# TD²SecIoT: Temporal, Data-Driven and Dynamic Network Layer Based Security Architecture for Industrial IoT

Dawit Dejene[1], Basant Tiwari[1], Vivek Tiwari[2]*

[1] Department of Computer Science, Institute of Technology, Hawassa University (Ethiopia)
[2] Computer Science & Engineering, Dr. S P Mukherjee IIIT-NR, Naya Raipur, Chhattisgarh (India)

Received 13 March 2020 | Accepted 24 August 2020 | Published 7 October 2020

LA UNIVERSIDAD
EN INTERNET

## Abstract

The Internet of Things (IoT) is an emerging technology, which comprises wireless smart sensors and actuators. Nowadays, IoT is implemented in different areas such as Smart Homes, Smart Cities, Smart Industries, Military, eHealth, and several real-world applications by connecting domain-specific sensors. Designing a security model for these applications is challenging for researchers since attacks (for example, zero-day) are increasing tremendously. Several security methods have been developed to ensure the CIA (Confidentiality, Integrity, and Availability) for Industrial IoT (IIoT). Though these methods have shown promising results, there are still some security issues that are open. Thus, the security and authentication of IoT based applications become quite significant. In this paper, we propose TD²SecIoT (Temporal, Data-Driven and Dynamic Network Layer Based Security Architecture for Industrial IoT), which incorporates Elliptic Curve Cryptography (ECC) and Nth-degree Truncated Polynomial Ring Units (NTRU) methods to ensure confidentiality and integrity. The proposed method has been evaluated against different attacks and performance measures (quantitative and qualitative) using the Cooja network simulator with Contiki-OS. The TD²SecIoT has shown a higher security level with reduced computational cost and time.

## Keywords

Cooja, Cryptography, Elliptic Curve Cryptography (ECC), IoT, NTRU, Security, CIA, Dynamic Network Layer.

DOI: 10.9781/ijimai.2020.10.002

## I. Introduction

The Internet of Things (IoT) is emerging as a novel thought for future technological advancement, which allows interaction between the physical and virtual world. IoT devices play the role of internet devices and provide mountainous improvement in Information and Communication Technology (ICT) applications [1]. It is predicted that total IoT devices will be 24 billion worldwide in the coming year, which will be connected through the internet [1]. IoT provides exclusive identity to objects that are accessible from the network for tracking down the status and location of the devices [2]. Later on, there will be numerous sorts of things conceivably interfacing with the internet such as heterogeneous industrial machines, self-driving cars, robots, smart supermarkets, mobile phones, shoes, plants, watches, etc. [3], [4].

IoT is applicable in various areas of fields such as Smart Homes, Smart Cities, Smart Industries, Military, eHealth, and several real-world applications by connecting domain-specific sensors [4] and more specifically known as Industrial IoT [2],[4]. IoT also provides tracking, monitoring, and controlling of devices and their services. This influences the minimum human interactions with the attached physical articles to work accordingly. IoT uses Wireless Sensor Network (WSN) infrastructure and technologies such as Radio Frequency ID Devices, IR sensors, laser scanner, Global Positioning System (GPS), etc. [4],[6]. The collected data through the IoT system is real-time data. IoT enables devices (things) to interact and coordinate with each other to perform domain-specific tasks that result in the less human intervention [7].

The Industrial IoT (IIoT) is composed of wireless sensors to control several processes remotely that are constantly connected with physical objects embedded in industrial devices. These devices are vulnerable for different security attacks like Distributed Denial of Services (DDoS), Man in the Middle attack (MiTM) targeted on power grids, hacking of industrial control systems, etc. [8]. These attacks are increasing and have a significant impact on the progress of industries. The ubiquitous nature of IIoT expands the scope of risks, especially when intersecting with hackers or illegitimate users. For example, in healthcare IIoT, riding insulin pumps or pacemakers is a striking concern. Similarly, in the food supply chain, it is a big concern that any agricultural drone surveying a farmland strike may come under the hacker control , etc.[6]. Thus, creation, movement, and processing of goods and services of industries need to be secured.

It is estimated that by 2023 the industrial IoT may have a 14.2 trillion growth to the global economy [9]. Hence, ensuring the CIA (Confidentiality, Integrity, and Availability)  of IoT devices is unquestionable. This security is related to information that is either

* Corresponding author.

E-mail address: viveknitbpl@gmail.com

stored or in transit. Therefore, it is vital to have a systematic study and research on designing and improving the security problems of IoT [10]. The standard IoT systems are composed of four main layers known as Information Generation (Perception), Information Transmission (Network), Information Management (Processing), and Integrated Application layer. Each layer is vulnerable for attacks associates with it [11]. An attacker can send a malicious code that can affect the integrity of the data. The Network layer is composed of routers, computers with wireless or wired networks with greater vulnerability for attacks. This includes hacker intrusion and network content security, Sybil, sinkhole, sleep deprivation, code injection, and man-in-the-middle attacks commonly pertain to this layer. The Application layer is a layer where smart environments are achieved. This layer ensures the CIA of data in real-time. Code injection, DoS, spear-phishing, and sniffing are attacks associated with the application layer [7].

Nowadays, actuators in IIoT service infrastructure are also added into the production floor that must be authenticated before the transmission of sensed data [12]. This is a safeguard for data that should be initiated from authentic devices rather than a fraudulent source. Thus, it is essential to ensure the IIoT devices and underlying data to provide data integrity and confidentiality. The solution must protect firmware (program enabled machines) against malicious activities to take less computational time as well as battery power. This work ensures a comprehensive package of security solutions, including confidentiality and integrity requirements, with rigid security components for IIoT devices. There is no universal agreed-upon common security mechanism available to implement in the device [13]. In this view, the manufacturer applies the techniques in devices which is available with them. This limitation leads to immature standards to protect IIoT devices. Moreover, this leads to little or no encryption for protecting sensed data either at the storage or in transit. This opens the way for security threats that include various attacks. There is some industrial process that is more vulnerable such that Human/Machine interface, industrial control system, Machine logic controller and industrial SCADA system, etc.

The securing IoT system is complex as well as challenging since IoT applications are distributed and heterogeneous. In addition, the centralized cloud services cannot cover attack detection, due to resource restrictions, distribution, scalability, latency, and mobility of devices, etc. [14]. Thus, securing IoT requires to be design strong encryption and authentication method and protocol with cryptographic techniques [7].

Proposed work deals with security augmented IoT architecture for ensuring data integrity and authentication in IoT industrial applications. This integrates ECDSA and NTRU cryptosystems using a co-design method, which overcomes the shortcomings of other traditional algorithms. The proposed model employs a dynamic key cryptosystem which uses temporality and dynamicity techniques to ensure confidentiality and integrity. The combination is needed to achieve the stringent performance in IIoT infrastructure.

This paper discussed the basic concepts of IoT and security problems. Section II reviews related works and literature on IoT security. Section III discusses and illustrates the proposed TD$^2$SecIoT architecture in detail. Section IV deals with simulation setup and configurations. Section V presents experimental results and comparative analysis of the proposed work with existing research work based on key generation, authentication, and encryption time and power (energy) usage. At the same time, Section VI deals with security analysis of the proposed work. Section VII concludes the work by discussing the major findings, future works, and recommendations.

## II. Related Work

Several applications of cryptosystems are proposed in recent years for ensuring security in resource-constrained IoT devices and applications [15]. Most of the current methods developed to secure IoT are based on symmetric and asymmetric cryptographic algorithms. Some of the researchers also employ these algorithms in a hybrid fashion [16].

S. B. Sadkhan, and Zainab Hamza [16] introduced various cryptosystems for IoT environment. They reported that RSA is the slowest to decrypt and takes the utmost time to encrypt as compared to other methods like ECC, and also demands more memory space. Blowfish takes less time among all for both encryption and decryption and demands the least memory. Blowfish is effective on software but depends on the platform on which it executes. AES and DES require optimal memory space as compared with RSA. Both AES and Blowfish have high unpredictability in the output information and are less vulnerable to attacks. D. Rivera *et al.* [17] proposed a security protocol that employs AES and SHA-3 for IoT smart toy platform. The author focused on confidentiality, integrity, and authenticity for the data generated by the smart toys. Their proposed platform involved different modules of the system and also have concluded that denial of service attacks on the smart IoT toys might be possible. V. C. Ashan [18] modified Welch Gong (WG-7) stream cipher method, which employs an involution function block to increase the security of data. It is a cryptosystem that is widely used in private networks to secure the data from several attacks, including denial of service. The author reported that this cryptosystem is insecure against algebraic types of attacks.

