cryptanalysis. The substitution box s-box takes some number of m input bits and however transform them to some number of output bits, n. The number of output bits, n: an m x n S-Box can be implemented with 2m words of n bits.
Figure 1: Key Expansion
Figure 2: S Boxes Configuration [4]

Figure 3: S - Box Configuration
4 EXPERIMENTAL RESULTS AND ANALYSIS

Figures 5, 6 and 7 are results from validations of our algorithm in Cooja Simulator environment with standard testing methods for assessment against AES-128 and HIGHT algorithms. A trade off was made between required security level and the resources required to achieve it. Performance metrics such as power and energy consumption, latency, and throughput are critical in determining the applicability of any lightweight algorithm within the IoT constrained environment. Furthermore, the efficiency of lightweight algorithms should be reflected in register, RAM and ROM usage as reflected in Table 1. Throughput was measured by considering the rate at which cipher text was being produced. ILE showed high throughput compared to the other two algorithms. In IoT real-time applications, Latency is especially a relevant factor to consider, for example automotive applications where quick responses to triggered actions are required. It can be measured by considering time between the initial request of a process and production of the output. This was measured by taking into consideration the difference between initial request for encryption of payload and the reply that yielded the equivalent cipher text. Likewise, power consumption was also recorded over a time period. The algorithm uses less power which is a desirable quality in lightweight cryptography. However, power consumption depends on many other such as the threshold voltage, clock frequency besides the algorithm used.

5 CONCLUSION

In this paper Intra-Optimized Lightweight Encipher, a data encryption algorithm for use in resource constrained IoT environments to mitigate the resource challenges of adopting Transport Layer Security (TLS) on erratic networks was presented. MQTT was used for implementation and testing of AES-128, HIGHT and ILE using Mosquito Broker in Contiki Cooja Simulator. The overall performance of the proposed algorithm was evaluated and the results show that ILE algorithm
Table 1. Software Metrics

<table>
<thead>
<tr>
<th>Cipher</th>
<th>Device</th>
<th>Block Size</th>
<th>Key Size</th>
<th>Code Size</th>
<th>Ram</th>
<th>Cycles (Enc)</th>
<th>Cycles (Dec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>AVR</td>
<td>64</td>
<td>128</td>
<td>1570</td>
<td>-</td>
<td>2739</td>
<td>3579</td>
</tr>
<tr>
<td>HEIGHT</td>
<td>AVR</td>
<td>64</td>
<td>128</td>
<td>5672</td>
<td>-</td>
<td>2964</td>
<td>2964</td>
</tr>
<tr>
<td>ILE</td>
<td>Atmega328</td>
<td>80</td>
<td>80</td>
<td>1228</td>
<td>15</td>
<td>2480</td>
<td>2860</td>
</tr>
</tbody>
</table>

Figure 4: Throughput (cycles per byte)

Figure 5: Power/ Energy Consumption (W)
had better performance in terms of throughput, latency and power dissipation for both encryption and decryption operations. We further recommend use cases that require additional security to further complement the algorithm with a well-organised distributed clustering algorithm for high security integrity through the use of watchdog mechanisms for intrusion detection. With the data generated by IoT autonomously everyday there is great need to be keen on privacy and protection issues to personal and corporate information. Adoption of platform supported security mechanisms to data both in transit and storage remains critical to the rapid acceptance of IoT by ordinary users since concerns remain on security and privacy issues.

References


