QUANTUM-RESISTANT SECURITY FRAMEWORK FOR SCADA COMMUNICATION IN INDUSTRIAL CONTROL SYSTEMS

by

Sagarika Ghosh

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
August 2023

Contents

List of Tables

List of Figures

List of Abbreviations

AES	Advanced Encryption Standard
AGA-12	American Gas Association
ASKMA	Advanced SCADA Key Management Architecture
ECDSA	Elliptic Curve Digital Signature
HKMA	Hybrid Key Management Architecture
HMI	Human-Machine Interface
IEDs	Intelligent Electronic Devices
LKH	Logical Key Hierarchy
MTU	Master Terminal Unit
PLCs	Programmable Logic Controllers
PQC	Post Quantum Cryptography
RSA	Rivest–Shamir–Adleman
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SHA-1	Secure Hash Algorithm
SIKE	Supersingular Isogeny Key Encapsulation
SKE	SCADA Key Establishment
SKMA	SCADA Key Management Architecture

Abstract

Supervisory Control and Data Acquisition (SCADA) systems are essential to monitor industrial processes in critical infrastructure. However, existing SCADA standards and protocols are susceptible to traditional cyber attacks and potential future attacks leveraging quantum computing. This thesis introduces a comprehensive security framework for safeguarding SCADA network communication against cyber attacks. It consists of three modules. Each module focuses on distinct aspects of SCADA security. Module 1 includes an intrusion-resistant SCADA scheme utilizing a quantum key generation based on B92 protocol for encryption and SPHINCS-256, a preimage and collision-resistant algorithm, for digital signature generation. To enhance the security of SPHINCS-256, a true random number generator based on Quantum Random Number Generator (QRNG) is incorporated. It replaces the message with the cipher obtained in the encryption phase, ensuring a higher level of security for SCADA systems. Module 2 focuses on a multi phase quantum resistant scheme for secure communication in SCADA systems. The key generation and exchange protocols involve BBM92 and Supersingular Isogeny Key Encapsulation (SIKE), along with hashed message authentication and masking based on PRESENT algorithm to strengthen the security and reduce the key size. It offers robustness against various attacks leveraging quantum computing capabilities. Module 3 introduces a Lightweight and Robust algorithm for secure SCADA networks. It employs a secure key generation scheme, incorporating random number generators along with Huffman coding to reduce key size while preserving integrity. It also included shared secret key establishment using the radioactive decay law algorithm, providing intrusion detection capabilities. Encryption is also performed using which ensures integrity, freshness, and resistance against replay and quantum key search attacks. It offers resilience against Man-in-the-Middle attacks compared to the current SCADA security standard proposed by American Gas Association (AGA-12).

Chapter 1

Introduction

1.1 Overview of SCADA Networks and its security in Industrial Control Systems

Industrial sectors, including smart power generation and distribution systems, water treatment facilities, chemical plants, and transportation systems, extensively utilize Supervisory Control and Data Acquisition (SCADA) systems for real-time process monitoring and control [1], [2]. Figure 1.1 illustrates the hierarchical structure found in a typical SCADA system. An Human-Machine Interface (HMI) collects data from the Master Terminal Unit (MTU) and issues control commands [1]. The MTU serves as the central control server, facilitating communication between the HMI and geographically dispersed Remote Terminal Unit (RTU). The RTUs enable the transmission of information and control commands between the MTU and field devices, such as sensors and actuators. The communication link connecting the various SCADA components can be established using dedicated serial lines or a Local Area Network (LAN). [1], [3].

SCADA systems are indispensable in the supervision of critical infrastructure processes, playing a crucial role in safeguarding data transmission between RTUs and MTUs against diverse cyber-attacks. In recent years, there has been a notable surge in the incidence of cyber-attacks specifically targeting SCADA systems deployed in power stations, oil and gas facilities, and nuclear control plants [4]. These attacks have evolved beyond conventional methods, such as Denial of Service (DoS) or Man-in-the-Middle attack (MITM) [2], [5].

In February 2021, a SCADA system compromise in a U.S. Water Treatment Facility exposed a vulnerability to unauthorized access. The attackers exploited this vulnerability to manipulate control commands, resulting in an increase in the concentration of sodium hydroxide during the water treatment process. Furthermore, several federal agencies have uncovered the presence of a disruptive malware named

Pipedream in liquefied natural gas plants. Pipedream specifically targets multiple Programmable Logic Controllers (PLCs) and has been employed to disrupt PLCs manufactured by Schneider Electric of France and Omron of Japan [6].

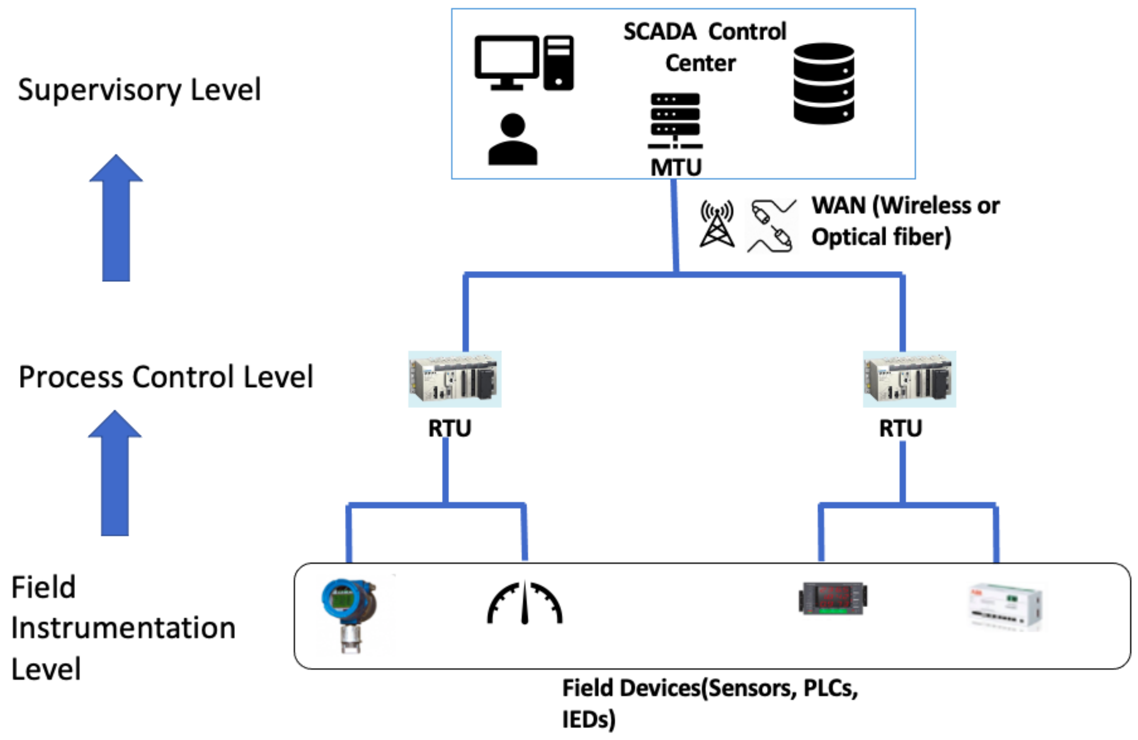

SCADA Control Center
MTU
Supervisory Level
WAN (Wireless or Optical fiber)
Process Control Level
RTU
RTU
Field Instrumentation Level
Field Devices(Sensors, PLCs, IEDs)

Figure 1.1: Block diagram of a SCADA system. MTU: Master Terminal Unit, RTUs: Remote Terminal Units, PLCs: Programmable Logic Controllers, and IEDs: Intelligent Electronic Devices

In addition to conventional attacks, the advent of quantum computing has introduced the cyber-physical sector to the realm of quantum attacks. With the rise of quantum capabilities, traditional algorithms like Rivest–Shamir–Adleman (RSA) based on integer factorization and discrete logarithm problems will become practically breakable using Shor's algorithm [7], [8], [9]. Furthermore, symmetric block ciphers such as Advanced Encryption Standard (AES) and ECC (Elliptic Curve Cryptography), which rely on the security of substitution and permutation schemes, are also vulnerable to quantum search algorithms like Grover's algorithm [10].

The current SCADA standards and protocols, including the AGA-12, are susceptible to both conventional and quantum attacks. These standards utilize RSA for key management, AES for encryption, ECDSA for digital signatures, and the Secure Hash

Algorithm (SHA-1) as the hash function [11]. The security of RSA is compromised by Shor's algorithm, as demonstrated by Gidney et al. [12], which calculates the required quantum resources for the attack. Similarly, Grover's algorithm weakens AES and Elliptic Curve Digital Signature (ECDSA), while collision resistance vulnerabilities have been identified in SHA-1 [13], [14]. Consequently, AGA-12 needs to be fortified with a robust scheme that can withstand both traditional and quantum attacks.

1.2 Mechanism and Security Requirement of SCADA systems

SCADA systems play a crucial role in the monitoring and control of diverse components such as terminal units, sensors, PLCs, and Intelligent Electronic Devices (IEDs). These field devices facilitate data transfer from the field instrumentation level to the supervisory level through the process control layer, where the server verifies the expected parameter values [15]. Once validated by supervisory level components, including the Master Terminal Unit (MTU), the data is transmitted back to the field instrumentation level. Additionally, the SCADA control center eficiently stores and presents the exchanged measured data in a centralized database, enabling comprehensive monitoring and system control. In the context of power plants or water plants, the network infrastructure often encompasses an extensive SCADA network consisting of numerous remote dedicated control units, thereby distributing the load on the centralized server [15], [16].

Cryptography serves as a fundamental tool for achieving secure data exchange, addressing the essential security goals of confidentiality, integrity, and authentication [17]. Our proposed scheme specifically focuses on ensuring confidentiality. It can be broadly classified into Classical and Quantum Cryptography, depending on the type of communication channels employed. Classical cryptography relies on mathematical computations and operates through a single communication channel. It further encompasses asymmetric and symmetric cryptography, which differ in the number and characteristics of keys employed for encryption [17].

In contrast, quantum cryptography leverages the principles of quantum mechanics and utilizes two distinct channels: a quantum channel and a public channel [18]. The quantum channel facilitates the exchange of qubits, the fundamental units of information in quantum computing. Meanwhile, the public channel, also known as the classical channel, enables the exchange of data in the form of classical bits between the sender and receiver. The presence of an intruder or environmental factors can introduce noise in the quantum channel, thereby disrupting the key exchange process. To address this, quantum cryptography employs error correction protocols to rectify errors that may arise during transmission.

In a SCADA system employing quantum cryptography, the Master Terminal Unit (MTU) and Remote Terminal Unit (RTU) uses quantum and classical channel. The quantum channel is used to distribute qubits for raw key exchange. Both RTU and MTU then independently select a basis, which represents a vector, to generate and measure the superposed state of qubits. The choice of basis is negotiated between the RTU and MTU via the classical channel. Following this negotiation, the RTU and MTU proceeds, via the classical channel, with the estimation of errors present in the respective keys through a process known as key sifting [18].

SCADA communication entails the exchange of sensitive data between remote controlled units, making it a prime target for cyber attackers seeking to compromise the operations performed by these dedicated devices. The security of a modern SCADA network hinges on several critical aspects:

- Confidentiality: Ensuring the confidentiality of data is paramount to prevent unauthorized access, data theft, and violations of data privacy via eavesdropping.

- Integrity: Maintaining data integrity is crucial to prevent malicious tampering of information, particularly through Man-in-the-Middle attacks, which can lead to unauthorized modifications.

- Authentication: The Distributed Network Protocol 3.0 used in SCADA systems must employ robust authentication mechanisms. Failure to do so may enable illegitimate interceptors to exploit control systems.

Overall, the SCADA infrastructure necessitates secure and authentic communication to preserve the confidentiality and integrity of the sensitive information exchanged over dedicated communication channels. As a part of future work, authorization and accounting can be introduced based on tailored security measures for specific applications and users, enhancing the overall system security in the future. It is vital to implement robust security measures to safeguard against potential cyber threats in a SCADA environment.

Several research studies have been conducted to enhance the secure communication in SCADA networks. These efforts have led to the development and widespread use of more than 40 standards and protocols across over ten countries globally [2], [19], [20], [21]. However, it is worth noting that the existing standards suffer from weak key management protocols, posing a significant security concern [2], [22], [23], [20], [21]. The key management schemes, categorized into symmetric, asymmetric, and hybrid approaches, exhibit certain drawbacks. Symmetric key cryptography, exemplified by SCADA Key Establishment (SKE), SCADA Key Management Architecture (SKMA), Logical Key Hierarchy (LKH), and Advanced SCADA Key Management Architecture (ASKMA), eficiently ensure message integrity and data availability [2]. However, these schemes lack authentication and confidentiality mechanisms. ID based Key Management Architecture, provide message integrity and authentication capabilities but fall short in ensuring data availability. Hybrid key management schemes, including Hybrid Key Management Architecture (HKMA) and Advanced Hybrid SCADA Key Management Architecture (AHSKMA). They do not address message integrity and authentication [2]. As a result, comprehensive research and advancements are necessary to develop robust key management schemes that effectively address the challenges associated with message integrity, authentication, and data availability in SCADA networks [2], [19], [22], [23].

Moreover, the advent of quantum computing poses a significant threat to the security protocols currently employed by cyber-physical devices. Existing standards and key management protocols have not been proven secure against quantum attacks. Recent theoretical work by Gidney et al. [12] has demonstrated the exploitability of Shor's algorithm against RSA, along with an estimation of the required quantum resources. Another well-known algorithm, Grover's algorithm, exhibits a quadratic speedup compared to classical algorithms, weakening the key strength of AES-128 to a mere 64 bits. In light of the adverse impact of quantum computing on traditional cryptography, researchers have developed various quantum-resistant cryptographic approaches, which can be categorized into Quantum Cryptography and Post Quantum Cryptography (PQC) [14], [24], [25].

Quantum cryptography leverages the fundamental principles of quantum mechanics, which impose unique constraints on the communication channel. Post Quantum cryptography encompasses hash-based, code-based, multivariate polynomial, lattice based, and supersingular isogeny based schemes. Among the various PQC schemes, Supersingular Isogeny Key Encapsulation (SIKE) has emerged as a robust cryptographic solution for standardization, as it generates small key sizes and ciphertext sizes suitable for SCADA control units. However, further optimization of SIKE is required [25]. Furthermore, Seo et al. [26] demonstrates the practical software implementation of the Supersingular Isogeny Key Encapsulation (SIKE) scheme, specifically targeting various security levels on 32-bit ARM Cortex-M4 microcontrollers and proves the small key size of SIKE is a promising factor for low-end microcontrollers in the quantum era. Post-quantum cryptography relies on the premise that existing quantum computers lack the computational power to solve complex problems. In general, post-quantum cryptography provides a security level of 2^{128} against quantum attacks. Nevertheless, in 2017, Chailloux et al. [27] developed a quantum algorithm that reduces the security level of PQC to $2^{119.6}$, emphasizing the need for an algorithm that combines quantum, post-quantum, and classical cryptography approaches.

The preceding discussion highlights the need for a secure and eficient protocol for SCADA communication, capable of protecting against current and potential quantum attacks. This thesis proposes a novel, robust, and quantum-resistant algorithm that combines three approaches: quantum, post-quantum, and classical cryptography, to ensure the secure communication between the MTU and RTU, minimizing the risk of breaches. The proposed approach follows a multi-phase architecture consisting of three main steps. First, the MTU and RTU establish key management, generating a shared secret key using post-quantum cryptography and a session key using quantum cryptography. The shared secret key is then utilized for authentication purposes, employing Hash based Message Authentication Code based on SHA3 (256 bits) during quantum key distribution. Once the authenticated session/symmetric key is established between the MTU and RTU, it is employed as input for the ASCON algorithm, enabling the encryption of the exchanged messages. Additionally, a copy of the message is created, and further processed using HMAC-SHA3 with the shared secret key to obtain the message authentication code. Finally, the message authentication code is appended to the encrypted message, ensuring its integrity, and transmitted from the RTU to the MTU or vice versa.

1.3 Motivation

SCADA monitors crucial industrial processes, such as electric grids, power plants, oil mining and even space substations. However, the security of SCADA Industrial control system (ICS) is vulnerable to various cyber threats, and failure to secure industrial operations can be catastrophic. In the previous work [2], [11], the following research questions are addressed: Does AGA-12 provide confidentiality, integrity, and authentication between RTU and MTU against quantum attacks? Do existing key management schemes resist the communication between RTU and MTU against Shor's algorithm? Will AGA-12 encryption and digital signature resist data exchanged between RTU and MTU against Grover's algorithm?

The boom of quantum computing is both beneficial and detrimental to cyber physical systems. Gidney et al.[12] proved that Shor's algorithm is capable of cracking RSA-2048 within 8 hours with 20 million qubits. However, while a classical search algorithm takes O(N), an exhaustive search by Grover's algorithm takes $O(\sqrt{N})$. Thus, it weakens the key strength of AES-256 to only 64 bits. A hash algorithm should be both preimage and collision-resistant. However, Google has recently cracked SHA-1 by launching a collision attack [28]. SHA-2 and SHA-3 remains safe at present. An attack on SHA-2 or SHA-3 requires on the order of 2^{256} queries in a non-quantum setting and, 2^{128} queries in a quantum setting [28]. Thus, SHA-2,3 has security level on the order of 2^{256} in classical hardware and 2^{128} in quantum hardware.

Quantum computing endangers stable cyber-physical security, and researchers are developing solutions to mitigate the quantum threat. The solutions involve quantum and post-quantum cryptography to secure cyber-physical systems from future attacks by quantum computing [24]. Quantum cryptography involves quantum key distribution protocols that exploit the laws of quantum physics, superposition and entanglement principle to secure against Shor's algorithm [29, 30]. Whereas, post-quantum cryptography uses algorithms based on mathematical operations that are quantum-resistant and can inter-operate with classical network protocols [31].

One of the many post-quantum cryptography areas is the hash-based algorithms to generate a digital signature, which is secure against preimage and collision attacks by a quantum computer [32]. A first preimage attack attempts to recover a message m from only the hash value such that H = hash(m). A second preimage attack finds another message n such that m = n and hash(m)= hash(n). A collision attack aims to retrieve two different messages m and n, such that hash(m) = hash(n) [28, 33]. SPHINCS-256 is a high-security post-quantum stateless hash-based signature scheme that uses two types of signatures, namely, an extension of Winternitz One-time Signature (WOTS+) and Hash to Obtain Random Subset (HORS) signature with Tree (HORST) signature with Tree (HORST) for few time signature [32]. SPHINCS-256 uses a pseudo-random number generator based on Chacha-12, for generating the HORST secret key. In 2008, Bernstein et al. [34] proposed Chacha, a cryptanalysis-resistant stream cipher algorithm that maps an input stream to a novel and irreversible output stream [35]. Chacha-12 is an extension of Chacha [36].

However, Chailloux et al. [27] proposed an eficient collision search algorithm based on quantum logic gates, which reduces the security level of post-quantum from 2^{128} to $2^{119.6}$. Thus, a quantum attack requires a reduced number of queries, which is on the order of $2^{119.6}$, to crack a post-quantum algorithm. Therefore, it is crucial to strengthen the security of post-quantum hash algorithms. Moreover, the emergence of quantum computing has facilitated the application of quantum physics in resource-constrained devices. For example, researchers at the University of Bristol have introduced the chip-based Quantum Key Distribution (QKD) in 2017, and the ID Quantique company has developed a QRNG Chip and QKD systems as well [37], [38].

SCADA networks are vulnerable to a range of attacks, including Denial of Service (DoS), Eavesdropping, and Man-in-the-Middle attacks, which can result in data injection and the propagation of malware like Irongate. DoS attacks on industrial control systems have targeted the availability and operational functionality of SCADA units. While availability is deemed a top priority in SCADA standards, intrusions often go undetected in SCADA systems, as evidenced by notable incidents such as Stuxnet and Irongate, where the intruders were able to evade intrusion detection systems [39], [40]. Successful malware attacks typically begin by breaching the confidentiality of wireless or wired communication, followed by a Man-in-the-Middle attack that compromises the confidentiality and integrity of SCADA security measures. The vulnerabilities in SCADA standards persist in protocols such as Distributed Network Protocol [41] and Modbus [42], as they lack authorized access and encryption algorithms [43]. AGA-12 provides key management, encryption, and authentication algorithms, however, it lacks an intrusion detection system capable of detecting data injection or malicious trafic. Furthermore, while various researchers have proposed algorithms, they often fail to address all three security priorities comprehensively [2], [44].

1.4 Objective

The primary objective of this thesis is to introduce a comprehensive security framework for safeguarding SCADA network communication against cyber attacks. The existing SCADA standards and protocols are vulnerable to traditional cyber attacks as well as potential future attacks leveraging quantum computing capabilities. To address these challenges, the thesis proposes three modules, each focusing on distinct aspects of SCADA security and making significant contributions. Module 1 introduces an intrusion-resistant SCADA framework incorporating quantum key generation and post-quantum digital signature algorithms to enhance security. Module 2 presents a multi phase quantum resistant framework, utilizing quantum and post quantum cryptography techniques. It includes authentication and key management. Module 3 introduces LARSCOM, a secure key generation algorithm with intrusion detection capabilities, employing a combination of pseudorandom and true random number generators. Encryption is performed using a robust algorithm to ensure integrity and resistance against attacks. These contributions strengthen the security of SCADA networks by providing confidentiality, integrity, and availability in SCADA systems. Formal analysis, security proofs, and simulations were conducted using Python and IBM Quantum. Extensive evaluations were also performed on a dedicated SCADA test bench. This evaluation process allowed for the assessment of the performance and eficiency of the security framework in real-world scenarios, providing valuable insights for further improvements.

1.5 Major Contributions

Figure 1.2 illustrates the major contributions of this Ph.D. work. It has three main modules.

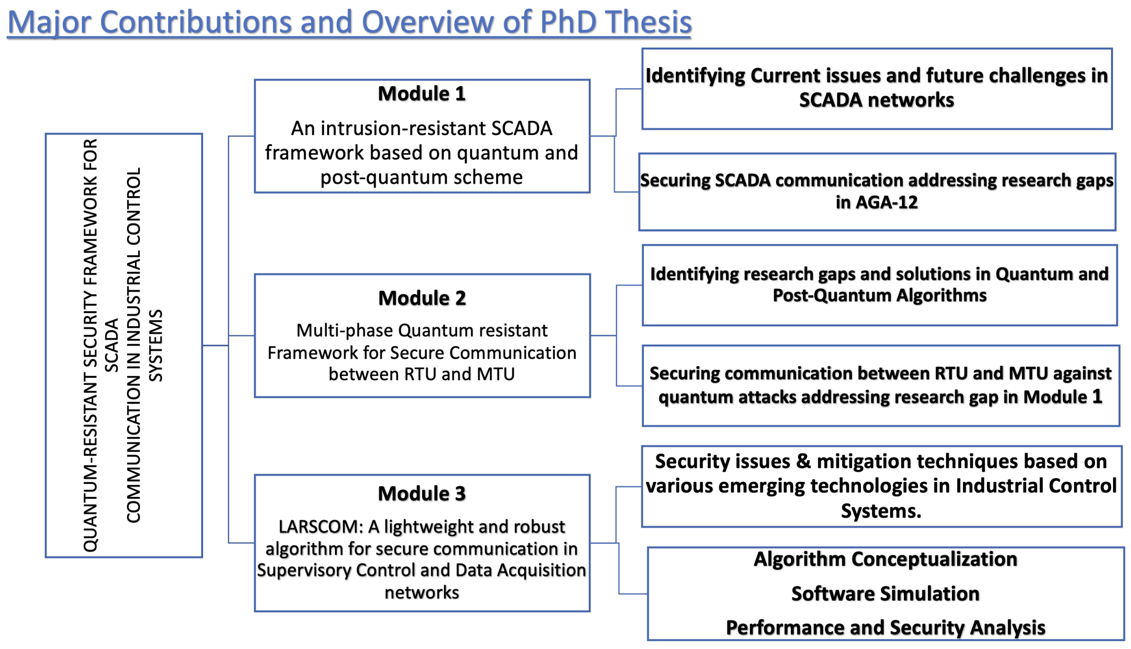

Major Contributions and Overview of PhD Thesis
QUANTUM-RESISTANT SECURITY FRAMEWORK FOR SCADA COMMUNICATION IN INDUSTRIAL CONTROL SYSTEMS
Module 1
An intrusion-resistant SCADA framework based on quantum and post-quantum scheme
Module 2
Multi-phase Quantum resistant Framework for Secure Communication between RTU and MTU
Module 3
LARSCOM: A lightweight and robust algorithm for secure communication in Supervisory Control and Data Acquisition networks
Identifying Current issues and future challenges in SCADA networks
Securing SCADA communication addressing research gaps in AGA-12
Identifying research gaps and solutions in Quantum and Post-Quantum Algorithms
Securing communication between RTU and MTU against quantum attacks addressing research gap in Module 1
Security issues & mitigation techniques based on various emerging technologies in Industrial Control Systems.
Algorithm Conceptualization
Software Simulation
Performance and Security Analysis

Figure 1.2: Major Contributions and Overview of PhD Thesis

1.5.1 Module 1: An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme

Module 1 proposes a novel security scheme for SCADA systems based on quantum and post-quantum schemes with the following contributions:

- This module proposes an intrusion-resistant framework based on quantum and post-quantum security schemes. It utilizes the B92 protocol to generate a secret quantum key for encryption and SPHINCS-256, a preimage and collision-resistant algorithm, to obtain a digital signature with a true random number generator.

- In general, SPHINCS-256 uses a Pseudo-Random Number Generator (PRNG) based on Chacha-12 to generate a secret key of the HORST tree. However, to increase the security of SPHINCS-256, QRNG is introduced to obtain a non-deterministic and truly random HORST secret key.

- SPHINCS-256 sends a message, public key, and digital signature over the public channel. In the proposed scheme, the message is replaced with the cipher obtained in the encryption phase.

1.5.2 Module 2: Multi-phase Quantum resistant Framework for Secure Communication in SCADA Systems

Module 2 proposes an intrusion-resistant SCADA framework based on a Quantum and Post-Quantum Scheme to incorporate a collision and presage resistance to SCADA security.

Table 1.1: State-of-art and Contribution of the proposed secured framework in Module 2.

State-of-the-art	Protocols used	Research Gaps	Author's Contribution
Current Standard (AGA-12)	•RSA •AES •ECDSA based on SHA-1	•Not Quantum resistant •Susceptible to Man-in-the-Middle Attack •SHA-1 already broken by Google	A quantum-resistant security framework that acts as intrusion-detection and provides confidentiality, integrity and authentication.
Quantum and Post Quantum Cryptography	•BBM92 •E91 •SIKE	•BBM92 and E91 lacks authentication •SIKE lacks security in the isogeny computation and has large key size.	•Incorporating hashed message authentication using SIKE shared secret key. •Adding masking based on PRISM to isogeny computation to strengthen the security of the key and reduce the key size.

The contributions of the Module 2 are summarized as follows:

- The proposed scheme generates a secure, shared key between the MTU and RTU that is resistant to quantum attacks. It includes SIKE, a post-quantum cryptographic scheme, to generate the shared secret key. However, as per the NIST report, SIKE needs further optimization of keys and secure isogeny computations. Thus, it involves the modification of SIKE by introducing a masking scheme based on the PRESENT [45] algorithm to compress the keys. The primary advantage of adding masking to SIKE is to increase its security by additional randomness to the computations. It becomes more challenging for an attacker to determine the keys being exchanged and specific side channel attacks that aim to gather information about the system by analyzing factors such as power consumption or electromagnetic emissions [46]. Thus, the compressed-key SIKE version increases the resilience and feasibility of the proposed scheme as per the SCADA protocol requirements. Moreover, it increases the secrecy of the key, which increases the security level of the modified SIKE.

- This module also proposes an authenticated quantum key distribution protocol, generating a session/symmetric key to secure the communication between MTU and RTU. Researchers have proposed several quantum key distribution protocols that provide resistance against quantum attacks based on Shor's algorithm [30]. But, they lack message authentication, authenticating the required communication during the establishment of the session key. In such a scenario, an adversary may have the ability to alter the data exchanged between RTU and MTU. The proposed scheme introduces a QKD protocol named BBM92 protocol which includes a hashed message authentication code based on SHA-3 using a shared secret key generated from modified SIKE to verify the integrity of the session key. Moreover, the randomness of the session key based on the fundamentals of quantum physics provides resistance against quantum attack.

- Further, this module proposes a multi-phase infrastructure by integrating the fundamentals of quantum, classical, and post-quantum schemes. It provides confidentiality to the SCADA communication channel by encrypting the sensitive messages exchanged between the RTU and MTU. It feeds the session key, generated by the BBM92 protocol, into the ASCON encryption algorithm. Thus, it obtains a cipher that provides robustness against classical and quantum attacks.

1.5.3 Module 3 : LARSCOM: A lightweight and robust algorithm for secure communication in Supervisory Control and Data Acquisition networks

Module 3 proposes a lightweight and robust algorithm for secure communication between RTU and MTU, and has the following contributions.

- Module 3 introduces a novel key generation scheme in accordance with SCADA protocol standards to ensure secure communication among MTU, sub-MTUs, and RTUs. The proposed scheme involves the generation of key pairs, namely public and private keys, for each node within the SCADA system. To achieve this, the algorithm incorporates a pseudorandom number generator function and a true random number generator that utilizes time-variant parameters such as the current date and time to introduce randomness into the key pairs. Additionally, the proposed scheme incorporates Huffman coding [47] during the key generation process to reduce its size while preserving key integrity and enabling lossless compression. This approach not only enhances the eficiency of encryption and decryption algorithms but also ensures secure and eficient key management within the SCADA network.

- The proposed scheme incorporates a novel method for establishing a shared secret key through the utilization of key pairs generated within the system. This process involves the application of the radioactive decay law algorithm, as described in the work by Brown et al. [48], to measure the decay or compression rate. By analyzing this rate, the scheme is able to detect any potential intrusions and generate a shared secret key accordingly. This approach enhances the security of the system by utilizing the principles of radioactive decay and enables the establishment of a robust shared secret key for secure communication within the SCADA network.

- The proposed scheme includes the utilization of the ASCON algorithm to encrypt the sensitive data transmitted between the RTU and MTU within the SCADA network. Unlike AES in AGA-12, ASCON is a lightweight authenticated encryption algorithm that generates a tag to verify the integrity of both the cipher and the tag itself. By utilizing ASCON, the scheme ensures the integrity and freshness of the encrypted data, providing resistance against replay attacks and quantum key search attacks [49]. This choice of algorithm enhances the security of the system and safeguards the integrity of the transmitted data within the SCADA network.

- LARSCOM demonstrates resilience against Man-in-the-Middle attacks and provides enhanced robustness and security compared to the existing SCADA security standard, AGA-12. The proposed security framework focuses on confidentiality, integrity, and availability to ensure the security goals of the SCADA system.

1.6 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 describes the related work to the proposed scheme, and describes the current solutions and future directions against quantum attacks. Chapter 3 discusses about the intrusion resistant SCADA framework based on quantum and post-quantum scheme. Chapter 4 involves quantum resistant proposed framework. Chapter 5 describes the lightweight and robust algorithm proposed for SCADA networks. Chapter 6 concludes the remarks on the overall security framework for SCADA communication.

Chapter 2

Background and Related Work

Various researchers and organizations have put forth proposals and advancements to address classical and potential quantum threats [25], [29]. The thesis presents a succinct literature review of pertinent algorithms, classifying the related work into three main categories: the current SCADA standard AGA-12, Quantum Key Distribution (QKD), and Post-Quantum Cryptography (PQC).

2.1 Current SCADA standard AGA-12

Traditional SCADA standards, such as IEC 60870, DNP3, IEC 61850, and Modbus, exhibit several shortcomings in terms of security properties [2]. However, the existing AGA-12 standard surpasses traditional standards by offering enhanced security features [50], [51]. AGA-12 incorporates three key algorithms: AES with a minimum key size of 124 bits, RSA with a minimum key size of 1024 bits, and ECDSA with a minimum key length of 160 bits, alongside SHA-1. These algorithms collectively ensure confidentiality, integrity, and authentication in AGA-12 [1], [2], [50].

However, the key management protocol used in AGA-12 relies on RSA, which has been compromised by Shor's algorithm. Additionally, AES and ECDSA are vulnerable to Grover's quantum search algorithm. Gidney et al. [12] have demonstrated that Shor's algorithm can break RSA-2048 within a mere 8 hours using 20 million qubits. In contrast, a classical search algorithm operates in O(N) time, while a quantum search based on Grover's algorithm operates in $O(\sqrt{N})$ time [11]. Grover's algorithm significantly reduces the brute force attack time to the square root of the original time. Consequently, the attack time on AES-128 is reduced to 2^{64}, and for AES-256, it is reduced to 2^{128}. Moreover, Google has successfully executed a collision attack to break SHA-1. These studies highlight the existing research gaps in addressing confidentiality, integrity, and authentication in the context of quantum computing [11].

2.2 Quantum Key Distribution Protocols

Researchers worldwide have proposed various quantum key distribution protocols to protect cyber-physical systems against quantum computing. The current QKD protocols can be categorized into two, namely, Prepare and Measure (P&M) and Entanglement-based (EB) schemes. Based on the literature review, the thesis concludes that BB84 is the parent node for QKD based on Heisenberg's Uncertainty principle, that is, Prepare and Measure, and it is the oldest QKD protocol. On the other hand, B92, the six-state protocol, and SARG04 are advancements of the BB84 protocol. Moreover, E91 is the parent node for entanglement-based schemes, and BBM92 and DPS protocols are advancements of E91 protocol [52], [53]. BB84 and B92 protocols are theoretically proven to be unconditionally secure. However, both are vulnerable to Photon Number Splitting (PNS) attacks[54]. B92 has two advantages over BB84; 1) B92 has a lower tolerance to noise than BB84, and 2) B92 is less complicated to implement practically than BB84. Besides BB84 and B92, the six-state protocol is another type of Prepare and Measure protocol that uses six polarization states and three bases to measure qubits. The six-state protocol has a higher key generation rate, can transmit qubits over a larger distance and has a higher tolerance to noise than that of BB84 [54]. However, deploying a six-state protocol is challenging as it uses a laser pulse instead of a single photon source to generate qubits. However, it is resistant to PNS attacks but vulnerable to incoherent PNS attacks [52], [53].

E91 protocol is entanglement-based and more secure than BB84 because it discards a higher number of bits. Thus, it has less chance of information leakage. However, the practical implementation of E91 is more challenging than that of BB84. In 1992, there was another entanglement-based protocol name BBM92, which is resistant to PNS and USD attacks that are used to extract the secret key. And then there is DPS(Differential Phase-Shift) which is another type EPR based protocol that is resistant against PNS attack [55]. However, security analysis of entanglement-based QKD protocols is ongoing research and is still in progress. Moreover, in 2004, Nicolas et al. [56] proposed a COW protocol quantum key distribution scheme. It is a coherent one-way quantum key distribution prototype that encodes logical bits in time. It is resistant to PNS attacks, beam-splitting attacks, and intercept-resend attacks. However, security analysis of COW protocol is still in progress [52], [57].

Table 2.1: Comparison of recent widely known QKD protocols

Author	Protocol	Overview	Advantage	Disadvantage
Bennet et al.[58], 1984	BB84	It is based on Heisenberg's Uncertainty Principle. BB84 uses the transmission of single polarized photons (as the quantum states). The polarization of the photons are four, and are grouped together in two different non orthogonal basis[52].	Theoretically BB84 protocol have been proven to provide unconditionally security[52][53].	Lower key rate than SARG04[52][59]. Vulnerable against Photon Number Splitting (PNS) attack[55].
Bennet [60], 1992	B92	The quantum protocol B92 is similar to the BB84 protocol but it uses only two states instead of four states. B92 protocol is also based On Heisenberg's Uncertainty Principle[52][53].	B92 protocol is proven to be unconditional secure[52][61]. The B92 protocol is easier to implement than BB84[61][62].	Channel loss dependent. Vulnerable against PNS attack. Reduced tolerance to noise than that of BB84. Thus, final key sizes are much smaller than that of BB84[61][59][62].
Gisin et al. [63], 1999	Six-State Protocol	This protocol using 6 polarization states and 3 measurement bases[53][61].	Improved version of BB84. Enhanced key generation rate and more tolerance to noise[53][61][59].	The type of detector is demanding in terms of technological level. The squash operator for the six-state does not exist. However, researchers are working on it[53][61][59].
V. Scarani [55], 2004	SARG04	This protocol designed by using attenuated laser pulse as photon source instead of single photon source[53][59][64].	Increased security against PNS attack. Higher key rate than BB84 protocol. Practical length of channel is bigger than that of BB84[53][61][59].	SARG04 is more robust than BB84 against incoherent PNS attacks. However, it is still vulnerable to incoherent PNS attacks[53][61][59].
Ekert [65], 1991	EPR or E91	Unlike BB84 and B92 protocols, this protocol uses Bell's inequality to detect the presence or absence of Eve as a hidden variable. The EPR quantum protocol is a 3-state protocol[52][53][59].	E91 protocol safer, because it discards a higher number of bits. More secured than BB84[59][64].	BB84 generate final key size of 50% of the initial bits where as E91 generates final key size of 22% of initial keys. Practical implementation is more dificult than that of BB84[52][53][59][66].
Bennet et al. [67], 1992	BBM92	Entanglement-based version of BB84 Protocol[52][53][68].	Loss-tolerant. Resistant against PNS and sequential unambiguous state discrimination (USD) attacks[52][53][59].	Security analysis of entanglement based QKD protocols is still the subject of very active research, with recent investigations and simplified proofs based on entanglement distillation protocols[52].
Nicolas et al.[56], 2004	COW	High speed coherent one-way quantum key distribution prototype[69].	A simple high-speed protocol. A high-performance detection. Also, PNS-attack resistant. Beam-splitting attack resistant. Intercept-resend attack resistant[69][62].	Proving the security of the COW protocol remains a work in progress, as per the standard security proof of BB84, B92 and SARG protocol[52][69][62].
K. Inoue et al. [70], 2003	DPS (Differential Phase-Shift)	Entanglement-based QKD protocol [53]	Resistant to photon number splitting (PNS) Attack[[57][71].	Security analysis of entanglement based QKD protocols are still the subject of very active research, with recent investigations and simplified proofs based on entanglement distillation protocols[52][62].

2.3 Post Quantum Crytpography

The emergence of quantum computing has threatened widely known public-key, key management protocols, and authentication schemes dependent on factorization, elliptic curve cryptography, and discrete logarithms. Researchers have developed post-quantum cryptography based on hard mathematical problems that will be highly costly for existing quantum computers to crack. The PQC are divided into five categories, namely, Code-based, Lattice-based, Supersingular elliptic curve isogeny, Multivariate based, and Hash-based cryptography [72], [73]. This thesis briefly reviews the related PQC schemes that address confidentiality.

In 1978, McEllice proposed a cryptographic scheme based on hidden Goppa code [74] that was dependent on modular arithmetic and NP-hard problem of decoding linear codes [75]. As per NIST report [25], it is faster than most cryptosystems, is CCA-resistant, and provides one-wayness CPA resistance. It also generates smaller ciphertexts as compared to other PQCs. However, it generates large public keys, and hence it is not feasible for resource-constrained devices. On the other hand, Hoffstein et al. [76] have developed an Nth Degree Truncated Polynomial Ring Units(NTRU). NTRU is a PQC based on Shortest Vector Problem (SVP) within a lattice and is proven to be quantum-resistant. Moreover, it generates smaller public and private keys as compared to McEllice. Therefore, it is widely used as an alternative to RSA and ECC. However, security analysis of NTRU is still on-going[25].

Lyubashevsky et al. [77] proposed a PQC scheme that is based on the earning with errors problem mapped to polynomial rings over finite fields. It is a flexible algorithm suitable for key management, encryption, and digital scheme. However, the major drawback of the Ring-LWE algorithm is that it generates large public and private keys compared to traditional cryptography. Hence, it is not feasible for resource-constrained devices [25].

In [78], Feo et al. proposed a Supersingular isogeny Difie–Hellman key exchange (SIDH) that is based on Supersingular Elliptic Curve Isogeny Cryptography. It depends on the complexity of finding isogenies in the supersingular elliptic curves. Supersingular Isogeny Key Encapsulation(SIKE) extends SIDH, which NIST sees as a potential candidate for future standardization. Both SIDH and SIKE offer small key sizes, and ciphertext sizes that require optimization and further compression techniques[25], [79]. It also provides smaller public keys as compared to that other PQCs. They also provide resistance against quantum and classical attacks. Thus, they are suitable for resource-constrained devices [25].

Existing post-quantum cryptography schemes depend on the hypothesis that current quantum computers do not have enough computation resources to crack the hard problems or isogeny computations that the PQCs are dependent on. A generic PQC provides a security level of 2^{128}. In 2017, Chailloux et al. [27] proved an eficient Grover's quantum search algorithm that dipped the security level of PQC from 2^{128} to $2^{119.6}$. Thus, entirely relying on PQC schemes is currently safe. However, PQC does face a potential threat by quantum computing.

The emergence of quantum computing in recent years has brought both benefits and risks to stable cyber-physical systems [24]. The most threatened cryptosystems are widely used public-key algorithms, key management schemes, and digital signatures dependent on factorization, elliptic curve cryptography and discrete logarithms [80]. This has motivated researchers to focus on the study and development of post-quantum cryptography. They are resistant against quantum as well as classical attacks and can be deployed in the existing network infrastructure [80]. Currently, there are five categories of primary post-quantum cryptography, namely; [81], [31]

- Code based cryptography [81], [31]

- Lattice based cryptography [81], [31]

- Supersingular elliptic curve isogeny [81], [31]

- Multivariate based cryptography [81], [31]

- Hash based cryptography [81], [31].

This thesis divides the Post Quantum Cryptography into two security goals: Confidentiality and Integrity. The following sections provide post-quantum cryptography based on the type of algorithms used.

2.4 Existing Post-Quantum Security Schemes addressing Confidentiality

2.4.1 Code based Cryptography

Code-based cryptography are cryptosystems, including symmetric and asymmetric, based on dificulty of error correcting codes [31]. It can be categorised into the following.

- Public-key encryption
- Digital signature
- Zero-knowledge protocols
- Pseudo-random number generator and stream cipher
- Hash functions

This thesis reviews the code-based public key cryptosystem as discussed in the below subsection. The most commonly known algorithm is McEllice cryptosystem [80].

McEllice cryptosystem

In 1978,a public key encryption scheme that was based on hidden Goppa code [74] was proposed by Robert McEllice. Goppa codes are the relation between algebraic geometry and codes and are used as error-correcting code. They rely on the NP-hard problem of decoding linear codes. The basic concept of Goppa code depends on modular arithmetic. When a number series approaches a higher number, and once it reaches a specific number, the series starts from 0 again [31, 80]. A classic McEllice cryptosystem includes the following phases (assuming Alice and Bob are the two legitimate participants) [74, 82].

Key Generation: Bob selects a Goppa polynomial g(z) of degree t, compute its corresponding generator matrix G, selects a random invertible matrix denoted as S, and a random permutation matrix denotes as P. Bob uses all the parameters to compute $G^{'} = SGP$, and announces his public key that includes $(G^{'}, t)$. Bob's private key includes (S,G,P).

Encryption: Alice encrypts her message, represented in binary strings, by selecting and combining a random error vector, e, that has weight $\leq$ t, to the mG'. Thus, Alice sends the following cipher, y in equation 2.1.

$$y = m \times G' + e \tag{2.1}$$

Decryption: Bob uses his matrix, P to derive y' as shown in equation 2.2. Then, Bob applies the decoding algorithm, computing e', to y' to correct the errors, and derive the codeword, m'. The m' is mS. Thus, Bob can easily derive m by $m' * S^{-1}$.

$$y' = y \times P^{-1} \tag{2.2}$$

According to the National Institute of Standards and Technology, Post-Quantum Cryptography(PQC) Standardization report [80], McEllice cryptosystem is suitable for replacing traditional cryptosystems. It is quantum-safe and provides security against Chosen-Ciphertext Attack (CCA) and One-Wayness against Chosen-Plaintext Attack (OW-CPA). Moreover, McEllice cryptosystem is faster than most cryptosystems. However, it also comes with its drawbacks. It generates large public keys that can cause an implementation problem, especially in resource-constrained devices. Moreover, the ciphertexts are smaller than other post-quantum Key Encapsulation Mechanisms(KEMS), but the ciphertexts are larger than the plaintexts. McEllice cryptosystem is not applicable to generate authentication scheme or digital signature scheme [80].

2.4.2 Lattice based cryptography

Lattice-based cryptography have been proven to be strongly resistant to sub-exponential as well as quantum threats. They are based on the concept of lattices, sets of points within an n-size periodic structured space as shown in Figure 2.1 [83]. In simple terms, lattice can be considered as any regularly spaced grid of points. The security of the lattice based cryptography depends on the complexity of lattice problems, mainly, Shortest Vector Problem (SVP), Closest Vector Problem (CVP), or Shortest Independent Vectors Problem (SIVP) [84], [83]. The SVP is deriving the minimum non-zero vector in the current lattice and is an NP-hard problem unsolvable by the present quantum algorithm[84].

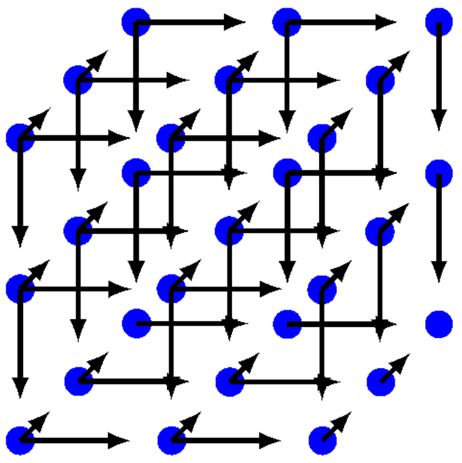


Figure 2.1: A lattice space in 3D

NTRU

In 1996, Hoffstein et al. [76] published an encryption scheme called the Nth Degree Truncated Polynomial Ring Units(NTRU). It is a public-key cryptography focused on SVP within a lattice [76]. It does not rely on factorization or disjunctive logarithms as the traditional public-key cryptosystem does. The basic NTRU operation is performed in the truncated polynomial rings as shown in equation 2.3, such that N as a prime number and Z_q is the ring of integers modulo q [76].

$$R_q = \frac{Z_q[X]}{X^N - 1} \tag{2.3}$$

A polynomial, f, in the R_q can be written in equation 2.4.

$$f = [f_0, f_1, \ldots., f_{N-1}] = \sum_{k=0}^{N-1} f_k X^k \tag{2.4}$$

Moreover, the multiplication is denoted as *. Thus, f * g, where f and g are two polynomials, is given as a cyclic convolution product as shown in equation 2.5.

$$f * g = h = [h_0, h_1, \ldots., h_{N-1}] \tag{2.5a}$$

$$h_k = \sum_{i+j = k \bmod N} f_i g_j \tag{2.5b}$$

The NTRU uses the above parameters to derive key pairs, encrypt the message and then, decrypt the cipher. Thus, the thesis lists the public parameters of the NTRU algorithm as follows [76],[82].

- N denotes a large prime number.
- p and q are positive numbers. Gcd(p,q) = 1, and, $p \square q$.
- d_f, d_g and d_r are integers to generate polynomials. The polynomials from which the private keys are uniformly chosen, belongs to the set $B(d_f)$, $B(d_g)$ and $B(d_r)$. The binding values in $B(d_r)$ is used during encryption.

Key Generation: Two random small polynomials, f and g, are selected, such that $f \epsilon B(d_f)$ and $g \epsilon B(d_g)$, $f_p = f^{-1}(\text{mod } p)$ and $f_q = f^{-1}(\text{mod } q)$. Then h is computed. Thus, the obtained public key is (N,h) and the private key is (f,f_q) [76], [82].

Encryption: To encrypt a message, m, a polynomial r is chosen randomly such that $r \epsilon B(d_r)$. Then, the message is encrypted to generate a cipher e as shown in equation 2.6[76][82].

$$e = p * r * h + m \ (\text{mod})\ q. \tag{2.6}$$

Decryption: The first step of decryption is to compute f * e (mod q), and transform the obtained value, a to polynomial whose coeficients are in the range [-q/2, q/2] . The following equations are used during decryption. The value of m can be derived from f_p * a (mod p)[76][82].

$$a = f * e \ (\text{mod } q) \tag{2.7}$$

$$=> a = f * (p * r * h + m) \ (\text{mod } q) \tag{2.8}$$

$$=> a = f * p * r * g * f_q + f * m \ (\text{mod } q) \tag{2.9}$$

$$=> a = p * r * g + f * m \ (\text{mod } q) \tag{2.10}$$

As per the NIST report [80], the NTRU remains unbreakable against current attacks as well as attacks using present quantum hardware. Moreover, it generates shorter key pairs as compared to McEliece's cryptosystem. Thus, NTRU is declared a suitable alternative to RSA and ECC.

Ring-LWE

Ring-LWE [77] is a post-quantum cryptosystem that relies on the Learning with Errors(LWE) problem assigned to polynomial rings over finite fields. In Ring-LWE, the coeficients of polynomials can be added and multiplied within a finite field, F*q, such that the coeficients are less than q [77, 82]. The Ring-LWE can be deduced to a SVP within a lattice.

A classic Ring-LWE follows the following steps, assuming Alice and Bob are the two participants [77].

1. Alice and Bob accord on a shared complexity value of n, such n such that n is the highest co-eficient power.

2. They both derive q such that $q = 2^n-1$.

3. The polynomial operations are computed with modulus of q.

4. Alice creates a set of polynomial values, A as in equation 2.11.

$$A = a_{n-1}x^{n-1} + ... + a_1x^2 + a_1x + a_0 \tag{2.11}$$

5. Alice divides A by $\varphi(x) = x^n + 1$.

6. Alice computes two polynomials: error polynomial (e) and secret polynomial (s).

$$e\_A = e_{n-1}x^{n-1} + ...e_1x + e_0 \tag{2.12}$$

$$s\_A = s_{n-1}x^{n-1} + ...s_1x + s_0 \tag{2.13}$$

7. Alice, then, computes v_A using A, e_A and, s_A as shown in equation 2.14.

$$v_A = A \square s_A + e_A \tag{2.14}$$

8. Alice sends A and v_A to Bob. And, Bob follows the same algorithm to generate his own error polynomial e_B and secret polynomial s_B.

9. Bob also creates v_B and sends it to Alice.

10. Alice multiplies the v_B with her own secret polynomial, and further computes it as shown in the equation 2.15.

$$\text{shared}_A = \frac{v_B \,\square\, s\_A}{x^n + 1} \tag{2.15}$$

$$\text{shared}_B = \frac{v_A \,\square\, s\_B}{x^n + 1} \tag{2.16}$$

11. In the meantime, Bob uses the same algorithm to generate its own shared secret key as in equation 2.16.

Then, both Alice and Bob extract the noise from the shared secret key to obtain the same shared secret value [77]. This algorithm can be implemented in the key-exchange scheme, digital signature, and homomorphic encryption, and the public and private key sizes are larger than that of a traditional public-key cryptosystem [77]. One of the post-quantum algorithms focuses on Ring-LWE is called New Hope [80].

2.4.3 Supersingular Elliptic Curve Isogeny based cryptography

Supersingular Elliptic Curve Isogeny Cryptography is quantum-safe cryptography that depends on the hardness to find isogenies among the supersingular elliptic curves. Feo et al. [85] developed a post-quantum ography algorithm, Supersingular isogeny Difie–Hellman key exchange (SIDH), analogous to Difie-Hellman, and is based on Supersingular isogeny problem defined as following [86, 87]. [86, 87] Given two super singular elliptic curves, E_1, E_2, in a finite field k, where $|E_1| = | E_2 |$, solve an isogeny function $f : E_1 \dashrightarrow E_2$.

Public Parameters: The public parameters are, first, agreed upon by both participants(Alice, Bob) during the SIDH protocol. Alice and Bob establishes a large prime p based on an super-singular elliptic curve, denoted as E, over a field, F_{p^2} such that F refers to the set of integers modulo, p, and E can represented as following[86, 87].

$$E : y^2 = x^3 + ax + b \tag{2.17}$$

$$p = 2^a + 3^b - 1, \tag{2.18}$$

where a and b $\in$ N. Moreover, they also agree on the basis P_A, Q_A of $E[2^A]$ and P_B, Q_B of $E[2^B]$.

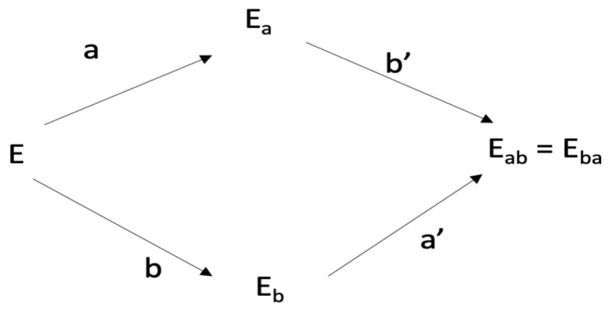


Figure 2.2: Isogeny computation by Alice and Bob to obtain the common shared secret key.

Key Generation: Alice selects a number, r_A randomly extracted from the set $0 \leq r_A < 2^b$-1. She uses her basis and the selected random number to compute the kernel to generate her isogeny function φ_A. Alice, using her isogeny function with ker φ_A (equation 2.19), generates the public key denoted as $E_A = \varphi_A(E), \varphi_A(P_B), \varphi_A(P_B)$. Her private key is r_A [86, 87].

$$\ker \varphi_A = \langle P_A + r_A Q_A \rangle \tag{2.19}$$

Parallely, Bob follows the same algorithm to generate his own key pairs. His private key is r_B randomly extracted from the set $0 \leq r_A < 3^b$-1. His public key is her public key denoted as $E_B = \varphi_B(E), \varphi_B(P_A), \varphi_B(P_A)$.

Shared Secret Key Generation: The public keys, E_A, E_B, are exchanged by Alice and Bob. Alice generates another isogeny function, ψ_A with kernel function based on the basis parameters, P_A, Q_B as described in equation 2.20[86, 87].

$$\ker(\psi_A) = \langle \varphi_B(P_A) + [r_A] \cdot \varphi_B(Q_A) \rangle \tag{2.20}$$

Alice then generates j-invariant of the curve $E_{AB} = \psi_A(E_B)$. Similarly, Bob calculates ψ_B and then proceeds to derive the j-invariant of curve $E_{BA} = \psi_B(E_A)$. The j-invariant derived by both Alice and Bob are the shared secret key as described in equation 2.21 and, is simply described in Figure 2.2 [86, 87].

$$j(\varphi_B(E_A) = j(E_{BA}) = j(E_{AB} = j(\varphi_A(E_B). \tag{2.21}$$

The post-quantum scheme that relies on the complexity of Supersingular isogeny curve problem, provides enough resistant against attack using quantum algoritm. A key encapsulation mechanism based on SIDH, named Supersingular Isogeny-Based

Key (SIKE), provides the smallest public key sizes as that of other post-quantum schemes, as well as small ciphertexts. However, the performance of the supersingular isogeny-based systems is lower than that of other post-quantum cryptosystems. Thus, it requires eficient optimization algorithms to increase the performance [80].

2.5 Existing Post-Quantum Security Schemes addressing Integrity

In this section, the thesis provides an overview of the types of post-quantum cryptography addressing integrity as a security goal. Currently, three types of post-quantum cryptography implement a signature scheme, mainly, Lattice-based, Multi-variate based, and Hash-based cryptography.

2.5.1 Lattice-based cryptography

The widely used and known lattice-based cryptographies are NTRUSign and BLISS. Lattice-based signatures mainly depend on the NP-hardness of the short vectors in lattice spaces [82]. The following subsections describe the NTRU Signature and BLISS.

NTRU-based public-key cryptography

NTRU Signature [88] is a signature scheme that follows after the NTRU encryption algorithm to provide authentication to the encrypted message. Like NTRU encryption, its security relies on the dificulty of solving the SVP. The basic operation of NTRU Signature occurs in the ring of polynomials, R, of degree less than N-1 represented as in equation 2.3. The NTRU Signature has three phases, mainly, Key Generation, Signature, and Verification [88]. The parameters used in the NTRU sign are as following [88].

- Integer Parameters: (N, p, q, D_{min}, D_{max}). The p and q are relatively prime, and N denotes a large prime number. D_{min}, D_{max} are the deviations caused by the reduction modulo function.

- Set of polynomials: F_f , F_g F_w, F_m.

Key Generation: Bob selects two polynomials, denoted as, f and g. They are further represented as $f = f_0 + p \cdot f_1$ and $g = g_0 + p \cdot g_1$, such that f_0 and g_0 are fixed universal polynomials, and, $f_1 \in F_f$ and $g_1 \in F_g$. Bob also computes the inverse of f modulo q and thus, deriving the key pairs. The public key h is represented as equation 2.22. The pair (f,g) is the obtained private key.

$$h \equiv f^{-1} \ast g \bmod q \tag{2.22}$$

Signing: Bob selects a polynomial w, such that $w \in F_w$. And, w can be represented as in equation 2.23.

$$w = m + w_1 + p \cdot w_2 \tag{2.23}$$

In equation 2.23, w_1 and w_2 are small polynomials. The w was further used to compute the signature, s in equation 2.24. The final signature includes the set (m,s).

$$s \equiv f \ast w \ (\bmod q) \tag{2.24}$$

Verification: Alice now verifies the Bob's signature, (m,s). Alice, first, verifies the signature is null or not. Alice first tests if the deviation satisfies, and then proceeds to use Bob's public key denoted as h. The h and the polynomial, t is further computed as described in equation 2.25. She then further verifies the deviation of t as well.

$$t \equiv h \ast s (\bmod q) \tag{2.25}$$

The performance of NTRU SIgn is similar to NTRU Encrypt, as it provides almost similar public and private key sizes as compared to RSA and ECC, and, provides higher performance and higher security level against traditional and quantum attacks. Thus, it is suitable for the current network infrastructure [80].

BLISS

BLISS[89] is a signature generating algorithm whose operation relies on the ring R_q with a power of 2 where q is the prime number as represented in equation 2.26 [82, 90]. The $x^n + 1$ has a root in Z_q. Moreover, $q = 1 \bmod 2n$. It also has three phases key generation, signing, and verifying as follows [89, 90].

$$R_q = \frac{Z_q[X]}{X^n + 1} \tag{2.26}$$

Key Generation: Thus phase denotes two polynomials as f and g with d_1 coeficients in $\{\pm 1\}$ and d_2 coeficients in $\{\pm 2\}$, where $d_1 = \delta_1 n$ and $d_2 = \delta_2 n$. The private key, $S \in R_{2q}^2$, and the public key $A \in R_{2q}^2$ are generated by computing the following equation.

$$S = (s_1, s_2) = (f, 2g + 1) \tag{2.27}$$

$$a_q = \frac{s_2}{s_1} \bmod q \tag{2.28}$$

$$A = (2a_q, q - 2) \tag{2.29}$$

Signing:The message to be signed is denoted as μ from the message space, M. At first, two polynomials y_1 and y_2 are selected from a disjunctive Gaussian Distribution D_σ, where σ is the width parameter of the distribution. And, y is a set of (y1,y2). Then, using a Hash function, H, a challenge variable is generated as described as following.

$$c = H(A \cdot y, \mu) \tag{2.30}$$

$$H : R \times M \dashrightarrow R \tag{2.31}$$

The Signing involves Greedy Approximation algorithm that produces a variable v, where v = Sc′ for c′= c mod 2. Then, it selects a bit value, b, such that b ϵ {0,1} to further compute the signature sign = y + bv.

Verification: The receiver receives the sign and c and is verified if it is smaller than the discrete Gaussian parameter that has a width variable σ. If the sign satisfies the condition, the receiver further verifies the equation 2.32.

$$H(az + qc \bmod 2q, \mu) \overset{?}{=} c \tag{2.32}$$

BLISS is a lattice-focused signature algorithm[82, 90]. It generates key and signature size similar to RSA algorithm, and is resistant against quantum attack [82, 90].

2.5.2 Multivariate Cryptography

A multivariate public key cryptosystem is dependent on the NP-hardness of deciphering the multivariate polynomials over the finite fields [83, 91]. The NIST report claims that various multivariate public cryptosystems have been proposed. However, some of them are broken[80]. The class of trapdoor one way functions is an integral property of PKC. For example, NTRU depends on the lattice structure, and ECC depends on the elliptic curve group. Multivariate Cryptography depends on the on-way function as a multivariate quadratic polynomial public map over a finite field[80, 83, 91]. It denotes the set of quadratic polynomials as $P = (p_1(w1, ...w_n), ... ,p_m(w_1, ...w_n))$, where each p_i is a quadratic polynomial in $w = (w_1, ...w_n)$[80, 83, 91]. One of the multivariate cryptography is Rainbow that has been selected as Round 3 finalists by NIST [80].

Rainbow

In 2004, Ding et al. [92] developed a multivariate post quantum signature focused on the Oil-Vinegar signature scheme. It's security relies on the NP-hardness of solving a set of the random multivariate quadratic scheme. Like any other signature scheme, it has three phases, including key generation, signature, and verification as following [91, 92].

Key Generation: Two keys are generated in this phase. The private key includes two invertible afine maps, L_1 and L_2. It also includes the map, denoted as, F. Moreover, the public key consist of the field, referred as, K as well as the composed map, P(x) [91, 92].

Signing: In signature generation phase, given a document $d \in \{0,1\}_{?}$, the sender uses a hash function, h = h(d). Then, it further computes it further, as shown in the following equations, to generate the signature, z.

$$x = L_1^{-1}(h) \tag{2.33}$$

$$y = F^{-1}(x) \tag{2.34}$$

$$z = L_2^{-1}(y) \tag{2.35}$$

Verification: The receiver computes the hash of the composite map on z, where h' = P(z), and then further computes the hash of the document. If the generated hash matches with the received hash, the signature is validated and accepted.

Rainbow offers shorter signatures, only 258 bits for the NIST level 1 security compared to other post-quantum signature schemes. Moreover, the algorithm used in signing the document and verifying the signature is highly eficient and faster than other post-quantum schemes. One drawback of the Rainbow algorithm is that the key generation process is slow and needs to be more eficient [80].

2.5.3 Hash-based Signature Scheme (HSS)

HSS schemes are signature generating algorithms that relies on cryptographic hash functions addressing at least one of the following security properties: pre-image , second-preimage, and collision resistance [81] [93]. The hash-based signature scheme can be further classified into two, mainly, stateless and stateful schemes. A stateful hash-based signature scheme relies on Merkle's tree using OTS parameters, whereas a stateless hash-based signature scheme includes a hyper-tree with both a one-time Signature scheme (OTS) and a Few Time Signature schemes (FTS) [94].

Stateful Signature Scheme

Lamport Signature: In 1970, Leslie Lamport [95] proposed a one-time signature algorithm which is resistant against traditional as well as quantum attacks. The primary advantage of Lamport signature is that it exploits a secure cryptographic hash functions to derive a public key, which is further, processed to sign a message[95]. Thus, the security of Lamport primarily depends on the secrecy of the hash function [95]. Moreover, Lamport signature generates large key sizes as well as signature sizes. Thus, it is unsuitable for almost any network infrastructure. The public key and private key(of size 256 bit) of Lamport signature can be represented as following [96].

$$\text{Private Key} = (x_0, y_0, x_1, y_1 \ldots, x_{255}, y_{255}) \tag{2.36}$$

$$\text{Public Key} = [h(x_0) \square h(y_0) \square h(x_1) \square h(y_1) \ldots. \square h(x_{255}) \square h(y_{255})] \tag{2.37}$$

Winternitz One-time Signature Scheme(WOTSS): To address the large key sizes of the Lamport signature scheme(LSS), WOTSS is proposed. The primary idea of WOTSS is to implement a certain count of chain of functions that start from feeding on random inputs[96]. In WOTS, the random data are the secret key, and the public key includes the output derived from the chains[96]. A message is signed by mapping it to one of the intermediate values of each chain. WOTS is an optimized version of LSS that uses a parameter, w. The Winternitz parameter, w, is inversely proportional to the signature size. A larger w generates a smaller signature. Thus, WOTSS is suitable for memory-constrained devices. However, the time complexity increases exponentially as w increases [96].

Merkle's Signature Scheme: To address the drawbacks of One Time Signature (OTS), Ralph Merkle proposed an algorithm named Merkle Signature Scheme(MSS). It merges various OTS key pairs and obtains multiple concatenated key pairs into a single binary hash tree [96]. During the tree construction, the signature keeps concatenating the string of intermediate nodes with respect to the tree root to generate the authentication path. The authentication path verifies the signature and generates the path of the tree [96]. A simple Merkle's tree is a binary tree with each node is a hash of its following child node. Thus, the root of the tree is considered to be the final public key, and the Merkle tree leaves are the hashes of the OTS public key. Figure 2.3 illustrates a simple Merkle Tree [96].

The OTS signatures generate large public keys, and it needs to generate a novel public key every time a message needs to be sent. Thus, it increases the computation cost [96]. The MSS obtains a public key for signing multiple messages, such that, the frequency of messages must be a power of 2[96]. Given $M = 2^n$, it generates the public key, X_i and private key, Y_i such that Y_i is within the interval $1 \leq i \leq 2n$, where i = n being the root level of the tree [96].

HORS: Reyzin et al. [97] proposed a Few Time Signature(FTS) scheme, using hash functions, that generates a secret key that contains n random numbers generated from a pseudo random number function. The public key is derived from computing the n hashes of the random elements in the secret key. The signature generated contains k secret key values [97].

$$m = k\log n = k\tau \tag{2.38}$$

The relationship between the message (m), public key and secret key values are shown in equation 2.38 , where $k \in N$ and $n = 2^\tau$ for $\tau \in N$ [97].

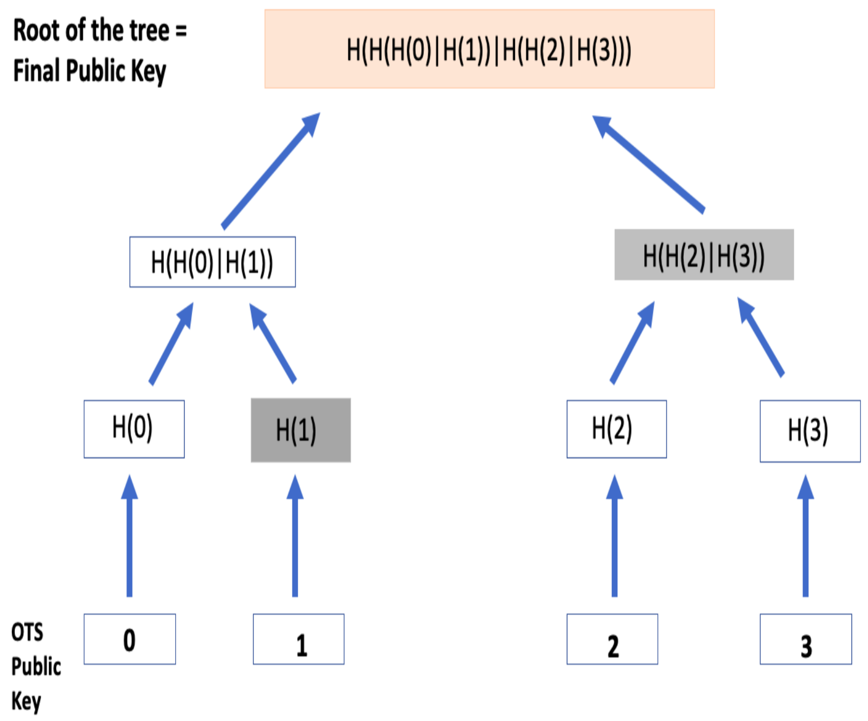


Figure 2.3: A simple illustration of Merkle Tree. The gray shaded nodes are authentication path, and the root of the tree is the public key.

Stateless Signature scheme

SPHINCS: It is a stateless signature scheme that is quantum resistant and its general construction involves hyper-tree with height h and d layers. Each intermediate tree is a Merkle tree with height $\frac{h}{d}$. The security parameters of a SPHINCS tree are, mainly, n refers to the HORST and WOTS hash bit sizes, m refers to the bit-length of message, h refers to the height of the hyper-tree, d refers to the layers of the hyper-tree, w is the WOTS parameter, t is the frequency of HORST secret key elements, k is the count of published HORST secret key elements [32, 98].

A general construction of a SPHINCS tree has the four following trees and has been shown in Figure 3.4[32, 98].

- The Hyper tree: The hyper tree is the main tree that generates the root as the public key. It has height denoted as h. The hyper-tree is further segmented to d-layers of type-2 tree. The hyper tree leaves are the instances of the type-4 trees, HORST tree. For example in SPHINCS-256, the h is 60 and the d is 12 [32, 98].

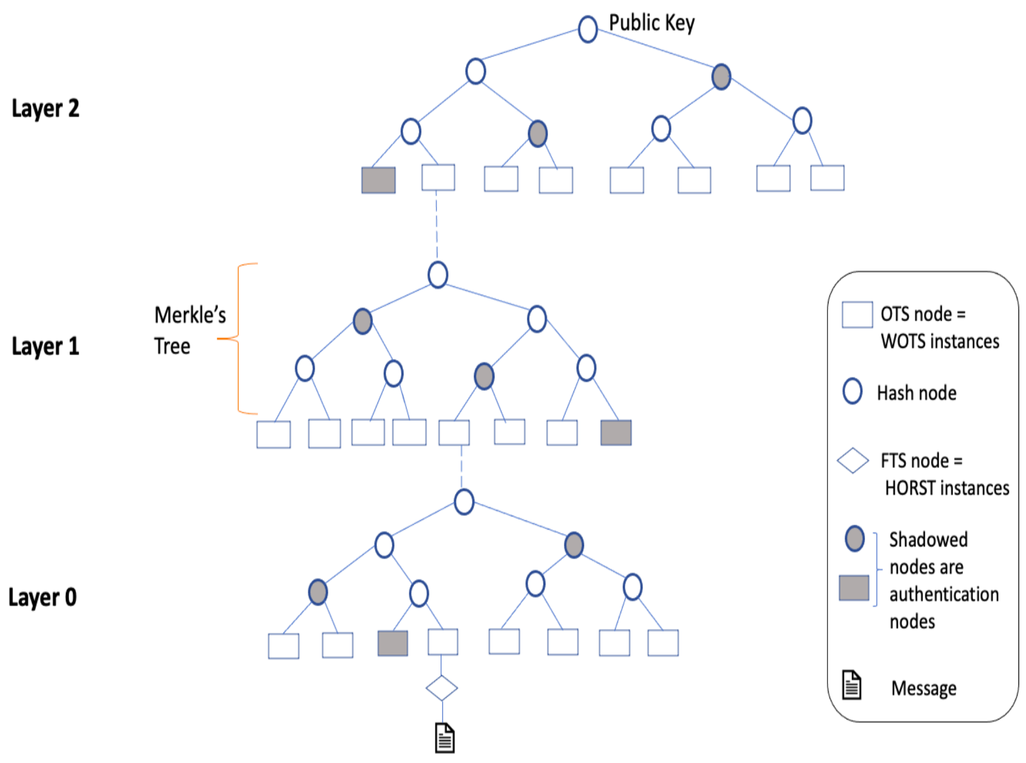

Public Key
Layer 2
Layer 1
Merkle's Tree
Layer 0
OTS node = WOTS instances
Hash node
FTS node = HORST instances
Shadowed nodes are authentication nodes
Message

Figure 2.4: General SPHINCS tree construction with h=9 and d=3.

- The sub trees: The sub-trees are the intermediate tress that is based on Merkle trees and have a height of $\frac{h}{d}$. The leaves of the sub-trees are the root of the type-3 trees; that is, the roots are compressed WOTS public keys that feed as the leaves to the next layer's tree.

- WOTS public key compression tree: They are known as L-trees of height $\lceil \log_2 \ell \rceil$, where ℓ is the count of leaves. The leaves in the sub-trees are derived from the WOTS public keys by computing an unbalanced binary tree that has l leaf nodes, known as L-trees [32, 98].

- HORS public key compression tree: In 2015, Bernstein et al.[32] proposed HORS tree (HORST) for implementing at the lowest level of the SPHINCS tree as the FTS. The bottom layer of the hyper-tree also contains the Merkle tree of height $\tau = \log_2 t$, such that t is the count if HORST public key instances. Unlike stateful hash-based schemes, the stateless signature scheme does not keep a history of used key pairs. Thus, SPHINCS uses multiple HORST key pairs to correctly uses key pairs for more than once [32, 98].

SPHINCS' proof of security solely depends on the underlying hash function. As per NIST [80], SPHINCS provides robust security against traditional and quantum algorithms. Thus, NIST declared SPHINCS to be the Round 3 finalist. Moreover, the public keys of SPHINCS are small enough. However, the signatures are large, and the signature generator process is slow. Thus, SPHINCS requires optimization techniques to increase the speed such that it is well-suitable for memory constrained devices [80].

Table 2.2: Comparison of Post Quantum Cryptography(PQC) algorithms.

Category	PQC Algorithms		Advantages	Disadvantages	Examples
Addressing Confidentiality	Code-based encryption		Strong proof of security Fast encryption. Ciphertext size is small.	Public keys are large	McEllice cryptosystem
	Lattice-based encryption		Ciphertext size is short. Public and private keys are small. Fast encryption process.	Need more understanding of the security. Further security analysis is required.	NTRU Encrypt Ring-LWE
	Super singular Elliptic curve Isogeny		Generates the smallest public key sizes of all PQC schemes. Generates small ciphertexts.	Computation cost is high Optimization techniques are needed.	SIDH SIKE
Addressing Integrity	Lattice based signature		Generates short public and private keys. Fast signature generation.	Need more understanding of the security. Further security analysis is required.	NTRU Sign BLISS
	Multivariate based signature		Generate short signatures.	Need more understanding of the security. Further security analysis is required.	Rainbow
	Hash based signature	Stateful	Smaller signature size. Faster signature generation.	Need maintenance in usage of non-repeated key pairs	Lamport signature WOTS MSS HORS
		Stateless	Do not need to monitor of non-repeated key pairs usage.	Larger signature size. Signature generation process is slower.	SPHINCS

Table 2.3: Comparison of Post Quantum Cryptography(PQC) algorithms that addresses confidentiality

PQC Encryption Algorithm	Overview	Advantage	Disadvantage	NIST Round 3 Finalist	Research Gap
McEllice Cryptosystem	It is based on hidden Goppa code	Faster than most cryptosystems CCA-resistant and one-wayness CPA resistant. Smaller ciphertexts than that of other PQCs.	Large public keys. Not suitable for resource constrained devices.	Yes	Compression Technique needed.
NTRU Encrypt	It is based on the hardness of SVP within a lattice.	Quantum resistant. Smaller public and private keys than McEllice. Suitable alternative to RSA and ECC.	Need more understanding of the security.	Yes	Further security analysis is required.
Ring-LWE	It relies on the Learning with errors problem referred to rings of polynomials over finite fields.	Versatile algorithm. Can be implemented as key management, Digital Signatures and encryption scheme.	Public and private keys are larger than that of traditional cryptography.	No	Compression algorithm is required.
SIDH based SIKE	It focuses on hardness to find isogenies among super singular elliptic curves.	Strong security against quantum and classical attacks. Smallest public keys of all PQCs. Generates small ciphertexts as well. Suitable for resource-constrained devices.	Performance is low.	Yes	Optimization is required to increase the eficiency.

Chapter 3

Module 1: An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme

The research work reported in this chapter has resulted in the following publications:

1. S. Ghosh, S. Sampalli, A survey of security in SCADA networks: Current issues and future challenges, IEEE Access, 2019 ;7:135812-31, doi: 10.1109/ ACCESS.2019.292644 (Impact Factor: 4.43)

2. S. Ghosh, M. Zaman, G. Sakauye, S. Sampalli, An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme, MDPI Appl. Sci. 2021, 11, 2082. https://doi.org/10.3390/app1105208. (Impact Factor: 2.7)

3.1 Summary of the Chapter

Based on the literature survey of current SCADA standards, specifically AGA-12, as well as quantum key distribution protocols and post-quantum digital signature schemes, the thesis addresses and categorizes the following research problems into two distinct sets.

Set 1 focuses on the research problem of AGA-12.

- Does AGA-12 provide confidentiality, integrity and authentication against quantum attacks?

- Does AGA-12 key management provide resistance against Shor's algorithm?

- Will AGA-12 encryption and digital signature provide enough resistance against Grover's algorithm?

Set 2 delves into the research problem pertaining to post-quantum digital signature schemes, specifically focusing on SPHINCS-256. The proposed scheme addresses the research questions raised in this context.

A significant contribution to the field came from Chailloux et al. [27] in 2017, where they introduced an eficient quantum collision search algorithm and explored its implications on symmetric cryptography. Their work demonstrated that this algorithm reduces the security level of post-quantum algorithms from 2^{128} to $2^{119.6}$. The implications of this finding are crucial to consider when designing the security scheme addressed in this thesis.

- Can the security level of SPHINCS-256 be increased?
- Can a post-quantum algorithm use the laws of quantum physics to increase security?

The proposed scheme is developed based on the research questions outlined in the two sets. It introduces a collision-resistant framework for SCADA systems that utilizes a combination of quantum and post-quantum algorithms to ensure both confidentiality and integrity against traditional and quantum attacks.

To address the research questions in Set 1, the thesis presents a comprehensive algorithm consisting of three key phases: key generation, encryption, and authentication. For key generation and encryption, the B92 protocol is adopted, while SPHINCS-256 is employed for integrity and authentication. Additionally, in order to tackle the research questions in Set 2, modifications are made to the SPHINCS-256 algorithm. This algorithm comprises four main components: a quantum random number generator (QRNG) is utilized to generate a truly random HORST secret key instead of a pseudo-random number generator (PRNG); hashing functions are employed for WOTS and HORS public-key generation; tree-based hashing is utilized; and the Chacha Permutation is incorporated in the trees.

In the proposed algorithm, the sender transmits a combination of the signature, cipher, and SPHINCS-256 public key to the receiver, instead of the message itself. Figure 3.1 provides a concise overview of the sequential steps involved in the proposed scheme.

In the following sections, the thesis presents an overview of the proposed scheme.