L. Ding *et al.* [19] proposed a modified WG-8 algorithm that is a variant of a known WG stream cipher method. This method has a relatively simple structure and uses 80 bits of secrete key and 80 bits of initial vectors (IV) as an input. The author figured out that these keys are generated in linear and nonlinear relationships. Hence, it is easily identified by the attacker in less than three minutes and vulnerable to key attacks. T. Suzaki *et al.* [20] stated that TWINE is another lightweight block cipher that supports 80 and 128-bit keys. This algorithm is platform-dependent, so it is efficient on the unification of encryption and decryption on software. The authors analyzed the security of the method against impossible deferential and saturation attacks, and they found that its security level should be improved in the future. William Diehl et al. [21] reported that TWINE has the highest Throughput-to-Area (TP/A) ratio as compared with the other stream ciphers. L. Singh and M. Singh [22] stated Elliptic Curve Cryptography (ECC) is a public-key cryptography that uses the curve equation $y^2 = x^3 + ax + b$. This is known as Weierstrass equation, with $4a^3 + 27b^2 \neq 0$, where a, b are constants. ECC can be implemented in various tasks of public-key cryptography like encryption, authentication, digital signature, and key agreement.

T. Daisy *et al.* [23] stated and defined the ECC equation degree and functionality. They conclude that higher-order encryption and decryption is possible provided that the equation should satisfy the standards proposed by NIST [24]. A. Bansal [25] proposed a security system based on the integration of ECC with the cloud. The author used ECC only to encrypt some bits of the data block in order to reduce computational cost on the client-side, which increases resource sharing in the cloud. The author reported that this security method is more effective than RSA with regard to computational cost. S. Singh. *et al.* [26] carried out a comparison between ECC and RSA after a thorough experimental analysis. They concluded that ECC is superior over the RSA method in terms of the ciphertext sizes, digital signature signing, and verification time. This makes ECC ideal for replacing the RSA method. S. Shankar *et al.* [27] proposed ECC based secure key distribution model for secure medical data transmission and resolving replay attacks by using timestamp values. P. Dhillon and S. Kalra [28] stated it is 10,000 times more difficult to break ECC as compared to an equivalent 2048-bit RSA without degrading the system performance. H. Hasan *et al.* [29] proposed BROSMAP protocol based on ECC for securing IoT. The authors stated that ECC is four times faster than RSA because it uses a smaller key size. M. Ahmed. *et al.* [30] proposed a payload based mutual authentication method that used the Advanced Encryption Standard (AES) with 128 bits key. P. Patil *et al.* [31] reported AES utilizes more computational resources. This indicates that a lightweight model is necessary to have a secure IoT environment.

M. Bunder *et al.* [32] reported that RSA is the first public-key cryptosystem. D. John *et al.* [33] proposed a modified RSA algorithm called binary RSA encryption for securing data in transit in wireless networks. A. Mektoubi *et al.* [34] proposed a hybrid algorithm that combined RSA and ECC methods. RSA is used to encrypt/decrypt the data and the ECC for digital signature, which exploits MQTT (Message Queuing Telemetry Transport) protocol. J. Wang *et al.* [35] discussed that the security of ECC is mainly based on the intractability of the Elliptic Curve Discrete Logarithm Problem (ECDLP). The author also stated that it is easy to calculate another point on the elliptic curve with point-multiplication defined on the finite field. Thus, it may not be appropriate to use for data integrity.

S. Singh *et al.* [36] and R. Tso and Y. Jheng [37] proposed the literature of the Nth-degree Truncated Polynomial Ring Units (NTRU) algorithm that is based on the difficulty of finding particularly small vectors in lattices and degree N with coefficients in the domain of $Z[X]/(X^N\text{-}1)$. The authors stated that it is the first system that does not depend on complicated mathematical problems. R. Chen and D. Peng [38] proposed an NTRU based authentication scheme that is used in the handover process of the wireless network. The authors stated that the conventional methods are unsafe with regard to quantum attacks, whereas the NTRU is a lightweight public-key cryptography algorithm based on lattice theory, which is powerful against quantum adversaries. Authors concluded that NTRU has many distinct benefits, such as less memory and computing requirements, higher encryption, and decryption. Y. Gao,X. Yin, and C. Hao [39] have conducted a security analysis to prove that NTRU is secure under the assumption of the NTRU lattice Approximate Closest Vector Problem. Further, R. Hassan and M.G Nadia. [40] described the NTRU algorithm that is based on the mathematical technique called polynomial algebra. R. Jha and A. K. Saini [41] conducted a comparison analysis and concluded that the NTRU algorithm is better than RSA and ECC. The authors concluded that NTRU security does not rely on the integer factorization problem or discrete logarithm problem. They mentioned that NTRU requires O(n log n) time, whereas the RSA, ECC, etc. requires $O(n^3)$ time. This makes NTRU ideal for constrained devices such as IoT sensors nodes and sensor-embedded devices.

## III. Proposed Work

Related work depicts that privacy and confidentiality should be considered as a fundamental security aspect of IoT and its applications. Cryptographic techniques are used to secure sensor's data and data in transmission. Traditional cryptographic algorithms such as AES, DES, Blowfish, 3DES provided admirable security, but they demand higher computational cost. ECC and NTRU, on the other hand, are ideal for constrained devices such as IoT sensors and sensor-embedded devices. Several security methods have been developed to ensure the confidentiality and integrity of the data for IoT applications. There are some issues with them:

- Most of the methods employ complex mathematical calculations and are based on prime factorization. Moreover, these methods require higher computational resources, time, and even are susceptible to IoT related attacks. For instance, zero-day attacks and several variants of existing attacks are the main issues of the applications.
- It is quite important to consider the sensors power, memory, etc. since it is employed at resource-constrained devices and in real-time operations. Hence, security models should demand low power, memory, and execution time without compromising security.

The proposed work TD²SecIoT makes it different from state-of-the-arts on the following points:

- Mutual authentication is made between two communicating parties without assistance from a certification authority, except the first time when the system is setup. So, the use of the Key Distribution Centre (KDC) is reduced.
- Two motes can generate keys between themselves by sensed data values.
- TD²SecIoT demands a smaller key size with less key generation, authentication, encryption/decryption time, as compared with other existing works. Hence, demand less computation time, energy requirement, and memory.
- It also prevents numerous attacks such as Reply attack, Man in the Middle (MiTM) attack, chosen cipher attack, Quantum attack, and Lattice-Based attack, which makes it efficient for securing resource-constrained IoT sensors and actuators.

Basically, IoT architecture has four layers that are used to manage IoT services. Such conventional architecture lacks security aspects [1], [2], [7]. The proposed TD²SecIoT offers to integrate a new layer called "Security Layer" in between the Perception layer and Network layer, as depicted in Fig. 1.

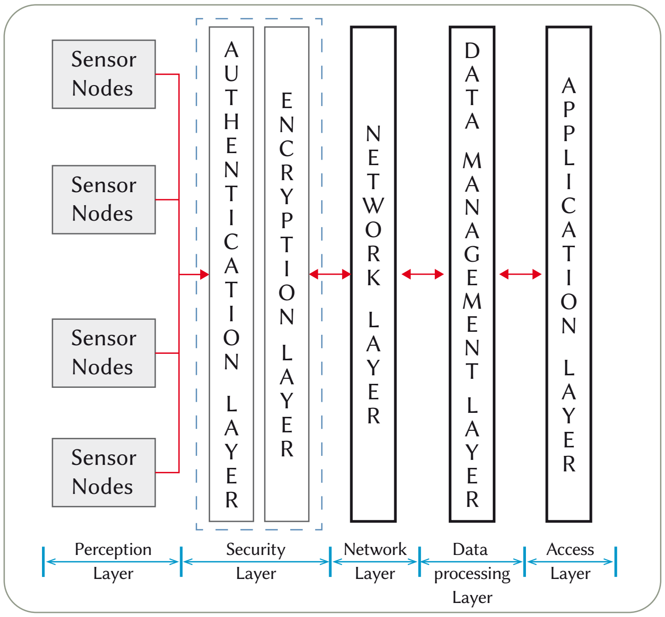


Fig. 1. The proposed architecture.

The security layer is further divided into two sub-layers as it is illustrated in Fig. 1. These layers constitute the Authentication layer; a layer that is responsible for the mutual authentication between the source and receiver nodes, and the Encryption layer; which is responsible for ensuring data integrity and confidentiality during transit. Each sensor node will have both the capabilities since they are embedded in every IoT devices that are deployed in smart environments or any industrial application. In this work, industrial IoT devices are considered for simulating our novel method TD²SecIoT.

The proposed security augmented architecture, which consists of authentication and encryption sub-layer, is discussed in the following sub-sections.

## A. Authentication Sub-layer

The TD²SecIoT proposes mutual authentication as one of its tasks, which establishes an authentic two-way communication between the source and receiver node, as depicted in Fig. 2. Once a new node is connected to the network, the third party, called Key Distribution Centre (KDC) is responsible for assigning key pair that will become a certificate for that node. These keys are used during the re-certification at the time of network reset or restart. In this regard, we proposed an ECC based mutual authentication for authentic communication between the source and receiver node.

