electronics

MDPI

Article

A Comparative Study of Post-Quantum Cryptographic Algorithm Implementations for Secure and Efficient Energy Systems Monitoring

Gandeva Bayu Satrya [1,*], Yosafat Marselino Agus [2] and Adel Ben Mnaouer [1,3,*]

[1] Faculty of Engineering, Applied Science and Technology, Canadian University Dubai, Dubai 117781, United Arab Emirates
[2] Telecom Infra Project Community Lab, Bandung 40257, Indonesia; marselino.agus@gcidesign.com
[3] College of Computer and Information Systems, Prince Sultan University, 66833 Rafha Street, Riyadh 11586, Saudi Arabia
* Correspondence: gandevabs@cud.ac.ae (G.B.S.); adel@cud.ac.ae (A.B.M.)

Abstract: The Internet of Things (IoT) has assumed a pivotal role in the advancement of communication technology and in our daily lives. However, an IoT system such as a smart grid with poorly designed topology and weak security protocols might be vulnerable to cybercrimes. Exploits may arise from sensor data interception en route to the intended consumer within an IoT system. The increasing integration of electronic devices interconnected via the internet has galvanized the acceptance of this technology. Nonetheless, as the number of users of this technology surges, there must be an aligned concern to ensure that security measures are diligently enforced within IoT communication systems, such as in smart homes, smart cities, smart factories, smart hospitals, and smart grids. This research addresses security lacunae in the topology and configuration of IoT energy monitoring systems using post-quantum cryptographic techniques. We propose tailored implementations of the Rivest–Shamir–Adleman (RSA), N-th degree Truncated Polynomial Ring Units (NTRU), and a suite of cryptographic primitives based on Module Learning With Rounding (Saber) as post-quantum cryptographic candidate algorithms for IoT devices. These aim to secure publisher–subscriber end-to-end communication in energy system monitoring. Additionally, we offer a comparative analysis of these tailored implementations on low-resource devices, such as the Raspberry Pi, during data transmission using the Message Queuing Telemetry Transport (MQTT) protocol. Results indicate that the customized implementation of NTRU outperforms both SABER and RSA in terms of CPU and memory usage, while Light SABER emerges as the front-runner when considering encryption and decryption delays.

Keywords: Internet of Things; NTRU; post-quantum encryption; RSA; SABER; system monitoring

Citation: Satrya, G.B.; Agus, Y.M.; Mnaouer, A.B. A Comparative Study of Post-Quantum Cryptographic Algorithm Implementations for Secure and Efficient Energy Systems Monitoring. *Electronics* **2023**, *12*, 3824. https://doi.org/10.3390/electronics12183824

Academic Editor: Hung-Yu Chien

Received: 1 August 2023
Revised: 25 August 2023
Accepted: 31 August 2023
Published: 10 September 2023

1. Introduction

The Internet of Things (IoT) is the newest convergence technology, integrating the sensing and transmission capabilities of things to collect useful data [1–4]. Devices with IoT capabilities can be utilized to observe and monitor important people or other various parameters such as physical, electrical, environmental, etc. This information is then used to examine, recognize, and determine different problems related to daily activities e.g., smart home [5], smart city [6], smart factory [7], smart hospital [8], smart grid [9], etc. Electrical power management is one of the problems to be addressed for achieving efficient power operation. Intelligent and IoT-enabled power monitoring devices can help to overcome this problem by giving detailed information about electricity consumption. A supervisory control and data acquisition system (SCADA), which is now called an intelligent electrical power management system [10,11], is used to improve cybersecurity [12,13] and commercial readiness in power plants [14–17].

Smart grids can be embraced as a mechanism of various interconnected systems such as virtual power plants, transmission grids, distribution grids, etc. Each one of them has its own set of devices and communication technologies, intelligent devices, automated control elements, and algorithms. However, the hurried deployment and poor management of smart grid technology could render critical infrastructures vulnerable to cybercrime [18–20]. On the other hand, due to the limited computational power of client processors in an IoT system (i.e., smart grid), the focus of security provisioning shifts from hardware-based to software-based in terms of both effectiveness and efficiency. Several physical layer security (PLS) techniques have also been proposed to safeguard future wireless communications [21–23]. Therefore, it becomes imperative to implement a lightweight encryption algorithm [24–27] that ensures both security and authentication of sensitive information, all while minimizing overhead in terms of computation, memory, time, and power.

The cybersecurity status of a smart grid can be assigned as dangerous when it threatens the confidentiality, integrity, or availability of information and communication technology (ICT) [28–31]. Therefore, the organizations involved must ensure good function of their smart grids, including their market, customers, operation, distribution, etc. Future computation technologies might bring risks to cybersecurity, along with the development of quantum computers, which are believed to be capable of breaking most of the commonly used encryption systems nowadays. This has brought an urgency to develop and examine many candidates for post-quantum cryptographic systems to secure data transmission in energy systems monitoring (ESM) [32,33].

The secure energy monitoring system proposed in this research has been implemented internally in buildings N, O, and P of Telkom University, as depicted in Figure 1. This implementation utilized four Raspberry Pi-4 (RPi-4) units, one AP, and one HPE ProLiant server equipped with an HTTPS service. Three of the RPi-4 units served as registered IoT devices, while the fourth acted as an unregistered IoT device. This setup was designed to simulate an insider attack scenario, where the attacker could be an employee, contractor, or any external party with physical access to the system. Specifically, there is a risk that an adversary could breach the local network when the IoT system operates on its default configuration, even if the message queuing telemetry transport (MQTT) protocol is active. Thus, the IoT system should prohibit access to unregistered devices. A more detailed discussion follows in the next section.

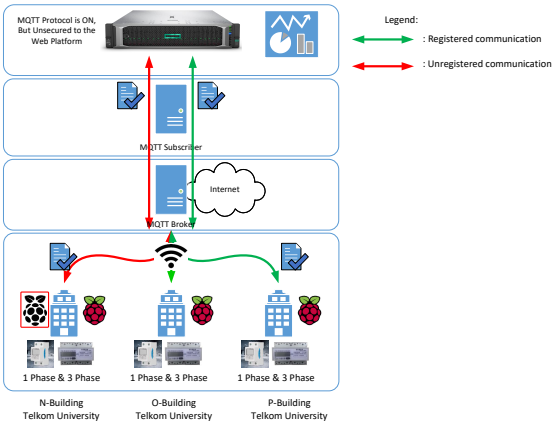

Figure 1. IoT Architecture for ESM.

To complement models, techniques, and results from previous works, this research contributes the following:

- Providing an appropriate review of IoT security outcomes and countermeasures;
- Suggesting a secure post-quantum IoT system using a customized MQTT public network protocol;

- Enforcing an optimized MQTT protocol configuration by adding lightweight cryptography from the physical layer to the application layer;
- Evaluating and benchmarking the proposed post-quantum cryptographic algorithms amongst themselves in an energy monitoring system.

The remaining sections are arranged as follows. Section 2 reviews IoT security issues and their countermeasures. Section 3 describes the proposed scheme to improve IoT network security and compares it with conventional encryption systems. Section 4 evaluates details of the unsecure and secure experiments to validate the proposed post-quantum encryption. Finally, Section 5 highlights the key take-away messages of this research.

2. Related Studies

2.1. Secure IoT Monitoring

Privacy in IoT-enabled smart homes is one of the major concerns of the research community. Ali et al. proposed a novel privacy-preserving method for smart grid-based home area networks (HAN) [34]. The proposed approach uses homomorphic Paillier encryption, the Chinese remainder theorem, and a one-way hash function to aggregate data from diverse household appliances. The proposed approach deploys a sink node between the household appliances and the smart meter, which not only filters false data injection attacks, but also provides for early fault tolerance. The smart grid is a network of different entities and different subnetworks working together. Each entity and subnetwork has its own requirements. Therefore, a universal standardized trusted framework is essential for any communication.

Some smart grids are vulnerable to False Data Injection (FDI) attacks in which an attacker manipulates meter data to disrupt grid operations. Reijsbergen et al. presented a threat model in which there are multiple operators, each with a partial view of the grid, and each can be fully compromised [35]. An effective defense against FDI in this setting is to share data between the operators. They proposed a blockchain solution with an incentive mechanism that rewards operators for uploading data but penalizes them if data are missing or anomalous. They also showed a formal analysis of our incentive mechanism and showed that operators are motivated to share data of as many meters as possible as long as the meter's error probability is sufficiently low. However, the security of blockchain clients is a generic challenge that is not specific to smart grids.

Lightweight encryption algorithms are needed due to resource limitations at the edge nodes of an IoT system. Maitra studied AES both with and without hardware accelerators [36]. The implementation of AES in low-resource embedded platforms was not feasible. The memory, power, execution time, and feasibility of the algorithms were analyzed by using XTEA. The experiment results showed that the energy efficiency and execution time of XTEA on a microcontroller without a hardware accelerator were almost equivalent to that of a device with a crypto engine. The power consumption of XTEA was around 60 times more efficient than the AES on an 8-bit PIC microcontroller.

A collaboration with a third-party service provider demands trust from both the owner and user of sensor data. The fees for their services also must be paid. Manzoor et al. suggested a blockchain-based proxy re-encryption scheme to address those scalability and trust issues, as well as to provide an automatic payment [37]. The scheme consisted of seven polynomial-time algorithms: `Setup`, `CertifiedUserKeyGen`, `Encrypt`, `ReKeyGen`, `ReEncrypt`, `Decrypt1`, and `Decrypt2`. The IoT data were stored in a distributed cloud after the encryption process. The system shared the collected IoT data by setting up runtime dynamic smart contracts between the sensor and the data user without trusted third-party involvement.