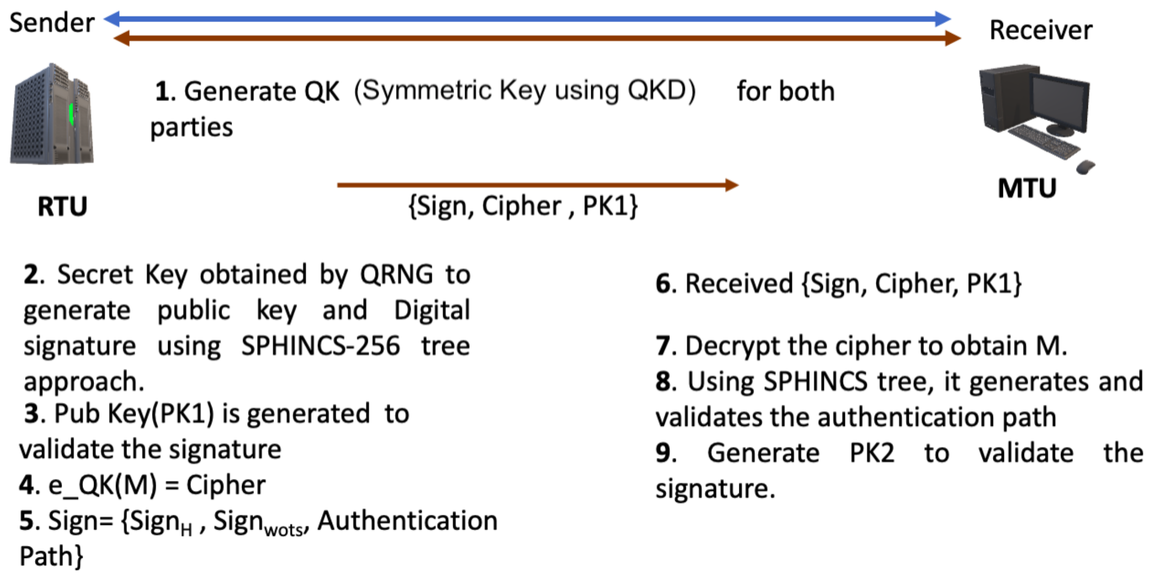

Sender
Receiver
1. Generate QK (Symmetric Key using QKD) for both parties
RTU
MTU
{Sign, Cipher , PK1}
2. Secret Key obtained by QRNG to generate public key and Digital signature using SPHINCS-256 tree approach.
3. Pub Key(PK1) is generated to validate the signature
4. e_QK(M) = Cipher
5. Sign= {Sign_H , Sign_wots, Authentication Path}
6. Received {Sign, Cipher, PK1}
7. Decrypt the cipher to obtain M.
8. Using SPHINCS tree, it generates and validates the authentication path
9. Generate PK2 to validate the signature.

Figure 3.1: The Proposed Scheme Model

- Key Generation: The quantum key is generated from the B92 protocol.
- Encryption: The quantum key is used to encrypt the message to obtain the cipher.
- Authentication: The SPHINCS-256 protocol is applied to generate a signature and public key for verification and validation.

3.1.1 Key Generation

This subsection explains the first phase of the proposed algorithm. It explains the mechanism of the B92 protocol to generate a key for encryption. For a better and easier understanding of the B92 protocol, this module have included the key sifting phase. However, key sifting of B92 is slightly different from that in BB84.

This module assumes that the Remote Terminal Unit (RTU) is the sender, and Master Terminal Unit (MTU) is the receiver. The key generation phase exhibits the B92 protocol, which comprises of the following sub-phases to generate a secret key, using quantum and public channel [30, 99].

3.1.2 Raw Key Exchange

The RTU generates a string of raw qubits with the help of either of the two basis, rectilinear and diagonal, randomly. The rectilinear basis polarizes the 0-bit into a 0° polarized photon, and the diagonal basis polarizes the 1-bit into a $+45^\circ$ polarized photon. The RTU sends the sequence of superposed state qubits, called raw qubits, to the MTU. Furthermore, the raw key exchange utilizes the quantum channel, and the other sub-phases use the public channel.

3.1.3 Key Sifting

In the B92 QKD protocol, Alice (sender) and Bob (receiver) share qubits encoded in two non-orthogonal states: the rectilinear basis state and the diagonal basis state. Alice prepares these qubits and sends them to Bob through the quantum channel and generates raw key.

The MTU has two sets of basis to measure the incoming string of qubits. The MTU has the same pair of basis. However, the analyzer of MTU has polarisation states orthogonal to that of the RTU's. The MTU has a rectilinear basis with polarization state 90° and a diagonal basis with polarization state -45°.

When the MTU can measure the photon with polarisation of 90°, that means the RTU must have sent the qubit with polarization of $+45^\circ$ and hence the state of the qubit is 1. Whereas, when the MTU can detect the photon with polarization -45°, the state of qubit must be 0. The MTU, measuring with a wrong basis, fails to read the qubit. The MTU sends the string of the wrong basis to RTU over the public channel. The RTU discards the bit corresponding to the MTU's wrong basis and obtains the sifted key. blackSince the bases of RTU and MTU match, they can determine which qubits were measured correctly and thus identify a subset of bits that generates the sifted key. Any bits corresponding to measurements where MTU's basis was incorrect are discarded. Once the sifted key is obtained, both RTU and MTU calculate the quantum bit error rate. They measure errors in their extracted key portions from sifted key, then discard those portions to obtain the sub sifted key. The quantum bit error rate (QBER) is obtained by calculating the ratio of mismatched bits and the total number of compared bits.

3.1.4 QBER Calculation

Both RTU and MTU extract a small portion from their sifted key and use it to calculate the error rate. The RTU obtains the quantum bit error rate (QBER) by computing the ratio of the number of mismatched bits and the total number of bits in MTU's extracted key. Both the units discard the extracted portion and acquire the sub sifted key. In the simulation, the thesis sets the threshold of noise or QBER up to 25%. If QBER > 25%, both units discard the process, and when the QBER ≤ 25%, both units proceed to the next sub-phase.

3.1.5 Error Correction Code and Privacy Amplification

To resolve the errors in the subsifted key, both the units exhibit Reed-Solomon (R-S) code. It is an error reconciliation algorithm that corrects multi-bit error with a low computational overhead [100, 101]. The R-S code also evaluates the confidentiality and integrity of the subsifted key. The RTU encodes the key, which involves adding extra parity bits and sends the codeword to the receiver. The MTU receives the codeword, decodes it, and resolves the errors. Meanwhile, the eavesdropper fails to read the subsiftedkey [101, 102].

The Reed Solomon(RS) error correction code has two modules: RS encoder and RS decoder. The RS encoder module involves encoding the information with a Generator polynomial to obtain the codeword. Therefore, it provides confidentiality to the information, however not to the codeword. The codeword is sent to the decoder, which applies a decoding algorithm to correct the errors in the received codeword. When both units perform error correction, they obtain the same subsifted key. However, to reduce any information leakage during R-S code, the MTU and the RTU hash the subsifted key to obtain the final key. This mechanism is called privacy amplification, as it increases the key secrecy.

3.1.6 Classical Channel Authentication in QKD

The primary contribution and focus of the research are to provide message authentication and integrity by using SPHINCS-256 with QRNG, instead of PRNG, in HORST and by considering quantum key derived from QKD.

Our proposed algorithm has the basic steps which are predefined in QKD, namely, Raw key generation (via Quantum channel), Key Sifting, Error Correction Code (ECC), and Privacy Amplification (all three sub-phases via classical channel). The basic steps ensure the key distribution of the quantum key. However, the classical channel in QKD must be authenticated.

To address the above concern, the chapter has introduced SPHINCS-256 in QKD to authenticate the classical channel. Figures 3.2 and 3.3 provides an overall structure of the proposed algorithm. Figure 3.2 illustrates all the sub phases of QKD. After Raw Key Generation, the key sifting and QBER involves sending a series of wrong basis and extracted portion of sifted key from the receiver to the sender over classical channel. During QBER and error correction code phases, the sender sends the codeword over the classical channel. Each information exchanged over the classical channel during QKD is fed to SPHINCS-256 for message authentication.

Our proposed algorithm includes symmetric key generation using quantum key distribution as opposed to relying on trusted third party or a pre-shared secret key scheme. The QKD, itself, does not rely on a pre-shared secret key scheme or a trusted third party. Moreover, the QKD uses a quantum-resistant post-quantum signature scheme during the classical communication phase. Thus, the proposed integration of QKD and SPHINCS-256, described in Figures 3.2 and 3.3, provides intrusion detection and security against both quantum and classical attacks, unlike other traditional algorithms.

3.1.7 Impact of the Use of QKD Technology on the Network Topology

A typical SCADA network topology consist of two primary units: (1) Master Terminal Unit acting as the control center (2) Remote Terminal Unit acting as the field site which gather information from sensors [2, 103, 104]. Our proposed algorithm mainly focuses on securing communication between the RTU and MTU. The communication link between the MTU and the RTU in the SCADA network topology is a point-to-point link.

When QKD is used on the network topology, the RTU and MTU are connected by two channels: Quantum Channel (fiber optic) or Classical Channel (Internet or fiber optic). If the SCADA network topology is based on hardwired communication,

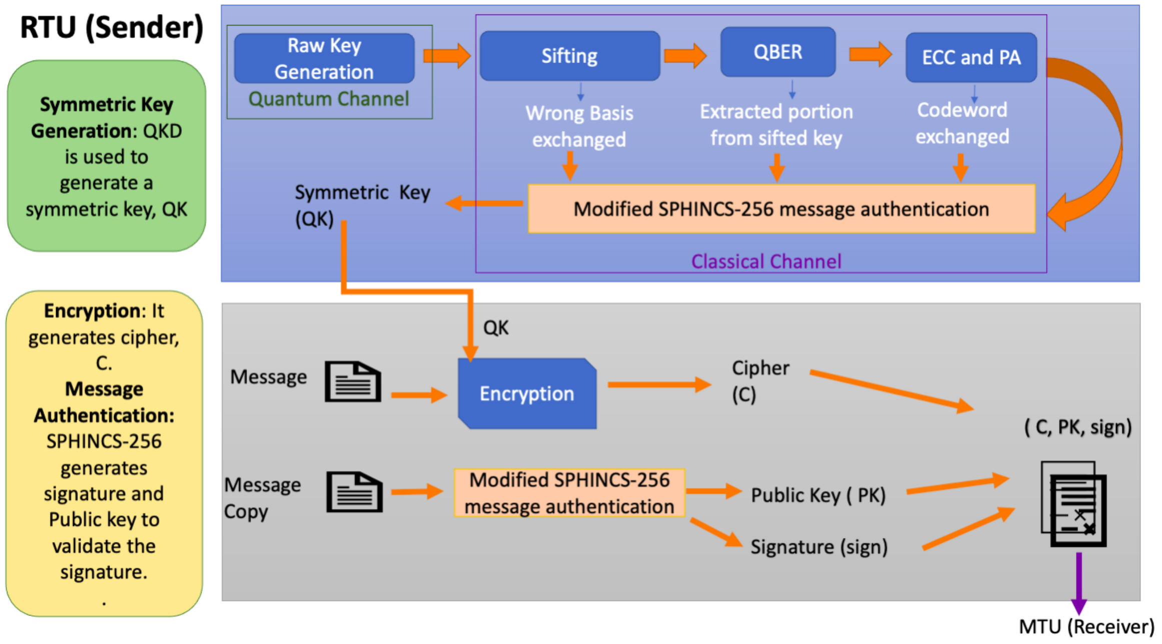

RTU (Sender)
Symmetric Key Generation: QKD is used to generate a symmetric key, QK
Encryption: It generates cipher, C.
Message Authentication: SPHINCS-256 generates signature and Public key to validate the signature.
Raw Key Generation
Quantum Channel
Sifting
QBER
ECC and PA
Wrong Basis exchanged
Extracted portion from sifted key
Codeword exchanged
Symmetric Key (QK)
Modified SPHINCS-256 message authentication
Classical Channel
QK
Message
Encryption
Cipher (C)
(C, PK, sign)
Message Copy
Modified SPHINCS-256 message authentication
Public Key (PK)
Signature (sign)
MTU (Receiver)

Figure 3.2: The structure of the proposed scheme. It includes symmetric key generation, Encryption and message authentication.

the thesis conclude that 2*N fiber optics are needed to deploy QKD [105]. The point-to-point link provides low latency and fast communication. One major constraint of QKD on point-to-point network topology is that if one of the communication links is broken, the entire network fails [104, 106]. However, the impact of authenticated QKD in SCADA network also includes resistance against intrusion and quantum attacks.

For future work in practical QKD research, a quantum channel can be designed which is capable of carrying both quantum and classical information. It may lead to reduction of number of fiber optics or, additional communication channel in a network topology.

3.1.8 Encryption and Digital Signature Algorithm

The procured quantum key encrypts the message to generate cipher. It addresses information confidentiality. It uses this generated cipher in the SPHINCS-256 algorithm and send the cipher along with the signature and public key via the public channel. It also uses the stateless hash signature scheme, namely, SPHINCS, with a hyper-tree of height h and d layers of trees. Each tree uses the Merkle approach and has a height

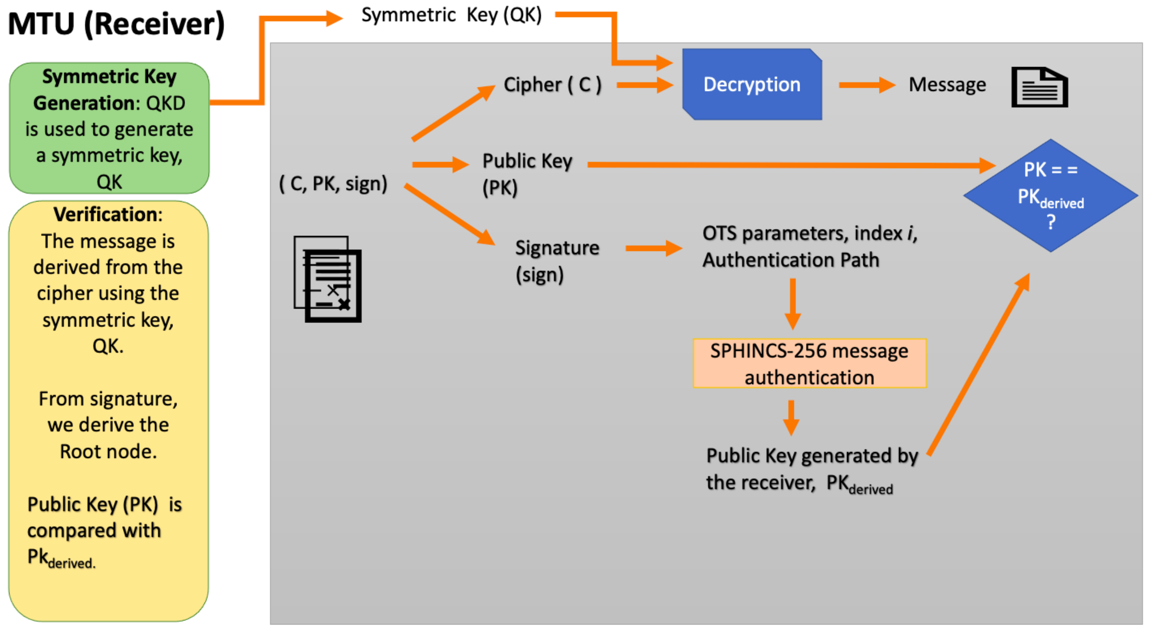

MTU (Receiver)
Symmetric Key Generation: QKD is used to generate a symmetric key, QK
Verification: The message is derived from the cipher using the symmetric key, QK.
From signature, we derive the Root node.
Public Key (PK) is compared with Pk_derived.
Symmetric Key (QK)
Cipher (C)
Decryption
Message
(C, PK, sign)
Public Key (PK)
Signature (sign)
OTS parameters, index i, Authentication Path
SPHINCS-256 message authentication
Public Key generated by the receiver, PK_derived
PK = = PK_derived ?

Figure 3.3: The structure of the proposed scheme. It includes decryption and signature validation.

of h/d. Between the trees, it uses Goldreich's construction by applying WOTS as a one-time signature (OTS) scheme [32]. However, to sign the messages, the HORST scheme is used as a few-time signature. Using hyper-tree and Goldreich's construction significantly reduces the total tree height and reduces the signature size. This chapter discusses the various aspects of SPHINCS-256 in the following sections.

3.2 Security Parameters of SPHINCS-256

The proposed scheme uses SPHINCS-256 to generate digital signature for integrity and authentication. It uses the following security parameters to provide a quality trade-off between speed and signature size [98].

- Security parameter n referring to the bit-length of hashes in HORST and WOTS, n = 256.
- Bit length of message, m = 512.
- Height of the hypertree, h = 60.
- Layers of the hypertree, d = 12.

- WOTS parameter, $w = 16$, used for generating One-time signature.
- Number of HORST secret key elements, $t = 2^{16}$.
- Number of published HORST secret key elements, $k = 32$.

The selection of security parameters in the proposed scheme focuses on the delicate balance between security, computational eficiency, and signature size. A higher n provides stronger resistance to attacks, while m accommodates larger messages; optimal values for h and d balance security and computational complexity; w and t influence the eficiency of signature generation and verification; and the trade-off between t and k ensures a suitable signature size. This parameter configuration is tailored to strike an equilibrium, enabling the scheme to generate secure digital signatures eficiently while accommodating practical constraints in various applications.

3.2.1 Security Level of SPHINCS-256

As mentioned above, Bernstein et al. [32] claim that SPHINCS-256 provides 2^{128} security against the quantum computing attack. However, Chailloux et al. [27] proposed an eficient quantum algorithm that claims to trim the post-quantum security from 2^{128} to $2^{119.6}$. Moreover, Philippe et al. [107] prove a practical attack exploiting the greedy algorithm on the HORS signature scheme. To mitigate such quantum threats, it is needed to increase the security of the post-quantum SPHINCS-256 scheme. In the proposed algorithm, we infuse the laws of quantum physics in SPHINCS-256, by using a quantum random number generator and a quantum key management algorithm.

3.2.2 SPHINCS-256 Tree Construction and Generating Digital Signature

Figure 3.4 provides the general construction of SPHINCS tree. Figure 3.5 provides the mechanism of private and public key generation of both original and modified HORST algorithm. In the following paragraphs, the chapter first explains the construction of SPHINCS tree and then dive into HORST private key generation.

Figure 3.4 provides a generic construction of SPHINCS tree, with $h = 9$ and $d = 3$. The tree provides a hyper tree construction which includes layers of Merkle's tree with OTS nodes as WOTS (+) scheme. At the bottom of the tree lies the FTS nodes denoted as ◇ . The Few Time Signature (FTS) scheme used in SPHINCS is HORST algorithm [108]. The HORST algorithm has two keys, namely, the internal secret key or the private key, and the public key. The module involves modification in HORST.

Generally, the HORST algorithm takes as input the seed and the bitmask Q. As shown in Figure 3.5, the initial secret key SK_H is obtained by feeding seed into a PRNG denoted as G_t. The G_t(seed) generates $SK_H = SK_1 , SK_2 ,. . ., SK_t$ [32]. However, in the proposed SPHINCS algorithm, the HORST replaces G_t with $QRNG_t$.

QRNG is a type of TRNG deployed in quantum hardware and it does not rely on seeding and a deterministic algorithm since it generates random numbers from measuring or observing quantum processes. A practical QRNG includes a source of randomness and, a measuring or detection system. The source of randomness is the entropy source for generating qubits. For example, an optical QRNG inherits randomness from quantum states of light with the help of a photon source, a polarizing beam splitter and detection systems [109, 110, 111].

The $QRNG_t$ generates the truly random secret key $SK_H = SK_1 , SK_2 , ... , SK_t$. It does not need the seed value. The HORST tree is a binary tree constructed using bitmask Q and, the leaves, Li, of the tree is computed using cryptographic hash function F on the elements of the secret key, SKi. Thus, it is denoted as $Li = F(SK_i)$ for $i \in [t-1]$. And the root node of the constructed binary tree on the L_i is the public key of HORST.

SPHINCS-256 tree is a combination of four types of trees: Hyper-tree that includes Merkle's, WOTS, and HORST. The hyper-tree comprises of Merkle's tree connected by WOTS key pairs. Furthermore, the leaves of the SPHINCS-256 are HORS trees. Using the above-mentioned security parameters, the height of each Merkle tree is $h/d = 60/12 = 5$. The HORST tree follows a Merkle's construction with height $\tau = \log_2 t$ [32, 112].

The SPHINCS-256 follows a stateless Goldreich Signature scheme. Each node of the tree has an OTS pair. The key pair at the root, at the outermost layer $(d - 1)$, has SPHINCS-256 public key, PK_H and private key, SK_{d-1} as shown in Figure 3.6. However, the message is first signed with the FTS scheme, named HORST, situated

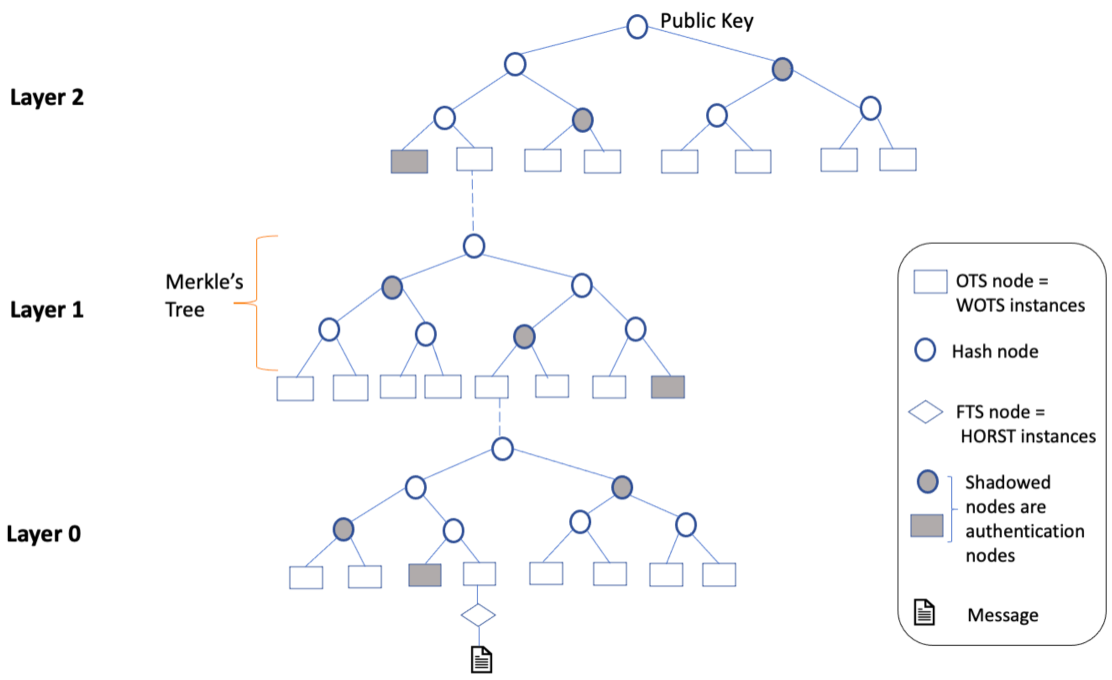

Public Key
Layer 2
Layer 1
Layer 0
Merkle's Tree
OTS node = WOTS instances
Hash node
FTS node = HORST instances
Shadowed nodes are authentication nodes
Message

Figure 3.4: General Construction of SPHINCS tree with h = 9 and d = 3.

is at the bottom of the tree. The module uses the following steps to generate the signature.

- The module first obtains the HORST secret key using QRNG, and using the HORS tree algorithm, it generates the root key, which is the public key. The obtained HORST instances are used to sign the message.

- The obtained HORST signature comprises of k keys, and their respective authentication paths are a part of the SPHINCS-256 signature.

- It then signs the public key of each tree, obtained from the lower layer trees, with WOTS instances as it climbs the SPHINCS-256 tree. The signatures obtained in each layer along with its corresponding authentication path to a public key, the root of SPHINCS-256 tree.

After obtaining the signature, the RTU sends the concatenation of SPHINCS-256 public key PK, cipher obtained, and the signature generated by SPHINCS-256 algorithm.

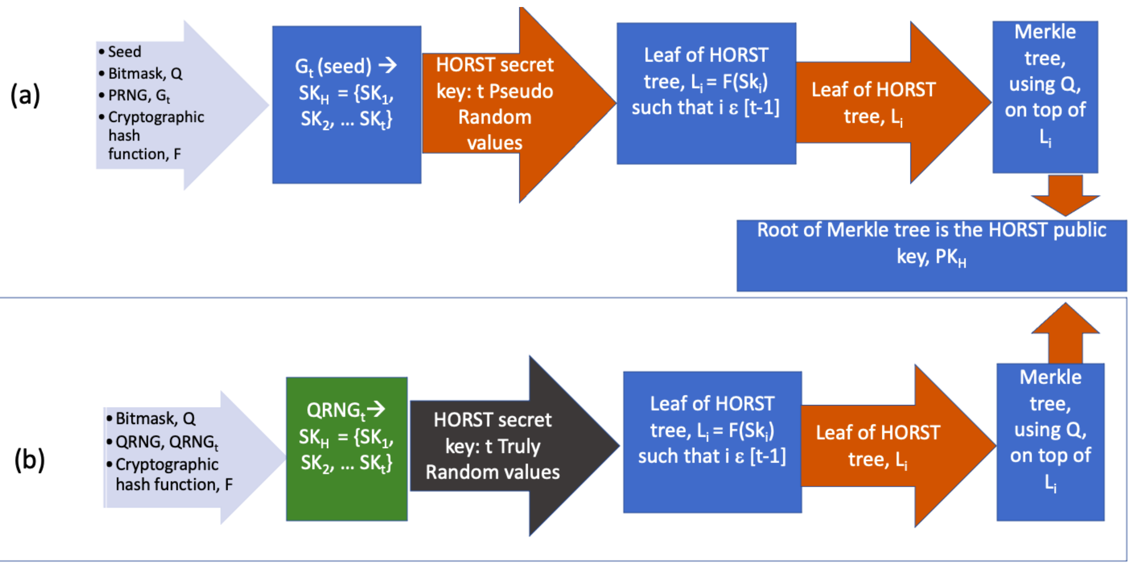

(a)
Seed
Bitmask, Q
PRNG, G_t
Cryptographic hash function, F
G_t (seed) → SK_H = {SK_1, SK_2, ... SK_t}
HORST secret key: t Pseudo Random values
Leaf of HORST tree, L_i = F(Sk_i) such that i ε [t-1]
Leaf of HORST tree, L_i
Merkle tree, using Q, on top of L_i
Root of Merkle tree is the HORST public key, PK_H
(b)
Bitmask, Q
QRNG, $QRNG_t$
Cryptographic hash function, F
$QRNG_t$→ SK_H = {SK_1, SK_2, ... SK_t}
HORST secret key: t Truly Random values
Leaf of HORST tree, L_i = F(Sk_i) such that i ε [t-1]
Leaf of HORST tree, L_i
Merkle tree, using Q, on top of L_i

Figure 3.5: Key Generation: (a) Original HORST algorithm and (b) Proposed HORST algorithm.

3.2.3 Verification

The MTU receives the package and deciphers the cipher using the same quantum key and, obtains the signature $Sign_h$, $Sign_{wots}$, Authentication Path. The RTU derives the digest of cipher by using the OTS parameters hidden in Sign. Using the SPHINCS tree algorithm, it generates and validates the authentication path. The MTU also obtains its public key, $PK_{derived}$, to validate the signature.

3.3 Relationship between Keys and Their Derivations in Our Proposed Scheme

In the following paragraphs, to address the relationship between keys and their derivations, it explains the proposed main framework, and then explain how the SPHINCS-256 public key is generated and to whom it is related or derived from. Three figures are added. Figures 3.2 and 3.3 provide the structure of the proposed scheme. Figure 3.4 provides the general construction of a SPHINCS tree. Figure 3.6 shows the computation of SPHINCS at the topmost layer of the tree.

In the proposed framework, as shown in Figures 3.2 and 3.3, two primary keys are used:

1. Symmetric Key, QK, derived from Quantum Key Distribution.

2. A public key, PK, derived from SPHINCS-256 algorithm.

Flow of the proposed algorithm: The symmetric key, QK, finalised from the authenticated QKD, is used to encrypt the message (data gathered by RTU) to generate cipher. The sender copies the message before encryption, and the message copy is fed to the SPHINCS-256 to generate signature and the public key, PK. A tuple containing Cipher, Signature and public key is sent to the receiver. The public key, PK, is used to validate the signature when received by the receiver.

From the above explanation, the public key, PK, of SPHINCS-256 is not derived from the symmetric key, QK. This feature makes the framework secure since the public key does not carry the essence of the symmetric key, QK because QK is used to encrypt the message.

However, the SPHINCS-256 algorithm has key generation phase which generates its own key pair, (Public key, Private key). The Public key is the PK used in the main framework, explained above and in Figures 3.2 and 3.3.

Key Generation in SPHINCS-256 [32, 108]: The SPHINCS-256 tree is based on hyper tree construction. When the SPHINCS-256 tree is seen as bottom to top approach, the bottom layer of tree has leaf trees as HORST tree with its own leaves. The leaf of the HORST tree is denoted as L_i and the root of the HORST tree is PK_H. The L_i is computed by hashing the elements of HORST secret/private key, SK_H. In Figure 3.4, the root node of the below tree is computed to generate the leaf of the following above tree which further generates its own root node. Thus, in simple terms, the thesis concludes that the private key of HORST tree is further computed, passing through the main tree via FTS node, OTS nodes and hash nodes, to generate the root of the main tree, SPHINCS-256 public key. Thus, SPHINCS-256 public key, PK, is deterministically derived from HORST private or secret key, SK_H.

In the proposed scheme, only the HORST secret key, SK_H , is generated by QRNG. Thus, the QRNG is used only once in the SPHINCS-256 tree. And, it is used only in the HORST tree.

For a detailed view, Figure 3.6 shows a SPHINCS tree with d layers, and the root computation of SPHINCS-256 public key, PK, at d-1 layer. The d-1 layer has the secret key, SK_{d-1}, computed and passed on from the below trees. The SK_{d-1} is further hashed to generate seed. The seed along with bitmask is fed to the WOTS scheme to generate public key of WOTS, PK_{WOTS} . The Leaf, $L_{i(d-1)}$ is computed by hashing PK_{WOTS}. And, the $L_{i(d-1)}$ is fed to binary hash tree to obtain the root of the main tree, PK.

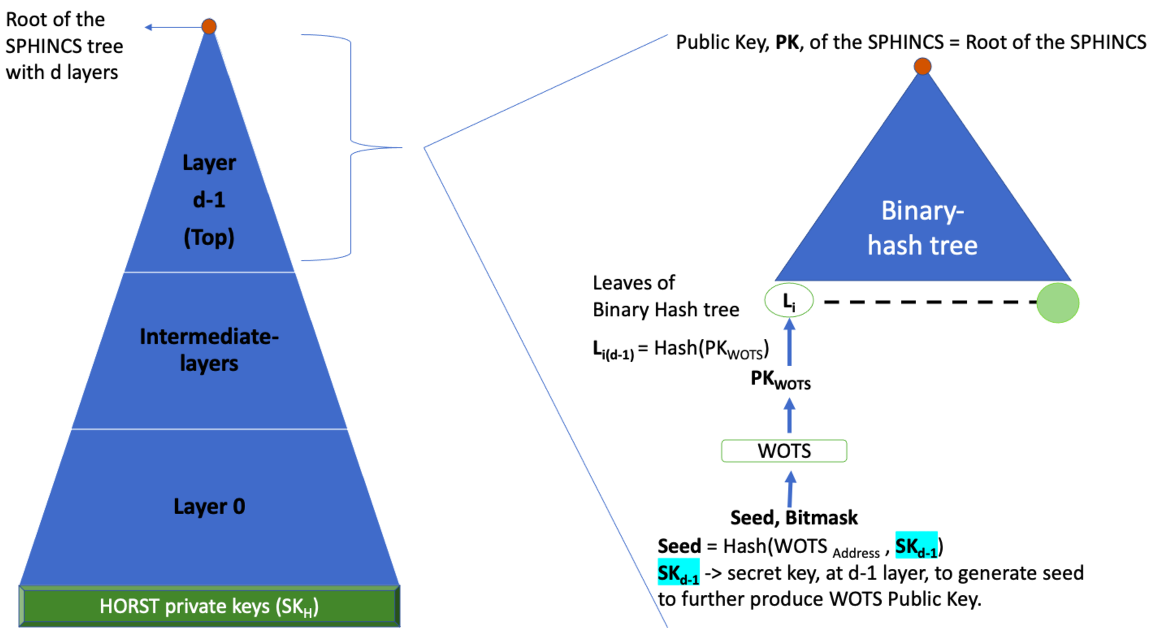

Root of the SPHINCS tree with d layers
Layer d-1 (Top)
Intermediate-layers
Layer 0
HORST private keys (SK_H)
Public Key, **PK**, of the SPHINCS = Root of the SPHINCS
Binary-hash tree
Leaves of Binary Hash tree
L_i
$L_{i(d-1)}$ = Hash(PK_{WOTS})
PK_{WOTS}
WOTS
Seed, Bitmask
Seed = Hash(WOTS $_{Address}$, SK_{d-1})
SK_{d-1} -> secret key, at d-1 layer, to generate seed to further produce WOTS Public Key.

Figure 3.6: SPHINCS tree with d layers. It shows the computation at d-1 layer. The d-1 layer is the topmost layer which has key pairs: (1) Public Key, PK. (2) Private or secret key, SK_{d-1}.

3.4 Formal Analysis of the Proposed Scheme

The thesis involves a formal analysis of the proposed scheme using PRISM [113], a probabilistic model checker, for key generation and Scyther [114] for encryption and digital signature. itverified the B92 protocol based on the Discrete-Time Markov chain by using the PRISM tool [115]. There are three main steps to build a PRISM model. The first step is system specification, involving building a model of a given system with modules, variables, and constants. it constructed three modules, mainly Alice, Bob, and Eve. The second step is property specification, which addresses the two hypothetical questions: How much information is leaked processing the protocol? Can B92 protocol discard or prevent the eavesdropping attack? Thus, it created two properties to address the questions, namely, Probability of detecting an eavesdropper (Eve) and, Probability that Eve measures more than half of the information correctly.

The third step is feeding the model into the PRISM tool [115]. This section has two models [116, 117], namely, B92 protocol with Intercept Resend attack, and B92 protocol with Random Substitute attack.

PRISM is a probabilistic model checker that indicates design flaws in the security protocol before moving to the simulation phase and deploying in a real-time hardware setting. Thus, PRISM does not address the challenges during hardware implementation. Based on Sibson et al. [38] 's experiment on the BB84 protocol, the researchers have observed that the secret key rate is 345 kbps with a clock rate of 560 MHz and QBER of 1.05%. Moreover, Rishab et al. [118] 's paper provides both experimental implementations and software simulation of the B92 protocol.
The practical implementation shows the key rate and QBER is 51 ± 0.5 Kbits/sec and 4.79% ± 0.01%, respectively. The B92 protocol simulation generates a key rate of 52.83 ± 0.36 Kbits/s and QBER of and 4.79% ± 0.01%.

While conducting formal analysis using PRISM, the worst-case scenario of information leakage is considered. The worst-case scenario is that Eve reads more than half of the qubits over the quantum channel correctly. Because, When Eve measures the qubits, there is a 50% probability of using the wrong basis. Further, the probability of getting a correct qubit state using the wrong basis is 50%. Thus, 25% of the qubits measured by the receiver is incorrect, and 75% of the qubits measured are correct. Therefore, currently, more than 50% information leakage is considered. Moreover, simulation of B92 in PRISM deals with a low number of qubits due to computational limitations and significantly high elapsed time.

3.4.1 B92 with Intercept-Resend Attack Model

The Intercept-Resend attack model is based on active eavesdropping. Eve tries to read the qubits or bits of information, exchanged through quantum and public channel, during key generation [115]. Figure 3.7 shows the probability of detecting Eve. Figure 3.8 shows the probability of detecting information leakage more than N/2, while N is the number of qubits.

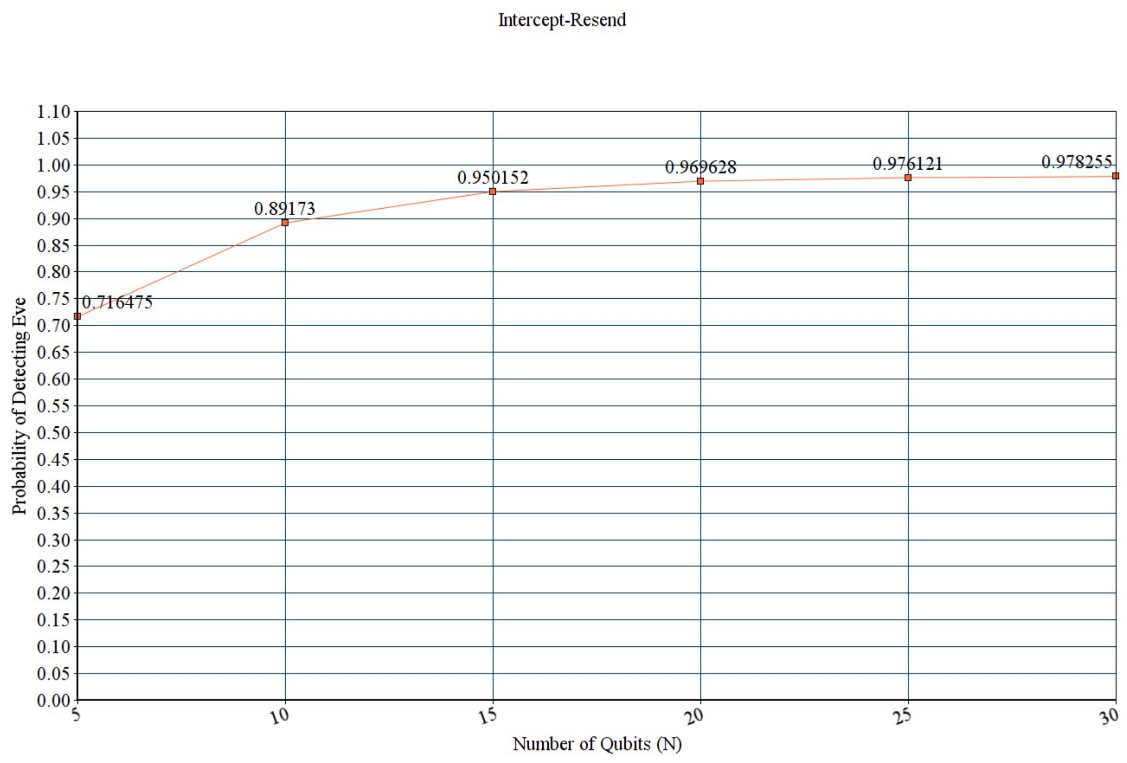


Figure 3.7: Probability of detecting Eve during Intercept Resend Attack.

3.4.2 B92 with Random-Substitute Attack Model

A Random-Substitute attack is a cloning attack. The eavesdropper measures the qubits, which disturbs the state of the qubit. The eavesdropper attempts to duplicate the original state of the qubit and sends it to Bob. Figure 3.9 shows the probability of detecting Eve. Figure 3.10 shows the probability of detecting information leakage more than N/2 by Eve, where N is the number of qubits and $N \in Z^+$.

Therefore, the thesis concludes that the probability that Eve is detected increases exponentially with the number of qubits. Furthermore, the probability that an Eve obtains a correct measurement result for over half the transmissions decreases exponentially with N. It also concludes that an Intercept-Resend attack is more plausible to cloak an Eve's presence than a Random-Substitute attack.

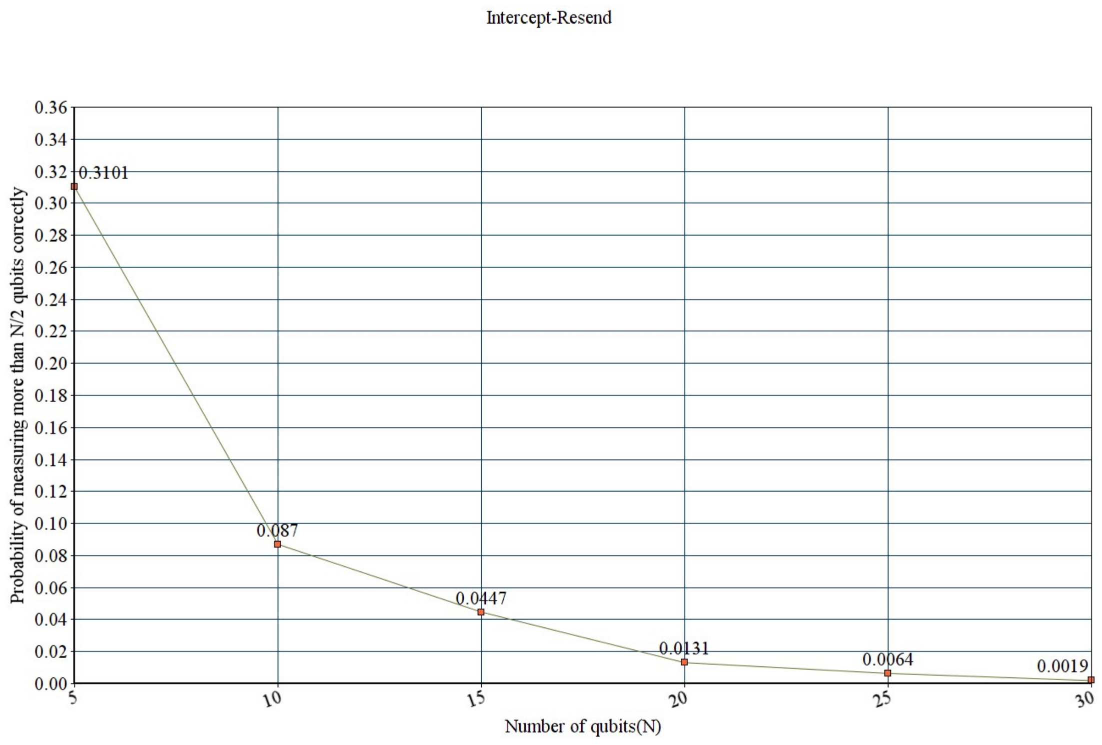


Figure 3.8: Probability that EVE measures more than half of the information correctly during Intercept Resend Attack.

3.4.3 Analysis of Encryption and Digital Signature Phase

This section refers to Scyther to verify the post-quantum scheme used in the proposed framework. Scyther is an open-source tool used for automatic verification of security protocols [114]. The verification is based on three main aspects: (1) Logical message components verify whether a key is public, secret, constant, or freshly generated in each run. (2) Message structure includes key pairing, encryption, signature, and hash schemes. (3) Message order verifies the synchronization and involvement of agents. To build a model in Scyther, it used roles and events to send and receive messages between two agents. The thesis have also used claims that refers to the intended security properties. It verifies two properties, secrecy and authentication. Secrecy analyzes, whether either of the involved agents, communicates to a trusted party on a dubious channel. Authentication addresses the aliveness of agents, synchronization, commitment of the protocol and, message agreement between two parties [114]. Figure 3.11 shows the verification results of the proposed scheme. The thesis concludes

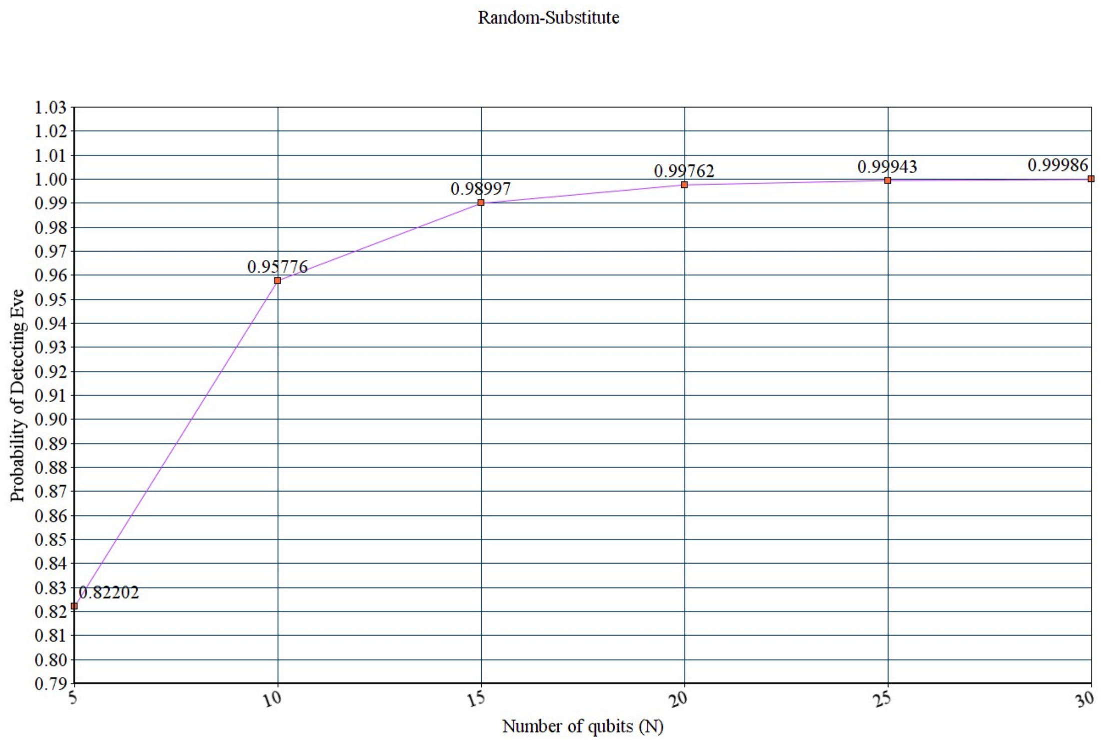


Figure 3.9: Probability of detecting Eve during Random Substitute Attack.

that using quantum scheme in the SPHINCS-256 algorithm makes the proposed algorithm resistant to classical and quantum threats.

3.5 Comparative Analysis and Results

This section includes results based on implementation of the proposed scheme in Python 3.6. It uses the Quantum Information Toolkit [119] to simulate the instances based on quantum physics. Furthermore, to validate the hypothesis, it has generated results and performed a statistical analysis of measured variables. Table 3.1 provides a list of NIST tests executed on the algorithms used in the current and the proposed scheme. The results of the algorithm in the following sections is presented.

3.5.1 Results Obtained in the Key Generation Phase of the Proposed Scheme

The proposed framework is implemented in Python using the Quantum Information toolbox. A practical quantum channel consists of noise from the channel imperfections.

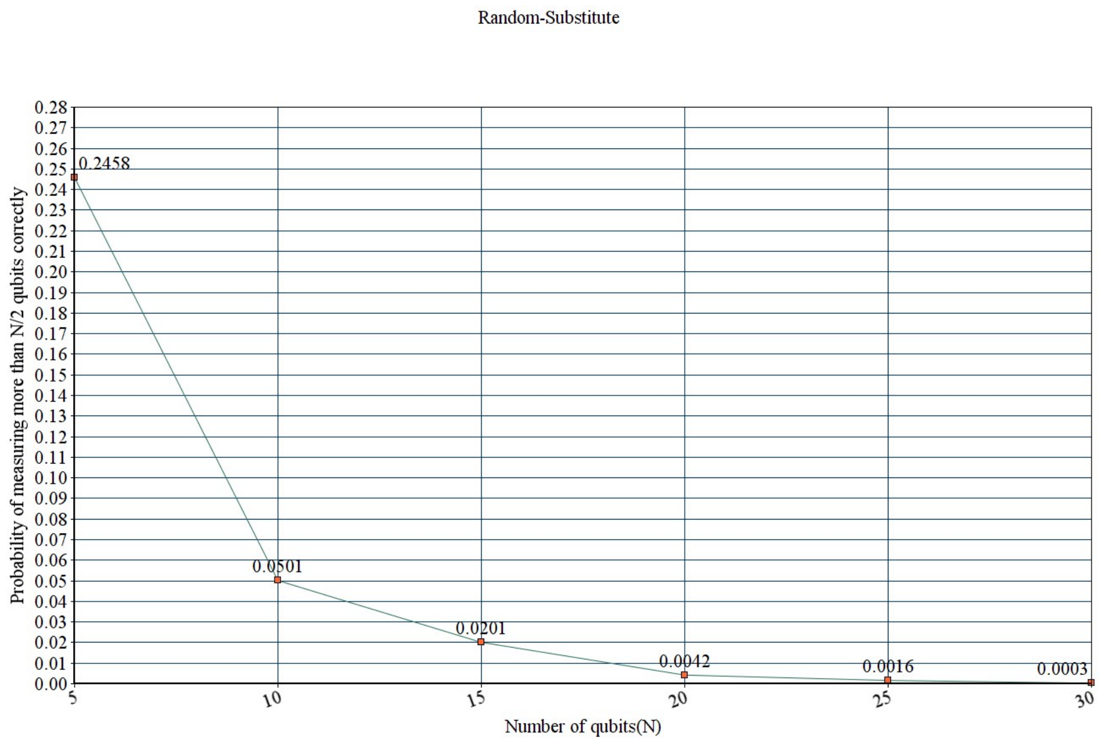
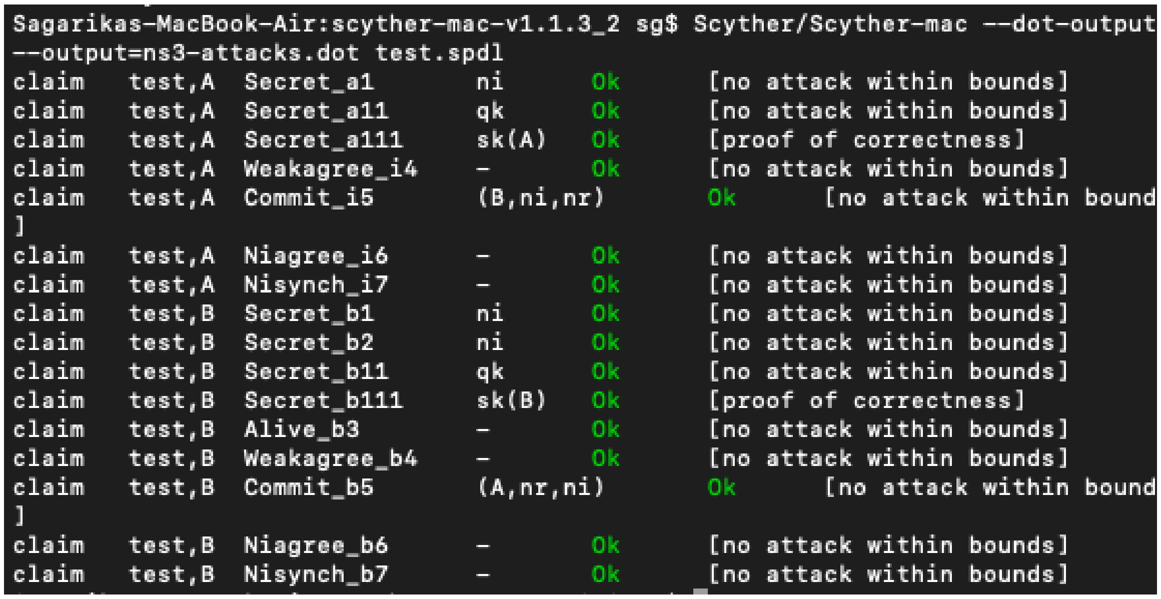


Figure 3.10: Probability that EVE measures more than half of the information correctly during Random Substitute Attack.

To simulate such a channel, the algorithm implemented the logic of the binary symmetric channel. It sends and receives a message in binary with error probability p [120].

During key generation, the phase implemented and observed two scenarios, with and without Eve. Figures 3.12 and 3.13 show that whenever Eve is present, 70% of the time, QBER >25. In the remaining 30% of the cases, the QBER was around 20%. In Figure 3.13, the green bar represents QBER < 25% and, the blue bar represents QBER > 25%. Figure 3.14 exhibits that out of 10 simulations, the mean execution time of the B92 protocol is 0.0198 s. To perform a comparative analysis between two scenarios, Eve and without Eve, we used the following variables.

- Sifted key size
- QBER
- Incorrect basis count



```
Sagarikas-MacBook-Air:scyther-mac-v1.1.3_2 sg$ Scyther/Scyther-mac --dot-output
--output=ns3-attacks.dot test.spdl
claim   test,A  Secret_a1       ni      Ok      [no attack within bounds]
claim   test,A  Secret_a11      qk      Ok      [no attack within bounds]
claim   test,A  Secret_a111     sk(A)   Ok      [proof of correctness]
claim   test,A  Weakagree_i4    -       Ok      [no attack within bounds]
claim   test,A  Commit_i5       (B,ni,nr)       Ok      [no attack within bound
]
claim   test,A  Niagree_i6      -       Ok      [no attack within bounds]
claim   test,A  Nisynch_i7      -       Ok      [no attack within bounds]
claim   test,B  Secret_b1       ni      Ok      [no attack within bounds]
claim   test,B  Secret_b2       ni      Ok      [no attack within bounds]
claim   test,B  Secret_b11      qk      Ok      [no attack within bounds]
claim   test,B  Secret_b111     sk(B)   Ok      [proof of correctness]
claim   test,B  Alive_b3        -       Ok      [no attack within bounds]
claim   test,B  Weakagree_b4    -       Ok      [no attack within bounds]
claim   test,B  Commit_b5       (A,nr,ni)       Ok      [no attack within bound
]
claim   test,B  Niagree_b6      -       Ok      [no attack within bounds]
claim   test,B  Nisynch_b7      -       Ok      [no attack within bounds]
```

Figure 3.11: The encryption and signature phase verified by Scyther tool.

Table 3.1: NIST tests on algorithms

NIST Tests for Randomness		
RSA and ECDSA (AGA-12)	B92(in the proposed algorithm)	QRNG and Chacha-12 SPHINCS-256(in the proposed algorithm)
Frequency Test (Monobit)	Frequency Test (Monobit)	Frequency Test (Monobit)
Frequency Test within a Block	Frequency Test within a Block	Frequency Test within a Block
Run Test	Run Test	Run Test
Longest Run of Ones in a Block	Longest Run of Ones in a Block	Longest Run of Ones in a Block
Discrete Fourier Transform (Spectral) Test	Discrete Fourier Transform (Spectral) Test	Binary Matrix Rank Test
Serial Test 1 and 2	Serial Test 1 and 2	Discrete Fourier Transform (Spectral) Test
Approximate Entropy Test	Approximate Entropy Test	Non-Overlapping Template Matching Test
Cumulative Sums Forward Test	Cumulative Sums Forward Test	Overlapping Template Matching Test
Cumulative Sums Reverse Test	Cumulative Sums Reverse Test	Maurer's Universal Statistical test
Random Excursions Test		Linear Complexity Test
Random Excursions Variant Test		Serial Test 1 and 2
		Approximate Entropy Test
		Cumulative Sums (Forward) Test
		Cumulative Sums (Reverse) Test
		Random Excursions Test
		Random Excursions Variant Test

• Final Key size

Table 3.2 depicts that only QBER gets affected by the presence of Eve. Since QBER is used to discard the keys, if QBER>25, the other variables, mainly the final key and sifted key, do not vary much. Furthermore, the incorrect basis count does not show much difference with or without Eve.

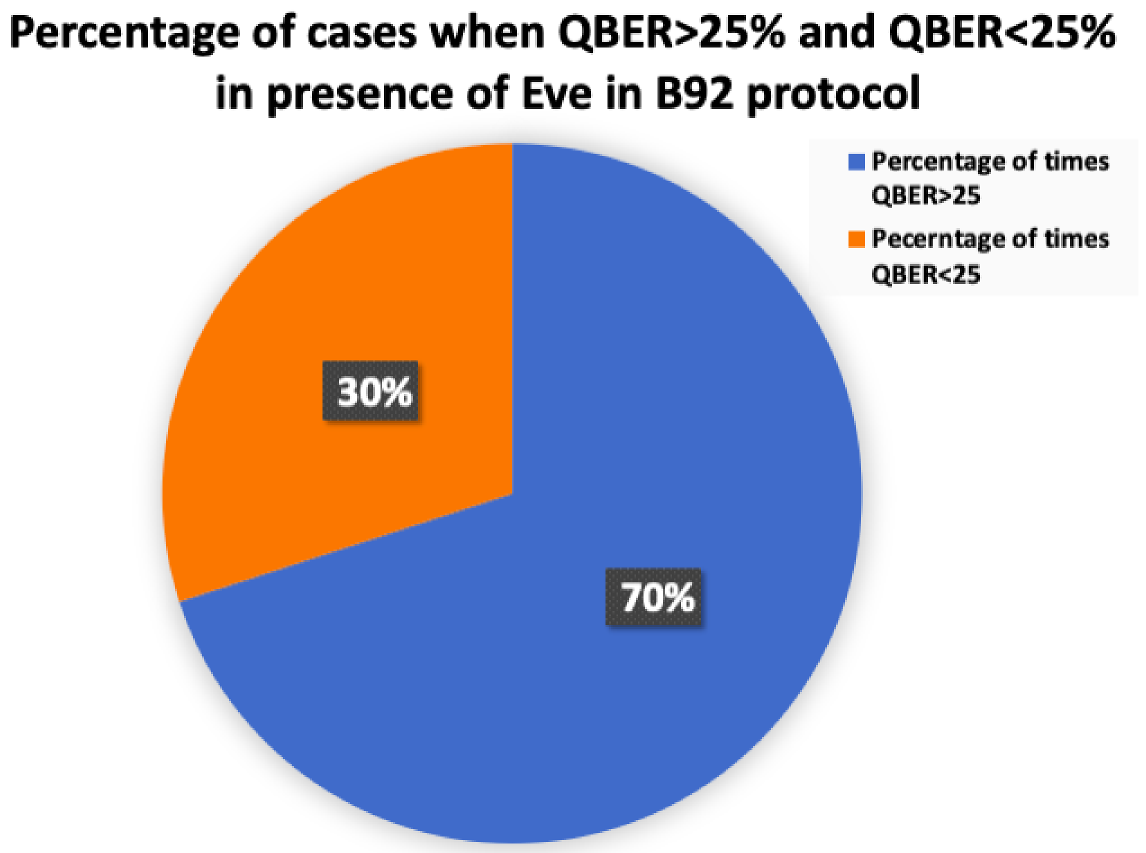


Figure 3.12: Percentage of cases when QBER is greater and less than 25% in the presence of an Eve in B92 protocol.

Table 3.2: Comparative Analysis of B92 in presence of Eve vs B92 in absence of Eve.

	Simulation in Presence of Eve					Simulation in Absence of Eve 6				
Variables	1	2	3	4	5		7	8	9	10
Final Key (bits)	63	63	63	63	61	62	62	63	63	60
Incorrect Basis Count (IBC)	252	264	259	260	259	256	260	262	248	268
QBER (%)	21.5	24.4	20	22	24	9.83	8	4	9.61	12.5
Sifted Key (bits)	260	248	253	252	253	256	252	250	264	244

There are 15 tests for randomness in the NIST statistical test suite. All tests are not suited or required for all random number generators as it depends on various factors, mainly, sample size and algorithms used [121, 122]. The proposed scheme is using a 512-bit key size for raw key and 62-bit key size for the final key. The statistical analysis in this thesis followed Doganaksoy et al.'s paper [122] to use the appropriate statistical tests on both raw key and final key. The module generated ten final and

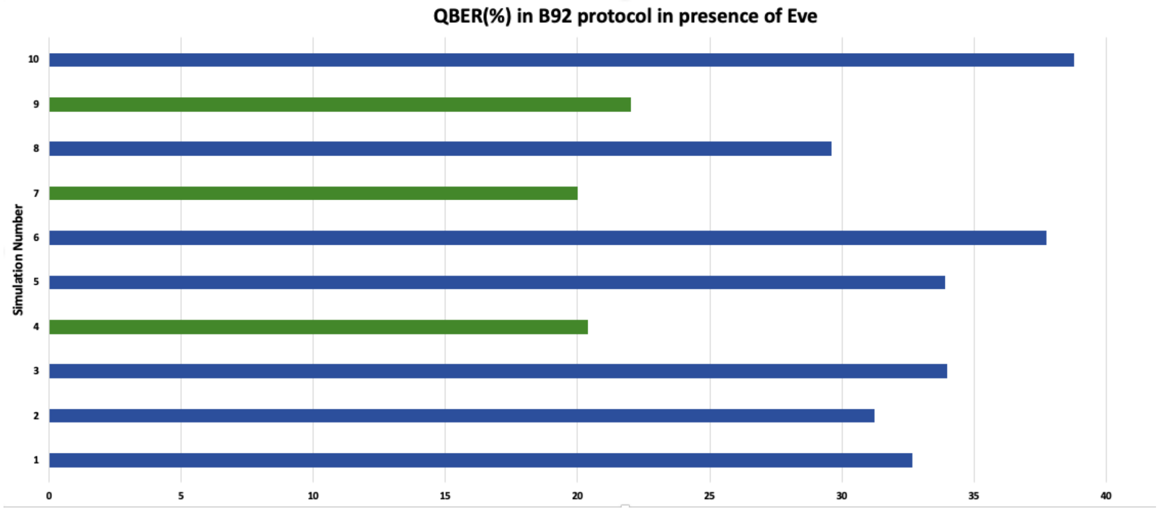


Figure 3.13: QBER in the presence of an Eve in B92 protocol over 10 simulations.

raw keys for testing. All ten keys passed the tests. Thus, The research work in this thesis conclude that both the raw key and final key are random in each simulation. Tables 3.3 and 3.4 provide the p-values of the statistical tests on raw key and final key, respectively. Since the p-values are greater than 0.01, the NIST tool concludes the sequences to be random.

Table 3.3: p-values of appropriate NIST statistical tests on Randomness applied to Raw Key (512 bit) of B92. Conclusion: Random.

p-Values of Raw Keys	Frequency Test	Frequency Test within a Block	Run Test	Longest Run of Ones in a Block	Discrete Fourier Transform (Spectral) Test	Serial Test 1	Serial Test 2	Approx-Imate Entropy Test	Cummulative Sums (Forward) Test	Cummu-Lative Sums (Reverse) Test
1	0.92956	0.65024	0.42611	0.13129	0.62649	0.99402	0.99932	1	0.89202	0.81876
2	0.11161	0.12026	0.54970	0.29407	0.93535	0.49896	0.07918	1	0.22321	0.18615
3	0.79088	0.44540	0.53400	0.02329	0.25614	0.15865	0.14452	1	0.49993	0.73751
4	0.47950	0.32088	0.67429	0.26711	0.46539	0.49896	0.23917	1	0.69601	0.36965
5	0.42632	0.94899	0.36125	0.89453	0.62649	0.49896	0.36062	1	0.57476	0.53660
6	0.72367	0.54742	0.33355	0.20087	0.93535	0.49896	0.36062	1	0.61422	0.92314
7	0.929568	0.66149	0.85941	0.37619	0.62649	0.69077	0.85568	1	0.85688	0.77868
8	0.536101	0.09507	0.98649	0.79974	0.93535	0.69077	0.36062	1	0.89202	0.36965
9	0.376759	0.75873	0.17306	0.73362	0.62649	0.06722	0.63695	1	0.46486	0.65476
10	0.790882	0.19682	0.01713	0.25351	0.93535	0.49896	0.49853	1	0.77868	0.73751

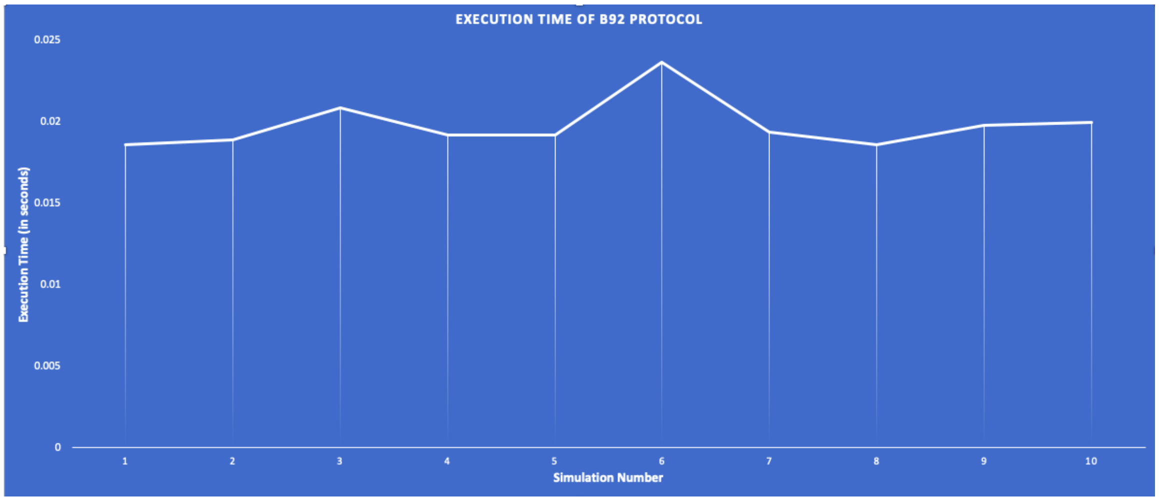


Figure 3.14: Execution Time of B92 protocol.

Table 3.4: p-values of appropriate NIST statistical Tests on Randomness applied to Final Key (mean size = 62 bit) of B92. Conclusion: Random.

p-Values of Final Keys	Frequency Test	Frequency Test within a Block	Run Test	Discrete Fourier Transform (Spectral) Test	Serial Test 1	Serial Test 2	Approximate Entropy Test	Cummulative Sums (Forward) Test	Cummulative Sums (Reverse) Test
1	0.44605	0.44605	0.34151	0.59996	0.49896	0.49853	1	0.25500	0.84788
2	0.52873	0.52873	0.49381	0.93090	0.49896	0.49853	1	0.85301	0.85301
3	0.52873	0.52873	0.06409	0.02604	0.49896	0.49853	1	0.41518	0.85301
4	0.20408	0.20408	0.06066	0.52153	0.49896	0.49853	1	0.19747	0.15080
5	0.10145	0.10145	0.03904	0.28487	0.49896	0.49853	1	0.11756	0.15551
6	0.44605	0.44605	0.55704	0.52153	0.49896	0.49853	1	0.50516	0.50516
7	0.44605	0.44605	0.55704	0.52153	0.49896	0.49853	1	0.50516	0.50516
8	0.52873	0.52873	0.02740	0.28487	0.49896	0.99813	1	0.94315	0.51267
9	0.20408	0.20408	0.04177	0.04177	0.49896	0.49853	1	0.32478	0.32478
10	0.61145	0.61145	0.19079	0.59996	0.49896	0.49853	1	0.73271	0.94027

3.5.2 Comparative Analysis between SPHINCS-QRNG and Current Algorithms Used in AGA-12

To address the first set of research questions, this section provides a comparative analysis between the Digital signature algorithm used, RSA and ECDSA, in AGA-12 vs. Quantum Key Distribution protocol and SPHINCS-256 with the QRNG algorithm in the proposed scheme. It performs five simulations for RSA and ECDSA and generated five key pairs for each algorithm. The research work in this thesis tested the keys by performing the appropriate NIST statistical tests for randomness. Figure 3.15 shows the observations that they do not satisfy 100% of them.

The 192-bit private key and 384-bit public key of ECDSA passed 96% of the NIST tests over five simulations. RSA private key size with 2048 bit passed 90% of the NIST tests over five simulations, and RSA public passed 98% of them. In contrast, the raw key and final key in QKD passed all the tests (100%) over ten simulations. Moreover, SPHINCS-256, with QRNG key pairs, passed 98% of the tests over five simulations.

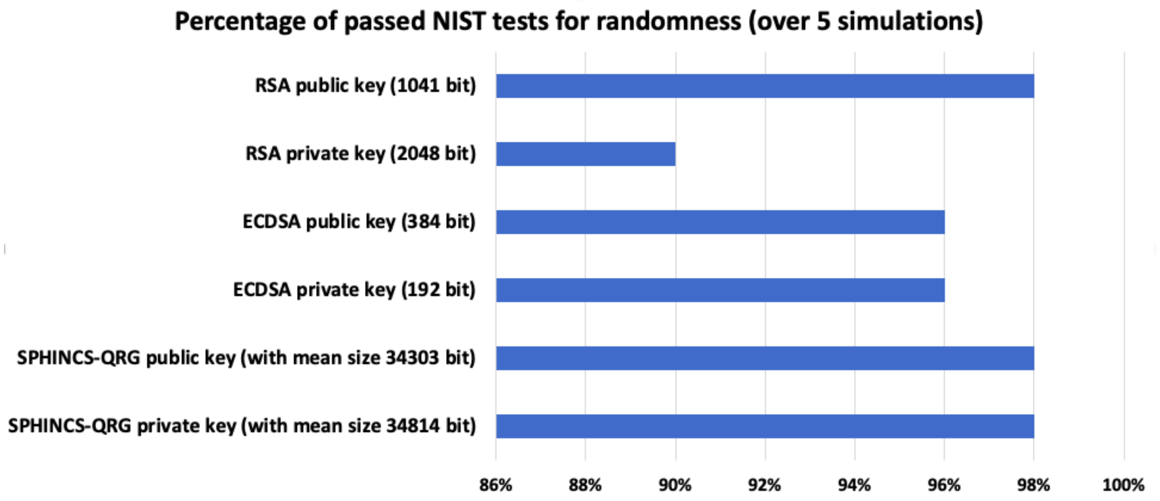


Figure 3.15: Percentage of passed tests, by RSA, ECDSA and SPHINCS-QRNG key pairs, based on NIST statistical test suite on randomness.

3.5.3 Results Obtained in the Signature Generation Phase

To address the second set of research questions, both algorithms: SPHINCS-256 using Chacha12 PRNG and SPHINCS-256 using QRNG generates HORST secret key. It includes two models SPHINCS-Chacha12 and SPHINCS-QRNG. For comparative analysis, It is considered that the root of the SPHINCS tree as the public key. And, it considered the private key of the SPHINCS tree at the topmost (d-1) layer. The signature size obtained in both the models is 27873 bytes. It generates five random numbers from each algorithm and fed each of them to the NIST statistical tool. The module compared both the algorithms by measuring execution time, testing randomness of the generators used in each algorithm, and based on Datcu et al.'s [123] research on testing Chacha12 PRNG.

3.5.4 Comparative Analysis of SPHINCS-QRNG and SPHINCS-Chacha12 Based on Execution Time

Figure 3.16 shows that the execution time of SPHINCS QRNG is more than that of the existing SPHINCS algorithm with PRNG. The mean execution time of SPHINCS-QRNG public key is 160 µseconds, and of the private key is 238.89 µseconds. Whereas, in SPHINCS-Chacha12 algorithm, the average time to generate the public key 112.15 µseconds and the private key is 110.12 µseconds.

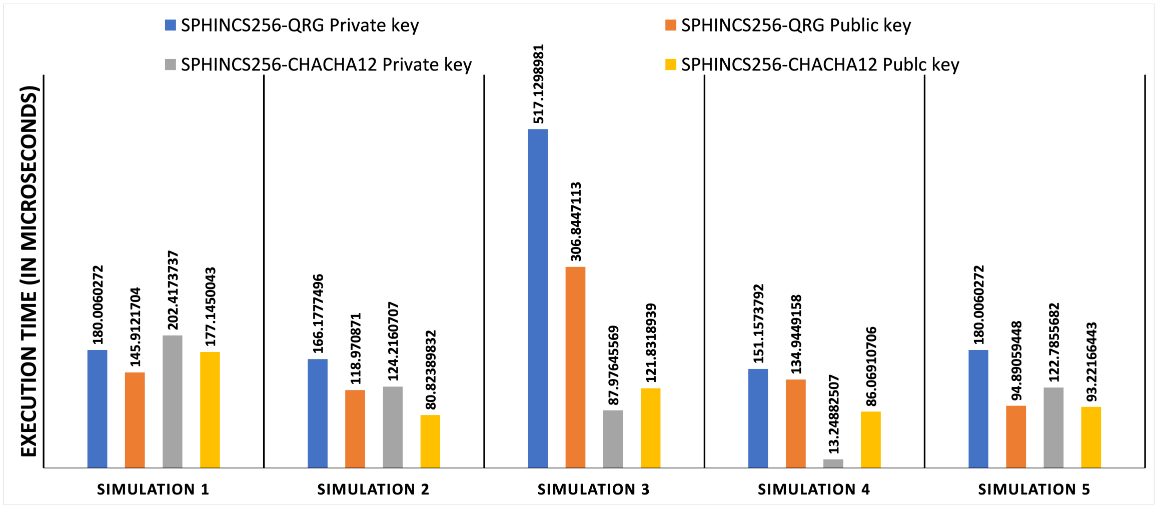


Figure 3.16: Execution Time for SPHINCS-Chacha12 vs SPHINCS-QRNG

Moreover, the thesis also involves a comparative analysis based on theoretical performance. Since SPHINCS is a h/d-ary certification tree, Daniel et al. [32] provided a rough theoretical run time values based on the count of pseudo random number functions (PRFs), PRNG and, hashes. The total height of hyper tree is h with d multiple layers of trees. It takes $d2^{h/d}$ OTS key generations, $d2^{h/d} + 2$ PRF calls, $d2^{h/d} + 1$ PRNG calls and $2t + d((l(w+1))2^{h/d} - 1)$ hashes. HORST uses parameter t, such that $t = 2^{\tau}$. The height of the HORST tree is $\tau = \log t$. WOTS uses a signature size and runtime tradeoff parameter w such that $w \in N$. However, the time complexity of RSA is $O(n^2)$ and that of ECC is $O(n^3)$) [124]. Further, all pseudo-random number generator requires O(n) bit operations. In contrast, QRNG based on Hadamard transformation can be computed in O(nlogn) operations in classical hardware and, in O(1) in quantum hardware [125, 126]. Therefore, the thesis concludes that theoretically, the computational complexity of the proposed algorithm is higher than that of existing SPHINCS based on PRNG and AGA-12.

3.5.5 Comparative Analysis of SPHINCS-QRNG and SPHINCS-Chacha12 Based on Randomness

To test the randomness of the quantum random number generator (QRNG), the analysis performed all 15 tests of the NIST statistical test suite and, the QRNG passed all of them in every simulation. Thus, it concludes that QRNG is truly random. Table 3.5 displays the p-values of random number generated by QRNG used in SPHINCS-256. As the p-values ≥ 0.01, the NIST tool concludes the sequence to be random with a confidence of 99%. Figures 3.17 and 3.18 display the results of Random Excursions Test and Random Excursions Variant Test, respectively for SPHINCS-QRNG. The Random Excursions test executes sub-tests on each of the following states; −4, −3, −2, −1, +1, +2, +3 and +4, to check the frequency of visits to a cumulative sums state within a cycle of a random walk matches with that one would expect for random sequence [121]. For a certain state, if the p-value ≥ 0.01, the sequence is random. For example, in Figure 3.17, the p-value of QRN 1 with state +1 is approximately 0.8492. Thus, the sequence at state +1 is random.

Table 3.5: p-values of appropriate NIST Statistical Tests for Random and PRNG applied on Quantum Random Number Generator(QRNG) used in SPHINCS-256.

p-values of QRNG Used in SPHINCS-256	QRN 1	QRN 2	QRN 3	QRN 4	QRN 5
Frequency Test (Monobit)	0.852444761	0.011735483	0.149302374	0.406538784	0.649828827
Frequency Test within a Block	0.161769992	0.733286939	0.575099934	0.665007898	0.518616736
Run Test	0.217961992	0.827155028	0.32712429	0.732869216	0.924945411
Longest Run of Ones in a Block	0.513396784	0.969974739	0.637115311	0.248794857	0.913444629
Binary Matrix Rank Test	0.262333935	0.822162028	0.862431288	0.111413103	0.071873969
Discrete Fourier Transform (Spectral) Test	0.142033423	0.011616891	0.157596656	0.776045999	0.613759295
Non-Overlapping Template Matching Test	0.63600041	0.504047187	0.415637608	0.169519974	0.262129471
Overlapping Template Matching Test	0.200756533	0.505736739	0.353171054	0.591057811	0.508073676
Maurer’s Universal Statistical test	0.124605474	0.243079644	0.01674513	0.73273627	0.889955415
Linear Complexity Test	0.8575379	0.547314553	0.126281648	0.190952376	0.371943788
Serial test 1	0.667804599	0.9238906	0.773287042	0.949470145	0.285436568
Serial test 2	0.691057349	0.953760341	0.7814567	0.920487151	0.576467535
Approximate Entropy Test	0.569869981	0.89057965	0.092335008	0.350697922	0.18177552
Cummulative Sums (Forward) Test	0.800359989	0.013619809	0.215229911	0.508140027	0.69796035
Cummulative Sums (Reverse) Test	0.943118012	0.018863213	0.268772338	0.771928763	0.764505358

The Random Excursions Variant test verifies whether the number of visits to a particular state in a cumulative sum random walk deviates from the expected number of visits in the random walk. It considers 18 states consisting of { −9, −8,

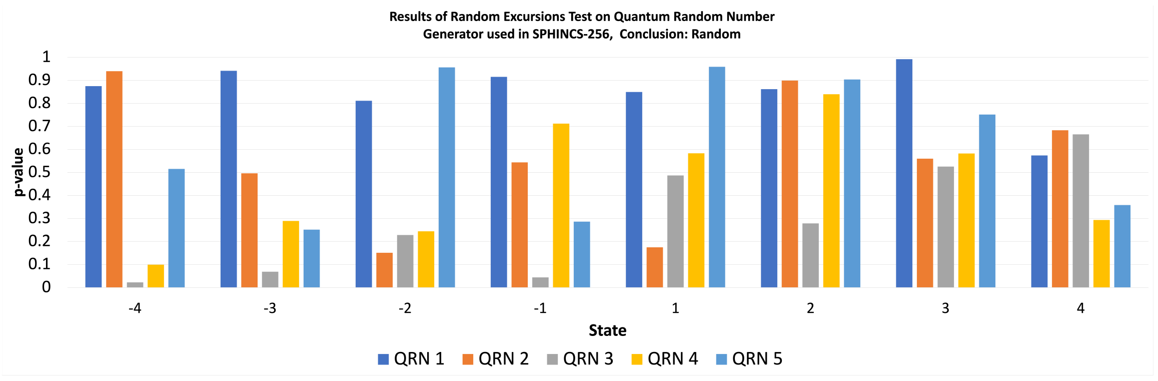


Figure 3.17: Results of Random Excursions Test on Quantum Random Number Generator(QRNG) used in SPHINCS-256.