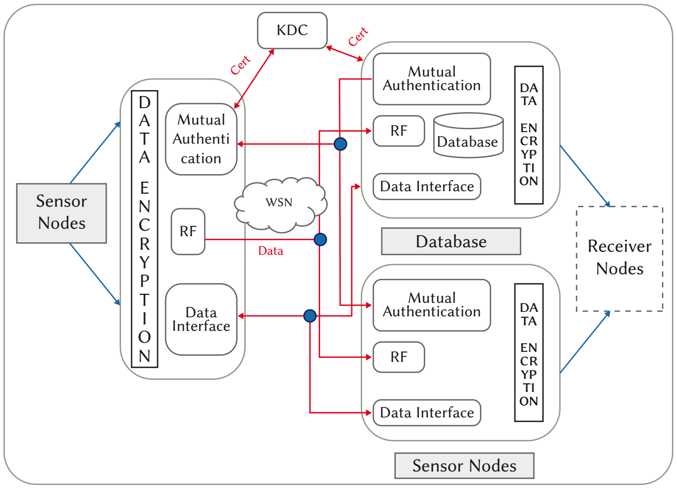


Fig. 2. The proposed Mutual Authentication Process.

### 1. Mutual Authentication Setup

The mutual authentication process begins by taking a key pair (certificate) from KDC as the network is reset/restarted, or any new node joins the communication. Source node and receiver node both send their public keys $Q_{src}$ and $Q_{rcv}$ to KDC to enable mutual authentication. The KDC uses its private key to generate the certificates and calls the ECDSA method to generate certificates and transfer certificates ($r_{src}$, $s_{src}$) to the source node. Similarly, ($r_{rcv}$, $s_{rcv}$) are transferred to receiver node with its own public key. The certificate consists of a pair of integers ($r_{src}$, $s_{src}$) for the source node and ($r_{rcv}$, $s_{rcv}$) for the receiver node. Here, $r_{rcv}$ and $s_{rcv}$ are the x coordinates of the distinct elliptic curve points. These certificates and a public key of KDC are used at the time of mutual authentication process when they send a certificate to each other. They use $Q_{ca}$ (public key of KDC) in the verification phase.

### 2. Mutual Authentication Between Nodes

As source and receiver nodes are ready to communicate, they need to calculate the hash value using key pairs (obtained from the KDC), and further exchange and verify to each other. If the hash is matched, both devices become authentic and start communication for data sharing. The brief steps of the process are:

The mutual authentication process starts with exchanging public keys $Q_{src}$ and $Q_{rcv}$, between Source (SRC) and Receiver (RCV) nodes. Further, they generate the secret key by multiplying their own private key to other's public key. The secret key is used to send the hash in encrypted form. Now, the RCV node will send the certificate/hash $H_{rcv}$ (hash of the receiver) to SRC in encrypted form. The SRC then decrypts $H_{rcv}$ and obtains the certificate of RCV node.

Furthermore, the SRC encrypts its certificate/hash and sends $H_{src}$ to RCV, which further decrypts it and obtains the SRC's certificate. Then, RCV checks received hash, and if found identical, then SRC and RCV authenticate to each other and are ready to start communication (IoT data transaction). Otherwise, it rejects the $H_{rcv}$ (send the negative acknowledgment to SRC). It may restart the mutual authentication process if required.

### 3. Mutual Re-authentication

Mutual Re-Authentication between nodes will happen only when a new node is trying to join the network for the first time or when some failure occurs (network failure, system failure, etc.). This will force each node in the network to re-authenticate each other by exploiting the same authentication procedures, as discussed in subsection A-2.

## B. Encryption Sub-layer

This section deals with the integrity of data being transferred among nodes in the IIoT system, which is another task of TD²SecIoT. The data is gotten encrypted at the source node using NTRU approach through a dynamic key, which is generated using the previously stored data in the database, as shown in Fig. 3.

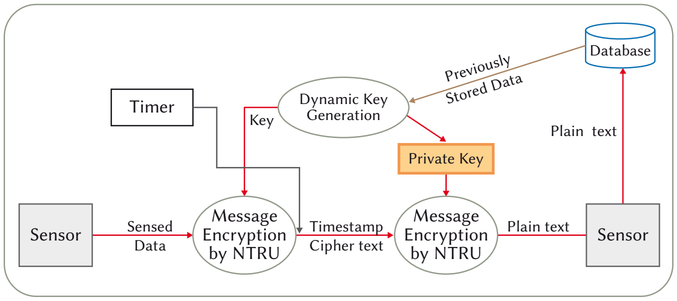


Fig. 3. Encryption Sub-layer.

Another feature of this sublayer is to provide a timestamp to each packet that increases the security level of the proposed system. System time (via timer) is stamped on the ciphertext generated so that the receiver understands that data does not eavesdrop during transmission. This mechanism is useful in identifying replay attacks [27]. Furthermore, timestamped cipher text routes to the receiver node. The communication methods represented by arrows in the Fig. 3 shows the transfer of packets between the sensor and receiver node using radio communication. The receiver node, which is another IoT mote or a server, first delimits timestamped ciphertext to obtain the actual text message and then decrypts the ciphertext using NTRU algorithm. The private key is used for such encryption, which is generated during key generation. Further, decrypted plain text has to be forwarded to the next IIoT layer, where it gets to settle down in repository.

RF propagation time has been used in work for the timestamp calculation [42], which determines the maximum time required for delivery. This timestamp value is used to check a possible reply attack by comparing the timestamp and current time. The Cooja simulation platform [43] facilitates to calculate the shortest distance between the two most distant motes in the network (aka. Network Diameter). The propagation time calculation steps are summarized as follows:

*RF propagation time = range / RF wave speed*

*Where:*

*RF wave speed = 3x108 m/s*

*RF propagation: maximum time for propagation*

*Range = Network Diameter*

Finally, the difference in delimited timestamp and current time must be less than or equal to RF propagation time.

## C. Algorithmic Procedure For Encryption

Consider, a sender (source node) needs to send confidential data (encrypted by NTRU cryptosystem) to a receiver node. At this point, the source employs the receiver's credential that is a public key, to encrypt the data. The NTRU cryptosystem consists of three phases: System Setup, Key Generation, Encryption, and Decryption that is described in following subsections 1, 2, and 3 with pseudocode. At the end of this section, we also described the key-regeneration process in subsection 4.

### 1. System-Setup Procedure

Every new node needs to set up three integers (*N, p, q*) and four polynomial sets ($L_f$ , $L_g$ ,$L_r$ , $L_m$) from the ring R = Z[X]/($X^N$ - 1), where 'N', 'p' and 'q' are prime numbers, and 'q' is considerably larger than 'p' with gcd(*p, q*)=*1.* Usually, 'p' is set to 3 (ternary polynomials), and 'q' for efficiency reasons is generally set to an integer number power of two i.e., q =$2^k$. The prime condition of product f * $f_p$=1 (mod p) and f * $f_q$ =1(mod q) must hold in the key generation process.

The following procedure has been used to select the next prime number from any randomly selected sensed data and to assign the value for N. The minimum value of N is set to 167. Shen et al. [44] suggest various security levels and found N= 167 give a better security level. Similarly, for standard, high, and highest security level, 251, 347, and 503 are suggested, respectively.

```
Input: integer n
 Algorithm:
 {Set n ← stored random value}
 {Set x | x ← 1000}
 {Set f | f ← 0}
 {Set i | i ← n+1}
 {Set j | j ← 2}
  For each i loop i to x do
      For each j loop j to j <i do
          If i mod j=0 then
              Set {f ← 1} && break the loop;
          Else
              Return i
          End if
End for
  End for
 {Set N ← i}
Output: N (A prime number)
```

Now, the initial parameter <N, p, q> are passed to the key generation and encryption/decryption operation.

### 2. Key Generation

This process performs the following procedure to generate keys (public and a private key) that are used for encryption and decryption process:

```
Input: A mote picks two random polynomials f ∈ Lf, g ∈ Lg
Output: The private key (f, fp) and mote's public key h.
Process:
1. Receiver mote computes:
   fq & fp satisfying fq * f ≡ 1 (mod q) and fp * f ≡ 1 (mod p).
2. Receiver mote computes:
   h = fq *g (mod q), this is a public key, and the private key is (f, fp).
```

### 3. Encryption and Decryption Operation

Source mote encrypts message m ∈ M with coefficients {-1, 0, 1} using NTRU encryption algorithm. The key generation procedure is described below:

```
NTRU Encryption algorithm
Input: m ∈ M, h, interval [-p/2, p/2], r, <N, p, q>
1. Select random polynomial 'r' with small coefficient not restricted
   to the set {-1, 0, 1}
2. Compute
   c = pr * h + m (mod q) = r * h+ m (mod q)
3. Attach timestamp,   c= (c||t)
4. Transfer 'c' to receiver.
Output: Cipher Text 'c'
```

The receiver mote decrypts ciphertext 'c' using the decryption function of NTRU, the receiver's private key, and parameters established during the setup procedure. The key generation procedure is as follows:

```
NTRU Decryption algorithm
Input: c, r, interval [-q/2, q/2], f
1. Select random polynomial 'r'
2. Compute
   a = f * c (mod q)
   a = f * (pr * h + m) (mod q)
   a = f * (pr * fq * g + m) (mod q)
   a = pr * g + f * m (mod q)
3. Calculate
   b = a (mod p) = f * m (mod p)
   , hence pr * g (mod p) = 0 & f * $f_p$ = 1
   b = pr * g + f * m (mod p)
   b = f * m (mod p)
   m = fp * b (mod p)
   m= m (mod p)
4. Transfer (m) to receiver or database.
Output: Plain Text 'm'
```

The aforesaid encryption and decryption procedure are shown diagrammatically in the Fig. 4.

