Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.054676

ARTICLE

End-To-End Encryption Enabled Lightweight Mutual Authentication Scheme for Resource Constrained IoT Network

Shafi Ullah[1,*], Haidawati Muhammad Nasir[2], Kushsairy Kadir[3,*], Akbar Khan[1], Ahsanullah Memon[4], Shanila Azhar[1], Ilyas Khan[5] and Muhammad Ashraf[4]

[1]Department of Computer Engineering, Balochistan University of Information Technology, Engineering and Management Sciences, Quetta, 87300, Pakistan

[2]Computer Engineering Section Malaysian Institute of Technology, Universiti Kuala Lumpur, Kuala Lumpur, 50250, Malaysia

[3]Electrical Engineering Section, Universiti Kuala Lumpur, British Malaysian Institute, Salengor, 53100, Malaysia

[4]Electrical Engineering Department, Mehran University, Khairpur-Mirs, 76062, Pakistan

[5]Department of Mathematics, College of Science Al-Zulfi, Majmaah University, Al-Majmaah, 11952, Saudi Arabia

*Corresponding Authors: Shafi Ullah. Email: shafi.ullah@buitms.edu.pk; Kushsairy Kadir. Email: kushsairy.kadir@unikl.edu.my

Received: 04 June 2024 Accepted: 14 September 2024 Published: 17 February 2025

ABSTRACT

Machine-to-machine (M2M) communication networks consist of resource-constrained autonomous devices, also known as autonomous Internet of things (IoTs) or machine-type communication devices (MTCDs) which act as a backbone for Industrial IoT, smart cities, and other autonomous systems. Due to the limited computing and memory capacity, these devices cannot maintain strong security if conventional security methods are applied such as heavy encryption. This article proposed a novel lightweight mutual authentication scheme including elliptic curve cryptography (ECC) driven end-to-end encryption through curve25519 such as (i): efficient end-to-end encrypted communication with pre-calculation strategy using curve25519; and (ii): elliptic curve Diffie-Hellman (ECDH) based mutual authentication technique through a novel lightweight hash function. The proposed scheme attempts to efficiently counter all known perception layer security threats. Moreover, the pre-calculated key generation strategy resulted in cost-effective encryption with 192-bit curve security. It showed comparative efficiency in key strength, and curve strength compared with similar authentication schemes in terms of computational and memory cost, communication performance and encryption robustness.

KEYWORDS

Mutual authentication; lightweight end-to-end encryption; elliptic curve cryptography; industrial internet of things; curve25519; machine-to-machine communication

1 Introduction

Resource-constrained MTCDs also known as autonomous IoT devices, are inexpensive, tiny, and autonomous, feasible to operate in environments where no or little human intervention is required. There are approx. 5 billion such devices are connected to wireless sensor networks (WSN), with 48% of the population using the internet and the amount is expected to reach 50 billion by 2025 [1–4].

These devices are heterogeneous, resource-constrained, and communicate in autonomously without human intervention such as monitoring events, synchronous data sharing, communicating remote instructions, autonomous driving assistance, environmental and security surveillance, smart cities and industrial IoTs are the main drivers of autonomous devices.

IoT devices lack the processing power and memory to function remotely and independently. Instead, they rely on external security solutions as they are unable to implement standard security protocols whereas the external security is embedded via software-based protocols. Additionally, these devices operate securely in an environment where every device must be a trusted entity to ensure security with optimal performance. Many experts established such trust by authenticating all devices within the network. However, the authentication schemes lack cost efficiency in performance and fail to acquire robust security against many identified security threats, as presented in Table 1.

This research proposes a lightweight mutual authentication scheme with a pre-calculation strategy to address resilient security and efficient performance. Moreover, the scheme pioneered the following novel functionalities:

1) The lightweight ash function applied one of the fastest curves "Curve25519" with a pre-calculated reference-list strategy where we assigned unique curve points to every American standard code for information interchange (ASCII) character through GNU (collection of freeware application used s an operating system) multiple precision (GMP) library. It performed efficiently in acquiring less code size read only memory and occupying less random-access memory.
2) ECC based 192-bit curve security is achieved through lightweight hash function for an 8-bit CPU based IoT devices.
3) Due to end-to-end encryption, the proposed scheme can work with IoT devices through all data transmission protocols such as local area network (LAN), wide area network (WAN), long term-evaluation (LTE), 3rd generation partnership project (3GPP), long range wide area network (LoRaWAN) types of networks due to its adequate data-block and code-size that includes the one-to-many (one master-many slaves) and many-to-many (many masters-many slaves) type of communication configurations.

The results shown in Section 5 prove that the proposed security scheme can be used in all types of autonomous network environments with minimum cost and optimum security.

Structure of the Article

The structure of the article is as followed:

- Section 2 presents literature review with related works.
- Section 3 presents proposed methodology and important parts of the mechanism including the novel has function, secret key generation and the lightweight encryption technique.
- Section 4 presents threat modeling where the proposed technique is explained in countering ten different attacks.
- Section 5 explains the proposed mechanism performance in terms of communication, storage and computational costs.
- Section 6 explains the limitations and future directions of the presented technique followed by conclusion.

2 Related Studies on Authentication Techniques

This section offers a variety of studies on local, group-based, and factor-based authentication. These studies aim to counter threats that have been identified, as indicated in Table 1, and to achieve efficient computational and communication performance in addition to basic security features like data integrity, privacy, and authentication.

The group-based authentication constructed key-agreeing protocols target LTE/LTE-A based 3GPP networks. These protocols are presented as authentication key agreeing (AKA) to improve security performance. A novel authentication technique was proposed by Lai et al. [1]. The protocol first used a fully authenticated home subscriber server (HSS) with the IoT device to authenticate other devices. However, it lacked resilience against several security attacks. To resolve the issue, Cao et al. [2] presented group-based access authentication (GBAAM), a group signature protocol in which each group was assigned a specific signature. It generated high computation overheads due to asymmetric cryptosystems including severe privacy preservation issues. Fu et al. [3] introduced a privacy-enforced protocol that generated ECC-based pseudo-identity. It successfully encountered basic security threats but produced high network overheads due to asymmetric key cryptosystem. Lai et al. [4] and Li et al. [5] proposed a similar sort of robust and lightweight technique group-based lightweight authentication scheme for resource-constrained machine-to-machine (GLARM-AKA). It was devised for resource-constrained IoT devices and produced lesser network overheads with unlink-ability but failed to incorporate current devices leaving the system (leakage of data) which endangered nodes against node capture attacks [6,7] proposed security enhanced group-based (SEGB) and dynamic group-based efficient and secure authentication (DGBES), respectively; a public key based group authentication with symmetric cryptosystem. These protocols depended on name server protocol (NSP) which is responsible for main computational tasks including key generation and authentication. Lin et al. [8] recommended offloading of computational tasks to save energy and improve performance for local authentication. The scheme enhanced resource management for resource constrained devices as computation was offloaded to the third party. However, the noisy signals resulted in loss of data and the mechanism did not support the regeneration of lost bytes. Ayub et al. [9] presented robust authentication to maximize security that do not target resource constrained devices. Choi et al. [10] endorse a protocol that incorporated unlink-ability but lacked in preserving device privacy and demonstrated vulnerability against identity theft attacks. Li et al. [11] enhanced unlink-ability problems in enduring policy in LTE-A, included privacy preservation and encountered impersonation attacks.

Wang et al. [12] announced a hybrid authentication through fusing local and remote access control features with lightweight cryptography. The authors aimed to compensate resource constrained device's operations by reducing data privacy related processes as the scheme relied on certificate-based authentication. Qiu et al. [13] proposed an AKA scheme intended for machine-to-machine (M2M) in ipv6-low power wireless personal area network (6LoWPAN) to overcome the insufficiencies referred to authentication and key agreeing encrypted system (AKAES) which is a blend of encryption and shared keys, applied for secure authentication for resource constrained IoT devices and offered tight security support to both static and portable devices in 6LoWPAN systems. Chen et al. [14] proposed authentication via identity-based cryptography without key escrow (AIBCwKE) model of authentication using identity-based cryptography (IBC) in which devices were assigned encrypted identities through ECC with third party key agreeing mechanisms. However, the model was tested on a more powerful machine compared to the small IoT devices.

Jiang et al. [15] integrated a two-factor based authentication in which a user would log in, authenticate and then share data. The authentication greatly suffered from user anonymity. Saqib et al. [16]

proposed robust and similar three-factor authentication, an extension to [17,18] work that aimed to achieve user anonymity in smart home environments (a third factor) as pre-shared keys. He et al. [19] compelled IoT devices for complex computation to achieve privacy. Works in [1,19], achieved user privacy in contradiction to the service provider's high energy usage. Wu et al. in [20], demonstrated image encryption through a rule selector in deoxyribonucleic acid (DNA) by cracking semantical information embedded in DNA sequences. The encryption could be used to map secret keys in authentication. A summary of the presented literature is shown in Table 1.

Table 1: Summary of the related work in the presented literature

Article	Method	Security features	Limitations	Achievements	Future aspects	Vulnerability
[5]	Robust and energy-efficient authentication protocol for industrial internet of things	• Authentication • Integrity	Higher complexity in protocol design	Strong security and energy efficiency for industrial IoTs	Simplify protocol design, enhance usability	Potential vulnerabilities due to higher complexity in protocol design affecting efficiency and usability.
[6]	Dynamic group-based efficient and secure authentication and key agreement protocol for MTC in LTE/LTE-A networks	• Authentication • Integrity	Potential for increased complexity and overhead	Dynamic and efficient authentication for MTC in LTE/LTE-A networks	Reduce complexity and overhead, enhance scalability	Vulnerable to complexity issues impacting efficiency and scalability.
[7]	Security-enhanced group-based AKA protocol for M2M communication in IoT-enabled LTE/LTE-A network	• Authentication • Integrity	Require more computational resources	Enhanced security for M2M communication in IoT-enabled LTE/LTE-A networks	Optimize computational resource usage, improve efficiency	Vulnerable to resource constraints affecting computational performance.
[9]	Three-factor anonymous authentication scheme for wireless sensor networks in internet of things environments	• Authentication • Integrity	Potentially higher complexity and overhead	Enhanced security and anonymity for WSNs in IoT environments	Simplify implementation, reduce overhead	Vulnerable to complexity issues affecting overhead and usability.
[10]	Lightweight authentication protocol for e-health clouds in IoT based applications through 5G technology	• Authentication • Integrity	Not scalable to larger networks	Lightweight and efficient authentication for e-health clouds in IoT applications	Enhance scalability, adapt to diverse healthcare scenarios	Vulnerable to scalability issues in larger network deployments.