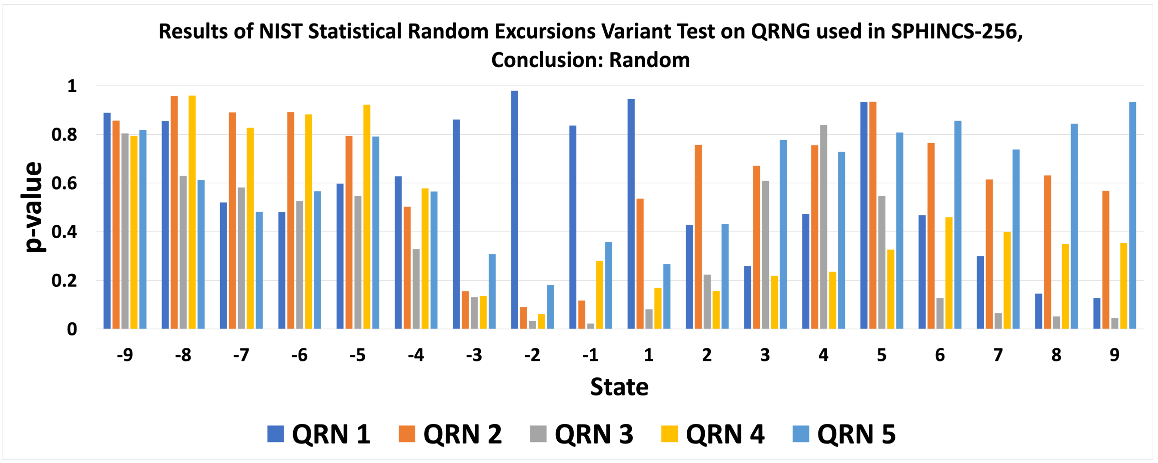


Figure 3.18: Results of Random Excursions Variant Test on Quantum Random Number Generator(QRNG) used in SPHINCS-256.

..., +8, +9} and, if the p-value of a particular state is greater than or equal to 0.01, that means the sequence is random. However, when it ran all 15 tests on a random number generated by SPHINCS-Chacha12, it does not pass 100% of the NIST statistical tests. The pseudo-random number, PRN 5, generated by Chacha-12, failed the Maurer's Universal Statistical test. Also, PRN 1 and PRN 5 does not pass the Random Excursions Test and Random Excursions Variant Test with all states. Table 3.6, Figures 3.19 and 3.20 exhibit the results of the randomness of Chacha-12 PRNG tested by NIST statistical test suite.

This section performs 15 statistical analysis on five random numbers for each algorithm. Thus, it performs 15 ×5 = 75 tests. Out of 75 tests, Chacha-12 passed

Table 3.6: p-values of appropriate NIST Statistical Tests for Random and PRNG applied on Chacha-12 used in SPHINCS-256.

p-Values of PRNG Used in SPHINCS-256	PRN 1	PRN 2	PRN 3	PRN 4	PRN 5
Frequency Test (Monobit)	0.756560956	0.61285665	1	0.87288107	0.76723008
Frequency Test within a Block	0.466408066	0.2519857	0.66537382	0.05620039	0.27892162
Run Test	0.046601502	0.41321088	0.95693516	0.93466707	0.13415598
Longest Run of Ones in a Block	0.39001338	0.57336093	0.31763597	0.60612676	0.43188342
Binary Matrix Rank Test	0.234372205	0.70947779	0.89969577	0.38005323	0.77390385
Discrete Fourier Transform (Spectral) Test	0.520636833	0.87603089	0.52063683	0.21202315	0.4798148
Non-Overlapping Template Matching Test	0.17548433	0.67642991	0.08873813	0.25081909	0.0158475
Overlapping Template Matching Test	0.341609907	0.93826949	0.03571624	0.88583617	0.19903682
Maurer’s Universal Statistical test	0.251209205	0.22365261	0.37912508	0.21287032	0.00739755
Linear Complexity Test	0.636767031	0.43256403	0.95718573	0.68049537	0.26611467
Serial test 1	0.266168112	0.04303137	0.85174723	0.77514063	0.06351778
Serial test 2	0.111246791	0.36341311	0.71254001	0.62943052	0.1555089
Approximate Entropy Test	0.991465303	0.50945017	0.72260759	0.55896218	0.70516401
Cummulative Sums (Forward) Test	0.550993298	0.64760991	0.550134	0.6264806	0.39388365
Cummulative Sums (Reverse) Test	0.834146968	0.85546589	0.550134	0.77562923	0.22547471

70 tests and, QRNG passed all of them. Figure 3.21 exhibits a graph that shows Chacha-12 PRNG scored 93.3%, and QRNG scored 100% for successfully passing the tests. The Chacha-12 algorithm is feasible for resource-constrained devices [127]. However, Datcu et al. [123], showed that secure PRNG Chacha does not pass all the statistical tests of Monte-Carlo analysis. Moreover, ID Quantique has developed a QRNG chip for critical infrastructure involving the Internet of Things (IoT) and other resource constrained devices [37]. Table 3.7 provides a synthesis of results obtained from comparative analysis of the proposed algorithm, RSA, ECDSA and SPHINCS-256.

3.6 Conclusion

The thesis have proposed a collision and preimage resistant framework for ensuring SCADA system security. SPHINCS-256, a post-quantum algorithm, provides 2^{128} security against a quantum threat. However, researchers have increased eficiency and speed of quantum algorithm based on Grover’s algorithm, which reduces the post-quantum security from 2^{128} to $2^{119.6}$ in a quantum setting. Therefore, the chapter proposes to use B92, a quantum key distribution protocol, to obtain the cipher and a quantum random number generator to generate a truly random number for the HORST secret key used in SPHINCS-256. It has formally verified the proposed scheme by using PRISM and Scyther. The thesis conclude that with the number of

Table 3.7: Synthesis of comparative analysis of the proposed algorithm, RSA, ECDSA and SPHINCS-256.

Algorithms	Public Key Size (bits)	Private Key Size (bits)	Raw key size (bits)	Final Key Size (bits)	QBER (mean)	Percentage of passed NIST randomness tests	Execution Time/ Time Complexity
RSA	1041	2048	N/A	N/A	No intrusion detection	Private key: 92% Public key: 98%	O (n^2)
ECDSA	384	192	N/A	N/A	No intrusion detection	96%	O (n^3)
The proposed algorithm	34303	34814	512	62	15.584% Detects eavesdropping.	Both keys: 98% QRNG: 100%	$d2^{h/d}$ OTS key generations $d2^{h/d}$ + 2 PRF calls. $d2^{h/d}$ + 1 PRNG calls and $2t + d((l(w+1))2^{h/d} - 1)$ hashes QRNG : O(nlogn) in classical hardware. O(1) in quantum hardware. Public Key Generation: 160 µseconds Private Key Generation: 238.89 µseconds
SPHINCS-256	34303	34814	N/A	N/A	No intrusion detection	Chacha-12: 93.3%	$d2^{h/d}$ OTS key generations $d2^{h/d}$ + 2 PRF calls. $d2^{h/d}$ + 1 PRNG calls and $2t + d((l(w+1))2^{h/d} - 1)$ hashes PRNG : O(n) bit operations. Public Key Generation: 112.5 µseconds Private Key Generation: 110.12 µseconds

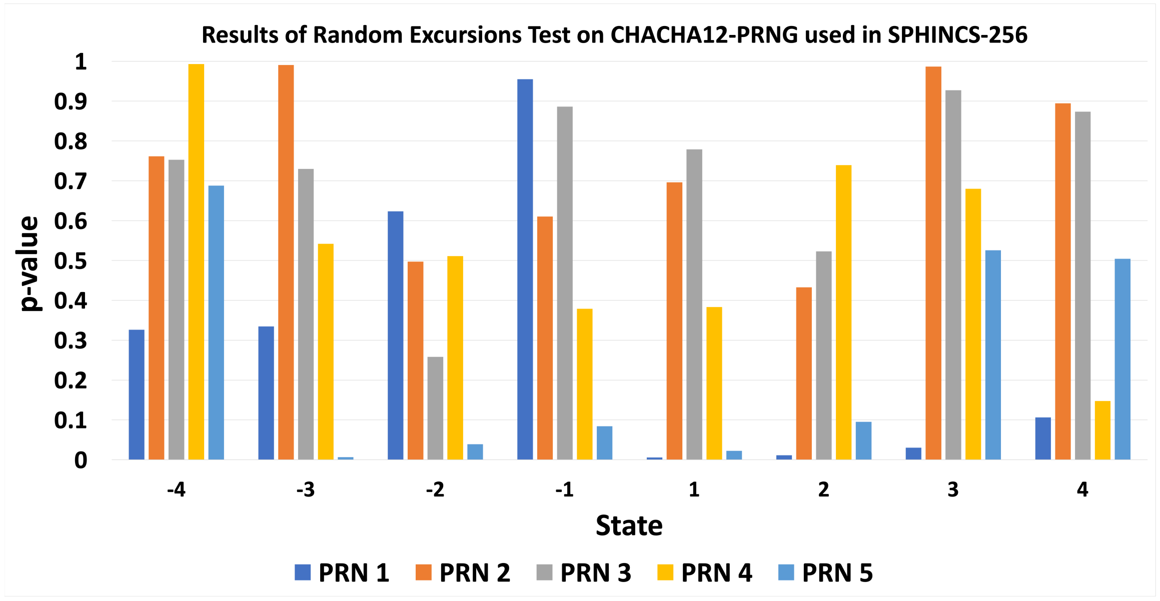


Figure 3.19: Results of Random Excursions Test on Chacha-12 PRNG used in SPHINCS-256.

qubits, the probability of detecting intruder Eve increases exponentially, decreasing the likelihood of information leakage. Furthermore, B92 is more likely to detect Eve's presence in case of a Random-Substitute attack than an Intercept-Resend attack. This chapter validated the hypothesis by simulating the proposed scheme and performing statistical analysis. The thesis work analyzes the randomness of RSA, ECDSA keys used in AGA-12 against the randomness of the quantum key used in the proposed algorithm. It observes that keys generated by QKD satisfy 100% of the NIST randomness tests, unlike RSA and ECDSA keys. RSA private key passed 90%, and ECDSA keys passed 96% of the tests. the module provides a comparative analysis between two algorithms, SPHINCS-256 with Chacha-12 and SPHINCS-256 with QRNG. It observes that the Quantum Random Number Generator passes all the statistical tests for randomness, unlike Chacha-12 PRNG. However, in computational hardware, the computation cost of the proposed SPHINCS-QRNG is higher than that of AGA-12 and the existing SPHINCS-PRNG algorithm. Thus, the thesis conclude that there is a trade-off between security and computation cost. Our proposed framework, using true random numbers based on uncertainty and quantum superposition principles, provides more resistance than AGA-12 and SPHINCS-256 against quantum and classical threats.

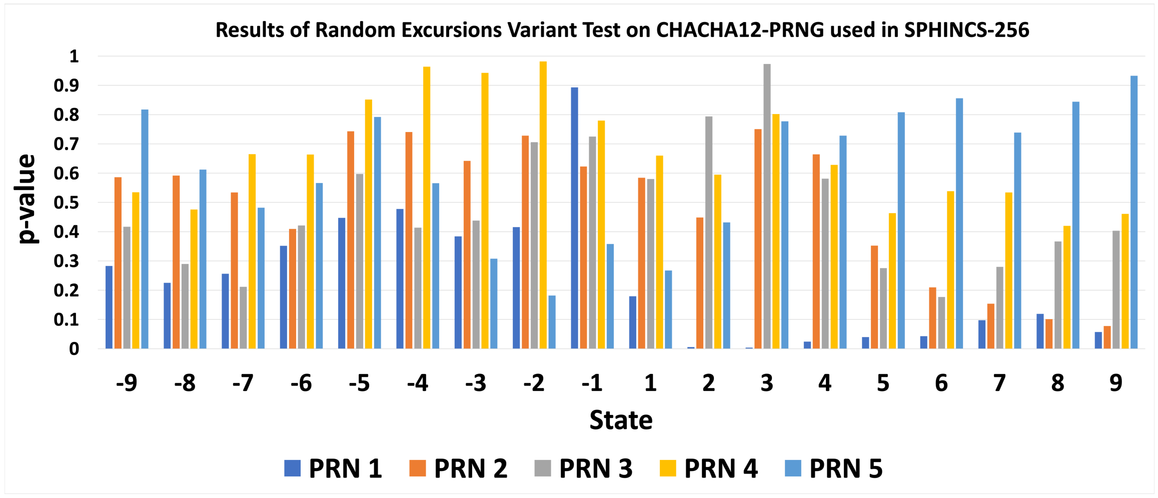


Figure 3.20: Results of Random Excursions Variant Test on Chacha-12 PRNG used in SPHINCS-256.

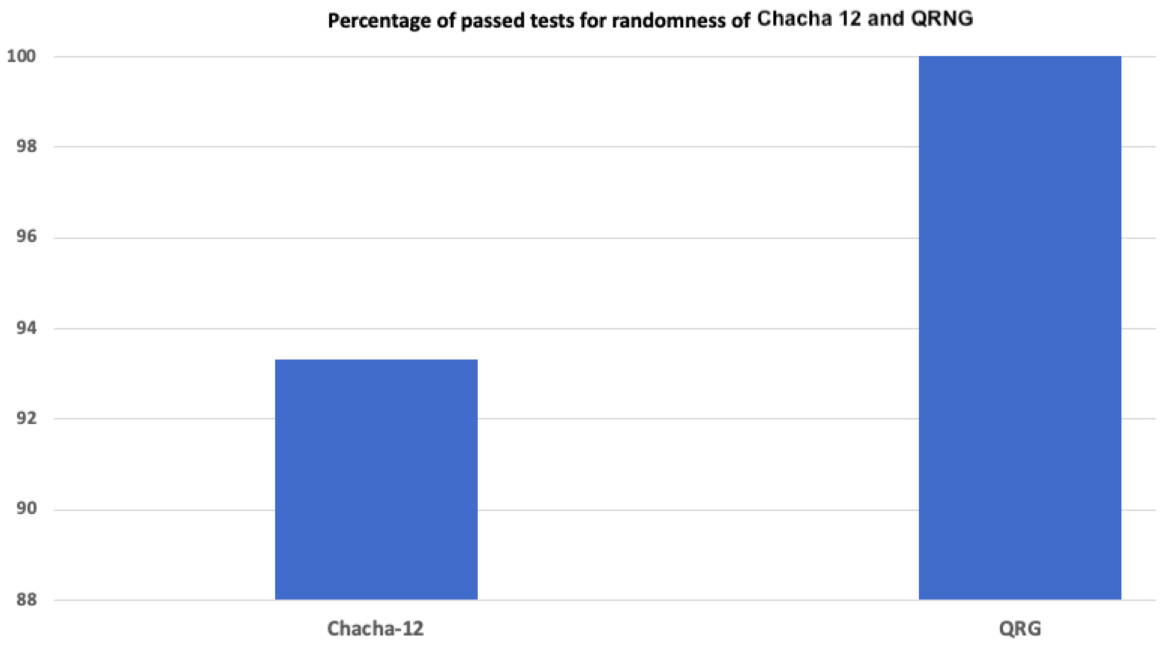


Figure 3.21: Percentage of passed tests for randomness scored by Chacha-12 and QRNG.

Chapter 4

Module 2: Multi-phase Quantum resistant Framework for Secure Communication in SCADA Systems

The research work reported in this chapter has resulted in the following publications:

1. S. Ghosh, M. Zaman, S. Sampalli, Quantum-Safe Asymmetric Cryptosystems: Current Solutions and Future Directions against Quantum Attacks, In Holistic Approach to Quantum Cryptography in Cyber Security (pp. 99-120), CRC Press 'Taylor and Francis Group', 2022. (Impact Factor: 4.2)

2. S. Ghosh, M. Zaman, R. Joshi , S. Sampalli, Multi-phase Quantum resistant Framework for Secure Communication between RTU and MTU", Manuscript under review after major revision, IEEE Transactions on Dependable and Secure Computing, June 2022. (Impact Factor: 6.79)

4.1 Summary of the chapter

Similar to other systems or networks, a Supervisory Control and Data Acquisition (SCADA) network is susceptible to various security threats and vulnerabilities. These threats pose potential risks to the proper functioning and integrity of the SCADA network. Thus, this thesis outlines the following potential threats that can affect a SCADA network [2].

- Disruption of availability can result in power outages and adversely impact the operational eficiency of the power supply. Such an event can trigger a chain reaction in the physical domain, leading to severe consequences. Therefore, ensuring availability as a fundamental security objective is of paramount importance in SCADA networks.

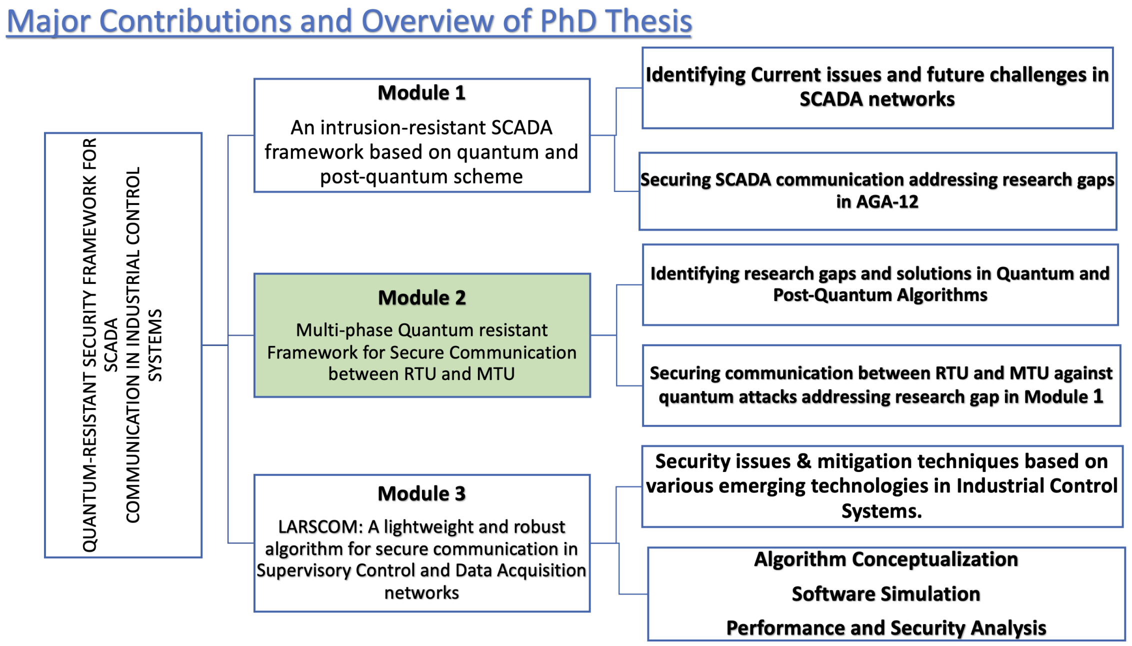

Major Contributions and Overview of PhD Thesis
QUANTUM-RESISTANT SECURITY FRAMEWORK FOR SCADA COMMUNICATION IN INDUSTRIAL CONTROL SYSTEMS
Module 1
An intrusion-resistant SCADA framework based on quantum and post-quantum scheme
Module 2
Multi-phase Quantum resistant Framework for Secure Communication between RTU and MTU
Module 3
LARSCOM: A lightweight and robust algorithm for secure communication in Supervisory Control and Data Acquisition networks
Identifying Current issues and future challenges in SCADA networks
Securing SCADA communication addressing research gaps in AGA-12
Identifying research gaps and solutions in Quantum and Post-Quantum Algorithms
Securing communication between RTU and MTU against quantum attacks addressing research gap in Module 1
Security issues & mitigation techniques based on various emerging technologies in Industrial Control Systems.
Algorithm Conceptualization
Software Simulation
Performance and Security Analysis

Figure 4.1: Major Contributions and Overview of PhD Thesis

- Breach of confidentiality occurs when an unauthorized entity eavesdrops on the communication channel, compromising the privacy of sensitive data and facilitating data theft.

- Compromise of integrity involves unauthorized alteration of data, where the recipient receives manipulated information instead of the original content. This type of attack can be accomplished through a Man-in-the-Middle attack, which opens the door to malware injection and IP spoofing.

This chapter refers to Module 2 research work as highlighted in Figure 4.1. The chapter presents a comprehensive and robust multi-phase framework designed to ensure the security of the communication channel between RTU and MTU, taking into account quantum resistance. The framework encompasses four major phases to establish secure communication. In Phase 1, a modified SIKE algorithm is employed for shared secret key generation. Phase 2 utilizes the authenticated BBM92 protocol to generate a session or symmetric key. Phase 3 employs the ASCON algorithm for data encryption using the symmetric key. Lastly, in Phase 4, message authentication is performed using SHA-3 and the shared secret key (SSK) obtained from the SIKE algorithm in Phase 1. Each phase contributes essential security features to the

SCADA framework, as depicted in Figure 5.2. It addresses availability, confidentiality, integrity, authentication, and scalability. However, it does not encompass a host-based intrusion detection system.

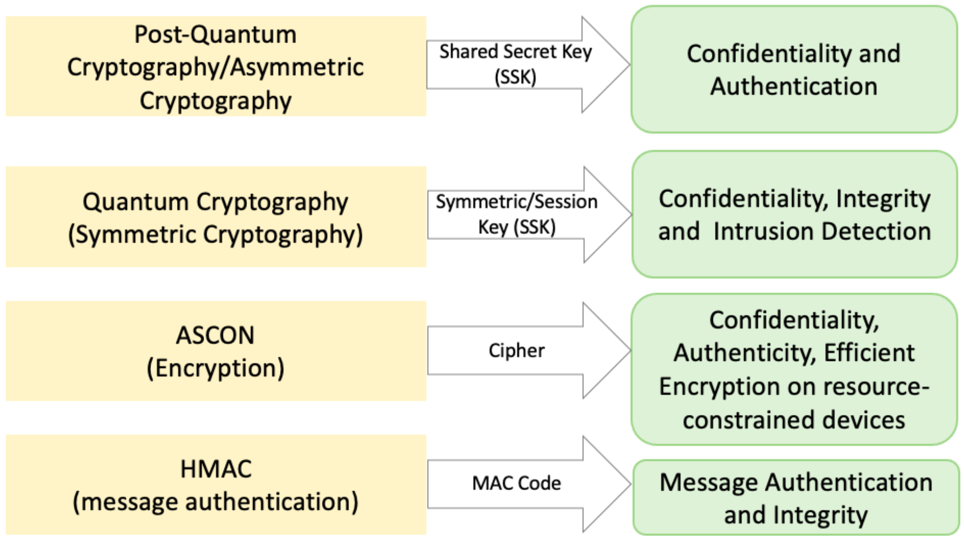

Post-Quantum Cryptography/Asymmetric Cryptography
Shared Secret Key (SSK)
Confidentiality and Authentication
Quantum Cryptography (Symmetric Cryptography)
Symmetric/Session Key (SSK)
Confidentiality, Integrity and Intrusion Detection
ASCON (Encryption)
Cipher
Confidentiality, Authenticity, Efficient Encryption on resource-constrained devices
HMAC (message authentication)
MAC Code
Message Authentication and Integrity

Figure 4.2: Security features provided by multiple phases of the proposed scheme.

In the proposed framework, a unique shared secret key and session/symmetric key are generated for each exchange of sensitive data between RTU and MTU. The secure exchange of these keys is facilitated by leveraging post-quantum cryptography for the shared secret key and quantum cryptography for the session key. During the key exchange process, authentication and validation are performed using HMAC to ensure the integrity of the keys. Once the shared secret key and session key are established, the original message is encrypted using the ASCON-128 algorithm, and a message authentication code is generated to verify the integrity of the data.

Given the critical nature of the sensitive data exchanged in SCADA control units, prioritizing both security and eficiency is crucial [128]. To address this, the thesis introduces two primary modules: a two-layered key management phase and an encryption and authentication phase. The two-layered key management phase incorporates the use of the post-quantum cryptographic scheme SIKE and the quantum cryptographic protocol BBM92. In the encryption phase, the ASCON-128 algorithm is employed for data encryption, while SHA-3 is utilized for hashed message authentication. A comprehensive overview of the proposed framework is illustrated in Figure 4.3.

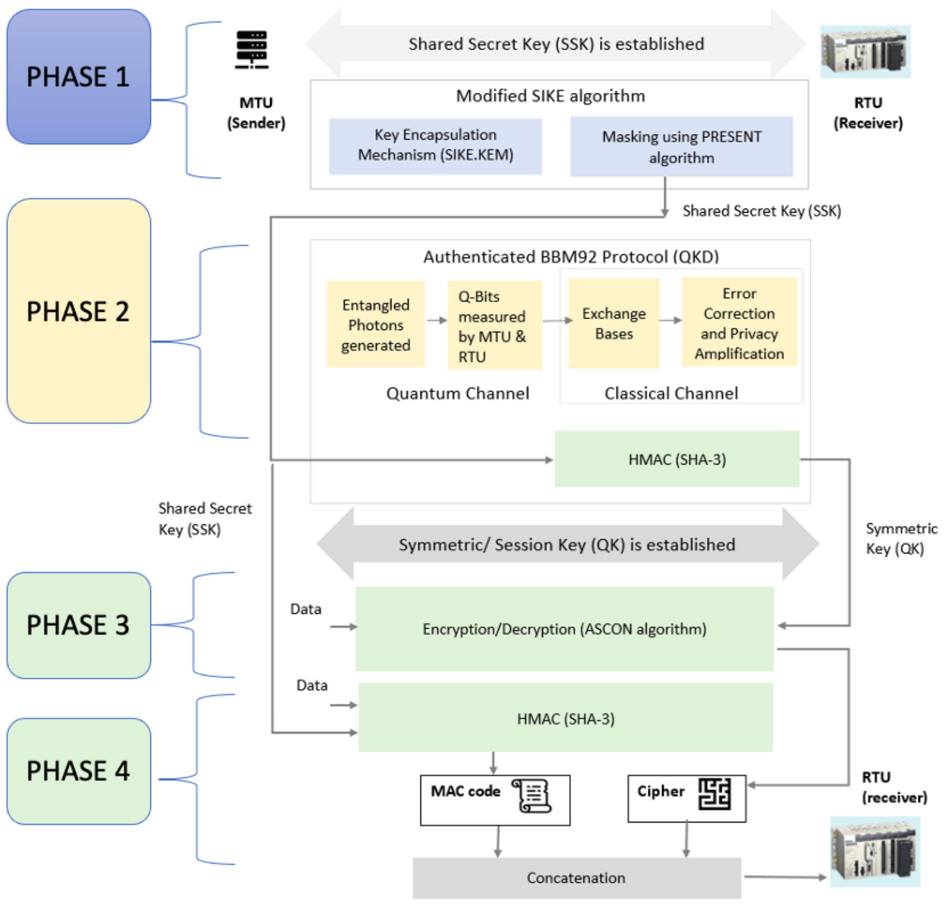

PHASE 1
PHASE 2
PHASE 3
PHASE 4
MTU (Sender)
RTU (Receiver)
Shared Secret Key (SSK) is established
Modified SIKE algorithm
Key Encapsulation Mechanism (SIKE.KEM)
Masking using PRESENT algorithm
Shared Secret Key (SSK)
Authenticated BBM92 Protocol (QKD)
Entangled Photons generated
Q-Bits measured by MTU & RTU
Exchange Bases
Error Correction and Privacy Amplification
Quantum Channel
Classical Channel
HMAC (SHA-3)
Shared Secret Key (SSK)
Symmetric Key (QK)
Symmetric/ Session Key (QK) is established
Data
Encryption/Decryption (ASCON algorithm)
Data
HMAC (SHA-3)
MAC code
Cipher
RTU (receiver)
Concatenation

Figure 4.3: Our proposed framework

4.2 Phase 1 and Phase 2: Secure Key Exchange

The key exchange processes are done in two separate stages. In the first stage, a shared secret key is generated using post-quantum SIKE, and in the second stage, the session key is generated using the quantum B92 protocol. Like any key exchange protocol, it has three phases, namely, key generation at the sender (assuming RTU), key distribution over the classical channel and quantum channel,and key extraction at the receiver unit (MTU) [129], [130].

4.2.1 Phase 1: Shared Secret key Generaction

This phase has modified the well-known post-quantum cryptography, namely, Supersingular Isogeny Key Encapsulation (SIKE) in the proposed scheme. The modified SIKE generates a shared secret key for message authentication in classical communication of BBM92 and during the exchange of ciphered data between RTU and MTU. The SIKE algorithm uses a set of defined public parameters as follows for key generation [129], [130].

1. The two positive integers e_2 and e_3 defined over a finite field F_{p^2} such that a prime $p = 2^{e_2} 3^{e_3} - 1$.
2. A public starte supersingular elliptic curve denoted as E_0/ F_{p^2}
3. A set of three x coordinated corresponding to $E_0[2^{e_2}]$ and $E_0[2^{e_3}]$, respectively.

The supersingular elliptic curve that is used as the starting point is be defined as $E_0/ F_{p^2} : y^2 = x^3 + 6x^2 + x$; and P and Q are two points on the curve. The three x-coordinates corresponding to $E_0[2^{e_2}]$ are x_{P_2} , x_{Q_2}, x_{R_2} where $R_2 = P_2 - Q_2$. And, other three x-coordinated $E_0[2^{e_3}]$ are x_{P_3} , x_{Q_3}, x_{R_3} where $R_3 = P_3 - Q_3$[129].

Key Generation: In this phase, the sender generates a key pair comprising of public key and secret key. The secret key sk_3 is randomly selected from the key space K_3 with range $\{ 0,1,\dots 2^{e_3} - 1\}$. The public key, pk_3, is generated by isogeny computation of secret key [129].

Key Encapsulation: After generating the key pair, the sender selects a random bit string $m \in M = \{ 0, 1 \}^n$ and concatenates with the public key pk_3. The concatenated version is fed to a hash function G to generate the digest r. Now, the digest r, the public key pk_3, and, the random string m is encrypted to derive the cipher pair (c_0, c_1). The cipher pair and the binary string is further fed to the hash function H to obtain the k-bit shared key K [129], [79].

Key Decapsulation: The receiver, after exchanging the public keys and the cipher pairs, decapsulates the ciphers and obtains the bit string m'. The bit string and the public key pk_3 is fed to the hash function G to obtain a digest r'. The digest is further processed based on isogeny computation to derive a cipher c_0'. The receiver then verifies and validates the derived cipher c_0' with the received cipher c_0. Once validated, the receiver concatenates the cipher pair (c_0, c_1) with m' and hashes it using a hash function H to get the shared key K. The three Hash functions The three hash functions F; G and H are SHA-3 derived function SHAKE256 as mentioned by NIST report [131]. Masking: This module contains modification of the SIKE protocol by adding a masking scheme. The sender and receiver establish a shared secret key (K) of 1500 bit. The key K is fed to the SHA3-512 hash function to generate a 512-bit digest. The digest is further chunked into four blocks of 128-bit each. Each pair is fed to the PRESENT encryption algorithm that generates a pair of 64-bit keys. These two sub-keys are concatenated to generate a compressed masked shared secret key of 128-bit. Algorithm 1 provides step-by-step instruction of the modified SIKE algorithm. Furthermore, Figure 4.4 provides an overview of masking in SIKE.

Algorithm 1 SHARED SECRET KEY GENERATION

The Modified SIKE Protocol

```
Data: Input: secret key, public key
Result: Output: Shared secret value
 function : KeyGeneration()
  $sk_3 \leftarrow K_3$
  $pk_3 \leftarrow isogen(sk_3)$
  return $(pk_3, sk_3)$
  function: KeyEncapsulation()
  $m \leftarrow 0, 1^n$
  $r \leftarrow G(m⯑ pk3)$
  $c \leftarrow Enc(pk3, m, r)$
  $c = (c_0, c_1)$
  $K \leftarrow H(m ⯑ c)$
  return($(c_0, c_1), K$)
  function: KeyDecapsulation()
  $m' \leftarrow Dec(sk_3, pk_3, c)$
  $r' \leftarrow G(m' ⯑ pk3)$
  $c'_0 \leftarrow isogen(r')$ if $c'_0 = c_0$ then
end
$K \leftarrow H(m ⯑ (c_0, c_1))$
  return K
  function: Masking() $K' \leftarrow SHA3 - 512(K)$
  $K'_0, K'_1, K'_2, K'_3 \leftarrow K$
  $SSK_0 \leftarrow PRESENT(K'_0, K'_1)$
  $SSK_1 \leftarrow PRESENT(K'_2, K'_3)$
  $SSK \leftarrow (SSK_0 ⯑ SSK_1)$
  return SSK
function: Main
  While (Session = END)
  $(pk_3, sk_3) \leftarrow$ KeyGeneration()
  $((c_0, c_1), K) \leftarrow$ KeyEncapsulation()
  $SSK \leftarrow$ Masking(K)
  $K \leftarrow$ KeyDecapsulation()
  Repeat State3
```

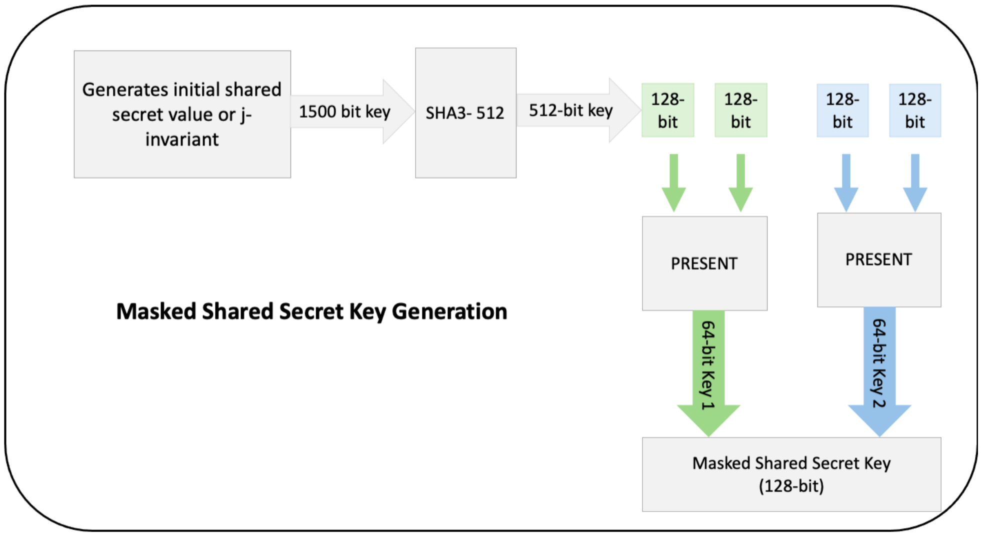

Generates initial shared secret value or j-invariant
1500 bit key
SHA3- 512
512-bit key
128-bit
128-bit
128-bit
128-bit
PRESENT
PRESENT
Masked Shared Secret Key Generation
64-bit Key 1
64-bit Key 2
Masked Shared Secret Key (128-bit)

Figure 4.4: The Masking Function of the modified SIKE algorithm

4.2.2 Phase 2: Session (Quantum) Key Generation

In the proposed scheme, BBM92[60] protocol is applied, an amalgam of E91[65] and BB84[58] protocols. It is a different version of BB84, which uses polarization entangled photon pairs. Moreover, it has similar steps except for the part that generates entangled photons. These pairs are then split up, with one photon from each pair being sent to the two parties, Alice and one to Bob. Alice and Bob randomly select to measure each photon they receive in one of two non-orthogonal bases, the horizontal or vertical basis. After a measurement run, where Alice and Bob have been measuring incoming photons for a certain time, they communicate publicly over a classical channel verifying the basis they used to measure each photon they received. Whenever the two units choose the same basis, each of them stores their measured value[52], [68]. The obtained measured value should be anti-correlated to form a secret key. Alice and Bob discard any measurement results with different bases because the results will be uncorrelated. This process is called sifting. Thus, they get a measurement result, convert their result to a classical bit, and sift their results to only those measured on the same basis. They use 10% of their measurements or the sifted bits to estimate the quantum bit error rate (QBER), and generate a final secure key from the rest of their measured bits after error correction and privacy amplification. The exchange of entangled photons between Alice and Bob based on a randomly chosen basis uses quantum channel[52], [68].

Furthermore, to generate raw key, the algorithm involves singlet state preparation device based on two X gates and Hadamard gate , that creates a singlet state. After this code runs, the qubits qr[0] and qr[1] that are entangled. After creating a singlet state, Charlie sends qubit qr[0] to Alice and qubit qr[1] to Bob, using registers with two quantum bits and four classical bits. Authenticated Quantum Session Key: While the sub-phases, including key sifting and estimation of quantum bit error rate, are performed using a quantum channel, error correction and privacy amplification is done via a classical channel. Thus, the research work have authenticated the data exchanged publicly. As shown in Figure 4.3, the proposed scheme is using HMAC based on SHA-3 to authenticate the classical communication during the BBM92 protocol to verify the data integrity and authenticity of the session key generated. The HMAC uses the shared secret key from modified SIKE to generate an authenticated session key.

4.3 Phase 3: Encryption

The RTU (sender) sends sensitive information to the MTU (receiver). The session (symmetric) key, generated by authenticated BBM92 protocol, is used by the RTU to encrypt the message and generate a cipher. The encryption algorithm used here is ASCON-128 [132], [133]. It is a lightweight authenticated encryption scheme that requires the following inputs, the plaintext P, a secret key SK with k bits, and a public message number (nonce). The output of ASCON is ciphertext C along with an authentication tag T [132].

4.4 Phase 4: Message authentication

Before encryption, the sender copies the sensitive data. It then feeds the copied message and the shared secret key (SSK) from modified SIKE to the HMAC based on SHA-3 to generate a message authentication code. Thus, in this step, the cipher from ASCON provides confidentiality, and the message authentication code provides integrity to the SCADA network [133].

4.5 Security Evaluation of the Proposed Framework

In the BBM92 protocol, Alice and Bob share a pair of photons from an EPR source. The photon pair generated and shared is entangled, as defined below[67].

$$\psi^{-} = \frac{1}{\sqrt{2}}(xy - yx) \tag{4.1}$$

or,

$$\psi^{+} = \frac{1}{\sqrt{2}}(xx + yy) \tag{4.2}$$

where x and y are two orthogonal polarization states. When Alice and Bob measure their photon with an x-y basis, their readings are correlated. Further, the thesis assumes the detection apparatus is trustable. Thus, the thesis work involves using a specific model for the behavior of the detection apparatus, which includes losses, and assume that the eavesdropper cannot modify the measurement apparatus beyond this model. The security of BBM92 and E91 protocol relies on the following three principles:

- Heisenberg's Uncertainty Principle: According to this principle [134], data is encoded in a qubit that holds the quantum properties such that any effort to measure or monitor the qubit disturbs its state in a detectable way. There is an inherent uncertainty when measuring a variable of a qubit.

- No Cloning Theorem: The theorem states that it is impossible to copy a qubit in a superposed or entangled state. Thus, an adversary can not create an independent and identical copy of an unmeasured qubit state [135].

- Bell's Inequality test: As per Bell's theory [136], no two particles can have anti-correlated or correlated value. Thus, no two particles can be entangled. And, the correlation value of two particles is set as -2. In case of entanglement, the qubit pair violates the Bell's theory, and generates a correlation value of around $-2\sqrt{2}$. QKD based on entanglement use the correlation value to detect an absence of entangled pairs, or disturbed state of entanglement caused due to an adversary [65].

In any realistic communication system errors are bound to occur, and some form of error correction is required. In quantum cryptography the errors typically arise from technological imperfections in the optics and detectors, but can also come from eavesdropping. In order to achieve noise-free communication these errors must be corrected, and this can be done through public discussion using error-detection codes. These codes allow the parties involved in the communication to detect errors in the exchanged keys, which could be introduced by an eavesdropper trying to intercept and observe the keys. If an error is detected, the parties can abort the key exchange and start over, reducing the risk of the exchanged keys being compromised[137].

QKD requires a direct line-of-sight between the sender and the receiver and is still an emerging technology. There are a few challenges for QKD to be deployed in real-time scenarios, in SCADA systems, such as the following.

1. Key Rate: In quantum cryptography, errors arise from technological imperfections in the optics and detectors and other environmental factors. Thus, obtaining a high key rate can be challenging in a hardware settings, mainly for higher volumes of network trafic. Therefore, a real-time QKD requires a detector with high eficiency and short dead time [11], [138].

2. Distance: Noise in the quantum channel is directly proportional to the distance between two units performing QKD. Thus, in a real-time scenario, the framework needs low-noise single-photon detection to tolerate losses [11], [138].

However, a few commercial QKD devices are currently small and cost-effective enough to be integrated into a SCADA network. ID Quantique collaborated with Hitachi ABB to provide security for critical infrastructure processes [139]. The solution includes hardware-based QKD and QRNG solutions such as Clavis XG QKD System [140] and Clavis^{300} Quantum Cryptography Platform [141] that has integrated QKD protocol with high key transmission rate on complex network topologies with Ethernet encryptors. Since RTUs are resource constrained, the QKD system devices can be installed and linked to the Modbus-RTU or Modbus (TCP/IP) RTU to perform secure quantum exchange between RTU and MTU units [139].

Table 4.1 lists the following attacks that the proposed scheme should be able to thwart. However, the thesis does not focus on Photon number splitting and Trojan horse attacks since they exploit vulnerabilities in the physical hardware. Both attacks compromise the integrity of the QKD device. It requires safeguard measures, mainly, tamper-evident packaging against attacks targeting physical components.

1. Intercept-Resend Attack: The attacker measures the quantum states (photons) sent by the sender and then sends replacement states to Receiver, prepared in the state measured by the attacker. However, the algorithm uses BBM92, which detects the noise generated by the adversary while trying to read the qubits [142], [143]. In the BBM92 protocol, the thesis involves detection of an eavesdropper by using quantum state tomography to verify the state of the entangled particles. This technique involves measuring the state of the entangled particles and using the measurement results to reconstruct the state of the particles. If the state of the particles has been disturbed by an external observer, such as an eavesdropper, the reconstruction will be inaccurate, and this can be used to detect the presence of an eavesdropper [67],[143].

2. Random-Substitute Attack: Here, the attacker copies the qubit and replaces the states with the copied state. This will again get detected by BBM92 protocol due to the No Cloning theorem [135]. To protect against a random-substitute attack, the protocol uses BBM92 protocol along with other security measures, such as authentication protocols, to ensure that Alice and Bob can detect when their messages have been tampered with. In addition, BBM92 use error-correction and privacy amplification techniques to ensure that the final secret key is secure even if some errors or noise are introduced during the key generation process.

3. Traditional Man-in-the-Middle(MITM) Attack: MITM attacks can be performed in a couple of ways. The traditional MITM attacks do not work on QC systems because laws of quantum mechanics step in. With traditional MITM attacks, an adversary intercepts the transmitted messages and send a copy in its place [143]. However this is impossible due to the No cloning theorem and the Heisenberg uncertainty principle. The traditional MITM comprises of an adversary pretending to be ’Sender’ to the Receiver and ’the Receiver’ to Sender. The adversary would then communicate with both the Sender and Receiver simultaneously thereby obtaining two keys, one for the Sender and one for the Receiver. The Sender’s key would be used to decrypt a message from Alice then re-encrypted by the Receiver’s key[142]. Thus, the adversary

intercepts the transmitted messages from the sender and sends a copy in its place to the receiver. Again, traditional MITM attack fails due to laws of quantum physics, such as the Heisenberg's Uncertainty Principle and the No Cloning Theorem, exploited in the algorithm[142], [143]. Further, the protocol is auhenticated by using a pre-shared key generated from SIKE along with SHA-3 as HMAC to authenticate both units.

4. Quantum Man-in-the-Middle Attack based on spoofing: The adversary pretends to be the sender to the receiver, and vice versa. The adversary would then communicate with both the sender and receiver simultaneously and thus, obtain two keys, one for the sender and one for the receiver. The sender's key would be used to decrypt a message from the sender and then re-encrypted by the receiver's key. This type of attack is possible. the proposed scheme uses an initial shared secret key derived from various parameters to prevent this type of attack, including the unit's id. the module uses SIKE in the algorithm to generate the initial shared secret key. Furthermore, the module uses this shared secret key for message authentication [144].

5. Quantum Man-in-the-Middle Attack based on coherent pulses: It involves the method through which photons are transmitted. The adversary may split a single proton from the burst without detection. They then stores the stolen photons until the basis used to create them is announced. This can be avoided using EPR or entangled photons. However, since it will try to split the proton, it will affect the paired proton. Hence, the module uses the entanglement-based BBM92 protocol in the algorithm[144].

6. Denial of Service Attack on Quantum communication channel: DoS attack in QKD is launched in two ways, namely, 1) by compromising the quantum cryptographic hardware, and 2) by introducing extra noise in the QKD system[145]. Our algorithm cannot guarantee resistance against hardware attacks. However, it will defend against the later DoS attack. In the later attack, the adversary inserts noise below the threshold to be acceptable in the communication system, such that the noise would be indistinguishable from eavesdropping. Furthermore, both RTU and MTU will either increase the

threshold or discard photons. That makes eavesdropping more successful for the attacker. However, in the BBM92 protocol, they have to be correlated due to entangled pairs of photons. So, any noise in the channel will disrupt their correlation. Thus, eavesdropping will be detected.

7. Brute Force Attack based on Shor's and Grover's algorithms: The adversary launches a brute force attack using Shor's algorithm on quantum hardware and a search algorithm based on Grover's to derive the secret key. For Shor's, the module uses the BBM92 protocol. And, for Grover's search attack, it applies the post-quantum cryptography, SIKE [142], [143].

Table 4.1 lists the following attacks that the proposed scheme should be able to thwart.

4.6 Security proof of the proposed framework detecting noise in the channel

Lemma 4.6.1. The framework uses the BBM92 protocol that can detect a presence of an adversary launching an Intercept-resend attack, a Random-Substitute attack as well as a Man-in-the-Middle attack. Assumption: An adversary measures each qubit towards Bob by randomly using a basis and obtains a measurement result. The adversary sends an identical photon to Bob. Alternatively, the adversary after measuring both qubits meant for Alice and Bob, substitutes with their own state.

Proof. When the adversary(E), measures the qubit, it collapses the superposition of the ψ state and ruins the coherence and fails to obtain perfect correlation values measured in any basis. Alice, denoted as A, and Bob, denoted as B, measures the qubits using their basis. However, due to No Cloning theorem and Bell's Inequality test, this disturbance in the state of qubit is detected by both units.

To determine the generate raw key is secure, Alice and Bob set the following thresholds[68].

1. Calculation of Quantum Bit Error Rate (QBER) as shown in equation 4.3, where NoE is the number of errors, $total_{bits}$ is the total number of bits transmitted

Table 4.1: Attacks defended by the proposed scheme.

Attacks detected/prevented	Description	Countermeasures
Intercept and Resend Attack	The Attacker measures the photons sent by Sender and replaces the states with a new one and sends it to the Receiver.	Using quantum key distribution protocol.
Random-substitute Attack	The Attacker measures the photons sent by Sender, copies and replaces the states, and sends them to the Receiver.	Using Quantum key distribution protocol.
Traditional Man-in-the-Middle Attack	Attacker intercepts the transmitted messages from sender and sends a copy in its place to the receiver.	Using Quantum key distribution protocol.
Quantum Man-in-the-Middle Attack based on spoofing	Attacker pretends to be ”Sender” to the Receiver and ”Receiver” to Sender.	It can be prevented if the units can be authenticated. Both sender and receiver have an initial shared secret for secure message and unit authentication.
Quantum Man-in-the-Middle Attack based on coherent pulses	Attacker may split a single proton from the burst without detection, stores them until the basis to create them are announced by both legitimate parties.	EPR pairs are used to avoid this type of attack.
Denial of Service Attack on Quantum communication channel	It inserts noise below the threshold to be acceptable in the communication system.	It discards the keys if it includes any noise. The photons need to correlate, i.e., no eve is present. If not, the key is not legitimate.
Brute Force attack using Shor’s algorithm	Attacker launches a brute force attack using Shor’s algorithm on quantum hardware to derive the secret key.	Entanglement-based quantum key distribution protocol is used.
Brute Force attack using Grover’s algorithm	Attacker launches a search algorithm on quantum hardware to derive the private key.	Thus, post-quantum algorithm, SIKE, as public key cryptography is implemented.

and e is the QBER.

$$e = \frac{NoE}{total_{bits}} \tag{4.3}$$

2. Calculation of the maximum Shannon Information between Alice and the adversary to verify the following inequation. If the below inequation holds true, the key

established between Alice and Bob is not secret and thus, discarded.

$$I^{max}_{A,E} > I^{max}_{A,B} \tag{4.4}$$

The Shannon information between Alice and the adversary can be determined by the equation given below, where e is the QBER.

$$I^{A,E}_{max} = \frac{2}{\ln 2^{e}} + O(e^{2}) \approx \frac{2}{\ln 2^{e}} \tag{4.5}$$

Further, based on Gisin et al's proof [146], the mutual information between Alice and Bob is calculated as follows.

$$I(A, B) = 1 + e\log_2(e) + (1 - e)\log_2(1 - e) \tag{4.6}$$

A secure secret key agreement is only established if $I(A, B) > I(A, E)$ or, $I(A, B) > I(B, E)$.

Further, as the error rate e increases, $I(A, E)$ or $I(B, E)$ significantly increases and $I(A, B)$ decreases. Eventually, when plotted on a graph, these two curves intersect at a certain error rate such that;

$$e = \frac{1 - \sqrt{\frac{1}{2}}}{2} \approx 14.6 \tag{4.7}$$

The derived value of error rate in above equation is the error threshold for the BBM92 protocol. Once the error rate is beyond the tolerable value, the key obtained is not secret even if error correction and privacy amplification is followed. Thus, the obtained key is discarded. □

Further, the probability of the adversary selecting the right base to obtain the correct measurements is 50%. Moreover, the probability of selecting the wrong basis is 50% and the outcome of obtaining incorrect bits is 25%. Thus, the presence of an adversary injects a total error rate of 25%, which is more than the threshold value and therefore, the error is detected by both Alice and Bob.

4.7 Performance Evaluation

This section now evaluates the performance of the quantum-robust scheme based on quantum and post-quantum scheme for secure RTU and MTU communication.

It has implemented the proposed algorithm in Python 3.6 (Spyder IDE) and IBM Qiskit [147], an open-source software development kit for simulating quantum circuits, pulses, and algorithms. It performed the following comparative analysis. Table 4.2 provides a summary of comparative analysis between current and proposed security algorithm along the following aspects:

- Comparative Analysis between RSA and modified SIKE algorithm
- Comparative Analysis between E91 and BBM92 protocol
- Comparative Analysis between AES and ASCON
- Comparative analysis between SHA-1 used in ECDSA and HMAC based on SHA-3

Each comparative analysis uses different sets of variables to validate the hypothesis. The research work involves simulation of the proposed scheme 30 times, thus generating 30 unique set of keys and ciphers, to be used for performance evaluation of algorithms.

Table 4.2: Comparative analysis between Traditional Security Algorithm and Proposed Security Algorithm

Variables Measured	Traditional Security Algorithm			Proposed Security Algorithm		
	RSA	AES	SHA-1 in ECDSA	m-SIKE-SSK	BBM92 + ASCON	HMAC using SHA3
Computational Speed (microseconds)	Private Key: 223.83 Public Key: 20.78	2258.861	2.53	1713.86	4400.54	36.9

Randomness(%)	84.37	87.5	N/A	87.5	87.5(BBM92)
					81.25(ASCON)
						N/A

Memory (Key size in bits)	1024	128	256	128	236	256
Known Vulnerability	Brute Force based on Shor's algorithm	Brute Force based on Grover's algorithm	Cracked by Google	Resistant against Quantum attack	Resistant against Quantum attack	Resistant against Quantum attack

4.7.1 Comparative Analysis between RSA and modified SIKE algorithm

The module has used the three variables for validating the hypothesis, mainly, Key Size of SIKE and RSA keys, Randomness of the shared secret key generated vs RSA key pairs, Execution Time to generate masked SSK vs RSA keys.

Key Size and CPU time to generate keys

The shared secret key generated by the proposed algorithm is 128-bits. The RSA-1024 public key (e, n) is (2-bit, 1024-bit), and the RSA-1024 private key (d, n) is (1024-bit,

Table 4.3: CPU/Execution Time of RSA and modified SIKE

Algorithms	Execution Time (in microseconds)
m-SIKE-SSK	1713.86
SIKE-SSK	1293022.696
RSA-1024 private key	223.83
RSA-1024 public key	20.78

1024-bit). Thus, the thesis concludes that the modified SIKE shared secret key size is much smaller than RSA key pairs.

The experiment involves simulation of the modified SIKE algorithm to generate the shared secret key. It then calculated the mean value of the execution time of SIKE, and also measured the CPU cycles of SIKE and speed of RSA on 1.8 GHz Intel Core i5. In SIKE, the key generation runs in 61029304 cycles, the encapsulation runs in 100200351 cycles and the decapsulation runs in 106600562 cycles. Based on the evaluation, the average execution time to generate a shared secret key is 1713.86 microseconds, and the CPU time to generate RSA key pairs is also calculated. For 43739 RSA-1024 private key operations, it took 9.79 seconds. Thus, for 1 RSA-1024 private key operation, it will take $\frac{9.79}{(43739)}$ = 0.00022383 seconds = 223.83 μ seconds. Whereas, for 1 RSA-1024 private key operation, it will take = $\frac{9.79}{(43739)}$ = 0.00022383 seconds = 223.83 μ seconds.

Randomness of SIKE(SSK keys) and RSA keys

This module has performed 16 NIST statistical tests [121] on SIKE and RSA key pairs. Since SIKE generated one shared secret key, the number of tests is 16. For RSA, there are two keys; public and private keys. Thus, each key has undergone 16 tests. Thus, it performed 32 tests on RSA key pairs. It observed that the Randomness of modified SIKE SSK is 87.5%, whereas the average randomness score of RSA key pairs is 84.37 %, as shown in Table 4.4.

4.7.2 Comparative Analysis between SIKE and modified-SIKE

This thesis also includes a comparative analysis between SIKE and the modified SIKE (m-SIKE), based on the key size, randomness-based NIST statistical test [121]

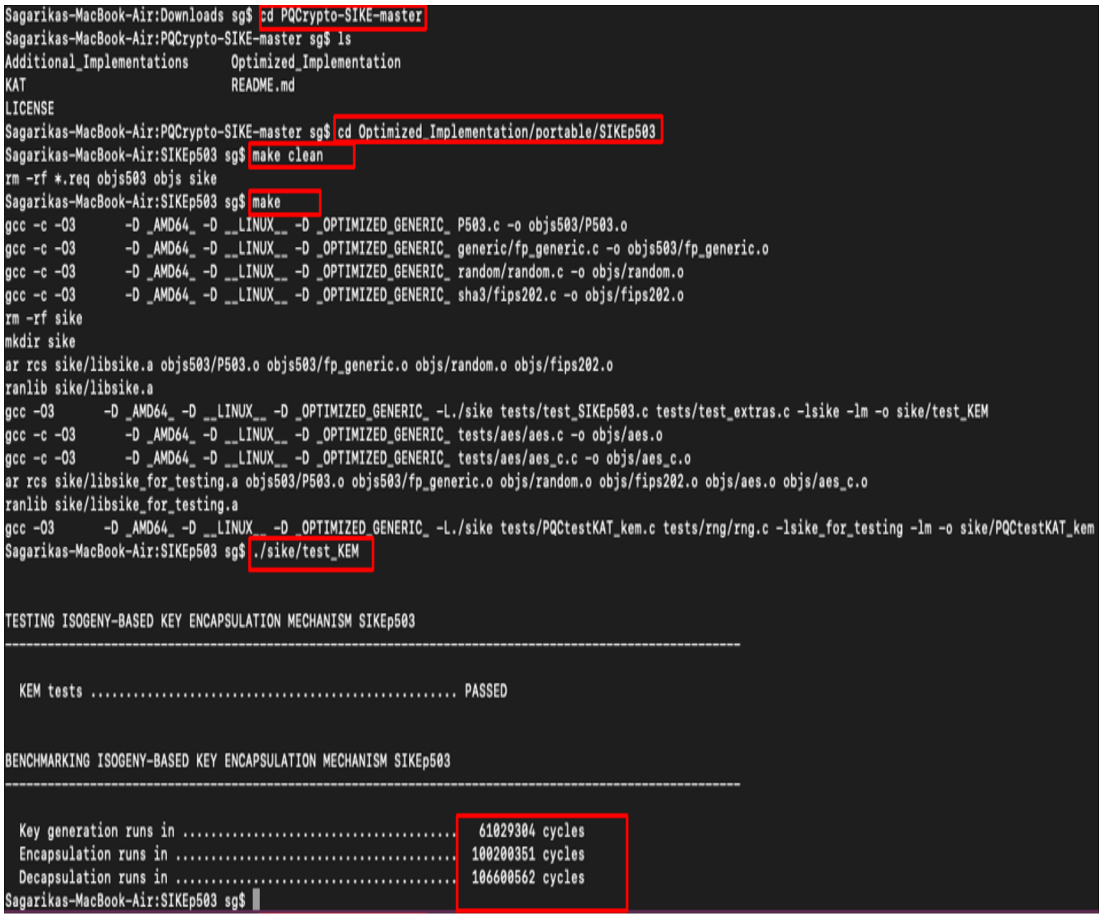
Table 4.4: Percentage(%) of NIST Randomness passed tests

Keys tested	Percentage(%) of NIST Randomness passed test	NIST Randomness Test (16 tests) passed
m-SIKE Shared secret key	87.5	14
SIKE shared secret key	81.25	13
RSA public key	93.75	15
RSA private key	75	12

Table 4.5: CPU/Execution Time of RSA and modified SIKE

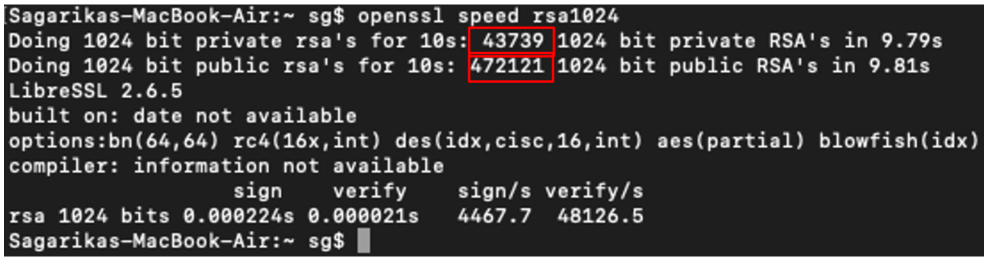
Algorithms	Execution Time (in microseconds)
m-SIKE-SSK	1713.86
SIKE-SSK	1293022.696
RSA-1024 private key	223.83
RSA-1024 public key	20.78

and execution time to generate the shared secret key. The average execution time to generate SIKE keys is 1293022.696 microseconds, significantly higher than that of the proposed m-SIKE. Our proposed m-SIKE takes 1713.86 microseconds to generate the shared secret key. Moreover, the average key size of SIKE is 1500 bits, and the average key size of m-SIKE is 128 bits. It performed 16 NIST statistical tests on both SIKE and m-SIKE. The key generated by SIKE failed the Longest Run of Ones in a Block test, Non-Overlapping Template Matching Test, and, Maurer's Universal Statistical test. However, the m-SIKE failed two tests, mainly, Maurer's Universal statistical test and Approximate Entropy test. Thus, the randomness of SIKE is 81.25% and that of m-SIKE is 87.25%. Thus, the compressed-key from m-SIKE increases the resilience and feasibility of the proposed scheme which is a key component of the SCADA security requirements.



```
Sagarikas-MacBook-Air:Downloads sg$ cd PQCrypto-SIKE-master
Sagarikas-MacBook-Air:PQCrypto-SIKE-master sg$ ls
Additional_Implementations      Optimized_Implementation
KAT                             README.md
LICENSE
Sagarikas-MacBook-Air:PQCrypto-SIKE-master sg$ cd Optimized_Implementation/portable/SIKEp503
Sagarikas-MacBook-Air:SIKEp503 sg$ make clean
rm -rf *.req objs503 objs sike
Sagarikas-MacBook-Air:SIKEp503 sg$ make
gcc -c -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ P503.c -o objs503/P503.o
gcc -c -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ generic/fp_generic.c -o objs503/fp_generic.o
gcc -c -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ random/random.c -o objs/random.o
gcc -c -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ sha3/fips202.c -o objs/fips202.o
rm -rf sike
mkdir sike
ar rcs sike/libsike.a objs503/P503.o objs503/fp_generic.o objs/random.o objs/fips202.o
ranlib sike/libsike.a
gcc -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ -L./sike tests/test_SIKEp503.c tests/test_extras.c -lsike -lm -o sike/test_KEM
gcc -c -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ tests/aes/aes.c -o objs/aes.o
gcc -c -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ tests/aes/aes_c.c -o objs/aes_c.o
ar rcs sike/libsike_for_testing.a objs503/P503.o objs503/fp_generic.o objs/random.o objs/fips202.o objs/aes.o objs/aes_c.o
ranlib sike/libsike_for_testing.a
gcc -O3       -D _AMD64_ -D __LINUX__ -D _OPTIMIZED_GENERIC_ -L./sike tests/PQCtestKAT_kem.c tests/rng/rng.c -lsike_for_testing -lm -o sike/PQCtestKAT_kem
Sagarikas-MacBook-Air:SIKEp503 sg$ ./sike/test_KEM


TESTING ISOGENY-BASED KEY ENCAPSULATION MECHANISM SIKEp503
--------------------------------------------------------------------------------

  KEM tests .................................................. PASSED


BENCHMARKING ISOGENY-BASED KEY ENCAPSULATION MECHANISM SIKEp503
--------------------------------------------------------------------------------

  Key generation runs in .......................................   61029304 cycles
  Encapsulation runs in ........................................  100200351 cycles
  Decapsulation runs in ........................................  106600562 cycles
Sagarikas-MacBook-Air:SIKEp503 sg$
```

Figure 4.5: CPU Cycles of SIKE on on 1.8 GHz Intel Core i5.



```
Sagarikas-MacBook-Air:~ sg$ openssl speed rsa1024
Doing 1024 bit private rsa's for 10s: 43739 1024 bit private RSA's in 9.79s
Doing 1024 bit public rsa's for 10s: 472121 1024 bit public RSA's in 9.81s
LibreSSL 2.6.5
built on: date not available
options:bn(64,64) rc4(16x,int) des(idx,cisc,16,int) aes(partial) blowfish(idx)
compiler: information not available
                  sign    verify    sign/s verify/s
rsa 1024 bits 0.000224s 0.000021s   4467.7  48126.5
Sagarikas-MacBook-Air:~ sg$
```

Figure 4.6: Speed of RSA on 1.8 GHz Intel Core i5.

4.7.3 Randomness of SIKE(SSK keys) and RSA keys

This section includes the results of 16 NIST statistical tests [121] on SIKE and RSA key pairs. Since SIKE generated one shared secret key, the number of tests is 16. For RSA, there are two keys; public and private keys. Thus, each key has undergone 16 tests. Thus, this module involves 32 tests on RSA key pairs. The randomness of modified SIKE SSK is 87.5%, whereas the average randomness score of RSA key pairs is 84.37 %, as shown in Table 4.4.

4.8 Comparative Analysis between E91 and BBM92 protocol

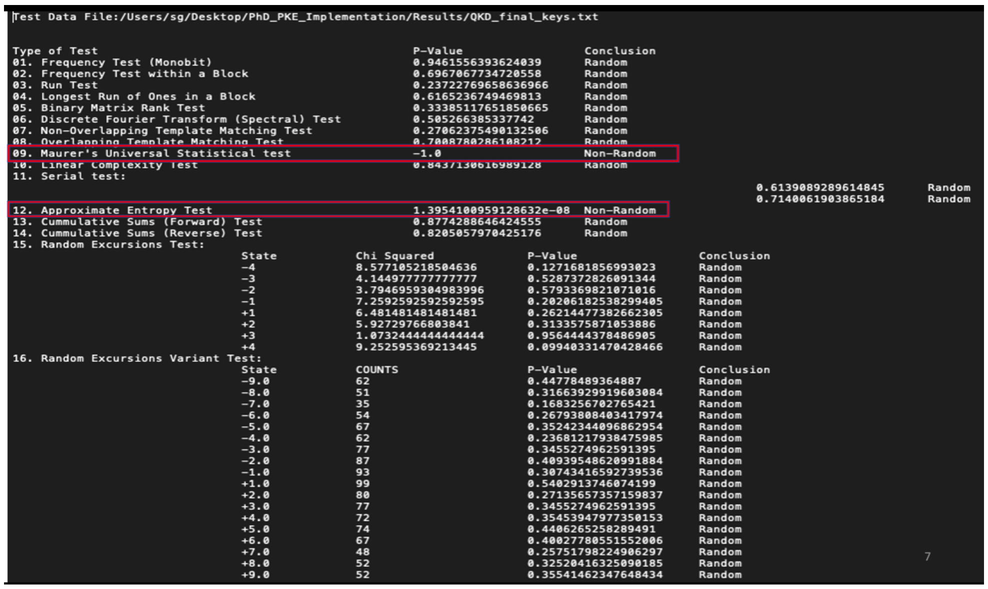
As per Central Limit theorem (CLT), the sample size (N) ≥ 30 is suficient for the theorem to hold[148]. Thus, the thesis simulated BBM92 and E91 protocol, 30 times, on IBM Qiskit to generate 30 quantum keys for each algorithm. The version of the qiskit packages are; qiskit-terra’: ’0.18.3’, ’qiskit-aer’: ’0.9.1’, ’qiskit-ignis’: ’0.6.0’, ’qiskit-ibmq-provider’: ’0.18.0’, ’qiskit-aqua’: ’0.9.5’, ’qiskit’: ’0.32.0’, ’qiskit-nature’: ’0.2.2’, ’qiskit-finance’: ’0.2.1’, ’qiskit-optimization’: ’0.2.3’, ’qiskit-machine-learning’: ’0.2.1’. The thesis involves the comparison of BBM92 and E91 based on the five variables, namely, randomness in the keys generated, final key size, intrusion detection variable (CHSH value in E91 vs QBER in BBM92), percentage of Alice and Bob’s key leakage, and, execution time to generate the quantum session key.

4.8.1 Randomness in the keys generated

This section shows 16 NIST statistical tests[121] on BBM92 and E92 protocol. The randomness of E91 and BBM92 keys are same (87.5%). Out of 16 tests, both of them failed Maurer’s Universal statistical test[121] and Approximate Entropy test [121] as shown in Figure 4.7 and 4.8.

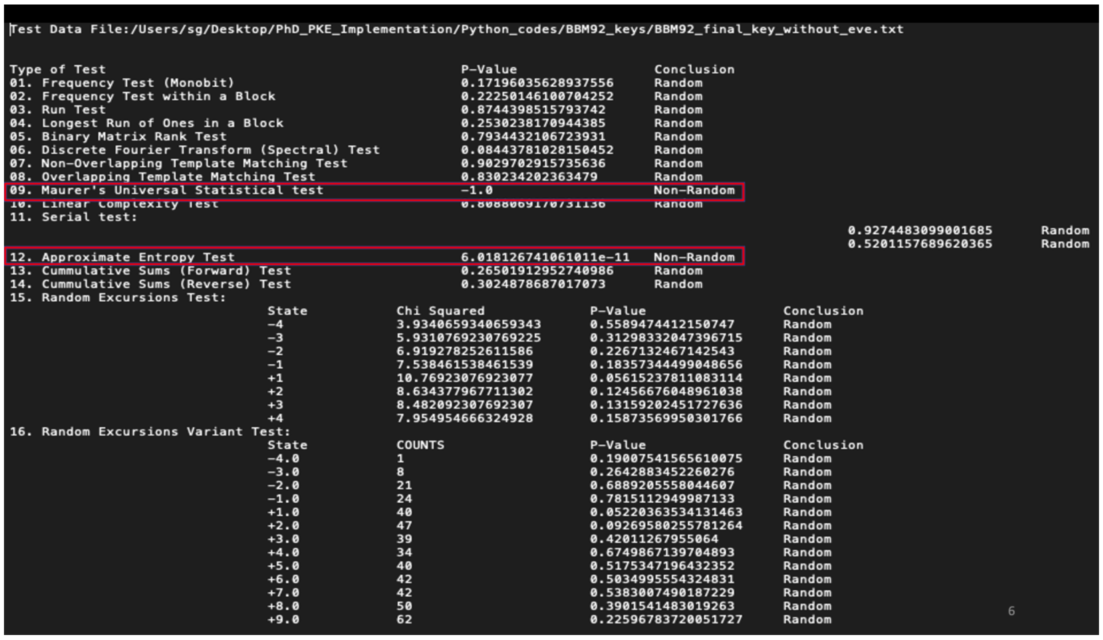
4.8.2 Final Key size, Intrusion Detection variable (CHSH value in E91 vs QBER in BBM92), and Percentage of Alice and Bob’s key leakage

The module simulates the E91 and BBM92 protocol 30 times, on IBM qiskit, and the module obtains the same raw key size. However, the final key size of E91 is approximately 110 bits and that of BBM92 is approximately 108 bits. In E91, the percentage of Alice’s key leakage is 93.02% and that of Bob’s key leakage is 92.968%. In BBM92, the percentage of Alice’s key leakage is 74.70% and that of Bob’s key leakage is 77.328%. Moreover, the E91 uses CHSH value or Bell’s Inequality test and, BBM92 uses QBER to detect the intrusion detection. Table 4.6 provides a comparison



```
Test Data File:/Users/sg/Desktop/PhD_PKE_Implementation/Results/QKD_final_keys.txt

Type of Test                                        P-Value                 Conclusion
01. Frequency Test (Monobit)                        0.9461556393624039      Random
02. Frequency Test within a Block                   0.6967067734720558      Random
03. Run Test                                        0.23722769658636966     Random
04. Longest Run of Ones in a Block                  0.6165236749469813      Random
05. Binary Matrix Rank Test                         0.33385117651850665     Random
06. Discrete Fourier Transform (Spectral) Test      0.505266385337742       Random
07. Non-Overlapping Template Matching Test          0.27062375490132506     Random
08. Overlapping Template Matching Test              0.7008780286108212      Random
09. Maurer's Universal Statistical test             -1.0                    Non-Random
10. Linear Complexity Test                          0.8437130616989128      Random
11. Serial test:
                                                                                                0.6139089289614845      Random
                                                                                                0.7140061903865184      Random
12. Approximate Entropy Test                        1.3954100959128632e-08  Non-Random
13. Cummulative Sums (Forward) Test                 0.8774288646424555      Random
14. Cummulative Sums (Reverse) Test                 0.8205057970425176      Random
15. Random Excursions Test:
                                State           Chi Squared             P-Value                 Conclusion
                                -4              8.577105218504636       0.1271681856993023      Random
                                -3              4.144977777777777       0.5287372826091344      Random
                                -2              3.7946959304983996      0.5793369821071016      Random
                                -1              7.2592592592592595      0.20206182538299405     Random
                                +1              6.481481481481481       0.26214477382662305     Random
                                +2              5.92729766803841        0.3133575871053886      Random
                                +3              1.0732444444444444      0.9564444378486905      Random
                                +4              9.252595369213445       0.09940331470428466     Random
16. Random Excursions Variant Test:
                                State           COUNTS                  P-Value                 Conclusion
                                -9.0            62                      0.44778489364887        Random
                                -8.0            51                      0.31663929919603084     Random
                                -7.0            35                      0.1683256702765421      Random
                                -6.0            54                      0.26793808403417974     Random
                                -5.0            67                      0.35242344096862954     Random
                                -4.0            62                      0.23681217938475985     Random
                                -3.0            77                      0.3455274962591395      Random
                                -2.0            87                      0.40939548620991884     Random
                                -1.0            93                      0.30743416592739536     Random
                                +1.0            99                      0.5402913746074199      Random
                                +2.0            80                      0.27135657357159837     Random
                                +3.0            77                      0.3455274962591395      Random
                                +4.0            72                      0.35453947977350153     Random
                                +5.0            74                      0.4406265258289491      Random
                                +6.0            67                      0.40027780551552006     Random
                                +7.0            48                      0.25751798224906297     Random      7
                                +8.0            52                      0.32520416325090185     Random
                                +9.0            52                      0.35541462347648434     Random
```

Figure 4.7: NIST statistical test results of E91 protocol



```
Test Data File:/Users/sg/Desktop/PhD_PKE_Implementation/Python_codes/BBM92_keys/BBM92_final_key_without_eve.txt

Type of Test                                        P-Value                 Conclusion
01. Frequency Test (Monobit)                        0.17196035628937556     Random
02. Frequency Test within a Block                   0.22250146100704252     Random
03. Run Test                                        0.8744398515793742      Random
04. Longest Run of Ones in a Block                  0.2530238170944385      Random
05. Binary Matrix Rank Test                         0.7934432106723931      Random
06. Discrete Fourier Transform (Spectral) Test      0.08443781028150452     Random
07. Non-Overlapping Template Matching Test          0.9029702915735636      Random
08. Overlapping Template Matching Test              0.830234202363479       Random
09. Maurer's Universal Statistical test             -1.0                    Non-Random
10. Linear Complexity Test                          0.8088069170731136      Random
11. Serial test:
                                                                                                0.9274483099001685      Random
                                                                                                0.5201157689620365      Random
12. Approximate Entropy Test                        6.018126741061011e-11   Non-Random
13. Cummulative Sums (Forward) Test                 0.26501912952740986     Random
14. Cummulative Sums (Reverse) Test                 0.3024878687017073      Random
15. Random Excursions Test:
                                State           Chi Squared             P-Value                 Conclusion
                                -4              3.9340659340659343      0.5589474412150747      Random
                                -3              5.9310769230769225      0.31298332047396715     Random
                                -2              6.919278252611586       0.2267132467142543      Random
                                -1              7.538461538461539       0.18357344499048656     Random
                                +1              10.76923076923077       0.05615237811083114     Random
                                +2              8.634377967711302       0.12456676048961038     Random
                                +3              8.482092307692307       0.13159202451727636     Random
                                +4              7.954954666324928       0.15873569950301766     Random
16. Random Excursions Variant Test:
                                State           COUNTS                  P-Value                 Conclusion
                                -4.0            1                       0.19007541565610075     Random
                                -3.0            8                       0.2642883452260276      Random
                                -2.0            21                      0.6889205558044607      Random
                                -1.0            24                      0.7815112949987133      Random
                                +1.0            40                      0.05220363534131463     Random
                                +2.0            47                      0.09269580255781264     Random
                                +3.0            39                      0.42011267955064        Random
                                +4.0            34                      0.6749867139704893      Random
                                +5.0            40                      0.5175347196432352      Random
                                +6.0            42                      0.5034995554324831      Random
                                +7.0            42                      0.5383007490187229      Random
                                +8.0            50                      0.3901541483019263      Random      6
                                +9.0            62                      0.22596783720051727     Random
```

Figure 4.8: NIST statistical test results of BBM92 protocol

based on the above discussed variables.

Table 4.6: Comparative Analysis of BBM92 and E91 based on; Final Key size, Intrusion Detection variable (CHSH value in E91 vs QBER in BBM92), and Percentage of Alice and Bob's key leakage

Variable Measured	Raw Key Size in bits	Final Key Size (with/without Eve) in bits	CHSH value without Eve	CHSH value with Eve	% of Alice's key leakage	% of Bob's key leakage	Mismatched bits	QBER
Mean value of E91 variables	500	110.633	-2.8464	-1.44556	93.0203	92.968	13.66	N/A
Mean value of BBM92 variables	500	108.66	N/A	N/A	74.7023	77.32833	64.05	26.077

4.8.3 Execution Time to generate BBM92 and E91 keys

This section provides results based on simulation of each protocol 30 times, thus, the thesis obtains 30 execution times. the thesis calculates the mean execution for each algorithm, and the thesis concludes based on the obtained calculation that the mean time to generate E91 key is 255.875 μ seconds, and that of BBM92 key is 148.941 μ seconds. Thus, as shown in Table 4.7, the execution time for E91 is 1.71 times higher than that of BBM92 protocol.

Table 4.7: Comparison based on Execution Time of BBM92 and E91

Algorithms	Mean Execution Time(in micro-seconds)
QKD E91	255.875
BBM92	148.941

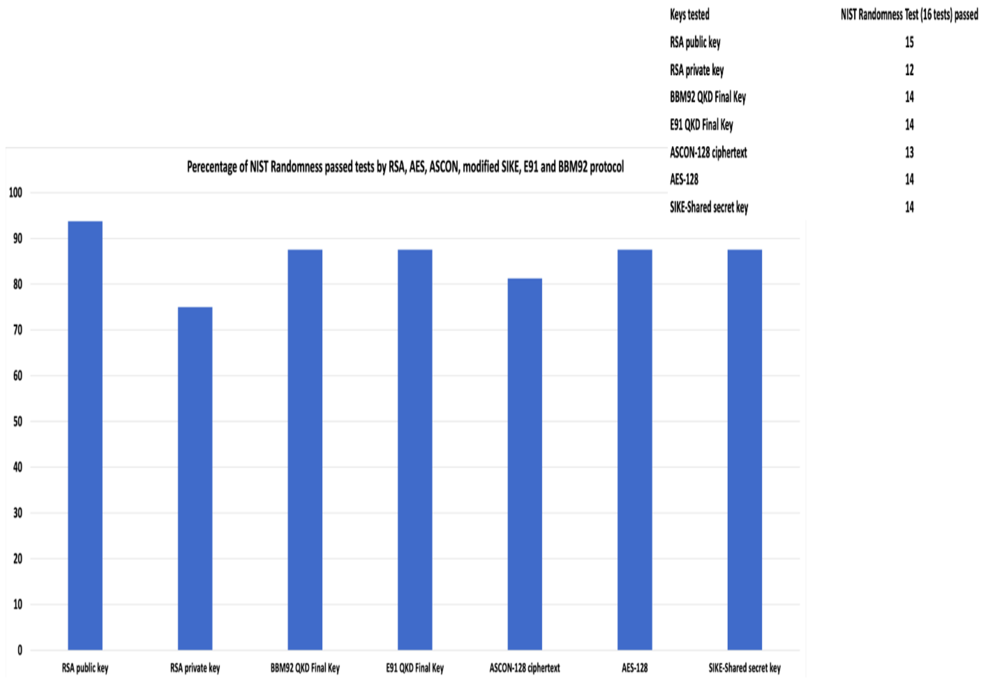
4.9 Comparative Analysis between AES and ASCON algorithm

AGA-12 is the current standard of SCADA networks. It uses AES-128 for encrypting sensitive data to be exchanged between control units. However, in the proposed scheme, ASCON-128 [149] algorithm is selected as an alternative as it is a lightweight and robust algorithm to generate cipher, eficient for resource constrained devices [149]. NIST has selected ASCON as the primary choice for lightweight authenticated encryption and is a finalist in the NIST Lightweight Cryptography competition[133], [150]. This section simulated ASCON and AES on Python 3.6 and generated their cipher. This algorthim steps involve feeding AES-128 and ASCON-128 cipher to NIST randomness statistical tools to measure their randomness. It also calculated the average execution time to generate the cipher for each algorithm. Figure 4.9 shows the randomness of ASCON-128 bit cipher is 81.25%, whereas the randomness score of AES-128 is 87.5%. Table 4.8 shows that, on the average, ASCON-128 costs

4251.599 μ seconds, and AES-128 costs 2258.861 μ seconds. Thus, the time to execute the ASCON-128 algorithm is 1.88 times higher than AES-128.