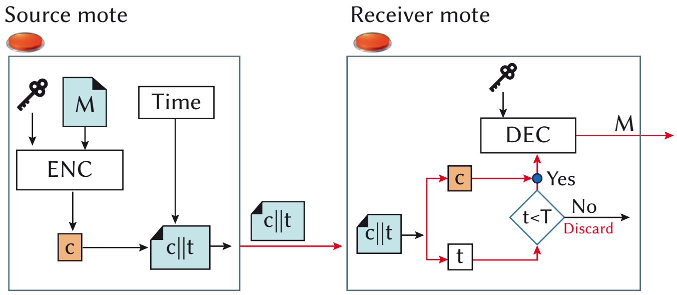


Fig. 4. Encryption and Decryption Process.

The database scheme uses specific sensed data values for a selection of a new key pair for NTRU encryption/decryption. Data exchange between connected nodes will continue with current parameters and keys until a fixed minimum or maximum period is over. Such a key pair is used for NTRU Encryption and Decryption during this interval.

### *4. Key Regeneration*

The new key pair is needed to be generated based on two following factors:

(1) The minimum allotted period is over, and the maximum allotted period is not over. For example, it was decided that one set of keys will be used for a minimum of 7 hours and a maximum of 8 hours. Then a new key pair will be regenerated between this period from the previously stored data in the database server.

(2) Key regeneration gets initiated, when a constantly very small amount of data without significant change is collected or sensed for some continuous period of time. Such continuous similar values for a while (assume 10 minutes) indicates that most of the devices are stopped. It may happen due to employee shift change (since most of the industries are operating in this fashion).

## IV. Simulation Setup & Configuration

Contiki-OS (a Linux based operating system) with a Cooja network simulator has been used for implementation [43]. A virtual machine of 64 bits Ubuntu operating system called Instant Contiki-2.7 is installed on a 64bits-x64 based processor. 8Gb RAM and Intel(R) Core (TM) i7-7500 CPU @ 2.90 GHz processor has been used. All the necessary packages and libraries of the Contiki-OS and Cooja simulator are also installed with an updated Java run time environment and open JDK-8 that supports the graphical representation of the Cooja simulator for the visualization of the radio traffic, packet flow, etc. A C-language code is written for encryption/decryption and mutual authentication and is compiled with the MSPGCC compiler (version 4.6.3).

Another useful tool, 'Power Tracker', is used, which is available in the menu Tools with the name "Mote Radio Duty Cycle" [45]. This tool allows us to measure the mean energy consumed by all the nodes and also measure the power consumption individual node wise.

Further, the network is built by inserting nodes in the Network window. We used Sky mote type because it supports 6LoWPAN, which is an advanced version of IPv6 [43]. It works with low-power radio frequency at the physical layer. We have implemented TD$^2$SecIoT using IPv6, because it facilitates IoT devices communication over the Internet separately one at a time. A Unit Disk Graph Medium (UDGM) has been used as a radio frequency model, which takes the ideal transmission range disk in which motes are allowed to share the data [45].

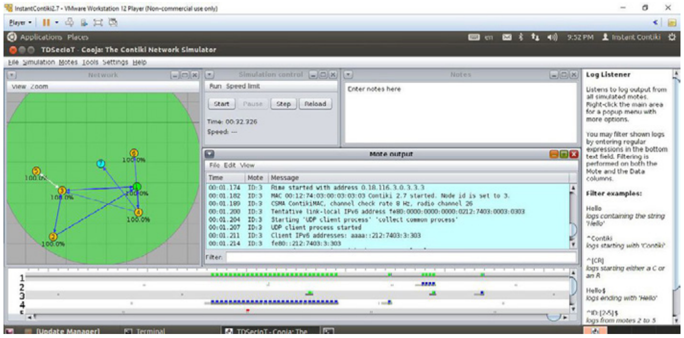
Tmote sky mote has the ability of fast wake up from sleep, which takes less than 6μs, and it is equipped with a 16 bits RISC processor that demands less power during active and in sleep mode [46]. Tmote Sky is equipped with an integrated onboard antenna, with a 50m range indoors and 125m range outdoors, radio (Chipcon CC2420), which provides reliable wireless communication. In this implementation, we use the Sky mote with (8MHz MSP430) low power microcontroller, 10KB RAM, and 48KB flash memory.

Sky motes also provide 250Kbps, 2.4GHz, and IEEE 802.15.4. There are three types of integrated sensors in Tmote Sky, includes Humidity, Temperature, and Light sensors. Tmote Sky provides ST M25P80 40MHz serial code flash for external code and data storage that can hold 1024 kilobytes of data. Sky motes also provide 250Kbps, 2.4GHz, and IEEE 802.15.4. This flash is divided into 16 segments, each with 64kB in size. Initially, a border router and a server are included in the simulated network. Additionally, it has been included five more sensor nodes of the type ''Sky Mote'' in the simulation to collect the data. We have also chosen the program 'temp-sensor.c', which is part of the Contiki OS and compiled on the sky mote 6. This application is used to sensor the real-time temperature of the environment where the mote is placed.

### *A. Algorithmic Simulation*

A C-language code is written for ECC and NTRU for the mutual authentication and further compiled on each mote in the network. We have deployed five motes as clients, one border router mote with compiled 'border-router.c' as a gateway that uses RPL for routing the packets, and one server mote as a web server (complied on sky motes).

The motes start transferring sensed data once the mutual authentication is successfully completed. This data is encrypted using NTRU cryptosystem, as described in section III. The simulation of the proposed work "TD$^2$SecIoT" using the Cooja network simulator of the Contiki Operating System (OS) with multiple windows is shown in Fig. 5. The left-top corner is the "Network Window", where network topology or motes positioning are shown. It allows us to access every mote in our network. The next window is the ''Simulation Control window," which facilitates 'start', 'pause', make a 'step' forward, and 'reload' the simulation. The window on the top-right corner is used to take notes about the simulation called "Notes Window". The window in the middle is called "Mote output", where the outputs of serial ports of each node are displayed. The lower window is the "Timeline Window|" that shows the data packets delivered over time from the source sensor node to the receiver node. This window is used to investigate the time taken for authentication and encryption/decryption of sensed data. Furthermore, a program called 'tunslip6', home/user/Contiki-2.7/tools/tunslip.c, is used in order to connect the border router with Cooja.



```
Time        Mote   Message
00:01.174   ID:3   Rime started with address 0.18.116.3.0.3.3.3
00:01.182   ID:3   MAC 00:12:74:03:00:03:03:03 Contiki 2.7 started. Node id is set to 3.
00:01.189   ID:3   CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26
00:01.200   ID:3   Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7403:0003:0303
00:01.204   ID:3   Starting 'UDP client process' 'collect common process'
00:01.207   ID:3   UDP client process started
00:01.211   ID:3   Client IPv6 addresses: aaaa::212:7403:3:303
00:01.214   ID:3   fe80::212:7403:3:303
```

**Log Listener**

Listens to log output from all simulated motes. Right-click the main area for a popup menu with more options.

You may filter shown logs by entering regular expressions in the bottom text field. Filtering is performed on both the Mote and the Data columns.

**Filter examples:**

Hello
*logs containing the string 'Hello'*

^Contiki
*logs starting with 'Contiki'*

^[CR]
*logs starting either a C or an R*

Hello$
*logs ending with 'Hello'*

^ID:[2-5]$
*logs from motes 2 to 5*

Fig. 5. TD$^2$SecIoT simulation on Cooja.

Table I summarizes the configuration setup, protocols, key exchange times, IP version, and the preferred distance between the motes.

TABLE I. Summary of the Simulation Configuration

| Parameters | Value |
|---|---|
| Motes | Sky |
| TX Ranges | 10m |
| IP Version | IPV6 |
| MAC Layer | IEEE 802.15.4 |
| Network Layer | 6LoWPAN |
| Radio Access | CSMA |
| Duty Cycle | Contiki MAC |
| Maximum Neighbors | 5 |
| Key Exchange life | 28,700-28,900 sec. time |

### *B. Database Implementation*

Contiki has a built-in database management system called "Antelope"; a lightweight SQL-like database manager for resource-constrained IoT devices, which enables a sensor or a mote to act as a database server [43]. It is a relational database management system that enables the dynamic creation of databases and complex data querying.

Additionally, a centralized database is designed and implemented using Contiki to store the sensed data (received from the sensor mote). This data consists of information about Sensor Nodes (node ID), Node Sequence (node_Seq), Sensed Temperature (node_TempC), and Sensed Light value of the node (node_Light) and it is represented using a relational database as depicted in Table II.