(Continued)

Table 1 (continued)

Article	Method	Security features	Limitations	Achievements	Future aspects	Vulnerability
[19]	Medical image encryption by content-aware DNA computing for secure healthcare	• Confidentiality • Integrity	Complex implementation and high computational requirements	Enhanced security for medical image encryption in healthcare	Simplify implementation, reduce computational requirements	Vulnerable to complexity in implementation and high computational requirements.
[21]	Robust and energy-efficient authentication protocol for industrial internet of things	• Authentication, • Integrity	Higher complexity in protocol design	Robust security and energy efficiency for industrial IoT	Simplify protocol design	–

3 Proposed Robust Mutual Authentication Scheme

This section describes the essential functions of the proposed scheme. The scheme aimed to enable resource-constrained IoT devices to achieve maximum affordable security with efficient performance in IoT networks. It further attempts to address the basic security features in the perception layer of IoT devices, as discussed in [22]. The process flow between two IoT devices (sender and receiver) is explained in the following:

To establish authentication, a device must authenticate the matching device before sending or receiving actual data to or from that device. To mutually authenticate, the sender and recipient devices will both carry out an Elliptic curve Diffie-Hellman (ECDH)-based authentication key agreement procedure. Once the mutual authentication is established, the receiving device must also ensure that the received data is not mutated during the transmission.

Similarly, data (during the transmission) must be kept secure from main in the middle (MiTM) and eavesdroppers. To achieve this, end-to-end encryption is applied [23] encryption scheme to address critical security issues such attacks by ensuring that both IoT devices and servers can authenticate each other securely before exchanging data. The protocol is optimized for efficiency, employing lightweight cryptographic techniques that reduce computational and communication overhead, which is important for the limited resources of IoT devices. Therefore, we adopted the fastest curve version "Curve25519" (discussed in Section 3.2) from ECC in the proposed lightweight hash function (discussed in Section 3.3) to ensure data integrity. The end-to-end encrypted communication (as shown in OK. 1) is conducted between the two IoT devices where the encryption is applied during the key exchange processes. The keys applied during key exchange processes are an embedded network key (N_k) and a public key (Pb_k). Reference [24] designed IoT devices that leverage the constrained application protocol (CoAP) to address security and efficiency challenges specific to IoT environments by providing a lightweight, yet robust, method for mutual authentication between devices.

3.1 Environmental Assumptions

1. IoT devices are resource constrained and autonomous and operate in a non-lossy network.
2. Devices that work in a single network, share identical network ID or a network key (N_k).
3. We assume that the network has adopted a local authentication environment.

3.2 Elliptic Curve End-to-End Encryption on Curve25519

The Curve25519 is a Montgomery curve with equation $y\text{^}2 = x\text{^}3 + [[486662x]]\text{^}2 + xmod(p)$ [25] over Fp (prime field). The $mod(p)$ is specified by $3 < p \leq 2^{255} - 19$, with initial basepoint of $x = 9$. The curve also utilizes flattened x-axis points to let the use of Montgomery ladder on X and Z coordinates. It is 255-bit curve with preceding standard curves of NIST P-256 and NIST-K233 of 128-bit, that calculates up to 256-bit security with fastest and safest ECC curves, presented by [26]. It is intended for ECDH based key agreement protocols to devise strong encrypted keys. In addition, it exhibits robust security [27] and consumes comparatively less memory in terms of key size. Fig. 1 illustrates end-to-end encryption between two IoT devices (M_1 and M_2) that share a message. The M_1 encrypts the message through Curve25519 based encryption before the ECDH key exchange process by devising a secret key S_k. Additionally, Fig. 2 shows elliptic curve-based encryption performance on similar IoT devices through calculating ECC processes costs such as point addition (PA), point doubling (PD), Scaler multiplication and key generation processes. The proposed encryption process uses the least encryption time due to pre-calculated curve point strategy through computational offloading as keys are generated before and only referenced from memory during encryption process.

Through $Pb\_k \oplus N\_k$ (point addition operation of Pb_k and N_k) that results in a large prime positive integer which is considered as a generator curve number G_LRPN for that message. After successful ECDH based key exchange, the secret key from G_LRPN is considered as the new generator curve point G_cp for that session whereas, G_cp will be renewed for next message.

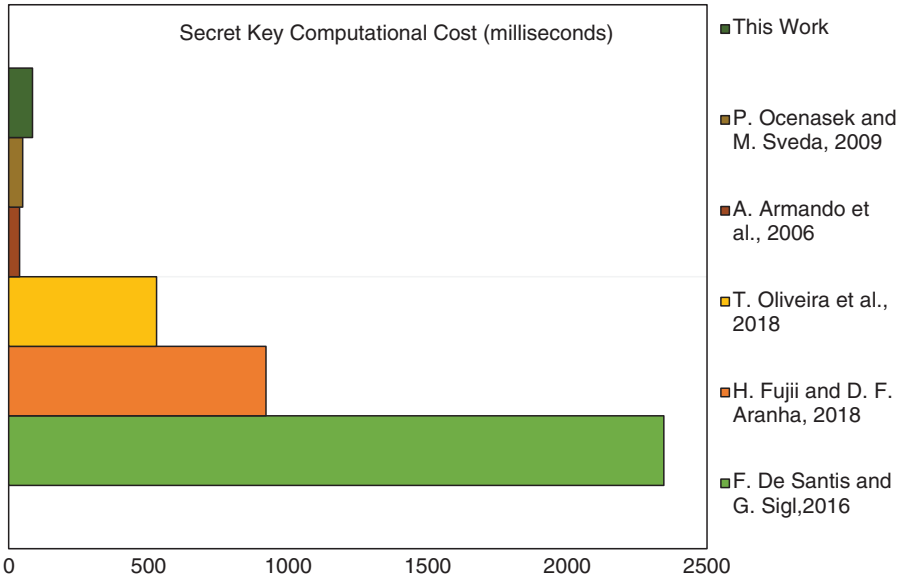


Figure 1: Elliptic curve based comparative secret key performance on similar IoT devices, References: De Santis et al., 2016 [28], Fujii et al., 2018 [29], Oliveira et al., 2018 [30], Armando et al., 2006 [31], Ocenasek et al., 2009 [32]

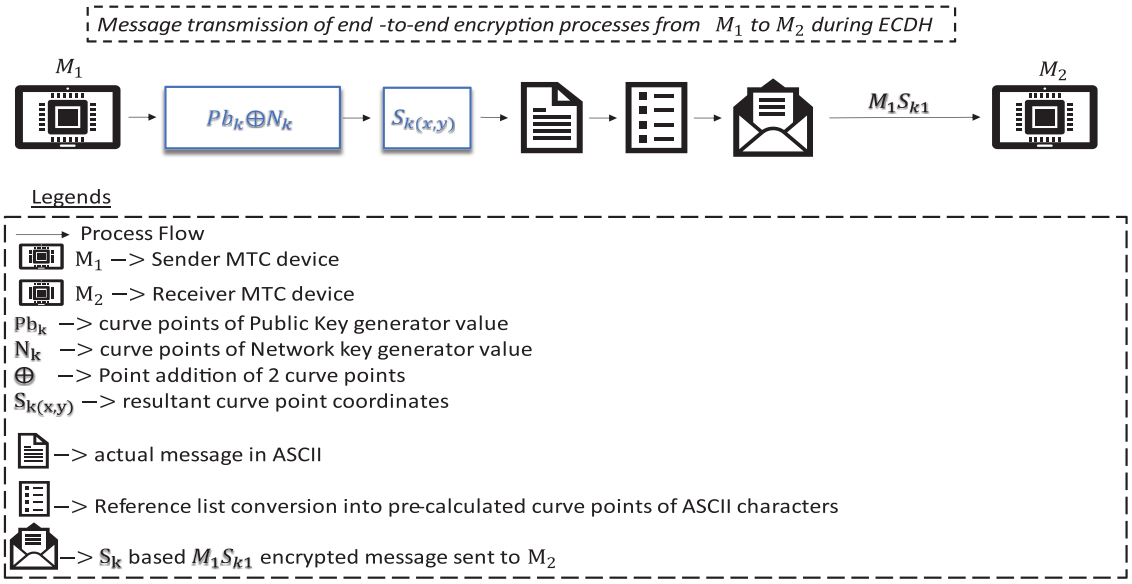


Figure 2: End-to-end encryption process of the proposed scheme

3.3 Secret Key (SK) Generation

Secret key (SK) has a crucial role in the hash function that can be generated from two parts:

1) SK can be generated from ECDH key exchanging process.
2) SK can be formed by an integer through random number generator function (RNGF) with a seed input of a built-in timer in AVR in microseconds.

Table 2 shows secret key (SK) generated by ECDH key exchange procedures where the resulting keys are considered as generator values (G). The proposed technique's execution cost is calculated using scalar multiplication and encryption process. In Table 2, encryption time (E.T) increases with large mod(p) values. Scalar multiplication calculates large curve points in ECC efficiently by converting the G value into bits. When the bit is 1, P.A is executed and P.D is executed if the bit is 0.

3.4 Curve Operation Cost

Because elliptic curve key generation requires a lot of work, it determines the cost of computing performance. Curve operations with a maximum size of 255 bits are performed on huge random prime numbers. Fixed-point multiplication, scaler multiplication, point addition, point doubling, multiplicative inverse, and verification are the several operations that make up curves. The two primary curve operations are point-addition and point-doubling, with other operations being their subsets. Fig. 1 compares the costs of our method with methods based on curve operations in Table 2 that used CPUs with higher processing power. The authors in [33], aimed to protect IoT systems from various security threats such as unauthorized access, data breaches, and impersonation attacks by enabling secure verification of identities between IoT devices and servers. The proposed solution employs efficient cryptographic techniques tailored for the resource constraints of IoT devices, ensuring both robust security and minimal computational overhead with an emphasis on side-channel attack defense. In [28], authors investigated the application of the X25519 elliptic curve Diffie-Hellman key exchange

on ARM Cortex-M4 processors. The difficulty of securing cryptographic operations on devices with limited resources, such as the ARM Cortex-M4, which is frequently utilized in embedded systems and IoT applications, is discussed in the paper. In [29], a study aimed at enhancing the application of the Curve25519 elliptic curve on ARM microcontrollers was presented which are popular in embedded systems because of their efficiency and low power consumption. The authors suggest methods for improving cryptographic operations' performance on these devices with limited resources, guaranteeing the effective execution of safe key exchange protocols. Authors in [30] explored pre-computation strategies, which can greatly improve the performance of cryptographic operations by lowering the computational load during execution. They also discussed methods for optimizing the computation of the Montgomery ladder, a technique widely used in elliptic curve cryptography for performing scalar multiplication securely and efficiently. Fig. 1 represents encryption time through a secret key based on the research presented in this section.