Table 4.8: Comparative Analysis based on execution Time(ASCON vs AES)

Algorithms	Mean Execution Time(in micro-seconds)
ASCON-128	4251.599
AES-128	2258.861



Keys tested	NIST Randomness Test (16 tests) passed
RSA public key	15
RSA private key	12
BBM92 QKD Final Key	14
E91 QKD Final Key	14
ASCON-128 ciphertext	13
AES-128	14
SIKE-Shared secret key	14

Figure 4.9: Percetange of NIST Randomness passed tests by RSA, AES, ASCON, AES, modified SIKE, E91 and BBM92 protocol

4.10 Comparative analysis between SHA-1 used in ECDSA and HMAC based on SHA-3

AGA-12 uses ECDSA based on SHA-1 to sign the message for integrity and authentication. However, SHA-1 has already been cracked by Google[151]. The arrival of quantum computing threatens all classical cryptosystems, including ECDSA. Thus, ECDSA must be replaced or updated. As per NIST [152], [131], SHA3 provides resistance against collision attacks using both classical and quantum settings. Theoretically, it takes $\sqrt[3]{2^{256}}$ quantum operation to obtain SHA3-256 collision [153]. However, practically, due to current quantum resources, SHA-3 is quantum-safe. SHA-3 provides computational eficiency in hardware over a wide range of platforms[152]. It has calculated the mean execution time for SHA-1 and SHA-3. SHA-1 takes 2.53 µ seconds and SHA-3 takes 36.9 µ seconds to derive a digest. Table 4.9 shows the comparison of SHA-1 and SHA-3 based on execution time.

Table 4.9: Comparative Analysis based on execution Time(SHA-1 vs SHA-3)

Hash Algorithms	Mean execution Time(in micro-seconds)
SHA1-256	2.53
SHA3-256	36.9

4.11 Conclusion

The security of critical industrial processes against cyber-attacks based on tradition and quantum computing is crucial as it affects public safety and reliability. This in turn requires an eficient and robust standard is required to strengthen the standard used to secure SCADA communications. It combines the classical cryptosystem with quantum and post-quantum cryptography in the proposed scheme. The post-quantum cryptography (SIKE) is modified to amplify the shared secret key's privacy and authenticate the quantum cryptography (BBM92) protocol for the integrity of the session key. The thesis involves simulation of the proposed scheme and the current algorithms used in SCADA and compared them to validate the hypothesis. Based on the results, the following conclusions are made.

- Modified SIKE shared secret key size is much smaller than RSA key pairs. Moreover, the time to generate one 128-bit modified sike shared secret key is higher than that of generating RSA-1024 key pairs. The randomness of modified SIKE SSK is 87.5%, whereas the average randomness score of RSA key pairs is 84.37 %.

- The randomness of E91 and BBM92 keys are the same (87.5%). E91 uses CHSH value or Bell's Inequality Test, and BBM92 uses QBER to detect intrusion detection. The percentage of information leakage is around 1.21 times higher in E91 than that of BBM92. The final key size of E91 is 1.01 times higher than that of BBM92. The mismatched bits are 4.7 times higher in BBM92 than that in E91. Furthermore, the execution time for E91 is 1.71 times higher than that of BBM92.

- Randomness of ASCON-128 bit cipher is 81.25%, whereas the randomness score of AES-128 is 87.5%. Moreover, the time to execute the ASCON-128 algorithm is 1.88 times higher than AES-128. However, ASCON-128 involved a message authentication tag that AES-128 lacks to provide to the encrypted message.
- The time to execute the SHA3-256 algorithm is 14.58 times higher than that of SHA1-256. However, SHA1 has already been cracked by Google. In addition, SHA-3 generates collision and quantum-resistance digest.

A master terminal unit and remote terminal unit exchange data on a point-to-point communication link in the SCADA network. Thus, a QKD protocol will require two channels, classical(Internet or fiber optic) and quantum(fiber optic). Thus, in case either of the links breaks, the entire communication fails[105], [106]. Thus, as a part of extended work, a quantum channel can be designed to manage quantum and classical communication[11].

Chapter 5

Module 3: LARSCOM: A Lightweight and Robust algorithm for Secure Communication in Supervisory Control and Data Acquisition Networks

5.1 Summary of the Chapter

The research work reported in this chapter has resulted in the following publications:

1. S. Ghosh, M. Zaman, R. Joshi, S. Sampalli, LARSCOM: A lightweight and robust algorithm for secure communication in Supervisory Control and Data Acquisition networks, submitted to IEEE Access.(Impact Factor: 4.43)
2. D. Upadhyay, S. Ghosh, H. Hohno, M. Zaman, S. Sampalli. Design and development of secure SCADA/IoT (Supervisory Control and Data Acquisition/ Internet of Things) based test bench for industrial control systems and performance evaluation of lightweight cipher on hardware emulator, submitted to Computers Security. (Impact Factor: 5.1)

Supervisory Control And Data Acquisition (SCADA) networks are responsible for monitoring and controlling critical infrastructure processes. However, with the increasing reliance of SCADA systems on the internet, they have become susceptible to cyber threats. Therefore, novel security solutions are required to address emerging threats and ensure the protection of critical infrastructures' functionality [2, 154].

SCADA systems typically follow a hierarchical architecture consisting of three main layers. The first layer, known as Level 1, encompasses field instrumentation control, which includes sensors for measuring process variables and actuators for controlling equipment or processes. The data collected by these sensors and actuators is then transmitted to Level 2, which comprises Remote Terminal Units (RTUs) or sub-Master Terminal Units (sub-MTUs). These intermediate units serve as a bridge between the field devices and the supervisory level. They collect data from the field devices, perform local processing based on control commands, and transmit the collected data to the supervisory level. The supervisory level, known as Level 3, consists of the Master Terminal Unit (MTU) and the Human-Machine Interface (HMI) user interface. This level is responsible for data analysis, alarm management, and visualization, serving as the central controller for the SCADA system [44, 155].

Furthermore, the sender generates an authentication tag that is sent along with the cipher to the receiver. The receiver decrypts the cipher, extracts the message, and validates the integrity of the message using the processed authentication tag. Additionally, the authentication tag is encrypted using the sender’s private or secret key and sent back to the sender for authentication tag validation. Finally, the sender decrypts the cipher tag and verifies the integrity of the tag. The main contributions of this chapter can be summarized as follows.

5.2 Our Proposed Scheme

This section presents a lightweight and robust secure protocol designed for the exchange of sensitive data between the Remote Terminal Unit (RTU) and Master Terminal Unit (MTU)/sub-MTU within the SCADA network infrastructure. The framework, illustrated in Figure 5.1, comprises three primary phases, each further divided into sub-phases.

The first phase, Key Generation and Establishment Phase, focuses on generating key pairs and a shared secret key. Additionally, this phase incorporates intrusion detection mechanisms to ensure secure communication. The second phase, Encryption, involves the generation of a cipher from the sensitive message. Finally, the third phase, Authentication, aims to generate and encrypt a tag that validates the integrity of the message, ensuring confidentiality.

The proposed scheme operates under the assumption that the RTU acts as the sender and the sub-MTU serves as the receiver. Both the RTU and sub-MTU store each other’s IDs. While SCADA standards and key management protocols developed by researchers worldwide have aimed to address vulnerabilities inherited from legacy systems and prevent potential exploitation in real-time communication, they fail to fully satisfy all the security requirements of SCADA infrastructure. Consequently, there is a pressing need for a hybrid security scheme that effectively addresses confidentiality, integrity, and availability.

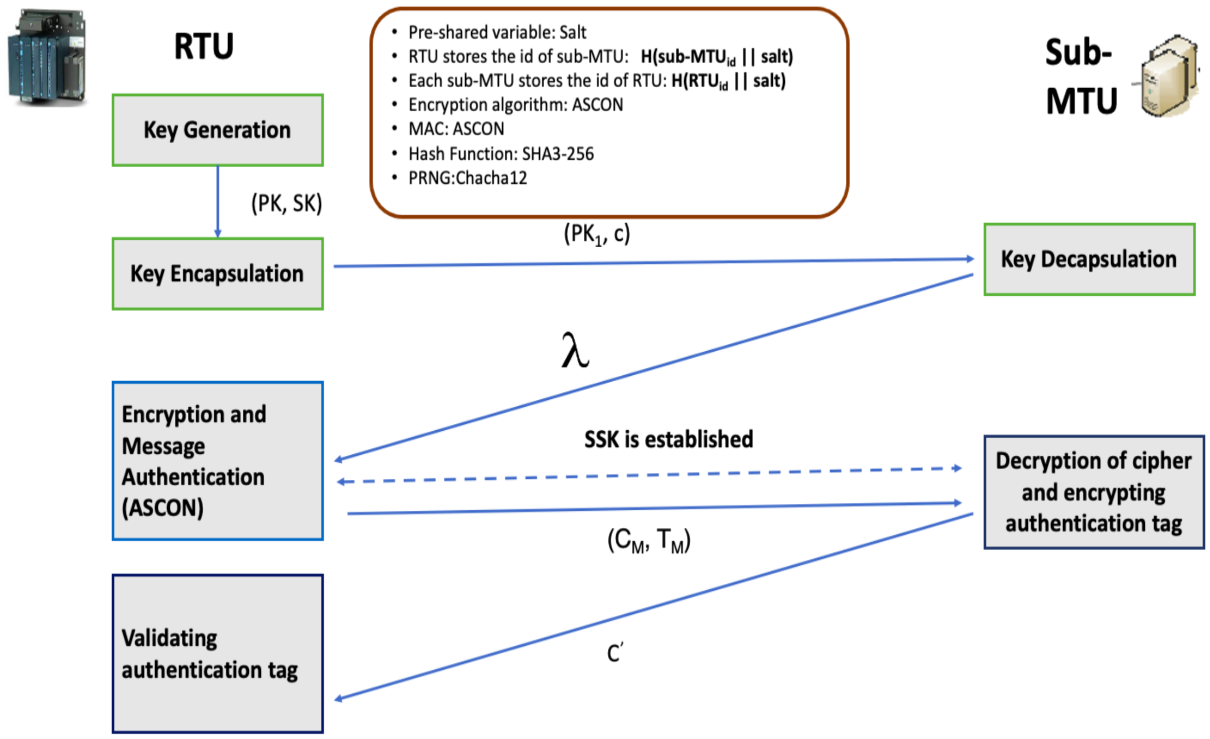

RTU
Sub-MTU
Pre-shared variable: Salt
RTU stores the id of sub-MTU: H(sub-MTU_id || salt)
Each sub-MTU stores the id of RTU: H(RTU_id || salt)
Encryption algorithm: ASCON
MAC: ASCON
Hash Function: SHA3-256
PRNG:Chacha12
Key Generation
(PK, SK)
Key Encapsulation
(PK1, c)
Key Decapsulation
λ
Encryption and Message Authentication (ASCON)
SSK is established
Decryption of cipher and encrypting authentication tag
(CM, TM)
Validating authentication tag
C′

Figure 5.1: Our proposed Security Framework: LARSCOM

5.2.1 Key Generation and Establishment

This section involves three sub-phases, mainly, Key Generation, Key Encapsulation and Key Decapsulation as discussed below.

Key Generation

Algorithm 2 shows the steps followed by RTU generating key pairs, public and secret key, (PK,SK). A raw key is generated initially with n number of bits, and then it is fed to Huffman encoding algorithm, that compresses the key to K_{128}. It generates the private key SK based on hash of RTU_{id}, K_{128}, time variant parameters denoted as TVP. The public key PK is generated based on hashed value of id of sub-MTU concatenated with salt and the private key. It further processes the the public key by feeding it to hash function. The hashed public key (PKH) and the public key is denoted as the final public key PK_1.

Algorithm 2 Key Generation

```
Step 1. $s \leftarrow \{0,1\}^n$ , $n \in Z$
Step 2. $K_{128} \leftarrow HF(F((s\text{⍰}t)))$
Step 3. $TVP \leftarrow \{t_1,t_2\}$
Step 4. $SK \leftarrow \{K_{128}, H(RTU_{id}), TVP\}$
Step 5. $PK \leftarrow \{ H(\text{sub-}MTU_{id} \text{ ⍰ } salt), SK\}$
Step 6. $PKH \leftarrow H(PK)$
Step 7. $PK_1 \leftarrow \{PK, PKH\}$
```

Prerequisites: Huffman Coding Function [47] HF, Hash Function H, Pseudo random number generator F, salt is already pre-shared in RTU and field node, TVP Time Variant Parameters which includes t as the current timestamp, t_1 the timestamp before Huffman Coding Function is performed, t_2 the timestamp after Huffman Coding is performed.

Key Encapsulation

This phase involves applying below Radioactive Decay Law equation (5.1) in Huffman encoding algorithm to calculate the decay or compression rate as shown in equation (5.2)[48].

$$N = N_0 e^{-\lambda t} \tag{5.1}$$

$$\Rightarrow \lambda = \frac{-\ln N \quad N_0}{t/} \tag{5.2}$$

where λ is the compression constant, N is the current key length, N_0 is the raw key length, t is the time elapsed to execute Huffman encoding algorithm, e is the mathematical constant with value 2.7182. The public key hash value of H(PKH) is split into two, mainly denoted as r and k. As shown in Algorithm 3, the compression rate λ is calculated, and is further encrypted with the essence of public key, r, to generate the cipher, c. The value of cipher c concatenated with public key parameter, k, is further hashed to generate the shared secret key, SSK.

Algorithm 3 Key Encapsulation

```
Step 8. (r,k) ← H(PKH)
Step 9. $\lambda \leftarrow \frac{-\ln N/N_0}{t}$
Step 10. t gets the current timestamp
Step 11. c ← Encrypt▯(λ, (PK ▯r))
Step 12. SSK ← H(c▯k)
Step 13. Sends public key and cipher ($PK_1$, c)
```

Algorithm 4 Key Decapsulation

```
Step 14. PKH ← H(PK)
Step 15. SK ▯ PK
Step 16. H(sub-$MTU_{id}$ ▯ salt) ▯ PK
Step 17. {$K_{128}$, H($RTU_{id}$), TVP} ← SK
Step 18. {$t_1$,$t_2$} ← TVP
Step 19. t gets $t_2 - t_1$
Step 20. (r,k) ← H(PKH)
Step 21. λ ← Decrypt▯(c, (PK ▯r))
Step 22. $\lambda_{derived} \leftarrow \frac{-\ln N/N_0}{t}$
if λ is = $\lambda_{derived}$ then
Communication is discarded.
elseIf λ == $\lambda_{derived}$
Derive SSK, And share $\lambda_{derived}$.
Step 23. SSK ← H(c▯k)
```

Key Decapsulation

The sub-MTU receives the encapsulated information (PK_1, c) from RTU. It extracts the the public key from the recieved data, obtains the hash of public key. It further processes the public key, to extract the parameters included, mainly, secret key, TVP. The sub-MTU decrypts the cipher, c with the essence of public key extracted and obtains the λ. It also follows the radioactive decay law equation to generate its own derived compression rate, $\lambda_{derived}$. These two values are compared to validate the integrity of the information exchanged. If $\lambda == \lambda_{derived}$, the sub-MTU generates the shared secret key, otherwise, the communication is discarded as shown in Algorithm 4.

5.2.2 Encryption and Message Authentication Tag generation

The RTU uses the ASCON-128 algorithm for encrypting the sensitive message as shown in Algorithm 5. The shared secret key and the message is fed to the ASCON algorithm to obtain the cipher and authentication tag. The authentication tag is further encrypted with secret key and sent to RTU to validate the message has been received.

Algorithm 5 Encryption and Authentication Tag Generation

```
Step 24. M ← Sensitive message
Step 25. ($C_M$, $T_M$) ← $Encrypt^{SSK}_{ASCON}$ (M) where $C_M$ is the encrypted message and $T_M$ is authentication tag.
Step 26. ($C_M$, $T_M$) is sent to sub-MTU.
```

5.2.3 Decryption and Validation

The sub-MTU recieves the encrypted message and authentication tag from RTU. It decrypts the cipher and validates it by geNerating its own tag, and comparing the generated tag with the received tag. The sub-MTU further encrypts the tag with its own private key, and sends the encrypted authentication tag denoted as c'. The RTU receives the c', derives the private key of sub-MTU as they store each other's id. It further decrypts the encrypted tag and validates it by comparing the decrypted tag and its own RTU tag to validate the integrity of the communication.

5.3 Security Evaluation of LARSCOM

In this section, the thesis provides the security proof of the proposed scheme in terms of randomness analysis of keys along with security analysis based on confidentiality, integrity and availability. The primary focus of LARSCOM remains on confidentiality, integrity, and availability. LARSCOM can be extended to address authorization and accounting by implementing additional layers of security measures tailored to specific applications and end users.

5.3.1 Random Analysis of Keys

Our proposed scheme starts with Key generation phase, where the initial parameters are random number as seed and Time Variant Parameters as TVP. The pre-shared parameters are mainly, 1) salt, 2) each MTU/sub-MTU stores the ids of RTUs in the form: $H(RTU_{id} \square salt)$, 3) each RTU stores the ids of sub-MTU/MTU in the form: $H(sub\text{-}MTU_{id} \square salt)$. Based on the above assumption, the thesis has two following cases.

1. Case 1: Seed and TVP are unique at each session.
2. Case 2: Seed is constant for two or more consecutive session. TVP is unique at each session.

Thus, the public, private and shared secret key are random at each session for both cases.

Proof. In Key Generation Phase, Algorithm 2 generates a raw key of 128 bit, $K_{128} \leftarrow HF(F((s \square t)))$. The variable s and t is both unique as s is the seed generated randomly and t is the current timestamp. Thus, K_{128} is random. Further, TVP is unique as it includes the timestamps before and after huffman coding function executed. Further, the private key is generated such that $SK \leftarrow \{K_{128}, H(RTU_{id}), TVP\}$. And, the public key is generated such that $PK \leftarrow \{ H(sub\text{-}MTU_{id} \square salt), SK \}$. Although, the hashed value of id and salt are constant, the K_{128} and TVP is unique for each communication. Thus, for case 1, the private key is random, further validating that the public key is random when seed and TVP are unique at each session.

For Case 2, the thesis will dive into ALgorithm 2, where the K_{128} is random even if the seed is constant, as it uses current timestamp, t. Thus, public key and private are random as well K_{128} and TVP are unique. Algorithm 3, the key encapsulation includes Step 8 as splitting the hashed valued of hashed public key into two, r and k such that $(r,k) \leftarrow H(PKH)$. Since PK is unique, the PKH is unique. Thus, the chunked variables r and k are unique as well. The compression rate, λ derived from $\lambda \leftarrow \frac{-\ln N/N_0}{t}$ is unique since t is the current timestamp. Further, the cipher generated, $\leftarrow$ Encrypt□(λ, (PK □r)), is random since λ, r and PK are all unique. Finally, the shared secret key is generated from hashing c and k such that $SSK \leftarrow H(c□k)$. Since, c and k are random, the SSK for every communication is random. □

5.3.2 Security Analysis based on CIA triad

This section evaluates the security of the proposed scheme, LARSCOM, based on confidentiality, integrity and availability(CIA) [156]. This thesis identified the following list of attacks, defended by the proposed algorithm addressing all the three security requirements of SCADA infrastructure as shown in Figure 5.2.

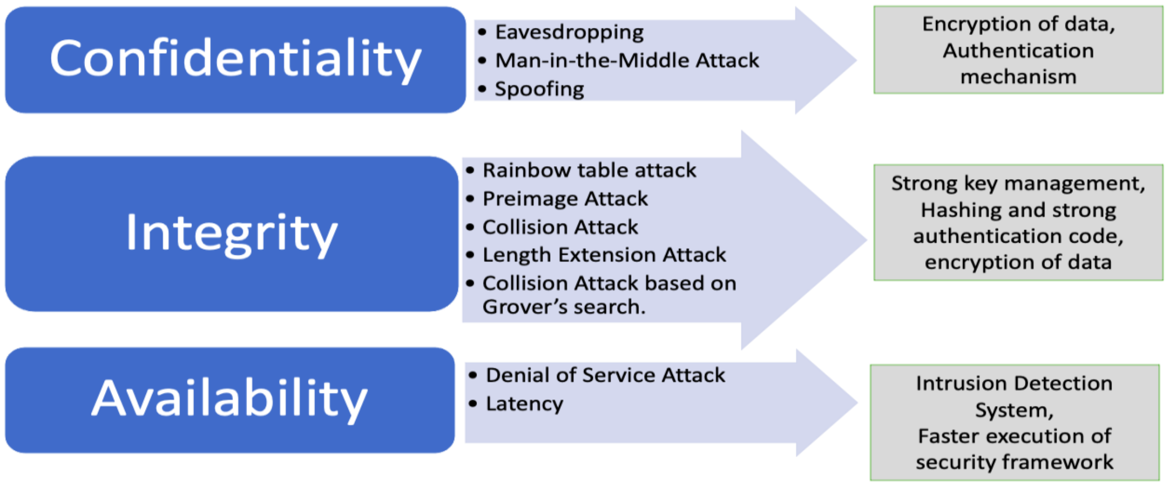

Confidentiality
• Eavesdropping
• Man-in-the-Middle Attack
• Spoofing
Encryption of data, Authentication mechanism
Integrity
• Rainbow table attack
• Preimage Attack
• Collision Attack
• Length Extension Attack
• Collision Attack based on Grover's search.
Strong key management, Hashing and strong authentication code, encryption of data
Availability
• Denial of Service Attack
• Latency
Intrusion Detection System, Faster execution of security framework

Figure 5.2: Security features provided by multiple phases of the proposed scheme

In SCADA networks, confidentiality is violated by eavesdrop, man-in-the-middle , and spoofing attack pretending as the authorised unit to gain sensitive data. Therefore, as a countermeasure, the algorithm is providing encryption of data while being exchanged, and providing strong authentication mechanism [1]. Integrity can e violated in various ways, involving rainbow table attack to crack the hashed values, preimage attack to determine the input of the hashed digest, collision attack by brute forcing to find a pair of different inputs that have the same hash, and length extension attack where the adversary must know the hash value and the length of the hash to crack it. Furthermore, Collision attack can be launched based on Grover's search in a quantum setting. Thus, as a countermeasure, the algorithm is using strong key management scheme, as well as hashing and authentication code along besides encryption of data [1], [2]. The third is availability which can be violated by denial of service attack and latency. To defend against such attacks, the algorithm acts as an intrusion detection system, and is a lightweight, robust security framework, as compared to the previous algorithms and AGA-12 [1],[2].

Case 1: Our proposed algorithm provides integrity as one of the security goals.

Proof. Secure Hash Algorithm (SHA-3) represents Keccak based sponge construction, and Keccak is the winner of SHA-3 NIST competition [157], [158]. It is secure against collision resistant, length-extension attack and second-preimage attacks. SHA-0 and SHA-1 are not collision resistant and are completely broken . Both SHA-2 (256 bit) and SHA-3 (256 bit) has security level of 128 [158], [131]. SHA-3 is resistant against collision and preimage attack using Grover's search on quantum setting [159][160]. Furthermore, in the algorithm, the module is time variant parameters which are unique at every session, and to generate the public private and shared secret key the module is using these unique parameters along with a proven secure hash function, which is SHA3. And, thus the keys are unique every time. For encryption, the module is using ASCON algorithm, based on sponge construction like SHA-3 and is proven to be collision-resistant [161]. Furthermore, the complexity of collision against sponge function is still unknown, and thus it can be defined as resistant against collision using quantum computing [161]. As a countermeasure against Rainbow table [162] attack; it is using salt to append the terminal unit ids before it is further hashed and fed to encryption algorithm. Thus, LARSCOM provides integrity to the SCADA communication. □

Case 2: Our proposed algorithm provides confidentiality as one of the security goals.

Proof. As per Proof 1, SHA3 is collision-resistant and pre-image resistant [157], [158]. Randomness of the keys are proven theoretically, and security proof of encryption algorithm ASCON-128 is well proven [149]. The Shared Secret Key and the cipher generated is unique at every session, and is not shared publicly, but generated by both units. Only the public key and the cipher (C) is exchanged, and unique at each session. Thus, the key exchange and encryption algorithm is proven to be resistant against eavesdropping. The public cipher is the encrypted version of the compression rate denoted as λ, such that, $c \leftarrow \text{Encrypt}_{[illegible]}(\lambda, (PK\ [illegible]r))$. The λ and the public key parameters are encrypted based on ASCON algorithm, using the private key of the sender. λ is used to validate the SSK established, and for detection of data tampering caused by Man-in-the-Middle attack. Further, ASCON generates cipher and authentication tag [149]. It validates the confidentiality and integrity of the compression rate, and, thus the shared secret key(SSK). The unit ids and the salt value is already pre-shared between RTU and sub-MTU, that is, they are secret. Thus, the algorithm generates authenticated keys and cipher, and is resistant against spoofing. □

Case 3: Our proposed algorithm provides availability as one of the security goals.

Proof. Our proposed algorithm generates compression rate to detect data tampering and output alarms during the attack, and discard the communication. Further, based on the comparative analysis between ASCON vs ACORN vs AES, ASCON provides a trade off between power and throughput as shown in Table 5.1, [163]. Thus, ASCON is chosen for the algorithm and it is suitable for resource constrained devices and is resistant against side-channel attacks [149], [163]. Thus, the algorithms provides resistance against denial of service attack as it detects any tampering in sensitive command messages, and, prevents latency. □

5.4 Formal Analysis of the proposed security scheme

The thesis provides a formal analysis, using Scyther tool [164], of the lightweight and robust algorithm for secure communication between sub-MTU/MTU and RTUs.

Table 5.1: Comparative review of ACORN, Ascon, AES based on power and throughput

Algorithm	Power	Throughput
ACORN	8.6 mW	18.7 mW
ASCON	15.9 mW	91.4 Mbps
AES	18.7 mW	76.7 Mbps

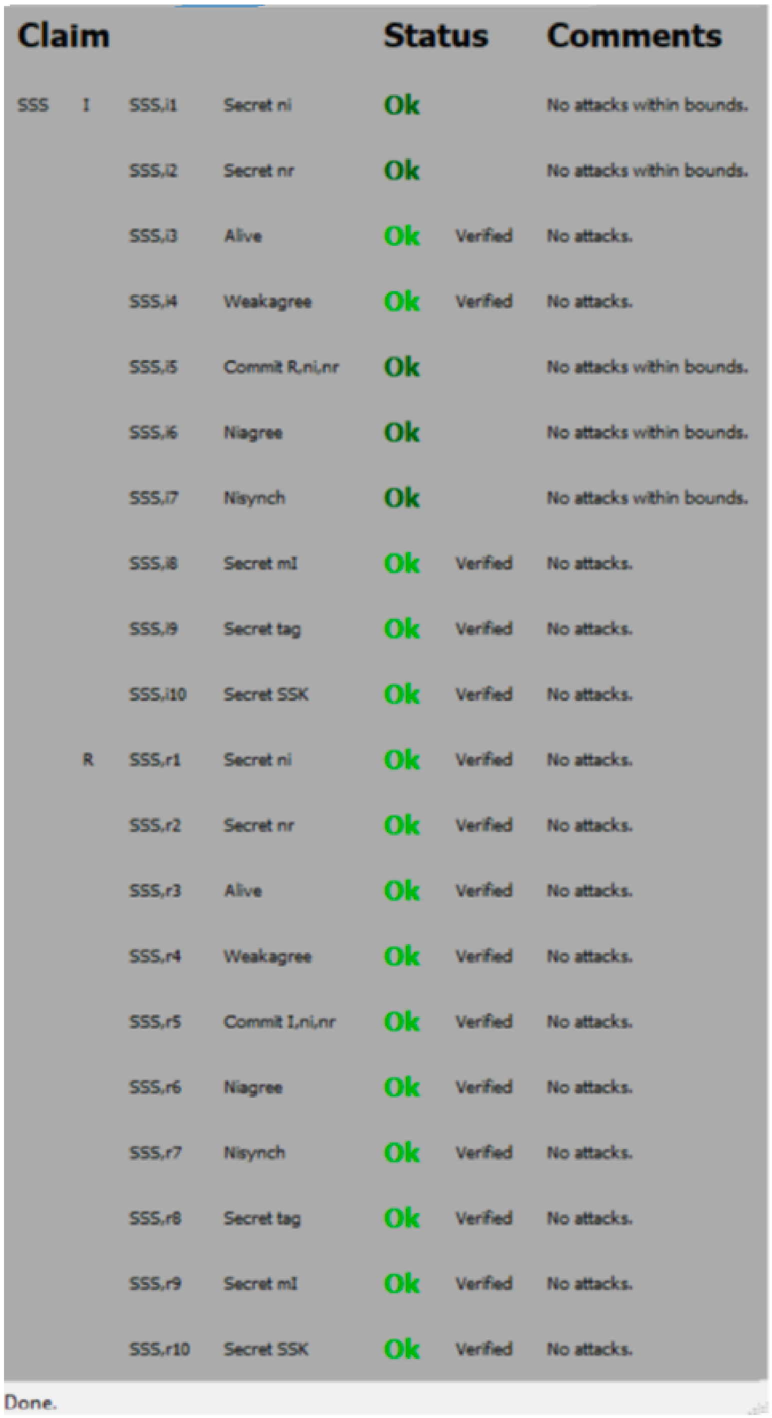
Table 5.2: Comparative Analysis between AGA-12 and LARSCOM based on randomness in keys

NIST Randomness Test (%)	RSA Public Key	RSA Private Key	Proposed Algorithm Public key	Proposed Algorithm Private Key	Proposed Algorithm SSK
N = 10	87.7	81.25	93.75	93.75	87.5
N = 50	81.25	81.25	93.75	93.75	93.75
N = 100	93.75	87.5	93.75	93.75	87.5
N = 500	93.75	87.5	93.75	93.75	93.75

Our proposed security model was built and fed to Scyther to verify the secrecy of the data exchanged. It verifies the sender and the receiver are trusted party and communicating over a secured channel. It also verifies authentication property which includes aliveness of the parties, synchronization and execution of the protocol, and message agreement between two parties. As shown in Figure 5.3, scyther tool [164] verifies the protocol is secured and satisfies the security properties claimed, with no attacks successfully found in the developed model.

5.5 Performance Evaluation

In the before-mention section, the proposed scheme is successfully mathematically and formally analyzed and verified. The performance of the security framework is evaluated. The section contains comparative analysis between AGA-12, the current SCADA standard and LARSCOM, the security model for SCADA networks. This module is implemented in Python 3.5 (Spyder IDE) on macOS 1.8 GHz Dual-Core Intel Core i5 processor.



Claim				Status		Comments
SSS	I	SSS,i1	Secret ni	Ok		No attacks within bounds.
		SSS,i2	Secret nr	Ok		No attacks within bounds.
		SSS,i3	Alive	Ok	Verified	No attacks.
		SSS,i4	Weakagree	Ok	Verified	No attacks.
		SSS,i5	Commit R,ni,nr	Ok		No attacks within bounds.
		SSS,i6	Niagree	Ok		No attacks within bounds.
		SSS,i7	Nisynch	Ok		No attacks within bounds.
		SSS,i8	Secret mI	Ok	Verified	No attacks.
		SSS,i9	Secret tag	Ok	Verified	No attacks.
		SSS,i10	Secret SSK	Ok	Verified	No attacks.
	R	SSS,r1	Secret ni	Ok	Verified	No attacks.
		SSS,r2	Secret nr	Ok	Verified	No attacks.
		SSS,r3	Alive	Ok	Verified	No attacks.
		SSS,r4	Weakagree	Ok	Verified	No attacks.
		SSS,r5	Commit I,ni,nr	Ok	Verified	No attacks.
		SSS,r6	Niagree	Ok	Verified	No attacks.
		SSS,r7	Nisynch	Ok	Verified	No attacks.
		SSS,r8	Secret tag	Ok	Verified	No attacks.
		SSS,r9	Secret mI	Ok	Verified	No attacks.
		SSS,r10	Secret SSK	Ok	Verified	No attacks.

Done.

Figure 5.3: Formal Analysis of LARSCOM by Scyther tool

Table 5.3: Comparative Analysis between AGA-12 and LARSCOM based on Key Generation Time

Algorithms	Key Size (in bits)	Key Generation Time (in seconds) for each sample				Average Key Generation Time (in seconds)
		N = 10	N = 50	N = 100	N = 500	
AGA-12 RSA Public Key	1024	0.0000119	0.000009	0.000011	0.000015	0.000011725
AGA-12 RSA Private Key	1024	0.000009	0.000013	0.000015	0.000009	0.0000115
LARSCOM Public Key	3388	0.00085948	0.00061965	0.00077462	0.00072917	0.000745729
LARSCOM Private Key	2624	0.00058289	0.00037315	0.00024517	0.00030979	0.000377748
LARSCOM Shared Secret Key	512	0.00033975	0.000324	0.00034934	0.00038438	0.000349367

Table 5.4: Comparative Analysis of AGA-12 and LARSCOM based on time to generate Cipher and signature.

Algorithms	Average execution time to generate Cipher and Signature (in seconds)			
	N=10	N = 50	N = 100	N = 500
LARSCOM (ASCON-128)	0.00314	0.00165	0.00174	0.00166
AGA-12(AES-128 and ECDSA-792)	0.00992	0.00997	0.92138	0.00961

Table 5.5: Storage Cost of keys of key management schemes proposed for SCADA networks

Key Management Schemes	MTU	Sub-MTU	RTU
SKE	m(1 + r)	1 + r	1
SKMA	m(1 + r)	1 + r	1
ASKMA	2m + mr	r + log m	2 + log r
ASKMA+	m	1 + r + log m	1 + log r
Hybrid	m + 2	2r + 1	1 + log r
CKMI	2 + r + m	r + 2	1
Proposed Key Management Scheme	m + 1	r + 1	2

5.5.1 Comparative Analysis based on NIST Randomness Test(%) on AGA-12 and LARSCOM

The proposed scheme and AGA-12 are executed n times, where n = 10, 50, 100, 500 times, and fed each key bitstream sample to NIST Randomness Test suite [165] to measure the randomness in the keys of each algorithm. AGA-12 includes RSA-1024 and the proposed algorithm has a novel key generation scheme that generates public key length of 3388 bit private key length of 2624 bit, and shared secret key length of 512 bit. For N =10, Randomness of RSA Public Key is 87.7% and public key generated in the proposed algorithm is 93.75%. And that of RSA private key is 81.25, and that of private key generated in the proposed algorithm is 93.75 and the shared secret key is 87.5%. Similarly for n = 50, 100, 150, the randomness of the proposed algorithm keys are higher than that of RSA keys as shown in Table 5.2.

5.5.2 Comparative Analysis between AGA-12 and LASRCOM based on key size and execution time to generate key pairs, cipher and signature

This section provides simulation results of LARSCOM and AGA-12 for n times, such that, n = 10, 50, 100, 500 and measured the time to generate each key and cipher in AGA-12 and the proposed algorithm, LARSCOM. As shown in Table 5.3 includes the calculated the average of each sample, and observed that time to generate RSA public keys and private keys are lower than that LARSCOM public, private and shared secret keys. However, the key sizes of LARSCOM is significantly higher than that of RSA key pairs. The section also includes the time taken to generate cipher and authentication tag or signaure in AGA-12 and LARSCOM. Based on the simulation results, LARSCOM has lower execution time as compared to that of AGA-12 for cipher and signature generation as shown in Table 5.4.

5.5.3 Storage Cost of Keys

Table 5.5 shows the storage cost of keys of each key management scheme developed for SCADA networks. It is concluded that the observed storage cost of LARSCOM keys are relatively low compared to the other schemes, except for ASKMA+. The MTU of Proposed is only m+1, which is lower than all the other schemes except ASKMA+. Additionally, the Sub-MTU of Proposed is only r+1, which is the lowest among all the schemes. This indicates that the Proposed Key Management Scheme requires less storage space for keys compared to the other schemes, except for ASKMA+. Table 5.5 has two variables m and r is the number of MTUs or sub-MTUs and RTUs, respectively. For (m,r) = (2,2), the visual representation is shown in Figure 5.4.

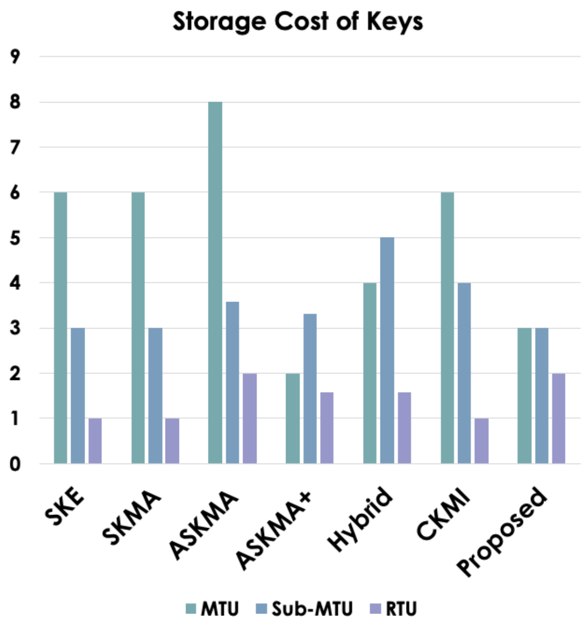


Figure 5.4: Storage Cost of keys for each key management schemes developed for SCADA networks.

5.5.4 Execution Time of proposed scheme and AGA-12 on hardware SCADA testbed

This section provides the experimental results observed of LARSCOM implemented on Raspberry Pi. In Table 5.6, the session key generation and distribution algorithm have the execution time of 0.277996 millisecond. Wheras, both encryption using a symmetric key and tag generation as well as decryption and validation using tag requires same execution time of 0.672602 milliseconds. Thus, both the operations require a similar amount of processing time and has higher execution time than that of session key generation and distribution algorithm. Furthermore, in Table 5.7, The time to generate Public Key is 1.495 milliseconds, while that of Private Key took 2.680 milliseconds. Thus, public key generation took less processing time on raspberry pi as compared to that of private key generation.

Table 5.8 displays the execution time of RSA algorithm and key sizes generated in AGA-12. For the RSA algorithm, both the RSA Public Key and RSA Private Key have a key size of 1024 bytes. The key generation process for both keys took approximately 108.485 milliseconds, with a negligible difference of 0.003 milliseconds between the public and private keys. In Table 5.9, the execution time of encryption

Table 5.6: Execution Time of LARSCOM on Raspberry Pi3

Execution Time of algorithms on Raspberry Pi	
Algorithms (executed on Raspberry Pi)	Execution Time (ms)
Session Key Generation & Distribution	0.277996
Encryption using Symmetric Key and tag	0.672602
Decryption and Signature/Tag Generation	0.672602

Table 5.7: Time taken to generate public and private key in LARSCOM

Execution Time of Algorithms on Raspberry Pi		
Keys Generated	Key Sizes (in bits)	Key Generation (in ms)
LARSCOM Public Key	3388	1.495
LARSCOM Private Key	2624	2.680

Table 5.8: Time taken to generate public and private key in AGA-12

Execution Time of AGA-12 Algorithms and Key Sizes on Raspberry Pi		
Keys Generated	Key Sizes(in bytes)	Key Generation (in ms)
RSA Public Key	1024	108.485
RSA Private Key	1024	108.488

Table 5.9: Execution Time of AGA-12 in Raspberry Pi

Execution Time of Algorithms on Raspberry Pi	
Algorithms (executed on Raspberry Pi)	Execution Time (ms)
AGA-12 Encryption	2.445
AGA-12 Signature Generation	68.419

and signature generation in AGA-12 is presented. The AGA-12 Encryption algorithm took 2.445 milliseconds to execute, while the AGA-12 Signature Generation algorithm took significantly longer at 68.419 milliseconds.

5.5.5 Memory Utilization of proposed scheme and AGA-12 on hardware SCADA testbed

Table 5.10 presents the observed memory utilization of LARSCOM algorithms on a Raspberry Pi, mainly, memory usage across key generation, SSK generation and distribution, encryption, and decryption phase. The memory parameters that have been observed are RSS, VMS, Shared, Text, and Data of each phase. The total memory utilization is the sum of all the memory parameters and that adds up to 118.59 MB for RSS, 207.56 MB for VMS, 51.44 MB for Shared, 15.04 MB for Text, and 69.75 MB for Data. The size of the data are displayed in megabyte(MB).

Table 5.10: Memory Utilization of LARSCOM in Raspberry Pi

Memory utilization of proposed security algorithms on Raspberry Pi					
Memory Parameters	Key Generation	SSK Generation & Distribution	Encryption and Tag generation	Decryption and Tag generation	Total
RSS	29.67 MB	29.64 MB	29.64 MB	29.64 MB	118.59
VMS	51.71 MB	51.97 MB	51.90 MB	51.98 MB	207.56
Shared	12.87 MB	12.85 MB	12.85 MB	12.87 MB	51.44
Text	3.76 MB	3.76 MB	3.76 MB	3.76 MB	15.04
Data	17.20 MB	17.46 MB	17.45 MB	17.64 MB	69.75

Table 5.11: Memory Utilization of AGA-12 in Raspberry Pi

Memory Utilization of AGA-12 on Raspberry Pi				
Memory Parameters	RSA Key Generation	AES Encryption	ECC Signature	Total
RSS	20.2 MB	20.6 MB	20.4 MB	61.2
VMS	34.7 MB	34.9 MB	34.7 MB	104.3
Shared	9.29 MB	9.31 MB	9.29 MB	27.89
Text	3.76 MB	3.76 MB	3.76 MB	11.28
Data	11.5 MB	11.8 MB	11.5 MB	34.8

A similar analysis of memory utilization of AGA-12 algorithms has been performed as shown in Table 5.11. For RSA Key Generation, the memory utilization is 20.2 MB for RSS, 34.7 MB for VMS, 9.29 MB for Shared, 3.76 MB for Text, and 11.5 MB for Data. For AES Encryption, the memory utilization is 20.6 MB for RSS, 34.9 MB for VMS, 9.31 MB for Shared, 3.76 MB for Text, and 11.8 MB for Data. For ECC Signature, the memory utilization is 20.4 MB for RSS, 34.7 MB for VMS, 9.29 MB for Shared, 3.76 MB for Text, and 11.5 MB for Data. The total memory utilization of AGA-12 algorithms is calculated and, is further analyzed for comparison between LARSCOM and AGA-12.

5.5.6 Comparative Analysis between AGA-12 and LARSCOM based on Memory Utilization and Execution Time

This section provides comparative analysis between LARSCOM and AGA-12 based on two factors: mainly, Execution Time, and Memory Utilization as shown in Table 5.12 and Table 5.13 respectively. The observed result in both tables provides an in-depth insight towards the performance characteristics of the algorithms and optimizing their execution on Raspberry Pi devices. Based on Table 5.12, proposed scheme takes 1.6210 milliseconds, and AGA-12 has a much higher total execution time of 287.837 milliseconds.

Table 5.12: Execution Time of AGA-12 vs LARCSOM on Raspberry Pi

Comparative Analysis based on Execution Time of the algorithms on Raspberry Pi	
Security algorithms	Total Execution time (in ms)
LARSCOM	1.6210
AGA-12	287.837

In Table 5.13, based on the observations, LARSCOM, the memory utilization is 118.59 for RSS, 207.56 for VMS, 51.44 for Shared, 15.04 for Text, and 69.75 for Data. And that of AGA-12 is 61.2 for RSS, 104.3 for VMS, 27.89 for Shared, 11.28 for Text, and 34.8 for Data. Thus, LARSCOM shows the higher utilization across all memory parameters, AGA-12 has the lower memory utilization. Based on the comparative analysis results, the thesis conclude that LARSCOM requires more resources, however, still performs reasonably well. AGA-12, on the other hand, has significantly higher memory utilization and execution times.

Table 5.13: Memory Utilization of AGA-12 vs LARCSOM on Raspberry Pi

Memory Utilization of the algorithms on Raspberry Pi					
Security Methods	RSS	VMS	Shared	Text	Data
LARSCOM	118.59	207.56	51.44	15.04	69.75
AGA-12	61.2	104.3	27.89	11.28	34.8

5.6 Concluding Remarks

SCADA systems serve as the backbone of industrial control system infrastructures, playing a critical role in monitoring and controlling essential processes that require robust protection against cyber attacks. To address this need, this thesis proposes a novel lightweight and robust security framework, called LARSCOM, designed for secure communication between terminal units within SCADA networks. LARSCOM effectively meets all the necessary security requirements outlined by SCADA standards, ensuring confidentiality, integrity, and availability of the communication.

The proposed scheme has undergone rigorous mathematical analysis and formal verification to validate its security claims. The randomness of the generated public, secret, and shared secret keys has been mathematically proven, instilling confidence in their cryptographic strength. The security proof of the algorithm has also been established through mathematical means. Notably, the Proposed Key Management Scheme requires less storage space for keys compared to other schemes, with the exception of ASKMA+. Additionally, LARSCOM exhibits a higher level of randomness, as demonstrated by passing the NIST statistical tests.

However, it should be noted that LARSCOM employs longer key sizes (public and private keys) compared to the AGA-12 standard. Consequently, the key generation process in LARSCOM takes significantly more time. On the other hand, the generation of ciphers and signatures in LARSCOM is considerably faster than that of AGA-12. This eficiency improvement contributes to enhanced real-time monitoring, control capabilities, and eficient data processing.

Looking ahead, future work can involve the development of a secure SCADA-based hardware test bench, utilizing two communication protocols, namely Modbus and MQTT (MQ Telemetry Transport). This endeavor would involve deploying Module 3 (the security module) and AGA-12 on a Raspberry Pi 3-based testbed to measure the computational speed and memory utilization of the algorithm in real-time applications. Experimental results have indicated that LARSCOM requires more resources compared to AGA-12, but still performs reasonably well. LARSCOM exhibits lower execution times, enabling its algorithms to complete their operations more quickly than those of AGA-12.

Chapter 6

Conclusion

This thesis presents a novel approach to enhance the security of SCADA networks. Despite the critical role of SCADA networks in ICS, they are vulnerable to current and potential cyber attacks due to the lack of essential security features, particularly in terms of confidentiality, integrity, and availability. Consequently, there is a pressing need to fortify the existing standards and establish robust security measures to ensure secure communication within these networks.

The thesis introduces a collision and preimage resistant framework designed to protect the security of SCADA systems. Additionally, it proposes a Multi phase Quantum Resistant Framework that leverages entanglement and supersingular isogeny based cryptography. This framework enhances the privacy of shared secret keys and provides authentication for quantum cryptography. It offers resistance against various types of attacks, including quantum attacks based on Shor's and Grover's search algorithms, as well as traditional and quantum man-in-the-middle attacks.

Furthermore, the thesis presents the LARSCOM algorithm, a lightweight and robust solution for secure communication within SCADA networks. This algorithm ensures confidentiality and integrity in communication and provides resistance against replay attacks and quantum attacks. Notably, the proposed algorithm outperforms the current SCADA security crypto-based standard, AGA-12.

By deploying the proposed security framework, SCADA networks can establish a robust defense mechanism against potential breaches and attacks, thereby safeguarding critical infrastructure and ensuring the uninterrupted operation of industrial processes.

Bibliography

[1] D. Pliatsios, P. Sarigiannidis, T. Lagkas, and A. G. Sarigiannidis, "A survey on scada systems: secure protocols, incidents, threats and tactics," IEEE Communications Surveys & Tutorials, vol. 22, no. 3, pp. 1942–1976, 2020.

[2] S. Ghosh and S. Sampalli, "A survey of security in scada networks: Current issues and future challenges," IEEE Access, vol. 7, pp. 135 812–135 831, 2019.

[3] Y. Cherdantseva, P. Burnap, A. Blyth, P. Eden, K. Jones, H. Soulsby, and K. Stoddart, "A review of cyber security risk assessment methods for scada systems," Computers & security, vol. 56, pp. 1–27, 2016.

[4] B. Miller and D. Rowe, "A survey scada of and critical infrastructure incidents," in Proceedings of the 1st Annual conference on Research in information technology, 2012, pp. 51–56.

[5] S. Nazir, S. Patel, and D. Patel, "Autonomic computing meets scada security," in 2017 IEEE 16th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC). IEEE, 2017, pp. 498–502.

[6] J. Menn. (2022) U.s. warns newly discovered malware could sabotage energy plants.

[7] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," SIAM review, vol. 41, no. 2, pp. 303–332, 1999.

[8] S. Lomonaco, "Shor's quantum factoring algorithm," in Proceedings of Symposia in Applied Mathematics, vol. 58, 2002, pp. 161–180.

[9] Z. Hu, S. Liu, and K. Chen, "Privacy-preserving location-based services query scheme against quantum attacks," IEEE Transactions on Dependable and Secure Computing, vol. 17, no. 5, pp. 972–983, 2018.

[10] L. K. Grover, "A fast quantum mechanical algorithm for database search," in Proceedings of the twenty-eighth annual ACM symposium on Theory of computing, 1996, pp. 212–219.

[11] S. Ghosh, M. Zaman, G. Sakauye, and S. Sampalli, "An intrusion resistant scada framework based on quantum and post-quantum scheme," Applied Sciences, vol. 11, no. 5, p. 2082, 2021.

[12] C. Gidney and M. Ekerå, "How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits," arXiv preprint arXiv:1905.09749, 2019.

[13] M. Amy, O. Di Matteo, V. Gheorghiu, M. Mosca, A. Parent, and J. Schanck, “Estimating the cost of generic quantum pre-image attacks on sha-2 and sha-3,” in International Conference on Selected Areas in Cryptography. Springer, 2016, pp. 317–337.

[14] V. Mavroeidis, K. Vishi, M. D. Zych, and A. Jøsang, “The impact of quantum computing on present cryptography,” arXiv preprint arXiv:1804.00200, 2018.

[15] D. Upadhyay, M. Zaman, R. Joshi, and S. Sampalli, “An eficient key management and multi-layered security framework for scada systems,” IEEE Transactions on Network and Service Management, 2021.

[16] F. M. Salem, E. Ibrahim, and O. Elghandour, “A lightweight authenticated key establishment scheme for secure smart grid communications,” Journal homepage: http://iieta. org/journals/ijsse, vol. 10, no. 4, pp. 549–558, 2020.

[17] S. Chandra, S. Paira, S. S. Alam, and G. Sanyal, “A comparative survey of symmetric and asymmetric key cryptography,” in 2014 international conference on electronics, communication and computational engineering (ICECCE). IEEE, 2014, pp. 83–93.

[18] X. Zhang, Z. Y. Dong, Z. Wang, C. Xiao, and F. Luo, “Quantum cryptography based cyber-physical security technology for smart grids,” 2015.

[19] D. Choi, H. Kim, D. Won, and S. Kim, “Advanced key-management architecture for secure scada communications,” IEEE Transactions on Power Delivery, vol. 24, no. 3, pp. 1154–1163, 2009.

[20] H. Abbas and S. Shaheen, “Future scada challenges and the promising solution: the agent-based scada,” vol. 10, 01 2014, pp. 307 – 333.

[21] E. Irmak and Erkek, “An overview of cyber-attack vectors on scada systems,” in 2018 6th International Symposium on Digital Forensic and Security (ISDFS), March 2018, pp. 1–5.

[22] A. Rezai, P. Keshavarzi, and Z. Moravej, “Key management issue in scada networks: A review,” Engineering science and technology, an international journal, vol. 20, no. 1, pp. 354–363, 2017.

[23] H. Saputra and Z. Zhao, “Long term key management architecture for scada systems,” in 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Feb 2018, pp. 314–319.

[24] X. Zhang, Z. Y. Dong, Z. Wang, C. Xiao, and F. Luo, “Quantum cryptography based cyber-physical security technology for smart grids,” 2015.

[25] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, J. Kelsey, Y.-K. Liu, C. Miller, D. Moody, R. Peralta et al., "Status report on the second round of the nist post-quantum cryptography standardization process," US Department of Commerce, NIST, 2020.

[26] H. Seo, M. Anastasova, A. Jalali, and R. Azarderakhsh, "Supersingular isogeny key encapsulation (sike) round 2 on arm cortex-m4," IEEE Transactions on Computers, vol. 70, no. 10, pp. 1705–1718, 2020.

[27] A. Chailloux, M. Naya-Plasencia, and A. Schrottenloher, "An eficient quantum collision search algorithm and implications on symmetric cryptography," in International Conference on the Theory and Application of Cryptology and Information Security. Springer, 2017, pp. 211–240.

[28] Z. Wilcox, "Lessons from the history of attacks on secure hash functions," May 2019. [Online]. Available: https://electriccoin.co/blog/lessons-from-the-history-of-attacks-on-secure-hash-functions/

[29] V. Padamvathi, B. V. Vardhan, and A. Krishna, "Quantum cryptography and quantum key distribution protocols: A survey," in 2016 IEEE 6th International Conference on Advanced Computing (IACC). IEEE, 2016, pp. 556–562.

[30] A. I. Nurhadi and N. R. Syambas, "Quantum key distribution (qkd) protocols: a survey," in 2018 4th International Conference on Wireless and Telematics (ICWT). IEEE, 2018, pp. 1–5.

[31] L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, Report on post-quantum cryptography. US Department of Commerce, National Institute of Standards and Technology, 2016, vol. 12.

[32] D. J. Bernstein, D. Hopwood, A. Hülsing, T. Lange, R. Niederhagen, L. Papachristodoulou, M. Schneider, P. Schwabe, and Z. Wilcox-O'Hearn, "Sphincs: practical stateless hash-based signatures," in Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2015, pp. 368–397.

[33] A. Sharma and S.K.Mittal, "Attacks on cryptographic hash functions and advances," vol. 5, pp. 89–96, 11 2018.

[34] D. J. Bernstein, "Chacha, a variant of salsa20," in Workshop Record of SASC, vol. 8, 2008, pp. 3–5.

[35] M. Goll and S. Gueron, "Vectorization on chacha stream cipher," in 2014 11th International Conference on Information Technology: New Generations. IEEE, 2014, pp. 612–615.

[36] A. R. Choudhuri and S. Maitra, "Differential cryptanalysis of salsa and chacha-an evaluation with a hybrid model." IACR Cryptol. ePrint Arch., vol. 2016, p. 377, 2016.

[37] V. Bergholm, J. D. Biamonte, and J. D. Whitfield, "Quantum information toolkit," Apr 2014. [Online]. Available: http://qit.sourceforge.net/docs/html/#

[38] P. Sibson, C. Erven, M. Godfrey, S. Miki, T. Yamashita, M. Fujiwara, M. Sasaki, H. Terai, M. G. Tanner, C. M. Natarajan et al., "Chip-based quantum key distribution," Nature communications, vol. 8, no. 1, pp. 1–6, 2017.

[39] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," IEEE Security & Privacy, vol. 9, no. 3, pp. 49–51, 2011.

[40] R. M. Lee. Irongate malware - thoughts and lessons learned for ics/scada defenders. [Online]. Available: https://www.sans.org/blog/irongate-malware-thoughts-and-lessons-learned-for-ics-scada-defenders/

[41] "Ieee standard for electric power systems communications-distributed network protocol (dnp3)," IEEE Std 1815-2012 (Revision of IEEE Std 1815-2010), pp. 1–821, 2012.

[42] I. N. Fovino, A. Carcano, M. Masera, and A. Trombetta, "Design and implementation of a secure modbus protocol," in Critical Infrastructure Protection III: Third Annual IFIP WG 11.10 International Conference on Critical Infrastructure Protection, Hanover, New Hampshire, USA, March 23-25, 2009, Revised Selected Papers 3. Springer, 2009, pp. 83–96.

[43] C. Alcaraz and S. Zeadally, "Critical infrastructure protection: Requirements and challenges for the 21st century," International journal of critical infrastructure protection, vol. 8, pp. 53–66, 2015.

[44] G. Yadav and K. Paul, "Architecture and security of scada systems: A review," International Journal of Critical Infrastructure Protection, vol. 34, p. 100433, 2021.

[45] M. Katagi, S. Moriai et al., "Lightweight cryptography for the internet of things," sony corporation, vol. 2008, pp. 7–10, 2008.

[46] E. Prouff and M. Rivain, "Masking against side-channel attacks: A formal security proof," in Advances in Cryptology–EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings 32. Springer, 2013, pp. 142–159.

[47] A. Moffat, "Huffman coding," ACM Computing Surveys (CSUR), vol. 52, no. 4, pp. 1–35, 2019.

[48] T. L. Brown, H. E. LeMay Jr, B. E. Bursten, and C. J. Murphy, “Chemistry: The central science,” Journal of Chemical Education, vol. 74, no. 4, p. 378, 1997.

[49] R. Lakshmanan. Nist standardizes ascon cryptographic algorithm for iot and other lightweight devices. [Online]. Available: https://thehackernews.com/2023/02/nist-standardizes-ascon-cryptographic.html

[50] B. Parvez, J. Ali, U. Ahmed, and M. Farhan, “Framework for implementation of aga 12 for secured scada operation in oil and gas industry,” in 2015 2nd international conference on computing for sustainable global development (indiacom). IEEE, 2015, pp. 1281–1284.

[51] R. E. Carlson, J. E. Dagle, and S. A. Shamsuddin, “Summary of control system security standards activities in the energy sector,” in United States. Ofice of Electricity Delivery & Energy Reliability, no. National SADA Test Bed. United States. Ofice of Electricity Delivery & Energy Reliability, 2005.

[52] S. Pirandola, U. L. Andersen, L. Banchi, M. Berta, D. Bunandar, R. Colbeck, D. Englund, T. Gehring, C. Lupo, C. Ottaviani et al., “Advances in quantum cryptography,” Advances in optics and photonics, vol. 12, no. 4, pp. 1012–1236, 2020.

[53] A. I. Nurhadi and N. R. Syambas, “Quantum key distribution (qkd) protocols: A survey,” in 2018 4th International Conference on Wireless and Telematics (ICWT). IEEE, 2018, pp. 1–5.

[54] G. Kato and K. Tamaki, “Security of six-state quantum key distribution protocol with threshold detectors,” Scientific Reports, vol. 6, no. 1, pp. 1–5, 2016.

[55] V. Scarani, A. Acin, G. Ribordy, and N. Gisin, “Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulse implementations,” Physical review letters, vol. 92, no. 5, p. 057901, 2004.

[56] N. Gisin, G. Ribordy, H. Zbinden, D. Stucki, N. Brunner, and V. Scarani, “Towards practical and fast quantum cryptography,” arXiv preprint quant-ph/0411022, 2004.

[57] K. Inoue and T. Honjo, “Robustness of differential-phase-shift quantum key distribution against photon-number-splitting attack,” Physical Review A, vol. 71, no. 4, p. 042305, 2005.

[58] C. H. Bennett and G. Brassard, “Quantum cryptography: Public key distribution and coin tossing,” in Proceedings of the International Conference on Computers, Systems and Signal Processing, 1984.

[59] O. Korchenko, Y. Vasiliu, and S. Gnatyuk, “Modern quantum technologies of information security against cyber-terrorist attacks,” Aviation, vol. 14, no. 2, pp. 58–69, 2010.

[60] C. H. Bennett, “Quantum cryptography using any two nonorthogonal states,” Physical review letters, vol. 68, no. 21, p. 3121, 1992.

[61] K. Tamaki, M. Koashi, and N. Imoto, “Unconditionally secure key distribution based on two nonorthogonal states,” Physical review letters, vol. 90, no. 16, p. 167904, 2003.

[62] M. Elboukhari, M. Azizi, and A. Azizi, “Quantum key distribution protocols: A survey.” International Journal of Universal Computer Science, vol. 1, no. 2, 2010.

[63] H. Bechmann-Pasquinucci and N. Gisin, “Incoherent and coherent eavesdropping in the six-state protocol of quantum cryptography,” Physical Review A, vol. 59, no. 6, p. 4238, 1999.

[64] C.-H. F. Fung, K. Tamaki, and H.-K. Lo, “On the performance of two protocols: Sarg04 and bb84,” arXiv preprint quant-ph/0510025, 2005.

[65] A. Ekert, ““quantum cryptography based on bell’s theorem”, phys. rev. lett.” 1991.

[66] L. C. Alvarez and P. C. Caiconte, “Comparison and analysis of bb84 and e91 quantum cryptography protocols security strengths,” International Journal of Modern Communication Technologies and Research, vol. 4, no. 9, p. 265683, 2016.

[67] C. H. Bennett, G. Brassard, and N. D. Mermin, “Quantum cryptography without bell’s theorem,” Physical review letters, vol. 68, no. 5, p. 557, 1992.

[68] C. Erven, “On free space quantum key distribution and its implementation with a polarization-entangled parametric down conversion source,” Master’s thesis, University of Waterloo, 2007.

[69] D. Stucki, C. Barreiro, S. Fasel, J.-D. Gautier, O. Gay, N. Gisin, R. Thew, Y. Thoma, P. Trinkler, F. Vannel et al., “Continuous high speed coherent one-way quantum key distribution,” Optics express, vol. 17, no. 16, pp. 13 326–13 334, 2009.

[70] K. Inoue, “Differential phase-shift quantum key distribution systems,” IEEE Journal of Selected Topics in Quantum Electronics, vol. 21, no. 3, pp. 109–115, 2014.

[71] Q. Zhang, H. Takesue, T. Honjo, K. Wen, T. Hirohata, M. Suyama, Y. Takiguchi, H. Kamada, Y. Tokura, O. Tadanaga et al., “Megabits secure key rate quantum key distribution,” New Journal of Physics, vol. 11, no. 4, p. 045010, 2009.

[72] D. J. Bernstein and T. Lange, “Post-quantum cryptography,” Nature, vol. 549, no. 7671, pp. 188–194, 2017.

[73] T. M. Fernández-Caramés, “From pre-quantum to post-quantum iot security: A survey on quantum-resistant cryptosystems for the internet of things,” IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6457–6480, 2019.

[74] A. Valentijn, “Goppa codes and their use in the mceliece cryptosystems,” 2015.

[75] K. S. Roy and H. K. Kalita, “A survey on post-quantum cryptography for constrained devices,” International Journal of Applied Engineering Research, vol. 14, no. 11, pp. 2608–2615, 2019.

[76] J. Hoffstein, J. Pipher, and J. H. Silverman, “Ntru: A ring-based public key cryptosystem,” in International algorithmic number theory symposium. Springer, 1998, pp. 267–288.

[77] V. Lyubashevsky, C. Peikert, and O. Regev, “On ideal lattices and learning with errors over rings,” in Annual international conference on the theory and applications of cryptographic techniques. Springer, 2010, pp. 1–23.

[78] L. De Feo, D. Jao, and J. Plût, “Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies,” Journal of Mathematical Cryptology, vol. 8, no. 3, pp. 209–247, 2014.

[79] P. Longa, “A note on post-quantum authenticated key exchange from supersingular isogenies.” IACR Cryptol. ePrint Arch., vol. 2018, p. 267, 2018.

[80] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, J. Kelsey, Y.-K. Liu, C. Miller, D. Moody, R. Peralta et al., “Status report on the second round of the nist post-quantum cryptography standardization process,” US Department of Commerce, NIST, 2020.

[81] D. J. Bernstein and T. Lange, “Post-quantum cryptography,” Nature, vol. 549, no. 7671, pp. 188–194, 2017.

[82] K. S. Roy and H. K. Kalita, “A survey on post-quantum cryptography for constrained devices,” International Journal of Applied Engineering Research, vol. 14, no. 11, pp. 2608–2615, 2019.

[83] T. M. Fernández-Caramés, “From pre-quantum to post-quantum iot security: A survey on quantum-resistant cryptosystems for the internet of things,” IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6457–6480, 2019.

[84] J. Blömer and S. Naewe, “Sampling methods for shortest vectors, closest vectors and successive minima,” Theoretical Computer Science, vol. 410, no. 18, pp. 1648–1665, 2009.

[85] L. De Feo, D. Jao, and J. Plût, “Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies,” Journal of Mathematical Cryptology, vol. 8, no. 3, pp. 209–247, 2014.

[86] H. Seo, M. Anastasova, A. Jalali, and R. Azarderakhsh, “Supersingular isogeny key encapsulation (sike) round 2 on arm cortex-m4,” IEEE Transactions on Computers, 2020.

[87] C. Costello, “Supersingular isogeny key exchange for beginners,” in International Conference on Selected Areas in Cryptography. Springer, 2019, pp. 21–50.

[88] J. Hoffstein, J. Pipher, and J. H. Silverman, “Nss: An ntru lattice-based signature scheme,” in International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2001, pp. 211–228.

[89] P. Pessl, L. G. Bruinderink, and Y. Yarom, “To bliss-b or not to be: Attacking strongswan’s implementation of post-quantum signatures,” in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 2017, pp. 1843–1855.

[90] R. Staffas, “Post-quantum lattice-based cryptography,” 2016.

[91] J. Ding and B.-Y. Yang, “Multivariate public key cryptography,” in Post-quantum cryptography. Springer, 2009, pp. 193–241.

[92] J. Ding and D. Schmidt, “Rainbow, a new multivariable polynomial signature scheme,” in International Conference on Applied Cryptography and Network Security. Springer, 2005, pp. 164–175.

[93] R. A. Perlner and D. A. Cooper, “Quantum resistant public key cryptography: a survey,” in Proceedings of the 8th Symposium on Identity and Trust on the Internet, 2009, pp. 85–93.

[94] S. Suhail, R. Hussain, A. Khan, and C. S. Hong, “On the role of hash-based signatures in quantum-safe internet of things: Current solutions and future directions,” IEEE Internet of Things Journal, 2020.

[95] L. Lamport, “Constructing digital signatures from a one-way function,” 1979.

[96] G. Becker, “Merkle signature schemes, merkle trees and their cryptanalysis,” Ruhr-University Bochum, Tech. Rep, 2008.

[97] L. Reyzin and N. Reyzin, "Better than biba: Short one-time signatures with fast signing and verifying," in Australasian Conference on Information Security and Privacy. Springer, 2002, pp. 144–153.

[98] J.-P. Aumasson and G. Endignoux, "Improving stateless hash-based signatures," in Cryptographers' Track at the RSA Conference. Springer, 2018, pp. 219–242.

[99] B. R. H. Nuhamara and N. R. Syambas, "An evaluation of quantum key distribution in quvis simulation software," in 2018 4th International Conference on Wireless and Telematics (ICWT). IEEE, 2018, pp. 1–4.

[100] S. Ruj and B. Roy, "Key predistribution schemes using codes in wireless sensor networks," in International Conference on Information Security and Cryptology. Springer, 2008, pp. 275–288.

[101] S. P. Choudhari and M. B. Chakole, "Reed solomon code for wimax network," in 2017 International Conference on Communication and Signal Processing (ICCSP). IEEE, 2017, pp. 0176–0179.

[102] M. Riley and I. Richardson, "An introduction to reed-solomon codes: principles, architecture and implementation," 2003.

[103] D. Upadhyay, J. Maneroy, M. Zaman, and S. Sampalli, "Gradient boosting feature selection with machine learning classifiers for intrusion detection on power grids," IEEE Transactions on Network and Service Management, pp. 1–1, 2020.

[104] Stouffer, Keith and Falco, Joseph and Kent, Karen, "Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control Systems Security," 01 2006.

[105] "Scalable Quantum Cryptography Network For Protected Automation Communication." Energy.gov.2017., US Department, 2017. [Online]. Available: https://www.energy.gov/sites/prod/files/2017/05/f34/Qubitekk QKD_FactSheet.pdf

[106] Bailey, David and Wright, Edwin, Practical SCADA for industry. Elsevier, 2003.

[107] J. P. Aumasson and G. Endignoux, "Clarifying the subset-resilience problem," IACR Cryptology ePrint Archive, vol. 2017, p. 909, 2017.

[108] Bernstein, Daniel J and Hülsing, Andreas and Kölbl, Stefan and Niederhagen, Ruben and Rijneveld, Joost and Schwabe, Peter, "The SPHINCS+ signature framework," in Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 2129–2146.

[109] Herrero-Collantes, Miguel and Garcia-Escartin, Juan Carlos, “Quantum random number generators,” Reviews of Modern Physics, vol. 89, no. 1, p. 015004, 2017.

[110] Stipcevic, Mario, “Quantum random number generators and their applications in cryptography,” in Advanced Photon Counting Techniques VI, vol. 8375. International Society for Optics and Photonics, 2012, p. 837504.

[111] “Quantum Versus Classical Random Number Generator,” Energy.gov.2017, ID Quantique, 2017. [Online]. Available: https://marketing.idquantique.com/acton/attachment/11868/f-64900ef6-6e7e-4b4c-a9f9-c912a2cfde59/1/-/-/-/

[112] J.-P. Aumasson and G. Endignoux, “Improving stateless hash-based signatures,” in Cryptographers’ Track at the RSA Conference. Springer, 2018, pp. 219–242.

[113] M. Kwiatkowska, G. Norman, and D. Parker, “Prism: Probabilistic symbolic model checker,” in International Conference on Modelling Techniques and Tools for Computer Performance Evaluation. Springer, 2002, pp. 200–204.

[114] C. Cremers, “Scyther user manual,” Department of Computer Science, University of Oxford: Oxford, UK, 2014.

[115] N. K. Papanikolaou, “Techniques for design and validation of quantum protocols,” Ph.D. dissertation, University of Warwick. Department of Computer Science, 2004.

[116] S. Kuppam, “Modelling and analysis of quantum key distribution protocols, bb84 and b92, in communicating quantum processes (cqp) language and analysing in prism,” arXiv preprint arXiv:1612.03706, 2018.

[117] A. Kuppam, “Modelling bb84, b92 in cqp and analysing in prism,” arXiv preprint arXiv:1612.03706v1, 2016.

[118] R. Chatterjee, K. Joarder, S. Chatterjee, B. C. Sanders, and U. Sinha, “qkdsim: An experimenter’s simulation toolkit for qkd with imperfections, and its performance analysis with a demonstration of the b92 protocol using heralded photon,” arXiv preprint arXiv:1912.10061, 2019.

[119] V. Bergholm, J. D. Biamonte, and J. D. Whitfield, “Quantum information toolkit,” Apr 2014. [Online]. Available: http://qit.sourceforge.net/docs/html/#

[120] C. Crépeau, “Eficient cryptographic protocols based on noisy channels,” in International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 1997, pp. 306–317.

[121] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," Booz-allen and hamilton inc mclean va, Tech. Rep., 2001.

[122] A. Doganaksoy, B. Ege, O. Koçak, and F. Sulak, "Statistical analysis of reduced round compression functions of sha-3 second round candidates." IACR Cryptology ePrint Archive, vol. 2010, p. 611, 2010.

[123] O. Datcu, C. Macovei, and R. Hobincu, "Chaos based cryptographic pseudo-random number generator template with dynamic state change," Applied Sciences, vol. 10, no. 2, p. 451, 2020.

[124] "RSA Cryptography Algorithm, url=http://algohub.me/algo/rsa-cryptography-algorithm.htm accessed: 2020-08-30.

[125] "How fast does a pseudorandom number generator have to be in order to be competitive?" Sep 2018. [Online]. Available: https://crypto.stackexchange.com/questions/62736/how-fast-does-a-pseudorandom-number-generator-have-to-be-in-order-to-be-competit

[126] A. Aung, B. P. Ng, and S. Rahardja, "Sequency-ordered complex hadamard transform: Properties, computational complexity and applications," IEEE Transactions on Signal processing, vol. 56, no. 8, pp. 3562–3571, 2008.

[127] M. A. Philip et al., "A survey on lightweight ciphers for iot devices," in 2017 International Conference on Technological Advancements in Power and Energy (TAP Energy). IEEE, 2017, pp. 1–4.

[128] R. C. B. Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," in 2014 7th International symposium on resilient control systems (ISRCS). IEEE, 2014, pp. 1–8.

[129] R. Azarderakhsh, M. Campagna, C. Costello, L. Feo, B. Hess, A. Jalali, D. Jao, B. Koziel, B. LaMacchia, P. Longa et al., "Supersingular isogeny key encapsulation," Submission to the NIST Post-Quantum Standardization project, vol. 152, pp. 154–155, 2017.

[130] B. Koziel, A.-B. Ackie, R. El Khatib, R. Azarderakhsh, and M. M. Kermani, "Sike'd up: Fast hardware architectures for supersingular isogeny key encapsulation," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 67, no. 12, pp. 4842–4854, 2020.

[131] M. J. Dworkin et al., "Sha-3 standard: Permutation-based hash and extendable-output functions," 2015.

[132] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1. 2," Submission to the CAESAR Competition, 2016.

[133] B. Rezvani, F. Coleman, S. Sachin, and W. Diehl, "Hardware implementations of nist lightweight cryptographic candidates: A first look," Cryptology ePrint Archive, 2019.

[134] P. Busch, T. Heinonen, and P. Lahti, "Heisenberg's uncertainty principle," Physics Reports, vol. 452, no. 6, pp. 155–176, 2007.

[135] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," Nature, vol. 299, no. 5886, pp. 802–803, 1982.

[136] A. Aspect, "Bell's inequality test: more ideal than ever," Nature, vol. 398, no. 6724, pp. 189–190, 1999.

[137] K. Cui, J. Wang, H.-F. Zhang, C.-L. Luo, G. Jin, and T.-Y. Chen, "A real-time design based on fpga for expeditious error reconciliation in qkd system," IEEE transactions on information forensics and security, vol. 8, no. 1, pp. 184–190, 2012.

[138] E. Diamanti, H.-K. Lo, B. Qi, and Z. Yuan, "Practical challenges in quantum key distribution," npj Quantum Information, vol. 2, no. 1, pp. 1–12, 2016.

[139] I. Quantique, "Critical infrastructure," Jan 2015. [Online]. Available: https://www.idquantique.com/quantum-safe-security/applications/critical-infrastructure/

[140] ——, "Clavis xg qkd system," Jan 2015. [Online]. Available: https://www.idquantique.com/quantum-safe-security/products/clavis-xg-qkd-system/

[141] ——, "Clavis300 quantum cryptography platform," Jan 2015. [Online]. Available: https://www.idquantique.com/quantum-safe-security/products/clavis300-quantum-cryptography-platform/

[142] C. Ghosh, A. Parag, and S. Datta, "Different vulnerabilities and challenges of quantum key distribution protocol: A review." International Journal of Advanced Research in Computer Science, vol. 8, no. 8, 2017.

[143] A.-B. Al-Ghamdi, A. Al-Sulami, and A. O. Aljahdali, "On the security and confidentiality of quantum key distribution," Security and Privacy, vol. 3, no. 5, p. e111, 2020.

[144] Y.-Y. Fei, X.-D. Meng, M. Gao, H. Wang, and Z. Ma, "Quantum man-in-the-middle attack on the calibration process of quantum key distribution," Scientific reports, vol. 8, no. 1, pp. 1–10, 2018.

[145] Y. Li, P. Huang, S. Wang, T. Wang, D. Li, and G. Zeng, "A denial-of-service attack on fiber-based continuous-variable quantum key distribution," Physics Letters A, vol. 382, no. 45, pp. 3253–3261, 2018.

[146] N. Gisin, G. Ribordy, W. Tittel, and H. Zbinden, "Quantum cryptography," Reviews of modern physics, vol. 74, no. 1, p. 145, 2002.

[147] A. Cross, "The ibm q experience and qiskit open-source quantum computing software," in APS March meeting abstracts, vol. 2018, 2018, pp. L58–003.

[148] M. R. Islam, "Sample size and its role in central limit theorem (clt)," Computational and Applied Mathematics Journal, vol. 4, no. 1, pp. 1–7, 2018.

[149] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1. 2: Lightweight authenticated encryption and hashing," Journal of Cryptology, vol. 34, no. 3, pp. 1–42, 2021.

[150] M. S. Turan. (2021, Oct) Csrc presentation: Final round of the nist lwc standardization. [Online]. Available: https://csrc.nist.gov/presentations/2021/final-round-of-the-nist-lwc-standardization

[151] R. Brandom, "Google just cracked one of the building blocks of web encryption (but don't worry)," The Verge, vol. 23, no. 02, 2017.

[152] D. A. Osvik, "Fast embedded software hashing." IACR Cryptol. ePrint Arch., vol. 2012, p. 156, 2012.

[153] D. J. Bernstein, "Cost analysis of hash collisions: Will quantum computers make sharcs obsolete," SHARCS, vol. 9, p. 105, 2009.

[154] J. Suaboot, A. Fahad, Z. Tari, J. Grundy, A. N. Mahmood, A. Almalawi, A. Y. Zomaya, and K. Drira, "A taxonomy of supervised learning for idss in scada environments," ACM Computing Surveys (CSUR), vol. 53, no. 2, pp. 1–37, 2020.

[155] S. Shitharth, K. M. Prasad, K. Sangeetha, P. R. Kshirsagar, T. S. Babu, and H. H. Alhelou, "An enriched rpco-bcnn mechanisms for attack detection and classification in scada systems," IEEE Access, vol. 9, pp. 156 297–156 312, 2021.

[156] S. Samonas and D. Coss, "The cia strikes back: Redefining confidentiality, integrity and availability in security." Journal of Information System Security, vol. 10, no. 3, 2014.

[157] W. Penard and T. van Werkhoven, "On the secure hash algorithm family," Cryptography in context, pp. 1–18, 2008.

[158] S. Nakov.

[159] M. Amy, O. Di Matteo, V. Gheorghiu, M. Mosca, A. Parent, and J. Schanck, "Estimating the cost of generic quantum pre-image attacks on sha-2 and sha-3," in Selected Areas in Cryptography–SAC 2016: 23rd International Conference, St. John's, NL, Canada, August 10-12, 2016, Revised Selected Papers. Springer, 2017, pp. 317–337.

[160] G. Brassard, P. Høyer, and A. Tapp, "Quantum cryptanalysis of hash and claw-free functions," in LATIN'98: Theoretical Informatics: Third Latin American Symposium Campinas, Brazil, April 20–24, 1998 Proceedings 3. Springer, 1998, pp. 163–169.

[161] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1. 2," Submission to the CAESAR Competition, vol. 5, no. 6, p. 7, 2016.

[162] H. Kumar, S. Kumar, R. Joseph, D. Kumar, S. K. S. Singh, A. Kumar, and P. Kumar, "Rainbow table to crack password using md5 hashing algorithm," in 2013 IEEE Conference on Information & Communication Technologies. IEEE, 2013, pp. 433–439.