TABLE II. Relational Database Under Contiki

| node_ID | node_Seq | node_TempC | node_Light |
|---|---|---|---|
| --- | --- | --- | --- |

Antelope procedure (Contiki directory /home/user/ Contiki-2.7/ examples/antelope/netdb/netdb.csc) has been used to store the sensed data from the sender node to sky server mote6. Furthermore, a python code 'Pydev' is written to get data from the sky mote server and further saves it into the MySQL database. Python needs an extension called 'pySerial' in order to communicate with the serial port of the sky server mote. In this regard, Python extension 'pyserial-3.1.1.tar.gz' has been installed. 'MySQL-python-1.2.4b4.tar.gz' file is utilized to connect the database to the MySQL library. The following commands in MySQL are written to create the database on sky mote6:

CREATE DATABASE mote6; -------------------------------------------(i)

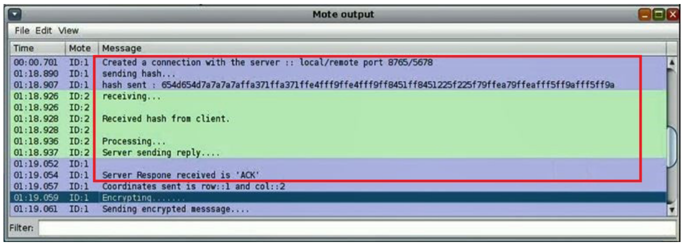
USE mote6; --------------------------------------------------------------(ii)

Create table BarelTempData (NODE_ID INT PRIMARY KEY AUTO_INCREMENT, NODE_SEQ INT (11) NOT NULL, NODE_TEMPC INT (4) NOT NULL, NODE_LIGHT INT (4) NOT NULL) -------------------(iii)

The command (i) creates a database called mote6, command (ii) make mote6 usable for database, and command (iii) creates a table named BarelTempData. This table has four fields Node_id, Node_sequence, Node_tamprature, and Node_light. Finally, we run the code "python serial-mysql.py" for transferring the sensed data and execute the query 'select * from BarelTempData' in MySQL to retrieve the data as shown in Fig. 13.

## V. Experimental Results

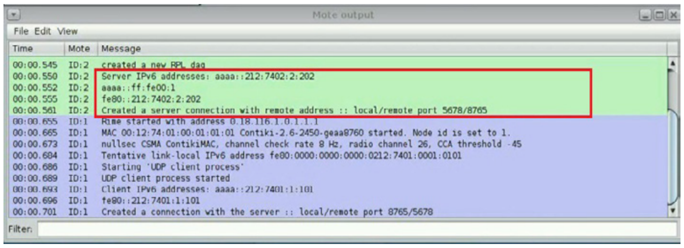
Fig. 6 depicts the simulation of the mote output. The first four lines show the connection establishment between two motes. Further, the paired server mote waits for the authentication request from the client to process the mutual authentication between motes in the network.



```
Time       Mote  Message
00:00.701  ID:1  Created a connection with the server :: local/remote port 8765/5678
01:18.890  ID:1  sending hash...
01:18.907  ID:1  hash sent : 654d654d7a7a7a7affa371ffa371ffe4fff9ffe4fff9ff8451ff8451225f225f79ffea79ffeafff5ff9afff5ff9a
01:18.926  ID:2  receiving...
01:18.926  ID:2
01:18.928  ID:2  Received hash from client.
01:18.928  ID:2
01:18.936  ID:2  Processing...
01:18.937  ID:2  Server sending reply....
01:19.052  ID:1
01:19.054  ID:1  Server Respone received is 'ACK'
01:19.057  ID:1  Coordinates sent is row::1 and col::2
01:19.059  ID:1  Encrypting.......
01:19.061  ID:1  Sending encrypted messsage....
```

Fig. 6. Connection Establishment of the Motes.

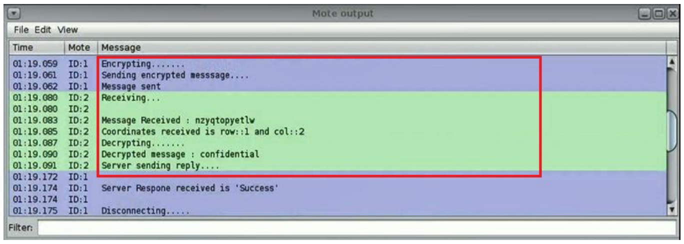
This process takes very less time (in milliseconds) to establish the connection, as indicated in Fig. 6. The further authentication process is shown in Fig. 7.



```
Time       Mote  Message
00:00.545  ID:2  created a new RPL dag
00:00.550  ID:2  Server IPv6 addresses: aaaa::212:7402:2:202
00:00.552  ID:2  aaaa::ff:fe00:1
00:00.555  ID:2  fe80::212:7402:2:202
00:00.561  ID:2  Created a server connection with remote address :: local/remote port 5678/8765
00:00.655  ID:1  Rime started with address 0.18.116.1.0.1.1.1
00:00.665  ID:1  MAC 00:12:74:01:00:01:01:01 Contiki-2.6-2450-geaa8760 started. Node id is set to 1.
00:00.673  ID:1  nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.684  ID:1  Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7401:0001:0101
00:00.686  ID:1  Starting 'UDP client process'
00:00.689  ID:1  UDP client process started
00:00.693  ID:1  Client IPv6 addresses: aaaa::212:7401:1:101
00:00.696  ID:1  fe80::212:7401:1:101
00:00.701  ID:1  Created a connection with the server :: local/remote port 8765/5678
```

Fig. 7. Mutual Authentication Process.

Fig. 7 depicts the remaining authentication process and the time taken by devices (client, server motes etc.) to get authenticated to each other. Once the connection is established (as shown in Fig. 6), the client sends the authentication request to the server with hash. The server verifies the hash and sends the positive acknowledgment if it is matched. This indicates mutual authentication between motes is successfully established. Motes now share any sensed data by using the NTRU encryption and decryption method.



```
Time       Mote  Message
01:19.059  ID:1  Encrypting.......
01:19.061  ID:1  Sending encrypted messsage....
01:19.062  ID:1  Message sent
01:19.080  ID:2  Receiving...
01:19.080  ID:2
01:19.083  ID:2  Message Received : nzyqtopyetlw
01:19.085  ID:2  Coordinates received is row::1 and col::2
01:19.087  ID:2  Decrypting.......
01:19.090  ID:2  Decrypted message : confidential
01:19.091  ID:2  Server sending reply....
01:19.172  ID:1
01:19.174  ID:1  Server Respone received is 'Success'
01:19.174  ID:1
01:19.175  ID:1  Disconnecting.....
```

Fig. 8. Encryption Decryption Process.

Fig. 8 depicts the encryption and decryption of the proposed work and the time taken by the method to complete the processes. The encryption and decryption time of the plain text is 2 ms and 7 ms, respectively, as shown in Fig. 8. It has been observed that the proposed method is faster and more secure than the other methods [22], [47], [48], and [49].

The maximum stack usage in Cooja is the estimated RAM consumption. Figu. 9 depicts the stack usage of mote, where the Y-axis shows the stack usage of a mote in bits, and the X-axis shows the execution time to transfer the data. We have seen that the proposed method is in-line with the mote's memory specification, and the code can be emulated on the actual motes.

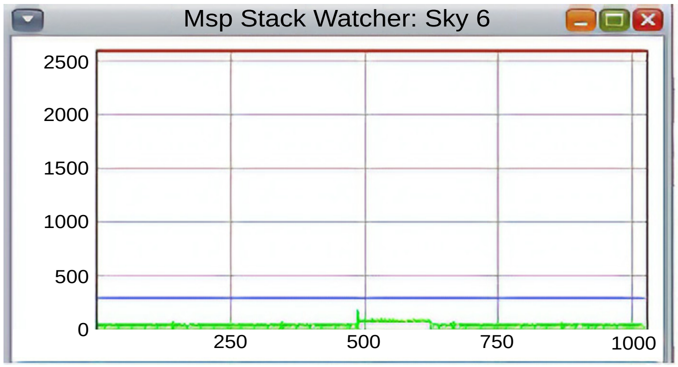


Fig. 9. Stack usage of motes.

Fig. 10 depicts the average power consumption (Milli-watt (mW)) of various states. For example, the energy needed for CPU in sensor motes, Transition energy, Listen energy, and LPM (Low Power Mode). Motes are indicating on the x-axis, and the four colors indicate the power consumption of the motes at different states. The following formula and procedure were employed in the presented research work

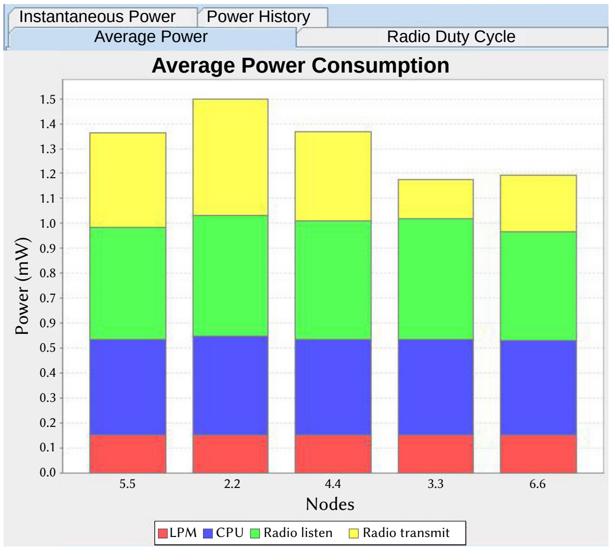


Fig. 10. Average Power Consumption of five sensor nodes.

to calculate the corresponding energy consumption for each one of the above states:

$$CPU\ Energy\ (CP)=c*1.8tm$$

$$\text{LPM Energy (LP)}=1*0.545tm$$

$$\text{Transmit Energy (LT)}=t*17.7tm$$

$$\text{Listen Energy (LR)}=r*20tm$$

$$Total\ Energy=CP+LP+LT+LR, \text{ and}$$

$$tm=c+l$$

Where:

*tm*: total time,

*c*: CPU time

*l*: is the time that the sensor was in Low Power Mode (LPM)

*t*: transmit time

*r*: Listen time

Based on the results obtained from the simulation of the proposed work (shown in Fig. 10), it is clear that the nodes demand less power to transmit (0.49mw) and receive data (0.35mw). Finally, the whole process in the proposed method takes a total of 1.41mw. This has a direct impact on motes life. The proposed method has reduced the power consumption (71.8%) as compared to the work presented in [48]. It is worth noticing that the proposed method does not compromise the security level.