Table 2: Secret key generation performance

mod	G	Secret key		Secret key size	P.A	P.D	E.T (ms)
		x	y				
30-bit	484924606752301	826625966	80435164	30-bit	25	48	1087
	2938674132359780000	136996696	68348709	28-bit	31	62	1406
	160949512909771	497086896	418609960	29-bit	24	48	1062
43-bit	484924606752301	198854107681	1226705872672	38-bit	25	48	1891
	2938674132359780000	1323613420771	2788657789809	42-bit	31	62	2349
	160949512909771	2741618876124	974370806151	42-bit	24	48	1811
62-bit	484924606752301	775724248908631391	3753518844464014710	62-bit	25	48	2623
	2938674132359780000	685910647662771392	3518772554951183129	61-bit	31	62	2907
	160949512909771	1295578424185598538	933837696196791576	60-bit	24	48	2523

3.5 Pre-Calculated Ascii Characters' Curve Points on Elliptic Curve

This technique is used to pre-calculate curve points for 46 ASCII characters consisting of characters a to z, 0 to 9, and 10 special characters. Other characters are excluded to pre-calculate minimize pre-calculation memory usage. This strategy requires approximately 20% storage in the program memory of IoT devices (for Arduino Mega 2560). Table 3 shows ASCII characters' dedicated G values and their respective curve points.

Table 3: Pre-calculated ASCII characters' curve points on elliptic curve

S#	ASCII	Generator no. G	Curve points (x, y)
1	a	111499594558235	6943371436, 703285746
2	b	2889216970938592	370473331, 9814017275
3	c	2768383894150671	7591555548, 617465919
4	d	6752820302129893	1658744261, 842278155
5	e	9368763179746217	183110606, 1102271714

(Continued)

Table 3 (continued)

S#	ASCII	Generator no. G	Curve points (x, y)
6	f	4134484689890461	1179809473, 782626044
7	g	4398252307637991	1189560000, 939147020
8	h	701422480221611	920636675, 800025404
9	j	7708354526621577	731536773, 384990209
10	k	4971564241326461	211051887, 664479360
11	l	2858603567224847	1114911791, 95276802
12	m	4106223294644557	3710347429, 177948275
13	n	8523275385733039	207177991, 1001485169
14	o	4418107557274013	4351796402, 461746467
15	p	2305145081172399	3691263203, 616366034
16	q	9458794062588701	4491813440, 200940412
17	r	535381727096971G	548976276, 16443194
18	s	7496981899680591G	619312283, 712899022
19	t	1630862487441159G	4027484439, 4545269340
20	u	262488422772491	182422723, 62605984
21	v	141698295657533	903414998, 482785657
22	w	1389325569854634	6059458268, 8931747116
23	x	20951118215536	3259429426, 7432935628
24	y	7798576044313577	8634865552, 8564002957
25	z	4005284346017878	4989425128, 2748185069
26	0	835827759657194	2691006138, 1523886684
27	1	8925413191395718	825668282, 9733636287
...	...	...	... , ...

These dedicated values are used for encryption by the hash function. Every ASCII character in Table 3 represents a long random prime integer as generator value (G). These generator values are then applied on ECC to find the respective curve points. These curve points are calculated before runtime to reduce runtime computation.

3.6 Lightweight Hash Function (lHf)

Lightweight hash function (*lHf*) in the proposed scheme is Curve25519 based lightweight encryption technique. A reference list contains limited 46 ASCII characters, where each character has specific ECC generator value (G). These values contain a unique x and y coordinates' curve points. *lHf* encrypts a message (string) based on the set of characters ($\sum$_46^$(i = 1)|Chi$) individually, such that string $s = \{(Ch\_1, Ch\_2, \ldots, Ch\_n)\}s$ where each character *Chi* retains a particular large random prime number (*LRPN*) as a generator value (G) on an elliptic curve. This (G) value is further applied on curve operations (point addition, point doubling) that resulted into a unique x, y curve points ($Gcp(x, y)$) for that particular ASCII character such that $Chi = \{(Gcp\_LRPNi)|\forall Gcp\_LRPNi \ \exists \ (xi, yi)\}$. The $Gcp\_((LRPNi))$ is precalculated, with a specific x, y coordinate points on the curve as the pre-calculation strategy saves runtime processing of curve operations such as scalar multiplication, point addition, point multiplication, modulus, and fixed-point arithmetic. *lHf* first finds corresponding

curve points (xi, yi) of every ASCII character *Chi* in a string s and then scatter the points in two steps. The scattering process further mutates the x, y values to decrease chances of discovering the exact points against statistical attacks. The steps are as following:

Step 1:

1) $\alpha = x\_1 \oplus y\_1$, where x_1, y_1 are corresponding curve points of first character Ch_1 in string.
2) $\beta = \{(x\_1 - y\_1) \dashv |x\_1 > y\_1 \vee (y\_1 - x\_1)|y\_1 > x\_1\}$, where x_1 *and* y_1 are points on curve of Ch_1 in a string s.

Step 2:

$$\gamma\_xi = \{(\gamma\_x1 + x\_i) \,|\, (\gamma\_x1 = \alpha + x\_1) \wedge i > 1\}$$

$$\gamma\_yi = \{(\gamma\_y1 + y\_i) \,|\, (\gamma\_y1 = \alpha + y\_1) \wedge i > 1\}$$

$$\delta\_xi = \{(\gamma)\_xi - \delta\_(x1))|\delta\_(x1) = [[(\gamma]]\_x1 - \beta) \wedge i > 1\} \quad (1)$$

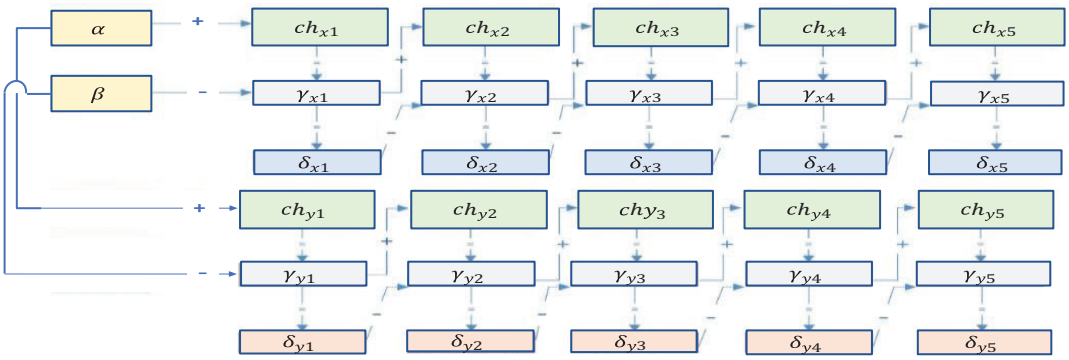
$$\delta\_yi = [[\{(\gamma]]\_i - \delta\_(y1))|\delta\_(y1) = [[(\gamma]]\_y1 - \beta) \wedge i > 1\} \quad (2)$$

δ_xi and δ_yi are mutated xi, yi curve points for *Chi*, respectively.

3.7 *Definition 1: Hash Function*

String s containing ASCII characters *Chi* such that $= \{([[Ch]]\_1, [[Ch]]\_2, \ldots, [[Ch]]\_n) \dashv |\forall\ Chi\ \exists\ (xi, yi)\}$, is encrypted into $Hf(s)$ such that $Hf(s) = \{(\delta\_1, \delta\_2, \ldots, \delta\_n) \dashv |\forall\ \delta\_i\ \exists\ (\delta\_xi, \delta\_yi)\}$, based on Eqs. (1) & (2). The size of δ_xi, δ_yi depends on the size of $mod(p)$ and S_k (secret key).

Fig. 3 illustrates the process of mutating curve points of five ASCII characters. Mutation process is performed as explained in Step 1 and Step 2.



Actual ASCII characters	ch_1		ch_2		ch_3		ch_4		ch_5	
Reference List points →	ch_{x1}	ch_{y1}	ch_{x2}	ch_{y2}	ch_{x3}	ch_{y3}	ch_{x4}	ch_{y4}	ch_{x5}	ch_{y5}
Mutation of points →	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Encrypted curce points	δ_{x1}	δ_{y1}	δ_{x2}	δ_{y2}	δ_{x3}	δ_{y3}	δ_{x4}	δ_{y4}	δ_{x5}	δ_{y5}

Figure 3: Lightweight hash function process of the proposed scheme

3.8 *Authentication Key Agreeing through Novel Lightweight Hash Encryption (NLHE-AKA)*

Mutual authentication process is an authentication frame (auth-frame) based authentication key agreeing (AKA) mechanism. Curve25519 is selected for overall elliptic curve encryption because of

the speedy calculations during scaler multiplication that best suits the resource constrained devices (as discussed in Section 3.2). All devices in the network contain a unique uniform network identification number (N_k—Network Key) and a public key Pb_k. Similarly, all devices generate new private keys (dP_r1 for sender, dP_r2 for receiver) upon newly generated session key. To ensure good security, we suggest renewing session keys after every successful message acknowledgement from sender (s_1_ack) as shown in msg#6 in Fig. 4. However, to accommodate performance efficiency, the suggested session key renewal time is set to 300 s in this proposed work.

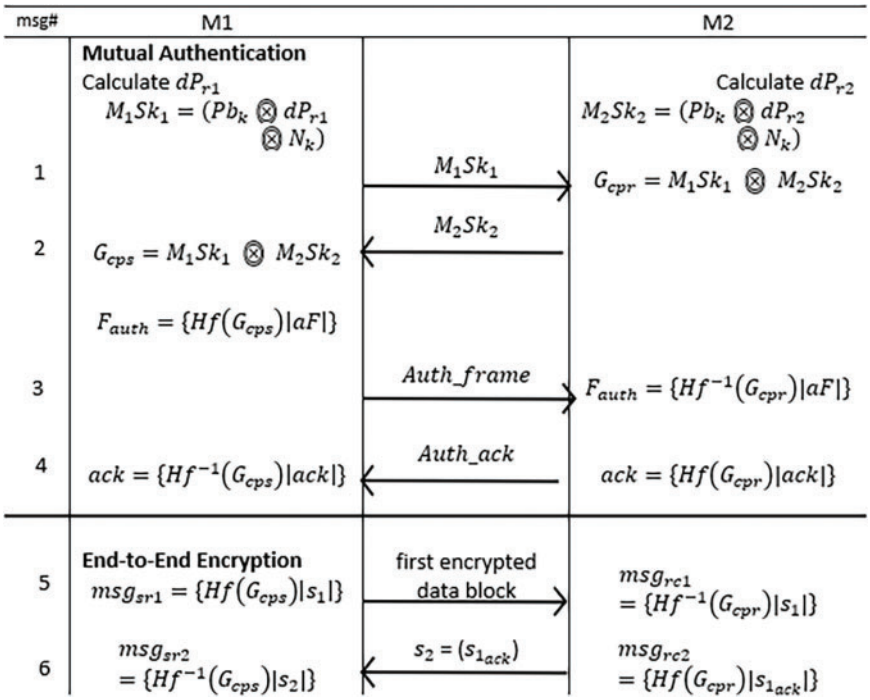


Figure 4: Proposed NLHE-mutual authentication process

Fig. 4 represents data flow of proposed NHLE-AKA based mutual authentication between M_1 & M_2 whereas Table 4 represents notations used in Fig. 1 with size in bits. The mutual authentication process executes irrespective of network related parameters. The proposed scheme is deployable in all sorts of M2M communication networks.