[163] W. Diehl, F. Farahmand, A. Abdulgadir, J.-P. Kaps, and K. Gaj, "Face-off between the caesar lightweight finalists: Acorn vs. ascon," in 2018 International Conference on Field-Programmable Technology (FPT). IEEE, 2018, pp. 330–333.

[164] C. J. Cremers, "The scyther tool: Verification, falsification, and analysis of security protocols: Tool paper," in Computer Aided Verification: 20th International Conference, CAV 2008 Princeton, NJ, USA, July 7-14, 2008 Proceedings 20. Springer, 2008, pp. 414–418.

[165] F. Pareschi, R. Rovatti, and G. Setti, "On statistical tests for randomness included in the nist sp800-22 test suite and based on the binomial distribution," IEEE Transactions on Information Forensics and Security, vol. 7, no. 2, pp. 491–505, 2012.

Appendix A

List of Publications from the Ph.D. thesis

1. S. Ghosh, S. Sampalli, A survey of security in SCADA networks: Current issues and future challenges. IEEE Access. 2019 Jul 2;7:135812-31, doi: 10.1109/ ACCESS.2019.292644 (Impact Factor: 4.43)

2. S. Ghosh, M. Zaman, G. Sakauye, S. Sampalli, An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme. Appl. Sci. 2021, 11, 2082. https://doi.org/10.3390/app1105208. (Impact Factor: 2.7)

3. S. Ghosh, M. Zaman, S. Sampalli, Quantum-Safe Asymmetric Cryptosystems: Current Solutions and Future Directions against Quantum Attacks. In Holistic Approach to Quantum Cryptography in Cyber Security (pp. 99-120). CRC Press 'Taylor and Francis Group', 2022. (Impact Factor: 4.2)

4. S. Ghosh, M. Zaman, R. Joshi , S. Sampalli, Multi-phase Quantum resistant Framework for Secure Communication between RTU and MTU", Manuscript under review after major revision, IEEE Transactions on Dependable and Secure Computing, June 2022. (Impact Factor: 6.79)

5. S. Ghosh, M. Zaman, B. Plourde, S. Sampalli, A Quantum-Based Signcryption for Supervisory Control and Data Acquisition (SCADA) Networks. Symmetry. 2022 Aug;14(8):1625. (Impact Factor: 3.11)

6. S. Ghosh, M. Zaman, R. Joshi, S. Sampalli, LARSCOM: A lightweight and robust algorithm for secure communication in Supervisory Control and Data Acquisition networks. Submitted to IEEE Access.(Impact Factor: 4.43)

7. D. Upadhyay, S. Ghosh, H. Hohno, M. Zaman, S. Sampalli. Design and development of secure SCADA/IoT (Supervisory Control and Data Acquisition/ Internet of Things)-based test bench for industrial control systems and performance

evaluation of lightweight cipher on hardware emulator. Submitted to Computers Security. (Impact Factor: 5.1)

Appendix B

Selected Publications from the Thesis

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 7, 2019, accepted June 8, 2019, date of publication July 2, 2019, date of current version October 1, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2926441

A Survey of Security in SCADA Networks: Current Issues and Future Challenges

SAGARIKA GHOSH, (Student Member, IEEE), AND SRINIVAS SAMPALLI, (Member, IEEE)
Emerging Wireless Technologies Laboratory, Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 1W5, Canada

Corresponding author: Srinivas Sampalli (srini@cs.dal.ca)

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) and industry partners Cistel Technology Inc. and Technologie Sanstream through a Collaborative Research Grant.

ABSTRACT Supervisory Control and Data Acquisition (SCADA) systems are used for monitoring industrial devices. However, their security faces the threat of being compromised due to the increasing use of open access networks. The primary objective of this survey paper is to provide a comparative study of the on-going security research in SCADA systems. The paper provides a classification of attacks based on security requirements and network protocol layers. To secure the communication between nodes of SCADA networks, various security standards have been developed by different organizations. We conduct a study of the security standards developed for SCADA networks along with their vulnerabilities. Researchers have proposed various security schemes to overcome the weaknesses of SCADA standards. The paper organizes security schemes based on current standards, detection, and prevention of attacks. It also addresses the future challenges that SCADA networks may face, in particular, from quantum attacks. Furthermore, it outlines directions for further research in the field.

INDEX TERMS Asymmetric cryptography, intrusion detection system, key management protocol, n-ary tree, symmetric cryptography, SCADA networks.

I. INTRODUCTION

SCADA systems are used as control systems for monitoring industrial processes such as oil mining, electric grids, traffic control systems, water treatment plants, space stations and nuclear systems. Modern SCADA systems have been exposed to a range of cyber attacks since they use open access networks to leverage efficiency. Failure to secure SCADA systems can be catastrophic [1]. For example, a malicious user can take control of the power supply to a city, shut down the water supply system, or cause the malfunction of a nuclear reactor.

Modern SCADA systems have a number of added features which increase the system complexities and are thus difficult to maintain. Some of the added features include control logic, communication protocols, user interfaces, and security. For example, many organizations do not tolerate data delay or data loss. Added features like firewall function and anti-virus software processes can lead to delayed delivery of data [2]. The systems must operate continuously and in tight timing [3]. Moreover, the communications are vulnerable to various threats. In the past few years, the number of cyber-attacks, in general, is rising and has been affecting power station, water, gas, and nuclear control systems. The pattern of cyber-attacks has also evolved beyond the simple attacks such as Denial of Service or Man-in-the-Middle [3].

In December 2015, due to a successful cyber-attack on SCADA, nearly 250,000 people were left without power for hours in Ukraine. After a year, another similar attack hit the country. This attack was launched by using spear phishing emails and is still in practice against industrial organizations. According to the U.S. Department of Justice, there was an attack on a small dam in Rye Brook, New York in 2013. The hackers gained access to the core command-and-control system by using a cellular modem. Although the breach occurred in 2013, it remained unreported until 2016. Furthermore, according to a FBI and Homeland Security joint report [4], there have been cyber-attacks on nuclear power plants throughout the U.S., in which the control systems were targeted. The main motive and severity of the attacks are not known, but the method used for the attack was spear phishing.

SCADA networks also comprise of resource-constrained devices such as Remote Terminal Units and Programming Logic Units, and these devices require lightweight ciphers.

The associate editor coordinating the review of this manuscript and approving it for publication was Chin-Feng Lai.

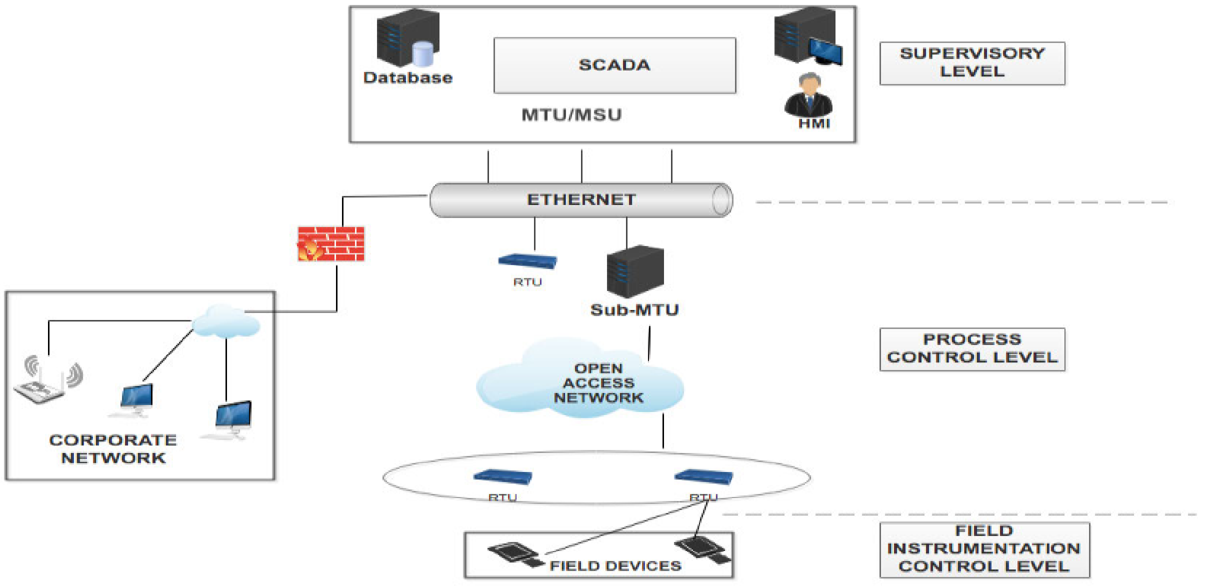


FIGURE 1. SCADA network communication architecture.

Traditional intrusion detection systems (IDSs) are now unable to protect from new threats [5]. Robust security schemes involving machine learning to detect intrusions and encryption algorithms are essential to ensure a secure encrypted communication between nodes in SCADA networks. These threats and attacks have motivated researchers and organizations to develop new robust and secure techniques for SCADA networks.

Although there are several survey papers on security threats, key management schemes, and intrusion detection systems in SCADA networks [6]–[8], the reviews do not provide a comprehensive comparison of the various schemes. The work by Sajid *et al.* [9] is an excellent survey on the security and challenges of the SCADA systems. However, the paper does not provide a comparison of all the security protocols and standards for SCADA systems. Motivated by this, our paper is an extension of the survey provided by Sajid *et al.* [9]. It gives a review of the SCADA communication structure and the recent threats faced by them. It then provides a classification and comparative study of the existing security protocols used and proposed to date. Based on the analysis, it also provides the limitations of each of the standards and protocols.

A. CONTRIBUTION OF THE SURVEY

The main contribution in this survey is to provide researchers and organizations with a report that discusses and analyzes the schemes and efforts proposed to secure the SCADA networks. It also gives a comparative study of the existing standards and schemes. Furthermore, it identifies a new threat based on quantum computing faced by SCADA networks.

B. ORGANIZATION OF THE SURVEY

Section II and Section III describe the SCADA communication structure and threats faced by such systems. Section IV describes the attacks on SCADA networks. Section V discusses the threat posed by quantum computing. Section VI gives a thorough study of on-going SCADA security schemes. Section VII discusses the primary factors used for comparison of all the schemes. Section VIII gives a critical analysis of the schemes used to secure the SCADA networks, and Section IX provides concluding remarks.

II. SCADA COMMUNICATION ARCHITECTURE

SCADA systems consist of several entities organized in a hierarchical structure [5]. They are used in monitoring various kinds of infrastructure and industries. They comprise the integration of data acquisition systems, data transmission systems and Human-Machine Interface (HMI) [5]. The HMI is a user interface that connects a person to a device. It is mainly used to visualize data, and monitor production time, machine inputs and outputs. Figure 1 illustrates a generic SCADA network communication architecture [10]–[12]. The HMI is a software interface while the hardware components are as follows [11], [12].

- Master Station Unit or Master Terminal Unit (MSU/MTU) is the control center of a SCADA network.
- Sub-MSU or Sub-MTU acts as a sub-control center. However, it is not needed in some cases. The MSU can connect to the remote station units directly.
- Remote Station Units are Remote Terminal Unit (RTU), Intelligent End Device (IED) and Programmable Logic Controller (PLC). They are used to monitor sensors and actuators to collect data values.

A communication link is shared between the MSU and Remote Station Units. Various types of communication links may be used, such as wired ethernet, WiFi or satellite link.

SCADA system architectures have four typical architectural styles [13]:

- **Monolithic:** In 1970s, control units or MTUs were hard-wired to RTUs.
- **Distributed:** In 1980s to 1990s, MTUs and RTUs communicated using communication protocols and servers. However, they did not allow Internet connection.
- **Networked:** In 2000s, SCADA architecture started using external networks like the Internet.
- **Web-based SCADA:** Currently, users can access SCADA systems using web browsers and mobile devices.

The evolution of SCADA has led to increased complexities. Some of the features responsible for this are the following [2], [13].

- Addition of new components such as computers, operating stations, communication servers and other types of resources.
- Increase in amount of data exchange between units with increase in the number of components.
- Increase in the amount of interactions between system components.
- Usage of firewalls and antivirus software that consequently slows down the processing power of the system and leads to delay in data transfer to other units.

Thus, as the size of the SCADA architecture and added features increase, the complexity of the SCADA architecture also increases. This makes managing large amount of data more difficult leading to loss of data availability. Furthermore, it makes the SCADA architecture susceptible to cyber-threats [2], [13].

III. SECURITY THREATS FACED BY SCADA NETWORKS

Like any other system or network, a SCADA network faces the following threats [1], [12].

- Loss of availability can cause power outages and can have a negative impact on the efficiency of power supply. This condition may have a cascading effect in the physical domain. Thus, achieving availability as a security goal should be one of the primary objectives of a SCADA network.
- Loss of integrity is a scenario when the attacker modifies the data, and thus, the receiver receives the changed data. This type of scenario is achievable by launching a Man-in-the-Middle attack, which can further result in malware injection and IP spoofing.
- Loss of confidentiality can be achieved by eavesdropping on a channel. It leads to the loss of privacy and stealing of data as private data is exposed.
- Repudiation is where the sender denies they have sent the data at that time.
- Slowloris, GoldenEye for operating system Kali Linux.
- And, another tool named Low Orbit Ion Cannon (LOIC) [28]
- Lack of authentication in the Distributed Network Protocol 3.0 (DNP 3) used in SCADA systems which can lead to an impersonation attack [14].

IV. ATTACKS ON SCADA NETWORKS

The usage of Internet connectivity, cloud computing, wireless communications, and social engineering on SCADA networks have made its architecture vulnerable [1]. One of the main reasons for the vulnerabilities in SCADA is the lack of strong encryption and real-time monitoring.

Attacks can occur at all layers from the supervisory level to the field instrumentation level [15]. The most common attacks are outlined in Table 1A and 1B [15]–[19].

They can also be categorized based on attacks on hardware, software, and network connection [15].

- *Attack on hardware:* This is a scenario where the hacker gets unauthenticated access to the units and tampers with them or their functions. The primary challenge in securing hardware is access control. For example, the doorknob-rattling attack [15] as explained in Table 1.
- *Attack on software:* The SCADA system utilizes a variety of software to enhance its efficiency by fulfilling the functional demands. However, due to poor implementation, it is vulnerable to SQL injection, trojan horse and buffer overflow. These are a few examples of attack on software [15].
- *Attack on network connection:* The attack on communication stack can be on the network layer, transport layer, and the application layer. Figure 2 gives a classification of attacks based on the layers of the Open Systems Interconnection (OSI) model and maps them to the violation of security goals, namely, confidentiality, integrity, availability, and non-repudiation [15].

V. POSSIBLE ATTACK USING QUANTUM COMPUTING

A. QUANTUM COMPUTER

Traditional computers are the digital electronic computers which encode information in bits, where each bit can be 0 or 1. They execute algorithms on bits using simple digital logic operations such as AND, OR, and NOT [33]. Instead, quantum computers encode information in qubits which are generated using atoms as digital bits [34]. The value of qubits is based on the rules of modern physics: superposition and entanglement principle. According to the superposition principle, each qubit can represent 0 or 1 or both at the same time. Entanglement occurs when two superposed qubits are allied with each other [34], [35]. Therefore, the number of qubits is directly proportional to the number of states held by the set of qubits [35], [36]. These two principles make quantum computing way faster than traditional computing.

A quantum algorithm was proposed to solve a binary maze problem [37]. Each line has one input and two outputs. The quantum algorithm attempted all the paths at the same time, and therefore, it solved the problem at extreme speed. Whereas, solving the maze problem was hard for a traditional computer since the size of the problem was doubling each time. For example, a 1000 step binary maze has 2^{1000} outcomes, and this took more time in the case of traditional approach [37].

TABLE 1. A. Standard attacks on SCADA networks.

Attacks	*Description of the attack*	*How can the attack be launched?*	*Few Tools that can be used to launch the attack*
Eavesdrop	It can be of two types: Passive eavesdropping and Active eavesdropping[20] .	The communication network can be wired or wireless. By accessing the network between the MTU and sub-MTUs or RTUs, the invader can install eavesdropping equipment in the network [21].	Wireshark, tcpdump and, dsniff [22].
Man-in-the-Middle (MiM)	This occurs when the attacker is in between two units and fetches the private information. The most common MiM attacks are the following [20]: Session Hijacking network server, IP Spoofing and Replay attack.	In MiM attack, the intruder monitors the traffic and injects abnormal data during the transmission and sends it to the receiver [21]. In case of a successful session hijacking and IP spoofing, it takes over the session and maintains the connection. The spoofing helps the attacker to go undetected [23].	Ettercap, SSLStrip and, Evilgrade [24].
Masquerade	The attacker uses a fake identity to pretend to be a legitimate user and steals information from the system or the network.	By launching IP Spoofing and a brute force password attack, they can use stolen passwords and logins to gain unauthorized access [21].	Ettercap, Arpspoof and Brutus[24].
Virus and worms	A malware is a malicious software or a program that corrupts the data stored in the computer. They can also lead to Distributed Denial of Service attack. Virus and worms are types of malware [25].	The intruder can send a file containing malicious code to the MTU after launching MiM or masquerade attack. For example, the virus or worm can spread through sending e-mail attachments, web link and peer-to-peer file sharing networks [25].	Any malicious code which is self-replicable and attached to .exe file in the device [25].
Trojan Horse	This is a type of malicious program disguised as a harmless file. However, unlike a virus, a trojan horse is not self-replicable. Therefore, hackers use social engineering tactics to transfer this type of virus [26].	After launching IP Spoofing or social engineering, the intruder can inject innocent looking malicious and executable code and send it as a web link or a free download to the target system. Thus, the hacker can gain access and hack the control systems [26].	Social engineering. For example, web links offering free software download.
Denial of Service (DoS)	This is a type of attack where a legitimate user is denied access to a resource. It attacks the availability requirement of a network [27].	An infected RTU by virus or worm can send random IP packets to the MTU and thus consume network bandwidth. It further leads to resource starvation.	• Slowloris, GoldenEye for operating system Kali Linux. • And, another tool named Low Orbit Ion Cannon (LOIC) [28]

D-wave, a quantum computing company, launched its first commercial quantum computer named D-Wave One in 2011, which is being used by National Aero-nautics and Space Administration (NASA) for in-depth space exploration. By 2013, they increased the number of qubits and released the D-Wave Two system. Google is also planning to use a quantum computer for big data mining [35].

TABLE 1. B. Standard attacks on SCADA networks.

Attacks	*Description of the attack*	*How can the attack be launched?*	*Few Tools that can be used to launch the attack*
Fragmentation	Fragmentation attack is a type of DoS leading to unavailability of resource [29][30].	It involves sending of over-sized datagrams. In this type of attack, the sizes of the sent datagrams are greater than network's maximum transmission unit [30].	Tools used to launch DoS attack can be used for Fragmentation attack.
Cinderella	The objective of this type of attack is to expire the security software license.	The hacker disguises their ID as a legitimate user and gains access to system by using a brute-force attack. Then the internal network clock is changed to expire the security software prematurely, thus increasing the network vulnerability [31]. Attackers can use the tools that are used to launch masquerade and brute-force attack.	For example, Ettercap for masquerade [32]. Ncrack, Hydra and Hashcat for brute-force attack [32].
Doorknob rattling	The type of attack when the failed attempts of a brute-force remains hidden from the detection system of the network [15].	At first the attacker will launch masquerade attack. Then, they try to attempt a random combination of username and passwords repeatedly on different devices to gain access. So, this leads to a few failed attempts. If the failed attempts are going undetected, this kind of attack can be successful [15].	• Ettercap for masquerade [32]. • Ncrack, Hydra and Medusa for brute-force attack [32].

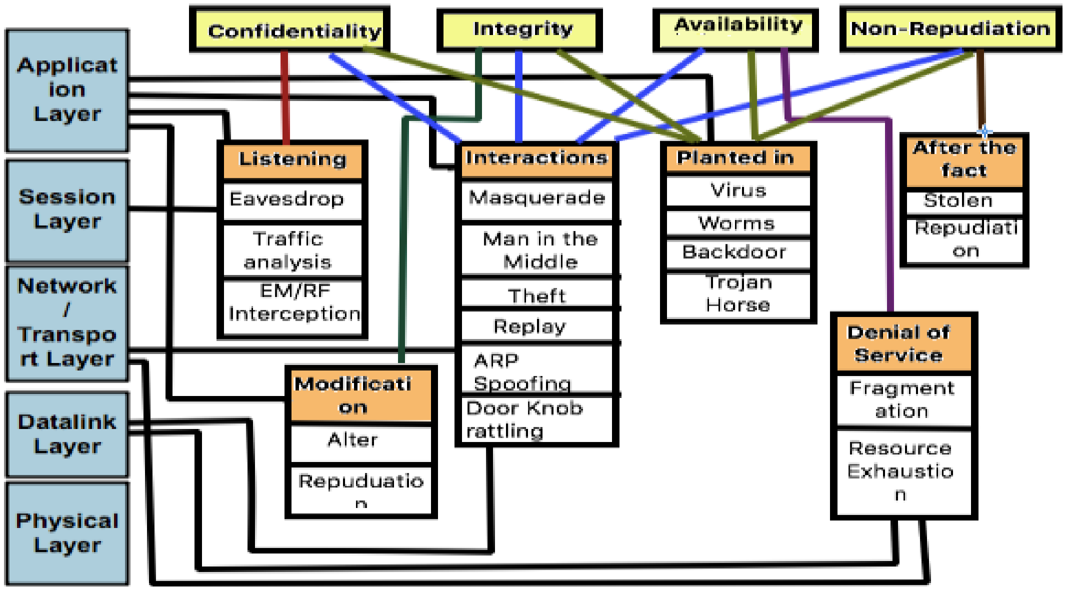


FIGURE 2. Classification of SCADA attacks in terms of security requirements and OSI layers.

B. BRUTE FORCE ATTACK BY USING A QUANTUM COMPUTER

The capacity and speed of quantum computer for solving mathematical problems make them a threat to traditional security schemes. All the encryption schemes are derived from mathematical logic. Cracking these schemes may be possible for quantum computers [38], [39]. One such problem is Elliptic curve cryptography (ECC or ECDSA).

TABLE 2. Steps in Shor's Algorithm.

Section 1: CLASSICAL PART	
Step 1:	Select a random positive integer m such that m<n. Then, calculate gcd(m,b) using the Euclidean algorithm. If gcd is not equal to 1, a non-trivial factor is obtained. Thus, the algorithm ends. Otherwise, go to Step 2.
Section 2: QUANTUM PART	
Step 2:	Calculate the period P of the sequence: x mod n, x^2 mod n, x^3 mod n, …
Step 3:	If p is odd, return to step 1. If p is even, go to step 4.
Step 4:	$\left(m^{p/2}-1\right)^2 = m^p - 1 = 0 \bmod n$, since p is even. If $m^{p/2} + 1 = 0 \bmod n$, then return to step 1. Else, go to step 5.
Step 5	Calculate $result = gcd\left(m^{p/2} - 1,\ n\right)$ using the Euclidean algorithm.

Using Shor's algorithm, a quantum computer can launch a brute force attack and crack ECC in a brief time [39].

Shor's algorithm is a quantum algorithm for factorizing a number [40]. It implies that any public key cryptography can be easily cracked. The algorithm has two sections as follows [41]. Table 2 shows the steps.

- The classical computer can compute Section 1 in Table 2. It reduces the factoring problem to an order finding problem using the Euclidean algorithm. The Euclidean algorithm is a fast scheme to calculate the greatest common divisor (gcd) of two integers [42].
- Section 2 is the quantum part which used order finding algorithm. It finds the period of the function using the Quantum Fourier Transform (QFT).

In step 2, to calculate the period of the function based on the series, Quantum Fourier Transform (QFT) is used. Using QFT increases the speed of the algorithm by evaluating the function at all points simultaneously [41]. The QFT is a linear operator when applied to any state of qubit transforms it into another state. In other words, it is applied to the vector of amplitudes of a quantum state. [43] For example, if QFT operates on a quantum state X, then it transforms it into a quantum state Y.

$$X : |x> = \sum_{i=0}^{N-1} x_i |i>$$

$$Y : |y> = \sum_{i=0}^{N-1} y_i |i>$$

The QFT refers to (1).

$$y_k = 1/\sqrt{N} \sum_{j=0}^{N-1} x_j \omega_n^{jk}, \quad k = 0, 1, 2, 3, \ldots, N-1 \tag{1}$$

where,
$\omega_n = e^{\frac{2\pi i}{N}}$ and is a primitive Nth root of unity, N is the length of vectors such that $N := 2^n$ [43].

Existing security standards and schemes are based on traditional cryptography such as Advanced Encryption System (AES), Elliptic-curve cryptography (ECC), and Secure Hash Algorithm (SHA). Therefore, they are vulnerable to quantum attacks. The transformation of quantum computing from theory to practice in the recent past has not only brought with its potential advantages but also increasing threats [38], [39].

VI. EXISTING SCADA SECURITY SCHEMES

An attack on a SCADA system may have many adverse effects. Due to this reason, organizations and researchers have been putting much effort into developing standards, protocols, and security schemes. The existing security schemes can be categorized based on: current standards, detection of SCADA attacks, and prevention of SCADA attacks.

Classification 1: Current standards can be divided into two categories: Standard Providing Guidelines and Standards acting as crypto-suites. These standards are used in practice depending on the particular industry's requirements. However, the mechanisms of thwarting attacks in the standards are either not clearly discussed or, are not strongly secure.

Thus, to add more security in the existing standards for SCADA, many researchers have proposed novel schemes. In this paper, the academic effort has been further classified into two following categories:

Classification 2: Detection of SCADA attacks consists of all the proposed intrusion detection systems for SCADA networks. The main objective is to overcome the lack of availability that is one of the security requirements.

Classification 3: Prevention of SCADA attacks consists of all the key management protocols proposed to secure the communication between the units.

A. CURRENT STANDARDS

Throughout the world, over 10 countries have proposed more than 40 standards and protocols. The available standards are described as follows [44], [45]. Few of the standards provide guidelines to secure an infrastructure from physical and cyber-attacks. Furthermore, the remaining standards include a major part that acts as a crypto-suite. In this paper, they are categorized into two: 1) Security guidelines-based Standards and 2) Crypto-suites based Standards.

1) SECURITY GUIDELINES BASED STANDARDS

a: IEEE 1402

Institute of Electrical and Electronics Engineers (IEEE) 1402-2000 is an IEEE Guide for Electric Power Substation

Physical and Electronic Security. The Power Engineering Society/Substations of IEEE sponsors the standard. It discusses security issues caused by human intrusion at power supply substations along with methods and schemes to mitigate physical and electronic intrusions [46].

In the guide, the intrusions are classified into four main categories: pedestrian, vehicular, projectile, and electronic intrusion [45], [46]. The paper also categorizes the security methods used at power control substations [45], [46].

The computer security systems include using passwords, dial-back verification, selective access, virus scans, and encryption. The guide also explains the substation security plan and categorizes it into three questions: Why is the plan required? Who may monitor the plan? What security methods are needed? According to the guide, these are the main criteria on which the security plan should be executed [45], [46].

IEEE 1402 does not solely focus on the information security. Rather, it gives a broad and general guideline for physical as well as cyber security.

b: ISO 17799 – "INFORMATION TECHNOLOGY – CODE OF PRACTICE FOR INFORMATION SECURITY MANAGEMENT"

The International Organization for Standardization (ISO) published ISO 17799 in December 2000. The ISO 17799 is an international guideline for monitoring information security management of any organization [45]. The standard refers to information as an asset that is valuable to industry. The main objective of the standard is to protect the asset by preserving confidentiality, integrity and availability [47].

ISO 17799 provides a structured guideline to control security and perform security risk assessment. It provides the following benefits [47].

- Organizational Security
- Asset Classification
- Personnel security
- Physical and environmental security
- Network management that involves media handling, backup schedules and logging.
- Access control
- Maintenance of cryptographic controls and system integrity.

ISO 17799 is the one standard that is dedicated to Information Security Management. However, ISO 17799 does not provide any evaluation methodology of a security scheme. It also does not deal with the requirements of functional and security components in an organization. ISO 15408 was developed in 2004 to alleviate some of these issues.

c: ISO 15408 – "COMMON CRITERIA FOR INFORMATION TECHNOLOGY SECURITY EVALUATION"

ISO developed the "Common Criteria got Information Technology Security Evaluation" in January 2004 [45]. The criteria are used to evaluate various functional classes as listed as follows [48].

- Audit
- Communication
- Cryptographic support
- User data protection
- Identification and authentication
- Security Management
- Privacy
- Security functions protection
- Resource Utilization
- Access
- Trusted path/channels

It has three sections. ISO 15408-1 provides the introduction and general model. ISO 15408-2 provides the functional security components, and ISO 15408-3 discusses the security assurance components [45].

However, the report does not focus on the utilization of cryptographic designs in communication and control applications [45]. Furthermore, it does not uniquely focus on the need of physical security in SCADA structure.

d: NERC SECURITY GUIDELINES – "SECURITY GUIDELINES FOR THE ELECTRICITY SECTOR: PHYSICAL SECURITY"

On June 14, 2002, North American Electric Reliability Council (NERC) releases a version 1.0 of NERC Security Guidelines discusses physical and cyber security along with the general practices for protecting the power supply infrastructure systems [45].

The general guideline focuses on the need of the physical security to maintain the integrity and availability of electric power systems, for example, promoting and deploying the security standards and procedures, periodic evaluation of the security measures, monitoring and reporting threats to the operating section, and quick recovery of the delivery services if damaged [49].

The report also guides to follow a strategy 'Protection in Depth'. The objective of this strategy is to delay the progress of an attacker. This buys time to the organization to defend and recover against the attack [49].

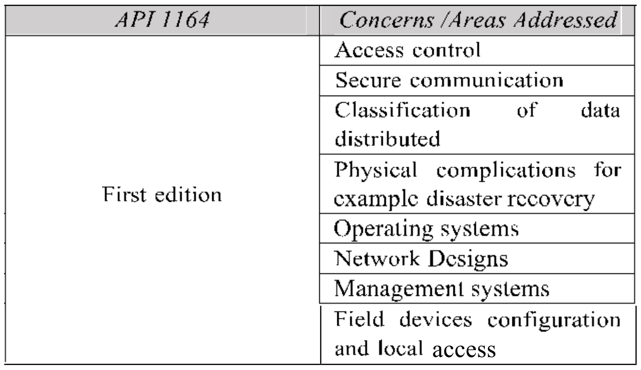
However, the security guidelines focus mainly on physical security. In 2003, NERC produced a report that deals with cyber security parameters.

e: NERC 1200 – "URGENT ACTION STANDARD 1200 – CYBER SECURITY" AND NERC 1300 – "CYBER SECURITY"

NERC developed a temporary standard named "Urgent Action Standard 1200" for setting a set of security requirements for the energy industry infrastructure. NERC adopted this standard on August 13th, 2003 for a one-year period and later, it extended the standard till August 2006 [45].

NERC developed NERC 1300 to replace NERC 1200 by addressing the security requirements and recommendations mentioned in NERC 1200 [45], [50]. NERC 1300 focuses on both physical and cyber security. The report has a section that implies that a responsible industry should follow the System Security Management to prevent any malicious

TABLE 3. Concerns addressed in API 1164.



API 1164	*Concerns /Areas Addressed*
First edition	Access control
	Secure communication
	Classification of data distributed
	Physical complications for example disaster recovery
	Operating systems
	Network Designs
	Management systems
	Field devices configuration and local access

cyber activity. The Management section mainly involves the following security measures [50]:

- Account and Strong Password management.
- Using security patch manager to check security updates.
- Using anti-virus monthly.
- Performing vulnerability assessment at least annually.
- Preserving and auditing system logs quarterly.
- Using operating status monitoring tools.
- Back-up of information on computer systems.
- Disabling unused ports.

NERC 1200 and NERC 1300 are security guidelines for the energy industry infrastructure. They do not provide security features for the oil and pipeline infrastructure. Therefore, the American Petroleum Institute developed a standard that provides security guidelines for control systems of oil and pipeline systems.

f: API 1164 – "SCADA SECURITY"

API 1164 has three editions. The first edition was released in September 2004. It specifies guidance to secure the SCADA system used in the oil and pipeline infrastructures [45], [51]. It addresses the following issues mentioned in Table 3 [45].

The second edition is the API – "Security Guidance for the Petroleum Industry." Oil and gas infrastructures utilize this standard to prevent terrorist attacks [45].

The American Petroleum Institute and the National Petrochemical and Refiners Association mutually developed the third edition named API- "Security Vulnerability Assessment Methodology for the Petroleum and Petrochemical Industries". It is utilized for evaluating various kinds of threats, vulnerabilities, and aftereffects of terrorist attacks [45].

The above-discussed standards are general guidelines to protect the infrastructure of an organization. They do not involve any in-depth discussion of cryptographic algorithms or any technical methodology to detect or protect from any attack. However, the following standards use crypto-suites.

2) CRYPTO-SUITE STANDARDS

a: IEC 62210 – "DATA AND COMMUNICATION SECURITY"

In 1999, IEC 62210 was developed by the International Electrotechnical Commission (IEC) as the report of IEC TC 57 AHWG06. Later, AHGW06 was systemized into Working Group (WG) 15 upon Data and Communications Security. Later, it was published in 2003. The IEC TC57 WG15 developed the cybersecurity standards for power control system communications [45].

The working group report describes the security process for the power control systems which involves the corporate security policy, network security protocol, and the end to end application security. The security scheme was also utilized for encrypting communication in the network [45].

AHWG06 issued the report recommending establishing the following tasks [45]:

- Consequence analysis combined with ISO 15408
- Attention to the application layer
- Address key management
- Address end-to-end security

However, the above recommended tasks were challenging to resolve at that time [45]. Therefore, the following standard was developed as an extension of IEC 62210.

b: IEC 62351 – "DATA AND COMMUNICATION SECURITY"

International Electrotechnical Commission (IEC) developed IEC 62351 to address the deficiency in IEC 62210. The standard is classified into as shown in Table 4 [45], [52]:

Using TLS security, IEC 62351 provides defense mechanisms against various attacks including spoofing, message replay attack and to some extent Denial-of-Service (DoS) attacks. However, it involves simple encryption schemes.

Immediately after the 9/11 attack, the American Gas Association (AGA) decided to improve the security mechanism which can protect SCADA communication from malicious users. The primary purpose of the standard was to develop a security scheme which can provide security as well as save time and computation cost [44].

c: AGA-12 – "CRYPTOGRAPHIC PROTECTION FOR SCADA COMMUNICATIONS GENERAL RECOMMENDATIONS"

Traditional security protocols used in SCADA systems such as IEC 60870, DNP3, IEC 61850 and Modbus lack proper security services [14]. However, the new protocol AGA-12 provides security features to the SCADA systems. It uses cryptographic suites to secure the wireless communication between field devices and the MTUs [14], [45]. The steps in AGA-12 is described in Table 5 [44].

AGA-12 provides confidentiality, integrity and authentication. However, it fails to provide availability. It does not defend against DoS attacks. Furthermore, AGA-12 uses RSA as the key management protocol which has been cracked recently [53].

Furthermore, the current standards including IEC 62210, IEC 62351 and AGA-12 fail to provide two main security requirements, namely, defense against DoS attacks and a strong key exchange protocol.

The aforesaid studies have research gaps that fail to address availability and secured communication channel. Therefore,

TABLE 4. Classification of IEC 62351.

Sections	*Schemes used*
Security for profiles including TCP/IP	It uses Transport Layer Security (TLS) for secure transactions over the internet. It provides confidentiality, integrity, and authentication.
Security for profiles including MMS	For Transport Layer which includes layer 1 to layer 4 of the OSI Reference Model, Transport Layer Security is used. The report describes a set of protocols, how to use them, and the requirements for Application Layer which includes layer 5 to layer 7 of the OSI Reference Model.
Security for derivatives (DNP 3.0)	For network versions which run over TCP/IP, the standard uses TLS encryption. For the serial version, it uses an authentication mechanism named Hashed Message Authentication Code (HMAC).
Security for IEC 61850 peer-to-peer profiles	For client/server, the standard utilizes TLS and MMS. For Generic Object-Oriented Substation Events (GOOSE), it uses analog and digital multicast.

researchers have proposed schemes to overcome these limitations in SCADA networks.

In this paper, the proposed schemes are categorized based on limitations addressed.

- Detection of SCADA attacks: It involves the security schemes addressing the availability issue in the SCADA networks. Most of the schemes are based on machine learning algorithms.
- Prevention of SCADA attacks: The discussed schemes address the key exchange and management issue in SCADA networks.

B. DETECTION OF SCADA ATTACKS

Traditional standards and Intrusion Detection Systems (IDSs) such as firewalls used in SCADA are not strong enough to cope up with emerging attacks [5]. To increase the immunity in SCADA, machine learning algorithms, such as Naïve Bayes, Random Forest, C4.5 decision tree algorithm, Support Vector Machine, etc. are used to detect intrusion in the network [54].

1) RULE-BASED INTRUSION DETECTION SYSTEM FOR SCADA NETWORKS

The proposed IDS uses a rule-based in-depth protocol analysis along with a Deep Packet Inspection (DPI) method. The model establishes a new set of intrusion recognition rules. The rule-based scheme contains two sub-schemes; namely, signature-based detection and model-based detection [55]. Signature-based detection utilizes a blacklist approach and is used for detecting a more significant amount of false spontaneous messages, unauthorized commands between nodes, and buffer-overflow. The model-based detection builds a model based on an in-depth analysis of the protocol. The created models portray the expected behavior of the protocol. It uses protocols and traffic pattern to generate the expected behavior [55]. It can detect known attacks as well as its source. Using the proposed IDS along with IEC/104 protocol, unknown attacks may be diagnosed in the SCADA network [55]. However, the proposed rule-based IDSs do not ensure the detection of novel or unidentified intrusions that pass through traditional IDS in open access networks.

2) NETWORK ANOMALY DETECTION FOR M-CONNECTED SCADA NETWORKS

Usually, IDSs and security schemes are for SCADA systems using open access networks. However, there is no intrusion detection mechanism for closed and isolated SCADA networks. This kind of SCADA architecture is referred to as an ‘m-connected’ SCADA network [56].

The model uses a dynamic detection for detecting intrusions with a packet logger and packet sniffer followed by a pattern matching algorithm. It generates new rules and stores them in a database. It further uses new rules for the next round [56]. The proposed scheme is based on rule-based intrusion detection and further research is needed for accurate implementation [56]. Furthermore, the scheme does not guarantee detection of unidentified attacks.

3) L_P - NORMS IN ONE-CLASS CLASSIFICATION FOR INTRUSION DETECTION IN SCADA SYSTEMS

In 2014, an intrusion detection system was proposed to detect abnormal activity in the network that is not detected by the traditional IDS or firewalls. It uses a machine learning based on the one-class classification algorithm for live detection of unnoticed cyberattacks [5].

The paper analyses two approaches: the support vector data description (SVDD), and the kernel method [5]. It uses kernel principle as non-linear methods to detect patterns, and interdependencies within the real-world data. SVDD maps the data to the subspace which is optimized for one-class classification. The paper concludes that the proposed method showed the highest error detection and the lowest false alarm rates after conducting tests on a real dataset with several cyber-attacks [5].

TABLE 5. Steps in the AGA-12 standard.

Steps	*Sub – Step(s)*	*Description*
Perform system security audit	• System-wide network audit must be done. • Following the audit, risk assessment is required. • Security goals must be set.	During risk assessment, cost-benefit analysis is done. When benefits outweigh the cost, the AGA-12 is implemented in the SCADA network.
Agreement of Hardware and Software Modules to be used	• Guidelines are provided for testing of hardware and software modules. • The guidelines must also provide the cryptographic process agreement.	The algorithms which are accepted and permitted by National Institute of Standards and Technology (NIST), AGA 12 are as follows. • Advanced Encryption System (AES) Encryption with a key length of minimum 124 bits. • Rivest-Shamir-Adleman (RSA) with a key length of minimum 1024 bit. • Elliptic Curve Digital Signature Algorithm (ECDSA) with a key length of minimum 160 bits. • Secure Hash Algorithm (SHA-1).
Performing a post-deployment security audit	• Implement AGA-12 • Post Implementation audit	After implementation, it involves a detailed audit throughout the network. If any security threat is detected, the necessary compliance level should be approached.

4) ONE-CLASS SUPPORT VECTOR MACHINE (OCSVM)

In 2014, Maglaras and Jiang [57] developed a One-Class Support Vector Machine model for detecting new attacks in the SCADA network. The proposed model addresses the following issues:

- The research community has developed many IDS algorithms for SCADA networks. Most of them are rule-based algorithms which make them incapable of detecting any new intrusions. In a real-time application, when any new anomaly is present, it fails to predict the behavior of the system [57].
- Other algorithms such as K-nearest neighbor (KNN), Hidden Markov models, and Support Vector Machines are used for detecting intrusion. However, they require learning of expected anomaly. Thus, these schemes may be sensitive to noise present in the training dataset [57].
- Negative selection algorithms can fail in the case of real-time application because of enormous diversity in real time data [57].

The proposed IDS is an algorithm to detect anomaly without any labeled data for training. Network traces train the OCSVM model without the use of open access net-works. These features help the proposed IDS to perform in real time. Table 6 outlines the steps in the detection process [57].

However, the OCSVM model does not manage false positive results.

TABLE 6. Steps to detect intrusion using OCSVM.

Step	*Description*
Step 1	Data analysis.
Step 2	Attributes in the network traces are extracted. The attributes, rate. and packet size, are used to train the model.
Step 3	Integration of OCSVM Module.
Step 3.1	The network traces data, and the extracted attributes are used to train and generate the model.
Step 3.2	The model is tested for real-time anomaly detection.
Step 3.3	The detected anomalies are classified based on the severities.
Step 3.4	The main correlator is alarmed regarding the detected anomalies.

5) OCSVM MODEL COMBINED WITH K-MEANS RECURSIVE CLUSTERING FOR INTRUSION DETECTION IN SCADA SYSTEMS

One-class classifiers suffer from false positives and overfitting. False positive is a scenario when the IDS detects abnormal behavior but there is no intrusion in real. Overfitting is a case when the model begins to learn the details and errors in the training data. These two factors decline the performance of the model on the new data [58].

To address these two issues, Maglaras and Jiang [58] developed an intrusion detection model to detect the malicious network traffic in SCADA. The model includes the One-Class Support Vector Machine (OCSVM) with Radial Basis Function (RBF) kernel and recursive k-means clustering [58]. OCSVM is an extension of support vector machines and is used to detect the outliers in the data. The k-means clustering algorithm is used to cluster the outliers and sort them with two clusters. OCSVM obtains two values, namely, maximum and minimum negative value [58]. The cluster which is near to minimum negative value represents severe alerts, and therefore, the cluster is used as input when there is recalling of k-means clustering. This step is repeated till the after-k means clustered are in a single cluster. After the completion of K-OCSVM phase, the model distributes the severe alerts among the nodes in the SCADA structure [58].

6) A HYBRID MODEL FOR ANOMALY-BASED INTRUSION DETECTION IN SCADA NETWORKS

Usually, intrusion detection systems when deployed in real time lead to high computational and time costs. These two factors affect the performance of a SCADA network [16].

In 2017, anomaly-based intrusion detection was developed using a feature selection model after removing redundant data. Irrelevant data can affect the efficiency of SCADA systems. This proposed scheme is time-saving, has low computational complexity and has 99.5% accuracy of detecting specific-attack labeled [9]. At first, the J48 classifier is used to train the dataset and then, to develop the model, Bayes Net classifier is utilized. The proposed model is tested on a database with three types of labeling as follows [9].

- Case 1: binary-labeled
- Case 2: categorized-labeled
- Case 3: specific attack labeled

The above-mentioned scheme focuses on the availability limitation in the SCADA networks. The schemes propose novel IDSs that detect any abnormal network behavior, which can lead to DoS attacks. However, the scheme fails to secure the communication channel. The following section on the prevention of SCADA attacks focuses on securing the communication channel with novel key exchange and management schemes in SCADA networks.

C. PREVENTION OF SCADA ATTACKS

The existing standards use vulnerable key management protocols that do provide a strong secure communication channel.

Encryption and key management are crucial in communication between nodes in a SCADA architecture. Key management schemes developed for SCADA can be categorized into two, namely, centralized key distribution and decentralized key distribution [6]. They can also be categorized into symmetric key cryptography, asymmetric key cryptography, and hybrid key cryptography [7]. In this paper, another classification concerning self-healing property is added.

1) SYMMETRIC KEY CRYPTOGRAPHY

a: SCADA KEY ESTABLISHMENT(SKE)

SKE categorizes SCADA communication into Controller - Subordinate (C-S) which uses symmetric key cryptography and Peer to Peer (P-P) which uses public key cryptography. The controller is the sub-MTU or sub-MSU, and the subordinate is the RTU. Peer-to-peer communication is between two sub-MTUSs or two RTUs [6].

For C-S communication, SKE uses four kinds of keys: Long-Term key, General Seed Key (GSK), General Key (GK) and Session Key (SK). The Long-Term Key (LTK) is manually distributed between the controller and subordinate [59]. The controller stores the GSK and is used by Cryptographic Authority (CA) to produce GK. By using two keys, GSK and LTK, the GK is generated and is then shared between the controller and the subordinate. While transmitting GK, it is encrypted by LTK. The session key is generated by using GK, sender's identity and TVP (Time-Varying Parameters). TVP field involves timestamp and a sequence number [6], [59].

For peer-to-peer communication, SKE uses four different keys: Cryptographic Authority Public Key (CAPK), Public key Signature Key (PKSK), Common Key (CK) and Session Key (SK). The CAPK is shared among sub-MTUs while the PKSK is shared among the sub-MTUS, MTU and Cryptographic Authority (CA). The common key is generated by following a key exchange algorithm. The methodology to generate session key is the same as that of C-S communication. The session key is used to encrypt the messages transmitted [6], [59].

However, the RTU to RTU communication is not directly allowed. Since the communications are treated differently in different conditions, it increases the overall overhead and complexity. Furthermore, the long-term keys are managed manually [6], [59].

b: SCADA KEY MANAGEMENT ARCHITECTURE (SKMA)

In comparison to SKE, the implementation of SKMA architecture is more simplified. The architecture establishes the key exchange protocol among the Key Distribution Cen-ter (KDC), and any two nodes. The long-term keys are accumulated only on the required nodes and on the KDC which is the third party. The design uses three main keys [12], [59]:

- Long-Term Node-KDC key is used to yield keys for communication and is manually shared between a node and the KDC.
- Long-Term Node- Node key is distributed between the nodes that require to communicate with each other.
- Session Key is used for encrypting the information transmitted from one node to another. Once the key establishment is completed, the session key is generated by using pseudorandom number function, nonce-key and a timestamp [12].

The SKMA scheme does not use GSK. The key exchange in SKMA only happens when a new node joins the SCADA network [12].

Nevertheless, the SKMA does not provide the following security features [59].

- SCADA systems mostly use broadcast communication. However, the SKMA cannot provide such a mechanism.

This protocol does not provide any confidentiality and integrity.

c: LOGICAL KEY HIERARCHY (LKH)

To address one of the issues, the LKH protocol was developed. LKH protocol provides secure broadcast communication [6] [7]. It is based on an architecture of the logical tree of keys [12]. It maps all the nodes of the SCADA network as the leaves of a structure tree. Each node stocks the entire symmetric keys from the root to its leaf. When a node leaves or joins, the node keys from its leaf to the root is updated so that the security of the network is preserved [12]. For example, Figure 3 [12] explains the mechanism when a node joins the network.

d: ADVANCED KEY-MANAGEMENT ARCHITECTURE (ASKMA)

To enhance the efficiency of SKMA and LKH, the ASKMA was proposed [6]. It provides both message broadcasting and secure communication. It also keeps a minimal load on the resource-constrained nodes [6], [12].

In ASKMA, the LKH protocol is used by Choi *et al.* [12] for message broadcasting in 2009. The nodes of the SCADA networks such as RTUs, sub-MTUs, and the MTU are organized in two tree structure: binary tree and n-ary tree. The MTU to sub-MTU follows a binary tree structure whereas the sub-MTUs to RTUs follows n-ary tree structure [6].

The ASKMA protocol evenly spreads the computations to the sub-MTUs and MTUs which are high power nodes and keeps a minimal load on the low power nodes like RTUs. Therefore, the nodes are arranged logically in a tree structure, n-ary or binary tree, depending on their computational power [12].

When a new node is added to the SCADA network, the ASKMA follows a Join Protocol. Any key received by a new RTU must be independent of any existing keys in the nodes of the tree. It preserves backward confidentiality. When a new node joins the tree, the KDC updates all the keys from its leaf to the root on the freshly joined RTU's path. It uses a hash function for renewing the keys. The Join Protocol has the following steps [12].

Step 1: The KDC renews all $K_{i,j}$ to $K^{/}_{i,j}$ where $K^{/}_{i,j} = H(K_{i,j})$.

Step 2: In case the RTUs have keys belonging to $K_{i,j}$, each RTU updates their key $K_{i,j}$ to $K^{/}_{i,j}$.

Step 3: With K_m, the KDC encrypts all $K^{/}_{i,j}$ and transmits the encrypted information to the newly joined RTU which is N_m.

When a node leaves the SCADA network, the ASKMA follows a Leave Protocol. Similar to the Join Protocol, all the keys throughout the key path updated with new keys [12]. However, the leaving node N_m should not be able to use the updated keys. This makes the Leave Protocol a little more complicated than Join Protocol. The following are the steps of Leave Protocol [12].

Step 1: The KDC removes the RTU which is parting.

Step 2: It then updates the remaining keys by executing a key generation algorithm such that the leaving RTU does not know the updated key. Consequently, the departed RTU is unable to compute the new keys.

Step 3: Each RTU updates its keys by using the hash function.

Step 4: If the RTUs are unaware of their sibling keys, KDC encrypts the new keys and sends them to those RTUs.

Step 5: The departed node knows all the ancestor keys of the sibling RTUs. Therefore, the KDC encrypts all the updated keys with sub-MTU's private key and transmits to the sub-MTU. The sub-MTU encrypts the received keys with the child RTU's key and then sends it to each child RTU.

ASKMA supports broadcast and multicast communication. However, it does not offer efficient multicast communication. To solve this issue, ASKMA+ was proposed [6]. By reducing the number of stored keys, it provides efficient multicast and broadcast mechanism [6]. However, ASKMA and ASKMA+ do not address the availability issue in SCADA [6].

2) HYBRID KEY CRYPTOGRAPHY

a: HYBRID KEY MANAGEMENT ARCHITECTURE(HKMA)

To satisfy the availability requirement, Choi *et al.* [60] proposed a Hybrid Key Management Architecture (HKMA) which supports a replace scheme [60]. The scheme includes an operation of the replace protocol in case of compromised or broken main device. It uses a public key cryptosystem in MTU to sub-MTU communication which has high performance, and symmetric key cryptosystem in sub-MTU to RTU which has low performance. Thus, it reduces the number of keys to be stored in the MTU [60].

b: ADVANCE HYBRID KEY MANAGEMENT ARCHITECTURE(AHSKMA)

Rezai *et al.* [6], [61] proposed a scheme based on hybrid key management architecture to tackle the availability issue in SCADA networks and to increase the performance and security of HKMA. It follows ECC for MTU to sub-MTU communication. Since RTUs have limited computational resources, symmetric cryptography is used for sub-MTU to its RTUs communication. This scheme makes the architecture suitable for the environments with resource constrained devices and supports unicast, multicast and broadcast communications [61]. Figure 4 shows the mechanism of the protocol.

The Iolus [62] Framework is used while connecting the MTU and RTUs. The MTUs act as the Group Security Control (GSC) and the sub-MTUs act as the group security intermediary (GSI). The architecture consists of four

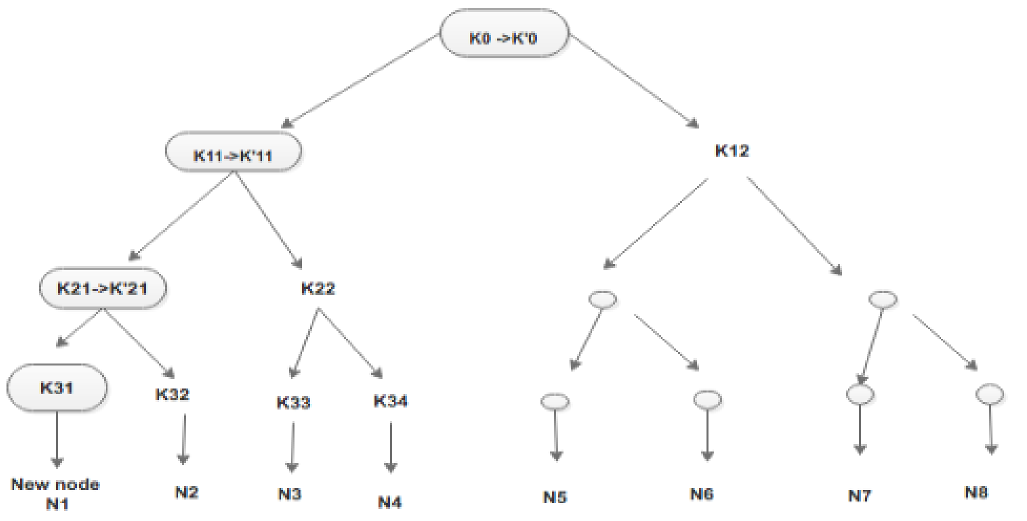


FIGURE 3. Update Mechanism of LKH protocol when a new node joins.

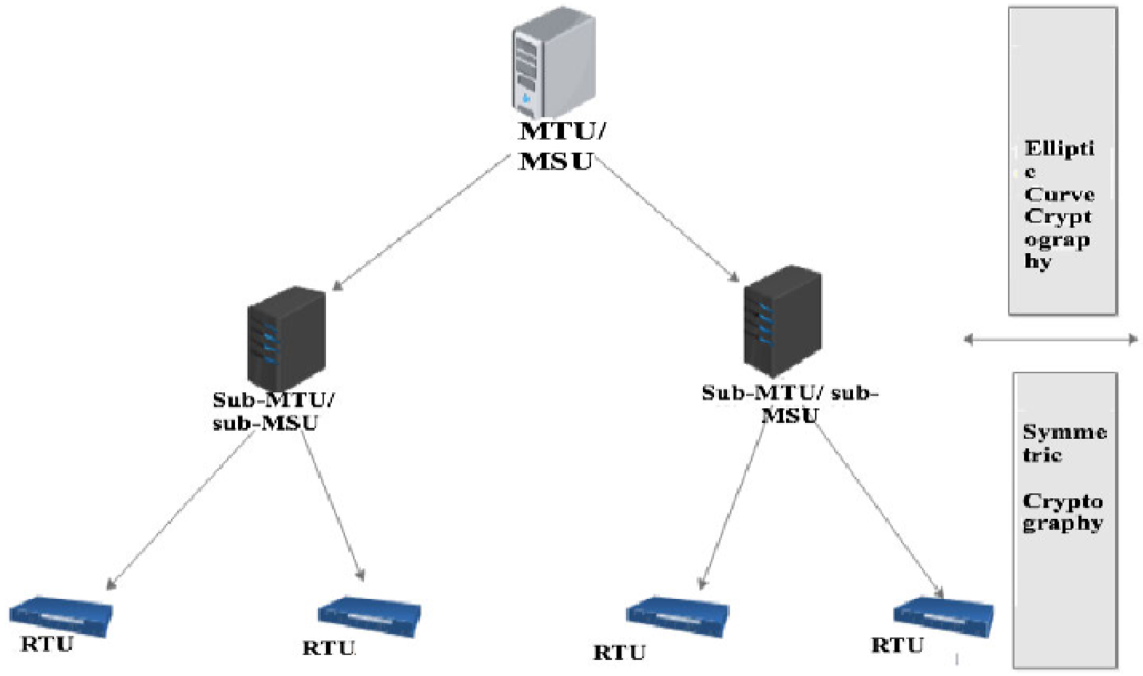


FIGURE 4. Mechanism of AHSKMA.

phases: Setup phase, Join Phase, Leave Phase and Replace phase [61].

a. **Setup Phase:** In the first phase, the group key is generated by the MTU and is shared with RTUs and IEDs.
b. **Join Phase:** Similar to LKH and AHSKMA, the MTU updates all the keys of the remaining nodes in the SCADA network as soon as a new node joins.
c. **Leave Phase:** This phase is also similar to the leave protocol of the AHSKMA.
d. **Replace Phase:** In case the MTU is damaged, it

is replaced by its backup device. Each MTU and sub-MTU has a backup device. While backing up the broken device, the Join phase and the Leave phase are performed concurrently.

The Replace Phase resolves the availability issue in SCADA networks. In this scheme, the session is produced using a hash function, a key, and TVP with a sequence number and timestamp [61]. So, HSKMA also guarantees the freshness of key along with availability.

Both HKMA and AHSKMA provides replace scheme to satisfy the availability requirement, but the affected devices stop working during the replacement. To solve this issue, LiSH+ was proposed [6], [63].

3) SELF-HEALING GROUP KEY DISTRIBUTION

a: LIMITED SELF-HEALING KEY DISTRIBUTION (LiSH+)

LiSH+ is an efficient group key management scheme which utilizes a self-healing procedure having collusion resistance capability and effective revocation [63]. The scheme involves five phases: initialization, rekeying, self-healing mechanism, the addition of new nodes, and reinitialization. It uses a bivariate polynomial to lower the storage burden from RTUs [63]. It also uses intrusion detection system to detect compromised and eliminate users. These features provided helps LiSH+ to enhance the security of SCADA networks [63].

However, the LiSH+ focuses on only two requirements: availability, and efficiency [63]. It does not focus on the authentication mechanism.

4) ASYMMETRIC KEY CRYPTOGRAPHY

a: ID-BASED KEY MANAGEMENT ARCHITECTURE

Lim [64] proposes an ID-based key management architecture (ID-KMA) based on pairing algorithm based on elliptic curves. The architecture addresses the issues of the public key cryptography with a digital signature. It involves fast and efficient session key establishment along with session key recovery protocol. It removes the concept of the digital certificate which minimizes the overhead.

The architecture involves the role of three units of SCADA: Key Management System (KMS), MTU and RTUs. The KMS is linked with the MTU, and the MTU is connected to the RTUs. The KMS communicates with RTUs through MTU [64].

The ID-based Key Management architecture uses four main keys [64] as described in Figure 5.

- **Emergency Key (EK):** This is a symmetric key stored in every component in the architecture: KMS, MTU, and RTUs. In case if the private key of each unit or master key of KMS gets compromised due to malicious attacks, the EK is used for key recovery or system restoration. Therefore, EK should be kept in a secure area.
- **Long Term Key (LTK):** It is an ID-based public and private keys of each node.
- **Update Key (UK):** It is a symmetric key distributed among the MTU and its RTUs. It is used to share the session key.
- **Session Key (SK):** The session key is shared among MTU and the RTUs. It is also shared between the RTUs. The SK is used to encrypt the communication.

The ID-based key management protocol is composed of four phases [64].

- **EK setup:** The EK is stored in each component of the architecture in advance.
- **Initialization:** The initialization has two stages. In the first stage, the KMS produces system parameters (SP) which are public and generates MTU's and RTU's LTK. The SK and LTK are encrypted with EK. The KMS shares the encrypted SP and LTK with the MTU. In the second stage, the KMS distributes a UK and LTK to each component so that the MTU can share an SK with RTUs. The first stage is LTK distribution and the second stage is the UK distribution.
- **RTU-RTU session key establishment**: This phase focuses on the secure communication between the RTUs with the usage of session key (SK) and initially shared update key (UK).
- **MTU-RTU session key establishment:** The session key is distributed among MTU and RTU to have a secure communication. The MTU sends a session key request to the RTU.
- **RTU-MTU session key establishment**: Similarly, the session key is established between MTU and RTU. The RTU sends a session key request to the MTU.

All the afore-mentioned key management protocols are based on traditional cryptography schemes which are vulnerable to quantum attacks [38]. Furthermore, public key algorithms tend to increase the computational and time cost [65].

Therefore, the following scheme named as Nth Degree Truncated Polynomial Ring (NTRU) is proposed to defend against quantum attacks.

b: NTRU CRYPTOGRAPHIC ALGORITHM FOR SCADA NETWORKS

The key management scheme is based on a faster and light-weight public key algorithm named NTRU cryptography [65]. The cryptographic algorithms in IEC62351 and AGA-12 have performance issues when apples to SCADA network security. They are time and power consuming [65], [66].

Due to various security and performance complexities of SCADA systems [66], [67] [6], NTRU was developed. It is a public key scheme based on lattice-based cryptogra-phy [68], [69]. The security of the cryptography depends on a hard problem known as Short Vector Problem [65], [68]. The encryption and decryption use polynomial operations which makes the system faster [65]. Therefore, it has better processing speed than traditional schemes and is suitable for real-time requirements of SCADA security [70].

The NTRU algorithm is also known as post-quantum cryptography and has been resistant to quantum attacks [65]. The scheme has two sub-algorithms, namely, NTRU Encrypt which is used for encryption, and NTRU Sign which is used for generating a digital signature. The scheme comprises of three phases [65]:

- **Key Generation and Certificate Creation phase:** In this phase, public and the private key of the RTU and its digital certificate is generated. For this, it uses a public key infrastructure. In this phase, other than RTU, two components play their roles. One, Local Key Store (LKS) and another, Certificate Authority (CA). The phase has the following steps [65].
  Step 1: The RTU generates a public key and private key using key generation algorithm. The algorithm uses algebraic structures of certain polynomial rings and is based on the Short Vector Problem. It then stores the generated key pair in the LKS.
  Step 2: The RTU then sends a request containing its public key to CA for generating a digital certificate. The CA in return creates a digital certificate and directs it to the RTU.

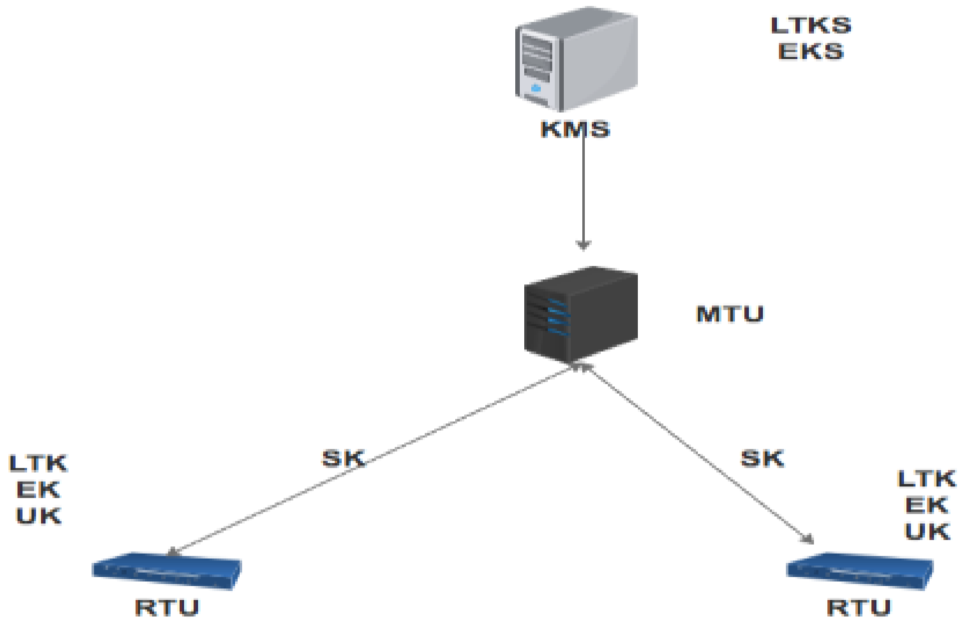


FIGURE 5. Architecture of ID-KMA.

- **NTRU Encryption:** In this phase, the RTU uses the receiver's public key, generated by the CA, to encrypt the messages. The messages are converted to a ring of truncated polynomials modulo. The receiver then decrypts the cipher using its private key.
- **NTRU based authentication:** In this phase, it is verified that the encoded message, which is in the state of a truncated polynomial ring, is validated. This phase uses a procedure built on a non-keyed hash function to ensure the integrity and authenticity of the message. The scheme creates a message digest by using the hash function. The message digest is then digitally signed by using the RTU's private key. Thus, it generates the digital signature. Therefore, the RTU sends the encrypted message and its digital signature to the receiver. The receiver uses its own NTRU private key to decode the message and generates the message digest (MD1) following the same procedure. The digital signature is then decrypted using the RTU's public key. The receiver gets the expected message digest (MD2). It then verifies whether MD1 and MD2 are equal or not. If they match, the signature is verified [65].

Even though NTRU is not yet vulnerable to quantum threats, a quantum computer can crack the algorithm using brute-force [71]. There are further challenges in post-quantum cryptography as follows [72].

- Need to improve the efficiency of the algorithm.
- Need to build confidence in the scheme.
- Need to improve the usability of the algorithm.

The existing standards have research gaps that have been addressed by the above-discussed security schemes. However, all the schemes are based on arithmetic operations. The emergence of quantum computers is proven to be beneficial as well as precarious to the cyber world. By launching a brute-force attack using Shor's or Grover's algorithms, these schemes can be broken. Therefore, there is a research gap in securing the SCADA networks from quantum attacks.

TABLE 7. Classification of current Standards.

Guideline based Standards	Crypto-suites based standards
IEEE 1402	IEC 62210
ISO 17799	IEC 62351
ISO 15408	AGA-12
NERC Security Guidelines	
NERC 1200	
NERC 1300	
API 1164	

VII. PRIMARY FACTORS USED FOR COMPARATIVE STUDY

The comparative analysis in this paper is based on the primary factors in each category. In case of current standards, the current standards are categorized into two classes as shown in Table 7. In this scenario, the primary factors used for comparison are as follows:

- Information Security Policy is a set of security rules governed by an industry that is imposed on the users of its system [73].
- Vulnerability and risk assessment are the processes where the weaknesses in a system are detected, analyzed and prioritized by the organization. The analyzed results are used to recommend security requirements in the system [74].
- Information security infrastructure is a set of security rules to protect only critical fundament such as airports,

TABLE 8. Comparative analysis of current standards used in SCADA systems.

CURRENT STANDARDS			*Organizational Security*	
Standards	*Information security policy*	*Vulnerability and risk assessment*	*Information security Infrastructure*	*Security of third-party access*
AGA 12	Yes	Yes	Yes	Yes
API 1164	Yes	No	No	Yes
ISO 17799	Yes	No	Yes	Yes
NSERC Security Guideline	Yes	Yes	Yes	Yes
NERC 1200	Yes	No	Yes	No
NERC 1300	Yes	Yes	Yes	Yes
IEC 62210	Yes	No	Yes	Yes
IEC 62351	Yes	No	No	No
IEEE 1402	Yes	No	Yes	No
ISO 15408	Yes	Yes	Yes	No

TABLE 9. Comparative analysis of crypto-suite based SCADA standards.

Crypto-suite based Standards	**Presence of Key Management scheme**		**Strength of Encryption**		**Sustaining Security Requirements**		
	Strong	Weak	Strong	Weak	Confidentiality	Integrity	Availability
IEC 62210	No		Yes, weak		Yes	Yes	No
IEC 62351	Yes, weaker than AGA-12		Yes, weaker than AGA-12		Yes	Yes	Yes
AGA-12	Yes, weak		Yes, weak		Yes	Yes	No

nuclear power plants and traffic control systems. It is similar to the Information Security Policy [73].
- Third Party access or Outsourcing is giving access to service providers, vendors and contractors that can lead to credential theft and data risk management. To overcome these security concerns, the organizations extend the security policy. For example, the third party can be given access to a separate domain from the internal network, by using firewalls [75].

Furthermore, the crypto-suites standards are compared based on the following factors.
- Presence of Key Management Protocol in the standard and the strength of the protocol.
- Presence of Strong Encryption and strength of the encryption algorithm used in the standard.
- Sustaining security requirements refers to the existence of confidentiality, integrity and availability in the security scheme of the standard.

The strength of the key management and encryption scheme depends on the resources and time utilized to crack the scheme.

In case of detection of SCADA attacks, the primary factors are as follows:
- Known Attack Detection is the scenario where any traffic is categorized as an attack if the features of that particular traffic fall under the domain of attacks stored in the IDS database.
- New Attack Detection is the scenario where any traffic with unique behavior will be detected.
- Open Access networks are the public networks where the connected devices are exposed to each other. The public networks are vulnerable to various cyber threats. The private networks that provides access to the legitimate user.
- False positive is the situation where the IDS can detect the false alarms. False positives are the consequences where an activity is classified as abnormal even if its behavior is normal.

In case of prevention of SCADA attacks, the primary factors used for critical analysis are as following:
- The efficiency of the encryption scheme depends on the amount of computation resources utilized by

TABLE 10. Comparative analysis of detection schemes of SCADA attacks.

Detection of SCADA attacks	*Known Attack Detection*	*New Attack Detection*	For Open access networks	Distinguish false positives
Rule-based	Yes	No	Yes	No
IDS for m-connected SCADA networks	Yes	No	No	No
l_p– norms in One-Class Classification	Yes	Yes	Yes	No
OCSVM	Yes	Yes	Yes	No
OCSVM with K-means	Yes	Yes	Yes	Yes
Hybrid model	Yes	No	Yes	No

TABLE 11. Comparative analysis of prevention schemes of SCADA attacks.

Prevent SCADA attacks	Cost	Confidentiality	Integrity	Non-repudiation	Availability	Authenticate	Broadcast Interaction	Self-Heal	Prone to QC attack
SKE	High	Yes	No	No	No	No	No	No	Yes
SKMA	Low	Yes	No	No	No	No	No	No	Yes
LKH	High	Yes	No	No	No	No	Yes	No	Yes
ASKMA	Low	Yes	No	No	No	No	Yes	No	Yes
HKMA	Low	Yes	No	No	Yes	No	Yes	No	Yes
AHSKMA	Low	Yes	No	No	Yes	No	Yes	No	Yes
LiSH+	Low	Yes	No	No	Yes	No	Yes	Yes	Yes
ID-based KMP	Low	Yes	No	Yes	No	Yes	Yes	No	Yes
NTRU	Low	Yes	Yes	Yes	No	Yes	Yes	No	No

the algorithm. Therefore, an algorithm with high overhead or cost is less efficient and vice versa.

- Confidentiality is the secured privacy of the data.
- Integrity is when the data remains intact and unmodified.
- Authentication is a security property focusing on verifying and validating the identity of the user in the network.
- Availability is the scenario where the server is always accessible to the client.
- Non-repudiation is when the sender cannot deny that the data has not been sent by him at a particular time.
- Broadcast communication is the one-to-many communication case in a network.
- Self-healing is the case the users of an attacked network can recover their lost session keys to secure the data communication.
- Vulnerability to quantum attack refers to the absence of security measures to protect a system from quantum attack.

VIII. CRITIQUE

We now present a critical analysis of the schemes developed for SCADA network security. The schemes are classified into three categories: current standards, detection, and prevention of SCADA attacks. The paper analyzes the schemes in each category. Moreover, all the schemes are compared with each other. The tables below show the comparison between the protocols.

Table 8 shows that AGA-12 is the best among all the standards providing cryptographic protection to the SCADA systems. However, AES relies on ECDSA, AES, RSA, and SHA which leads to high computational and time cost.

It also does not involve an intrusion detection system and a strong key management protocol.

Table 9 provides the comparison of all the crypto-suite based standards and AGA-12 is by far the best standard. However, unlike IEC 62351, it does not provide defense against DoS attacks. Thus, the scheme has lack of availability property.

In all the standards, the key management protocols and encryption scheme used are weak and vulnerable to quantum attacks.

Table 10 compares all the intrusion detection system proposed for SCADA network security. In this category, OCSVM with K-means emerged as the best detection scheme for SCADA systems using open access networks. However, it is unclear whether it will be efficient when used for closed access networks.

Table 11 compares all the proposed key management protocols for SCADA networks. NTRU is the best scheme among the proposed schemes. It satisfies the main security requirements: confidentiality, integrity, and authentication. The scheme is not yet vulnerable to attacks from quantum computers. However, a quantum computer may able to crack the NTRU algorithm in the future.

IX. CONCLUSION

The paper provides a study of the SCADA communication architecture along with its threats and attacks. It discusses and classifies the frequent attacks on SCADA networks, and potential attack by a quantum computer. Furthermore, it lists all current standards used by organizations standards and provides the security threats of each standard. The two main security threats are lack of defense against Denial of Service attack and, using weak key exchange protocol. To address these two security requirements, researchers offered various novel security schemes. The paper classifies these schemes based on the addressed issue of the current stan-dards. Thus, the schemes are categorized into detection of attacks on SCADA networks, and prevention of the attacks. It further explores and compares all the protocols that fall under each main category. It also addresses the security con-cerns and requirements that a SCADA security scheme needs to approach in future. Thus, the article gives a foundation to provide a course for further researches and assist an organization to decide on a suitable standard and scheme.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable feedback and comments, which have substantially improved the quality of the paper.

REFERENCES

[1] D.-J. Kang, J.-J. Lee, S.-J. Kim, and J.-H. Park, "Analysis on cyber threats to SCADA systems," in *Proc. Transmiss. Distrib. Conf. Expo., Asia–Pacific*, Oct. 2009, pp. 1–4.

[2] *Managing SCADA Complexity-Minimizing Risk: Balancing System Growth Against Destabilizing Uncertainty*, Trihedral Eng. Ltd., Bedford, NS, Canada, 2016.

[3] S. Nazir, S. Patel, and D. Patel, "Autonomic computing meets SCADA security," in *Proc. IEEE 16th Int. Conf. Cogn. Inform. Cogn. Comput. (ICCI CC)*, Jul. 2017, pp. 498–502.

[4] Department of Homeland Security and The Federal Bureau of Investigation, "Russian government cyber activity targeting energy and other critical infrastructure sectors," Tech. Rep. TA18-074A, Mar. 2018, pp. 1–18.

[5] P. Nader, P. Honeine, and P. Beauseroy, "LP-norms in one-class classification for intrusion detection in SCADA systems," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2308–2317, Apr. 2014.

[6] A. Rezai, P. Keshavarzi, and Z. Moravej, "Key management issue in SCADA networks: A review," *Eng. Sci. Technol., Int. J.*, vol. 20, no. 1, pp. 354–363, 2017.

[7] R. J. Robles, M. Balitanas, R. Caytiles, Y. Gelogo, and T.-H. Kim, "Comparison of encryption schemes as used in communication between SCADA components," in *Proc. Int. Conf. Ubiquitous Comput. Multimedia Appl. (UCMA)*, Apr. 2011, pp. 115–118.

[8] R. L. Perez, F. Adamsky, R. Soua, and T. Engel, "Machine learning for reliable network attack detection in SCADA systems," in *Proc. IEEE 17th Int. Conf. Trust, Secur. Privacy Comput. Commun./IEEE 12th Int. Conf. Big Data Sci. Eng.*, Aug. 2018, pp. 633–638.

[9] A. Sajid, H. Abbas, and K. Saleem, "Cloud-assisted IoT-based SCADA systems security: A review of the state of the art and future challenges," *IEEE Access*, vol. 4, pp. 1375–1384, 2016.

[10] M. Endi, Y. Z. Elhalwagy, and A. Hashad, "Three-layer PLC/SCADA system architecture in process automation and data monitoring," in *Proc. 2nd Int. Conf. Comput. Automat. Eng. (ICCAE)*, vol. 2, Feb. 2010, pp. 774–779.

[11] H. Saputra and Z. Zhao, "Long term key management architec-ture for SCADA systems," in *Proc. IEEE 4th World Forum Internet Hings (WF-IoT)*, Feb. 2018, pp. 314–319.

[12] D. Choi, H. Kim, D. Won, and S. Kim, "Advanced key-management architecture for secure SCADA communications," *IEEE Trans. Power Del.*, vol. 24, no. 3, pp. 1154–1163, Jul. 2009.

[13] H. A. Abbas, "Future SCADA challenges and the promising solution: The agent-based SCADA," *Int. J. Crit. Infrastruct.*, vol. 10, nos. 3–4, p. 307, 2014.

[14] *SCADAPack E Security Technical Reference*, Control Microsyst., Ottawa, ON, Canada, 2013.

[15] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of cyber attacks on SCADA systems," in *Proc. 4th Int. Conf. Internet Things Int. Conf. Cyber, Phys. Social Comput. (iThings/CPSCom)*, Oct. 2011, pp. 380–388.

[16] I. Ullah and Q. H. Mahmoud, "A hybrid model for anomaly-based intrusion detection in SCADA networks," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 2160–2167.

[17] E. Irmak and I. Erkek, "An overview of cyber-attack vectors on SCADA systems," in *Proc. 6th Int. Symp. Digit. Forensic Secur.*, Mar. 2018, pp. 1–5.

[18] K. Holl. (2003). SANS Security Essentials GSEC Practical Assignment Version 1.4b OSI Defense in Depth to Increase Application Security. GIAC Certifications. Accessed: Oct. 21, 2018. [Online]. Available: https://www.giac.org/paper/gsec/2868/osi-defense-in-depth-increase-application-security/104841

[19] H. Hilal and A. Nangim, "Network security analysis SCADA system automation on industrial process," in *Proc. Int. Conf. Broadband Commun., Wireless Sensors Powering (BCWSP)*, Nov. 2017, pp. 1–6.

[20] J. Melnick. (2018). Top 10 Most Common Types of Cyber Attacks. Netwrix Blog. Accessed: Apr. 6, 2019. [Online]. Available: https://blog.netwrix.com/2018/05/15/top-10-most-common-types-of-cyber-attacks/

[21] Y. Zhang, Y. Xiang, and L. Wang, "Reliability analysis of power grids with cyber vulnerability in SCADA system," in *Proc. IEEE PES Gen. Meeting|Conf. Expo.*, Jul. 2014, pp. 1–5.