Fig. 11 depicts the bar graph to show the number of packets received by the motes on the x-axis. The red bar at the y-axis indicates received packets, and the blue bars indicate the retransmitted or duplicate packets. In the simulation, we have seen that there is no packet duplicity (retransmission) to the mote, so the number of packets retransmitted is zero. UDP uses unreliable packet delivery, which results in packet loss due to electrical interference, congestion, or physical disconnection, etc. This indicates that the proposed method reduces congestion and physical disconnection. It is efficient for preventing the retransmission of the packets. Further, it has a great impact on the processing and lifetime of the motes with reliability.

The temperature captured by motes is shown in Fig. 12, where the x-axis indicates the time of sensing and y-axis shows the temperature range in Celsius. This data is stored temporarily in the mote configured with an antelope database. Further, it is sent to MySQL database server after connecting mote's serial port to the database as shown in Fig. 13.

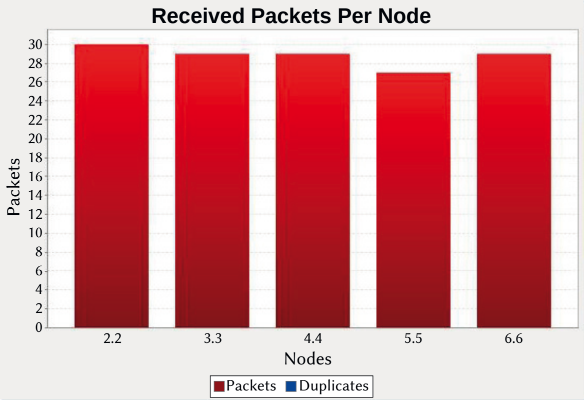


Fig. 11. Number of packets received by each mote.

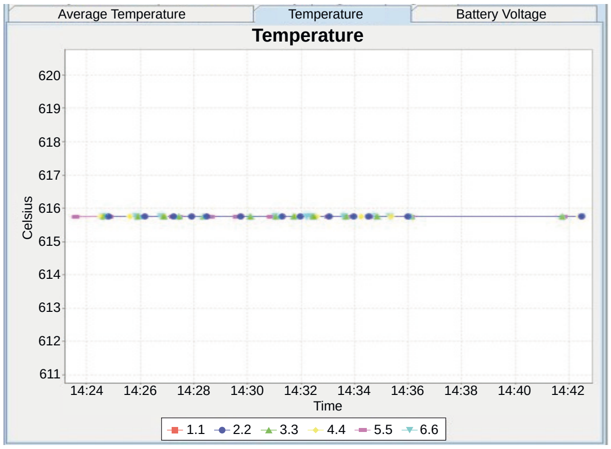
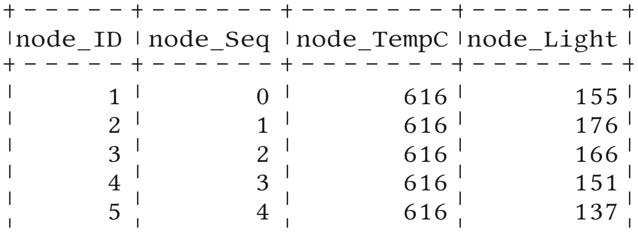


Fig. 12. Sensed temperature values by six motes.



```
+------+-------+--------+--------+
|node_ID |node_Seq |node_TempC |node_Light |
+------+-------+--------+--------+
|     1 |       0 |       616 |       155 |
|     2 |       1 |       616 |       176 |
|     3 |       2 |       616 |       166 |
|     4 |       3 |       616 |       151 |
|     5 |       4 |       616 |       137 |
```

Fig. 13. Antelope database output of temperature values.

In Fig. 12, the doted colors indicate different motes with data at a given time interval. We simulate it with temperature ranges for the operational barrel temperature zones in the first 14:24ms. It has been observed that motes have sensed the higher temperature in degrees Celsius, as it is recorded in Fig. 13. We used this data for generating the dynamic key that elevates the security level of the proposed method TD$^2$SecIoT.

TABLE III. Comparative Analysis of TD$^2$SecIoT and State of the Art Works

| Parameters | **TD$^2$SecIoT** | [22] | [47] | [48] | [49] |
|---|---|---|---|---|---|
| Key generation(s) | 0.195 | 0.195 | 0.20161 | NA | 0.77 |
| Authentication(s) | 0.0215 | NA | NA | NA | 0.22 |
| Encryption (s) | 0.002 | 0.093 | 0.41137 | NA | NA |
| Decryption (s) | 0.007 | 0.14 | 0.37679 | NA | NA |
| Power(mw) | 1.41 | NA | NA | 5.0 | NA |

**NA: Not Available, s: time in second, mw: Milli-watt**

Fig. 6, 7 and 8 show timing (in millisecond) at left most of the side, that is converted into seconds for the sake of comparison. This time is used to record key generation time, mutual authentication time, as well as encryption/decryption time. Table III represents the comparative analysis of proposed TD²SecIoT with state-of-the-art works [22], [47], [48], and [49].

The same result can also be visualized as depicted in Fig. 14. It is clearly observed that the proposed model (TD²SecIoT) performed better than other works.

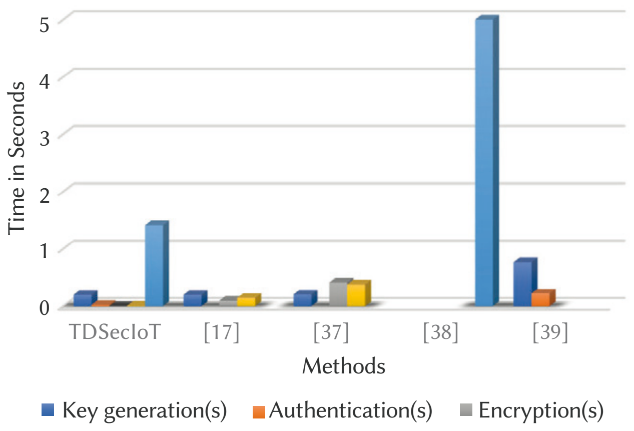


Fig. 14. Comparative Analysis of TD²SecIoT.

## VI. Security Analysis & Discussion

The security analysis of the TD²SecIoT against IoT attacks has been analyzed as follows:

**Replay Attack:** It comes under the network attack where attackers try to delay valid data transmission repeatedly so that it seems fraudulent [50], [51], [52]. This process will take some time to modify or generate new data [53]. TD²SecIoT incorporates timestamp (attached to the ciphertext) to prevent replay attacks. It works because the replay causes a time greater than the maximum propagation time, so the receiver will discard the ciphertext without any further processes.

**Man-in-The-Middle Attack:** An attacker observes a session opening on a network. Once the authentication process is completed between two parties, it attacks the client computer to immobilize it through IP spoofing (claim the legitimate client) [50], [54]. This attack has been handled under TD²SecIoT, where the SRC sends the message containing its identity directly in an encrypted form. The man in the middle cannot decipher the message, because it does not have the private key of the receiver nor the sensed data values by which keys pairs are generated.

**Security against Chosen Cipher Text Attacks**: The proposed scheme is secure against chosen-ciphertext attacks [51], [55]. If the attacker anyhow gets the key from the previous ciphertext and tries to decipher the new ciphertext, it cannot be able to decrypt it because the proposed work uses a dynamic and data-driven key feature.

**Key-Recovery Attack:** It is an attack that attempts to recover the cryptographic key of an encryption scheme [50]. Since we have used a dynamic key that changes with the specified time and is generated using the previously-stored sensed data, it is difficult for the adversary to recover the keys.

**Non-Repudiation:** Non-repudiation is the assurance that someone cannot deny something [50], [53]. That is, the SRC mote cannot deny that ciphertext is not sent by it. The KDC or receiver mote himself can verify that the SRC sends the ciphertext through the verification procedure.