Another approach by Tanveer et al. proffered a secure and anonymous authenticated key exchange (AKE) scheme for smart grids (SG), called SAAS-SG, for establishing a secure communication channel between smart meter (SM) and service provider (SPR) [38]. SAAS-SG utilizes the hash algorithm, Esch256, and authenticated encryption algorithm AEGIS to perform the AKE process. In addition, SAAS-SG ensures the integrity and

confidentiality of AKE messages while preserving the anonymity of SMs and SPR because, through informal analysis, SAAS-SG is defended against different covert and fatal security vulnerabilities e.g., denial-of-service (DoS), replay, man-in-the-middle (MINM), and service providers (SPRs)/smart meters (SMs) impersonation attacks. Moreover, SAAS-SG requires fewer computational, communication, and storage resources without compromising the security compared to related AKE schemes proposed for the SG system.

Remotely controlled IoT has to deal with challenges, e.g., scalability, secure communication, and privacy preservation. The conventional solutions (e.g., hypertext transfer protocol secure—HTTPS) also display poor scaling problems and privacy concerns. Jin et al. used DNS with privacy preservation to design a novel lightweight, secure, and remote IoT monitoring system [39]. The remote monitoring utilized the DNS protocol, whereas the communication between IoT devices and gateways still used the conventional protocols, i.e., the constrained application protocol (CoAP) and MQTT. The communication between IoT devices and the IoT gateway only occurred in the home network for data transmission. The IoT gateway encrypted the received data using asymmetric cryptography and encoded the ciphertext using base64. The base64-encoded ciphertext was registered to the internal DNS server under the name of the IoT device by using the DNS TXT record. This method ensures that only the designated users receive and decrypt the data, which makes it one of the candidates for solving issues present in conventional solutions.

2.2. Post-Quantum IoT

Kumari et al. presented a robust and lightweight scheme for authentication and encryption in resource-constrained IoT devices, considering the high complexity and vulnerability of conventional cryptographic approaches to quantum attacks [40]. Their approach utilizes post-quantum lattice-based techniques and combines them with code-based hybrid encryption. The authentication scheme, based on Ring-Learning with Errors (Ring-LWE), incorporates Bernstein reconstruction in polynomial multiplication to minimize computational costs. This enhancement enables reliable authentication on resource-limited IoT devices. The security analysis of the proposed method confirms its effectiveness in ensuring security and privacy against specific attacks in IoT systems. Notably, the authentication scheme significantly reduced the time required for signature generation and verification to 13.299 ms and 0.735 ms, respectively, by incorporating Bernstein's reconstruction in sparse polynomial multiplication.

Liu et al. proposed a collaborative signature scheme with deterrability (MCSD) designed to achieve post-quantum security in the Industrial Internet of Things (IIoT) environment [41]. Their scheme provides formal security proofs based on the assumption that the asymmetric module learning with errors (AMLWE) and asymmetric module short integer solution (AMSIS) problems are NP-hard. They implemented their scheme using Dilithium (Dmcsd) and Aigis-sig (Amcsd) with varying security levels. By increasing the number of signing parties from 5 to 300, they observed changes in the sizes of the public key, secret key, and signature for different security levels, namely, 1024-bit, 1280-bit, and 1536-bit security. The results indicated that the repetition number increased with the number of signing parties, leading to a higher communication round and reduced protocol efficiency. Additionally, their construction, utilizing the Dilithium multiparty signature and double authentication-preventing signature, enables support for both aggregated signatures and public deterrability.

Public-key cryptosystem holds an important role in IoT security. But, due to their complicated encryption and decryption processes, the classic public-key cryptosystems like RSA and ECC are not applicable for IoT. Shuai et al. [42] introduced a group-based NTRU-like public-key cryptosystem called a group theory research unit (GTRU). To obtain high performance of encryption and decryption processes in GTRU, group G must cater to four conditions stated in [42]. For the NTRU, min L_f, L_j (with chosen p and q) determines the key security of GTRU while Lm (encrypted message) determines the message security of GTRU. The analysis results showed that the proposed GTRU was more secure than

NTRU in countering lattice-based attacks. In other words, the proposed GTRU was proven to be a safe and efficient public-key cryptosystem for IoT.

The Internet of Things (IoT) is a growing paradigm in internet networks that securely connects billions of devices to the Internet. Other well-known paradigms are quantum computing and the Shor algorithm. Both are proven as threats to most cybersecurity cryptographic protocols. Agus et al. proposed an NTRU-based communication protocol to prevent unregistered IoT devices from connecting to the network [27]. The research was carried out using three Raspberry Pi3 B+, one AP, and one server with HTTPS service. The proposed method was deployed in `NTRU-401` and *NTRU*-593. The results confirmed that NTRU encryption could effectively secure end-to-end communications in IoT.

Li et al. [43] proposed a lightweight homomorphic encryption-based privacy-preserving scheme for Industrial IoT (IIoT). The study investigated the privacy issues between data owners, third-party cloud servers, and data users. The breach of information is an important matter in many critical IIoT systems, e.g., smart grids, industrial critical systems (ICS)/supervisory control and data acquisition (SCADA) systems, etc. The topics of privacy assurances in critical infrastructures and IoT services have been the focus of the recent literature. Li et al. created an air quality monitoring system. It enabled remote and non-confident cloud computing to run complex computational on encrypted data and allowed data owners and users to justify the decryption accuracy.

Khalid et al. [44] observed the feasibility of deploying various classes of quantum-resistant cryptography schemes, i.e., Lattice-Based Cryptography (LBC). The study evaluated and compared the deployment of novel LBC on low-resource devices in terms of low-power footprint, remote area, and compact bandwidth requirements with high performance including low-power FPGAs and embedded microprocessors. The implementation processes were optimized by using specific techniques for Cortex-M4 in assembly. The results showed that the proposed method had the most suitable key sizes compactness and simplicity compared to other quantum-safe schemes. Nevertheless, the LBC schemes still face challenges to be implemented in real-world systems because they need a larger public-key size than traditional public-key schemes.

3. Proposed Scheme for IoT

3.1. Design Architecture

Figure 2 depicts the IoT system installed in buildings N, O, and P at Telkom University. This IoT system comprises sensors connected to the RPi-4. The RPi-4 collects data, which are then transmitted using the MQTT protocol over a Wi-Fi network to the HPE Server, referred to as the publisher. This study recommends the addition of a message encryption mechanism to the devices of the publisher. The publisher forwards messages to the broker, which then prioritizes these messages in a queue before routing them. Subsequently, the messages are distributed directly to specific receivers, termed 'subscribers'. This system employs the RSA, NTRU, and SABER encryption methods in its encryption process. These methods were selected due to their high-security levels and minimal resource consumption. With this system in place, adversaries cannot alter the sent messages. Although adversaries might predict the MQTT `topic` dispatched by the original publishers, without the genuine data content, any messages they send will not be processed by the broker.

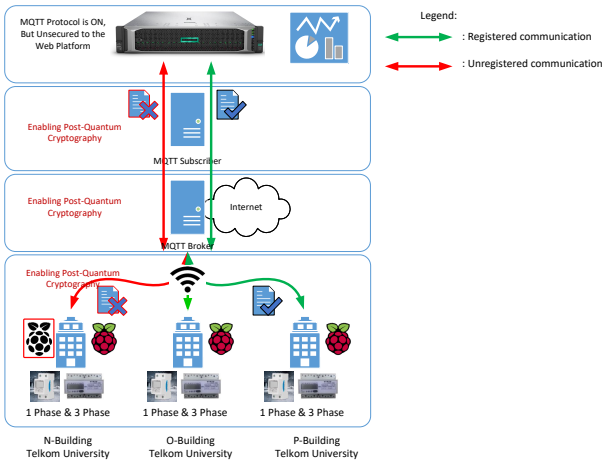

Figure 2. The adversary's perspective.

3.2. Dealing with Public and Private Keys

The public and private keys are generated on the server before being deployed on the RPi-4. The encryption scheme was implemented with moderate (128-bit) and high (196-bit) security levels. The private keys on the publisher's devices are embedded with the public keys for RSA, NTRU, and SABER encryption systems. The public-key encryption systems for each security level are depicted in Table 1. The parameters of the SABER encryption system are summarized in Table 2.

This research customized and adopted the RSA-2048, RSA-7680 [45], NTRU-401, NTRU-593 [46], SABER [47], and Light SABER [48] standards. The NTRU and SABER encryption systems with these standards have a low computational load, but the encrypted character length is limited. The character length that can be encrypted in the NTRU and SABER encryption systems is only 16 bytes for the 128-bit security and 32 bytes for the 196-bit security.

Table 1. Public-Key Encryption (PKE) schemes.

Security Level	Scheme	Artifact Sizes (byte)	
		pk	*sk*
Moderate: 128-bit	RSA 2048	1678	450
	NTRU 401	557	607
	Light SABER	1568	672
High: 196-bit	RSA 7680	5972	1404
	NTRU 593	821	891
	SABER	2304	992

Table 2. SABER's Security Level.