[22] Tutorialspoint. *Ethical Hacking Sniffing Tools*. Accessed: Apr. 6, 2019. [Online]. Available: https://www.tutorialspoint.com/ethical_hacking/ethical_hacking_sniffing_tools.htm

[23] Grey Campus. *Session Hijacking Process|Ethical Hacking*. Accessed: Apr. 6, 2019. [Online]. Available: https://www.greycampus.com/opencampus/ethical-hacking/session-hijacking-process

[24] Hacking Like a Pro. (2015). *11 Offensive Security Tools for SysAdmins*. Accessed: Apr. 6, 2019. [Online]. Available: https://hackinglikeapro.blogspot.com/2015/06/11-offensive-security-tools.html

[25] Kaspersky Lab. *Computer Viruses vs. Network Worms|Kaspersky Lab US*. Accessed: Apr. 6, 2019. [Online]. Available: https://usa.kaspersky.com/resource-center/threats/computer-viruses-vs-worms
[26] M. Rouse. (2018). What is Trojan Horse (Computing)?—Definition From WhatIs.com. TechTarget. Accessed: Apr. 7, 2019. [Online]. Available: https://searchsecurity.techtarget.com/definition/Trojan-horse
[27] R. Kalluri, L. Mahendra, R. K. S. Kumar, and G. L. G. Prasad, "Simulation and impact analysis of denial-of-service attacks on power SCADA," in *Proc. Nat. Power Syst. Conf. (NPSC)*, Dec. 2016, pp. 1–5
[28] A. Sharma. (2017). *Top 10 PowerFull DoS/DDoS Attacking Tools for Linux, Windows & AMP; Android—TheHackerStuff.* Accessed: Apr. 7, 2019. [Online]. Available: https://thehackerstuff.com/top10-powerfull-ddos-tools-linux-windows/
[29] J. Anderson. (2001). *An Analysis of Fragmentation Attacks*. Accessed: Oct. 21, 2018. [Online]. Available: http://www.ouah.org/fragma.html
[30] N. Sayegh, A. Chehab, I. H. Elhajj, and A. Kayssi, "Internal security attacks on SCADA systems," in *Proc. 3rd Int. Conf. Commun. Inf. Technol. (ICCIT)*, Jun. 2013, pp. 22–27.
[31] A. Luis, *Cybersecurity Lexicon*, vol. 158. Berkeley, CA, USA: Springer, 2016, pp. 1–199.
[32] INFOSEC. (2019). *Popular Tools for Brute-force Attacks [Updated for 2019]*. Accessed: Apr. 7, 2019. [Online]. Available: https://resources.infosecinstitute.com/popular-tools-for-brute-force-attacks/#gref
[33] L. Chang. (2017). How Secure is Today's Encryption Against Quantum Computers? BetaNews. Accessed: Oct. 21, 2018. [Online]. Available: https://betanews.com/2017/10/13/current-encryption-vs-quantum-computers/
[34] P. K. Amiri, "Quantum computers," *IEEE Potentials*, vol. 21, no. 5, pp. 6–9, Dec. 2002.
[35] X. Zhang, Z. Y. Dong, Z. Wan, C. Xiao, and F. Luo, "Quantum cryptography based cyber-physical security technology for smart grids," in *Proc. 10th Int. Conf. Adv. Power Syst. Control, Operation Manage. (APSCOM)*, Nov. 2017, pp. 1–6.
[36] S. K. Routray, M. K. Jha, L. Sharma, R. Nyamangoudar, A. Javali, and S. Sarkar, "Quantum cryptography for IoT: APerspective," in *Proc. Int. Conf. IoT Appl. (ICIOT)*, May 2017, pp. 1–4.
[37] N. Kumar and D. Goswami, "Quantum algorithm to solve a maze: Converting the maze problem into a search problem," 2013, *arXiv:1312.4116*. [Online]. Available: https://arxiv.org/abs/1312.4116
[38] S. Castellanos. (Sep. 13, 2017). *Nascent Quantum Computing Poses Threat to Cybersecurity—CIO Journal.—WSJ*. Accessed: Oct. 21, 2018. [Online]. Available: https://blogs.wsj.com/cio/2017/09/13/nascent-quantum-computing-poses-threat-to-cybersecurity/
[39] R. Dennis. Quantum Computers are the Most Powerful Tech Threat to Cryptocurrency. ICO ALERT. Accessed: Oct. 21, 2018. [Online]. Available: https://blog.icoalert.com/quantum-computers-are-the-most-powerful-tech-threat-cryptocurrency-will-face-9b271e76edda
[40] Quantiki. (2015). *Shor's Factoring Algorithm*. Accessed: Nov. 5, 2018. [Online]. Available: https://www.quantiki.org/wiki/shors-factoring-algorithm
[41] S. Blanda. (Apr. 2014). *Shor's Algorithm—Breaking RSA Encryption—AMS Grad Blog*. Accessed: Jul. 3, 2019. [Online]. Available: https://blogs.ams.org/mathgradblog/2014/04/30/shors-algorithm-breaking-rsa-encryption/
[42] B. Lynn. Number Theory—Euclid's Algorithm. Standford University. Accessed: Nov. 5, 2018. [Online]. Available: https://crypto.stanford.edu/pbc/notes/numbertheory/euclid.html
[43] F. X. Lin, "Shor's algorithm and the quantum Fourier transform," McGill Univ., Montreal, QC, Canada, Tech. Rep. 12-13/nt/ Fangxi-Lin.pdf, 2014. [Online]. Available: http://www.math.mcgill.ca/darmon/courses/12-13/nt/projects/Fangxi-Lin.pdf
[44] B. Parvez, J. Ali, U. Ahmed, and M. Farhan, "Framework for implementation of AGA 12 for secured SCADA operation in oil and gas industry," in *Proc. 2nd Int. Conf. Comput. Sustain. Global Develop. (INDIACom)*, Mar. 2015, pp. 1281–1284.
[45] R. E. Carlson, E. D. Jeffery, S. A. Shamsuddin, and R. P. Evans, "A summary of control system security standards activities in the energy sector," Nat. Scada Test Bed, U.S. Dept. Energy Office Electr. Del. Energy Rel., Oct. 2005, pp. 1–13.
[46] *IEEE Guide for Electric Power Substation Physical and Electronic Security*, IEEE Standard 1402-2000, 2000.
[47] T. Carlson, "Information security management: Understanding ISO 17799," Lucent Technol., 2001.
[48] *ISO 15408 Compliance, IS Decisions*. Accessed: Oct. 22, 2018. [Online]. Available: https://www.isdecisions.com/compliance/ISO-15408-compliance.htm
[49] *Security Guideline for the Electricity Sector: Physical Security*, NERC, Atlanta, GA, USA, 2012.
[50] NERC. (2004). *Standard 1300-Cyber Security*. Accessed: Oct. 22, 2018. [Online]. Available: https://www.nerc.com/pa/Stand/CyberSecurityPermanent/Draft_Version_1_Cyber_Security_Standard_1300_091504.pdf
[51] American Petroleum Institute, *Pipeline SCADA Security*, 2nd ed., API Standard 1164, Pipeline SCADA Security, Jun. 2009, pp. 1–22.
[52] R. Schlegel, S. Obermeier, and J. Schneider, "Assessing the security of IEC 62351," in *Proc. 3rd Int. Symp. ICS SCADA Cyber Secur. Res.*, Jan. 2015, pp. 11–19.
[53] A. Williams. (2019). RSA Encryption Cracked Easily (Sometimes). Hackaday. Accessed: Apr. 4, 2019. [Online]. Available: https://hackaday.com/2019/01/16/rsa-encryption-cracked-easily-sometimes/
[54] J. M. Beaver, R. C. Borges-Hink, and M. A. Buckner, "An evaluation of machine learning methods to detect malicious SCADA communications," in *Proc. 12th Int. Conf. Mach. Learn. Appl. (ICMLA)*, vol. 2, Dec. 2013, pp. 54–59, Dec. 2013.
[55] Y. Yang, K. McLaughlin, T. Littler, S. Sezer, and H. F. Wang, "Rule-based intrusion detection system for SCADA networks," in *Proc. 2nd IET Renew. Power Gener. Conf. (RPG)*, 2013, p. 1.05.
[56] S.-J. Kim, B.-H. Kim, S.-S. Yeo, and D.-E. Cho, "Network anomaly detection for M-connected SCADA networks," in *Proc. 8th Int. Conf. Broadband Wireless Comput., Commun. Appl. (BWCCA)*, Oct. 2013, pp. 351–354.
[57] L. A. Maglaras and J. Jiang, "Intrusion detection in SCADA systems using machine learning techniques," in *Proc. Sci. Inf. Conf.*, Aug. 2014, pp. 626–631.
[58] L. A. Maglaras and J. Jiang, "OCSVM model combined with K-means recursive clustering for intrusion detection in SCADA systems," in *Proc. 10th Int. Conf. Heterogeneous Netw. Qual., Rel., Secur. Robustness (QSHINE)*, vol. 1, Aug. 2014, pp. 133–134.
[59] R. Dawson, C. Boyd, E. Dawson, and J. M. G. Nieto, "SKMA: A key management architecture for SCADA systems," in *Proc. Australas. Workshops Grid Comput. E-Res.*, vol. 54, pp. 183–192, 2006.
[60] D. Choi, H. Jeong, D. Won, and S. Kim, "Hybrid key management architecture for robust SCADA systems," *J. Inf. Sci. Eng.*, vol. 29, no. 2, pp. 281–298, 2013.
[61] A. Rezai, P. Keshavarzi, and Z. Moravej, "Advance hybrid key management architecture for SCADA network security," *Secur. Commun. Netw.*, vol. 9, no. 17, pp. 4358–4368, Nov. 2016.
[62] S. Mittra, "Iolus: A framework for scalable secure multicasting," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 4, pp. 277–288, 1997.
[63] R. Jiang, R. Lu, J. Luo, C. Lai, and X. Shen, "Efficient self-healing group key management with dynamic revocation and collusion resistance for SCADA in smart grid," *Secur. Commun. Netw.*, vol. 8, no. 6, pp. 1026–1039, 2014.
[64] Y.-H. Lim, "IKMS—An ID-based key management architecture for SCADA system," in *Proc. 7th Int. Conf. Netw. Comput.*, Sep. 2011, pp. 139–144.
[65] A. P. Premnath, J.-Y. Jo, and Y. Kim, "Application of NTRU cryptographic algorithm for SCADA security," in *Proc. 11th Int. Conf. Inf. Technol., New Gener. (ITNG)*, Apr. 2014, pp. 341–346.
[66] B. Babu, T. Ijyas, M. P., and J. Varghese, "Security issues in SCADA based industrial control systems," in *Proc. 2nd Int. Conf. Anti-Cyber Crimes (ICACC)*, Mar. 2017, pp. 47–51.
[67] G. M. Coates, K. M. Hopkinson, S. R. Graham, and S. H. Kurkowski, "A trust system architecture for SCADA network security," *IEEE Trans. Power Del.*, vol. 25, no. 1, pp. 158–169, Jan. 2010.
[68] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring-based public key cryptosystem," in *Proc. Int. Algorithmic Number Theory Symp.* Berlin, Germany: Springer, Jun. 1998, pp. 267–288.
[69] F. B. Advised and P. C. Bachoc, *Lattice-Based Cryptography*. Bordeaux, France: Univ. of Bordeaux, 2016.
[70] N. Challa and J. Pradhan, "Performance analysis of public key cryptographic systems RSA and NTRU," *Int. J. Comput. Sci. Netw. Secur.*, vol. 7, no. 8, pp. 87–96, 2007.
[71] E. Williams. (2015). Quantum Computing Kills Encryption. Hackaday. Accessed: Oct. 22, 2018. [Online]. Available: https://hackaday.com/2015/09/29/quantum-computing-kills-encryption/

[72] D. J. Bernstain, ''Introduction to post-quantum cryptography,'' in *Post-Quantum Cryptography*, no. 1978. Berlin, Germany: Springer, 2009, pp. 1–14.
[73] D. Kostadinov. (2018). *Key Elements of an Information Security Policy, General Security, Infosec*. Accessed: Apr. 7, 2019. [Online]. Available: https://resources.infosecinstitute.com/key-elements-information-security-policy/#gref
[74] A. Jøsang, B. AlFayyadh, T. Grandison, M. AlZomai, and J. McNamara, ''Security usability principles for vulnerability analysis and risk assessment,'' in *Proc. 23rd Annu. Comput. Secur. Appl. Conf. (ACSAC)*, Dec. 2007, pp. 269–278.
[75] H. Barwick. (2012). Security Threats Explained: Third Party Access—Computerworld. ComputerWorld from IDG. Accessed: Apr. 7, 2019. [Online]. Available: https://www.computerworld.com.au/article/429271/security_threats_explained_third_party_access/

SAGARIKA GHOSH received the B.Tech. degree in information technology from the Maulana Abul Kalam Azad University of Technology, West Bengal, India, in 2015. She is currently pursuing the master's degree in computer science with Dalhousie University, Halifax, NS, Canada, and will be continuing as a Ph.D. candidate with a focus on network security. She has industrial experience in the areas of database management and Web development. Her research interests include the Internet of Things, cryptography, data privacy and security, supervisory control and data acquisition system security, quantum computing, and quantum cryptography.

SRINIVAS SAMPALLI received the B.E. degree from Bangalore University and the Ph.D. degree from the Indian Institute of Science (IISc), Bengaluru, India. He is currently a Professor and a 3M National Teaching Fellow with the Fac-ulty of Computer Science, Dalhousie Univer-sity, Halifax, NS, Canada. He has led numerous industry-driven research projects on the Internet of Things, wireless security, vulnerability analysis, intrusion detection and prevention, and applications of emerging wireless technologies in healthcare. He currently supervises five Ph.D. and ten master's students in his EMerging WIreless Technologies (MYTech) Laboratory and has supervised over 120 graduate students in his career. His primary joy is in inspiring and motivating students with his teaching and research. He received the Dalhousie Faculty of Science Teaching Excellence Award, the Dalhousie Alumni Association Teaching Award, the Association of Atlantic Universities' Distinguished Teacher Award, a teaching award instituted in his name by the students within his faculty, the Atlantic Canada Section IEEE Outstanding Educator Award, and the 3M National Teaching Fellowship, Canada's most prestigious teaching acknowledgement. Since September 2016, he has been holding the honorary position of the Vice President (Canada) of the International Federation of National Teaching Fellows (IFNTF), a consortium of national teaching award winners from around the world.

applied sciences

MDPI

Article

An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme

Sagarika Ghosh [1], Marzia Zaman [2], Gary Sakauye [3] and Srinivas Sampalli [1,*]

[1] Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada; Sagarika.Ghosh@dal.ca
[2] Research and Development, Cistel Technology, Ottawa, ON K2E 7V7, Canada; marzia@cistel.com
[3] Research and Development, Technologie Sanstream, Gatineau, QC J8Y 2V5, Canada; gsakauye@sanstream.ca
* Correspondence: srini@cs.dal.ca

Abstract: The rapid emergence of quantum computing threatens current Supervisory Control and Data Acquisition (SCADA) security standards, mainly, American Gas Association (AGA)-12. Therefore, researchers are developing various security schemes based on either quantum or post-quantum algorithms. However, the efficiency of quantum algorithms impacts the security of the post-quantum digital signature scheme. We propose an intrusion resistant algorithm exploiting and applying quantum principles in the post-quantum signature algorithm. We use the Bennett 1992 (B92) protocol, a quantum key distribution scheme, to obtain the cipher, and the practical Stateless Hash-based Signatures (SPHINCS)-256 protocol to obtain a post-quantum signature. However, instead of Chacha-12, a well-known cryptographically secure pseudo-random number generator, we apply a quantum random number generator to obtain a truly random Hash to Obtain Random Subset (HORS) signature with Tree (HORST) secret key used in SPHINCS-256. We have implemented the design in Python with the Quantum Information Toolkit. We have validated the proposed algorithm using the Probabilistic Model Checking for Performance and Reliability Analysis (PRISM) and Scyther tools. Moreover, the National Institute of Standards and Technology (NIST) statistical tests show that the proposed algorithm key pairs have randomness of 98% and RSA and ECDSA are below 96%.

Keywords: Network security; quantum cryptography; qubits; post-quantum cryptography

•€•‚ƒ„…†
•€•‚ƒ„…

Citation: Ghosh, S.; Zaman, M.; Sakauye, G.; Sampalli, S. An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme. Appl. Sci. **2021**, 11, 2082. https://doi.org/10.3390/app11052082

Academic Editor: Nikola Paunković and André Souto

Received: 2 February 2021
Accepted: 18 February 2021
Published: 26 February 2021

1. Introduction

Many industries and organizations have adopted Supervisory Control and Data Acquisition (SCADA) systems to control and monitor industrial processes such as water management, oil and gas industry, traffic management, and nuclear power plants. The significance of SCADA systems is directly proportional to the required security level to protect them from cyber attacks [1,2]. Existing standards, including the American Gas Association (AGA)-12, used in SCADA systems are vulnerable to traditional and quantum attacks. The primary quantum algorithms to crack traditional cryptographic schemes are Shor's algorithm [3] and Grover's algorithm [4]. Shor's algorithm is an approach to solve the factoring problem by finding the order or period of the particular function in the problem. A classical computer can easily calculate the periods of a function. However, for large numbers, we need a quantum computer using superposition to solve the order-finding problem efficiently. Grover's approach is a quantum search algorithm to find a particular element, given an array of m elements [5,6].

AGA-12 mainly adopts three algorithms; namely, Rivest–Shamir–Adleman (RSA) for key management, Advanced Encryption Standard (AES) for encryption and Elliptic Curve Digital Signature (ECDSA) for digital signature with the Secure Hash Algorithm (SHA-1) hash function [1].

The boom of quantum computing is both beneficial and detrimental to cyber-physical systems. Gidney et al. [7] practically proved that the Shor's algorithm is capable of cracking RSA-2048 within 8 hours with 20 million qubits. However, while a classical search algorithm takes $O(N)$, an exhaustive search by Grover's algorithm takes $O(\sqrt{N})$. Thus, it weakens the key strength of AES-256 to only 64 bits. A hash algorithm should be both preimage and collision-resistant. However, Google has cracked SHA-1 by launching a collision attack [8]. The SHA-2,3 remains safe at present. An attack on SHA-2,3 requires on the order of 2^{256} queries in a non-quantum setting and, 2^{128} queries in a quantum setting [8]. Thus, SHA-2,3 has security level on the order of 2^{256} in classical hardware and 2^{128} in quantum hardware.

Quantum computing endangers stable cyber-physical security, and researchers are developing solutions to mitigate the quantum threat. The solutions involve quantum and post-quantum cryptography to secure cyber-physical systems from future attacks by quantum computing [9]. Quantum cryptography involves quantum key distribution protocols that exploit the laws of quantum physics, superposition and entanglement principle to secure against Shor's algorithm [10,11]. Whereas, post-quantum cryptography uses algorithms based on mathematical operations that are quantum-resistant and can inter-operate with classical network protocols [12].

One of the many post-quantum cryptography areas is the set of hash-based algorithms to generate a digital signature, which is secure against preimage and collision attacks by a quantum computer [13]. A first preimage attack attempts to recover a message m from only the hash value such that $H = hash(m)$. A second preimage attack finds another message n such that $m = n$ and $hash(m) = hash(n)$. A collision attack aims to retrieve two different messages m and n, such that $hash(m) = hash(n)$ [8,14]. SPHINCS-256 is a high-security post-quantum stateless hash-based signature scheme that uses two types of signatures, namely, an extension of Winternitz One-time Signature (WOTS+) and Hash to Obtain Random Subset (HORS) signature with Tree (HORST) for few time signature [13]. SPHINCS-256 uses a pseudo-random number generator based on Chacha-12, for generating the HORST secret-key. In 2008, Bernstein et al. [15] proposed Chacha, a cryptanalysis-resistant stream cipher algorithm that maps an input stream to a novel and irreversible output stream [16]. Chacha-12 is an extension of Chacha [17].

However, Chailloux et al. [18] proposed an efficient collision search algorithm based on quantum logic gates, which reduces the security level of post-quantum from 2^{128} to $2^{119.6}$. Thus, a quantum attack requires reduced number of queries, which is on the order of $2^{119.6}$, to crack a post-quantum algorithm. Therefore, it is crucial to strengthen the security of post-quantum hash algorithms. Moreover, the emergence of quantum computing has facilitated the application of quantum physics in resource-constrained devices. For example, researchers at the University of Bristol have introduced the chip-based Quantum Key Distribution (QKD) in 2017, and the ID Quantique company has developed a true Quantum Random Number Generator (QRNG) Chip and QKD systems as well [19,20].

1.1. Contributions

This paper proposes a novel security scheme for SCADA systems with the following contributions:

- We propose an intrusion immune SCADA framework by incorporating quantum and post-quantum security scheme. We use the B92 protocol to generate a secret quantum key for encryption and SPHINCS-256, a preimage and collision-resistant algorithm to obtain a digital signature with a true random number generator.
- In general, SPHINCS-256 uses a Pseudo-Random Number Generator (PRNG) based on Chacha-12 to generate a secret key of the HORST tree. However, to increase the security of SPHINCS-256, we introduce QRNG to obtain a non-deterministic and truly random HORST secret key.
- In general, SPHINCS-256 sends a message, public key and digital signature over the public channel. In our proposed scheme, we replace the message with the cipher obtained in the encryption phase.

1.2. Outline of the Paper

Section 2 describes the related work to our proposed scheme. Section 3 discusses the research questions for existing SCADA security standards based on current attacks and possible quantum threats. We also discuss the research questions for the post-quantum scheme, namely, SPHINCS-256. Section 4 describes the proposed framework. Sections 5 and 6 present the formal analysis and statistical analysis of the framework, respectively. Section 7 provides conclusion and scope for future work. Table 1 provides a list of abbreviations.

Table 1. List of Abbreviations and Notations.

Abbreviations/Notations	**Definitions**
SCADA	Supervisory control and data acquisition
AGA-12	American Gas Association-12
RSA	Rivest–Shamir–Adleman
AES	Advanced Encryption Standard
ECDSA	Elliptic Curve Digital Signature
SHA	Secure Hash Algorithm
SPHINCS	Stateless Hash-based Signatures
B92	Bennett 1992 protocol
BB84	Bennett and Brassard 1984 protocol
QKD	Quantum Key Distribution
HORS	Hash to Obtain Random Subset
HORST	Hash to Obtain Random Subset signature with Tree
OTS	One-time signature
FTS	Few-Time signature
WOTS+	Winternitz One-time Signature
PRISM	Probabilistic Model Checking for Performance and Reliability Analysis
NIST	National Institute of Standards and Technology
QRNG	Quantum Random Number Generator
PRNG	Pseudo-Random Number Generator
LOTSS	Lamport One-Time Signature Scheme
MSS	Merkle-Signature Scheme
RTU	Remote Terminal Unit
MTU	Master Terminal Unit
QBER	Quantum bit error rate
ECC	Error Correction Code
R-S	Reed-Solomon code
IoT	Internet of Things
w	Parameter used to generate Winternitz One-time Signature
M	Number of messages
X_i	Public key of Merkle-Signature Scheme
Y_i	Private key of Merkle-Signature Scheme
i	Leaf index
K	Number of published HORST secret key elements

Table 1. Cont.

Abbreviations/Notations	Definitions
t	Number of HORST secret key elements
n	Security parameter refers to bit-length
m	Bit-length of message
d	Layers of the hypertree
h	Height of the hypertree
N	Number of qubits

2. Related Work

Cyber-physical systems, including SCADA systems, are threatened by classical and quantum attacks [1]. Researchers and organizations have proposed various standards and algorithms to provide resistance against these threats. We briefly review part of related work from four aspects, namely, current standards for SCADA, quantum key distribution, and post-quantum digital signature.

2.1. Current Standard: AGA-12

The American Gas Association (AGA) has proposed a standard with the primary goal of providing a cost and time-efficient security scheme for SCADA systems. The AGA-12 standard has three main phases, namely, key management, encryption, and digital signature [1,21] that use the following algorithms, respectively.

- RSA with a blue minimum key size of 1024 bits
- AES with a minimum key size of 128 bits
- ECDSA Digital Signature using SHA-1 with a minimum key size of 160 bits.

Gidney et al [7] have hypothetically broken the RSA algorithm using Shor's algorithm and provides estimation of the quantum resources needed. Grover's algorithm has weakened AES. ECDSA algorithm is no longer secure against quantum computing and, SHA-1 fails to provide collision resistance [22,23]. Therefore, AGA-12 is a weak standard to protect SCADA systems from existing attacks as well as possible quantum attacks.

2.2. Quantum Key Distribution

In a traditional key exchange protocol, the key strength depends on the mathematical complexity. Moreover, classical computers operate with 'bits' with a state 0 or 1. Shor's algorithm on quantum hardware has been proven to crack traditional cryptography like RSA. However, we can mitigate attacks based on quantum algorithms, which factorize large numbers, by using quantum key distribution (QKD). Existing QKD protocols exploit quantum physics laws and make it difficult to measure the quantum state of a qubit without disturbing its a state. The quantum computers operate with 'qubits' as the basic unit of information with a state 0 or 1 or a superposed state [9,10,24].

The two QKD protocol categories, based on the principles used, are mainly, Heisenberg's uncertainty principle and quantum entanglement. In any QKD protocol using Heisenberg's uncertainty principle, the protocol prepares a qubit such that it is not possible to copy the state of qubit without disturbing its state. It is based on the No-Cloning theorem [25]. During the transmission from sender to receiver, the disturbance in the qubit state enables the system to detect an eavesdropper. In the entanglement-based QKD protocol, both the sender and receiver use entangled photons to generate the secret key [10,11,26].

In this paper, we review two primary examples of Heisenberg's uncertainty principle-based QKD, namely, BB84 and B92 protocol. The BB84 protocol, proposed by Bennett and Brassard in 1984 [27], involves two parties generating a secret key using two channels, namely, quantum and public channel. The sender polarizes the photon by using one of the two bases, rectilinear and diagonal. The basis refers to the pairs of orthogonal states.

The rectilinear basis generates either of the two polarization states: 0°, 90°. The diagonal basis generates either 45° or 135° [9,11]. When the receiver measures the qubits, there is a 50% probability of using the wrong basis. Further, the probability of getting a correct qubit state using the wrong basis is 50%. Thus, 25% of the qubits measured by the receiver are incorrect [9,24]. In 1992, Bennett proposed a simplified version of the BB84 protocol named B92 [28]. It handles only one of two polarization states for each basis. The rectilinear basis encodes a 0-bit state into a 0° polarized state. The diagonal basis encodes a 1-bit state into a 45° polarized state. In B92, when the receiver selects the wrong basis, it does not measure anything. This condition is called erasure [11].

Both BB84 and B92 algorithms have four main phases, namely, quantum key exchange, key sifting, error calculation, and error correction and privacy amplification. The mechanism in the key sifting phase of B92 differs from that in BB84. In B92, the key sifting is more efficient since a wrong basis does not measure qubits. Unlike BB84, B92 uses only one of two polarization states for each basis, making the protocol more efficient than BB84. In BB84, the wrong basis can be used to measure qubits, and there is a 50% probability of getting the correct qubit and a 50% probability of getting an incorrect qubit. In B92, when the receiver selects the wrong basis, it does not measure anything. This condition is called erasure. Thus, the receiver does not need to worry about reading incorrect bits because of measuring them wrong.

Nurhadi et al. [11] provides a survey on QKD protocols involving BB84 and B92, performed simulations that show B92 has a lower error value than BB84. One hypothetical reason is that the number of attempts to measure the attacker's qubits is low in B92. Another reason is that in the B92 protocol, the threshold for tolerating error can be set to a lower value. The lower the threshold, the more secure the protocol.The QKD protocol focuses on secure key distribution. It mainly focuses on attacks launched by quantum algorithms that factorize more significant numbers. It does not address authentication.

2.2.1. Possible Technical Limitations in QKD Deployment Scenarios

When deploying low-cost QKD; the following limitations can affect the performance of the security framework.

- Key Rate: Achieving a high key rate can be challenging while deploying the security framework in a low-cost hardware setting. For high volumes of network traffic, QKD needs to improve the development of secure key rate generation. One of the approaches is to increase the performance of the detectors. A detector with high efficiency and short resolving time can increase the key rate [29]. Eleni et al. [29]'s research shows that a QKD can be performed on an optical fiber at a data rate of 1Mbps over a distance of 50 km.
- Distance: QKD protocols provided only point-to-point communication between the RTU and MTU using a quantum channel with noise from channel imperfections. The distance between the RTU and MTU can be increased by increasing the length of the quantum channel. However, the distance between the two units is directly proportional to the noise in the channel. Thus, increasing the point-to-point distance in QKD protocols is not practically feasible. In a QKD, it is crucial to reduce the channel noise as much as possible to improve the key rate and to estimate the noise value accurately. Moreover, the low noise of single-photon detectors plays a crucial role in tolerating losses. For example, a superconducting nanowire single-photon detector in a sub-zero temperature can tolerate a loss of 72 dB, which is equivalent to around 360 km of standard single-mode fiber [29].

2.3. Post Quantum Digital Signature Scheme

A classical exhaustive search algorithm costs $O(N)$ and a Grover's search costs $O(\sqrt{N})$. A preimage and collision attack on existing classical hash functions can be launched using Grover's algorithm with quantum settings [30]. Both attacks need 2^{256} queries in non-quantum hardware, and 2^{128} queries in a quantum black box model [22].

Therefore, researchers have proposed various post-quantum digital signatures and hash algorithms for authentication.

There are four families of post-quantum cryptography, namely, lattice-based, multivariate, hash-based, and code-based cryptography. In this paper, we briefly review the following hash signature schemes.

2.3.1. Lamport Signature

Lamport One-Time Signature Scheme (LOTSS) uses any secure cryptographic hash functions to generate a public key to sign a message. The advantage of using any hash function makes LOTSS versatile. However, LOTSS becomes insecure if there is a possibility of hash function exposure. Another disadvantage is the large size of the public and private keys. For example, a message $M = \{0, 1\}^k$, when k ⍰ Z^+, will have 2*k hash values from the hash function of at least 160 bits to provide a security level of minimum $O(2^{80})$. Thus, both private and public key size is of at least 160*2*k = 320*k bits [31].

2.3.2. Winternitz One-Time Signature Scheme

To address the large key size of LOTSS, Winternitz One-time Signature Scheme (WOTS), an optimized version of LOTSS, uses a parameter w. A larger value of w means smaller signature size. However, the time cost of the WOTS scheme increases as w increases. The signature size decreases linearly, and the time cost increases exponentially [31].

2.3.3. Merkle-Signature Scheme

The main disadvantages of the one-time signature schemes (OTS) are (i) large public key size and (ii) the need for a new public key generation for each message. The Merkle-Signature Scheme (MSS) addresses the flaw of OTS by generating one public key to sign more than one message. The number of messages should be a power of 2 such that $M = 2^n$. It generates the public X_i and private keys Y_i such that Y_i is within $1 \leq i \leq 2n$ and hash function is $h_i = H(Y_i)$. It creates a hash tree with leaf level i = 0 and the root level i = n [31].

2.3.4. HORS and HORST

In 2002, Reyzin et al. [32] proposed a few-time signature scheme using hash functions. It uses two parameters $t = 2^\tau$ for τ ⍰ N and k ⍰ N. The hash value of message is m such that $m = k \log t = k\tau$. HORS uses two keys, a secret key consisting of t random values obtained from a pseudorandom number generator. The public key is obtained from the t hashes of the random values in the secret key, and the signature contains k secret key values [33].

In 2015, Bernstein et al. [13] proposed HORS tree (HORST) for the SPHINCS protocol, which we have used in our proposed framework. HORST reduces the public key size and signature of the HORS scheme. However, it increases the runtime of the algorithm. The public key generated in HORST is the root of a binary hash tree of height $\log t$. The leaves of the tree are public-key elements of the root key. The signature generated has k secret values and authentication path for each value [13]. SPHINCS-256, exhibited in our proposed scheme, comprises Merkle tree, WOTS, and HORST signature scheme. Due to the rapid inception of quantum computing, quantum algorithms have the potential to crack long term security protocols, and improving the efficiency of quantum algorithms can crack post-quantum schemes. Therefore, we need to build up a security system that mitigates such threats. Table 2 provides a synthesis of all the research work related to the proposed algorithm.

Table 2. Related algorithms to provide security current and possible attacks.

Authors	Algorithm/ Organization	Overview	Advantages	Disadvantages
American Gas Association	AGA-12	AGA-12 was developed to provide security as well as to save time and computation cost.	It provides confidentiality, integrity and authentication.	It fails to provide availability and is vulnerable against quantum attacks.
Bennett et al. 1984	BB84	It is Heisenberg's uncertainty principle based QKD proposed by Bennett and Brassard in 1984. It uses two polarization states for each basis.	It is resistant against quantum attack, mainly, Shor's algorithm.	There are possible practical challenges, based on distance and key rate, in real time application.
Bennett 1992	B92	In 1992, Bennett proposed a simplified version of the BB84 protocol named B92. It handles only one of two polarization states for each basis.	It is resistant against quantum attack, mainly, Shor's algorithm.	There are possible practical challenges, based on distance and key rate, in real-time application.
Lamport 1992	LOTSS	It uses any secure cryptographic hash functions to generate a public key to sign a message	It is versatile.	Exposure of hash function reduces the security of LOTSS. It generates large key size.
Robert 1979	WOTS	It is an optimized version of LOTSS.	It has smaller signature size.	The signature size decreases linearly, and the time cost increases exponentially
Merkle 1979	MSS	It is a stateful hash-based signature scheme which creates a single binary hash tree to generate signature and key pair, using OTS scheme.	It uses one public key to sign multiple messages.	The signature size and key sizes have been improved later in the extension of MSS.
Leonid et al. 2002	HORS and HORST	HORS is a few-time signature scheme using hash functions. HORST is HORS with a binary hash tree.	Both of them efficiently generates quantum safe signatures. HORST reduces the public key size and signature of the HORS scheme.	When compared to WOTS, the signature size is relatively larger. HORST increases the time cost of the HORS algorithm.
Bernstein et al. 2015	SPHINCS	It is a stateless hash signature scheme using hyper-tree with OTS and FTS.	It efficiently generates quantum safe signatures. It provides 2^{128} security against quantum gates.	Chailloux et al. [18] proposed an efficient algorithm to reduce the security level of post quantum from 2^{128} to $2^{119.6}$. It sends message, public key and digital signature over the public channel
The authors. 2020	The proposed algorithm	It uses an intrusion resistant security framework based on quantum and post-quantum security scheme.	It replaces the message in SPHINCS-256 with cipher. It uses quantum random number generator to obtain truly random HORST secret key.	The computation cost of our proposed SPHINCS-QRNG is higher than that of AGA-12 and the existing SPHINCS-PRNG algorithm

3. Research Questions

We propose a collision-resistant framework for SCADA systems which uses both quantum and post-quantum algorithm. We developed the proposed scheme based on the two sets of research questions.

3.1. Set1

Set 1 focuses on the research problem of AGA-12.

- Does AGA-12 provide confidentiality, integrity and authentication against quantum attacks?
- Does AGA-12 key management provide resistance against Shor's algorithm?
- Will AGA-12 encryption and digital signature provide enough resistance against Grover's algorithm?

3.2. Set2

Set 2 focuses on the research problem of the post-quantum digital signature, namely, SPHINCS-256. In 2017, Chailloux et al. [18] proposed an efficient quantum collision search algorithm and presented its implications on symmetric cryptography. The authors also proved that it reduces the security level of post-quantum algorithms from 2^{128} to $2^{119.6}$. It reckons the following research questions, which we address in our scheme.

- Can we increase the security level of SPHINCS-256?
- Can a post-quantum algorithm use the laws of quantum physics to increase security?

4. Proposed Scheme

To address the research questions in Set 1, we developed an algorithm that includes the following three phases: key generation, encryption and authentication. We have used the B92 protocol for key generation and encryption, and SPHINCS-256 for integrity and authentication. Moreover, to address the research questions in Set 2, we have modified the SPHINCS-256 algorithm. The SPHINCS-256 algorithm has four main components, namely, PRNG based on Chacha-12 algorithm for HORST secret-key generation and expansion, hashing in WOTS and HORS public-key generation, hashing in trees, Chacha Permutation used in trees. Furthermore, the sender transmits a combination of signature, message, and SPHINCS-256 public key to the receiver. In our proposed algorithm, we use the quantum random number generator (QRNG), instead of pseudo random number generator (PRNG), to generate a truly random HORST secret key. Instead of the message, the sender sends the cipher along with signature and public key to the receiver. Figure 1 provides a brief overview of the steps in our proposed scheme.

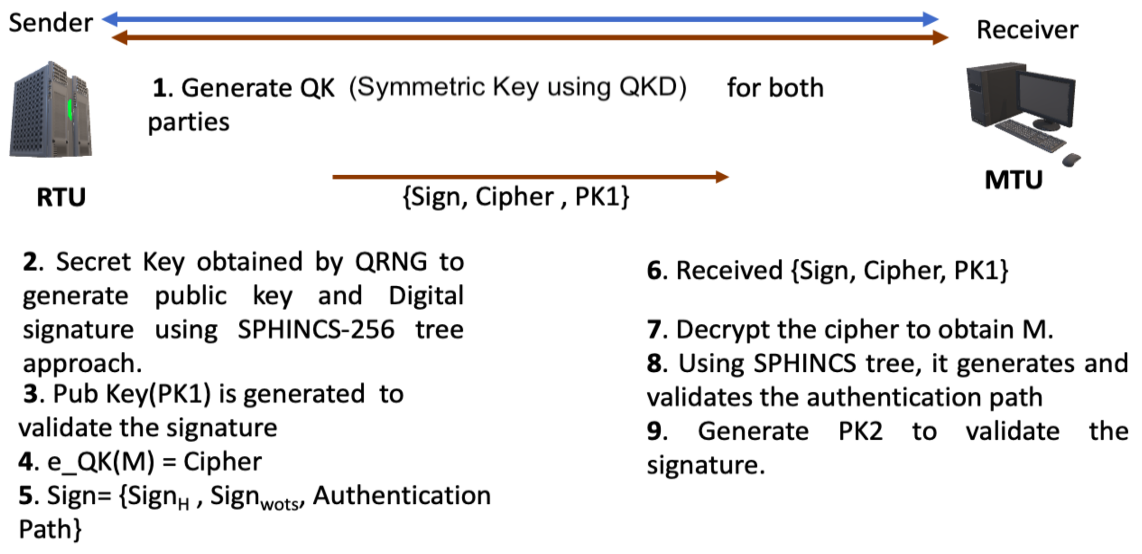

Sender
Receiver
1. Generate QK (Symmetric Key using QKD) for both parties
RTU
MTU
{Sign, Cipher , PK1}
2. Secret Key obtained by QRNG to generate public key and Digital signature using SPHINCS-256 tree approach.
3. Pub Key(PK1) is generated to validate the signature
4. e_QK(M) = Cipher
5. Sign= {Sign$_H$, Sign$_{wots}$, Authentication Path}
6. Received {Sign, Cipher, PK1}
7. Decrypt the cipher to obtain M.
8. Using SPHINCS tree, it generates and validates the authentication path
9. Generate PK2 to validate the signature.

Figure 1. The Proposed Scheme Model

In the following, we present an overview of our proposed scheme.

- Key Generation: The quantum key is generated from the B92 protocol.
- Encryption: The quantum key is used to encrypt the message to obtain the cipher.
- Authentication: The SPHINCS-256 protocol is applied to generate a signature and public key for verification and validation.

4.1. Key Generation

This subsection explains the first phase of our proposed algorithm. It explains the mechanism of the B92 protocol to generate a key for encryption. For a better and easier understanding of the B92 protocol, we have included the key sifting phase. However, key sifting of B92 is slightly different from that in BB84.

We assume that the Remote Terminal Unit (RTU) is the sender, and Master Terminal Unit (MTU) is the receiver. The key generation phase exhibits the B92 protocol, which comprises of the following sub-phases to generate a secret key, using quantum and public channel [11,34].

4.1.1. Raw Key Exchange

The RTU generates a string of raw qubits with the help of either of the two basis, rectilinear and diagonal, randomly. The rectilinear basis polarizes the 0-bit into a 0° polarized photon, and the diagonal basis polarizes the 1-bit into a $+45^\circ$ polarized photon.

The RTU sends the sequence of superposed state qubits, called raw qubits, to the MTU. Furthermore, the raw key exchange utilizes the quantum channel, and the other sub-phases use the public channel.

4.1.2. Key Sifting

The MTU has two sets of basis to measure the incoming string of qubits. The MTU has the same pair of basis. However, the analyzer of MTU has polarisation states orthogonal to that of the RTU's. The MTU has a rectilinear basis with polarization state 90° and a diagonal basis with polarization state -45°.

When the MTU can measure the photon with polarisation of 90°, that means the RTU must have sent the qubit with polarization of $+45^\circ$ and hence the state of the qubit is 1. Whereas, when the MTU can detect the photon with polarization -45°, the state of qubit

must be 0. The MTU, measuring with a wrong basis, fails to read the qubit. The MTU sends the string of the wrong basis to RTU over the public channel. The RTU discards the bit corresponding to the MTU's wrong basis and obtains the sifted key.

4.1.3. QBER Calculation

Both RTU and MTU extract a small portion from their sifted key and use it to calculate the error rate. The RTU obtains the quantum bit error rate (QBER) by computing the ratio of the number of errors and the total number of bits in MTU's extracted key. Both the units discard the extracted portion and acquire the sub-sifted key. In our simulation, we set the threshold of noise or QBER up to 25%. If $QBER > 25\%$, both units discard the process, and when the $QBER \leq 25\%$, both units proceed to the next sub-phase.

4.1.4. Error Correction Code and Privacy Amplification

To resolve the errors in the subsifted key, both the units exhibit Reed-Solomon (R-S) code. It is an error reconciliation algorithm that corrects multi-bit error with a low computational overhead [35,36]. The R-S code also evaluates the confidentiality and integrity of the subsifted key. The RTU encodes the key, which involves adding extra parity bits and sends the codeword to the receiver. The MTU receives the codeword, decodes it, and resolves the errors. Meanwhile, the eavesdropper fails to read the subsiftedkey [36,37].

The Reed Solomon(RS) error correction code has two modules: RS encoder and RS decoder. The RS encoder module involves encoding the information with a Generator polynomial to obtain the codeword. Therefore, it provides confidentiality to the information, however not to the codeword. The codeword is sent to the decoder, which applies a decoding algorithm to correct the errors in the received codeword. When both units perform error correction, they obtain the same subsifted key. However, to reduce any information leakage during the R-S code, the MTU and the RTU hash the subsifted key to obtain the final key. This mechanism is called privacy amplification, as it increases the key secrecy.

4.1.5. Classical Channel Authentication in QKD

The primary contribution and focus of our research are to provide message authentication and integrity by using SPHINCS-256 with QRNG, instead of PRNG, in HORST and by considering quantum key derived from QKD.

Our proposed algorithm has the basic steps which are predefined in QKD, namely, Raw key generation (via Quantum channel), Key Sifting, Error Correction Code (ECC), and Privacy Amplification (all three sub-phases via classical channel). The basic steps ensure the key distribution of the quantum key. However, the classical channel in QKD must be authenticated.

To address the above concern, we have introduced SPHINCS-256 in QKD to authenticate the classical channel. Figures 2 and 3 provides an overall structure of our proposed algorithm. Figure 2 illustrates all the sub phases of QKD. After Raw Key Generation, the key sifting and QBER involves sending a series of wrong basis and extracted portion of sifted key from the receiver to the sender over classical channel. During QBER and error correction code phases, the sender sends the codeword over the classical channel. Each information exchanged over the classical channel during QKD is fed to SPHINCS-256 for message authentication.

Our proposed algorithm includes symmetric key generation using quantum key distribution as opposed to relying on trusted third party or a pre-shared secret key scheme. The QKD, itself, does not rely on a pre-shared secret key scheme or a trusted third party. Moreover, the QKD uses a quantum-resistant post-quantum signature scheme during the classical communication phase. Thus, the proposed integration of QKD and SPHINCS-256, described in Figures 2 and 3, provides intrusion detection and security against both quantum and classical attacks, unlike other traditional algorithms. Moreover, for our future work, we will be performing comparative analysis of QKD authenticated by SPHINCS-256 and QKD authenticated by other universal hash algorithms.

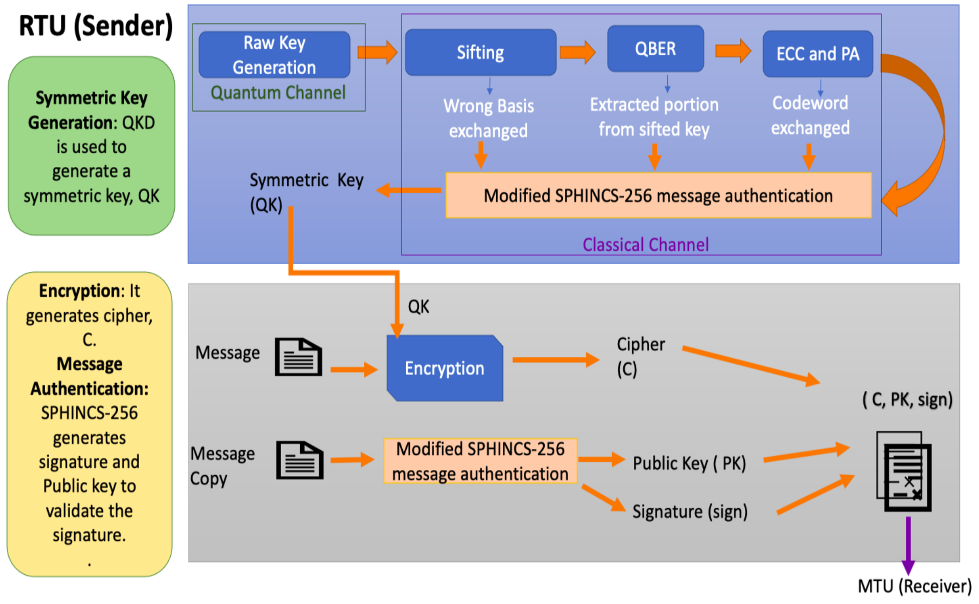

RTU (Sender)
Symmetric Key Generation: QKD is used to generate a symmetric key, QK
Encryption: It generates cipher, C.
Message Authentication: SPHINCS-256 generates signature and Public key to validate the signature.
Raw Key Generation
Quantum Channel
Sifting
QBER
ECC and PA
Wrong Basis exchanged
Extracted portion from sifted key
Codeword exchanged
Symmetric Key (QK)
Modified SPHINCS-256 message authentication
Classical Channel
QK
Message
Encryption
Cipher (C)
(C, PK, sign)
Message Copy
Modified SPHINCS-256 message authentication
Public Key (PK)
Signature (sign)
MTU (Receiver)

Figure 2. The structure of our proposed scheme. It includes symmetric key generation, Encryption and message authentication.

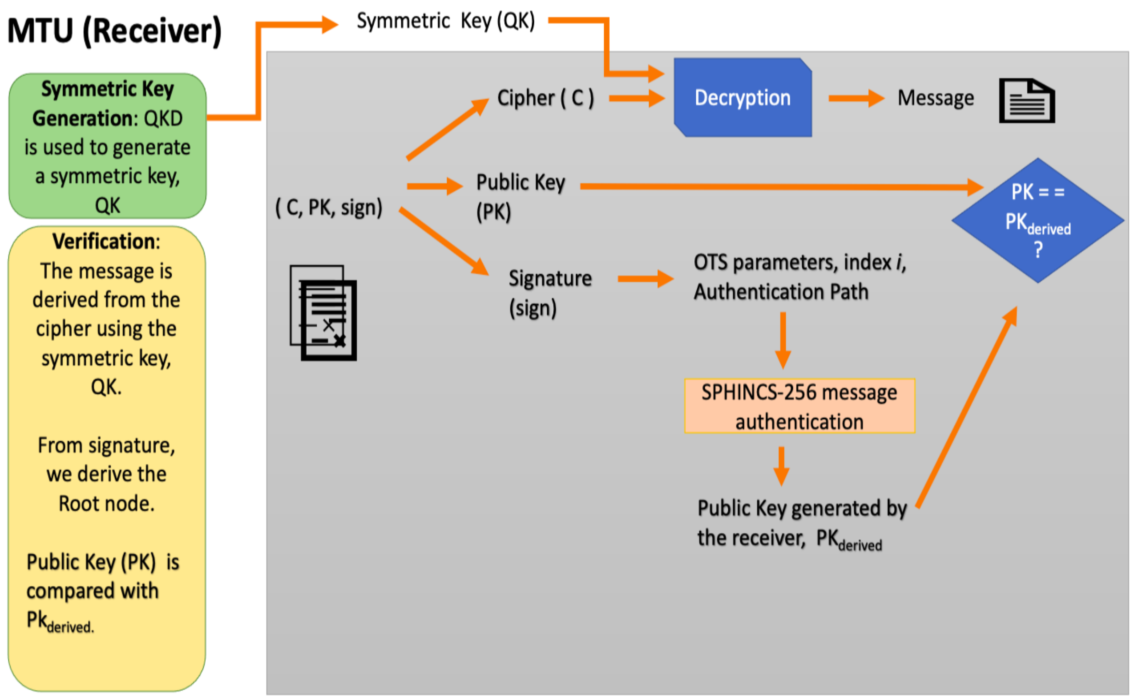

MTU (Receiver)
Symmetric Key Generation: QKD is used to generate a symmetric key, QK
Verification: The message is derived from the cipher using the symmetric key, QK.
From signature, we derive the Root node.
Public Key (PK) is compared with Pk_derived.
Symmetric Key (QK)
Cipher (C)
Decryption
Message
(C, PK, sign)
Public Key (PK)
PK = = PK_derived ?
Signature (sign)
OTS parameters, index i, Authentication Path
SPHINCS-256 message authentication
Public Key generated by the receiver, PK_derived

Figure 3. The structure of our proposed scheme. It includes decryption and signature validation.

4.1.6. Impact of the Use of QKD Technology on the Network Topology

A typical SCADA network topology consist of two primary units: (1) Master Terminal Unit acting as the control center (2) Remote Terminal Unit acting as the field site which gather information from sensors [1,38,39]. Our proposed algorithm mainly focuses on securing communication between the RTU and MTU. The communication link between the MTU and the RTU in the SCADA network topology is a point-to-point link.

When QKD is used on the network topology, the RTU and MTU are connected by two channels: Quantum Channel (fiber optic) or Classical Channel (Internet or fiber optic). If the SCADA network topology is based on hardwired communication, we can say that 2*N fiber optics are needed to deploy QKD [40]. The point-to-point link provides low latency and fast communication. One major constraint of QKD on point-to-point network topology is that if one of the communication links is broken, the entire network fails [39,41]. However, the impact of authenticated QKD in SCADA network also includes resistance against intrusion and quantum attacks.

For future work in practical QKD research, a quantum channel can be designed which is capable of carrying both quantum and classical information. It may lead to reduction of number of fiber optics or, additional communication channel in a network topology.

4.2. Encryption and Digital Signature Algorithm

The procured quantum key encrypts the message to generate cipher. It addresses information confidentiality. We use this generated cipher in the SPHINCS-256 algorithm and send it along with the signature and public key via the public channel. We use the stateless hash signature scheme, namely, SPHINCS, with a hyper-tree of height h and d layers of trees. Each tree uses the Merkle approach and has a height of h/d. Between the trees, it uses Goldreich's construction by applying WOTS as a one-time signature (OTS) scheme [13]. However, to sign the messages, the HORST scheme is used as a few-time signature. Using hyper-tree and Goldreich's construction significantly reduces the total tree height and reduces the signature size. We discuss the various aspects of SPHINCS-256 in the following sections.

4.2.1. Security Parameters of SPHINCS-256

In our proposed scheme, we use SPHINCS-256 to generate digital signature for integrity and authentication. It uses the following security parameters to provide a quality trade-off between speed and signature size [33].

- Security parameter n referring to the bit-length of hashes in HORST and WOTS, $n = 256$.
- Bit length of message, $m = 512$.
- Height of the hypertree, $h = 60$.
- Layers of the hypertree, $d = 12$.
- WOTS parameter, $w = 16$, used for generating One-time signature.
- Number of HORST secret key elements, $t = 2^{16}$.
- Number of published HORST secret key elements, $k = 32$.

4.2.2. Security Level of SPHINCS-256

As mentioned above, Bernstein et al. [13] claim that SPHINCS-256 provides 2^{128} security against the quantum computing attack. However, Chailloux et al. [18] proposed an efficient quantum algorithm that claims to trim the post-quantum security from 2^{128} to $2^{119.6}$. Moreover, Philippe et al. [42] prove a practical attack exploiting the greedy algorithm on the HORS signature scheme. To mitigate such quantum threats, we need to increase the security of the post-quantum SPHINCS-256 scheme. In our proposed algorithm, we infuse the laws of quantum physics in SPHINCS-256, by using a quantum random number generator and a quantum key management algorithm.

4.2.3. SPHINCS-256 Tree Construction and Generating Digital Signature

Figure 4 provides the general construction of SPHINCS tree. Figure 5 provides the mechanism of private and public key generation of both original and modified HORST algorithm. In the following paragraphs, we first explain the construction of SPHINCS tree and then dive into HORST private key generation.

Figure 4 provides a generic construction of SPHINCS tree, with $h = 9$ and $d = 3$. The tree provides a hyper tree construction which includes layers of Merkle's tree with OTS nodes as WOTS (+) scheme. At the bottom of the tree lies the FTS nodes denoted as ◇. The Few Time Signature (FTS) scheme used in SPHINCS is HORST algorithm [43]. The HORST algorithm has two keys, namely, the internal secret key or the private key, and the public key. We have made modification in HORST.

Generally, the HORST algorithm takes as input the seed and the bitmask Q. As shown in Figure 5, the initial secret key SK_H is obtained by feeding seed into a PRNG denoted as G_t. The G_t(seed) generates $SK_H = SK_1, SK_2, \ldots, SK_t$ [13]. However, in our proposed SPHINCS algorithm, the HORST replaces G_t with $QRNG_t$.

QRNG is a type of TRNG deployed in quantum hardware and it does not rely on seeding and a deterministic algorithm since it generates random numbers from measuring or observing quantum processes. A practical QRNG includes a source of randomness and, a measuring or detection system. The source of randomness is the entropy source for generating qubits. For example, an optical QRNG inherits randomness from quantum states of light with the help of a photon source, a polarizing beam splitter and detection systems [44–46].

The $QRNG_t$ generates the truly random secret key $SK_H = SK_1, SK_2, \ldots, SK_t$. It does not need the seed value. The HORST tree is a binary tree constructed using bitmask Q and, the leaves, Li, of the tree is computed using cryptographic hash function F on the elements of the secret key, SKi. Thus, we can denote it as $Li = F(SK_i)$ for i *e* [t-1]. And the root node of the constructed binary tree on the L_i is the public key of HORST.

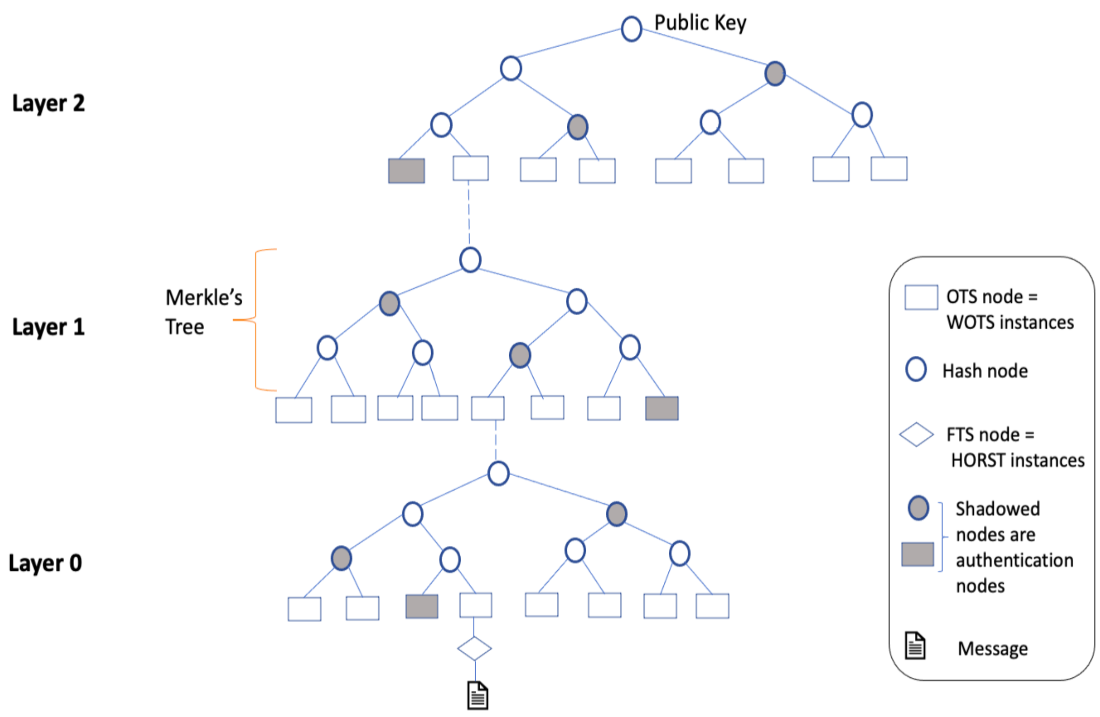

Public Key
Layer 2
Layer 1
Merkle's Tree
Layer 0
OTS node = WOTS instances
Hash node
FTS node = HORST instances
Shadowed nodes are authentication nodes
Message

Figure 4. General Construction of SPHINCS tree with h = 9 and d = 3.

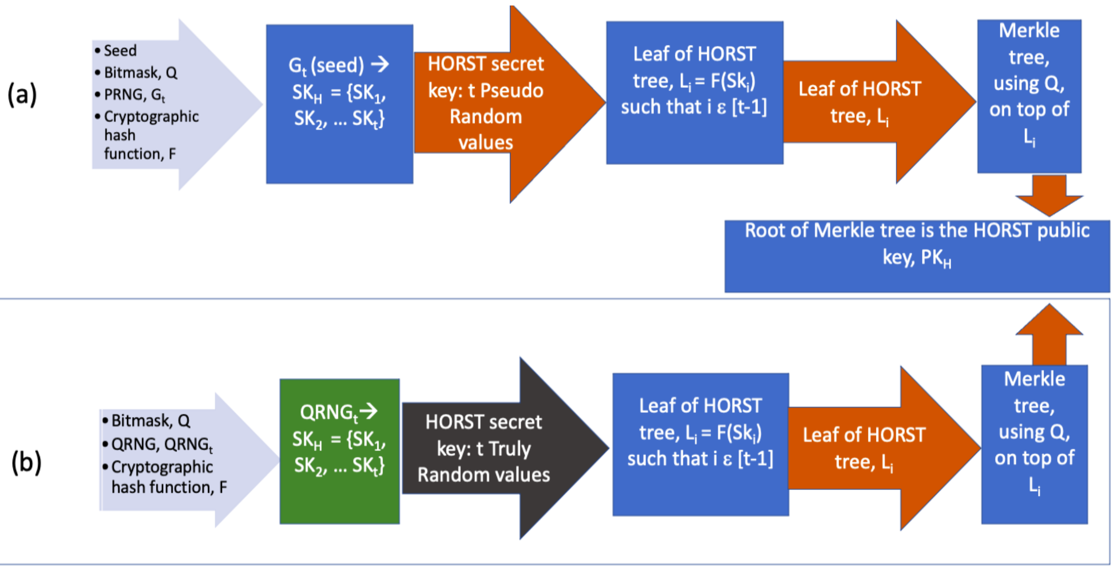

(a)
• Seed
• Bitmask, Q
• PRNG, G_t
• Cryptographic hash function, F
G_t (seed) → SK_H = {SK_1, SK_2, ... SK_t}
HORST secret key: t Pseudo Random values
Leaf of HORST tree, L_i = F(Sk_i) such that i ε [t-1]
Leaf of HORST tree, L_i
Merkle tree, using Q, on top of L_i
Root of Merkle tree is the HORST public key, PK_H
(b)
• Bitmask, Q
• QRNG, $QRNG_t$
• Cryptographic hash function, F
$QRNG_t$ → SK_H = {SK_1, SK_2, ... SK_t}
HORST secret key: t Truly Random values
Leaf of HORST tree, L_i = F(Sk_i) such that i ε [t-1]
Leaf of HORST tree, L_i
Merkle tree, using Q, on top of L_i

Figure 5. Key Generation: (**a**) Original HORST algorithm and (**b**) Proposed HORST algorithm.

SPHINCS-256 tree is a combination of four types of trees: Hyper-tree that includes Merkle's, WOTS, and HORST. The hyper-tree comprises of Merkle's tree connected by WOTS key pairs. Furthermore, the leaves of the SPHINCS-256 are HORS trees. Using the above-mentioned security parameters, the height of each Merkle tree is h/d = 60/12 = 5. The HORST tree follows a Merkle's construction with height $\tau = \log_2 t$ [13].

The SPHINCS-256 follows a stateless Goldreich Signature scheme. Each node of the tree has an OTS pair. The key pair at the root, at the outermost layer (d − 1), has SPHINCS-256 public key, PK_H and private key, SK_{d-1} as shown in Figure 6. However, the message is first signed with the FTS scheme, named HORST, situated is at the bottom of the tree. We used the following steps to generate the signature.

- We obtain the HORST secret key using QRNG, and using the HORS tree algorithm, we obtain its root key, which is the public key. The obtained HORST instances are used to sign the message.
- The obtained HORST signature comprises of k keys, and their respective authentication paths are a part of the SPHINCS-256 signature.
- We then sign the public key of each tree, obtained from the lower layer trees, with WOTS instances as we climb the SPHINCS-256 tree. The signatures obtained in each layer along with its corresponding authentication path to a public key, the root of SPHINCS-256 tree.

After obtaining the signature, the RTU sends the concatenation of SPHINCS-256 public key PK, cipher obtained, and the signature generated by SPHINCS-256 algorithm.

4.2.4. Verification

The MTU receives the package and deciphers the cipher using the same quantum key and, obtains the signature $Sign_h$, $Sign_{wots}$, Authentication Path. The RTU derives the digest of cipher by using the OTS parameters hidden in Sign. Using the SPHINCS tree algorithm, it generates and validates the authentication path. The MTU also obtains its public key, $PK_{derived}$, to validate the signature.

4.3. Relationship between Keys and Their Derivations in Our Proposed Scheme

In the following paragraphs, to address the relationship between keys and their derivations, we explain our proposed main framework, and then explain how the SPHINCS-256 public key is generated and to whom it is related or derived from. We have also added three figures. Figures 2 and 3 provide the structure of our proposed scheme. Figure 4 provides the general construction of a SPHINCS tree. Figure 6 shows the computation of SPHINCS at the topmost layer of the tree.

In our proposed framework, as shown in Figures 2 and 3, we use two primary keys:

1. Symmetric Key, QK, derived from Quantum Key Distribution.
2. A public key, PK, derived from SPHINCS-256 algorithm.

Flow of the proposed algorithm: The symmetric key, QK, finalised from the authenticated QKD, is used to encrypt the message (data gathered by RTU) to generate cipher. The sender copies the message before encryption, and the message copy is fed to the SPHINCS-256 to generate signature and the public key, PK. A tuple containing Cipher, Signature and public key is sent to the receiver. The public key, PK, is used to validate the signature when received by the receiver.

From the above explanation, we can conclude that the public key, PK, of SPHINCS-256 is not derived from the symmetric key, QK. This feature makes the framework secure since the public key does not carry the essence of the symmetric key, QK because QK is used to encrypt the message.

However, the SPHINCS-256 algorithm has key generation phase which generates its own key pair, (Public key, Private key). The Public key is the PK used in our main framework, explained above and in Figures 2 and 3.

Key Generation in SPHINCS-256 [13,43]: The SPHINCS-256 tree is based on hyper tree construction. When the SPHINCS-256 tree is seen as bottom to top approach, the bottom layer of tree has leaf trees as HORST tree with its own leaves. The leaf of the HORST tree is denoted as L_i and the root of the HORST tree is PK_H. The L_i is computed by hashing the elements of HORST secret/private key, SK_H. In Figure 4, we can see that the root node of the below tree is computed to generate the leaf of the following above tree which further generates its own root node. Thus, in simple terms, we can say that the private key of HORST tree is further computed, passing through the main tree via FTS node, OTS nodes and hash nodes, to generate the root of the main tree, SPHINCS-256 public key. Thus, SPHINCS-256 public key, PK, is deterministically derived from HORST private or secret key, SK_H.

In our proposed scheme, only the HORST secret key, SK_H, is generated by QRNG. Thus, the QRNG is used only once in the SPHINCS-256 tree. And, it is used only in the HORST tree.

For a detailed view, Figure 6 shows a SPHINCS tree with d layers, and the root computation of SPHINCS-256 public key, PK, at d-1 layer. The d-1 layer has the secret key, SK_{d-1}, computed and passed on from the below trees. The SK_{d-1} is further hashed to generate seed. The seed along with bitmask is fed to the WOTS scheme to generate public key of WOTS, PK_{WOTS}. The Leaf, $L_{i(d-1)}$ is computed by hashing PK_{WOTS}. And, the $L_{i(d-1)}$ is fed to binary hash tree to obtain the root of the main tree, PK.

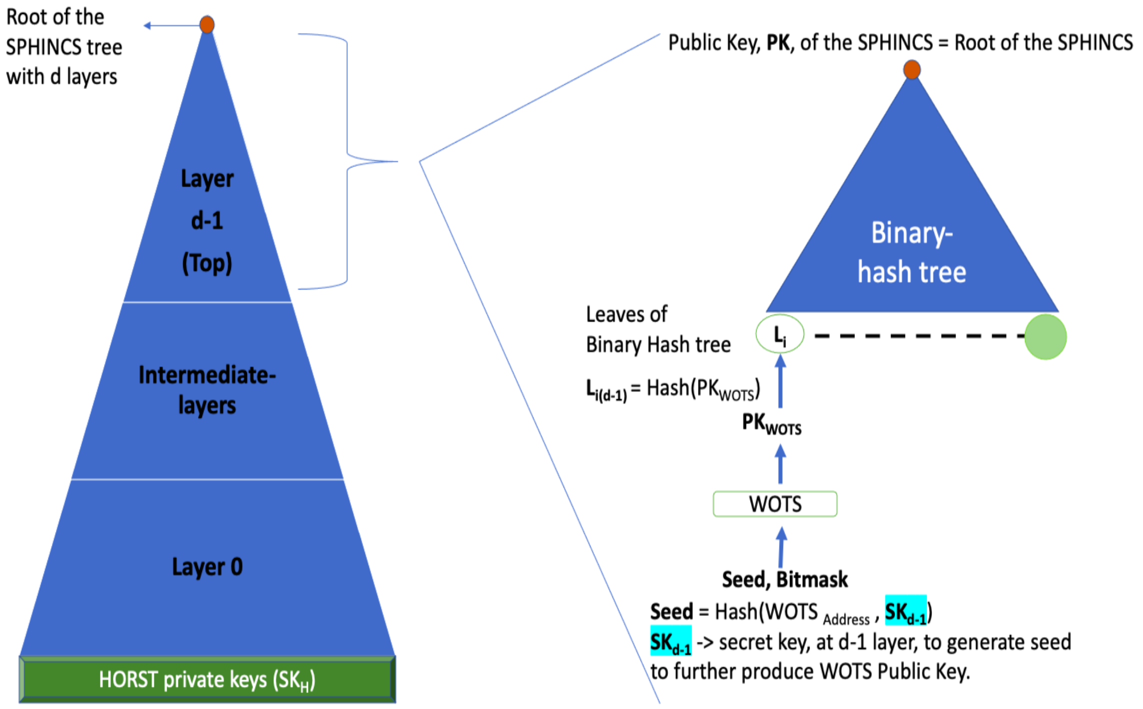

Root of the SPHINCS tree with d layers
Layer d-1 (Top)
Intermediate-layers
Layer 0
HORST private keys (SK_H)
Public Key, PK, of the SPHINCS = Root of the SPHINCS
Binary-hash tree
Leaves of Binary Hash tree
L_i
L_i(d-1) = Hash(PK_WOTS)
PK_WOTS
WOTS
Seed, Bitmask
Seed = Hash(WOTS Address, SK_d-1)
SK_d-1 -> secret key, at d-1 layer, to generate seed to further produce WOTS Public Key.

Figure 6. SPHINCS tree with d layers. It shows the computation at d-1 layer. The d-1 layer is the topmost layer which has key pairs: (1) Public Key, PK. (2) Private or secret key, SK_{d-1}.

5. Formal Analysis of the Proposed Scheme

We have performed a formal analysis of our proposed scheme. We used PRISM [47], a probabilistic model checker, for key generation and Scyther [48] for encryption and digital signature.

5.1. Analysis of Key Generation Phase

We verified the B92 protocol based on the Discrete-Time Markov chain by using the PRISM tool [49]. There are three main steps to build a PRISM model. The first step is system specification, involving building a model of a given system with modules, variables, and constants. We constructed three modules, mainly Alice, Bob, and Eve. The second step is property specification, which addresses the two hypothetical questions: How much information is leaked processing the protocol? Can the B92 protocol discard or prevent the eavesdropping attack? Thus, we created two properties to address the questions, namely, Probability of detecting an eavesdropper (Eve) and, Probability that Eve measures more than half of the information correctly. The third step is feeding the model into the PRISM tool [49]. We created two models [50,51], namely, B92 protocol with Intercept Resend attack, and B92 protocol with Random Substitute attack.

PRISM is a probabilistic model checker that indicates design flaws in the security protocol before moving to the simulation phase and deploying in a real-time hardware setting. Thus, PRISM does not address the challenges during hardware implementation. Based on Sibson et al. [20] 's experiment on the BB84 protocol, the researchers have observed that the secret key rate is 345 kbps with a clock rate of 560 MHz and QBER of 1.05%. Moreover, Rishab et al. [52] 's paper provides both experimental implementations and software simulation of the B92 protocol. The practical implementation shows the key rate and QBER is 51 ± 0.5 Kbits/sec and 4.79% ± 0.01%, respectively. The B92 protocol simulation generates a key rate of 52.83 ± 0.36 Kbits/s and QBER of and 4.79% ± 0.01%.

While conducting formal analysis using PRISM, we considered the worst-case scenario of information leakage. The worst-case scenario is that Eve reads more than half of the qubits over the quantum channel correctly. Because, When Eve measures the qubits, there is a 50% probability of using the wrong basis. Further, the probability of getting a correct qubit state using the wrong basis is 50%. Thus, 25% of the qubits measured by the receiver is incorrect, and 75% of the qubits measured are correct. Therefore, currently, we are considering more than 50% information leakage. Moreover, simulation of B92 in PRISM deals with a low number of qubits due to computational limitations and significantly high elapsed time.

5.1.1. B92 with Intercept-Resend Attack Model

The Intercept-Resend attack model is based on active eavesdropping. Eve tries to read the qubits or bits of information, exchanged through quantum and public channels, during key generation [49]. Figure 7 shows the probability of detecting Eve. Figure 8 shows the probability of detecting information leakage more than N/2, while N is the number of qubits.

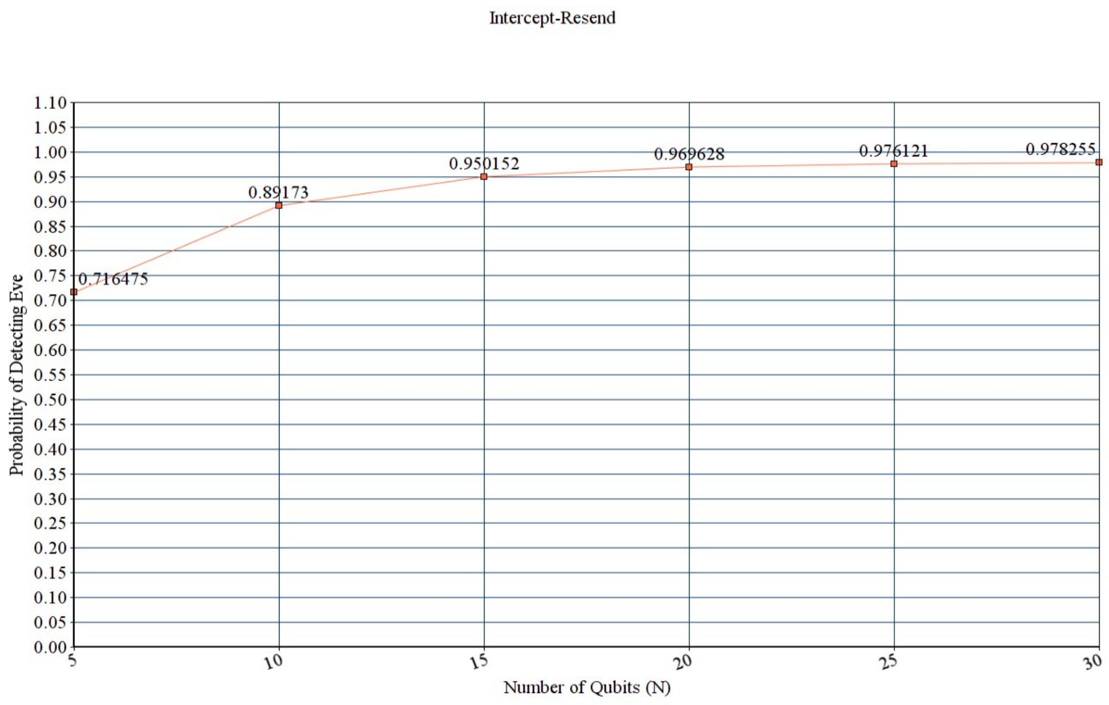


Figure 7. Probability of detecting Eve during Intercept Resend Attack.

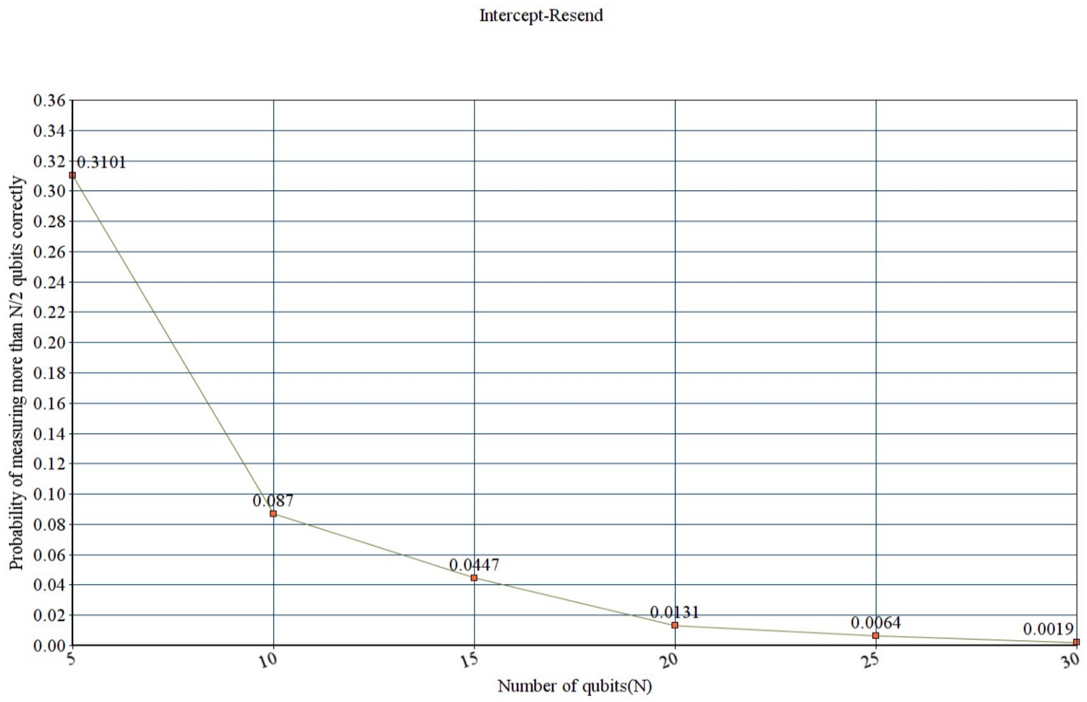


Figure 8. Probability that EVE measures more than half of the information correctly during Intercept Resend Attack.

5.1.2. B92 with Random-Substitute Attack Model

A Random-Substitute attack is a cloning attack. The eavesdropper measures the qubits, which disturbs the state of the qubit. The eavesdropper attempts to duplicate the original state of the qubit and sends it to Bob. Figure 9 shows the probability of detecting Eve. Figure 10 shows the probability of detecting information leakage more than N/2 by Eve, where N is the number of qubits and N ⍰ Z⁺.

Therefore, we conclude that the probability that Eve is detected increases exponentially with the number of qubits. Furthermore, the probability that an Eve obtains a correct measurement result for over half the transmissions decreases exponentially with N. We also conclude that an Intercept-Resend attack is more plausible to cloak an Eve's presence than a Random-Substitute attack.

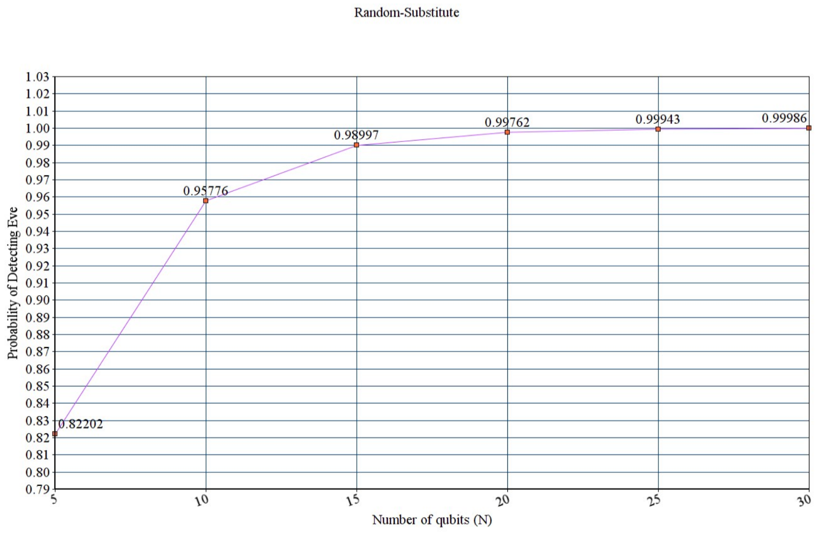


Figure 9. Probability of detecting Eve during Random Substitute Attack.