**Identity Attacks:** It is also called 'spoofing.' An attack is trying to gain access to a node by masquerading as an authenticated one [50], [51], [55]. The proposed ECDSA authentication scheme makes it hard to masquerade. Attackers first need to contact KDC to get a certificate and key pair that is used at the time of mutual authentication. Thus, our scheme is more secure against these types of attacks.

**Confidentiality:** The proposed scheme employs ECC based process to generate key pairs for sharing an encrypted certificate issued by the KDC. It is further used for mutual authentication that ensures all nodes in the network are authenticated. Data transfer between the nodes is possible after encrypting the data, which makes network analysis more difficult for the adversary. Thus, the proposed scheme provides data confidentiality.

**Integrity:** It ensures that an unauthorized node does not alter information. The proposed scheme uses NTRU Encrypt for encryption and decryption processes to ensure integrity. If an unauthorized node has altered timestamped ciphertext from C to C', it can be revealed during the decryption process by inspecting the timestamp value. This alteration can also be checked at the time of the verification process. It might be denied by the receiver to accept, and hence integrity is ensured.

Several security methods have been proposed by other researchers to ensure confidentiality and integrity against various types of attacks in IoT applications. Some state-of-the-art works have been compared with proposed TD²SecIoT and presented in Table IV. TD²SecIoT offers stronger Confidentiality, Integrity, Unforgeability, and Non-repudiation than existing techniques [23], [27], [56]. The standard ECC and NTRU methods have been employed separately in existing works [23], [27], [56] and were not modified to synchronize with the process. TD²SecIoT process is strong and advanced from others in such a way that it is using NTRU for encryption and ECC for mutual authentication. The aforesaid methods have been modified (at the system setup procedure), which uses previously stored value to generate the key. The existing works [23], [27] do not ensure a replay attack, whereas the proposed work handles it through time stamping. The proposed work ensured security against key recovery attack that is not offered by other researchers [23], [27], [56]. TD²SecIoT offers a dynamic key concept that makes it difficult to recover the keys. The proposed scheme is also secured (through dynamic and data-driven key feature) against chosen-ciphertext attacks, whicht is missing in existing techniques [23], [27], [56].

TABLE IV. Qualitative Comparison of Proposed TD²SecIoT With State of the Art Works

| Qualitative Parameters | TD²SecIoT | [23] | [27] | [56] |
|---|---|---|---|---|
| Confidentiality | Strong | Moderate | Moderate | Moderate |
| Integrity | Strong | Moderate | Moderate | Poor |
| Unforgeability | Strong | Moderate | Moderate | Moderate |
| Non-repudiation | Strong | Moderate | Poor | Moderate |
| Security against Reply Attack (Timestamping) | Yes | No | Yes | No |
| Security against Key-Recovery Attack | Yes | No | No | No |
| Security against Chosen Cipher Text Attacks | Yes | No | No | No |
| Dynamic key generation | Yes | No | No | No |

## VII. Conclusion

This paper proposes a security architecture (TD$^{2}$SecIoT) for ensuring the security of IoT applications. The architecture performs the mutual authentication, encryption, and decryption of the data between motes with less computational cost and higher security levels by integrating public key cryptosystems ECC and NTRU based methods.

The presented system has been evaluated, analyzed, and proved that TD$^{2}$SecIoT gives better performance as per the security matrices. It has been depicted from the simulation that the proposed method demands a smaller key size and takes less time to key generation, authentication, encryption/decryption, as compared with other existing works. TD$^{2}$SecIoT ensures integrity and confidentiality with better security with less computational costs. It also prevents various attacks like a replay attack, a man in the middle attack, a chosen cipher attack, a Quantum attack, and a Lattice-Based attack. It makes TD$^{2}$SecIoT efficient for securing resource-constrained IoT sensors and actuators.

The implementation was on Cooja network simulator and Contiki operating system. The following points have been recommended as future work:

1. Emulation of the code to the actual device.
2. Ensuring the availability issue to prevent against DoS attack.
3. Incorporate IDS/IPS by using an appropriate learning algorithm to secure IIoT.

## References

[1] A. R. Sfar, Z. Chtourou, and Y. Challal, “A systemic and cognitive vision for IoT security: a case study of military live simulation and security challenges,” Smart, Monit. Control. Cities, IEEE, 2017, pp. 17–19.

[2] Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A.R. and Tarkoma, S., 2017, June. Iot sentinel: Automated device-type identification for security enforcement in iot. In 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), pp. 2177-2184

[3] Frustaci, Mario, Pasquale Pace, Gianluca Aloi, and Giancarlo Fortino. “Evaluating critical security issues of the IoT world: Present and future challenges.” IEEE Internet of things journal, vol. 5, no. 4, 2017, pp. 2483-2495.

[4] Zhai, Chuanying, Zhuo Zou, Qiang Chen, Lida Xu, Li-Rong Zheng, and Hannu Tenhunen. “Delay-aware and reliability-aware contention-free MF–TDMA protocol for automated RFID monitoring in industrial IoT.” Journal of Industrial Information Integration, vol. 3, 2016, pp. 8-19.

[5] S. N. Swamy, D. Jadhav, and N. Kulkarni, “Security threats in the application layer in IOT applications. In International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), IEEE, 2017 , pp. 477-480.

[6] Cheng, Jiangfeng, Weihai Chen, Fei Tao, and Chun-Liang Lin. “Industrial IoT in 5G environment towards smart manufacturing.” Journal of Industrial Information Integration, vol. 10, 2018, pp. 10-19.

[7] S. Vashi, J. Ram, J. Modi, S. Verma and C. Prakash, “Internet of Things (IoT): A vision, architectural elements, and security issues,” 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, 2017, pp. 492-496.

[8] Urquhart, Lachlan, and Derek McAuley. “Avoiding the internet of insecure industrial things.” Computer law & security review, vol. 34, n. 3, 2018, pp. 450-466.

[9] L. Urquhart and D. McAuley, “Avoiding the Internet of Insecure Industrial Things,” Comput. Law Secur. Rev. Elsevier, 2017, pp. 1–17.

[10] Yousefi, A. and Jameii, S.M., 2017, May. Improving the security of internet of things using encryption algorithms. In 2017 IEEE International Conference on IoT and Application (ICIOT), pp. 1-5.

[11] R. Marwat, “Internet of Things.” [Online]. Available: https://www.slideshare.net/RehmatMarwat/introduction-to-internet-of-things-45172425. [Accessed: 07-Aug-2019].

[12] R. Elhabob, Y. Zhao, I. Sella, and H. Xiong, “An efficient certificateless public key cryptography with authorized equality test in IIoT”. Journal of Ambient Intelligence and Humanized Computing, Springer, vol. 11, n. 3, 2020, pp. 1065-1083.

[13] Boyes, H., Hallaq, B., Cunningham, J., & Watson, T. “The industrial internet of things (IIoT): An analysis framework”, Computers in industry, Elsevier, 2018, Vol. 101, pp. 1-12.

[14] A. A. Diro and N. Chilamkurti, “Distributed attack detection scheme using deep learning approach for Internet of Things,” Futur. Gener. Comput. Syst. Elsevier, 2017, pp. 761-768.

[15] S. Vashi, J. Ram and J. Modi, “A Vision, Architectural Elements, and Security Issues” in International conference on I-SMAC, IEEE,2017, pp. 492-496.

[16] S. B. S.- Smieee, “Cryptosystems used in IoT- Current Status and Challenges,” ICCIT, IEEE, 2017, pp. 58–62.

[17] D. Rivera, A. Garc, L. Mart, B. Alarcos, and G. Ana, “Secure Communications and Protected Data for a Internet of Things Smart Toy Platform,” J. Internet of Things , IEEE, vol. 4662, 2019, pp. 1–11.

[18] V. C. Ashan, “Implementation of WG Stream Cipher with Involution Function,” Procedia Technol., vol. 24, 2016, pp. 790–795.

[19] L. Ding, C. Jin, J. Guan, and Q. Wang, “Cryptanalysis of Lightweight WG-8 Stream Cipher,” ICCIT, IEEE, vol. 9, n. 4, 2014, pp. 645–652.

[20] T. Suzaki, K. Minematsu, S. Morioka, and E. Kobayashi, “TWINE: A Lightweight Block Cipher,” Lect. Notes Comput. Sci., vol. 7707, 2013, pp. 339–354.

[21] W. Diehl, F. Farahmand, P. Yalla, J. Kaps, K. Gaj, and C. Engineering, “Comparison of Hardware and Software Implementations of Selected Lightweight Block Ciphers,” International Conference on Field Programmable Logic and Applications (FPL), IEEE, 2017, pp. 1-4.

[22] L. D. Singh and K. M. Singh, “Implementation of Text Encryption using Elliptic Curve Cryptography,” Procedia Comput. Sci. Elsevier, vol. 54, n. 1, 2015, pp. 73–82.

[23] T. D. P. Bai, K. M. Raj, and S. A. Rabara, “Elliptic Curve Cryptography based Security Framework for Internet of Things ( IoT ) Enabled Smart Card,” WCCCT, IEEE, 2016, pp. 1–4.