Table 4: Notations used in proposed scheme

Notations	Description	Generator size	Key pair size
M_1Sk_1	Secret key of M_1 device	64	128
M_2Sk_2	Secret key of M_2 device	64	128
Pb_k	Public key	64	128
dP_{r1}	Dynamic private key of device M_1	64	128
dP_{r2}	Dynamic private key of device M_2	64	128
S_k	Session key/Secret key	64	128
N_k	Unique network key	32	64

(Continued)

Table 4 (continued)

Notations	Description	Generator size	Key pair size
G_{cps}	Generator curve points of sender	64	128
G_{cpr}	Generator curve points of receiver	64	128
F_{auth}	Authentication frame function	128 ∗ d	256 ∗ d
$\|aF\|$	Authentication frame string	128 ∗ d	256 ∗ d
$Hf\ (i)$	Hash function for encryption using *ith* generator point	64 ∗ c	128c
$Hf^{-1}\ (i)$	Hash function for decryption using *ith* generator point	64 ∗ c	128c
msg_{sri}	Sender encrypted data block	64 ∗ d	128 ∗ d
msg_{cri}	Receiver encrypted data block	64 ∗ d	128 ∗ d
$\|s_i\|$	String of *ith* data block	–	–
$\oplus$	Scalar multiplication	–	–

Note: C → no. of ASCII characters; d → no. of data block characters.

The following steps can elaborate the auth-frame based mutual authentication between sender M_1 and a receiver M_2 as:

1) All communicating devices contain pre-shared keys i.e., Pb_k, N_k and a dynamic private key dP_rx. Dynamic private keys are considered as generator points (G) that contain specific x, y coordinate points on a curve such that $f\_sm(G) = \{(Gx, Gy) \dashv |G \in P\}$ *where* $P = (Pb\_k, N\_k, dP\_r1$ *for* M_1 & dP_r2 *for* $M\_2)$
2) M_1 and M_2 calculates their resultant keys $M\_1\ Sk\_1$ and $M\_2\ Sk\_2$ which are unique curve points that resulted from a scaler multiplication function $f\_sm(G)$
3) M_1 sends $M\_1\ Sk\_1$ to M_2 where M_2 calculates $G\_cpr = M\_2\ Sk\_2 \oplus M\_1\ Sk\_1$ by adding the two curve points through point addition. (msg#1 in Fig. 4)
4) M_2 sends back $M\_2\ Sk\_2$ so that M_1 can also calculate the same resultant point $G\_cpr = M\_1\ Sk\_1 \oplus M\_2\ Sk\_2$. (in msg#2 in Fig. 4)
5) M_1 sends encrypted pre-defined auth-frame $|aF|$ to M_2 using a generator point G_cps. (msg#3 in Fig. 4)
6) M_2 decrypts auth-frame and sends back encrypted authentication acknowledgement frame $|aF|$ to M_1.
7) Both devices are considered authenticated after receiving and decrypting $|ack|$ frame by M_1 using G_cps. (msg#4 in Fig. 4)
8) M_1 sends msg_sr1 which is first encrypted message after successful authentication ($s\_1 \rightarrow$ *actual data*) (msg#5 in Fig. 4)
9) M_2 decrypts s_1 and sends encrypted message received acknowledgment ($s\_2 \rightarrow s\_1$ *receive acknowledgement*) (msg#6 in Fig. 4)

S_k resets whereas $M\_1, 2$ will renew their respective dynamic private keys dP_rx.

4 Threat Modelling

This section describes security analysis of the proposed scheme in terms of threat modelling. The section also provides a summary of comparative analysis of security features in similar AKA schemes. The presented model needs to be analyzed in terms of security by devising a threat model. A

comparative summary is presented in Table 5 with other similar authentication schemes. Fig. 4 shows the crucial variables and steps followed in the following threat models. Moreover, Features shown in Table 5 can be elaborated through the following scenarios.

Table 5: Comparative analysis of security features with similar techniques

Techniques	F1	F2	F3	F4	F5	F6	F7	F8	F9
[1]	✓	✓	✓	×	✓	×	✓	✓	×
[2]	✓	×	✓	×	✓	✓	✓	✓	✓
[3]	✓	✓	✓	×	✓	✓	✓	✓	✓
[4]	✓	×	✓	×	✓	×	✓	✓	✓
[5]	✓	✓	✓	×	✓	×	✓	✓	×
[6]	✓	✓	✓	×	✓	✓	✓	✓	✓
[7]	✓	✓	✓	×	✓	✓	✓	✓	✓
[8]	✓	×	✓	×	✓	×	✓	✓	×
[9]	✓	✓	✓	×	✓	×	✓	✓	×
NLHE-AKA	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note: ✓-Achieved, ×-Not Achieved, F1-MITM, F2-Device Anonymity, F3-Device Privacy, F4-End-End-Encryption, F5-Impersonator Attack, F6-Link–Unlink ability, F7-Data Confidentiality, F8-Session Key agreement, F9-DOS/replay attack.

4.1 Impersonator Attack on M_1

The attacker is in possession of M_1 such that the knowledge of N_k and Pb_k is known. The attacker then replaces it with a fake device fM_1 that calculates dP_r1 through a random number generator and finds curve points G_cps, then sends $M\_1\, Sk\_1$ over to M_2. The M_2 sends back $M\_2\, Sk\_2$ and calculates G_cpr. The Impersonator still need to know $|aF|$ to confirm authentication. $|aF|$ is impossible for the impersonator to know as it's embedded in the firmware.

4.2 Impersonator Attack on M_2

The attacker is in possession of M_2 and replaces it with a fake device fM_2 knowing all parameters as discussed above. Similarly, fM_2 calculates dP_r2 and receives $M\_1\, Sk\_1$ from M_1. The M_2 then sends back $M\_2\, Sk\_2$ and calculates G_cpr. fM_2 then receives $|aF|$ from M_1 and now M_1 will have wait to receive $|ack|$ from fM_2. Without $|ack|$, the authentication process will not proceed. Because of the unavailability of $|ack|$, M_1 will not confirm authentication and the actual data will not be transmitted to fM_2 hence the session key S_k will expire. Thus, it proves that the proposed scheme is robust against impersonation attacks.

4.3 MiTM, Data Spoofing and Mutation

Attacker is consistently monitoring messages between M_1 and M_2 over the transmission channel. MiTM can over time understand and mimic the transmitted data. Similarly in data spoofing, it is possible to analyze $M\_1\, Sk\_1$ and $M\_2\, Sk\_2$ over long period of monitoring and can then predict the keys and eventually discover $|aF|$ and $|ack|$. However, the proposed scheme is equipped with end-to-end encryption thus it encrypts the authentication frames ($|aF|$ and $|ack|$). The elliptic curve based end-to-end encryption makes it extremely difficult for MiTM to analyze the transmitted data. Furthermore, a 1-bit mutation in the encrypted data block will produce points at infinity. Thus, M_2

will not be to calculate points at infinity and the whole data block will result in null values in case of a single bit mutation.

4.4 Device Anonymity and Privacy

The attacker tries to extract data from certain devices by successfully possessing an IoT device. Since the communication pattern of all devices (as shown in Fig. 1) is extremely precise, thus knowing only the pattern, will not allow the impersonator to extract the data or status of other devices despite providing full authorization over M_1 or M_2. The scheme is designed such that M_1 will only know M_2 during single session. Once the session expires, M_1 and M_2 will have to mutually re-authenticate each other. Moreover, it is also not possible for the network administrator to decrypt certain transmission data blocks without knowing dynamic private keys $dP\_r1, 2$ as dP_rx updates with every session Thus, there exists strong anonymity and privacy for all the devices in the network.

4.5 DOS/Replay Attack

To apply DOS/Replay attacks, the attacker must first connect the mutated device to the network. Attacker should also know set of parameters $\{Pb\_k, N\_k, mod(p), Gcp\_LRPN\}$ to calculate the exact G_cpx for each dedicated session which is difficult and computationally exhaustive operation. Consequently, either M_1 or M_2 will calculate curve points at infinity thus ECDH process will not proceed. Similar is the case of replay attack, as the message pattern changes in each step as shown in Fig. 4. While the attacker sends identical message patterns consecutively, the corresponding device will calculate curve points at infinity thus the message will be rejected, and session will expire.

4.6 Link-Unlink Ability

The attacker attaches (links) a fake device to extract data and analyze message access code (MAC) pattern. As mentioned previously; to carry out such operation, the attacker must know the exact set of parameters $\{Pb\_k, N\_k, mod(p), Gcp\_LRPN\}$ to be identified as valid device in the network as M_1 or M_2. Secondly, the received messages at malicious device will have to be decrypted based on session key Sk which is updated on every dedicated timeframe. Additionally, it is computationally extensive to decrypted ECC based encrypted data block where each data block is encrypted with different session key (Sk). Similarly, the attacker will not be able to investigate source code when detached (Unlinked) the device because it is hardcoded into permanent memory (Firmware) as it is extremely difficult to extract and decode data from firmware of a device.

A comparative study of achieving various important security features discovered for M2M communication in a network, is shown in Table 5. The study reflects that existing AKA protocols for M2M communication lack in providing robust security and forfeit link-unlink ability during distinguishable sessions. It can also be observed from Table 5 that the proposed scheme contains ECC's Curve25519 based end-to-end encryption that improves communication security. Furthermore, the scheme effectively prevents the known attacks in M2M communication networks.

5 Results and Analysis

This section represents the implementation and simulation results of the proposed scheme. Additionally, to analyze feasibility of the scheme, a comparative study of similar AKA techniques, is also presented in the following subsections. The technique is implemented by first generating ASCII curve points followed by generating secret keys to initial mutual authentication. The secret keys are also responsible for initiating the encryption process. The devices then mutually authenticate each

other followed by encrypting the real message and sending it to a receiver device which then decrypts the message.

5.1 Experimental Setup

The proposed scheme is implemented on AVR Atmega2560, a resource-constrained IoT device. The device contains 8-bit @ 16-Mhz CPU capability with 4 Kbytes internal memory. In addition, the device's power ratings are 5 V input with 20 mA current supply. AKA and end-to-end encryption techniques are implemented directly on the specified hardware. However, the networking performance is recorded in Contiki Cooja simulator by using an identical hardware known as SkyMote. In total, 9 different tests were conducted in dynamic topologies for 60 s where every node sends and receives approx. 60 packets.

5.2 GMP (GNU—Multi Precision Arithmetic Library)

The implementation of the proposed scheme on the specified hardware required arithmetic operations on large prime-signed integers, which AVR ATmega coding platforms do not support directly. Therefore, we adopted the GMP library that possesses an embedded support library (mini-GMP) for the AVR ATmega hardware family. The GMP library is freeware and applied mostly for multiple precision arithmetic on 32-to-64 bit signed integers and high precision floating points through fast and precision-capable algorithms. Its loops are partly coded in assembly language to deliver fast results. Additionally, the library works with GNU-gcc and C++ and C language platforms. We utilized version 6.2.0 equipped with a mini support library consuming minimum code size with limited functionality which is specially designed for resource constraint devices.