Scheme	l	n	q	p	T	μ	Security	p_{fail}
Light SABER	2	256	2^{13}	2^{10}	2^3	10	NIST-I	2^{-120}
SABER	3	256	2^{13}	2^{10}	2^4	10	NIST-II	2^{-136}
Fire SABER	4	256	2^{13}	2^{10}	2^6	10	NIST-V	2^{-165}

3.3. Developing RSA on RPi-4

This section briefly introduces the RSA. Named after Rivest, Shamir, and Adleman, this method was published in 1977. The security of RSA relies on the practical difficulty of factoring the products of two large prime numbers, also called the 'factoring problem'. Breaking the RSA encryption is known as the RSA problem; whether it is as difficult as

the factoring problem is an open question. There are no published methods to defeat the system using a sufficient key.

The public key consists of modulus n and public (or encryption) exponent e, while the private key consists of private (or decryption) exponent d, which must be kept secret. p, q, and $\lambda(n)$ must also be kept secret because they can be used to calculate d. However, they can all be discarded after d has been computed as RSA key generation. The paramaters for RSA are explained in Table 3, the implementation of RSA encryption (in Algorithm 1), and RSA decryption (in Algorithm 2) for securing MQTT payload make use of Algorithm 3.

Table 3. RSA's Parameters.

Notation	Definition
p, q	Choose two distinct prime numbers
N	The modulus for both the public and private keys
$\phi(n)$	Carmichael's totient function for kept secret p and q)
gcd	Computing the greatest common divisor (gcd) of two prime numbers
pk	Public key that has been released
sk	Secret key that will be used

Algorithm 1 RSA Encryption

$m, Message$
$c = m^e \bmod n$

Algorithm 2 RSA Decryption

$c, Ciphertext$
$c = c^d \bmod n$

Algorithm 3 Customized RSA on Publisher

INITIALIZE pk, Public Key; m, message;
INITIALIZE e, Encryption Byte Capability
INITIALIZE *publisher_id*
INITIALIZE $\mathcal{C}_a$, Ciphertext Array
INITIALIZE RSA.encrypt, RSA Encryption function
INITIALIZE base64.encode, base64 Encoding function
INITIALIZE MQTT.publish, MQTT publish function
$x = \mathsf{length}(m)$
$i = 0$
while $\frac{x-(x \bmod e)}{e} + 1 > i$ **do**
    $m' \leftarrow \mathsf{RSA.encrypt}(c, pk)$
    $\mathcal{C}_a[i] \leftarrow m'$
    $i = i + 1$
end while
MQTT.Publish(base64.Encode($\mathcal{C}_a$))

3.4. Developing NTRU on RPi-4

A public-key cryptosystem, which is called the NTRU cryptosystem, involves a set of parameters (N, P, q). The key parameters are summarized in Table 4. A prime N and an integer q form a ring of convolutional polynomials, where all arithmetic operations take place according to Equation (1):

$$R_{N,q} = (\mathbb{Z}/\mathbb{Z}q)[X]/([X]^{-1}) \quad (1)$$

The key generation process takes place when all necessary parameters (N, p, q, d) act as the input. In order to obtain the key pair of the NTRUEncrypt, the steps on NTRU key generation have to be performed. The Encryption function takes input message M

and public key $h(x)$, and runs the steps on Algorithm 4 to generate ciphertext $c(x)$ from message M under the NTRUEncrypt $h(x)$.

Table 4. NTRU's Parameters.

Notation	Definition
$\mathbb{Z}$	Integers
N, q	Ring parameters [Equation (1)]
p	Message space modulus.
d_1, d_2, d_3	Non-zero coefficient counts for product form polynomial terms
d_g	Non-zero coefficient counts for private key components g
d_m	Message representative Hamming weight constraint

Algorithm 4 NTRU Encryption

$m \in R_p$
$r \in T(d, d)$
$c = pr \cdot h + m \pmod{n}$

To perform successful decryption on encrypted $c(x)$ using $h(x)$, the receiver party needs a private key $f(x)$ that correlates with $h(x)$, while, to recover message M, the procedures in Algorithm 5 should be carried out. Implementation of NTRU encryption for securing the MQTT payload can be seen in Algorithm 6.

Algorithm 5 NTRU Decryption

$f \cdot e \pmod{q}$
$r \in T(d, d)$
$c = pr \cdot h + m \pmod{q}$

Algorithm 6 Customized NTRU on Publisher

INITIALIZE *pk*, Public Key; *m*, message;
INITIALIZE *e*, Encryption Byte Capability
INITIALIZE *publisher_id*
INITIALIZE $\mathcal{C}_a$, Ciphertext Array
INITIALIZE NTRU.encrypt, NTRU Encryption function
INITIALIZE base64.encode, base64 Encoding function
INITIALIZE MQTT.publish, MQTT publish function

$x = \mathsf{length}(m)$
$i = 0$
while $\frac{x-(x \bmod e)}{e} + 1 > i$ **do**
    $m' \leftarrow \mathsf{NTRU.Encrypt}(c, pk)$
    $\mathcal{C}_a[i] \leftarrow m'$
    $i = i + 1$
end while
$\mathsf{MQTT.Publish}(\mathsf{base64.Encode}(\mathcal{C}_a))$

3.5. Developing SABER on RPi-4

This section defines SABER as a lattice-based post-quantum key encapsulation scheme (KEM) that can be transformed into a secure public-key encryption (PKE) scheme. As respectively shown in Algorithms 7 and 8, the algorithms used in SABER consist of two moduli q and p, with $\in_q$ and $\in_p$ bit lengths. On generating public key *pk* and secret key *sk*, 32 bytes of random $seed_A$ is used. It expands inside the *gen* function in SABER key generation and uses SHAKE-128 to generate a random matrix A. The complete parameters for SABER are explained in Table 5.

Algorithm 7 SABER Encryption

$m \in \mathcal{M}; r$
$pk = (b, seed_A)$
$A \leftarrow \mathsf{gen}(seed_A)$
$s' \leftarrow \beta_\mu(R_q^{l \times 1})$
$b' = \mathsf{bits}(A^T s' + h, \in_q, \in_p) \in R_p^{l \times 1}$
$v' = b^T \mathsf{bits}(s', \in_p, \in_p) \in R_p$
$c_m = \mathsf{bits}(v' + h_1 + 2^{\in_p - 1} m, \in_p, \in_r + 1)$
Return $c := (c_m, b')$

Algorithm 8 SABER Decryption

$c = (c_m, b')$
$sk = s$
$v = b'^T \mathsf{bits}(s, \in_p, \in_p) \in R_p$
$m' = (v - 2^{\in_p - \in_t - 1} c_m + h_2, \in_p, 1) \in R_2$
Return m'

Table 5. SABER's Parameters.

Notation	Definition
p, q	Choose two distinct prime numbers
N	The modulus for both the public and private keys
$\phi(n)$	Carmichael's totient function for kept secret (p and q)
gcd	Computing the greatest common divisor (gcd) of two prime numbers
pk	Public key that has been released
sk	Secret key that will be used

On the Encryption operation in Algorithm 7, the message $m \in \mathcal{M}$ is an element of R_q, where $\mathcal{M}$ is the message space range in $\mathcal{M} \in \{0,1\}^n$. The ciphertext is generated through matrix generation, binomial sampling, matrix–vector multiplication, and binomial sampling. The decryption in Algorithm 8 is performed by computing the bit switch of vector multiplication. The implementation of SABER encryption for securing MQTT payload can be seen in Algorithm 9.

Algorithm 9 Customized SABER on Publisher

INITIALIZE *pk*, Public Key; *m*, message;
INITIALIZE *e*, Encryption Byte Capability
INITIALIZE *publisher_id*
INITIALIZE $\mathcal{C}_a$, Ciphertext Array
INITIALIZE SABER.encrypt, SABER Encryption function
INITIALIZE base64.encode, base64 Encoding function
INITIALIZE MQTT.publish, MQTT publish function

$x = \mathsf{length}(m)$
$i = 0$
while $\frac{x - (x \bmod e)}{e} + 1 > i$ **do**
    $m' \leftarrow \mathsf{SABER.encrypt}(c, pk)$
    $\mathcal{C}_a[i] \leftarrow m'$
    $i = i + 1$
end while
$\mathsf{MQTT.Publish}(\mathsf{base64.Encode}(\mathcal{C}_a))$

3.6. Data Specification

Transmitted data from monitoring results are shown in Table 6 for one phase and Table 7 for three phases. The data were processed according to specifications from Table 8 for one phase and Table 9 for three phases.

Table 6. Data Stream from 1-Phase Example.

1-Phase Stream Data
"4438ca6fc2a575cfab4fa4f1d4569964", "2022-02-28", "13:59:07", 215.4, 0.03, 3.185766, 5.622129, 6.462, 0.493, 50, 1.49

Table 7. Data Stream from 3-Phase Example.

3-Phase Stream Data
'6cc5dc0059d39c5392784721c78d4bb3', '2022-04-13', '11:22:03', '223.8', '223.8', '0.0', '1.70', '3.50', '0.00', '0.31', '0.63', '0.00', '0.31', '0.07', '0.35', '0.00', '0.42', '0.31', '0.71', '0.00', '0.52', '0.22', '0.50', '0.31', '19265.00', '11065.00', '30331.00', '50.00'

Table 8. The 1-Phase Data Specification.

Parameter	Description	Unit
token	Data Token	-
date	Monitoring Date	-
time	Monitoring Time	-
V	Voltage (V)	V
I	Current	A
P	Power	Watt
Q	Reactive Power (VAr)	VAr
S	Appearance Power (VA)	VA
PF	Energy (kWh)	-
F	Frequency (Hz)	Hz
E	Total Energy (kWh)	kWh

Table 9. The 3-Phase Data Specification.