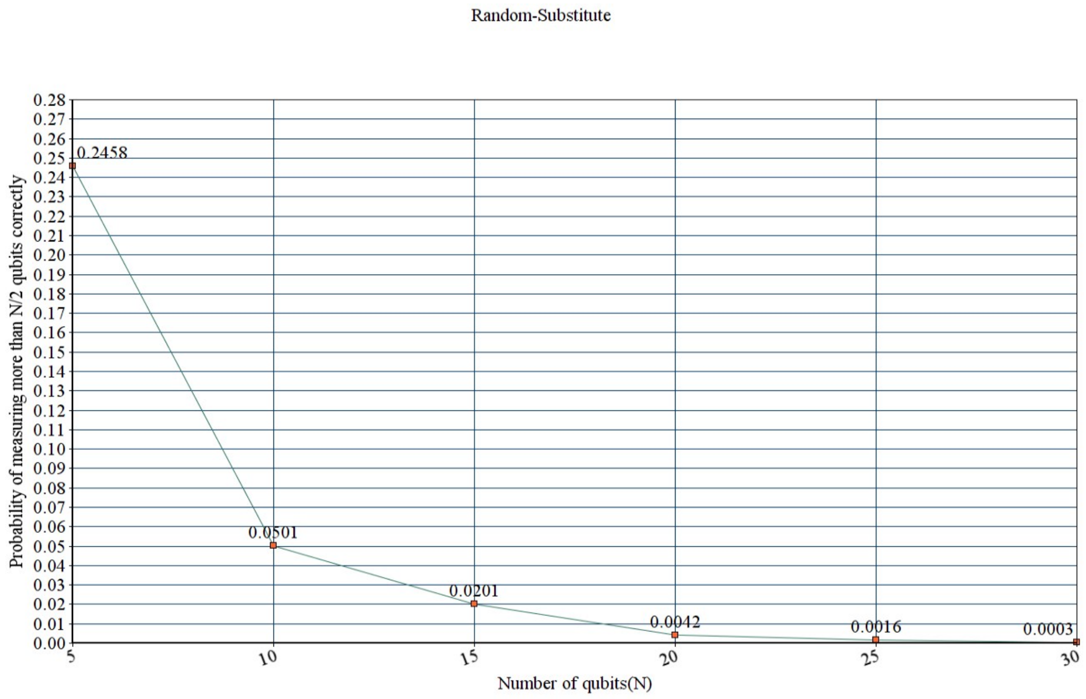
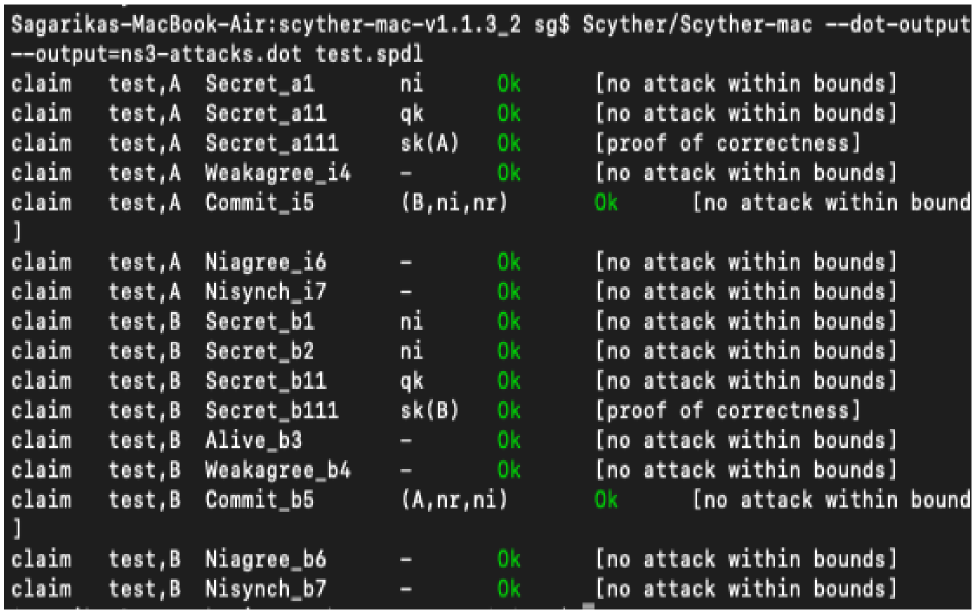


Figure 10. Probability that EVE measures more than half of the information correctly during Random Substitute Attack.

5.2. Analysis of Encryption and Digital Signature Phase

We use Scyther to verify the post-quantum scheme used in our proposed framework. Scyther is an open-source tool used for automatic verification of security protocols [48]. The verification is based on three main aspects: (1) Logical message components verify whether a key is public, secret, constant, or freshly generated in each run. (2) Message structure includes key pairing, encryption, signature, and hash schemes. (3) Message order verifies the synchronization and involvement of agents.

To build a model in Scyther, we use roles and events to send and receive messages between two agents. We have also used claims that refers to the intended security properties. We verified two properties, secrecy and authentication. Secrecy analyzes, whether either of the involved agents, communicates to a trusted party on a dubious channel. Authentication addresses the aliveness of agents, synchronization, the commitment of the protocol and, message agreement between two parties [48]. Figure 11 shows the verification results of the proposed scheme. We conclude that using quantum scheme in the SPHINCS-256 algorithm makes our proposed algorithm resistant to classical and quantum threats.



```
Sagarikas-MacBook-Air:scyther-mac-v1.1.3_2 sg$ Scyther/Scyther-mac --dot-output
--output=ns3-attacks.dot test.spdl
claim   test,A  Secret_a1       ni      Ok      [no attack within bounds]
claim   test,A  Secret_a11      qk      Ok      [no attack within bounds]
claim   test,A  Secret_a111     sk(A)   Ok      [proof of correctness]
claim   test,A  Weakagree_i4    -       Ok      [no attack within bounds]
claim   test,A  Commit_i5       (B,ni,nr)       Ok      [no attack within bound
]
claim   test,A  Niagree_i6      -       Ok      [no attack within bounds]
claim   test,A  Nisynch_i7      -       Ok      [no attack within bounds]
claim   test,B  Secret_b1       ni      Ok      [no attack within bounds]
claim   test,B  Secret_b2       ni      Ok      [no attack within bounds]
claim   test,B  Secret_b11      qk      Ok      [no attack within bounds]
claim   test,B  Secret_b111     sk(B)   Ok      [proof of correctness]
claim   test,B  Alive_b3        -       Ok      [no attack within bounds]
claim   test,B  Weakagree_b4    -       Ok      [no attack within bounds]
claim   test,B  Commit_b5       (A,nr,ni)       Ok      [no attack within bound
]
claim   test,B  Niagree_b6      -       Ok      [no attack within bounds]
claim   test,B  Nisynch_b7      -       Ok      [no attack within bounds]
```

Figure 11. The encryption and signature phase verified by Scyther tool.

6. Results

We have implemented our proposed scheme in Python 3.6. We used the Quantum Information Toolkit [53] to simulate the instances based on quantum physics. Furthermore, to validate our hypothesis, we have generated results and performed a statistical analysis of measured variables. Table 3 provides a list of NIST tests executed on the algorithms used in the current and our proposed scheme. We present the results of our algorithm in the following sections.

Table 3. NIST tests on algorithms.

NIST Tests for Randomness		
RSA and ECDSA (AGA-12)	B92 (in our proposed algorithm)	QRNG and Chacha-12 SPHINCS-256 (in our proposed algorithm)
Frequency Test (Monobit)	Frequency Test (Monobit)	Frequency Test (Monobit)
Frequency Test within a Block	Frequency Test within a Block	Frequency Test within a Block
Run Test	Run Test	Run Test
Longest Run of Ones in a Block	Longest Run of Ones in a Block	Longest Run of Ones in a Block
Discrete Fourier Transform (Spectral) Test	Discrete Fourier Transform (Spectral) Test	Binary Matrix Rank Test
Serial Test 1 and 2	Serial Test 1 and 2	Discrete Fourier Transform (Spectral) Test
Approximate Entropy Test	Approximate Entropy Test	Non-Overlapping Template Matching Test
Cumulative Sums Forward Test	Cumulative Sums Forward Test	Overlapping Template Matching Test
Cumulative Sums Reverse Test	Cumulative Sums Reverse Test	Maurer’s Universal Statistical test
Random Excursions Test		Linear Complexity Test
Random Excursions Variant Test		Serial Test 1 and 2
		Approximate Entropy Test
		Cumulative Sums (Forward) Test
		Cumulative Sums (Reverse) Test
		Random Excursions Test
		Random Excursions Variant Test

6.1. Results Obtained in the Key Generation Phase of the Proposed Scheme

We have implemented our proposed framework in Python using the Quantum Information toolbox. A practical quantum channel consists of noise from the channel imperfections. To simulate such a channel, we implemented the logic of the binary symmetric channel. It sends and receives a message in binary with error probability p [54].

During key generation, we implemented and observed two scenarios, with and without Eve. Figures 12 and 13 show that whenever Eve is present, 70% of the time, QBER > 25. In the remaining 30% of the cases, the QBER was around 20%. In Figure 13, the green bar represents QBER < 25% and, the blue bar represents QBER > 25%. Figure 14 exhibits that out of 10 simulations, the mean execution time of the B92 protocol is 0.0198 s.

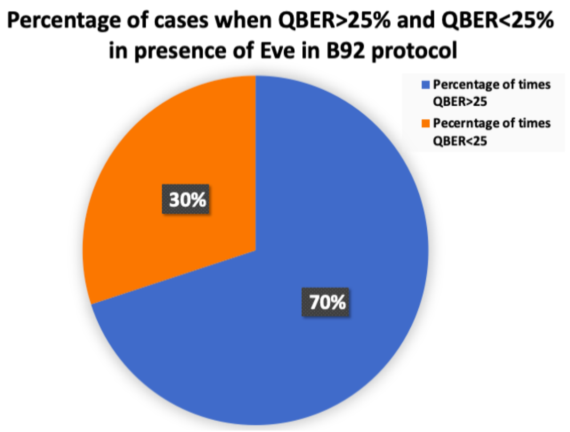


Figure 12. Percentage of cases when QBER is greater and less than 25% in the presence of an Eve in B92 protocol.

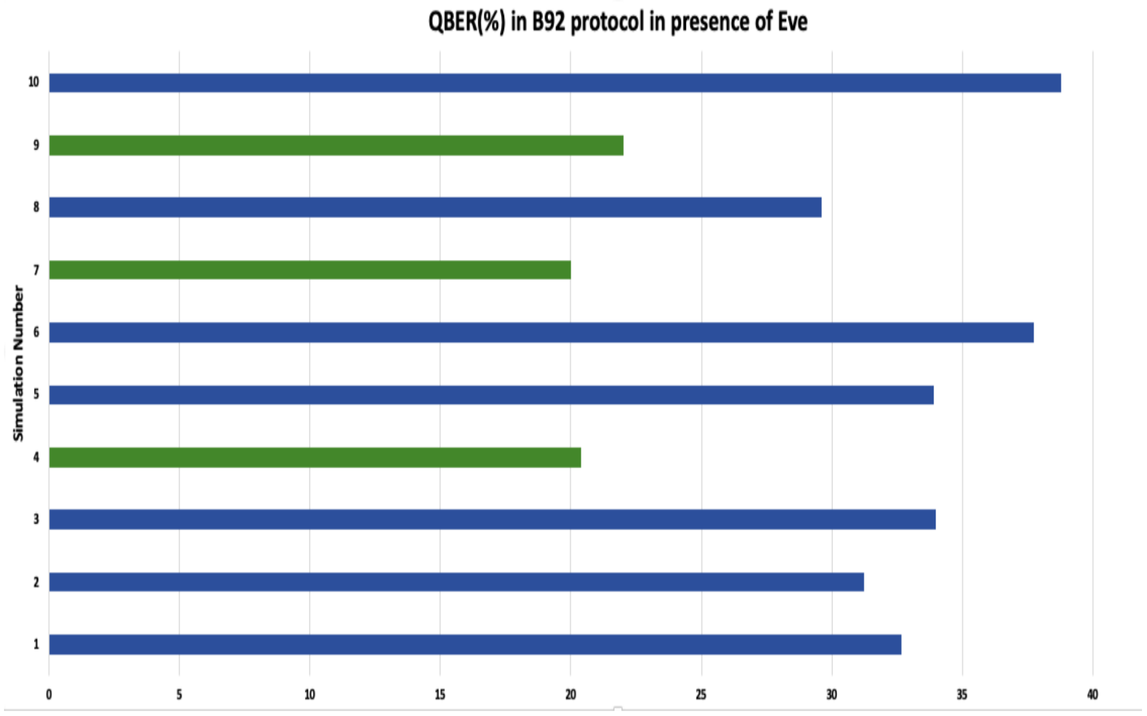


Figure 13. QBER in the presence of an Eve in B92 protocol over 10 simulations.

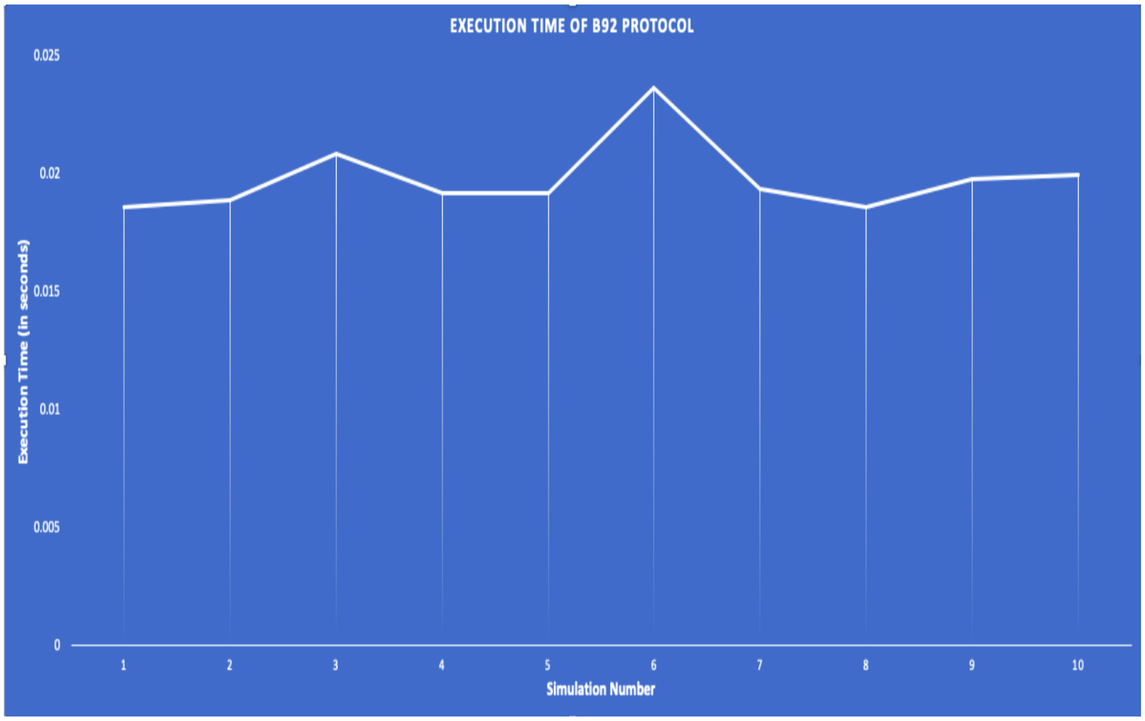


Figure 14. Execution Time of B92 protocol.

To perform a comparative analysis between two scenarios, Eve and without Eve, we used the following variables.

- Sifted key size
- QBER
- Incorrect basis count
- Final Key size

Table 4 depicts that only QBER gets affected by the presence of Eve. Since QBER is used to discard the keys, if QBER>25, the other variables, mainly the final key and sifted key, do not vary much. Furthermore, the incorrect basis count does not show much difference with or without Eve.

Table 4. Comparative Analysis of B92 in presence of Eve vs B92 in absence of Eve.

Variables Measured	Simulation in Presence of Eve					Simulation in Absence of Eve				
	1	2	3	4	5	6	7	8	9	10
Final Key (bits)	63	63	63	63	61	62	62	63	63	60
Incorrect Basis Count (IBC)	252	264	259	260	259	256	260	262	248	268
QBER (%)	21.5	24.4	20	22	24	9.83	8	4	9.61	12.5
Sifted Key (bits)	260	248	253	252	253	256	252	250	264	244

There are 15 tests for randomness in the NIST statistical test suite. All tests are not suited or required for all random number generators as it depends on various factors, mainly, sample size and algorithms used [55,56]. We are using a 512-bit key size for raw key and 62-bit key size for the final key. We followed Doganaksoy et al.'s paper [56] to use the appropriate statistical tests on both raw key and final key. We generated ten final and raw keys for testing. All ten keys passed the tests. Thus, we conclude that both the raw key and final key are random in each simulation. Tables 5 and 6 provide the p-values of the statistical tests on raw key and final key, respectively. Since the p-values are greater than 0.01, the NIST tool concludes the sequences to be random.

Table 5. p-values of appropriate NIST statistical tests on Randomness applied to Raw Key (512 bit) of B92. Conclusion: Random.

p-Values of Raw Keys	Frequency Test	Frequency Test within a Block	Run Test	Longest Run of Ones in a Block	Discrete Fourier Transform (Spectral) Test	Serial Test 1	Serial Test 2	Approx-Imate Entropy Test	Cumulative Sums (Forward) Test	Cummu-Lative Sums (Reverse) Test
1	0.92956	0.65024	0.42611	0.13129	0.62649	0.99402	0.99932	1	0.89202	0.81876
2	0.11161	0.12026	0.54970	0.29407	0.93535	0.49896	0.07918	1	0.22321	0.18615
3	0.79088	0.44540	0.53400	0.02329	0.25614	0.15865	0.14452	1	0.49993	0.73751
4	0.47950	0.32088	0.67429	0.26711	0.46539	0.49896	0.23917	1	0.69601	0.36965
5	0.42632	0.94899	0.36125	0.89453	0.62649	0.49896	0.36062	1	0.57476	0.53660
6	0.72367	0.54742	0.33355	0.20087	0.93535	0.49896	0.36062	1	0.61422	0.92314
7	0.929568	0.66149	0.85941	0.37619	0.62649	0.69077	0.85568	1	0.85688	0.77868
8	0.536101	0.09507	0.98649	0.79974	0.93535	0.69077	0.36062	1	0.89202	0.36965
9	0.376759	0.75873	0.17306	0.73362	0.62649	0.06722	0.63695	1	0.46486	0.65476
10	0.790882	0.19682	0.01713	0.25351	0.93535	0.49896	0.49853	1	0.77868	0.73751

Table 6. p-values of appropriate NIST statistical Tests on Randomness applied to Final Key (mean size = 62 bit) of B92. Conclusion: Random.

p-Values of Final Keys	Frequency Test	Frequency Test within a Block	Run Test	Discrete Fourier Transform (Spectral) Test	Serial Test 1	Serial Test 2	Approximate Entropy Test	Cumulative Sums (Forward) Test	Cumulative Sums (Reverse) Test
1	0.44605	0.44605	0.34151	0.59996	0.49896	0.49853	1	0.25500	0.84788
2	0.52873	0.52873	0.49381	0.93090	0.49896	0.49853	1	0.85301	0.85301
3	0.52873	0.52873	0.06409	0.02604	0.49896	0.49853	1	0.41518	0.85301
4	0.20408	0.20408	0.06066	0.52153	0.49896	0.49853	1	0.19747	0.15080
5	0.10145	0.10145	0.03904	0.28487	0.49896	0.49853	1	0.11756	0.15551
6	0.44605	0.44605	0.55704	0.52153	0.49896	0.49853	1	0.50516	0.50516
7	0.44605	0.44605	0.55704	0.52153	0.49896	0.49853	1	0.50516	0.50516
8	0.52873	0.52873	0.02740	0.28487	0.49896	0.99813	1	0.94315	0.51267
9	0.20408	0.20408	0.04177	0.04177	0.49896	0.49853	1	0.32478	0.32478
10	0.61145	0.61145	0.19079	0.59996	0.49896	0.49853	1	0.73271	0.94027

6.2. Comparative Analysis between SPHINCS-QRNG and Current Algorithms Used in AGA-12

To address the first set of research questions, we implemented and performed a comparative analysis between the Digital signature algorithm used, RSA and ECDSA, in AGA-12 vs. Quantum Key Distribution protocol and SPHINCS-256 with the QRNG algorithm in our proposed scheme. We performed five simulations for RSA and ECDSA and generated five key pairs for each algorithm. We tested the keys by performing the appropriate NIST statistical tests for randomness. Figure 15 shows the observations that they do not satisfy 100% of them.

The 192-bit private key and 384-bit public key of ECDSA passed 96% of the NIST tests over five simulations. RSA private key size with 2048 bit passed 90% of the NIST tests over five simulations, and RSA public passed 98% of them. In contrast, the raw key and final key in QKD passed all the tests (100%) over ten simulations. Moreover, SPHINCS-256, with QRNG key pairs, passed 98% of the tests over five simulations.

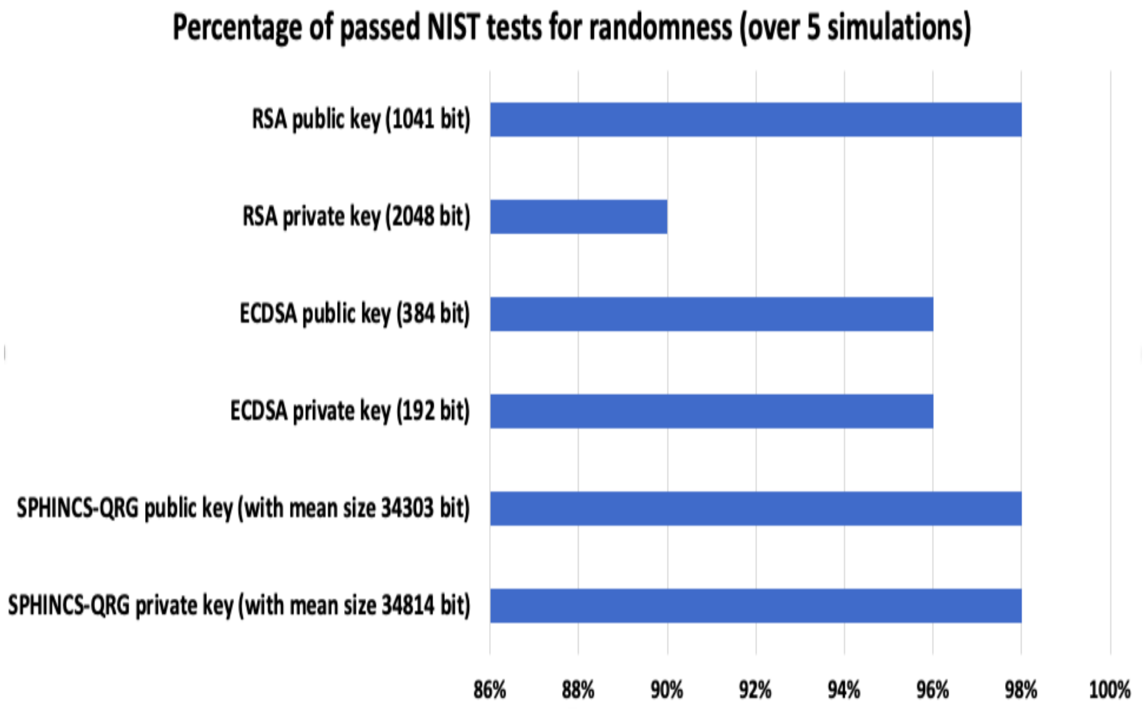


Figure 15. Percentage of passed tests, by RSA, ECDSA and SPHINCS-QRNG key pairs, based on NIST statistical test suite on randomness.

6.3. Results Obtained in the Signature Generation Phase

To address the second set of research questions, we implemented both algorithms: SPHINCS-256 using Chacha12 PRNG and SPHINCS-256 using QRNG to generate HORST secret key. We named the two models SPHINCS-Chacha12 and SPHINCS-QRNG. For comparative analysis, we considered the root of the SPHINCS tree as the public key. And, we considered the private key of the SPHINCS tree at the topmost (d-1) layer. The signature size obtained in both models is 27873 bytes. We generated five random numbers from each algorithm and fed each of them to the NIST statistical tool. We compared both the algorithms by measuring execution time, testing randomness of the generators used in each algorithm, and based on Datcu et al.'s [57] research on testing Chacha12 PRNG.

6.3.1. Comparative Analysis of SPHINCS-QRNG and SPHINCS-Chacha12 Based on Execution Time

Figure 16 shows that the execution time of SPHINCS QRNG is more than that of the existing SPHINCS algorithm with PRNG. The mean execution time of SPHINCS-QRNG public key is 160 µs, and of the private key is 238.89 µs. Whereas, in SPHINCS-Chacha12 algorithm, the average time to generate the public key 112.15 µs and the private key is 110.12 µs.

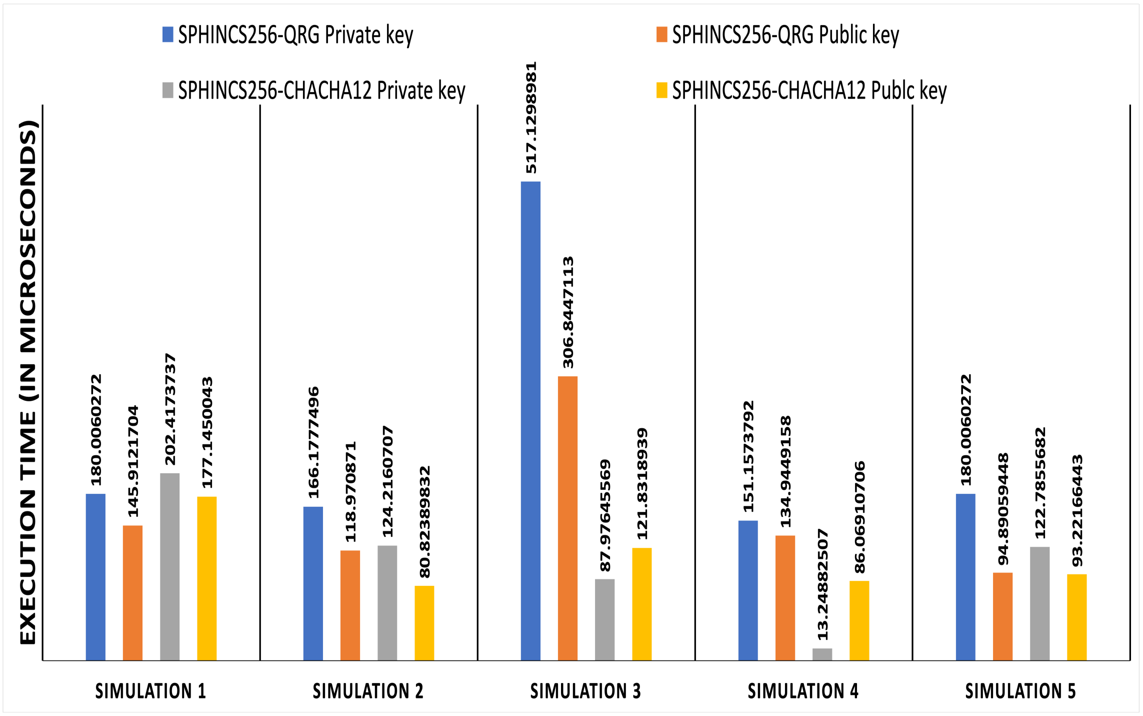


Figure 16. Execution Time for SPHINCS-Chacha12 vs. SPHINCS-QRNG

Moreover, we also performed a comparative analysis based on theoretical performance. Since SPHINCS is a h/d-ary certification tree, Daniel et al. [13] provided a rough theoretical run time values based on the count of pseudo random number functions (PRFs), PRNG and, hashes. The total height of hyper tree is h with d multiple layers of trees. It takes $d2^{h/d}$ OTS key generations, $d2^{h/d} + 2$ PRF calls, $d2^{h/d} + 1$ PRNG calls and $2t + d((l(w+1))2^{h/d} - 1)$ hashes. HORST uses parameter t, such that $t = 2^{\tau}$. The height of the HORST tree is $\tau = \log t$. WOTS uses a signature size and runtime tradeoff parameter w such that w [illegible] N. However, the time complexity of RSA is $O(n^2)$ and that of ECC is $O(n^3)$) [58]. Further, all pseudo-random number generator requires O(n) bit operations. In contrast, QRNG based on Hadamard transformation can be computed in O(nlogn) operations in classical hardware and, in O(1) in quantum hardware [59,60]. Therefore, we conclude that theoretically, the computational complexity of our proposed algorithm is higher than that of existing SPHINCS based on PRNG and AGA-12.

6.3.2. Comparative Analysis of SPHINCS-QRNG and SPHINCS-Chacha12 Based on Randomness

To test the randomness of the quantum random number generator (QRNG), we used all 15 tests of the NIST statistical test suite and, the QRNG passed all of them in every simulation. Thus, we conclude that QRNG is truly random. Table 7 displays the p-values of random number generated by QRNG used in SPHINCS-256. As the p-values $\geq$ 0.01, the NIST tool concludes the sequence to be random with a confidence of 99%. Figures 17 and 18 display the results of Random Excursions Test and Random Excursions Variant Test, respectively for SPHINCS-QRNG. The Random Excursions test executes sub-tests on each of the following states; −4, −3, −2, −1, +1, +2, +3 and +4, to check the frequency of visits to a cumulative sums state within a cycle of a random walk matches with that one would expect for random sequence [55]. For a certain state, if the p-value $\geq$ 0.01, the sequence is random. For example, in Figure 17, the p-value of QRN 1 with state +1 is approximately 0.8492. Thus, the sequence at state +1 is random.

Table 7. p-values of appropriate NIST Statistical Tests for Random and PRNG applied on Quantum Random Number Generator(QRNG) used in SPHINCS-256.

p-Values of QRNG Used in SPHINCS-256	QRN 1	QRN 2	QRN 3	QRN 4	QRN 5
Frequency Test (Monobit)	0.852444761	0.011735483	0.149302374	0.406538784	0.649828827
Frequency Test within a Block	0.161769992	0.733286939	0.575099934	0.665007898	0.518616736
Run Test	0.217961992	0.827155028	0.32712429	0.732869216	0.924945411
Longest Run of Ones in a Block	0.513396784	0.969974739	0.637115311	0.248794857	0.913444629
Binary Matrix Rank Test	0.262333935	0.822162028	0.862431288	0.111413103	0.071873969
Discrete Fourier Transform (Spectral) Test	0.142033423	0.011616891	0.157596656	0.776045999	0.613759295
Non-Overlapping Template Matching Test	0.63600041	0.504047187	0.415637608	0.169519974	0.262129471
Overlapping Template Matching Test	0.200756533	0.505736739	0.353171054	0.591057811	0.508073676
Maurer’s Universal Statistical test	0.124605474	0.243079644	0.01674513	0.73273627	0.889955415
Linear Complexity Test	0.8575379	0.547314553	0.126281648	0.190952376	0.371943788
Serial test 1	0.667804599	0.9238906	0.773287042	0.949470145	0.285436568
Serial test 2	0.691057349	0.953760341	0.7814567	0.920487151	0.576467535
Approximate Entropy Test	0.569869981	0.89057965	0.092335008	0.350697922	0.18177552
Cumulative Sums (Forward) Test	0.800359989	0.013619809	0.215229911	0.508140027	0.69796035
Cumulative Sums (Reverse) Test	0.943118012	0.018863213	0.268772338	0.771928763	0.764505358

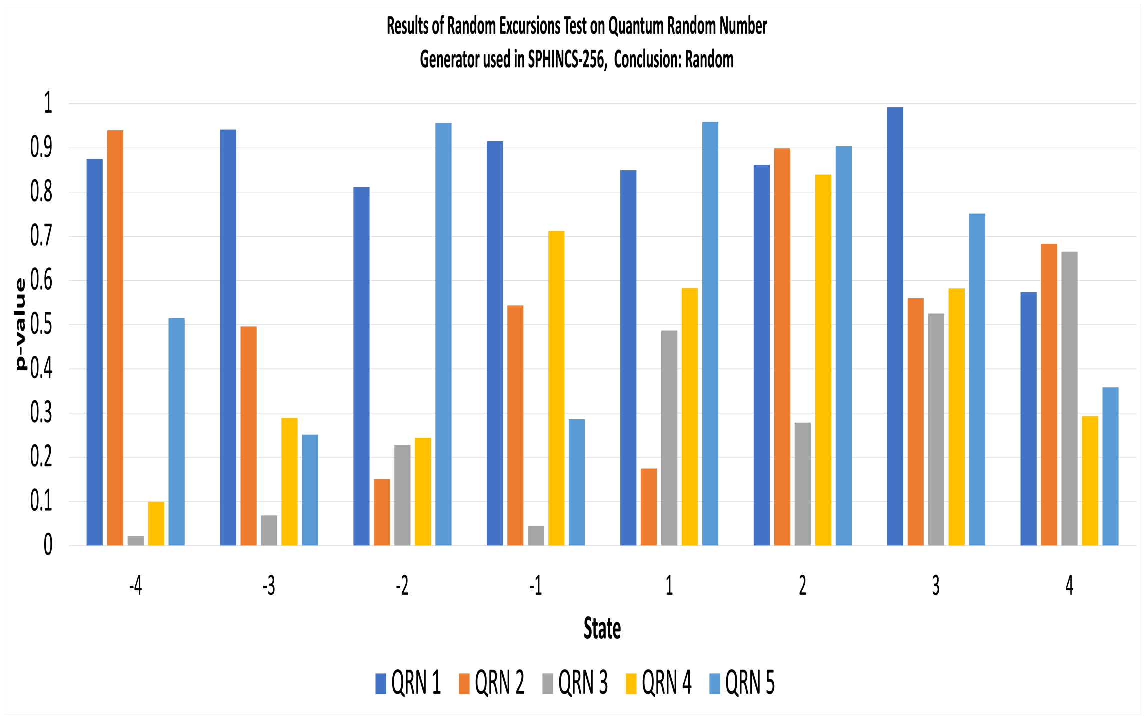


Figure 17. Results of Random Excursions Test on Quantum Random Number Generator(QRNG) used in SPHINCS-256.

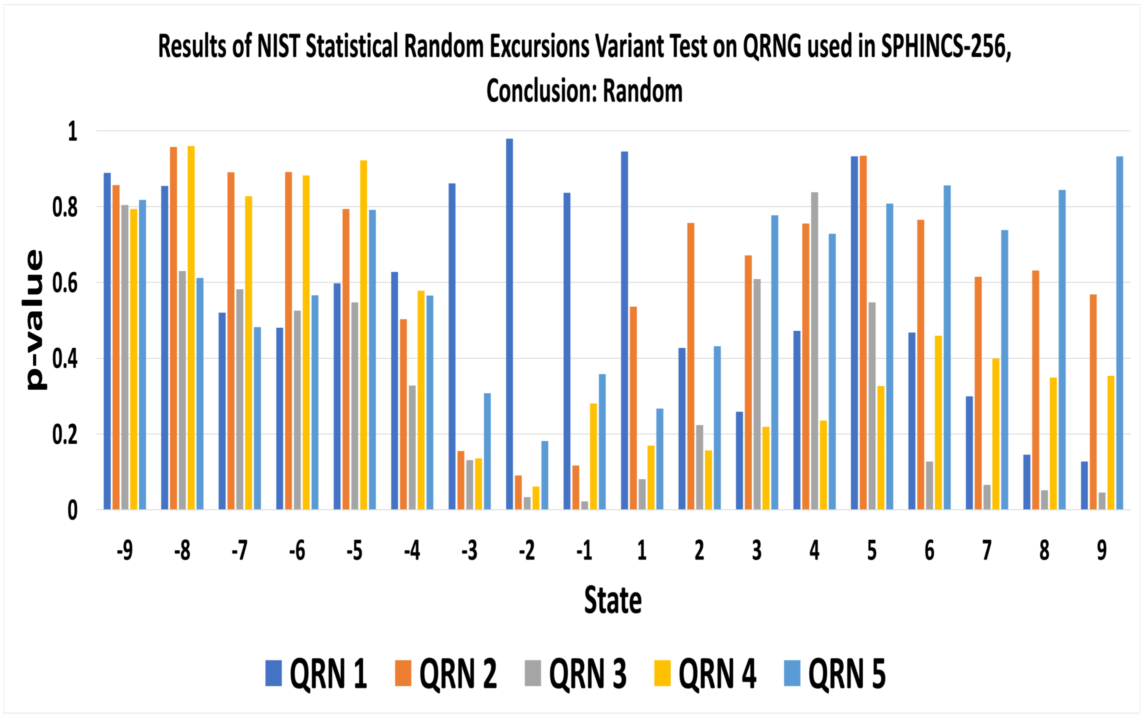


Figure 18. Results of Random Excursions Variant Test on Quantum Random Number Generator(QRNG) used in SPHINCS-256.

The Random Excursions Variant test verifies whether the number of visits to a particular state in a cumulative sum random walk deviates from the expected number of visits in the random walk. It considers 18 states consisting of $\{-9, -8, \ldots, +8, +9\}$ and, if the p-value of a particular state is greater than or equal to 0.01, that means the sequence is random. However, when we ran all 15 tests on a random number generated by SPHINCS-Chacha12, it does not pass 100% of the NIST statistical tests. The pseudo-random number, PRN 5, generated by Chacha-12, failed the Maurer's Universal Statistical test. Also, PRN 1 and PRN 5 does not pass the Random Excursions Test and Random Excursions Variant Test with all states. Table 8, Figures 19 and 20 exhibit the results of the randomness of Chacha-12 PRNG tested by NIST statistical test suite.

We performed 15 statistical analysis on five random numbers for each algorithm. Thus, we performed $15 \times 5 = 75$ tests. Out of 75 tests, Chacha-12 passed 70 tests and, QRNG passed all of them. Figure 21 exhibits a graph that shows Chacha-12 PRNG scored 93.3%, and QRNG scored 100% for successfully passing the tests. The Chacha-12 algorithm is feasible for resource-constrained devices [61]. However, Datcu et al. [57], showed that secure PRNG Chacha does not pass all the statistical tests of Monte-Carlo analysis. Moreover, ID Quantique has developed a QRNG chip for critical infrastructure involving the Internet of Things (IoT) and other resource constrained devices [19]. Table 9 provides a synthesis of results obtained from comparative analysis of the proposed algorithm, RSA, ECDSA and SPHINCS-256.

Table 8. p-values of appropriate NIST Statistical Tests for Random and PRNG applied on Chacha-12 used in SPHINCS-256.

p-Values of PRNG Used in SPHINCS-256	PRN 1	PRN 2	PRN 3	PRN 4	PRN 5
Frequency Test (Monobit)	0.756560956	0.61285665	1	0.87288107	0.76723008
Frequency Test within a Block	0.466408066	0.2519857	0.66537382	0.05620039	0.27892162
Run Test	0.046601502	0.41321088	0.95693516	0.93466707	0.13415598
Longest Run of Ones in a Block	0.39001338	0.57336093	0.31763597	0.60612676	0.43188342
Binary Matrix Rank Test	0.234372205	0.70947779	0.89969577	0.38005323	0.77390385
Discrete Fourier Transform (Spectral) Test	0.520636833	0.87603089	0.52063683	0.21202315	0.4798148
Non-Overlapping Template Matching Test	0.17548433	0.67642991	0.08873813	0.25081909	0.0158475
Overlapping Template Matching Test	0.341609907	0.93826949	0.03571624	0.88583617	0.19903682
Maurer's Universal Statistical test	0.251209205	0.22365261	0.37912508	0.21287032	0.00739755
Linear Complexity Test	0.636767031	0.43256403	0.95718573	0.68049537	0.26611467
Serial test 1	0.266168112	0.04303137	0.85174723	0.77514063	0.06351778
Serial test 2	0.111246791	0.36341311	0.71254001	0.62943052	0.1555089
Approximate Entropy Test	0.991465303	0.50945017	0.72260759	0.55896218	0.70516401
Cumulative Sums (Forward) Test	0.550993298	0.64760991	0.550134	0.6264806	0.39388365
Cumulative Sums (Reverse) Test	0.834146968	0.85546589	0.550134	0.77562923	0.22547471

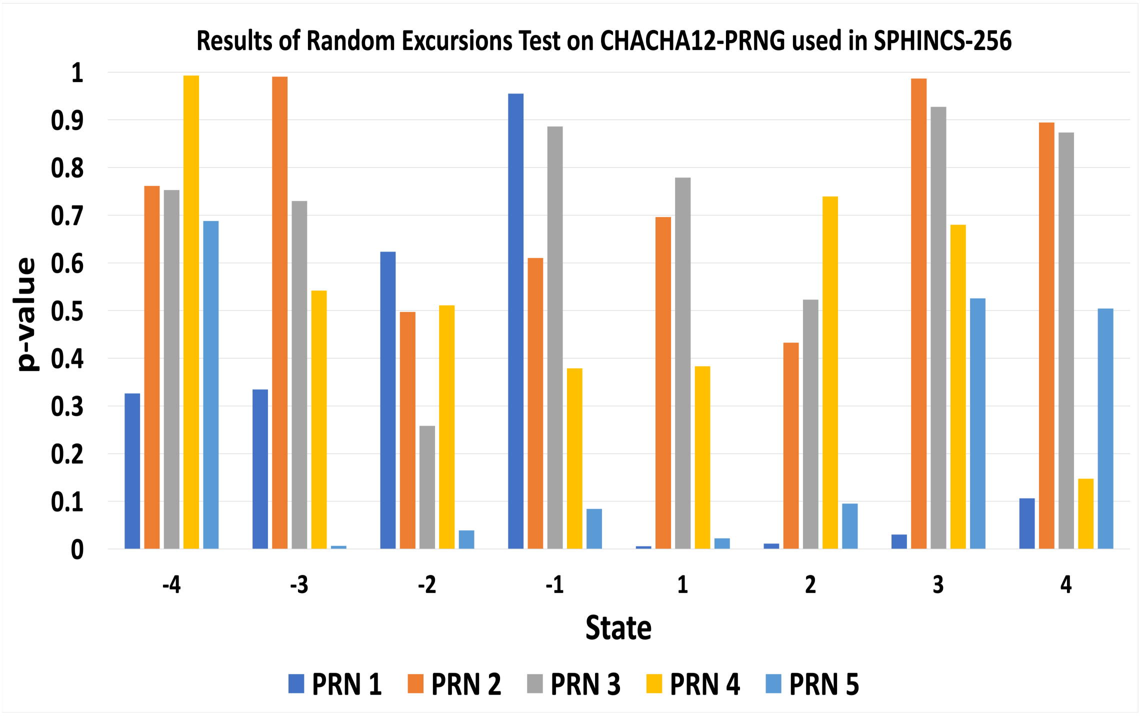


Figure 19. Results of Random Excursions Test on Chacha-12 PRNG used in SPHINCS-256.

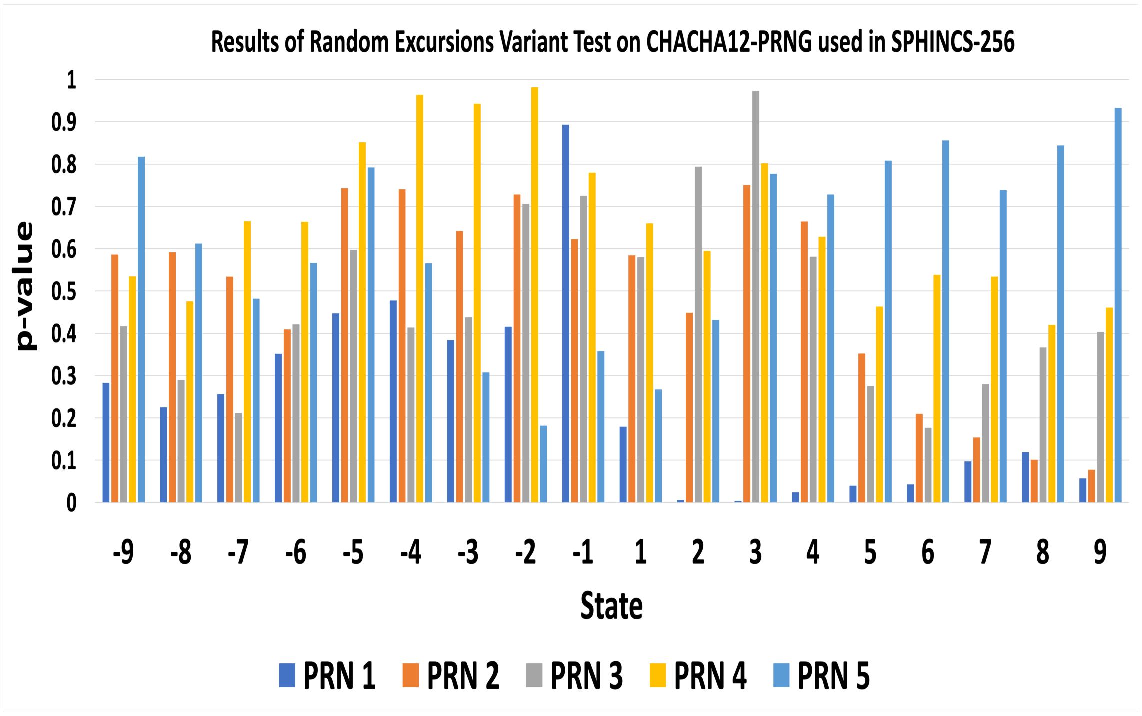


Figure 20. Results of Random Excursions Variant Test on Chacha-12 PRNG used in SPHINCS-256.

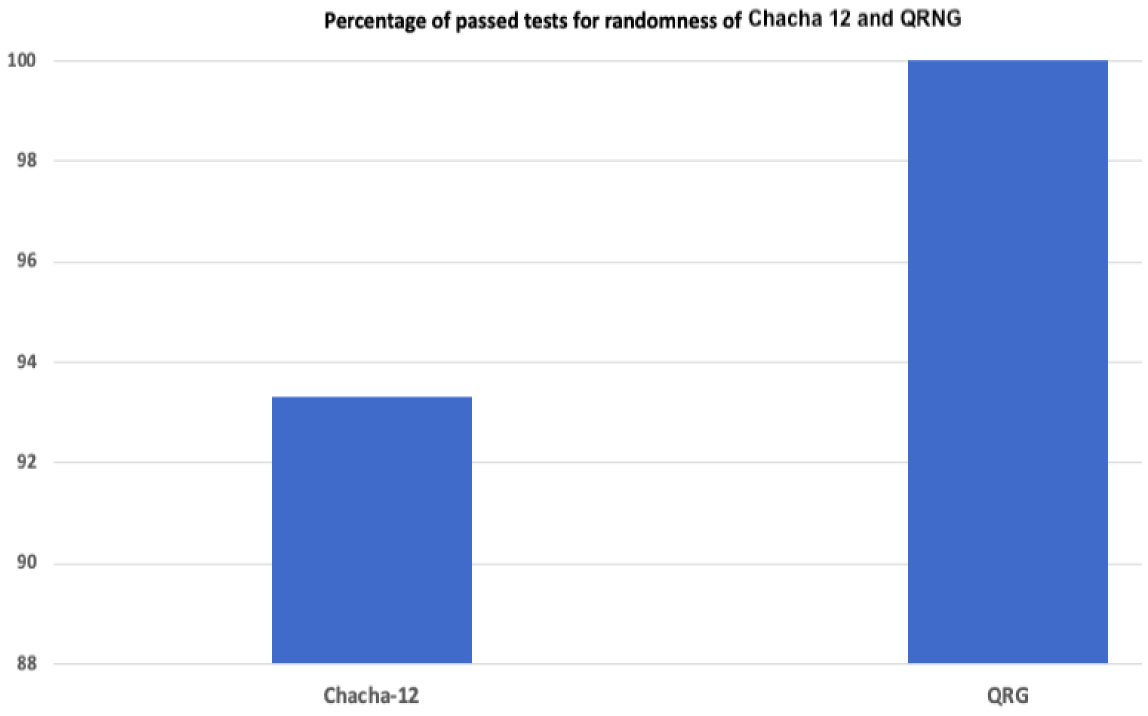


Figure 21. Percentage of passed tests for randomness scored by Chacha-12 and QRNG.

Table 9. Synthesis of comparative analysis of the proposed algorithm, RSA, ECDSA and SPHINCS-256.

Algorithms	**Public Key Size (bits)**	**Private Key Size (bits)**	**Raw key Size (bits)**	**Final Key Size (bits)**	**QBER (mean)**	**Percentage of Passed NIST Randomness Tests**	**Execution Time/ Time Complexity**
RSA	1041	2048	N/A	N/A	No intrusion detection	Private key: 92% Public key: 98%	O (n^2)
ECDSA	384	192	N/A	N/A	No intrusion detection	96%	O (n^3)
The proposed algorithm	34303	34814	512	62	15.584% Detects eavesdropping.	Both keys: 98% QRNG: 100%	$d2^{h/d}$ OTS key generations $d2^{h/d}$ + 2 PRF calls. $d2^{h/d}$ + 1 PRNG calls and $2t + d((l(w+1))\ 2^{h/d} -)1$ hashes QRNG : O(nlogn) in classical hardware. O(1) in quantum hardware. Public Key Generation: 160 µs Private Key Generation: 238.89 µs
SPHINCS-256	34303	34814	N/A	N/A	No intrusion detection	Chacha-12: 93.3%	$d2^{h/d}$ OTS key generations $2d^{h/d}$ + 2 PRF calls. $d2^{h/d}$ + 1 PRNG calls and $2t + d((l(w+1))\ 2^{h/d} -)1$ hashes PRNG : O(n) bit operations. Public Key Generation: 112.5 µs Private Key Generation: 110.12 µs

7. Conclusions

We have proposed a collision and preimage resistant framework for ensuring SCADA system security. SPHINCS-256, a post-quantum algorithm, provides 2^{128} security against a quantum threat. However, researchers have increased the efficiency and speed of quantum algorithm based on Grover's algorithm, which reduces the post-quantum security from 2^{128} to $2^{119.6}$ in a quantum setting. Therefore, we proposed to use B92, a quantum key distribution protocol, to obtain the cipher and a quantum random number generator to generate a truly random number for the HORST secret key used in SPHINCS-256. We have formally verified our proposed scheme by using PRISM and Scyther. We conclude that with the number of qubits, the probability of detecting intruder Eve increases exponentially, decreasing the likelihood of information leakage. Furthermore, B92 is more likely to detect Eve's presence in case of a Random-Substitute attack than an Intercept-Resend attack. We validated our hypothesis by simulating our proposed scheme and performing statistical analysis. We analyzed the randomness of RSA, ECDSA keys used in AGA-12 against the randomness of the quantum key used in our proposed algorithm. We observe that keys generated by QKD satisfy 100% of the NIST randomness tests, unlike RSA and ECDSA keys. RSA private key passed 90%, and ECDSA keys passed 96% of the tests. We also performed a comparative analysis between two algorithms, SPHINCS-256 with Chacha-12 and SPHINCS-256 with QRNG. We observe that the Quantum Random Number

Generator passes all the statistical tests for randomness, unlike Chacha-12 PRNG. However, in computational hardware, the computation cost of our proposed SPHINCS-QRNG is higher than that of AGA-12 and the existing SPHINCS-PRNG algorithm. Thus, we conclude that there is a trade-off between security and computation cost. Our proposed framework, using true random numbers based on uncertainty and quantum superposition principles, provides more resistance than AGA-12 and SPHINCS-256 against quantum and classical threats. Moreover, as part of future work, we will propose a quantum-resistant encryption algorithm and compare it with various encryption algorithms to obtain a more effective cipher for the SCADA security framework.

Author Contributions: S.G.: conceptualization, formal analysis, methodology, writing—first draft; M.Z.: project administration, writing—review and editing; G.S.: project administration; S.S.: funding acquisition, supervision, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the support in part by the Natural Sciences and Engineering Research Council (NSERC) and industry partners Cistel Technology Inc., through a Collaborative Research Grant.

Institutional Review Board Statement: "Not applicable" for studies not involving humans or animals.

Informed Consent Statement: "Not applicable" for studies not involving humans or animals.

Data Availability Statement: Data is available within the manuscript.

Acknowledgments: Thanks to Rohit Joshi of Cistel Technologies for valuable feedback.

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

References

1. Ghosh, S.; Sampalli, S. A survey of security in SCADA networks: Current issues and future challenges. IEEE Access **2019**, 7, 135812–135831. [CrossRef]
2. Kang, D.J.; Lee, J.J.; Kim, S.J.; Park, J.H. Analysis on cyber threats to SCADA systems. In Proceedings of the 2009 Transmission & Distribution Conference & Exposition: Asia and Pacific; Seoul, Korea (South), IEEE: 2009; pp. 1–4.
3. Lomonaco, S. Shor's quantum factoring algorithm. In Proceedings of Symposia in Applied Mathematics, 2002; American Mathematical Society, Providence, Rhode Island(USA); Volume 58, pp. 161–180.
4. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, New York, United States, 1996; pp. 212–219.
5. Yanofsky, N.S.; Mannucci, M.A. Quantum Computing for Computer Scientists; Cambridge University Press: Cambridge, UK, 2008. [CrossRef]
6. Kaye, P.; Laflamme, R.; Mosca, M. An Introduction to Quantum Computing; Oxford University Press: Oxford, UK, 2007.
7. Gidney, C.; Ekerå, M. How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits. arXiv **2019**, arXiv:1905.09749.
8. Wilcox, Z. Lessons From The History Of Attacks On Secure Hash Functions. Available online: https://electriccoin.co/blog/lessons-from-the-history-of-attacks-on-secure-hash-functions/ (accessed on 29 August 2020).
9. Zhang, X.; Dong, Z.Y.; Wang, Z.; Xiao, C.; Luo, F. Quantum Cryptography Based Cyber-Physical Security Technology for Smart Grids. In Proceedings of the 10th International Conference on Advances in Power System Control, Operation & Management (APSCOM 2015), Hong Kong, China, 8–12 November 2015; pp. 1–6. [CrossRef]
10. Padamvathi, V.; Vardhan, B.V.; Krishna, A. Quantum cryptography and quantum key distribution protocols: A survey. In Proceedings of the 2016 IEEE 6th International Conference on Advanced Computing (IACC); Bhimavaram, India, 27–28 February 2016; pp. 556–562.
11. Nurhadi, A.I.; Syambas, N.R. Quantum key distribution (QKD) protocols: A survey. In Proceedings of the 2018 4th International Conference on Wireless and Telematics (ICWT), Bali, Indonesia, 12–13 July 2018; pp. 1–5.
12. Chen, L.; Chen, L.; Jordan, S.; Liu, Y.K.; Moody, D.; Peralta, R.; Perlner, R.; Smith-Tone, D. Report on Post-Quantum Cryptography; US Department of Commerce, National Institute of Standards and Technology: Gaithersburg, MD, USA, 2016; Volume 12.
13. Bernstein, D.J.; Hopwood, D.; Hülsing, A.; Lange, T.; Niederhagen, R.; Papachristodoulou, L.; Schneider, M.; Schwabe, P.; Wilcox-O'Hearn, Z. SPHINCS: Practical stateless hash-based signatures. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015; 2015; pp. 368–397.
14. Sharma, A.; Mittal, S.K. Attacks on Cryptographic Hash Functions and Advances. Int. J. Inf. Comput. Sci. **2018**, 5, 89–96.
15. Bernstein, D.J. ChaCha, a variant of Salsa20. In Workshop Record of SASC; Lausanne, Switzerland, 2008; Volume 8; pp. 3–5.

16. Goll, M.; Gueron, S. Vectorization on ChaCha stream cipher. In Proceedings of the 2014 11th International Conference on Information Technology: New Generations, Las Vegas, NV, USA, 7–9 April 2014; pp. 612–615.
17. Choudhuri, A.R.; Maitra, S. Differential Cryptanalysis of Salsa and ChaCha-An Evaluation with a Hybrid Model. IACR Cryptol. ePrint Arch. **2016**, 2016, 377.
18. Chailloux, A.; Naya-Plasencia, M.; Schrottenloher, A. An efficient quantum collision search algorithm and implications on symmetric cryptography. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security; Springer: Hong Kong, China, 2017; pp. 211–240.
19. Quantis QRNG Chip System-on-Chip for Automotive, Computing, Critical Infrastructure, IoT, Mobile & sEcurity Applications. Available Online: https://www.idquantique.com/random-number-generation/products/quantis-qrng-chip/ (accessed on 30 August 2020).
20. Sibson, P.; Erven, C.; Godfrey, M.; Miki, S.; Yamashita, T.; Fujiwara, M.; Sasaki, M.; Terai, H.; Tanner, M.G.; Natarajan, C.M. Chip-based quantum key distribution. Nat. Commun. **2017**, 8, 1–6. [CrossRef] [PubMed]
21. Parvez, B.; Ali, J.; Ahmed, U.; Farhan, M. Framework for implementation of AGA 12 for secured SCADA operation in Oil and Gas Industry. In Proceedings of the 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 11–13 March 2015; pp. 1281–1284.
22. Amy, M.; Di Matteo, O.; Gheorghiu, V.; Mosca, M.; Parent, A.; Schanck, J. Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3. In Proceedings of the International Conference on Selected Areas in Cryptography; Springer: St. John's, NL, Canada, 2016; pp. 317–337.
23. Mavroeidis, V.; Vishi, K.; Zych, M.D.; Jøsang, A. The impact of quantum computing on present cryptography. arXiv **2018**, arXiv:1804.00200.
24. Routray, S.K.; Jha, M.K.; Sharma, L.; Nyamangoudar, R.; Javali, A.; Sarkar, S. Quantum cryptography for iot: Aperspective. In Proceedings of the 2017 International Conference on IoT and Application (ICIOT); IEEE: Nagapattinam, India, 2017; pp. 1–4.
25. Wootters, W.K.; Zurek, W.H. A single quantum cannot be cloned. Nature **1982**, 299, 802–803. [CrossRef]
26. Muller, A.; Zbinden, H.; Gisin, N. Quantum cryptography over 23 km in installed under-lake telecom fibre. EPL Europhys. Lett. **1996**, 33, 335. [CrossRef]
27. Bennett, C.H.; Brassard, G. Quantum cryptography: Public key distribution and coin tossing. In Proceedings of the International Conference on Computers, Systems and Signal Processing, Bangalore, India, 9–12 December 1984; pp. 175–179.
28. Bennett, C.H. Quantum cryptography using any two nonorthogonal states. Phys. Rev. Lett. **1992**, 68, 3121. [CrossRef] [PubMed] 29. Diamanti, E.; Lo, H.K.; Qi, B.; Yuan, Z. Practical challenges in quantum key distribution. npj Quantum Inf. **2016**, 2, 1–12. [CrossRef] 30. Brassard, G.; Høyer, P.; Tapp, A. Quantum cryptanalysis of hash and claw-free functions. ACM Sigact News **1997**, 28, 14–19. [CrossRef]
31. Becker, G. Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis; Ruhr-University Bochum: Bochum, Germany, 2008.
32. Reyzin, L.; Reyzin, N. Better than BiBa: Short one-time signatures with fast signing and verifying. In Proceedings of the Australasian Conference on Information Security and Privacy; Springer: Berlin/Heidelberg, Germany, 2002; pp. 144–153.
33. Aumasson, J.P.; Endignoux, G. Improving stateless hash-based signatures. In Cryptographers' Track at the RSA Conference; Springer: Cham, Switzerland, 2018; pp. 219–242.
34. Nuhamara, B.R.H.; Syambas, N.R. An Evaluation of Quantum Key Distribution in QuVis Simulation Software. In Proceedings of the 2018 4th International Conference on Wireless and Telematics (ICWT); IEEE: Bali, Indonesia, 2018; pp. 1–4.
35. Ruj, S.; Roy, B. Key predistribution schemes using codes in wireless sensor networks. In Proceedings of the International Conference on Information Security and Cryptology; Springer: Beijing, China,2008; pp. 275–288.
36. Choudhari, S.P.; Chakole, M.B. Reed solomon code for WiMAX network. In Proceedings of the 2017 International Conference on Communication and Signal Processing (ICCSP); IEEE: Chennai, India, 2017; pp. 0176–0179.
37. Riley, M.; Richardson, I. An Introduction to Reed-Solomon Codes: Principles, Architecture and Implementation. Available online: https://www.cs.cmu.edu/~guyb/realworld/reedsolomon/reed$_$solomon$_$codes.html (accessed on 30 August 2020).
38. Upadhyay, Darshana and Sampalli, Srinivas. SCADA (Supervisory Control and Data Acquisition) systems: Vulnerability assessment and security recommendations. Comput. Secur. **2020**, 89, 101666.
39. Stouffer, Keith and Falco, Joseph and Kent, Karen. Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control Systems Security. 2006. Available online: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf (accessed on 30 August 2020).
40. Scalable Quantum Cryptography Network For Protected Automation Communication. Energy.gov.2017. US Department. Available online: https://www.energy.gov/sites/prod/files/2017/05/f34/Qubitekk$_$QKD$_$FactSheet.pdf.2017 (accessed on 30 August 2020).
41. Bailey, David and Wright, Edwin. Practical SCADA for Industry; Elsevier: Amsterdam, The Netherlands, 2003.
42. Aumasson, J.P.; Endignoux, G. Clarifying the subset-resilience problem. IACR Cryptol. Eprint Arch. **2017**, 2017, 909.
43. Bernstein, Daniel J and Hülsing, Andreas and Kölbl, Stefan and Niederhagen, Ruben and Rijneveld, Joost and Schwabe, Peter. The SPHINCS+ signature framework. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, London, UK, 11–15 November 2019; pp. 2129–2146.
44. Herrero-Collantes, Miguel and Garcia-Escartin, Juan Carlos. Quantum random number generators. Rev. Mod. Phys. **2017**, 89, 015004. [CrossRef]

45. Stipcevic, Mario. Quantum random number generators and their applications in cryptography. In Advanced Photon Counting Techniques VI; International Society for Optics and Photonics: Chennai, India, 2012; Volume 8375, p. 837504.
46. ID Quantique. Quantum Versus Classical Random Number Generator. Available online: https://marketing.idquantique.com/acton/attachment/11868/f-64900ef6-6e7e-4b4c-a9f9-c912a2cfde59/1/-/-/-/-/Classical$%$20RNG$%$20Vs$%$20QRNG$_$White$%$20Paper.pdf (accessed on 30 August 2020).
47. Kwiatkowska, M.; Norman, G.; Parker, D. PRISM: Probabilistic symbolic model checker. In Proceedings of the International Conference on Modelling Techniques and Tools for Computer Performance Evaluation; Springer: Berlin/Heidelberg, Germany, 2002; pp. 200–204.
48. Cremers, C. Scyther User Manual; Department of Computer Science, University of Oxford: Oxford, UK, 2014.
49. Papanikolaou, N.K. Techniques for Design and Validation of Quantum Protocols. Ph.D Thesis, Department of Computer Science, University of Warwick, Coventry, UK, 2004.
50. Kuppam, S. Modelling and Analysis of Quantum Key Distribution Protocols, BB84 and B92, in Communicating Quantum Processes (CQP) language and Analysing in PRISM. arXiv **2018**, arXiv:1612.03706.
51. Kuppam, A. Modelling BB84, B92 in CQP and Analysing in PRISM. arXiv **2016**, arXiv:1612.03706v1.
52. Chatterjee, R.; Joarder, K.; Chatterjee, S.; Sanders, B.C.; Sinha, U. qkdSim: An experimenter's simulation toolkit for QKD with imperfections, and its performance analysis with a demonstration of the B92 protocol using heralded photon. arXiv **2019**, arXiv:1912.10061.
53. Bergholm, V.; Biamonte, J.D.; Whitfield, J.D. Quantum Information Toolkit. Available online: http://qit.sourceforge.net (accessed on 30 August 2020).
54. Crépeau, C. Efficient cryptographic protocols based on noisy channels. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques; Springer: Berlin/Heidelberg, Germany, 1997; pp. 306–317.
55. Rukhin, A.; Soto, J.; Nechvatal, J.; Smid, M.; Barker, E. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications; Technical Report; Booz-Allen and Hamilton inc Mclean va: Tysons Corner, VA, USA, 2001.
56. Doganaksoy, A.; Ege, B.; Koçak, O.; Sulak, F. Statistical Analysis of Reduced Round Compression Functions of SHA-3 Second Round Candidates. IACR Cryptol. Eprint Arch. **2010**, 2010, 611.
57. Datcu, O.; Macovei, C.; Hobincu, R. Chaos Based Cryptographic Pseudo-Random Number Generator Template with Dynamic State Change. Appl. Sci. **2020**, 10, 451. [CrossRef]
58. RSA Cryptography Algorithm. Available online: http://algohub.me/algo/rsa-cryptography-algorithm.html (accessed on 30 August 2020).
59. How Fast Does a Pseudorandom Number Generator Have to be in Order to be Competitive? Available online: https://crypto.stackexchange.com/questions/62736/how-fast-does-a-pseudorandom-number-generator-have-to-be-in-order-to-be-competit (accessed on 30 August 2020).
60. Aung, A.; Ng, B.P.; Rahardja, S. Sequency-ordered complex Hadamard transform: Properties, computational complexity and applications. IEEE Trans. Signal Process. **2008**, 56, 3562–3571. [CrossRef]
61. Philip, M.A. A survey on lightweight ciphers for IoT devices. In Proceedings of the 2017 International Conference on Technological Advancements in Power and Energy (TAP Energy); IEEE: Kollam, India, 2017; pp. 1–4.

6 Quantum-Safe Asymmetric Cryptosystems

Current Solutions and Future Directions against Quantum Attacks

Sagarika Ghosh, Marzia Zaman, and Srinivas Sampalli

CONTENTS

DOI: 10.1201/9781003296034-6

6.1 INTRODUCTION

Cryptography is a crucial part of securing all cyber-physical systems to attain confidentiality and integrity as security goals. The security level of cryptographic algorithms, including Rivest–Shamir–Adleman (RSA) and elliptic curve cryptography (ECC), relies on the intractability of certain problems using traditional computers. However, the boom of quantum computing has placed current cryptographic algorithms at stake.

Quantum computers exploit the principles of quantum physics, mainly superposition and entanglement principles, to process information.

The standard quantum algorithms, mainly Shor's [20] and Grover's [16] algorithms, are a threat to RSA and ECC cryptosystems, respectively. Researchers and organizations have been developing various quantum attack-resistant algorithms, using either complex mathematical problems or exploiting quantum physics [23, 24]. The algorithms that exploit quantum physics principles are quantum cryptography, and the algorithms based on hard mathematical problems are post-quantum cryptography. While quantum cryptography relies on quantum hardware, post-quantum cryptography can be deployed on the same hardware infrastructure of the current networks [8]. In this chapter, we review post-quantum cryptography that is resistant to quantum attacks.

6.1.1 Security threat Faced by Quantum Computing

The progress in developing quantum computers on a large scale has threatened the most commonly used symmetric and asymmetric key cryptosystems. A modern computer encodes information in bits with a value of either 0 or 1. However, a quantum computer or hardware encodes information in qubits that follows the rules of quantum physics. The qubit can be in s state 0 or 1 or both at the same time, as per the superposition principle. The two qubits are in an entangled state when their states are allied with each other. The superposition and entanglement principles make quantum computers way faster than any modern computers [14, 34].

6.2 SHOR'S ALGORITHM

In 1994, Peter Shor [30] developed an algorithm to derive the prime factorization of any positive integer. We denote it as *N*. Shor's algorithm has two phases: Classical and quantum. The first section uses the Euclidean algorithm to derive an order-finding problem reduced from the factoring problem. Moreover, the quantum phases find the approximate superposition of periods of the function by applying quantum Fourier transform (QFT) [6, 14].

The widely adopted RSA generates its public key using a product of *p* and *q* such that $N = p*q$, where both *p* and *q* are private prime numbers. Thus, RSA's secu-rity depends on the complexity of obtaining the factors *p* and *q* [22]. Thus, Shor's algorithm, when applied to quantum hardware, can crack the RSA cryptosystem.

Moreover, Gidney *et al*. [15] proved that Shor's algorithm efficiently cracks RSA-2048 within 8 h with 20 million qubits [14].

6.3 GROVER'S ALGORITHM

In 1996, Lov Grover [16] developed an algorithm that searches databases faster than a classical algorithm. Grover's approach is based on amplitude amplification and the property of quantum physics to provide a quantum search algorithm that can find a particular element, given an array of x number of elements. A classical search algorithm takes $O(N)$ while a quantum search algorithm by Grover has $O(\sqrt{N})$, a quadratic speedup.

Grover's algorithm can be used for the problem of obtaining a key or for the study of block ciphers. Thus, Grover's algorithm weakens the commonly used cryptosystem, namely AES-256/128 [2, 6, 14].

6.4 EXISTING POST-QUANTUM SECURITY SCHEMES ADDRESSING CONFIDENTIALITY

The emergence of quantum computing in recent years has brought both benefits and risks to stable cyber-physical systems [34]. The most threatened cryptosystems are widely used public-key algorithms, key management schemes, and digital signatures dependent on factorization, elliptic curve cryptography, and discrete logarithms [1]. This has motivated researchers to focus on the study and development of post-quantum cryptography. They are resistant against quantum as well as classical attacks and can be deployed in the existing network infrastructure [1]. Currently, there are five categories of primary post-quantum cryptography, namely [6, 8]:

- Code-based cryptography [6, 8]
- Lattice-based cryptography [6, 8]
- Supersingular elliptic curve isogeny [6, 8]
- Multivariate-based cryptography [6, 8]
- Hash-based cryptography [6, 8]

We divide post-quantum cryptography into two security goals: Confidentiality and integrity. The following sections provide post-quantum cryptography based on the type of algorithms used.

6.5 CODE-BASED CRYPTOGRAPHY

Code-based cryptography is cryptosystem, including symmetric and asymmetric, based on the difficulty of error-correcting codes [8]. It can be categorized into the following:

- Public-key encryption

- Digital signature
- Zero-knowledge protocols
- Pseudo-random number generator and stream cipher
- Hash functions

In this survey chapter, we review the code-based public-key cryptosystem as discussed in the following subsection. The most commonly known algorithm is the McEllice cryptosystem [1].

6.5.1 McEllice CryptoSystem

In 1978, a public-key encryption scheme that was based on hidden Goppa code [33] was proposed by Robert McEllice. Goppa codes are the relation between algebraic geometry and codes and are used as error-correcting codes. They rely on the NP-hard problem of decoding linear codes. The basic concept of the Goppa code depends on modular arithmetics. When a number series approaches a higher number, and once it reaches a specific number, the series starts from 0 again [1, 8]. A classic McEllice cryptosystem includes the following phases (assuming Alice and Bob are the two legitimate participants) [28, 33].

Key Generation: Bob selects a Goppa polynomial $g(z)$ of degree t, computes its corresponding generator matrix G, selects a random invertible matrix denoted as S and a random permutation matrix denoted as P. Bob uses all the parameters to compute $G' = SGP$ and announces his public key that includes (G', t). Bob's private key includes (S, G, P).

Encryption: Alice encrypts her message, represented in binary strings, by selecting and combining a random error vector, e, that has weight $\leq$ t, to mG'. Thus, Alice sends the following cipher, y in Equation 6.1.

$$y = m \times G' + e \tag{6.1}$$

Decryption: Bob uses his matrix, P, to derive y' as shown in Equation 6.2. Then, Bob applies the decoding algorithm, computing e', to y' to correct the errors and derive the codeword, m'. The m' is mS. Thus, Bob can easily derive m by $m' * S^{-1}$.

$$y' = y \times P^{-1} \tag{6.2}$$

According to the National Institute of Standards and Technology (NIST) Post-Quantum Cryptography (PQC) Standardization report [1], the McEllice cryptosystem is suitable for replacing traditional cryptosystems. It is quantum safe and provides security against Chosen Ciphertext Attack (CCA) and One-Wayness against Chosen Plaintext Attack (OW-CPA). Moreover, the McEllice cryptosystem is faster than most cryptosystems. However, it also comes with its drawbacks. It generates large public keys that can cause an implementation problem, especially in resource-constrained devices. Moreover, the ciphertexts are smaller than other post-quantum

Key Encapsulation Mechanisms (KEMS), but the ciphertexts are larger than the plaintexts. The McEllice cryptosystem is not applicable to generating authentication scheme or digital signature scheme [1].

6.6 LATTICE-BASED CRYPTOGRAPHY

Lattice-based cryptography has been proven to be strongly resistant to subexponential as well as quantum threats. They are based on the concept of lattices, sets of points within an n-size periodic structured space as shown in Figure 6.1 [13]. In simple terms, lattice can be considered as any regularly spaced grid of points. The security of the lattice-based cryptography depends on the complexity of lattice problems, mainly the shortest vector problem (SVP), the closest vector problem (CVP), or the shortest independent vectors problem (SIVP) [7, 13]. The SVP is deriving the minimum nonzero vector in the current lattice and is an NP-hard problem unsolvable by the present quantum algorithm [7].

6.6.1 Nth-Degree Truncated Polynomial Ring Units

In 1996, Hoffstein *et al*. [17] published an encryption scheme called the Nth-Degree Truncated Polynomial Ring Units (NTRU). It is public-key cryptography focused on SVP within a lattice [17]. It does not rely on factorization or disjunctive logarithms as the traditional public-key cryptosystem does. The basic NTRU operation is performed in the truncated polynomial rings as shown in Equation 6.3, such that N is a prime number and Z_q is the ring of integers modulo q [17].

$$R_q = \frac{Z_q\left[X\right]}{X^N - 1} \tag{6.3}$$

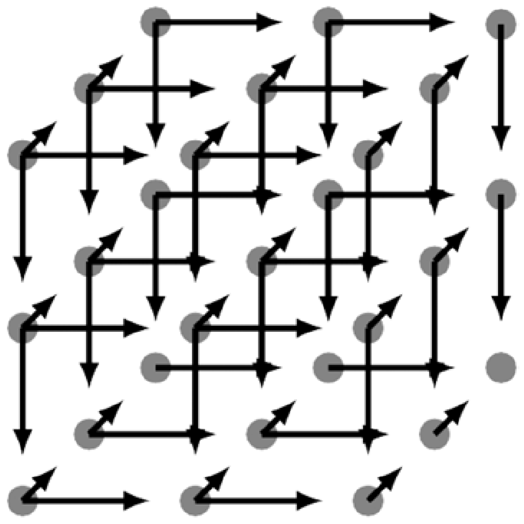


FIGURE 6.1 A lattice space in 3D.

A polynomial, f, in R_q can be written in Equation 6.4.

$$f = \left[f_0, f_1, ..., f_{N-1} \right] = \sum_{k=0}^{N-1} f_k X^K \tag{6.4}$$

Moreover, we denote the multiplication as *. Thus, $f * g$, where f and g are two polynomials, is given as a cyclic convolution product as shown in Equation 6.5.

$$f * g = h = [h_0, h_1, ..., h_{N-1}] \tag{6.5a}$$

$$h_k = \sum_{i+j=k \bmod N} f_i g_j \tag{6.5b}$$

The NTRU uses the above parameters to derive key pairs, encrypt the message, and then decrypt the cipher. Thus, we list the public parameters of the NTRU algorithm as follows [17, 28].

- N denotes a large prime number.
- p and q are positive numbers. $Gcd(p, q) = 1$, and, $p << q$.
- d_f, d_g, and d_r are integers to generate polynomials. The polynomials from which the private keys are uniformly chosen belong to the set $B(d_f)$, $B(d_g)$, and $B(d_r)$. The binding values in $B(d_r)$ are used during encryption.

Key Generation: Two random small polynomials, f and g, are selected, such that $f \epsilon B(d_f)$ and $g \epsilon B(d_g)$, $f_p = f^{-1}(\text{mod } p)$ and $f_q = f^{-1}(\text{mod } q)$. Then h is computed. Thus, the obtained public key is (N,h) and the private key is (f, f_q) [17, 28].

Encryption: To encrypt a message, m, a polynomial r is chosen randomly such that $r \epsilon B(d_r)$. Then, the message is encrypted to generate a cipher e as shown in Equation 6.6 [17, 28].

$$e = p * r * h + m(\text{mod})q \tag{6.6}$$

Decryption: The first step of decryption is to compute $f * e$ (mod q), and transform the obtained value, a, to polynomial whose coefficients are in the range $[-q/2, q/2]$. The following equations are used during decryption. The value of m can be derived from $f_p * a$ ($mod\ p$) [17, 28].

$$a = f * e(\text{mod}\, q) \tag{6.7}$$

$$\Rightarrow a = f * (p * r * h + m)(\text{mod}\, q) \tag{6.8}$$

$$=> a = f * p * r * g * f_q + f * m(\text{mod}\, q) \tag{6.9}$$

$$=> a = p*r*g + f*m(\mathrm{mod}\, q) \tag{6.10}$$

As per the NIST report [1], the NTRU remains unbreakable against current attacks as well as attacks using present quantum hardware. Moreover, it generates shorter key pairs as compared to McEllice's cryptosystem. Thus, NTRU is declared a suitable alternative to RSA and ECC.

6.6.2 ring-lWe

Ring learning with errors (Ring-LWE) [21] is a post-quantum cryptosystem that relies on the learning with errors (LWE) problem assigned to polynomial rings over finite fields. In Ring-LWE, the coefficients of polynomials can be added and multiplied within a finite field, *F*q*, such that the coefficients are less than q [21, 28]. The Ring-LWE can be deduced to an SVP within a lattice.

A classic Ring-LWE problem follows the following steps, assuming Alice and Bob are the two participants [21].

6.6.2.1 Alice and Bob accord on a shared complexity value of n, such that n is the highest coefficient power.

6.6.2.2 They both derive q such that $q = 2^n - 1$.

6.6.2.3 The polynomial operations are computed with a modulus of q.

6.6.2.4 Alice creates a set of polynomial values, A, as in Equation 6.11.

$$A = a_{n-1}x^{n-1} + ... + a_1x^2 + a_1x + a_0 \tag{6.11}$$

6.6.2.5 Alice divides A by $\varphi(x) = x^n + 1$.

6.6.2.6 Alice computes two polynomials: Error polynomial (e) and secret polynomial (s).

$$e\_A = e_{n-1}x^{n-1} + ...e_1x + e_0 \tag{6.12}$$

$$s\_A = s_{n-1}x^{n-1} + ...s_1x + s_0 \tag{6.13}$$

6.6.2.7 Alice, then, computes v_A using A, e_A and, s_A as shown in Equation 6.14.

$$v_A = A * s_A + e_A \tag{6.14}$$

6.6.2.8 Alice sends A and v_A to Bob, and Bob follows the same algorithm to generate his own error polynomial e_B and secret polynomial s_B.

6.6.2.9 Bob also creates v_B and sends it to Alice.

6.6.2.10 Alice multiplies the v_B with her own secret polynomial, and further computes it as shown in Equation 6.1

6.6.2.11 In the meantime, Bob uses the same algorithm to generate its own shared secret key as in Equation 6.16.

Then, both Alice and Bob extract the noise from the shared secret key to obtain the same shared secret value [21]. This algorithm can be implemented in the key-exchange scheme, digital signature, and homomorphic encryption, and the public and private key sizes are larger than that of a traditional public-key cryptosystem [21]. One of the post-quantum algorithms that focuses on Ring-LWE is called New Hope [1].

6.7 SUPERSINGULAR ELLIPTIC CURVE ISOGENY-BASED CRYPTOGRAPHY

Supersingular elliptic curve isogeny cryptography is quantum-safe cryptography that depends on the hardness to find isogenies among the supersingular elliptic curves. Feo *et al*. [10] developed a post-quantum cryptography algorithm, supersingular isogeny Diffie–Hellman (SIDH) key exchange, analogous to Diffie–Hellman, and is based on supersingular isogeny problem defined as following [9, 29].