5.3 Secure & Performance-Efficient Curve Adoption

Choosing the right curve and $mod(p)$ in elliptic curve cryptography defines the encryption robustness and performance efficiency. The proposed work targets affordable curve performance during ECDH-based AKA protocols that rely on the efficiency of curve operations such as scalar multiplication. In addition, the curve must exhibit completeness, indistinguishability, and resilience against statistical attacks, twisted ladder and side channel attacks. That is why we adopted the Montgomery curve (as discussed in Section 3) based elliptic curve cryptographic technique as it plays a vital role in ensuring optimized performance and complete curve security for encryption. Moreover, we adopted the proven safe mod(p) values to test the proposed mothed from [34], where many curves are statistically tested against certain $mod(p)$ values and the results are also made public for research purposes.

5.4 Contiki Cooja Simulator

Contiki is a Linux-based OS specially designed for resource-constraint IoT devices in WSN. It includes a Cooja simulator which simulates nodes (IoT devices), providing platforms for networked, WSN, and 6LoWPAN types of networks. The platform offers monitoring of several node parameters such as transmission power, low power mode, packet delivery ratio, and routing protocols used in TCP/UDP-IP and sensory data for all nodes. It also provides overall network statistics and code-based node analyses which may contain a mixture of nodes. We used Contiki 2.7v which supports MSP430 and Atmel AVR microcontrollers. Fig. 5 offers an illustration of simulation in Contiki Cooja of the proposed scheme where network topology, historical and average power consumption and sensor map simulated data charts are illustrated.

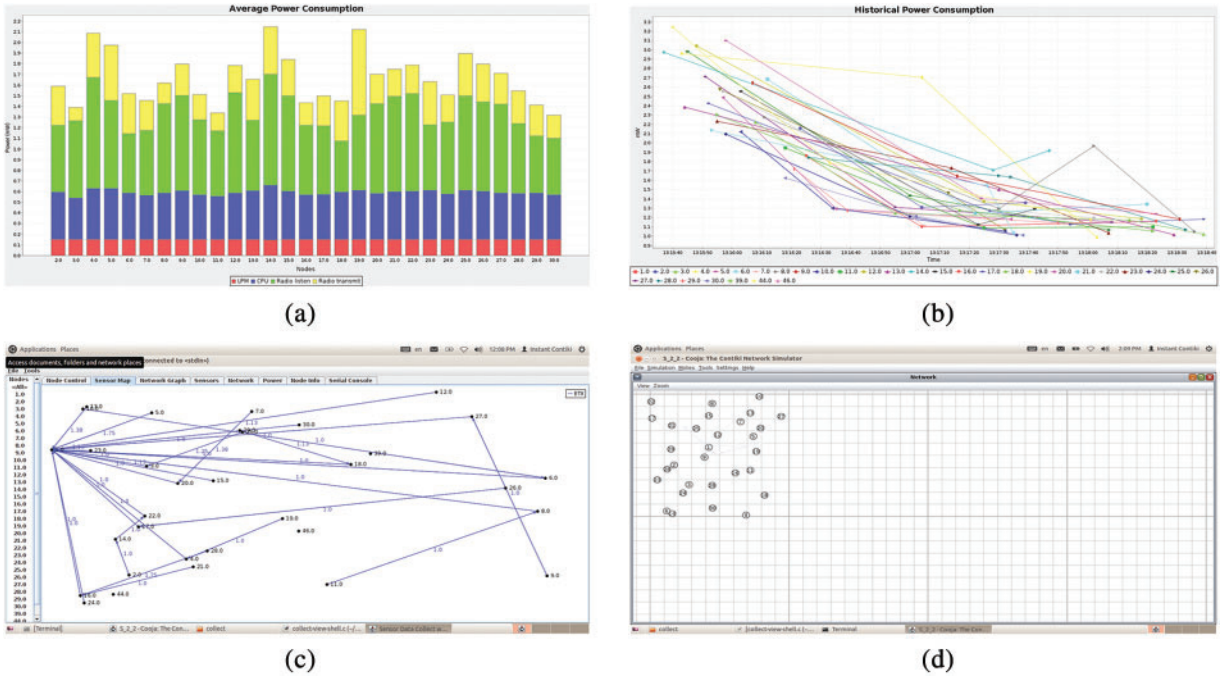


Figure 5: Contiki-Cooja simulation of the proposed scheme (a. average power consumption, b. historical power consumption, c. sensor map of communication with devices, d. dynamic topology of devices)

5.4.1 Simulation Parameters

The Cooja provides predefined simulator library files for different types of communication protocols and resource constraint IoT devices. Collector-View is a similar sort of predefined simulator file that is used to collect information from nodes during simulation. Similarly, Cooja provides different types of nodes such as SkyMote with hardware specifications nearly identical to Atmel AVR microcontroller in a Collector-View simulation environment. The proposed simulation mimics WSN communication protocols containing 30 SkyMotes as illustrated in Fig. 5. The nodes execute program code separately for both M_1 (sender) and M_2 (receiver) with {32, 43, 62}-bit $mod(p)$ and {30, 43, 51}-bit S_k for 60 s.

5.4.2 Simulation Results

This section describes simulation performance of the proposed scheme. The performance is categorized in terms of computational and communication costs, as illustrated in Figs. 6 & 7. The results are based on 30 nodes in WSN environment with random node placements (random topology). Moreover, the simulation is tested on 9 different parameter values (3 different S_k values against 3 different $mod(p)$ values).

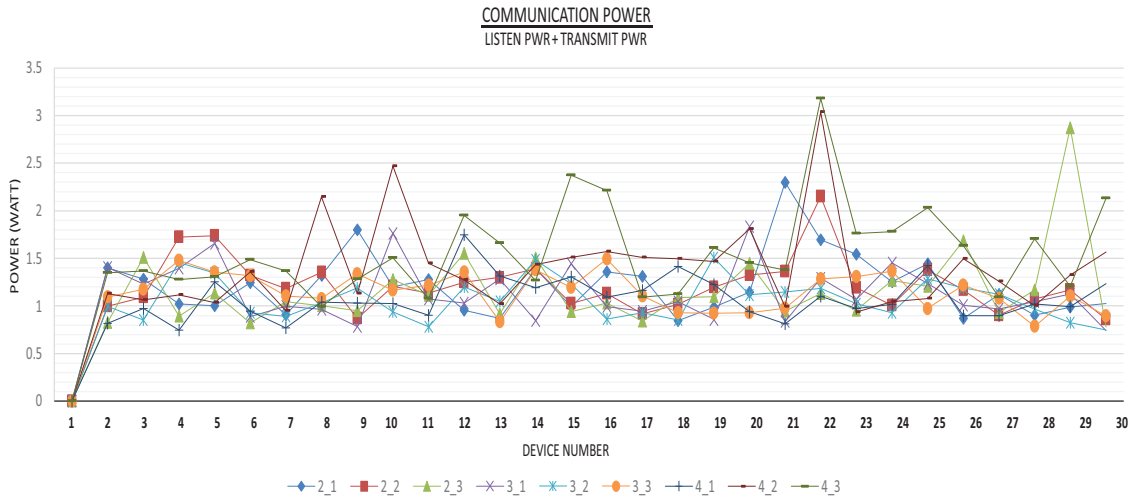


Figure 6: Communication cost performance of the proposed scheme

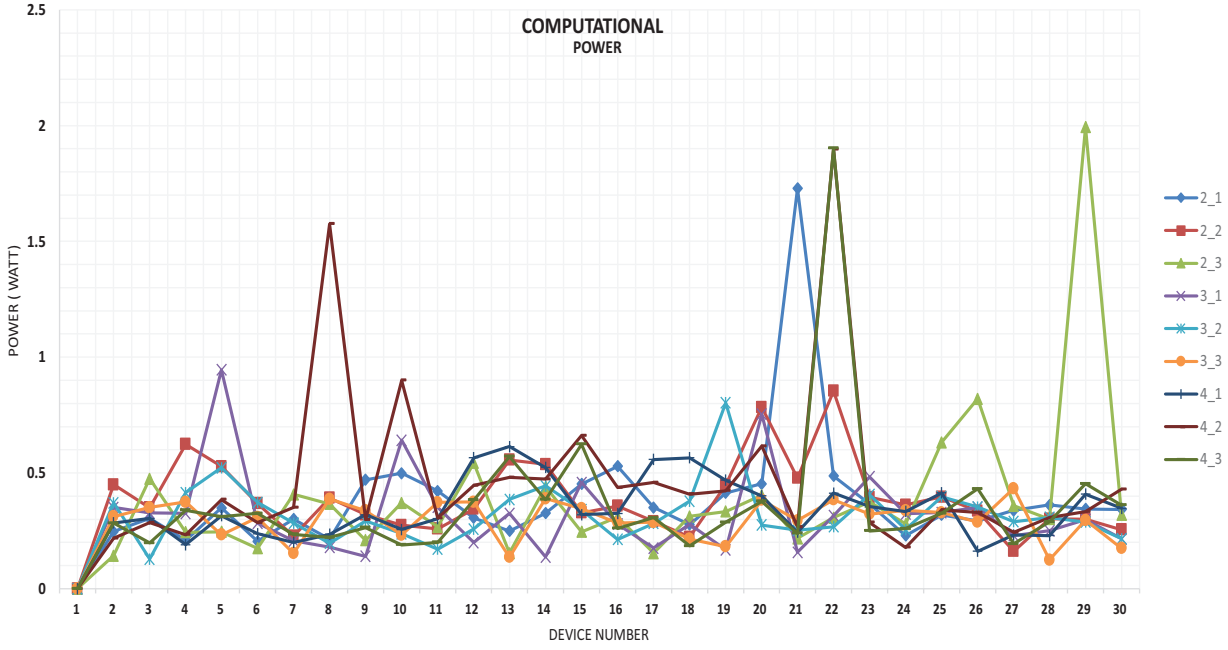


Figure 7: Computational power performance of the proposed scheme

5.4.3 Computational Power Consumption Performance

Development of the proposed scheme is motivated to best suit resource-constrained devices that mostly operate in remote areas under severe conditions and are battery-powered. IoT devices must maintain efficient power consumption to operate for longer hours. Thus, computation power performance depends on the remote operational efficiency of such devices. Power consumptions are further divided into CPU time (where CPU is used) and Low Power Mode time (LPM—where CPU waits, halts or goes to sleep if no task is being performed) to save battery power. Fig. 6 shows the

results of the overall average computational power consumption (CPU + LPM) of all 9 simulation configurations. The results show that there is merely 100 mill-watt difference between ≈192-bit security (62-bit $mod(p)$ with 51-bit S_k, labelled as 4_3 in Fig. 6) and ≈96-bit (32-bit $mod(p)$ with 30-bit S_k, labelled as 2_1 in Fig. 6) security modes.