Parameter	Description	Measure Unit
token	Transmission Token	-
date	Monitoring Date	-
time	Monitoring Time	-
V_a	Voltage Phase R	V
V_b	Voltage Phase S	V
V_c	Voltage Phase T	V
I_a	Current Phase R	A
I_b	Current Phase S	A
I_c	Current Phase T	A
Q_a	Reactive Power R	VAr
Q_b	Reactive Power S	VAr
Q_c	Reactive Power T	VAr
Q_t	Reactive Power Total	VAr
P_a	Active Power R	Watt
P_b	Active Power S	Watt
P_c	Active Power T	Watt
P_t	Active Power Total	Watt
S_a	Appearance Power R	VA
S_b	Appearance Power S	VA
S_c	Appearance Power T	VA
S_t	Appearance Power Total	VA

Table 9. *Cont.*

Parameter	Description	Measure Unit
PF_a	Power Factor R	-
PF_b	Power Factor S	-
PF_c	Power Factor T	-
E_p	Energy Power	kWh
F_p	Forwarded Energy	kWh
S_p	Total Energy	kWh
$Freq$	Frequency	Hz

3.7. Formal Security Proof

For security proof of the applied cryptographic method used in this research, we adjust the encryption scheme parameter according to Bernstein's work on lattice-based security proof comparison [49].

- Table 10 shows the set of multipliers G for each PKE;
- Table 11 shows how short elements a are generated;
- Table 12 shows how the difference $A_a - G$ is generated; this concludes key generation;
- Encryption generates b and $B \approx G_b$ as in Tables 11 and 12, respectively;
- For Product NTRU, Table 13 shows how $A_b + M$ is converted to C and Table 14 shows the set of encoded messages M.

Table 10. Set of Multipliers for Each of The Target PKEs.

System	Parameter Set	Type	Set of Multipliers
NTRU	hps2048401	Quotient	$(\mathbb{Z}/2048)[x]/(x^{401}-1)$
NTRU	hps2048593	Quotient	$(\mathbb{Z}/2048)[x]/(x^{593}-1)$
saber	main	Product	$(\mathbb{Z}/8192)[x]/(x^{256}+1)^{2\times 2}$
saber	fire	Product	$(\mathbb{Z}/8192)[x]/(x^{256}+1)^{3\times 3}$

Table 11. Distribution of Short Elements for Each of The Target PKEs.

System	Parameter Set	Short Element
NTRU	hps2048401	$\mathbf{Z}[x]/(x^{401}-1); -1, 0, 1$
NTRU	hps2048593	$\mathbf{Z}[x]/(x^{593}-1); -1, 0, 1$
SABER	main	$(\mathbf{Z}[x]/(x^{256}+1))^3$; $\sum_{0\leq i<8}\ \{-0.5, 0.5\}$
SABER	fire	$(\mathbf{Z}[x]/(x^{256}+1))^4$; $\sum_{0\leq i<6}\ \{-0.5, 0.5\}$

Table 12. An Approximation Key Offset of the Noisy or Rounded Product NTRU PKEs.

System	Parameter Set	Key Offset
NTRU	hps2048401	$\mathbf{Z}[x]/(x^{401}-1); -1, 0, 1$ weight $127, 127$
NTRU	hps2048593	$\mathbf{Z}[x]/(x^{593}-1); -1, 0, 1$ weight $127, 127$
SABER	main	round $\mathbf{Z}/8192$ to $8\mathbf{Z}$
SABER	fire	round $\mathbf{Z}/8192$ to $8\mathbf{Z}$

Table 13. An Approximation Cipher Offset of the Noisy or Rounded Product NTRU PKEs.

System	Parameter Set	Ciphertext Offset
NTRU	hps2048401	not applicable
NTRU	hps2048593	not applicable
SABER	main	round $\mathbf{Z}/8192$ to $512\mathbf{Z}$
SABER	fire	round $\mathbf{Z}/8192$ to $128\mathbf{Z}$

Table 14. Set of Encoded Messages for Each of The Target Product NTRU PKEs.

System	Parameter Set	Set of Encoded Messages
NTRU	hps2048401	not applicable
NTRU	hps2048593	not applicable
SABER	main	$\sum_{0 \leq i < 256} \{0.4096\} x^i$
SABER	fire	$\sum_{0 \leq i < 256} \{0.4096\} x^i$

4. Evaluation and Analysis

This section discusses the testing and compares the encryption time consumption of RSA 2048, RSA 7680, NTRU 401, NTRU 539, SABER, and Light SABER on the system as explained in the previous chapter along with the reliability tests of the Light SABER encryption system when implemented on the MQTT protocol. The measurements of the encryption time consumption were carried out on IoT RPi-4 devices, while the encryption reliability tests on the MQTT protocol were carried out on RPi-4 publisher devices, Raspberry Pi broker devices, and desktop subscriber devices. The test was performed by calculating the encryption speed of several encryption standards on RPi-4 devices. The object to be encrypted was a file containing electrical energy monitoring for one-phase and three-phase voltages at buildings N, O, and P at Telkom University.

4.1. 1/3-Phase Encryption & Decryption

Each encryption algorithm was successfully implemented on RPi-4 from 27 January 2022 until time of publication. There were only 500 transmitted pieces of data taken from the ESM server for the testing phase. As a comparison with conventional MQTT, Figure 3 shows the examples of ESM transmitted data in one-phase. The generated data were different from the data in the three-phase, but only in the data length. For example, the plaintext data sent from registered RPi-4 were `"device_id, voltage, current, and power"`; even by enabling MQTT, the `message` still can be derived as plaintext. This verifies the possibility of eavesdropping by unregistered RPi-4 in a public network. Meanwhile, Figure 4 shows that the adversary still can conduct sniffing, but the messages that have been received will be meaningless without the private key.

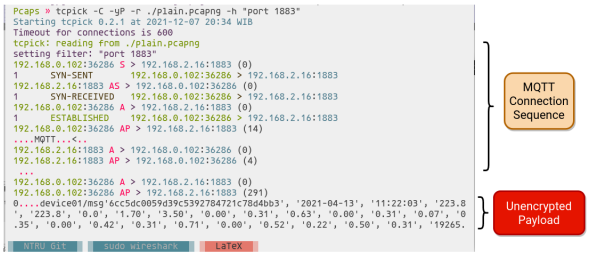

Figure 3. Network sniffing result on the conventional topology in 1-phase.

Figure 4. Analyzing the secured topology in 1-phase.

4.2. Security Performance Evaluation

This research tested and analyzed the data transmission security system from the publisher to the subscriber. The publisher used 1/3 phase of data power in buildings N, P, and O at Telkom University, while the subscriber used an ESM server that was also sent via Telkom Digital Service Division (DDS). Based on Shodan's data, as of June 2022, there were still 134,879 brokers or IoT devices in the world that used port 1883, known as the default port for the MQTT protocol, with South Korea being the largest user. Therefore, this research also discusses four types of attacks against ESM systems for end-to-end communication security.

Based on Figure 5, the publishers send data to the subscriber through a broker using the MQTT protocol based on the topics that have been created by the publishers. If the broker and subscriber are streamed through the internet, it is very possible that several attacks may occur and could be harmful to the whole system. Figure 5 also shows the possibilities of adversaries acting as publishers or subscribers. The attacks tested on this ESM were data privacy, authentication, data integrity, and DDoS mitigation.

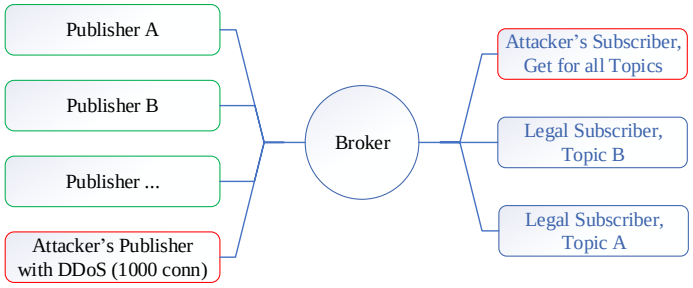

Figure 5. Device threats and attack scenarios.

4.2.1. Attack on Data Privacy

In a scenario where the adversary is in the same network as the publishers, the adversary can sniff the traffic passing through the network, as shown in Figure 6. As can be seen in the screenshot, the data sent by the publisher, "*This data should've been hidden*", should not be allowed to be seen by unregistered publishers.

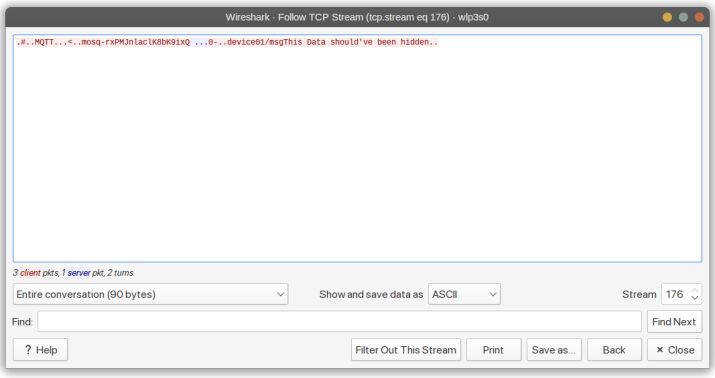

Figure 6. Adversary performs sniffing using Wireshark on unencrypted data.