[24] F. Özdemir Sönmez, “A Conceptual Model for a Metric Based Framework for the Monitoring of Information Security Tasks’ Efficiency” Procedia Comput. Sci. Elsevier, vol. 54, no. 1, 2019, pp. 181–188.

[25] A. Bansal, “Providing Security , Integrity and Authentication Using ECC Algorithm in cloud storage,” International Conference on Computer Communication and Informatics (ICCCI), IEEE, 2017, pp. 1-5.

[26] S. R. Singh, “Performance Evaluation of RSA and Elliptic Curve Cryptography,” Trans. Inf. Secur, vol. 65537, IEEE, 2016, pp. 302–306.

[27] S. K. Shankar, A. S. Tomar, and G. K. Tak, “Secure Medical Data Transmission by using ECC with Mutual Authentication in WSNs,” Eco-friendly Comput. Commun. Syst. vol. 70, Elsevier, 2015, pp. 455–461.

[28] P. D. K. and S. Kalra, “Elliptic Curve Cryptography for Real Time,” Futur. Gen. Comp., IEEE, 2016 pp. 1–6.

[29] H. Hasan, T Salah, D Shehada., “Secure Lightweight ECC-Based Protocol for Multi- Agent IoT Systems,” 13th Int. Conf. Wirel. Mob. Comput. IEEE, 2017, pp. 1-8.

[30] M. Ahmad, F. Khan, M. Alam, and M. Usman, “A payload-based mutual authentication scheme for Internet of Things,” Futur. Gener. Comput. Syst. Elsevier, 2017, pp. 1028-1039.

[31] P. Patil, P. Narayankar, D. G. Narayan, and S. M. Meena, “A Comprehensive Evaluation of Cryptographic Algorithms: DES ,DES, 3DES, AES, RSA and Blowfish,” Procedia - Procedia Comput. Sci. Elsevier, vol. 78, 2016, pp. 617–624.

[32] M. Bunder, A. Nitaj, W. Susilo, and J. Tonien, “A generalized attack on RSA type cryptosystems” Theor. Comput. Sci. vol. 704, Elsevier, 2017, pp. 74–81.

[33] D. John and L. Martin, “Binary RSA Encryption Algorithm,”, In International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), IEEE, 2016, pp. 178–181.

[34] A. Mektoubi, H. L. Hassani, H. Belhadaoui, and Rifi, “New approach for securing communication over MQTT protocol A comparaison between RSA and Elliptic Curve,” Third International Conference on Systems of Collaboration, IEEE, 2016, pp. 1-6.

[35] J. Wang, K Han, A Alexandridis, Z Zilic, Y Pang, W Wu., “A novel security scheme for Body Area Networks compatible with smart vehicles,” Comput. Networks, Elsevier, vol. 143, 2018, pp. 74–81.

[36] S. Singh, S Padhye "Cryptanalysis of NTRU with n Public Keys," In Asia Security and Privacy (ISEASP), IEEE, 2017, pp. 1-6.

[37] R. Tso and Y.-S. Jheng, "Security Analysis of a NTRU-based Mutual Authentication Scheme," In 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), IEEE, 2016, pp. 1-3.

[38] R. Chen and D. Peng, "A novel NTRU-based handover authentication scheme for wireless networks," Futur. gen. comp, vol. 7798, IEEE, 2017, pp. 1–4.

[39] Y. Gao, X. Yin, C. Hao, "Sequential Digital Multi-Signature Scheme Based on NTRU," Int. Conf. Comput. Commun. IEEE, 2016, pp. 236–240.

[40] H. R. Yassein and N. M. Al-saidi, "A Comparative Performance Analysis of NTRU and Its Variant Cryptosystems," In International Conf. on Current Research in Computer Science and Information Technology (ICCIT), IEEE, 2017, pp. 115-120.

[41] R. Jha and A. K. Saini, "A Comparative Analysis & Enhancement of NTRU Algorithm for Network Security and Performance Improvement," In International Conference on Communication Systems and Network Technologies, IEEE, 2011, pp. 80–84.

[42] A. Joshi, S. Dhongdi, K. R. Anupama, P. Nahar and Rishabh Sethunathan," Implementation of Protocol Stack for Three-DimensionalWireless Sensor Network",Procedia Comput. Sci. Elsevier, vol. 54, no. 1, 2016, pp. 193–202.

[43] I. Romdhani, "Cooja_Simulator_Manual." [Online]. Available:https://www.researchgate.net/publication/304572240_Cooja_Simulator_Manual. [Accessed: 07-Aug-2019].

[44] X. Shen, Z. Du, and R. Chen, "Research on NTRU algorithm for mobile java security". International Conference on Scalable Computing and Communications; Eighth International Conference on Embedded Computing, pp. 366-369, 2009, IEEE.

[45] M. J. Kaur and P. Maheshwari, "Building Smart Cities Applications using IoT and Cloud-based Architectures," In International Conference on Industrial Informatics and Computer Systems (CIICS), IEEE, 2016, pp. 1-5.

[46] Tmote Sky: Product Description. [Online] http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf. [Accessed: 11-Aug-2019].

[47] P. K. Panda, "A Hybrid Security Algorithm for RSA Cryptosystem," In International Conference on Advanced Computing and Communication Systems (ICACCS), IEEE, 2017, pp. 1-6.

[48] T. K. Goyal and V. Sahula, "Lightweight security algorithm for low power IoT devices," In Int. Conf. Adv. Comput. Commun. Informatics, IEEE, 2016, pp. 1725–1729.

[49] S. Athmani, A. Bilami, and D. E. Boubiche, "EDAK: An Efficient Dynamic Authentication and Key Management Mechanism for heterogeneous WSNs," In Future Generation Computer Systems, Vol. 92, Elsevier,2019, pp. 789-799.

[50] J. Bozzelli, "Temprature sensors," 2014. [Online]. Available: https://www.ptonline.com/columns/how-to-set-barrel-zone-temps. [Accessed: 25-Jun-2020].

[51] J. Sengupta, S. Ruj, and S. D. Bit "A Comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT", Journal of Network and Computer Applications, 149, p. 102481, 2020.

[52] Tiwari, Vivek, and Basant Tiwari. "A Data Driven Multi-Layer Framework of Pervasive Information Computing System for eHealthcare", International Journal of E-Health and Medical Communications (IJEHMC), vol. 10, n. 4, 2019, pp. 66-85.

[53] Choudhary, M., Tiwari, V., & Venkanna, U., "Iris anti-spoofing through score-level fusion of handcrafted and data-driven features", Applied Soft Computing, vol. 91, 106206, 2020 Elsevier. https://doi.org/10.1016/j.asoc.2020.106206,

[54] Yadav, Sonal, Vivek Tiwari, and Basant Tiwari. "Privacy preserving data mining with abridge time using vertical partition decision tree." In Proceedings of the ACM Symposium on Women in Research 2016, pp. 158-164.

[55] Nema, Aditi, Basant Tiwari, and Vivek Tiwari. "Improving accuracy for intrusion detection through layered approach using support vector machine with feature reduction." In Proceedings of the ACM Symposium on Women in Research 2016, pp. 26-31.

[56] Suo, Hui, Jiafu Wan, Caifeng Zou, and Jianqi Liu. "Security in the internet of things: a review." In 2012 IEEE international conference on computer science and electronics engineering, 2012, vol. 3, pp. 648-651.

### Dr. Basant Tiwari

Dr. Basant Tiwari, is currently serving as Assistant Professor in Computer Science department at Hawassa University, Ethiopia. He has rich experience in teaching undergraduate and postgraduate classes. He has many international and national publications to his credit in conferences and Journals and edited Handbooks in Springer and IGI Global. He is a Senior Member of IEEE, Senior Member of ACM, CSI and IACSIT. Prof. Basant Tiwari has organized various National and International conferences and delivered invited talks and also chaired the technical sessions. He is reviewer of various reputed International journals and Books. He did his M. Tech. (CSE) from Rajiv Gandhi Technical University, Bhopal and Ph. D. from School of Electronics, Devi Ahilya University, Indore, India. His current area of research is Pervasive computing specially in Healthcare and IoT with Information and Network Security.

### Dawit Dejene

Dawit Dejene earned his MSc. in computer science from Hawassa University, Ethiopia in 2018, and is currently a lecturer in department of computer science in the institute of technology. His research interests include network security, IoT security & analytics, Intelligent Security, Big Data Analytics and AI.

### Vivek Tiwari

Professor in the Department of Computer Science and Engineering at DSPM IIIT-Naya Raipur, C. G. India. He received the B.Eng. degree (Comp. Sci. & Engg.) from the Rajiv Gandhi Technical University, Bhopal, in 2004, and the M.Tech. degree (Comp. Sci. & Engg.) from SATI, Vidisha (MP) in 2008. He has obtained Ph.D. degree from National Institute of Technology, Bhopal (MANIT), India in 2015. Dr. Tiwari is a recipient of Young Scientist Fellowship (MPYSC_2014_814) for the year 2014-2016 by the MPCST (Madhya Pradesh Council of Science & Technology), Govt. of M.P. His broad research interest areas include Data Mining, Data Warehousing, Pattern Recognition, Machine learning, and Predictive analytics.