5.4.4 Communication Power Performance

To communicate seamlessly, the proposed scheme must exhibit decent communication performance. There are two types of modes in communication operations, a device consuming power during data transmitting is considered as transmit power and is considered as listen power during data retrieval.

Hence, transmit and listen power defines overall communication cost of the scheme in a network. Fig. 7 shows average communication cost of 30 nodes with 9 different $mod(p)$ and S_k values in a WSN configuration. It is observed that large S_k values produced large spikes in power consumptions. Thus, the size of S_k directly affects cost of communication in the proposed scheme. However, the overall communication cost is affordable for resource constrained devices as ≈192-bit security mode consumes 1.5-to 2 milli-watt while ≈96-bit security consumes 1 to 1.5 milli-watt communication power.

5.4.5 Comparative Analyzes of Communication Overhead

Comparative analysis in terms of communication overhead of the proposed scheme with similar AKA protocols in IoT communication is discussed in this section. These AKA protocols cover different types of networks such as group-based (used in 3GPP & LTE) and local (for LAN & WAN and LoraWAN) authentication protocols. The proposed scheme serves end-to-end encryption in IoT communication which can be applied on group-based, local, and hybrid networks. Consequently, the communication overhead is analyzed irrespective of network-related parameters. Table 6 represents a summary of communication costs during data transmission, comparing with similar authentication schemes. Every data block ($\mathbf{msg_x}$) represents communication cost in bits. The scheme consists of authentication protocols with similar patterns of message transmission processes for n nodes forming m groups. It can be elaborated from Table 7 that the proposed scheme costs the least overhead considering one successful AKA for 2 nodes (M_1 & M_2) with five data blocks (d = 5); d refers to $|aF|$ and $|ack|$ as shown in Fig. 1 where the messages show mathematical cost of each packet. Similarly, Fig. 8a,b shows comparison of the proposed scheme with existing AKA protocols in IoT communication networks. In (a), The proposed scheme costs the least in communication overhead while it gets slightly higher for large networks/groups because of end-to-end encryption capability. It is also observable that the transmission is efficiently done in small number of groups, however the efficiency decreases as number of groups increases due to end-to-end encryption capability.

Table 6: Comparative analysis of communication overhead

Techniques	msg1	msg2	msg3	msg4	msg5	msg6	Total
[1]	544m	544m	816m	496m	64m	480n	1040 (n − m) + 2464m
[2]	128m + 128n	128m + 128n + 128	128n + 736m	419n	64m	480n	3363n + 1888m + 832

(Continued)

Table 6 (continued)

Techniques	msg1	msg2	msg3	msg4	msg5	msg6	Total
[3]	448n	384n + 64m	384n + 192m	992m	864m	864m	2144n + 2176m
[4]	448n	384n + 64m	384n + 104m	1072m	688m	688m	1968n + 1992m
[5]	U_i, R_P W_i	Bi	{A_i, B_i, C_i, D_i}	$S_{j\ calculation}$	$GWN_{submission}$	$SK_g ? = SK_j$	2688m
[6]	384nm − 1	128nm − 1	320mn − 1 + 168	688	432nm − 1	432nm − 1	1520n + 1480m
[7]	384n	320n + 128m	320n + 168m	256n + 688m	128n + 432m	128n + 432m	1600n + 1912m
[8]	*UE* reg.	$CA \rightarrow UE$	$CA \rightarrow Trustee$	$Trustee \rightarrow CA$	$CA \rightarrow UE$	U_s	4096 bits max
[9]	id_u, $h_{(x)}$	U∗	*X, Y, A, U*	$auth_s$	$auth_u$	U_{new}, V_{new}	2112m
The proposed (NLHE-AKA)	**64n**	**64n**	**(128) 5n**	**(128) 5n**	**(128)** [d]	**(64 ∗ 5)** [d]	**1720m**

Note: n = number of IoT D, d = data block, m = number of groups/networks, msgx = message number.

Table 7: Comparative computational cost of IoT devices

Techniques	Total
[1]	$4T_{hash} + (3T_{hash}) * (n - 1)$
[2]	$4T_{mul} + 2T_{hash} * n$
[3]	$6T_{hash} + 2T_m + (2T_{hash}) * n$
[4]	$4T_{hash} + 2T_{hash} * n$
[5]	$19T_{hash} + (8T_s + 3T_{mul}) * n$
[6]	$4T_{hash} + 2T_{hash} + T_{aes}$
[7]	$4T_{hash} + 2T_{aes} + 2T_{hash} * n$
[8]	$(4T_{ECC\_add} + 2T_{hash}) * n$
[9]	$10T_{hash} + 3T_p * n$
Our	$\mathbf{2T_{ecdh} + 2T_{hash}}$

Note: **hash** → hash function, **mul** → scalar multiplication, **ecdh** → elliptic curve Diffie-Hellman key exchange, **s** → fixed point multiplication, **aes** → AES algorithm, **p** → key pair generation, **m** → modulus.

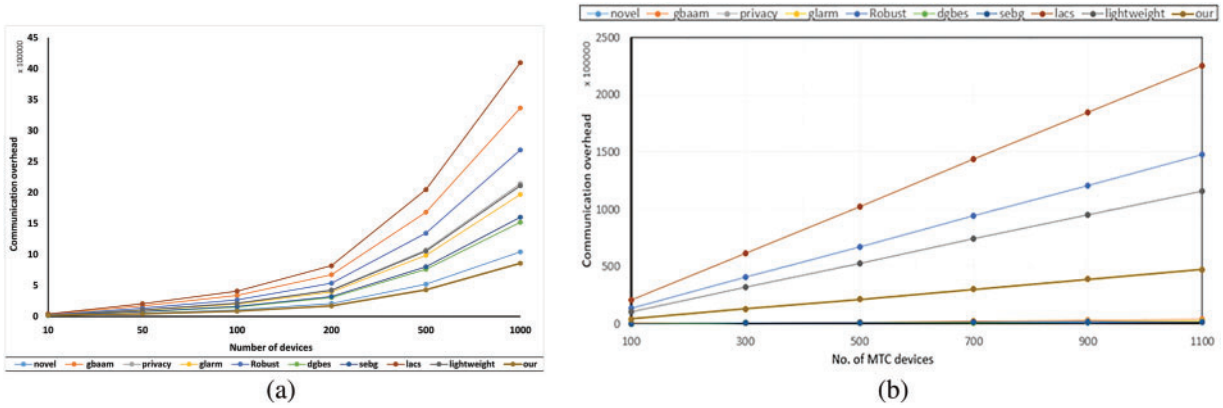


Figure 8: Communication overhead comparison with similar aka protocols (a) m = 1, (b) m = 50

5.4.6 Comparative Analysis of Computational Overhead

Computational evaluation on one IoT device can be done through Lagrange Time component (L_T) [11]. The component assigns time cost values to cryptographic functions. In this section, we applied L_T on the proposed hardware to evaluate time costs. The L_T at hash function is ($T_{hash} = 0.75$), scalar multiplication forming one key pair ($T_{ecc\_add} = 0.5$, $T_p = 0.15$ & $T_s = 0.1$), American encryption system AES at ($T_{aes} = 1$) and ECDH based key exchange at ($T_{ecdh} = 0.5$). The cost of T_{ecdh} is comparatively higher than other ECC operations as key exchange operations can be costly if session key (S_K) is large. Since we used ≈64-bit [[dP]]_rx dynamic key generator values to generate 128-bit S_K pair that is why the computation cost of single IoT device (M_1 or M_2) costs slightly higher comparatively, in resource constrained hardware specifications. Table 7 shows computational comparative analysis costs whereas Fig. 9 illustrates comparative computational overhead analysis of the proposed scheme with similar AKA protocols. It can be observed that the proposed end-to-end encryption capable scheme costs efficient computation with tight security.

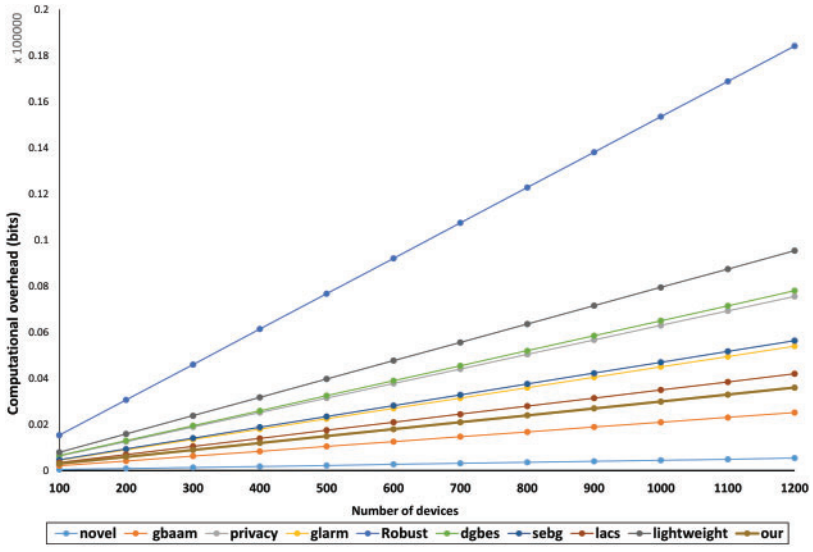


Figure 9: Comparison analysis of computational overhead of proposed scheme

The proposed scheme is only costly compared to GBAAM-and-Novel AKA protocol as they lack acquiring resilience against statistical and DOS/Replay type attacks, respectively.

5.4.7 Comparative Analysis of Storage Cost

Comparative study of storage cost analysis is presented in this section as shown in Table 8. It refers to minimum storage required by an IoT device as pre-requisite parameters to operate on certain protocols. In the proposed scheme, of Pb_k, N_k and ASCII characters' pre-calculated curve point, i.e., $\sum\_46\hat{}(i = 1)Chi\_((x, y))$. Storage cost of the proposed scheme primarily depends on the size of $mod(p)$, large $mod(p)$ value will result in more storage cost. Additionally, Table 8 shows comparative analysis of storage cost. We used ≈64-bit $mod(p)$ value whose storage cost resulted in 896-bits per IoT device. The 896-bits include encrypted $|aF|$ and $|ack|$ which contain five characters each, for M_1 and M_2. Moreover, storage cost is directly affected by $mod(p)$ value. Whereas the size of p is selected based on IoT device's computing and storage capability which is a trade-off between cost-efficiency in performance and security of end-to-end encryption. Extremely large $mod(p)$ values could result in memory overflow and heavy computation while small $mod(p)$ values could risk the security strength of cryptography. Hence, the adopted $mod(p)$ value is selected so that no additional storage is required. Fig. 10 illustrates storage cost's comparative analysis of operational IoT devices in a network for similar AKA schemes where it shows that the proposed scheme requires additional storage because of large precalculated curve points for ASCII characters on 64-bit $mod(p)$ value.