To prevent that scenario from occurring, the payload (data content) must be secured using encryption algorithms. This research implemented several comparison algorithms, e.g., RSA, NTRU, and SABER. The example shown in Figure 7 uses a post-quantum cryptography algorithm, namely, NTRU. As seen in Figure 7, the adversary cannot sniff any critical data because the MQTT traffic is now only transmitting NTRU ciphertext. All communications from the publishers to the subscribers are encapsulated with an NTRU post-quantum cryptography header.

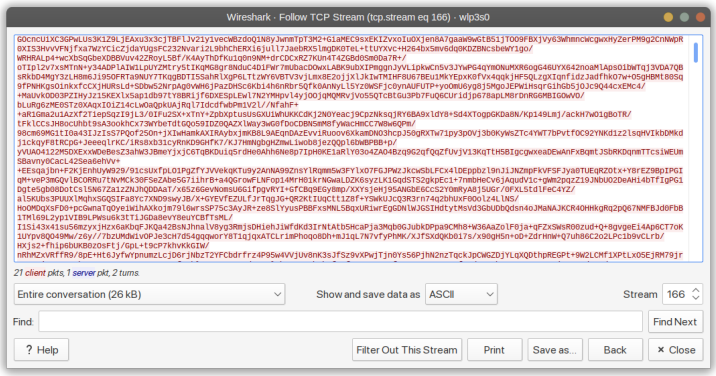

Figure 7. Sniffed Publisher NTRU-Encrypted Payload on Wireshark.

4.2.2. Attack on Authentication

Based on Figure 8, the subscribers are able to decrypt if the keys exchanged between the publishers and the subscribers are matched. If the keys match, then the communication can be carried out according to the payload (data content) sent by the publisher. Then, the ciphertext data can be changed into plaintext again and read by the subscriber.

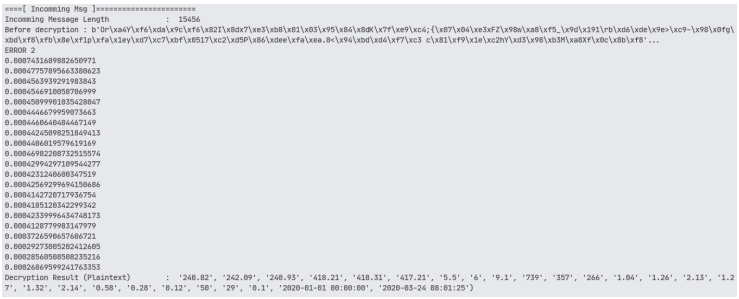

Figure 8. Data transmission between subscribers and publishers occurs with the correct keys (private key and public key).

Based on the first authentication attack scenario, the adversary tries to impersonate an existing IoT device (publisher). The adversary also tries to send a tampered payload containing the original payload to the subscriber. Figure 9 shows that an error, "*incorrect padding*", occurs when the publisher tries to enter the subscriber and send the payload. The size of the plaintext data sent by the publisher will not affect the ciphertext because it is wrapped again with the base64 encoding method.

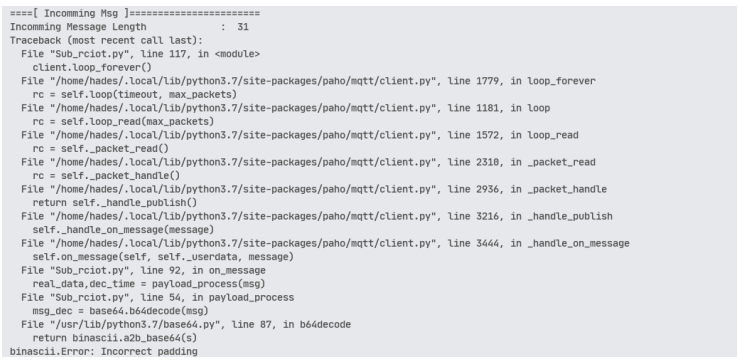

Figure 9. Subscriber trying to receive a payload from the publisher with the wrong public key.

The second authentication attack scenario was anyone who tried to subscribe to the network traffic. Legitimate subscribers only receive an encrypted payload. Without using the correct private key, the incoming data are considered as babbles without content.

Figure 10 shows that unauthorized subscribers receive data that is completely incomprehensible. This scenario was made to simulate dealing with adversaries originating from the subscribers.

Figure 10. Subscriber trying to subscribe using the wrong private key.

4.2.3. Attack on Data Integrity

Any changes to a valid encrypted payload that are made by the adversary damage its structure. This can happen when the adversary (publisher) performs sniffing on the traffic and retransmits the modified payloads to the subscribers. The modified payload could be malicious. On the subscriber's side, the decryption process on the malicious payloads fails because it was modified without the correct public key. The effect of this process is the damage to the payload structures and the creation of data that are considered invalid, as addressed in Table 15 and depicted in Figure 11.

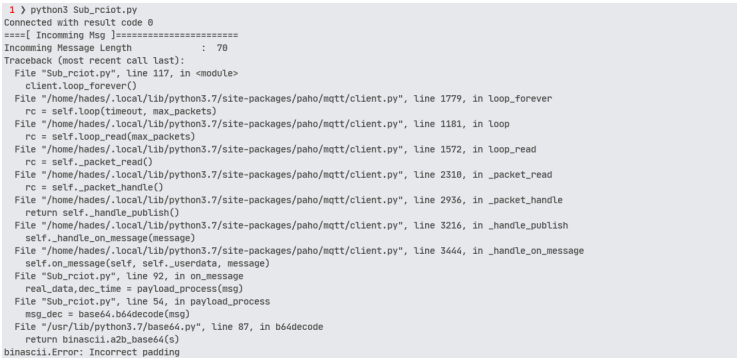

Figure 11. Attacks to data integrity in the publisher's devices.

Table 15. Firewall rule.

Rule No.	Source	Destination	Protocol	Port	Action
1	Unregistered	Subscriber	TCP	MQTT	deny
2	Subscriber	Unregistered	TCP	MQTT	deny
3	Registered	Subscriber	TCP	MQTT	allow
4	Subscriber	Registered	TCP	MQTT	allow

4.2.4. DDOS Mitigation

A whitelist, as shown in Figure 12, is used to overcome DDoS on the broker's devices. In other words, only devices on the whitelist are allowed to transmit through the broker using the MQTT protocol as shown in Figure 13. Figure 14 shows that the adversary, who is not on the list, performs the DDoS, and the device is automatically rejected with the message, "*Error: Connection timed out*". Furthermore, on the firewall side, a schedule can be made to carry out periodic monitoring and show the firewall log on the broker, as shown in Figure 15.

```
❯ sudo ufw show added
Added user rules (see 'ufw status' for running firewall):
ufw allow 'Nginx Full'
ufw allow 22/tcp
ufw allow 8883/tcp
ufw allow from 182.244.130.112 to any port 1883
ufw allow from 182.244.130.100 to any port 1883
ufw allow from 182.244.130.101 to any port 1883
ufw allow from 182.244.130.103 to any port 1883
```

Figure 12. The firewall rules on the broker's devices.

```
❯ mosquitto_pub -m "connection test" -t "device01/msg" -h nnag.xyz -d
Client mosq-N2pG6efIosuzDc4zn0 sending CONNECT
Client mosq-N2pG6efIosuzDc4zn0 received CONNACK (0)
Client mosq-N2pG6efIosuzDc4zn0 sending PUBLISH (d0, q0, r0, m1, 'device01/msg', ... (15 bytes))
Client mosq-N2pG6efIosuzDc4zn0 sending DISCONNECT
```

Figure 13. Devices successfully delivered the payload (whitelisted).

```
❯ mosquitto_pub -m "connection test" -t "device01/msg" -h nnag.xyz -d
Error: Connection timed out
```

Figure 14. Devices not on the whitelist fail the transmission process.

```
❯ sudo dmesg | grep '\[UFW' | grep "180.244.130.112"
[746625.625583] [UFW BLOCK] IN=ens33 OUT= MAC=00:0c:29:54:3d:76:b4:43:26:a2:96:a5:08:00 SRC=180.244.130.112 DST=192.168.100.135 LEN=60 TOS=0x00 PREC=0x00 TTL=55 ID=650$
6 DF PROTO=TCP SPT=7628 DPT=1883 WINDOW=64240 RES=0x00 SYN URGP=0
[746626.654611] [UFW BLOCK] IN=ens33 OUT= MAC=00:0c:29:54:3d:76:b4:43:26:a2:96:a5:08:00 SRC=180.244.130.112 DST=192.168.100.135 LEN=60 TOS=0x00 PREC=0x00 TTL=55 ID=650$
7 DF PROTO=TCP SPT=7628 DPT=1883 WINDOW=64240 RES=0x00 SYN URGP=0
[746628.677889] [UFW BLOCK] IN=ens33 OUT= MAC=00:0c:29:54:3d:76:b4:43:26:a2:96:a5:08:00 SRC=180.244.130.112 DST=192.168.100.135 LEN=60 TOS=0x00 PREC=0x00 TTL=55 ID=650$
```

Figure 15. Log of the firewall on the broker thwarting payload delivery.

4.3. Embedded System Performance Analysis

Figures 16 and 17 show the average CPU and RAM usage of the RPi-4 while running the RSA, NTRU, and SABER encryption and decryption for more than 500 payload transmissions using MQTT protocol. The average RAM consumption for the encryption process in the 128-bit security level was 37% for RSA-7680, 18% for NTRU-401, and 26% for Light Saber. While in the 196-bit security level, the average RAM consumption was 53% for RSA-2048, 24% for NTRU-539, and 27% for Light SABER. The average CPU consumption for the encryption process in the 128-bit security level was 37% for RSA-7680, 18% for NTRU-401, and 26% for Light SABER. In the 196-bit security level, the average CPU consumption for the encryption process was 53% for RSA-7680, 24% for NTRU-539, and 27% for SABER. The average RAM consumption of the decryption process with a 128-bit security level was 8% for RSA-2048, 3% for NTRU-401, and 4% for Light SABER. With a 196-bit security level, the average RAM consumption was 18% for RSA-7680, 3% for NTRU-539, and 7% for SABER. The average CPU consumption of the decryption process with 128-bit security was 23% for RSA-7680, 18% for NTRU-401, and 16% for Light SABER, while, with 196-bit security, it was 22% for RSA-7680, 19% NTRU-539, and 20% for SABER.

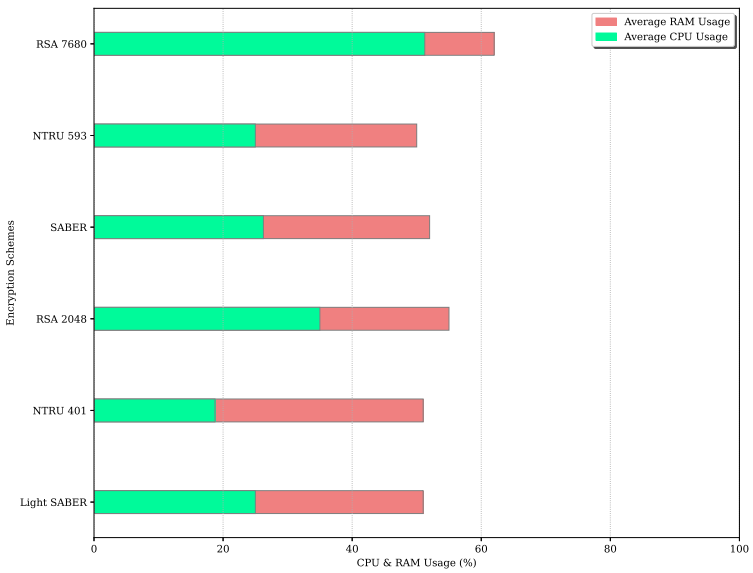

Figure 16. Average CPU andRAM Usage for the Encryption Process.

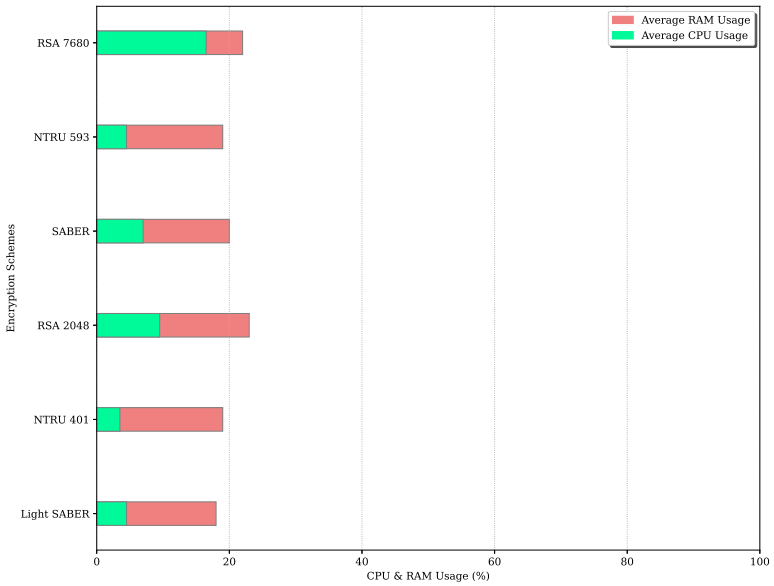

Figure 17. Average CPU and RAM Usage for the Decryption Process.

The data transmitted from the publisher to the subscriber were securely encrypted for this scenario. When comparing the average encryption time of each customized algorithm with the RSA-4028 at the 128-bit security level, it can be concluded that Light SABER was 43.556 times faster and NTRU-401 was 4.668 times faster than the RSA-4028. The comparison with RSA-7680 at the 196-bit security level shows that the SABER was 8.123 times faster and the NTRU-539 was 8.889 times faster than the RSA-7680. The average decryption time comparison in the 128-bit security level summarizes that the Light SABER was 16.956 times faster and the NTRU-401 was 9.953 times faster than RSA-2048. In the 192-bit security level, the SABER was 269.955 times faster and the NTRU-593 was 5.301 times faster than the RSA-7680. This proves that a post-quantum encryption system is a feasible scheme for RPi-4 or other low-resource devices on the Internet of Things. As for the validation of RSA, NTRU, and SABER, the average time for the encryption and decryption processes were averaged and are shown in Figures 18 and 19, respectively.

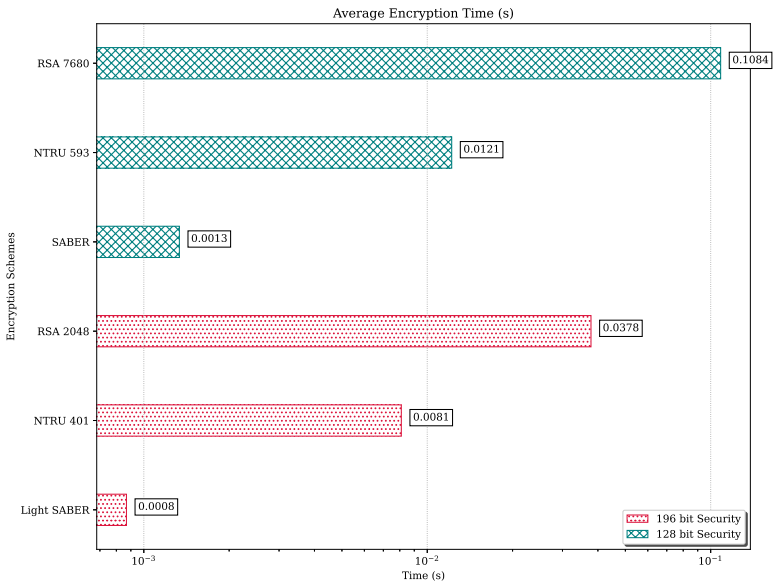

Figure 18. Encryption Average Time for Publisher.

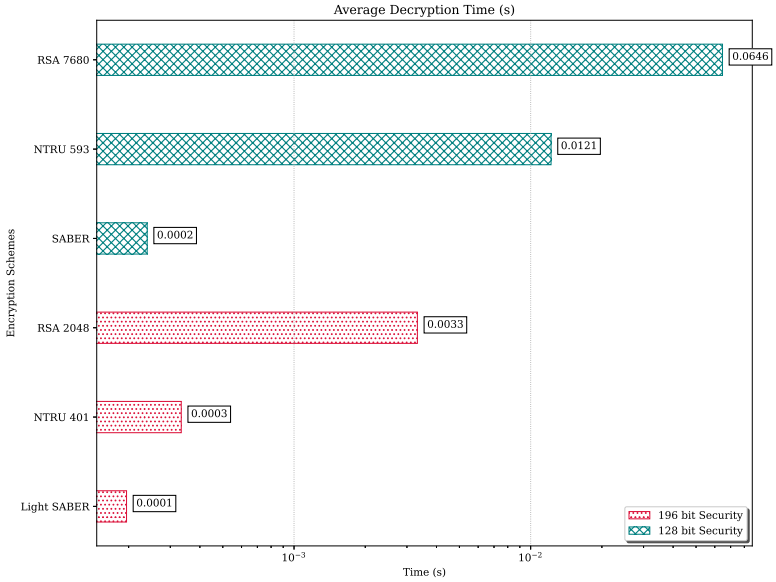

Figure 19. Decryption Average Time for Subscriber.

4.4. Discussion

The proposed design was verified by evaluating existing benchmark encryption techniques, e.g., RSA-2048, RSA-7680, NTRU-401, NTRU-593, SABER, and Light SABER. The detailed key specifications for those encryption algorithms are explained in Table 1. The performance evaluation for those encryption algorithms was calculated by using the encryption time (as shown in Table 16) and the decryption time (as shown in Table 17). The benchmark results of the encryption and decryption with 500 stream data on RPi-4 in a power monitoring system indicate that SABER and Light SABER outperformed both NTRU schemes and the standard encryption (RSA-2048 and RSA-7680), as shown in Figures 18–21. Based on the evaluation of CPU and RAM usage, the post-quantum encryptions have been proven to consume slightly fewer resources while maintaining the same security level. Considering the post-quantum encryptions, the NTRU and SABER encrypted the messages sent via the MQTT protocol in less than 100 ms. Thus, the encryption and decryption time have a significant impact on low-resource hardware.

Table 16. Encryption time (in s) between publisher and subscriber.