Table 8: Comparative analysis of storage cost

Techniques	Storage cost (Bits)
[1]	816m
[2]	128n + 736m
[3]	992mn
[4]	1072mn
[5]	≅2096mn
[6]	≅688mn
[7]	688mn
[8]	1024nm
[9]	672mn
Our	**896mn**

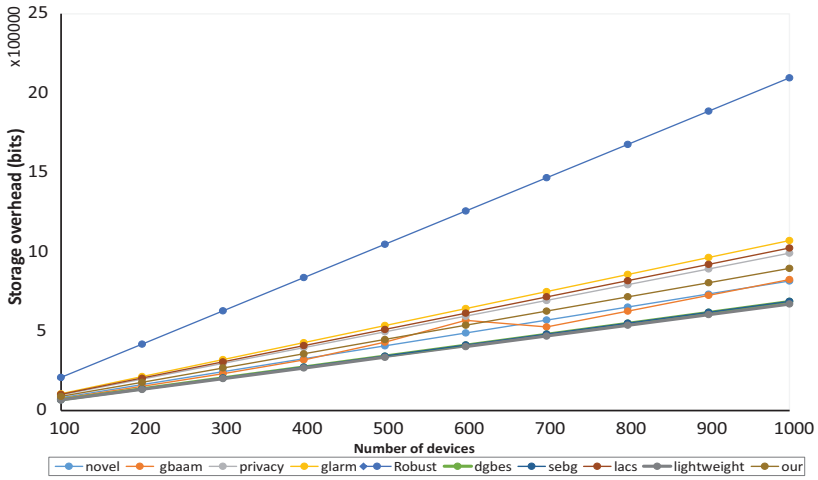
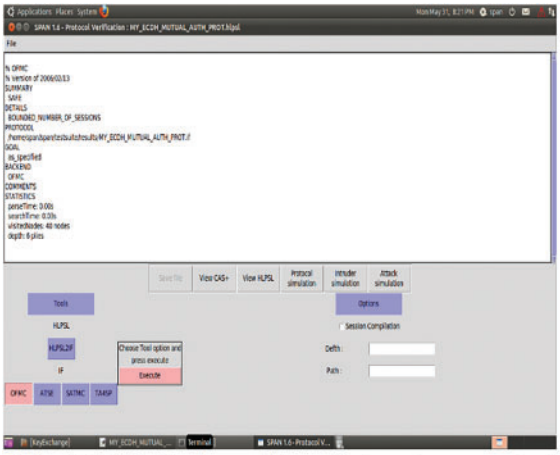


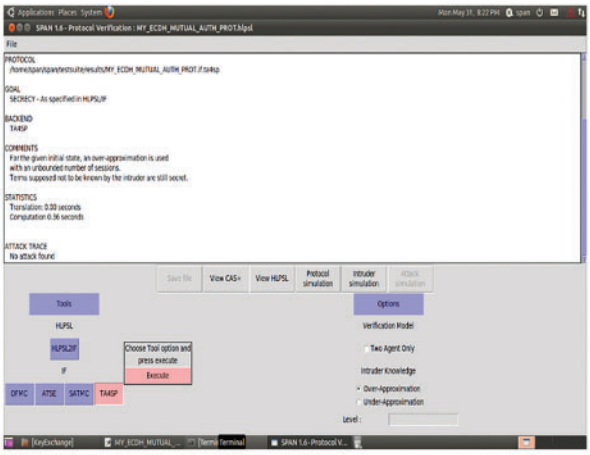
Figure 10: Storage overhead comparison of the proposed scheme with similar techniques in IoT communication

5.5 Formal Security Verification Using AVISPA

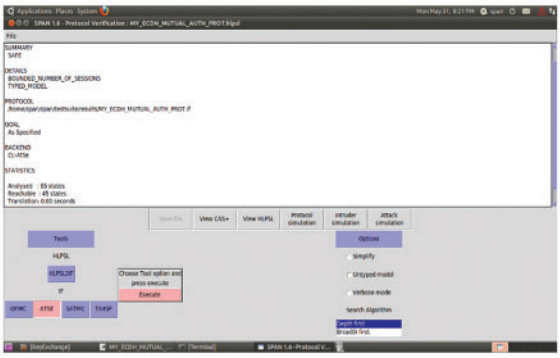
Automated validation of Internet security protocols and applications (AVISPA) is a popular tool [31] for formal security verification of cryptographic protocol in a Linux operating system. It uses the high-level protocol specification language (HLPSL) to model and simulate the presented mutual authentication protocol to verify the security properties. CompTIA advanced security (CAS+) description of the proposed protocol is first converted into Alice (as M_1) and Bob (as M_2) notation which is then applied as input to security protocol animator (SPAN) that simulates the protocol and converts it into HLPSL script. The HLPSL script is then forwarded as input to an intermediate format [32] (IF) translator to analyze it through backend verification models. AVISPA uses the proposed backends, on-the-fly model-checker (OFMC), constraint-logic-based attack searcher (CL-AtSe), satisfiability-based model-checker (SATMC) and tree automata based on automatic approximations for the analysis of security protocols (TA4SP), to verify the goals. Moreover, the proposed backend models execute the protocol via numerous finite iterations until the protocol is deemed safe for the number of sessions or an attack is discovered. We applied three attack models, i.e., OFMC, CLA-AtSe and TASP on the presented security protocol shown in Figs. 11a–c, respectively. It can be observed from the figures that the presented scheme is found safe during all three attack simulations. Additionally, Figs. 12a–c shows the security validation of incoming events (variables before authentication process), in-transit events (variables during the process), and past events (variables after mutual authentication process) of the proposed model.



(a) - OFMC attack model applied on the presented mutual authentication security model

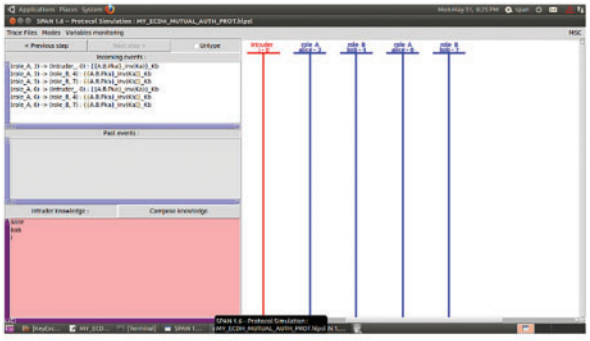


b)- TA4SP attack model applied on presented mutual authentication security model,

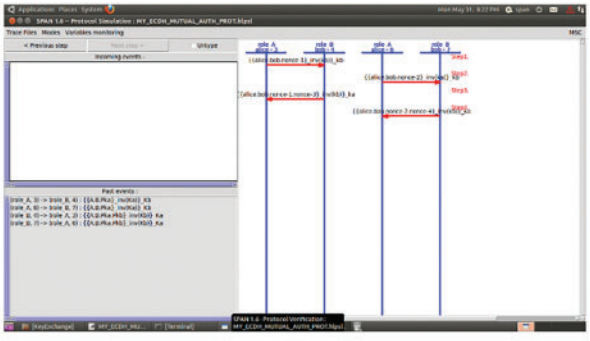


(c)- CL-AtSe attack model on presented mutual authentication security,

Figure 11: Format security verification model attacks on the proposed scheme (a): OFMC, (b): TA4SP, (c): CL-AtSe

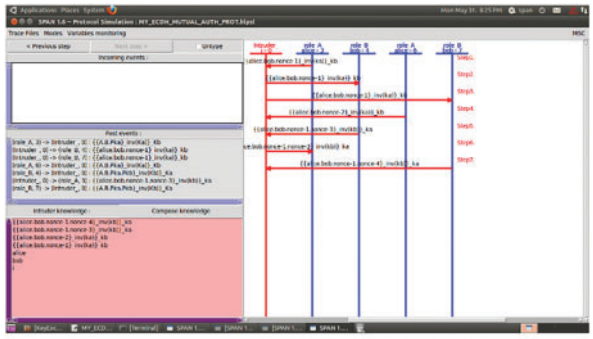


(a)



(b)

Figure 12: (Continued)



(c)

Figure 12: Formal security validation of mutual authentication events in the proposed protocol (a): incoming event, (b): in-transit event, (c): past event

6 Discussion and Future Directions

The proposed scheme successfully achieved lightweight end-to-end encryption which is a very crucial feature for perception layer security that emphasizes the integrity of data. Moreover, the scheme was able to encrypt and decrypt only limited ASCII characters to limit the usage of limited internal memory. Additionally, the scheme can be used in all sorts of communication protocols as it is designed irrespective of the type of communication system such as master-in-slave-out (MSIO), and master-out-slave-in (MOSI) configurations. Moreover, in this article, we acknowledge this trade-off by analyzing various $mod(p)$ sizes affects encryption time. Greater security is achieved by increasing the complexity of cryptographic operations through larger $mod(p)$ sizes at the cost of longer execution times (as shown in Table 2). On the other hand, smaller modulus sizes resulted in less computational overhead, which boosted performance but may jeopardize security. Selecting $mod(p)$ values were experimented extensively with different sizes to manage the trade-off in practice, trying to find a compromise that provides enough security while preserving respectable performance levels. After several experiments, we determined the ideal configuration by considering both large and small values when choosing the $mod(p)$ sizes (discussed in [5]). We intend to provide a more thorough examination of this balancing mechanism in subsequent work, including investigating adaptive approaches that dynamically modify the modulus size in accordance with the demands of the environment or application.

6.1 Limitations

Since the scheme is specifically designed for resource constrained IoT devices that is why the scheme can only afford limited ASCII characters. The communication performance is not tested in large networks containing numerous nodes and networks where large amount of data is shared. However, the proposed scheme can withstand large networks with limited data blocks in a single session. The scheme provides a maximum of 192-bit curve security. The curve-point table values in Table 3 can be increased for stronger security provisions if required.

6.2 Future Directions

The proposed scheme allows many improvements for specific operations:

1. Introducing large $mod(p)$ based pre-calculated curve points, as shown in Table 3 can result in stronger 256–512-bit curve security for more robust security.

2. The encrypted data blocks consist of ASCII characters only (as shown in Table 2). Data blocks in images, audio and video files can enlarge the scope of the scheme.
3. The scheme is limited to local authentication process for limited connectivity. Using the scheme in different authentication environments will enhance the capability of the scheme.
4. The scheme lacks communication support during power and communication failures in which case the whole communication terminates. Thus, adding data availability feature will improve robustness against power and communication failures.
5. The suggested method can be used and contrasted with current protocols because it is made to be energy-efficient for encryption and authentication procedures on devices with low power supplies.
6. Contiki simulator is used to capture networking performance in this article. Concrete measures in lossy IoT networks, like packet loss rates, throughput, and latency, can be further studied.