	Moderate Security Level			**High Security Level**		
N-th Data	**Light SABER**	**NTRU 401**	**RSA 2048**	**SABER**	**NTRU 593**	**RSA 7680**
1	0.000154	0.0111	0.0307	0.00027	0.01219	0.1181
2	0.000778	0.0050	0.0487	0.00177	0.014098	0.1499
3	0.000769	0.0085	0.0264	0.00186	0.014003	0.1217
4	0.000774	0.0131	0.0348	0.00162	0.015553	0.1234
...	...	...	...	...	...	...
498	0.000841	0.0109	0.0448	0.00063	0.015934	0.1230
499	0.000776	0.0062	0.0236	0.00041	0.004614	0.1183
500	0.000772	0.0120	0.0493	0.00079	0.006216	0.0930

Meanwhile, the reliability of the security system from post-quantum cryptography was evaluated by attacking data privacy, authentication, data integrity, and DDoS. From these four types of attacks, the proposed system was proven to be able to fend off SCADA system cyberattacks and cyber-warfare. Post-quantum cryptography can also help to defend against those risks, protect critical applications and data, and recover from breaches or failures.

Table 17. Decryption time (in s) between publisher and subscriber.

	Moderate Security Level			High Security Level		
N-th Data	**Light SABER**	**NTRU 401**	**RSA 2048**	**SABER**	**NTRU 593**	**RSA 7680**
1	0.00004	0.00029	0.00336	0.00006	0.01219	0.07623
2	0.00017	0.00029	0.00295	0.00017	0.01410	0.06292
3	0.00019	0.00030	0.00324	0.00018	0.01400	0.06221
4	0.00020	0.00029	0.00326	0.00021	0.01555	0.06240
...	...	...	...	...	...	...
498	0.00017	0.00031	0.00320	0.00030	0.01593	0.06443
499	0.00017	0.00031	0.00330	0.00008	0.00461	0.06283
500	0.00023	0.00047	0.00346	0.00051	0.00622	0.06269

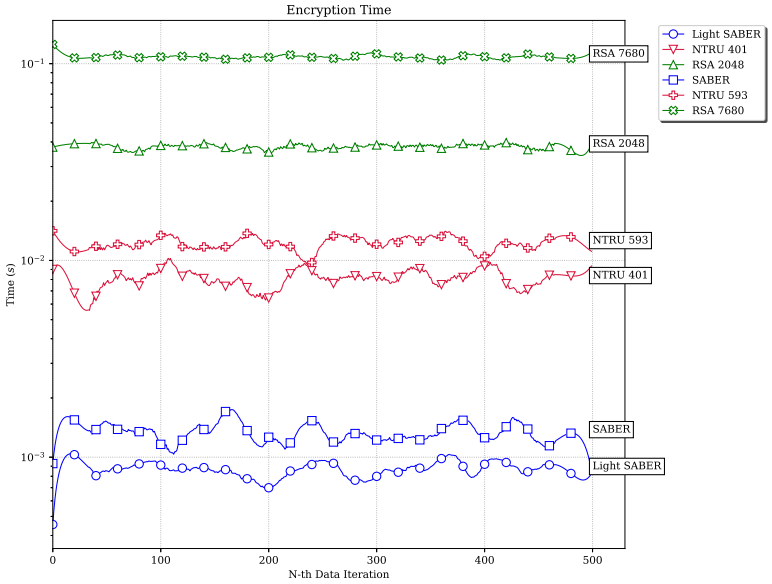

Figure 20. Encryption Time.

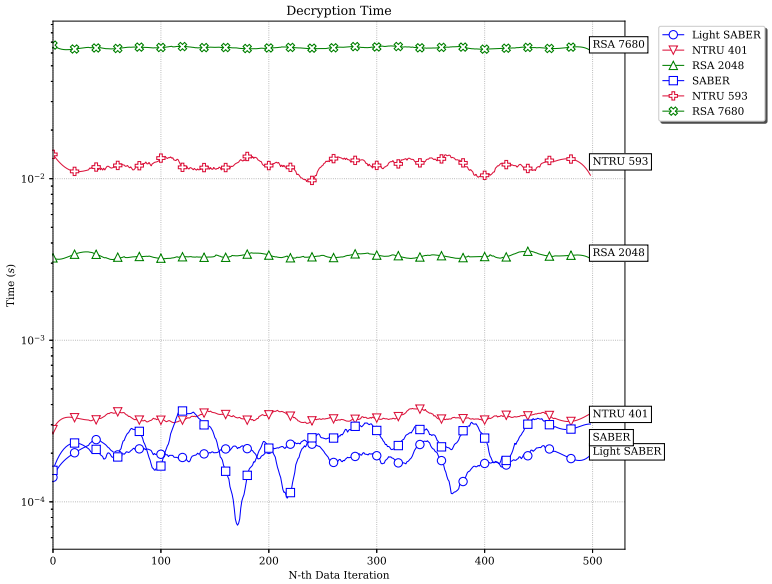

Figure 21. Decryption Time.

5. Concluding Remarks

This research concludes that the IoT data transmission system utilizing the MQTT protocol on RPi-4 devices has the potential for deployment in real-world IoT networks

while ensuring secure communications. We recommend that general IoT systems (ARMv8-A) integrate the Light SABER, SABER, and NTRU post-quantum encryption methods. This adoption will enhance their reliability, bringing them closer to real-time operation without demanding additional resources. Our experiments with data transmission in ESMs using SABER encryption affirm that high-security encryption systems are viable for low-resource IoT devices. Messages of both one-phase and three-phase lengths were fully encrypted using the padding feature and then transmitted via the MQTT protocol. Future refinements to the system proposed in this study might involve evaluating the design of an IoT data delivery system, employing the MQTT protocol with alternative cost-effective microcomputer devices. This could pave the way for more affordable IoT technology. In subsequent studies, we will aim to incorporate other post-quantum encryption systems into IoT platforms that utilize the MQTT protocol.

Author Contributions: Conceptualization, G.B.S.; methodology, Y.M.A.; software, Y.M.A.; validation, G.B.S., Y.M.A., and A.B.M.; experiment and analysis, Y.M.A.; writing—original draft preparation, G.B.S. and Y.M.A.; writing—review and editing, G.B.S., Y.M.A., and A.B.M.; supervision, A.B.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the School of Engineering, Applied Science, and Technology at Canadian University Dubai, Dubai, UAE.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This research was conducted by the School of Engineering, Applied Science, and Technology at Canadian University Dubai, and Telecom Infra Project Community Lab at Telkom University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mohamad Noor, M.b.; Hassan, W.H. Current research on Internet of Things (IoT) security: A survey. *Comput. Netw.* **2019**, *148*, 283–294. [CrossRef]
2. Sadhu, P.K.; Yanambaka, V.P.; Abdelgawad, A. Internet of Things: Security and Solutions Survey. *Sensors* **2022**, *22*, 7433. [CrossRef] [PubMed]
3. Stoyanova, M.; Nikoloudakis, Y.; Panagiotakis, S.; Pallis, E.; Markakis, E.K. A Survey on the Internet of Things (IoT) Forensics: Challenges, Approaches, and Open Issues. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1191–1221. [CrossRef]
4. Ali, A.; Mateen, A.; Hanan, A.; Amin, F. Advanced Security Framework for Internet of Things (IoT). *Technologies* **2022**, *10*, 60. [CrossRef]
5. Shi, Q.; Yang, Y.; Sun, Z.; Lee, C. Progress of advanced devices and internet of things systems as enabling technologies for smart homes and health care. *ACS Mater. Au* **2022**, *2*, 394–435. [CrossRef]
6. Rejeb, A.; Rejeb, K.; Simske, S.; Treiblmaier, H.; Zailani, S. The big picture on the internet of things and the smart city: A review of what we know and what we need to know. *Internet Things* **2022**, *19*, 100565. [CrossRef]
7. Soori, M.; Arezoo, B.; Dastres, R. Internet of things for smart factories in industry 4.0, a review. *Internet Things Cyber-Phys. Syst.* **2023**, *3*, 192–204. [CrossRef]
8. Abdulkareem, K.H.; Mohammed, M.A.; Salim, A.; Arif, M.; Geman, O.; Gupta, D.; Khanna, A. Realizing an Effective COVID-19 Diagnosis System Based on Machine Learning and IOT in Smart Hospital Environment. *IEEE Internet Things J.* **2021**, *8*, 15919–15928. [CrossRef] [PubMed]
9. Kimani, K.; Oduol, V.; Langat, K. Cyber security challenges for IoT-based smart grid networks. *Int. J. Crit. Infrastruct. Prot.* **2019**, *25*, 36–49. [CrossRef]
10. Bhamare, D.; Zolanvari, M.; Erbad, A.; Jain, R.; Khan, K.; Meskin, N. Cybersecurity for industrial control systems: A survey. *Comput. Secur.* **2020**, *89*, 101677. [CrossRef]
11. Upadhyay, D.; Sampalli, S. SCADA (Supervisory Control and Data Acquisition) systems: Vulnerability assessment and security recommendations. *Comput. Secur.* **2020**, *89*, 101666. [CrossRef]
12. Kulik, T.; Tran-Jørgensen, P.W.V.; Boudjadar, J. Compliance verification of a cyber security standard for Cloud-connected SCADA. In Proceedings of the 2019 Global IoT Summit (GIoTS), Aarhus, Denmark, 17–21 June 2019; pp. 1–6. [CrossRef]
13. Cifranic, N.; Hallman, R.A.; Romero-Mariona, J.; Souza, B.; Calton, T.; Coca, G. Decepti-SCADA: A cyber deception framework for active defense of networked critical infrastructures. *Internet Things* **2020**, *12*, 100320. [CrossRef]

14. Gumaei, A.; Hassan, M.M.; Huda, S.; Hassan, M.R.; Camacho, D.; Del Ser, J.; Fortino, G. A robust cyberattack detection approach using optimal features of SCADA power systems in smart grids. *Appl. Soft Comput.* **2020**, *96*, 106658. [CrossRef]
15. Ferrag, M.A.; Babaghayou, M.; Yazici, M.A. Cyber security for fog-based smart grid SCADA systems: Solutions and challenges. *J. Inf. Secur. Appl.* **2020**, *52*, 102500. [CrossRef]
16. Chehri, A.; Fofana, I.; Yang, X. Security Risk Modeling in Smart Grid Critical Infrastructures in the Era of Big Data and Artificial Intelligence. *Sustainability* **2021**, *13*, 3196. [CrossRef]
17. Baker, T.; Asim, M.; MacDermott, Á.; Iqbal, F.; Kamoun, F.; Shah, B.; Alfandi, O.; Hammoudeh, M. A secure fog-based platform for SCADA-based IoT critical infrastructure. *Softw. Pract. Exp.* **2020**, *50*, 503–518. [CrossRef]
18. Alanazi, M.; Mahmood, A.; Chowdhury, M.J.M. SCADA vulnerabilities and attacks: A review of the state-of-the-art and open issues. *Comput. Secur.* **2023**, *125*, 103028. [CrossRef]
19. Aikins, S.K. Managing Cybersecurity Risks of SCADA Networks of Critical Infrastructures in the IoT Environment. In *Security, Privacy and Trust in the IoT Environment*; Springer International Publishing: Cham, Switzerland, 2019; pp. 3–23. [CrossRef]
20. Islam, S.N.; Baig, Z.; Zeadally, S. Physical Layer Security for the Smart Grid: Vulnerabilities, Threats, and Countermeasures. *IEEE Trans. Ind. Inform.* **2019**, *15*, 6522–6530. [CrossRef]
21. Lin, Z.; Lin, M.; Champagne, B.; Zhu, W.P.; Al-Dhahir, N. Secrecy-Energy Efficient Hybrid Beamforming for Satellite-Terrestrial Integrated Networks. *IEEE Trans. Commun.* **2021**, *69*, 6345–6360. [CrossRef]
22. Yan, X.; Yan, K.; Rehman, M.U.; Ullah, S. Impersonation Attack Detection in Mobile Edge Computing by Levering SARSA Technique in Physical Layer Security. *Appl. Sci.* **2022**, *12*, 10225. [CrossRef]
23. Marabissi, D.; Mucchi, L.; Stomaci, A. IoT Nodes Authentication and ID Spoofing Detection Based on Joint Use of Physical Layer Security and Machine Learning. *Future Internet* **2022**, *14*, 61. [CrossRef]
24. Song, J.; Seo, S.C. Secure and Fast Implementation of ARX-Based Block Ciphers Using ASIMD Instructions in ARMv8 Platforms. *IEEE Access* **2020**, *8*, 193138–193153. [CrossRef]
25. Medileh, S.; Laouid, A.; Nagoudi, E.M.B.; Euler, R.; Bounceur, A.; Hammoudeh, M.; AlShaikh, M.; Eleyan, A.; Khashan, O.A. A flexible encryption technique for the internet of things environment. *Ad Hoc Netw.* **2020**, *106*, 102240. [CrossRef]
26. Liu, Z.; Azarderakhsh, R.; Kim, H.; Seo, H. Efficient Software Implementation of Ring-LWE Encryption on IoT Processors. *IEEE Trans. Comput.* **2020**, *69*, 1424–1433. [CrossRef]
27. Agus, Y.M.; Murti, M.A.; Kurniawan, F.; Cahyani, N.; Satrya, G. An Efficient Implementation of NTRU Encryption in Post-Quantum Internet of Things. In Proceedings of the 2020 27th International Conference on Telecommunications (ICT), Bali, Indonesia, 5–7 October 2020; pp. 1–5. [CrossRef]
28. Zhuang, P.; Zamir, T.; Liang, H. Blockchain for Cybersecurity in Smart Grid: A Comprehensive Survey. *IEEE Trans. Ind. Inform.* **2021**, *17*, 3–19. [CrossRef]
29. Kim, Y.; Hakak, S.; Ghorbani, A. Smart grid security: Attacks and defence techniques. *IET Smart Grid* **2023**, *6*, 103–123. [CrossRef]
30. Tolba, A.; Al-Makhadmeh, Z. A cybersecurity user authentication approach for securing smart grid communications. *Sustain. Energy Technol. Assess.* **2021**, *46*, 101284. [CrossRef]
31. Konstantinou, C.; Mohanty, S.P. Cybersecurity for the Smart Grid. *Computer* **2020**, *53*, 10–12. [CrossRef]
32. Agarkar, A.; Agrawal, H. A review and vision on authentication and privacy preservation schemes in smart grid network. *Secur. Priv.* **2019**, *2*, e62. [CrossRef]
33. Qian, J.; Cao, Z.; Lu, M.; Chen, X.; Shen, J.; Liu, J. The Secure Lattice-based Data Aggregation Scheme in Residential Networks for Smart Grid. *IEEE Internet Things J.* **2021**, *9*, 2153–2164. [CrossRef]
34. Ali, W.; Din, I.U.; Almogren, A.; Kim, B.S. A Novel Privacy Preserving Scheme for Smart Grid-Based Home Area Networks. *Sensors* **2022**, *22*, 2269. [CrossRef]
35. Reijsbergen, D.; Maw, A.; Dinh, T.T.A.; Li, W.T.; Yuen, C. Securing Smart Grids Through an Incentive Mechanism for Blockchain-Based Data Sharing. In Proceedings of the Twelfth ACM Conference on Data and Application Security and Privacy, Baltimore-Washington DC Area, MD, USA, 25–27 April 2022; Association for Computing Machinery: New York, NY, USA, 2022; CODASPY '22, pp. 191–202. [CrossRef]
36. Maitra, S.; Richards, D.; Abdelgawad, A.; Yelamarthi, K. Performance Evaluation of IoT Encryption Algorithms: Memory, Timing, and Energy. In Proceedings of the 2019 IEEE Sensors Applications Symposium (SAS), Sophia Antipolis, France, 11–13 March 2019; pp. 1–6.
37. Manzoor, A.; Liyanage, M.; Braeke, A.; Kanhere, S.S.; Ylianttila, M. Blockchain based proxy re-encryption scheme for secure IoT data sharing. In Proceedings of the 2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Seoul, Republic of Korea, 14–17 May 2019; pp. 99–103.
38. Tanveer, M.; Ahmad, M.; Khalifa, H.S.; Alkhayyat, A.; El-Latif, A.A.A. A new anonymous authentication framework for secure smart grids applications. *J. Inf. Secur. Appl.* **2022**, *71*, 103336. [CrossRef]
39. Jin, Y.; Tomoishi, M.; Fujikawa, K.; Kafle, V.P. A lightweight and secure IoT remote monitoring mechanism using DNS with privacy preservation. In Proceedings of the 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 11–14 January 2019; pp. 1–2.
40. Kumari, S.; Singh, M.; Singh, R.; Tewari, H. A post-quantum lattice based lightweight authentication and code-based hybrid encryption scheme for IoT devices. *Comput. Netw.* **2022**, *217*, 109327. [CrossRef]

41. Liu, J.; Wen, J.; Zhang, B.; Dong, S.; Tang, B.; Yu, Y. A post quantum secure multi-party collaborative signature with deterability in the Industrial Internet of Things. *Future Gener. Comput. Syst.* **2023**, *141*, 663–676. [CrossRef]
42. Shuai, L.; Xu, H.; Miao, L.; Zhou, X. A Group-Based NTRU-Like Public-Key Cryptosystem for IoT. *IEEE Access* **2019**, *7*, 75732–75740. [CrossRef]
43. Li, S.; Zhao, S.; Min, G.; Qi, L.; Liu, G. Lightweight Privacy-Preserving Scheme using Homomorphic Encryption in Industrial Internet of Things. *IEEE Internet Things J.* **2021**, *9*, 14542–14550. [CrossRef]
44. Khalid, A.; McCarthy, S.; O'Neill, M.; Liu, W. Lattice-based Cryptography for IoT in A Quantum World: Are We Ready? In Proceedings of the 2019 IEEE 8th International Workshop on Advances in Sensors and Interfaces (IWASI), Otranto, Italy, 13–14 June 2019; pp. 194–199. [CrossRef]
45. Rivest, R.L.; Shamir, A.; Adleman, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
46. Hoffstein, J.; Pipher, J.; Silverman, J.H. NTRU: A ring-based public key cryptosystem. In *Algorithmic Number Theory*; Buhler, J.P., Ed.; Springer: Berlin/Heidelberg, Germany, 1998; pp. 267–288.
47. D'Anvers, J.P.; Karmakar, A.; Sinha Roy, S.; Vercauteren, F. Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM. In Proceedings of the Progress in Cryptology—AFRICACRYPT 2018, Marrakesh, Morocco, 7–9 May 2018; Joux, A., Nitaj, A., Rachidi, T., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 282–305.
48. Theodorakis, G.; Koliousis, A.; Pietzuch, P.; Pirk, H. LightSaber: Efficient Window Aggregation on Multi-Core Processors. In *SIGMOD '20, Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020*; Association for Computing Machinery: New York, NY, USA, 2020; pp. 2505–2521. [CrossRef]
49. Bernstein, D.J. Comparing proofs of security for lattice-based encryption. *IACR Cryptol. ePrint Arch.* **2019**, *2019*, 691.