7 Conclusion

The article proposed an end-to-end encryption-enabled lightweight mutual authentication scheme with a lightweight hash function. The proposed scheme achieved known security features in the perception layer of IoT devices. Additionally, the performance analysis proves that the proposed scheme is comparatively more efficient in computation and communication than other similar techniques. Further, the scheme is especially designed for resource-constraint IoT devices to ensure maximum affordable security and performance in terms of less communication overhead, power consumption, internal memory, and cost of computation. For future work, potential improvements could include enhancing data availability in case of communication disruption, ensuring that devices can still authenticate in local authentication environment. Additionally, incorporating group-based and hybrid authentication methods could provide more robust security solutions. Exploring the use of larger size $mod(p)$ values could further strengthen the cryptographic resilience of the scheme. These enhancements would contribute to a more resilient and versatile authentication framework suitable for a wider range of IoT applications.

Acknowledgement: The authors acknowledge that the research was not funded by any significant funding.

Funding Statement: This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Author Contributions: Shafi Ullah conceptualized the study, designed the translation framework, and contributed to the simulation and coding; Haidawati Muhammad Nasir developed the algorithms and implemented the translation model; Akbar Khan collected and performed the encryption and decryption blocks analysis, ensuring its quality and relevance; Ahsanullah Memon conducted the user studies, gathered feedback, and evaluated the system's performance in an IoT framework; Kushsairy Kadir contributed to the literature review, discussing related work; Shanila Azhar assisted in writing and revising the manuscript, ensuring clarity and coherence whereas Ilyas Khan and Muhammad Ashraf oversaw the project, supervised the team, and provided critical insights throughout the research process. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: No data or materials are available.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

[1] C. Lai, H. Li, X. Li, and J. Cao, "A novel group access authentication and key agreement protocol for machine-type communication," *Trans. Emerg. Telecomm. Technol.*, vol. 26, no. 3, pp. 414–431, 2015. doi: 10.1002/ett.2635.

[2] J. Cao, M. Ma, and H. Li, "GBAAM: Group-based access authentication for MTC in LTE networks," *Secur. Commun. Netw.*, vol. 8, no. 17, pp. 3282–3299, 2015. doi: 10.1002/sec.1252.

[3] A. Fu, J. Song, S. Li, G. Zhang, and Y. Zhang, "A privacy-preserving group authentication protocol for machine-type communication in LTE/LTE-A networks," *Secur. Commun. Netw.*, vol. 9, no. 13, pp. 2002–2014, 2016. doi: 10.1002/sec.1455.

[4] C. Lai, R. Lu, D. Zheng, H. Li, and X. S. Shen, "GLARM: Group-based lightweight authentication scheme for resource-constrained machine to machine communications," *Comput. Netw.*, vol. 99, no. 4, pp. 66–81, 2016. doi: 10.1016/j.comnet.2016.02.007.

[5] X. Li, J. Peng, J. Niu, F. Wu, J. Liao and K. -K. R. Choo, "A robust and energy efficient authentication protocol for industrial internet of things," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1606–1615, 2017. doi: 10.1109/JIOT.2017.2787800.

[6] S. Gupta, B. L. Parne, and N. S. Chaudhari, "DGBES: Dynamic group based efficient and secure authentication and key agreement protocol for MTC in LTE/LTE-A networks," *Wirel. Pers. Commun.*, vol. 98, no. 3, pp. 2867–2899, 2018. doi: 10.1007/s11277-017-5005-6.

[7] B. L. Parne, S. Gupta, and N. S. Chaudhari, "SEGB: Security enhanced group based AKA protocol for M2M communication in an IoT enabled LTE/LTE-A network," *IEEE Access*, vol. 6, pp. 3668–3684, 2018. doi: 10.1109/ACCESS.2017.2788919.

[8] Y. H. Lin, J. J. Huang, C. I. Fan, and W. T. Chen, "Local authentication and access control scheme in M2M communications with computation offloading," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3209–3219, 2018. doi: 10.1109/JIOT.2018.2837163.

[9] M. F. Ayub, K. Mahmood, S. Kumari, and A. K. Sangaiah, "Lightweight authentication protocol for e-health clouds in IoT based applications through 5G technology," *Digit. Commun. Netw.*, vol. 7, no. 2, pp. 235–244, 2020. doi: 10.1016/j.dcan.2020.06.003.

[10] D. Choi, H. K. Choi, and S. Y. Lee, "A group-based security protocol for machine-type communications in LTE-advanced," *Wirel. Netw.*, vol. 21, no. 2, pp. 405–419, 2015. doi: 10.1007/s11276-014-0788-9.

[11] J. Li, M. Wen, and T. Zhang, "Group-based authentication and key agreement with dynamic policy updating for MTC in LTE-A networks," *IEEE Internet Things J.*, vol. 3, no. 3, pp. 408–417, 2016. doi: 10.1109/JIOT.2015.2495321.

[12] H. Wang and Q. Li, "Achieving distributed user access control in sensor networks," *Ad Hoc Netw.*, vol. 10, no. 3, pp. 272–283, 2012. doi: 10.1016/j.adhoc.2011.01.011.

[13] Y. Qiu and M. Ma, "A mutual authentication and key establishment scheme for M2M communication in 6LoWPAN networks," *IEEE Trans. Ind. Inform.*, vol. 12, no. 6, pp. 2074–2085, 2016. doi: 10.1109/TII.2016.2604681.

[14] S. Chen, M. Ma, and Z. Luo, "An authentication scheme with identity-based cryptography for M2M security in cyber-physical systems," *Secur. Commun. Netw.*, vol. 9, no. 10, pp. 1146–1157, 2016. doi: 10.1002/sec.1407.

[15] Q. Jiang, J. Ma, F. Wei, Y. Tian, J. Shen and Y. Yang, "An untraceable temporal-credential-based two-factor authentication scheme using ECC for wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 76, no. 1, pp. 37–48, 2016. doi: 10.1016/j.jnca.2016.10.001.

[16] M. Saqib, B. Jasra, and A. H. Moon, "A lightweight three factor authentication framework for IoT based critical applications," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 9, pp. 6925–6937, 2022. doi: 10.1016/j.jksuci.2021.07.023.

[17] M. Fakroon, M. Alshahrani, F. Gebali, and I. Traore, "Secure remote anonymous user authentication scheme for smart home environment," *Internet Things*, vol. 9, no. 7, 2020, Art. no. 100158. doi: 10.1016/j.iot.2020.100158.
[18] S. Banerjee, V. Odelu, A. K. Das, S. Chattopadhyay, and Y. Park, "An efficient, anonymous and robust authentication scheme for smart home environments," *Sensors*, vol. 20, no. 4, 2020, Art. no. 1215. doi: 10.3390/s20041215.
[19] D. He, S. Chan, and M. Guizani, "Accountable and privacy-enhanced access control in wireless sensor networks," *IEEE Trans. Wirel. Commun.*, vol. 14, no. 1, pp. 389–398, 2015. doi: 10.1109/TWC.2014.2347311.
[20] Y. Wu, L. Zhang, S. Berretti, and S. Wan, "Medical image encryption by content-aware dna computing for secure healthcare," *IEEE Trans. Ind. Inform.*, vol. 19, no. 2, pp. 2089–2098, 2022. doi: 10.1109/TII.2022.3194590.
[21] X. Li, S. Liu, S. Kumari, and C. -M. Chen, "PSAP-WSN: A provably secure authentication protocol for 5G-based wireless sensor networks," *Comput. Model. Eng. Sci.*, vol. 135, no. 1, pp. 711–732, 2023. doi: 10.32604/cmes.2022.022667.
[22] R. Melki, H. N. Noura, and A. Chehab, "Lightweight multi-factor mutual authentication protocol for IoT devices," *Int. J. Inf. Secur.*, vol. 19, no. 6, pp. 679–694, 2020. doi: 10.1007/s10207-019-00484-5.
[23] I. Alshawish and A. Al-Haj, "An efficient mutual authentication scheme for IoT systems," *J. Supercomput.*, vol. 78, no. 14, pp. 16056–16087, 2022. doi: 10.1007/s11227-022-04520-5.
[24] S. G. Oliver and T. Purusothaman, "Lightweight and secure mutual authentication scheme for IoT devices using CoAP protocol," *Comput. Syst. Sci. Eng.*, vol. 41, no. 2, pp. 767–780, 2022.
[25] H. Nguyen, T. Hoang, and L. Tran, "Efficient hardware implementation of elliptic-curve diffie-hellman ephemeral on Curve25519," *Electronics*, vol. 12, no. 21, 2023, Art. no. 4480. doi: 10.3390/electronics12214480.
[26] M. B. Niasar, R. El Khatib, R. Azarderakhsh, and M. Mozaffari-Kermani, "Fast, small, and area-time efficient architectures for key-exchange on Curve25519," in *2020 IEEE 27th Symp. Comput. Arithmet. (ARITH)*, Portland, OR, USA, IEEE, 2020. pp. 72–79.
[27] J. Chia, J. J. Chin, and S. C. Yip, "Evaluating pairing-free identity-based identification using curve25519," in *Advances in Cyber Security. ACeS 2020. Communications in Computer and Information Science*, M. Anbar, N. Abdullah, S. Manickam, Editors, Singapore: Springer, 2021, vol. 1347, pp. 179–193.
[28] F. De Santis and G. Sigl, "Towards side-channel protected X25519 on ARM Cortex-M4 processors," in *Proc. Softw. Perform. Enhan. Encrypt. Decrypt., Benchmark.*, Utrecht, The Netherlands, 2016, pp. 19–21.
[29] H. Fujii and D. F. Aranha, "Efficient Curve25519 implementation for ARM microcontrollers," in *Anais Estendidos do XVIII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, 2018, pp. 57–64.
[30] T. Oliveira, J. López, H. Hışıl, A. Faz-Hernández, and F. Rodríguez-Henríquez, "How to (pre-) compute a ladder: Improving the performance of X25519 and X448," in *Selected Areas Cryptograp.-SAC 2017: 24th Int. Conf.*, Ottawa, ON, Canada, Springer, 2018, pp. 172–191.
[31] A. Armando *et al.*, "The AVISPA tool for the automated validation of internet security protocols and applications," in *Comput. Aided Verif. (CAV 2005)*, 2005, pp. 281–285.
[32] P. Ocenasek and M. Sveda, "AVISPA: Towards practical verification of communication properties," *IFAC Proc.*, vol. 42, no. 1, pp. 153–156, 2009. doi: 10.3182/20090210-3-CZ-4002.00032.
[33] P. K. Panda and S. Chattopadhyay, "A secure mutual authentication protocol for IoT environment," *J. Reliab. Intell. Environ.*, vol. 6, no. 2, pp. 79–94, 2020. doi: 10.1007/s40860-020-00098-y.
[34] T. Lange, "SafeCurves: Choosing safe curves for elliptic-curve cryptography," 2014. Accessed: Sep. 12, 2024. [Online]. Available: https://safecurves.cr.yp.to