Definition 6.1. [9, 29] Given two supersingular elliptic curves, E_1, E_2, in a finite field

k, where $|E_1| = |E_2|$, solve an isogeny function f: $E_1 \dashrightarrow E_2$.

Public Parameters: The public parameters are, first, agreed upon by both participants (Alice, Bob) during the SIDH protocol. Alice and Bob establish a large prime p based on a supersingular elliptic curve, denoted as E, over a field, F_p2 such that F refers to the set of integers modulo, p, and E can be represented as the following [9, 29].

$$E : y^2 = x^3 + ax + b \tag{6.17}$$

$$p = 2^a + 3^b - 1, \tag{6.18}$$

where a and $b \in N$. Moreover, they also agree on the basis P_A, Q_A of $E[2^A]$ and P_B,

Q_B of $E[2^B]$.

Key Generation: Alice selects a number, r_A, randomly extracted from the set $0 \leq r_A < 2^b$-1. She uses her basis and the selected random number to compute the kernel to generate her isogeny function φ_A. Alice, using her isogeny function with ker φ_A (Equation 6.19), generates the public key denoted as $E_A = \varphi_A(E)$, $\varphi_A(\mathrm{P}_B)$, $\varphi_A(\mathrm{P}_B)$. Her private key is r_A [9, 29].

$$\ker\varphi_A = \langle P_A + r_A Q_A \rangle \tag{6.19}$$

Parallely, Bob follows the same algorithm to generate his own key pairs. His private key is r_B randomly extracted from the set $0 \le r_A < 3^b - 1$. His public key is her public key denoted as $E_B = \varphi_B(E)$, $\varphi_B(P_A)$, $\varphi_B(P_A)$.

Shared Secret Key Generation: The public keys, E_A, E_B, are exchanged by Alice and Bob. Alice generates another isogeny function, ψ_A, with kernel function based on the basis parameters, P_A, Q_B, as described in Equation 6.20 [9, 29].

$$\ker\left(\psi_A\right) = \langle \varphi_B\left(P_A\right) + \left[r_A\right] \varphi_B\left(Q_A\right)\rangle \tag{6.20}$$

Alice then generates j-invariant of the curve $E_{AB} = \psi_A(E_B)$. Similarly, Bob calculates ψ_B and then proceeds to derive the j-invariant of curve $E_{BA} = \psi_B(E_A)$. The j-invariant derived by both Alice and Bob are the shared secret key as described in Equation 6.21 and is simply described in Figure 6.2 [9, 29].

$$j(\varphi_B\left(E_A\right) = j\left(E_{BA}\right) = j(E_{AB} = j(\varphi_A\left(E_B\right). \tag{6.21}$$

The post-quantum scheme that relies on the complexity of supersingular isogeny curve problem provides enough resistance against attack using a quantum algorithm. A key encapsulation mechanism based on SIDH, named supersingular isogeny-based key (SIKE), provides the smallest public key sizes as that of other post-quantum schemes, as well as small ciphertexts. However, the performance of the supersingular isogeny-based cryptosystems is lower than that of other post-quantum cryptosystems. Thus, it requires efficient optimization algorithms to increase the performance [1].

6.8 EXISTING POST-QUANTUM SECURITY SCHEMES ADDRESSING INTEGRITY

In this section, we provide an overview of the types of post-quantum cryptography addressing integrity as a security goal. Currently, three types of post-quantum cryptography implement a signature scheme, mainly, lattice-based, multivariate-based, and hash-based cryptography.

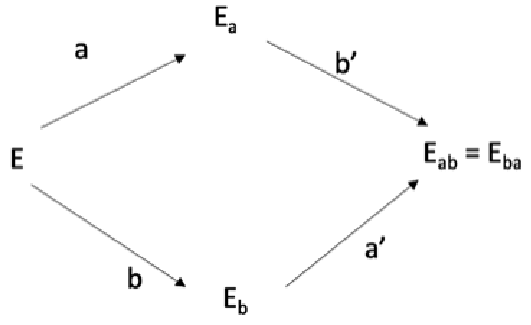

$$\begin{tikzcd}
 & E_a \arrow[dr, "b'"] & \\
E \arrow[ur, "a"] \arrow[dr, "b"'] & & E_{ab} = E_{ba} \\
 & E_b \arrow[ur, "a'"'] &
\end{tikzcd}$$

FIGURE 6.2 Isogeny computation by Alice and Bob to obtain the common shared secret key.

6.8.1 Lattice Based

The widely used and known lattice-based cryptographies are NTRU sign and BLISS. Lattice-based signatures mainly depend on the NP-hardness of the short vectors in lattice spaces [28]. The following subsections describe the NTRU signature and BLISS.

6.8.1.1 NTRU Signature

NTRU signature [18] is a signature scheme that follows after the NTRU encryption algorithm to provide authentication to the encrypted message. Like NTRU encryption, its security relies on the difficulty of solving the SVP. The basic operation of NTRU Signature occurs in the ring of polynomials, R, of degree less than $N-1$ represented as in Equation 6.3. The NTRU signature has three phases, mainly, Key Generation, Signature, and Verification [18]. The parameters used in the NTRU sign are as the following [18].

- Integer parameters: $(N, p, q, D_{min}, D_{max})$. p and q are relatively prime, and N denotes a large prime number. D_{min}, D_{max} are the deviations caused by the reduction modulo function.
- Set of polynomials: F_f, F_g F_w, F_m.

Key Generation: Bob selects two polynomials, denoted as, f and g. They are further represented as $f = f_0 + p \cdot f_1$ and $g = g_0 + p \cdot g_1$, such that f_0 and g_0 are fixed universal polynomials, and, $f_1 \in F_f$ and $g_1 \in F_g$. Bob also computes the inverse of f modulo q and thus deriving the key pairs. The public key h is represented as Equation 6.22. The pair (f, g) is the obtained private key.

$$h \equiv f^{-1} * g mod\, q \tag{6.22}$$

Signing: Bob selects a polynomial w, such that $w \in F_w$. And w can be represented as in Equation 6.23.

$$w = m + w_1 + p \cdot w_2 \tag{6.23}$$

In Equation 6.23, w_1 and w_2 are small polynomials. The w was further used to compute the signature, s, in Equation 6.24. The final signature includes the set (m, s).

$$s \equiv f * w\left(mod\, q\right) \tag{6.24}$$

Verification: Alice now verifies Bob's signature, (m, s). Alice, first, verifies the signature is null or not. Alice first tests whether the deviation satisfies, and then proceeds to use Bob's public key denoted as h. The h and the polynomial, t, is further computed as described in Equation 6.25. She then further verifies the deviation of t as well.

$$t \equiv h * s\left(mod\, q\right) \tag{6.25}$$

The performance of NTRU Sign is similar to NTRU Encrypt, as it provides almost similar public and private key sizes as compared to RSA and ECC and provides higher performance and a higher security level against traditional and quantum attack. Thus, it is suitable for the current network infrastructure [1].

6.8.1.2 BLISS

BLISS [26] is a signature-generating algorithm whose operation relies on the ring R_q with a power of 2 where q is the prime number as represented in Equation 6.26 [28, 31].

The $x^n + 1$ has a root in Z_q. Moreover, $q = 1 \; mod \; 2n$. It also has three phases such as key generation, signing, and verifying as the following [26, 31].

$$R_q = \frac{Z_q\left[X\right]}{X^n - 1} \tag{6.26}$$

Key Generation: We denote two polynomials as f and g with d_1 coefficients in $\{\pm 1\}$ and d_2 coefficients in $\{\pm 2\}$, where $d_1 = \delta_1 n$ and $d_2 = \delta_2 n$. The private key $S \in R_{2q}^2$ and the public key $A \in R_{2q}^2$ are generated by computing the following equations.

$$S = \left(s_1, s_2\right) = \left(f, 2g + 1\right) \tag{6.27}$$

$$a_q = \left(s_2/s_1\right) mod\, q \tag{6.28}$$

$$A = \left(2a_q, q - 2\right) \tag{6.29}$$

Signing: The message to be signed is denoted as μ from the message space, M. At first, two polynomials y_1 and y_2 are selected from a disjunctive Gaussian distribution D_σ, where σ is the width parameter of the distribution. And, y is a set of (y_1, y_2). Then, using a Hash function, H, a challenging variable is generated as described in the following.

$$c = H\left(A \times y, \mu\right) \tag{6.30}$$

$$H : R \times M - ® \;\; R \tag{6.31}$$

The signing involves Greedy approximation algorithm that produces a variable v, where $v = Sc'$ for $c' = c \; mod \; 2$. Then, we select a bit value, b, such that $b \in \{0,1\}$ to further compute the signature $sign = y + bv$.

Verification: The receiver receives the sign and c is verified if it is smaller than the discrete Gaussian parameter that has a width variable σ. If the sign satisfies the condition, the receiver further verifies Equation 6.32.

$$H\left(az+qc\,mod\,2q,\mu\right)\overset{?}{=}c \tag{6.32}$$

BLISS is a lattice-focused signature algorithm [28, 31]. It generates key and signature size similar to the RSA algorithm, and is resistant against quantum attack [28, 31].

6.8.2 Multivariate Cryptography

A multivariate public-key cryptosystem is dependent on the NP-hardness of deciphering the multivariate polynomials over the finite fields [12, 13]. The NIST report claims that various multivariate public cryptosystems have been proposed. However, some of them are broken [1]. The class of trapdoor one-way functions is an integral property of PKC. For example, NTRU depends on the lattice structure, and ECC depends on the elliptic curve group. Multivariate cryptography depends on the one-way function as a multivariate quadratic polynomial public map over a finite field [1, 12, 13]. We denote the set of quadratic polynomials as $P = (p_1(w1,...w_n),...,p_m(w_1,...w_n))$, where each p_i is a quadratic polynomial in $w = (w_1,...w_n)$ [1, 12, 13]. One of the multivariate cryptography is Rainbow that has been selected as Round 3 finalists by NIST [1].

6.8.2.1 Rainbow

In 2004, Ding *et al*. [11] developed a multivariate post-quantum signature focused on the Oil-Vinegar signature scheme. Its security relies on the NP-hardness of solving a set of random multivariate quadratic schemes. Like any other signature scheme, it has three phases, including key generation, signature, and verification as the following [11, 12].

Key Generation: Two keys are generated in this phase. The private key includes two invertible affine maps, L_1 and L_2. It also includes the map, denoted as, F. Moreover, the public key consists of the field referred to as K as well as the composed map, $P(x)$ [11, 12].

Signing: In the signature generation phase, given a document $d \in \{0,1\}$ ▯, the sender uses a hash function, $h = h(d)$. Then, it computes it further, as shown in the following equations, to generate the signature, z.

$$x = L_1^{-1}\left(h\right) \tag{6.33}$$

$$y = F^{-1}\left(x\right) \tag{6.34}$$

$$z = L_2^{-1}(y) \tag{6.35}$$

Verification: The receiver computes the hash of the composite map on *z*, where *h'* = *P*(*z*), and then further computes the hash of the document. If the generated hash matches with the received hash, the signature is validated and accepted.

Rainbow offers shorter signatures, only 258 bits for the NIST level 1 security compared to other post-quantum signature schemes. Moreover, the algorithm used in signing the document and verifying the signature is highly efficient and faster than other post-quantum schemes. One drawback of the Rainbow algorithm is that the key generation process is slow and needs to be more efficient [1].

6.8.3 Hash-Based Signature Scheme

HSS schemes are signature-generating algorithms that rely on cryptographic hash functions addressing at least one of the following security properties: Pre-image, second pre-image, and collision resistance [6, 25]. The hash-based signature scheme can be further classified into two, mainly, stateless and stateful schemes. A stateful hash-based signature scheme relies on Merkle's tree using OTS parameters, whereas a stateless hash-based signature scheme includes a hyper-tree with both a one-time signature (OTS) scheme and a few-time signature (FTS) schemes [32].

6.8.3.1 Stateful Signature Scheme

Lamport Signature: In 1970, Leslie Lamport [19] proposed a one-time signature algorithm which is resistant against traditional as well as quantum attacks. The primary advantage of the Lamport signature is that it exploits a secure cryptographic hash function to derive a public key, which is further processed to sign a message [19]. Thus, the security of Lamport primarily depends on the secrecy of the hash function [19]. Moreover, the Lamport signature generates large key sizes as well as signature sizes. Thus, it is unsuitable for almost any network infrastructure. The public key and private key (of size 256 bit) of the Lamport signature can be represented as the following [4].

$$\text{Private Key} = \left(x_0, y_0, x_1, y_1 \ldots, x_{255}, y_{255}\right) \tag{6.36}$$

$$\text{Public Key} = \left[h(x_0)h(y_0) \,\|\, h(x_1) \,\|\, h(y_1) \ldots. \,\|\, h(x_{255}) \,\|\, h(y_{255})\right] \tag{6.37}$$

Winternitz One-time Signature Scheme (WOTSS): To address the large key sizes of the Lamport signature scheme (LSS), WOTSS is proposed. The primary idea of WOTSS is to implement a certain count of the chain of functions that start from feeding on random inputs [4]. In WOTS, the random data are the secret key, and the public key includes the output derived from the chains [4]. A message is signed by mapping it to one of the intermediate values of each chain. WOTS is an optimized version of LSS that uses a parameter, *w*. The Winternitz parameter, *w*, is inversely proportional to the signature size. A larger *w* generates a smaller signature. Thus, WOTSS is suitable for memory-constrained devices. However, the time complexity increases exponentially as *w* increases [4].

Merkle's Signature Scheme: To address the drawbacks of one-time signature (OTS), Ralph Merkle proposed an algorithm named Merkle Signature Scheme (MSS). It merges various OTS key pairs and obtains multiple concatenated key pairs into a single binary hash tree [4]. During the tree construction, the signature keeps concatenating the string of intermediate nodes with respect to the tree root to generate the authentication path. The authentication path verifies the signature and generates the path of the tree [4]. A simple Merkle's tree is a binary tree with each node is a hash of its following child node. Thus, the root of the tree is considered to be the final public key, and the Merkle tree leaves are the hashes of the OTS public key. Figure 6.3 illustrates a simple Merkle tree [4].

The OTS signatures generate large public keys, and it needs to generate a novel public key every time a message needs to be sent. Thus, it increases the computation cost [4]. The MSS obtains a public key for signing multiple messages, such that the frequency of messages must be a power of 2 [4]. Given $M = 2^n$, it generates the public key, X_i, and private key, Y_i, such that Y_i is within the interval $1 \le i \le 2n$, where $i = n$, being the root level of the tree [4].

HORS: Reyzin *et al*. [27] proposed a few-time signature (FTS) scheme, using hash functions, that generates a secret key that contains *n* random numbers generated from a pseudo-random number function. The public key is derived from computing the *n* hashes of the random elements in the secret key. The signature generated contains *k* secret key values [27].

$$m = klogn = k\tau \tag{6.38}$$

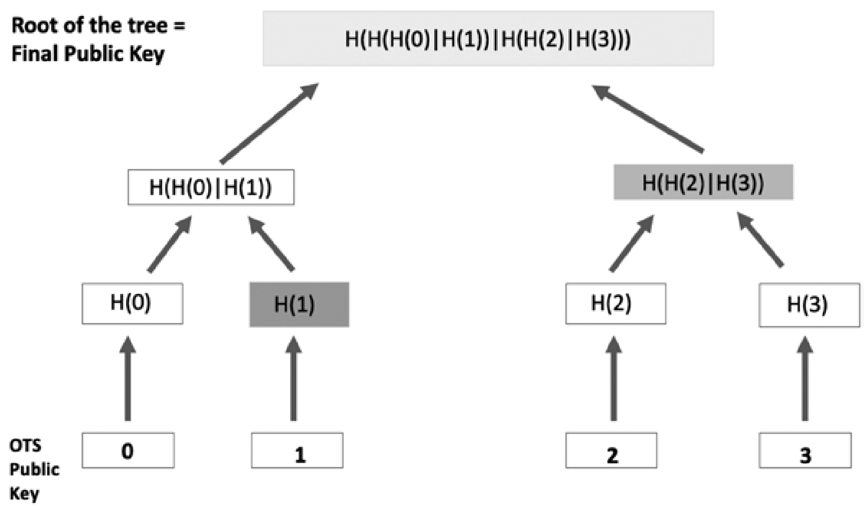

Root of the tree = Final Public Key
H(H(H(0)|H(1))|H(H(2)|H(3)))
H(H(0)|H(1))
H(H(2)|H(3))
H(0)
H(1)
H(2)
H(3)
OTS Public Key
0
1
2
3

FIGURE 6.3 A simple illustration of Merkle tree. The gray-shaded nodes are authentication path, and the root of the tree is the public key.

The relationship between the message (*m*), public key, and secret key values are shown in Equation 6.38, where $k \in \mathrm{N}$ and $n = 2^{\tau}$ for $\tau \in \mathrm{N}$ [27].

6.8.3.2 Stateless Signature Scheme

SPHINCS: It is a stateless signature scheme that is quantum resistant, and its general construction involves hyper-tree with height *h*-layer and *d*-layer. Each intermediate tree is a Merkle tree with height *h*/*d*. The security parameters of a SPHINCS tree are, mainly, *n* refers to the HORS tree (HORST) and WOTS hash bit sizes, *m* refers to the bit-length of the message, *h* refers to the height of the hyper-tree, *d* refers to the layers of the hyper-tree, *w* is the WOTS parameter, *t* is the frequency of HORST secret key elements, and *k* is the count of published HORST secret key elements [3, 5].

The general construction of a SPHINCS tree has the four following trees and has been shown in Figure 6.4[3, 5].

- The hyper-tree: The hyper-tree is the main tree that generates the root as the public key. It has height denoted as *h*. The hyper-tree is further segmented into *d*-layers of a *type-2* tree. The hyper-tree leaves are the instances of the *type-4 trees,* the HORST tree. For example in SPHINCS-256, *h* is 60 and *d* is 12 [3, 5].
- The sub-trees: The sub-trees are the intermediate tress that is based on Merkle trees and have a height of h/d. The leaves of the sub-trees are the root of the *type-3* trees; that is, the roots are compressed WOTS public keys that feed as the leaves to the next layer's tree.
- WOTS public-key compression tree: They are known as L-trees of height $\log_2 l$, where *l* is the count of leaves. The leaves in the sub-trees are derived

AU: Is h/d ok as written?

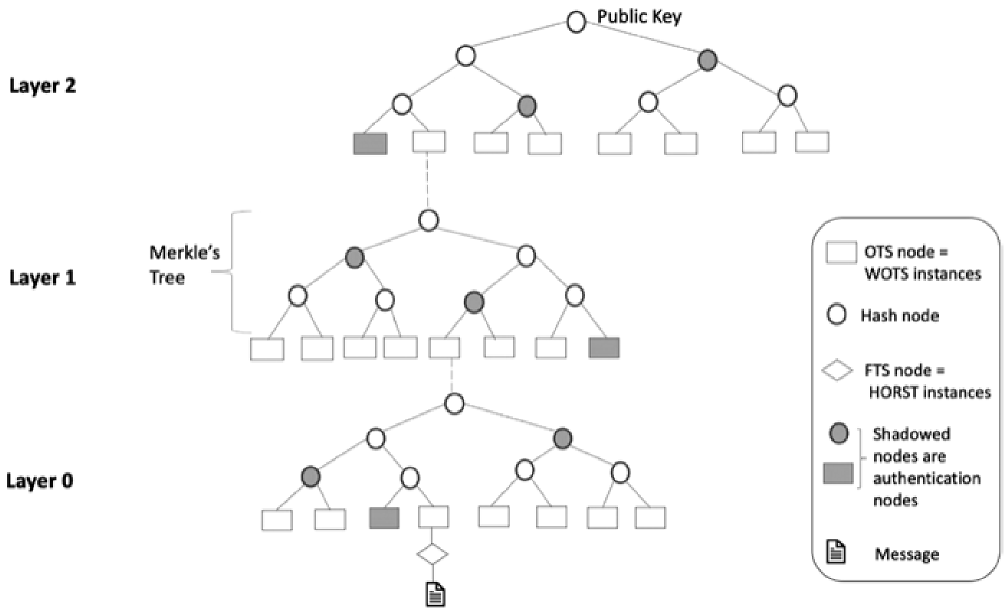

Public Key
Layer 2
Layer 1
Merkle's Tree
Layer 0
OTS node = WOTS instances
Hash node
FTS node = HORST instances
Shadowed nodes are authentication nodes
Message

FIGURE 6.4 General SPHINCS tree construction with $h = 9$ and $d = 3$.

from the WOTS public keys by computing an unbalanced binary tree that has *l* leaf nodes, known as L-trees [3, 5].

- HORS public-key compression tree: In 2015, Bernstein *et al*. [5] proposed HORS tree (HORST) for implementing at the lowest level of the SPHINCS tree as the FTS. The bottom layer of the hyper-tree also contains the Merkle tree of height $\tau = \log_2 t$, such that t is the count if HORST public-key instances. Unlike stateful hash-based schemes, the stateless signature scheme does not keep a history of used key pairs. Thus, SPHINCS uses multiple HORST key pairs to correctly use key pairs more than once [3, 5].

SPHINCS' proof of security solely depends on the underlying hash function. As per NIST [1], SPHINCS provides robust security against traditional and quantum algorithms. Thus, NIST declared SPHINCS to be the Round 3 finalist. Moreover, the public keys of SPHINCS are small enough. However, the signatures are large, and the signature generator process is slow. Thus, SPHINCS requires optimization techniques to increase the speed such that it is well suitable for memory-constrained devices [1].

6.9 A GENERIC HYBRID CRYPTOSYSTEM AGAINST CLASSICAL AND QUANTUM ATTACK

Currently, various cyber-physical fields and their frameworks including industrial control systems used in nuclear power plants, oil and gas industry, traffic management, and space stations rely on classical cryptography protocols. The emergence of quantum computing threatens the security framework of the current cyber-physical systems. Thus, we propose an abstract concept of a generic hybrid cryptosystem to enhance the security of current cyber-physical systems against both classical and quantum attacks. The hybrid cryptosystem comprises a post-quantum key encapsulation mechanism, such as a code-based algorithm followed by the classical encryption scheme, such as a widely used advanced encryption standard (AES) algorithm along with a post-quantum digital signature or a classical quantum-resistant signature. Our future work focuses on the proposed hybrid security framework to secure industrial control systems.

6.10 CONCLUSION

This chapter provides a study of the well-known post-quantum signatures against the quantum attack. It first discusses the motivation of the emergence of the research area on post-quantum cryptography by focusing on two quantum algorithms, Shor's and Grover's algorithms. The traditional asymmetric cryptography, mainly RSA and ECC, is already broken by Shor's algorithm. However, the symmetric cryptography, AES, is not broken but weakened by Grover's search algorithm. Furthermore, this chapter lists the current well-known and few NIST Round 3 finalist post-quantum schemes and categorized them based on two criteria: their mathematical model and their security goal. Table 6.1 provides a generic comparison of post-quantum cryptography (PQC) algorithms. Table 6.2 lists the PQC algorithms addressing confidentiality. Table 6.3 provides the comparison of PQC algorithms that focuses on integrity. Post-quantum cryptography can be divided into the following five categories based on the types of mathematical models: code-based, lattice-based, supersingular elliptic curve isogeny-based, multivariate-based, and hash-based scheme. The hash-based scheme can further be categorized into two: stateless signature and stateful signature. Based on their security goal, the algorithms can be further categorized into two: algorithms addressing confidentiality and algorithms addressing integrity. Furthermore, this chapter also provides the research gap in the algorithms and provides a foundation for future research and improvements.

REFERENCES

1. Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., Kelsey, J., Liu, Y.-K., Miller, C., Moody, D., Peralta, R., et al. (2020). *Status report on the second round of the NIST post-quantum cryptography standardization process*. US Department of Commerce, NIST.
2. Amy, M., Di Matteo, O., Gheorghiu, V., Mosca, M., Parent, A., and Schanck, J. (2016). Estimating the cost of generic quantum pre-image attacks on sha-2 and sha-3. In International Conference on Selected Areas in Cryptography, pages 317–337. Springer.
3. Aumasson, J.-P. and Endignoux, G. (2018). Improving stateless hash-based signatures. In Cryptographers' Track at the RSA Conference, pages 219–242. Springer.
4. Becker, G. (2008). *Merkle signature schemes, merkle trees and their cryptanalysis*. Ruhr-University Bochum, Tech. Rep.
5. Bernstein, D. J., Hopwood, D., Hu¨lsing, A., Lange, T., Niederhagen, R., Papachristodoulou, L., Schneider, M., Schwabe, P., and Wilcox-O'Hearn, Z. (2015). Sphincs: Practical stateless hash-based signatures. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 368–397. Springer.
6. Bernstein, D. J. and Lange, T. (2017). Post-quantum cryptography. *Nature*, 549(7671):188–194.
7. Bl¨omer, J. and Naewe, S. (2009). Sampling methods for shortest vectors, closest vectors and successive minima. *Theoretical Computer Science*, 410(18):1648–1665.
8. Chen, L., Chen, L., Jordan, S., Liu, Y.-K., Moody, D., Peralta, R., Perlner, R., and Smith-Tone, D. (2016). *Report on post-quantum cryptography*, volume 12. US Department of Commerce, National Institute of Standards and Technology.
9. Costello, C. (2019). Supersingular isogeny key exchange for beginners. In International Conference on Selected Areas in Cryptography, pages 21–50. Springer.
10. De Feo, L., Jao, D., and Pluˆt, J. (2014). Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209–247.
11. Ding, J. and Schmidt, D. (2005). Rainbow, a new multivariable polynomial signature scheme. In International Conference on Applied Cryptography and Network Security, pages 164–175. Springer.
12. Ding, J. and Yang, B.-Y. (2009). Multivariate public key cryptography. In *Post-quantum cryptography*, pages 193–241. Springer.
13. Fern´andez-Caram´es, T. M. (2019). From pre-quantum to post-quantum iot security: A survey on quantum-resistant cryptosystems for the internet of things. *IEEE Internet of Things Journal*, 7(7):6457–6480.

TABLE 6.1
Comparison of Post-Quantum Cryptography (PQC) Algorithms

	PQC			
Category	**Algorithms**	**Advantages**	**Disadvantages**	**Examples**
Addressing confidentiality	Code-based encryption	Strong proof of security. Fast encryption. Ciphertext size is small.	Public keys are large	McEllice cryptosystem
	Lattice-based encryption	Ciphertext size is short. Public and private keys are small. Fast encryption process.	Need more understanding of the security. Further security analysis is required.	NTRU Encrypt Ring-LWE
	Supersingular elliptic curve isogeny	Generates the smallest public key sizes of all PQC schemes. Generates small ciphertexts.	Computation cost is high Optimization techniques are needed.	SIDH SIKE

Addressing integrity	Lattice-based signature		Generates short public and private keys. Fast signature generation.	Need more understanding of the security. Further security analysis is required.	NTRU Sign BLISS
	Multivariate-based signature		Generate short signatures.	Need more understanding of the security. Further security analysis is required.	Rainbow
	Hash-based signature	Stateful	Smaller signature size. Faster signature generation.	Need maintenance in usage of non-repeated key pairs	Lamport signature WOTS MSS HORS
		Stateless	Do not need to monitor non-repeated key pairs usage.	Larger signature size. Signature generation process is slower.	SPHINCS

TABLE 6.2

Comparison of Post-Quantum Cryptography (PQC) Algorithms That Addresses Confidentiality

PQC Encryption Algorithm	Overview	Advantage	Disadvantage	NIST Round 3 Finalist	Research Gap
McEllice cryptosystem	It is based on hidden Goppa code	Faster than most cryptosystems CCA-resistant and one-wayness CPA resistant. Smaller ciphertexts than that of other PQCs.	Large public keys. Not suitable for resource-constrained devices.	Yes	Compression technique needed.
NTRU encrypt	It is based on the hardness of SVP within a lattice.	Quantum resistant. Smaller public and private keys than McEllice. Suitable alternative to RSA and ECC.	Need more understanding of the security.	Yes	Further security analysis is required.
Ring-LWE	It relies on the learning with errors problem referred to rings of polynomials over finite fields.	Versatile algorithm. Can be implemented as key management, digital signatures and encryption scheme.	Public and private keys are larger than that of traditional cryptography.	No	Compression algorithm is required.

SIDH based SIKE	It focuses on hardness to find isogenies among supersingular elliptic curves.	Strong security against quantum and classical attacks. Smallest public keys of all PQCs. Generates small ciphertexts as well. Suitable for resource-constrained devices.	Performance is low.	Yes	Optimization is required to increase the efficiency.

TABLE 6.3
Comparison of Post-Quantum Cryptography (PQC) Algorithms That Addresses Integrity

PQC Signature Scheme	Overview	Advantage	Disadvantage	NIST Round 3 Finalist	Research Gap
NTRU sign	It relies on the hardness of SVP in a lattice.	Smaller public and private keys compared to RSA and ECC. Provides high performance.	Is complementary with NTRU encrypt.	Yes	Further security analysis is required.
BLISS	It relies on a ring with a power of 2 in a lattice space.	Key and signature size similar to RSA	Need more understanding of the security. Further security analysis is required.	No	Further security analysis is required.
Rainbow	Based on Oil-Vinegar Signature scheme.	Generates shorter signature, for NIST security level 1, as compared to other PQCs. Signature and verification are efficient and fast.	Key generation process is slow.	Yes	Optimization technique is needed.
Lamport signature (LOTSS)	It relies on a secure cryptographic hash algorithm to sign a message.	Versatile.	Security solely lies in the secrecy of the hash function. Generates large key size.	No	Key sizes need to be compressed.
WOTS	Optimized version of Lamport signature	Smaller signature size	Time cost increases exponentially as signature size increases.	No	Optimization algorithm is required.

HORS and HORST	It is a few-time signature scheme	Both of them are quantum resistant. HORST has smaller public key and signature size than that of HORS.	Signature size is larger than WOTS. Time cost of HORST is greater than that of HORS.	No	Optimization and compression technique is required.
MSS	Stateful hash-based scheme that uses a single binary hash tree to generate public keys and signature.	Uses one public to sign more than one messages.	Signature size and key sizes are still large in MSS. They were improved in the extension of MSS.	No	Need maintenance of the usage of non-repeated key pairs
SPHINCS	It is a stateless hash signature scheme based on hyper-tree including Merkle tree, OTS and FTS scheme.	Do not need maintaining the non-repeated key pairs usage.	Larger signature size. Signature generation process is slower.	Yes	Optimization and compression techniques are required.

14. Ghosh, S. and Sampalli, S. (2019). A survey of security in scada networks: Current issues and future challenges. *IEEE Access*, 7: 135812–135831.
15. Gidney, C. and Eker˚a, M. (2019). How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits. *arXiv preprint arXiv:1905.09749*.
16. Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pages 212–219.
17. Hoffstein, J., Pipher, J., and Silverman, J. H. (1998). Ntru: A ring-based public key cryptosystem. In International Algorithmic Number Theory Symposium, pages 267–288. Springer.
18. Hoffstein, J., Pipher, J., and Silverman, J. H. (2001). Nss: An ntru lattice-based signature scheme. In International Conference on the Theory and Applications of Cryptographic Techniques, pages 211–228. Springer.
19. Lamport, L. (1979). Constructing digital signatures from a one-way function. pp. 1–7. SRI International.
20. Lomonaco, S. (2002). Shor's quantum factoring algorithm. *Proceedings of Symposia in Applied Mathematics*, volume 58, pages 161–180.
21. Lyubashevsky, V., Peikert, C., and Regev, O. (2010). On ideal lattices and learning with errors over rings. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 1–23. Springer.
22. Minni, R., Sultania, K., Mishra, S., and Vincent, D. R. (2013). An algorithm to enhance security in rsa. In 2013 4th International Conference on Computing, Communications and Networking Technologies (ICCCNT), pages 1–4. IEEE.
23. Nurhadi, A. I. and Syambas, N. R. (2018). Quantum key distribution (qkd) protocols: a survey. In 2018 4th International Conference on Wireless and Telematics (ICWT), pages 1–5. IEEE.
24. Padamvathi, V., Vardhan, B. V., and Krishna, A. (2016). Quantum cryptography and quantum key distribution protocols: A survey. In 2016 IEEE 6th International Conference on Advanced Computing (IACC), pages 556–562. IEEE.
25. Perlner, R. A. and Cooper, D. A. (2009). Quantum resistant public key cryptography: a survey. In Proceedings of the 8th Symposium on Identity and Trust on the Internet, pages 85–93.
26. Pessl, P., Bruinderink, L. G., and Yarom, Y. (2017). To bliss-b or not to be: Attacking strongswan's implementation of post-quantum signatures. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pages 1843–1855.
27. Reyzin, L. and Reyzin, N. (2002). Better than biba: Short one-time signatures with fast signing and verifying. In Australasian Conference on Information Security and Privacy, pages 144–153. Springer.
28. Roy, K. S. and Kalita, H. K. (2019). A survey on post-quantum cryptography for constrained devices. *International Journal of Applied Engineering Research*, 14(11):2608–2615.
29. Seo, H., Anastasova, M., Jalali, A., and Azarderakhsh, R. (2020). Supersingular isogeny key encapsulation (sike) round 2 on arm cortex-m4. *IEEE Transactions on Computers*, 70.10(2020): 1705–1718.
30. Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, 41(2):303–332.
31. Staffas, R. (2016). Post-quantum lattice-based cryptography, pp. 1–67. KTH SCI.
32. Suhail, S., Hussain, R., Khan, A., and Hong, C. S. (2020). On the role of hash-based signatures in quantum-safe internet of things: Current solutions and future directions. *IEEE Internet of Things Journal*. Jul 30; 8(1): 1–7.

33. Valentijn, A. (2015). Goppa codes and their use in the mceliece cryptosystems. Syracuse University Honors Program Capstone Projects. 845. pages 1-40. Available: https://www.semanticscholar.org/paper/Goppa-Codes-and-Their-Use-in-the-McEliece-Valentijn/6cd1b5657d1c228c30d7705f41dcb2e3a6fe2d74
34. Zhang, X., Dong, Z. Y., Wang, Z., Xiao, C., and Luo, F. (2015). Quantum cryptography based cyber-physical security technology for smart grids. 10th International Conference on Advances in Power System Control, Operation & Management (APSCOM 2015): 5-61.

Article

A Quantum-Based Signcryption for Supervisory Control and Data Acquisition (SCADA) Networks

Sagarika Ghosh [1], Marzia Zaman [2], Bernard Plourde [3] and Srinivas Sampalli [1,*]

1 Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 4R2, Canada; sagarika.ghosh@dal.ca
2 Research and Development, Cistel Technology, Ottawa, ON K2E 7V7, Canada; marzia@cistel.com
3 Sanstream Technology, Gatineau, QC J8Y 2V5, Canada; bplourde@sanstream.ca
* Correspondence: srini@cs.dal.ca

Abstract: Supervisory Control and Data Acquisition (SCADA) systems are ubiquitous in industrial control processes, such as power grids, water supply systems, traffic control, oil and natural gas mining, space stations and nuclear plants. However, their security faces the threat of being compromised due to the increasing use of open-access networks. Furthermore, one of the research gaps involves the emergence of quantum computing, which has exposed a new type of risk to SCADA systems. Failure to secure SCADA systems can lead to catastrophic consequences. For example, a malicious attack can take control of the power supply to a city, shut down the water supply system, or cause malfunction of a nuclear reactor. The primary purpose of this paper is to identify the new type of attack based on quantum computing and design a novel security scheme to defend against traditional attacks as well as the quantum attack. The methodology of the proposed signcryption is built on the foundation of the classical Bennett and Brassard 1984 (BB84) cryptographic scheme and does not involve computationally expensive third-party validation. The proposed signcryption scheme provides both encryption and intrusion detection. In particular, it detects the man-in-the-middle attack that can lead to other types of attacks. We have simulated the proposed algorithm using the Quantum Information Toolkit in Python. Furthermore, we have validated and analyzed the proposed design through security verification tools, namely, Scyther and PRISM.

Keywords: network security; quantum cryptography; qubits; post-quantum cryptography

check for updates

Citation: Ghosh, S.; Zaman, M.; Plourde, B.; Sampalli, S. A Quantum-Based Signcryption for Supervisory Control and Data Acquisition (SCADA) Networks. Symmetry **2022**, 1, 1625. https://doi.org/10.3390/sym14081625

Academic Editor: Qin Liao, Chin-Ling Chen and Jan Awrejcewicz

Received: 14 June 2022
Accepted: 4 August 2022
Published: 7 August 2022

1. Introduction

Supervisory Control and Data Acquisition (SCADA) systems monitor infrastructure or industrial processes such as in electric grids, water treatment plants, nuclear plants, and oil and natural gas pipelines. The SCADA architecture follows a hierarchical structure with three levels: (1) the supervisory level includes the Master Terminal Unit (MTU) and Human–Machine Interface (HMI); (2) The process control level involves sub-Master Terminal Unit (sub-MTU); (3) The field instrumentation control level contains field devices called Remote Terminal Units (RTUs) [1–4].

Due to their open access nature [5], SCADA systems suffer from various cyber-attacks and need appropriate detection and prevention techniques. For example, in December 2015, a cyber-attack using spear-phishing emails on power plants in Ukraine left 230,000 people without power for hours. Researchers and organizations throughout the world have proposed various security standards and protocols to improve SCADA network security. However, these traditional schemes do not exploit all the domains of security requirements and are therefore ineffective against many attacks [4].

The dawn of the quantum computer is not only valuable but also a risk to cybersecurity. According to Shor's algorithm [6] and Grover's algorithm [7], a quantum computer can crack classical encryption schemes, including Elliptic Curve Cryptography (ECC) [8]. The existing standards and communication protocols are inadequate for satisfactory protection against specific traditional attacks and quantum attacks.

There are two types of cryptography, namely, asymmetric and symmetric. The current SCADA standard, AGA-12, uses the popular RSA asymmetric algorithm. The security of RSA relies on the difficulty of factorizing large prime numbers. However, unlike classical algorithms, a quantum algorithm can easily crack the factorization problem by using the superposition principle. A classical algorithm consumes $O(N)$; however, an extensive search based on Grover's algorithm in a quantum setting takes $O(\sqrt{N})$ [9]. Thus, symmetric algorithms such as AES are weakened by Grover's quantum search algorithm. Moreover, Akinori et al. [10] have recently theoretically studied the quantum collision attack on the SHA-256 and SHA-512 algorithm. The advancement of quantum computing brings the requirement of stronger cyber-security to SCADA systems. We conclude that organizations and industries based on critical infrastructures should implement an amalgamation of quantum and classical algorithms to thwart both types of attacks.

Concerns about security of SCADA networks are not new [4]:

- Existing standards do not provide strong confidentiality, integrity and availability.
- Existing intrusion detection systems do not provide confidentiality.
- Existing key management protocols fail to provide confidentiality and availability.
- Current security schemes do not provide resistance against an attack launched by a quantum computer.

To the best of our knowledge, we did not come across any prior research work that provides a robust and secure system for existing SCADA systems using a quantum-secure encryption scheme and key exchange algorithm.

- We have identified one possible attack based on quantum computing on SCADA systems.
- We propose a new security scheme, to prevent unauthorized access to SCADA systems, and protect against the traditional as well as the quantum attack. Furthermore, the proposed scheme provides both encryption and intrusion detection. The scheme generates a signcrypted message by using the Bennett and Brassard 1984 (BB84) protocol and a one-time digital signature. Unlike other signcryption schemes, this scheme does not depend on a third party.

The difference between our approach and the existing works is that we use quantum key distribution instead of a traditional key exchange algorithm. Furthermore, we use the quantum key to generate the signcrypted data. The microchip circuits developed at QET-Labs can generate and distribute keys encoded in qubits by utilizing the quantum properties of superposition and entanglement. This chip presents an opportunity to apply Quantum Key Distribution (QKD) to resource-constrained devices [11]. This rapid development and adoption of chip-based QKD is the motivation of the proposed signcryption scheme.

Outline of the Paper

Section 2 provides an in-depth literature survey on existing error correction protocols used in Quantum Key Distribution and One-time Digital Signature schemes used to develop our proposed scheme. In Section 3, we identify the possible quantum attack on SCADA networks and list the attacks (both traditional and quantum) that can be defended by our proposed scheme. We also provide a brief survey on current SCADA security protocols. Section 4 describes the proposed scheme for SCADA networks. Section 5 provides the formal analysis of the scheme. Section 6 gives the evaluation results of the implemented proposed scheme. Section 7 provides conclusion and future work, respectively.

2. Background

This section provides a background survey on existing procedures used in quantum key exchange protocols as well as the ones that have contributed to developing the proposed scheme. Cryptography provides secure data exchange by providing the three main goals of security, namely, confidentiality, integrity, and authentication [12]. Based on the number of communication channels, it can be categorized into classical and quantum cryptography. Classical cryptography is based on the mathematical computation that uses a single communication channel. It can be further categorized into asymmetric and symmetric cryptography, depending upon the number and the characteristics of the keys used for encryption [12]. On the other hand, quantum cryptography relies on the principles of quantum mechanics, and it uses two channels, namely, a quantum channel and a public channel [13]. A quantum channel exchanges qubits, which are the fundamental units of information in quantum computing. A public channel or a classical channel is one where data are exchanged in the form of bits between the sender and the receiver.

Table 1 provides a description of some of the terminology used in quantum cryptography. To obtain more in-depth background on this subject, we refer the reader to the articles [12–20].

Table 1. Terminology of basic concepts in quantum cryptography.

Term	**Description**
Traditional or Classical Cryptography	It is a type of cryptography that is based on mathematical computation that uses a single communication channel [12].
Quantum Cryptography	Cryptography dependent on the principles of quantum mechanics and two channels, namely, quantum and public channels [13].
Heisenberg's Uncertainty Principle	This principle states that it is not possible to obtain the position and momentum of a photon with absolute accuracy [14].
Principle of Photon Polarization	This principle states that a photon can have a superposition of two or more quantum states at a time [15].
No-Cloning Theorem	This theorem states that one cannot produce an identical copy of an arbitrary quantum state of a photon [16].
Quantum Bit Error Rate (QBER)	QBER is the fraction of mismatched qubits exchanged between the sender and the receiver [13].
Error Correction Code (ECC)	ECC is an algorithm that detects and corrects errors in transmitted data [17].
Quantum Channel	Quantum Channel exchanges qubits and can create noise in the presence of an intruder or due to environmental factors [13].
Qubit	Qubit is a basic unit of data in quantum computing. It follows the properties of Principle of Photon polarization and Heisenberg's Uncertainty Principle [13].
Basis	Basis is a vector used to generate the superposed state of qubits [18].
Signcryption	An authenticated encryption scheme to provide both confidentiality and authenticity [19].
One-Time Signature (OTS)	OTS is based on hash-function that signs one message per key pair [20].

A quantum channel can create noise in the presence of an intruder or due to environmental factors, thus disrupting the key exchange. Quantum cryptography uses error correction protocols to resolve these errors. After the raw quantum key exchange, the system follows a process called key sifting. In a SCADA system that uses quantum cryptography, both the MTU and RTU randomly choose the basis, i.e., a vector, to generate and measure the superposed state of qubits. The sender and receiver negotiate their choice of basis via the less secure public channel. After the negotiation, they estimate the errors present in their respective keys [13].

This section presents the background work on existing error correction algorithms used in quantum key exchange protocols and the algorithms that have contributed to developing the proposed scheme. It has the following sub-sections:

- Existing Error Correction Protocols Used in Quantum Key Exchange System;
- Error Correction Protocol proposed for wireless network;
- Error Correction Protocol used in the Proposed Security Scheme;
- One-time Digital Signature.

2.1. Existing Error Correction Protocols Used in Quantum Key Exchange Systems

In this section, we discuss the following protocols that have been proposed and used to reconcile the errors while preserving security in the exchanged key [17].

2.1.1. Cascade

The Cascade protocol is one of the most prominent error reconciliation protocols developed by Bennett and Brassard [17]. In this scheme, the sender and the receiver calculate the error estimation and agree on block size and seed. They segment their keys into agreed-sized blocks and exchange the two-bit parity of each of their blocks. When there is a parity mismatch, they perform a binary search of the corresponding block to find a single-bit error. After the search, the error is detected and resolved, which is the first complete pass of the protocol. In each succeeding pass, they double the block size and repeat the same process [17]. It is a simple protocol which is computationally efficient and involves frequent interaction between the sender and receiver.

2.1.2. Winnow

Butler et al. [17,21] propose a protocol named Winnow, which significantly reduces the interaction between the sender and receiver. Instead of using binary search, it uses Hamming code to identify and correct single-bit errors. Furthermore, this protocol introduces errors in the key during the process in case of non-uniform error distribution [17,21].

The Winnow segments the primary key into blocks. Before segmentation, the sender and receiver perform error estimation. Based on the error rate, they agree on the block size in increasing powers of 2, for example, 8, 16, 32, 64, and 128. They exchange and compare the parity of each block. In case the parity does not match, they compare the syndrome deduced by using the Hamming hash function. The number of passes depends on the error rate [17,21].

Hamming Code Structure: A Hamming code follows even parity bits. It detects two-bit errors and resolves single-bit errors. In a Hamming codeword, parity bits are bits with positions of power of 2. The remaining bits are data. For example, when a message with four binary digits uses the Hamming function, it adds three parity bits by giving a (7,4) codeword, as shown in Figure 1.

D7	D6	D5	P4	D3	P2	P1

Figure 1. Hamming code structure (P4, P2 and P1 are parity bits. The rest are data bits).

Hamming Code Algorithm [17,21,22]: To determine the value of parity bits, the sequence of bits is alternatively checked and skipped. For example, suppose a sender sends data of binary digits 1101 to the receiver. For P1, we check and skip 1 bit, and the checked positions are (1, 3, 5, 7, 9). For P2, we check and skip 2 bits, which gives (2, 3; 6, 7; 10, 11) checked positions. For P4, we check and skip 4 bits, which returns the following bit positions: (4, 5, 6, 7; 12, 13, 14, 15).

For P1, the parity for D3, D5, D7 or 101 is 0 because it follows even parity. Since the number of 1s is even, the parity is 0; else, it will be 1. Therefore, P1 is 0. For P2, the parity for D3, D6, D7 or 111 is 1, and thus, P2 is 1. Similarly, the P4 for D5, D6, D7 or 011 is 0. Therefore, the obtained Hamming codeword is 1100110 for the message 1101. At the receiver's side, the bits on positions (1, 3, 5, 7), (2, 3 , 6, 7) and (4, 5, 6, 7) are checked using even parity. If P1, P2 and P4 is 0, then no error is present in the received codeword. If the error is present in any of the parities, the parity value is 1. For example, if P1 and P4 are 1, then the received codeword is wrong. The error word is P4, P2, and P1, which yields 101. The decimal value of 101 is obtained and depicts that the fifth bit of the codeword is incorrect, and thus, it is flipped.

As per the property of Hamming code, it can detect and correct one error per block. If that block contains a large number of errors, it introduces a new error to the block. We can use a smaller block size to avoid this situation. However, it results in a large number of blocks with a consequent increase in the number of parity bits and the amount of information leaked [17].

2.1.3. Low-Density Parity Check (LDPC)

The LDPC uses a parity check matrix H and a generator matrix G. The dimensions of both the matrices is $m \times n$ such that $n \times k = m \times j$ where m is the number of rows and n is the number of columns. The j is the number of 1s in each row, and the k is the number of 1s in each column. G denotes the Generator matrix, and H denotes the parity matrix. The values of j and k are small compared to the number of rows and code length. Therefore, H has a low density of 1s. The H is called a low-density parity check matrix. Moreover, the code defined by H is called a low-density parity check code [17].

In quantum transmission, the error correction code does not use the Generator matrix, and the Parity matrix is perceived as a Tanner graph [17]. In the H matrix, each row is the check node with each column being the variable node. The check node signifies the parity check based on syndrome calculation. The variable node represents the single bits of the message [17].

In a single information exchange, the LDPC resolves all the errors in the transmitted key. Unlike Cascade and Winnow, it does not follow any parity or key segmentation. The sender calculates the syndrome for the key and sends it to the receiver. The receiver calculates the syndrome based on his obtained key. It then uses the sender^{0}s syndrome to detect and resolve the errors in the key. The receiver uses a decoding algorithm to detect the error locations in the key. The most common algorithm used in LDPC is the Sum-Product algorithm [17].

In Winnow, the message segments use the Hamming Code and can only correct single-bit errors. In LDPC, the syndrome is larger. Therefore, it can correct multiple errors with less communication overhead as compared to Winnow and Cascade. However, the computational complexity is significantly higher than that of the other two protocols.

2.2. Error Correction Protocol Proposed for Wireless Networks: Low Complexity Parity Check (LCPC)

SCADA involves RTUs and field devices that are resource constrained. Thus, we conducted a literature review on the error correction schemes for wireless networks. This section focuses on the Low Complexity Parity Check (LCPC) protocol [23], as described as follows.

Low Complexity Parity Check (LCPC) Protocol

Alabady et al. [17,23] proposed a low complexity parity check code for wireless network applications. It has less complexity and requires less memory as compared to that of LDPC. The LDPC performs better in case of a larger codeword and a low density parity matrix. However, it consumes more memory and requires complex decoding and computation [17,23].

The LCPC segments the binary message into equal bits. Each segmented source datum follows the error-correcting algorithm. Thus, it yields a codeword which is sent via the public channel. The receiver checks for errors in the received codeword. If any error is present, it resolves them using syndrome. Finally, it decodes the codeword [23].

The LCPC has the following methods to encode and decode the codeword [23].

Step 1, Segmentation: The source data are segmented into equal blocks. We discuss LCPC (9,4) as an example. It uses 56-bit source data which follows segmentation into 4-bit length blocks. The source code now contains fourteen 4-bit blocks.

Step 2, LCPC encoding: : Each 4-bit block x_i uses LCPC (9,4). In this step, the algorithm generates a codeword using a parity-check matrix and Generator matrix. For example, source data (SD) = x_i is 1010.

For x_i = 1010, each bit is represented as $\beta1$ = 1, $\beta2$ = 0, $\beta3$ = 1, and $\beta4$ = 0. The LCPC follows even parity. The parities are obtained using the following equations [23].

$$
\begin{aligned}
P5 &= \beta1 \oplus \beta2 \oplus \beta3 \oplus \beta4\\
P6 &= \beta1 \oplus \beta2 \oplus \beta3\\
P7 &= \beta1 \oplus \beta2 \oplus \beta4\\
P8 &= \beta1 \oplus \beta3 \oplus \beta4\\
P9 &= \beta2 \oplus \beta3 \oplus \beta4,
\end{aligned}
$$

where ⊕ is the XOR operator symbol.

Thus, the deduced parity bits are P5 = 0, P6 = 0, P7 = 1, P8 = 0 and P9 = 1. When parity bits are added to the x_i, it gives the codeword CD = 101000101. This codeword is sent to the receiver. Furthermore, the Generator matrix G and the Parity check matrix H obtained are as follows.

$$
H = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0\\
1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0\\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0\\
1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0\\
0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

$$
G = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0\\
0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & & \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & & & & \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & &
\end{bmatrix}
$$

Step 3, LCPC Decoding: The receiver obtains the codeword with or without error. It performs three sub-steps. Sub-step1: It detects any error present in the codeword. Sub-step2: If an error is detected, it checks and determines the error pattern. Sub-step3: Finally, it corrects the error and decodes the codeword. Let the codeword c be transmitted, and vector r is the received codeword such that it satisfies the following equation.

$$r = c + e \tag{1}$$

where e is the error vector.

The technique uses the syndrome vector for error detection and correction. A syndrome vector indicates whether the equation is satisfied for that particular codeword. If the value of syndrome s is zero, it denotes that no error has occurred, and the received codeword r is the correct one (c). Otherwise, it detects that the codeword is with an error.

If H is the parity-check matrix of c, then,

$$Hr^T = H(c + e)^T = Hc^T + Hr^T \tag{2}$$

$$Hc^T = 0 \tag{3}$$

$$Hr^T = He^T = s \tag{4}$$

Hr^T is the syndrome of r.

When the syndrome has a non-zero value, the column of the H matrix which is a scalar multiple of the s is searched. If no such column is found, the code contains more than one error and it fails to correct multiple errors. Otherwise, if the syndrome is α times that particular column j, then the vector is added with $-\alpha$ on the jth bit-position and with 0 on the rest bit-positions.

After the error correction, the receiver decodes the codeword using a masking process on its last left four bits. It involves using the AND operator between the corrected codeword and 111100000.

Although LCPC is more efficient in respect to computational cost and memory requirement as compared to other protocols, it fails to correct burst errors. In quantum transmission, there are significant chances of burst errors in the quantum key. Cascade and Winnow protocols can correct single-bit errors, and LDPC can correct multiple bit errors.

2.3. Error Correction Protocol Used in the Proposed Security Scheme

This section provides a brief overview of Reed–Solomon (R-S) protocol which is further explained in Section 4.1.3. In our proposed security scheme, we have used the R-S protocol for error correction to protect against partial or tampered data [24]. Due to the high computational and complexity cost, LDPC requires and exhausts more hardware resource which hampers the system applications. Reed–Solomon is a multi-bit error-correcting protocol with a low computational overhead as compared to LDPC [24].

2.4. Post-Quantum Digital Signature: One-Time Digital Signature

This section focuses on the one-time digital signature (OTS) scheme. It is based on a hash function that signs one message per key pair. A generic OTS is a set of three algorithms, namely, (1) Generating one-time key pairs that include public and private keys, (2) Obtaining a one-time signature, and (3) Verifying the obtained signature [20]. One of the well-known OTS schemes is the Lamport–Diffie scheme where the key used for signing (sk) is randomly generated, and the verification key (vk) is generated by applying a hash function on the sk [25–27]. This feature of Lamport Signature scheme has been used in the proposed scheme to generate a signcrypted message.

3. Related Work and Possible Attacks on SCADA Networks

In Section 2, we discussed the background work that is required to understand the fundamentals of the proposed scheme. In this section, we will discuss the current standards and protocols proposed and widely used in SCADA systems. We also discuss the possible attacks that can be launched based on quantum computing on SCADA networks.

SCADA is one of the critical infrastructures that collects real-time data and controls industrial processes. In recent years, several successful attacks have been launched on SCADA networks that have been a wake-up call for implementing updated security protocols. One of the major attacks on SCADA systems was the Stuxnet worm [28] that propagated using USB drives or network connections. It presented to the infected host as one of the SCADA systems and performed a man-in-the-middle attack. Thus, it led to a violation of data integrity and confidentiality. The main goals of a man-in-the-middle attack on SCADA systems are to disable the SCADA systems, exchange false or improper control commands, sabotage the PLC commands, and gain sensitive information in industrial processes [28].

The US Department of Homeland Security's (DHS) Industrial Control Systems Cyber Emergency Team (ICS-CERT) has released various guidelines to mitigate the malware such as Stuxnet. One of the guidelines involved the requirement of an algorithm with adequate entropy to generate sufficiently random keys [29]. Our proposed algorithm used Quantum Key Distribution that generates truly random keys. Furthermore, various researchers have proposed multiple mitigation strategies involving host-based and network-based intrusion detection systems. For example, Carcano et al. [30] proposed an IDS that monitors the network traffic based on state anomalies. Furthermore, Ponomarev et al. [31] proposed hardware fingerprinting to detect SSH host spoofing, and the National Institute of Standards and Technology has proposed attack mitigation techniques based on firewalls to defend against man-in-the-middle attacks.

There have been significant research work and organizational efforts to protect SCADA network from cyber-attacks [4]. We classified the existing security schemes into three: current standards used for SCADA networks, detection of SCADA attacks and prevention of SCADA attacks. All of the existing protocols and security schemes use traditional cryptography. Even the current standards, including IEC 62351 and AGA-12, are based on traditional algorithms. It makes them vulnerable to attacks from a quantum computer.

Furthermore, existing public key algorithms tend to increase computational and time cost [4].

3.1. Possible Attacks on SCADA Networks

In this paper, we identify the following primary attacks on SCADA networks: (1) traditional man-in-the-middle attack (2) brute-force with quantum computer. Table 2 summarizes the attacks on SCADA networks.

Table 2. Attacks on SCADA systems.

CLASSICAL ATTACKS	**Description**	
Attack against Confidentiality	Packet Sniffing	The intruder intercepts the incoming and outgoing traffic in a network and fetches sensitive information by decoding the data packets. By using Wireshark and Tcpdump, sniffing can be attained.
	Eavesdrop	The intruder can install an eavesdropping equipment in the wired or wireless network between the RTU and MTU. The ongoing conversations can be wiretapped. Tools that can be used include Wireshark and dnsiff.
Attack against Integrity	Man-in-the-middle attack (MiM)	In an MiM attack, the intruder monitors the traffic between the two nodes. The data packets traded between two victim nodes are captured. The intruder then injects abnormal data during the transmission and sends it to the receiver. It can launch IP spoofing and a Session Hijacking attack. A few tools that are used to launch MiM attack are Ettercap, SSLStrip and Evilgrade.
	Session Hijacking	After a successful MiM attack, the intruder accesses the information and services in the MTU and RTU. It accesses the session ID and launches a replay attack. A few examples of tools are Ettercap and Evilgrade.
	Data Injection	The intruder can successfully alter the data after launching an MiM attack. A few tools that can be used are Wireshark and Ettercap.
	Replay Attack	The attacker can launch a replay attack by performing session hijacking and IP spoofing. By imitating as a friendly unit and using the session ID, it stores the old data and sends it to other units later. Tools that can be used are Ettercap and Evilgrade.
Attack against Authentication	Masquerade	By using IP spoofing, the attacker uses a fake identity to pretend as a original unit and steals essential data from the system or the network. For example, it can fetch passwords and gain access to the system. Tools that can be used for launching this type of attack are Ettercap, Arpspoof and Brutus.
Attack against Availability	Denial of Service (DoS)	This kind of attack occurs when a compromised unit is used to target a system by sending huge traffic or a large amount of junk data. A unit can be compromised in several ways after a successful MiM attack. The examples of DoS attack tools are Slowloris and GoldenEye.
QUANTUM ATTACK		
Quantum Attack	Brute Force Attack by a Quantum Computer	The emergence of the quantum computer brings with it benefits as well as risks to the cyber field. A quantum computer is way faster and more efficient than traditional computers. Using Shor's and Grover's algorithm, a quantum computer can launch a brute force attack and crack the traditional encryption schemes in a brief time. One such problem is elliptic curve cryptography (ECC or ECDSA).

3.1.1. Man-in-the-Middle Attack

In our previous paper [4], we have discussed standard attacks on SCADA networks. An attacker can launch a man-in-the-middle attack between two victims to fetch the information passing over the communication channel between the RTU and the MTU by eavesdropping and capturing packets. By using tools such as Ettercap and Evilgrade [32], the man-in-the-middle attack can be extended to other attacks such as data injection [4].

3.1.2. Brute-Force Attack Using Quantum Computer

In recent years, the rapid development of quantum computers has posed a threat to cybersecurity. A quantum computer can solve and crack mathematical operations, for example, the problem of factoring enormous numbers, which is the core of any encryption scheme. AGA-12 uses RSA-1024 bit as the key management protocol and AES as the encryption scheme. In [4], we discussed that the two quantum algorithms, namely, Shor's algorithm and Grover's algorithm, can theoretically crack RSA and AES schemes, respectively. Since Grover's algorithm provides a square root acceleration over the classical search algorithm, the AES-256 offers an adequate key strength of 64 bits [9]. Therefore, Grover's algorithm has weakened AES but not broken it. However, recently, Gidney et al. [33] practically proved that the Shor's algorithm cracks RSA-2048 within 8 h with 20 million qubits. Therefore, we believe that a quantum computer can successfully launch a brute force attack on AGA-12 standard to crack the RSA protocol.

Modern SCADA networks rely on Internet connectivity, cloud computing, and wireless communications. These have made its infrastructure susceptible to various attacks. Mostly, the man-in-the-middle (MiM) attack is the source of every other attack. Thus, the proposed quantum-resistant security scheme mainly focuses on the detection and prevention of the MiM attack and the brute force attack by Shor's algorithm. It also provides security against the attacks discussed [4].

4. Proposed Security Scheme

In a generalized signature-then-encryption scheme, a digital signature scheme is initially applied, and a private key is used to encrypt the plain text and the digital signature together [19,34]. However, a generalized signcryption involves using a secret key that is obtained from the public key of the receiver. The secret key is used to encrypt the message and obtain the cipher. Simultaneously, it uses the private key of the sender to generate a digital signature. The cipher and the signature are sent to the receiver [19,34].

However, in our proposed scheme, we do not follow the above-mentioned signature-then-encryption scheme steps. Instead, we follow the generalized steps of a signcryption scheme. However, in place of the public key, we use a quantum key, which is secret as well as shared between the sender and receiver. The private key is obtained from the quantum key to generate the digital signature. Simultaneously, the quantum key encrypts the message to generate a cipher.

To the best of our knowledge, we believe that our proposed scheme is the first solution for SCADA networks against quantum computing.

The current security schemes in SCADA networks use key management and authentication protocol that are weak against quantum algorithms. In this paper, we propose a new scheme to guard the communication channel between RTU and MTU from quantum as well as traditional attacks. Moreover, Fröhlich et al. [35] demonstrated that BB84 protocol can provide successful key distribution over distances up to 240 km.

In our proposed scheme, we assume the following:

- MTU has the identities and hashed IDs of all RTUs.
- The ID of MTU is embedded in each and every RTU.
- The RTU and MTU are aware of hash functions used to generate the private key.
- The data stored in the legitimate units are secure.
- The distance between the RTU and the MTU is maximum 200 km.

Sibson et al. [11] have developed a chip-based QKD in 2015. This evolution of Quantum Key Distribution (QKD) has motivated us to propose a quantum-based signcryption scheme for SCADA networks since they can be deployed in RTUs as well. However, both RTU and MTU need a few hardware changes. There is a need to amalgamate a monolithically integrated transmitter and a receiver with a photonic circuit using thermo-optic phase shifters in the RTU as well as in the MTU.

Quantum Cryptography exploits Heisenberg's Uncertainty Principle [14] and the Principle of Photon Polarization [22]. According to Heisenberg^{0}s Uncertainty Principle, it is not possible to obtain the position and momentum of a photon with absolute accuracy [14]. The Principle of Photon Polarization states that a photon can have a superposition of two or more quantum states at a time [15]. Furthermore, the No-Cloning Theorem [16] states that one can not produce an identical copy of an arbitrary quantum state of a photon. It makes quantum cryptography a feasible scheme to resist the threats of both a quantum and traditional computer. BB84 protocol is a popular Quantum Key Distribution (QKD) protocol and is the most suitable for IoT applications [36–38]. Our proposed scheme uses BB84 and has three main phases:

Phase A: Quantum Key Distribution;
Phase B: Signcryption;
Phase C: Un-Signcryption.

4.1. Quantum Key Distribution: BB84 Protocol

This phase applies the BB84 protocol to generate a final quantum key [39] for signcryption. The Quantum Key Distribution protocol uses basis to generate and measure a qubit state.

A standard or canonical basis is denoted by state $|0>, \ldots, |n-1>$ which represents the following state [18].

$$|0> = \begin{bmatrix} 1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{bmatrix} \ldots\ldots |n-1> = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix}$$

The normalized sum of all the standard basis is a diagonal basis vector. Since BB84 uses 45-degree and 135-degree polarization for a diagonal basis, the value of n is 2, the diagonal state $|D> = |0>, |1>$ is represented in the following equation [18].

$$|D> = \frac{|0> + |1>}{\sqrt{2}} \tag{5}$$

The bases are used to generate qubits with superposed states. It uses two bases: horizontal–vertical linear and diagonal directions. The key generation process uses the polarization of light. Each photon is polarized using one of the two bases randomly. The protocol employs two channels: a quantum channel for key generation and distribution, and classical channel for information transmission and eavesdrop detection. This phase has the following further steps:

- Quantum Key Generation;
- Key Sifting;
- Error Correction;
- Privacy Amplification.

4.1.1. Quantum Key Generation

The sender generates the first qubits by randomly using one of the bases and sends it to the receiver via the quantum channel. For example, RTU acts as the sender and MTU acts as the receiver, as shown in Figure 2. The series of first qubits is called raw bits [13,36].

The MTU reads each qubit with either of the two bases randomly and independently. The series of qubits received by the MTU is called the raw key. There are two cases of measuring the raw bits as following.

- Case 1: The receiver has a 50% success rate of choosing the right basis to measure the bits and thus getting the correct bits.
- Case 2: The receiver has a 50% failure rate where it selects the wrong basis. However, the outcome of using the wrong machine is random, which is either 0 or 1. Thus,

the probability of incorrect bits in the received bits is 25%, and that of correct bits is 75%. This ratio persists in the absence of any eavesdropper [13,36].

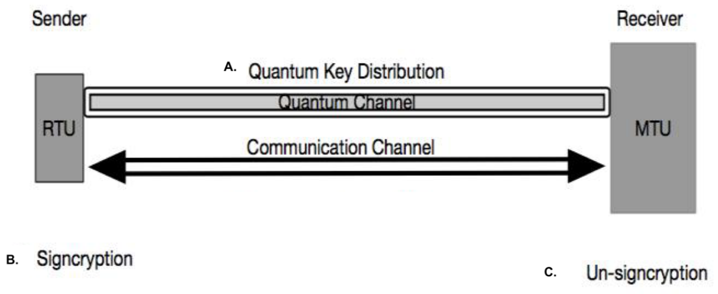


Figure 2. The proposed scheme model.

When the qubits are measured using any basis, their state changes randomly. Furthermore, the states of the qubits cannot be cloned, which helps in the detection of an intruder. When an eavesdropper tries to read the qubits in the quantum channel, it disrupts the state of the qubits. Thus, the MTU receives the disrupted qubits. The MTU measures the tampered raw key, and the rate of incorrect qubits exceeds 25% [13,36].

4.1.2. Key Sifting

The MTU sends the randomly chosen basis to the RTU via the public channel. The RTU verifies its chosen basis with that of MTUs. Then, the RTU sends the incorrect basis to the MTU. Both the units discard the bits measured by the incorrect basis and obtain the sifted key. In case of no noise in the quantum channel, the sifted key of both the units is the same. In case of any presence of noise, there is an error in the sifted key deduced by MTU [13].

4.1.3. Error Correction Protocol

The error correction protocol in the quantum key distribution has the following substeps [13,36].

Step 1: Determine Quantum Bit Error Rate (QBER). A Quantum Bit Error Rate (QBER) is the fraction of mismatched qubits exchanged between the sender and the receiver [13]. The RTU calculates the QBER. The RTU and MTU randomly extract a part of its sifted key. The MTU discloses its extracted part to RTU. The RTU obtains the QBER by calculating the ratio of error and total number of bits in MTU's extracted key. Both of the units discard the exposed part and obtain the sub-sifted key.

- Case 1: If QBER is higher than 25%, both units discard the sifted key, and it generates the raw key again.
- Case 2: If QBER is less than 25%, the units follow the error correction protocol and privacy amplification.

Step 2: Error Correction Protocol (ECP) used: Reed Solomon Code. In this phase, the sender and the receiver resolve the error in the sub-sifted key via the public channel. The error correction protocol phase is crucial after a quantum key exchange for the following reasons.

- It helps both the units check the confidentiality and integrity of the obtained sub-sifted key.
- The RTU sends its sub-sifted key encoding it with ECP protocol to MTU. The encoded key is called the codeword. The encoding involves adding extra bits or parity bits to the original data. It helps the receiver to detect and resolve the errors. Therefore, the eavesdropper is unable to read the original key. When the codeword is modified, it is detected as well as resolved by the MTU.
- In this phase, based on the QBER, the sub-sifted key is corrected as the errors are reconciled.

As mentioned in Section 2, the most common error correction protocols are the Cascade protocol, Winnow protocol, Low-Density Parity Check (LDPC) protocol, Low Complexity Parity Check (LCPC) and the Reed–Solomon protocol (R-S) [17,23,24].

The most feasible protocol for wireless networks are the LCPC and Reed–Solomon protocols. The purpose of LCPC is to detect and correct single- and double-bit errors. However, during the quantum key exchange, the errors can occur in bursts. In that scenario, as mentioned in Section 2, the Reed–Solomon (R-S) code is a better suited protocol for implementing with the BB84 protocol. The R-S code is an efficient algebraic code which can correct a large number of errors with low overhead and low complexity. It has the power to correct errors in a cluster. Various storage systems, broadcast systems, and wireless networks widely adopts R-S code.

Characteristic of Reed–Solomon (R-S) code: It is a subgroup of Bose–Chaudhuri–Hocquenghem (BCH) codes [24] and linear codes which performs their arithmetic operations in a Galois field or finite field. BCH is a cyclic error-correcting code that involves using polynomials over data blocks. The code word generated in this algorithm consists of polynomials, which is divisible by another fixed short-length polynomial. The fixed polynomial is called a Generator polynomial [40].

A Reed–Solomon code is represented as R-S(nk) with s-bit symbols. It implies that the encoder takes k data symbols with s bits each. Then, it adds parity symbols, thus obtaining a code word of n symbols. The parity symbols of s bits each are n − k. The R-S decoder can resolve up to t symbol errors in a codeword. It implies that it can automatically correct errors up to t bytes. The length of parity is calculated as follows [24,40]:

$$2t = n - k \tag{6}$$

The maximum codeword length (n) can be calculated as follows:

$$n = 2^s - 1 \tag{7}$$

R-S Encoder: In R-S encoding, the sub-sifted key is the message which is represented as a polynomial i(x). The polynomial is multiplied with the Generator polynomial g(x) [24,40].

$$c(x) = g(x)\ i(x) \tag{8}$$

where

$c(x)$ is the valid codeword.
$i(x)$ is the information block.
$g(x)$ is the generator polynomial.

Using Lagrange interpolation, the polynomial is evaluated as:

$$p(x) = i(x)\ x^{n-k} \mod g(x) \tag{9}$$

R-S Decoder: An error occurs when an incorrect bit is present in the codeword. An erasure occurs when the position of the incorrect bit is known [24,40]. The decoding algorithm uses the following outlined process to correct up to t errors or up to 2t erasures.

The received codeword can be represented as follows:

$$r(x) = c(x) + e(x) \tag{10}$$

where r(x) is the received codeword, c(x) is the recovered codeword and e(x) is the error pattern present in the r(x).

The decoder follows the succeeding steps.

1. Syndrome calculator: It calculates the syndrome which is used to identify the symbol errors. One symbol error occurs when either 1 bit is incorrect or all the bits are incorrect in a symbol.

2. Error locator: It then finds the symbol error locations by calculating the error locator polynomial. It uses Euclid's algorithm.
3. Calculate magnitude of error: Then, it finds the roots of the error locator polynomial.
4. Error evaluation: To calculate the symbol error values, the Forney algorithm is used.

   Finally, a recovered codeword is received.

4.1.4. Privacy Amplification

From the received and recovered codeword, the MTU extracts the sub-sifted key. To reduce any information leakage during error correction protocol and to increase the secrecy of the key, the MTU hashes the sub-sifted key. Both MTU and RTU obtain the finalized key or quantum key (QK).

4.2. Signcryption

Both the components have the finalized quantum key (QK). The RTU executes the following steps [41] as shown in Figure 3.

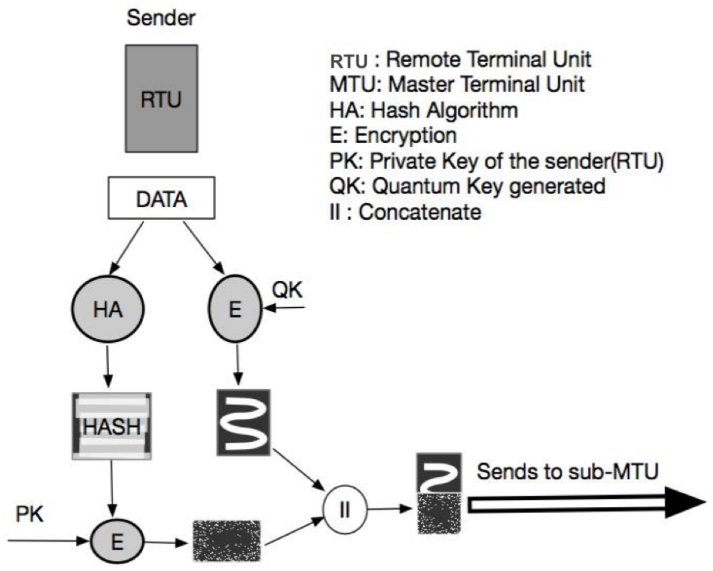

Sender
RTU
RTU : Remote Terminal Unit
MTU: Master Terminal Unit
HA: Hash Algorithm
E: Encryption
PK: Private Key of the sender(RTU)
QK: Quantum Key generated
II : Concatenate
DATA
HA
E
QK
HASH
PK
E
II
Sends to sub-MTU

Figure 3. Operations of RTU (sender). The signcrypted message is sent to sub-MTU/MTU (receiver).

- Encryption: The RTU makes a copy of the data and encrypts the data with the finalized quantum key.
- One-Time Digital Signature: The RTU hashes the copy of the data. It then encrypts the hash with its private key (PK). It segments the quantum key into equal chunks. It then generates a private key by applying a hash function on one of the segments of the QK. It concatenates the hashed message, hashed unique ID of the RTU and a timestamp. The PK is used to encrypt the concatenated data, thus generating a one-time digital signature.

  The RTU sends the signcrypted data to the MTU over the classical channel.

4.3. Un-Signcryption

The MTU receives the signcrypted data and executes the following steps. Furthermore, we assume that the MTU has the database which stores the information, including IDs of all the RTUs and their hash values. The MTU is also aware of the signcryption algorithm used by the RTU.

- Decryption: The MTU decrypts the encrypted data with the quantum key (QK).

- Validation: The MTU also decrypts the encrypted hashed value with the private key (PK). The MTU hashes the copied data by the same algorithm used by the RTU. Thus, the timestamp and the hashed ID is extracted and verified.

5. Formal Analysis of Proposed Model

The proposed scheme has two major parts, quantum and classical. We have performed the experiments on macOS with 1.8 GHz Intel Core i5 processor manufcatured by Apple Inc. and sourced from Halifax, Canada. The quantum part involves the BB84 protocol, and the classical part involves the Signcryption algorithm. Therefore, in this paper, we have used two tools for the formal analysis of the proposed scheme.

- Modeling and Analysis of BB84 protocol in Prism.
- Modeling and Analysis of Signcryption in Scyther.

Our proposed scheme is simulated in PRISM and Scyther for the specification, analysis, and verification of our security protocol. First, we verify the Quantum Key Distribution protocol on PRISM. We created the BB84 model that contains four modules: (i) Alice refers to RTU, (ii) Bob refers to MTU, (iii) Eve refers to the eavesdropper, and (iv) the quantum channel. In the simulation, we calculate the probability of detecting Eve and the extent of information leakage. After obtaining the quantum key, we model and analyze the signcryption in Scyther. We formalize the security requirements of our signcryption model based on the discrete logarithm problem (DLP) in Scyther. We claim specified security requirements, namely, secrecy of the key, data exchanged, and aliveness of the two parties symbolizing authentication. After the formal analysis of our proposed scheme, we implemented our algorithm on Python 3.6, involving three parties: RTU as the sender, MTU as the receiver, and Eve as the eavesdropper. We have also simulated the quantum channel, characterizing the noise of binary symmetric channel, on Python 3.6 with the help of a Quantum Information Toolkit (QIT).

5.1. Modeling and Analysis of Quantum Phase (BB84 Protocol) in Prism

The formal analysis of the security of Quantum Key Distribution has been completed by PRISM. However, ensuring provable practical security and breaching time analysis of the proposed scheme by considering the quantum adversary will require extensive quantum hardware or a quantum computer. This is planned for future work.

Prism is a probabilistic model checker to model and to analyze the systems based on probabilistic behavior. It automatically investigates the systems to find out flaws and errors in the system specification. The model checker feeds the following two types of inputs [42].

- The description of the to-be-designed system. It mostly expresses the information in process algebras such that it acts as an input in model checker.
- A set of rules or properties that the system must follow.

There are two stages to build a model in a Prism [42]:

Stage 1: Model the system where it represents all the states and transitions of the system.
Stage 2: Model the system where it expresses its properties in temporal logic statements.

When we execute the temporal logic statements against the model, it verifies whether and with what probability the properties hold for the system.

In this paper, the Discrete-Time Markov Chain (DTMC) model has been used to design the BB84 protocol system. While building the system, the following two properties of the system are defined:

- Public channel handles the transmission of messages in such a way that the system monitors the process. However, the eavesdropper is unable to monitor the messages.
- Quantum channel handles message exchange in such a way that any attempt by an eavesdropper to monitor the channel causes an alteration in the message and thus creates a noise.

Thus, the system detects any eavesdropping attack as well as cloning attack.

The following two types of attacks have been tested for this system [43].

1. Intercept–Resend Attack: The eavesdropper uses the basis once to measure the qubit. It measures a qubit, and the state of the qubit changes randomly.
2. Random–Substitute Attack: The eavesdropper uses the basis twice. At first, it uses the basis to measure the qubit. After fetching the value of the qubit, it reads the same qubit again to replace its value. It is an attempt to clone the state of the qubit.

In this paper, we analyze the following three factors of the BB84 protocol:

- Whether the protocol detects any intrusion;
- How much information is leaked processing the protocol;
- Can BB84 protocol discard or prevent the eavesdropping attack.

The following six variables have been calculated and used in the models [43]:

- P1 = Probability of detecting an eavesdropper (EVE);
- P2 = Probability that EVE measures more than half of the information correctly;
- N = No. of bits transferred;
- Correct bits measured by Eve $\geq$ N/2;
- L = LUCKY = Probability of obtaining correct value with wrong basis;
- REPLACE = 0.5 = Probability of substituting with 0 or 1.

In this model, the probability value ranges from 0 to 1. Based on the type of attacks, there are two following major threat models.

- Model 1: BB84 with intercept–resend eavesdropping attack;
- Model 2: BB84 with random-substitute eavesdropping attack.

Tables 3 and 4 show the probability of detecting the intruder launching eavesdropping attack or MiM attack. It also shows the probability of high information leakage.

Tables 5 and 6 display the detection rate of the intruder attempting to clone the data or qubits. Simultaneously, it provides the probability of maximum information leakage.

Table 3. Probability of detecting of Intercept–Resend eavesdropping when LUCKY is 0.5.

MODEL 1	**P1**	**P2**
N = 4	0.938	0.145
N = 5	0.969	0.155
N = 6	0.984	0.065
N = 7	0.992	0.067
N = 8	0.996	0.028
	LUCKY = 0.5	

Table 4. Probability of detecting of Intercept–Resend eavesdropping when N is 5.

MODEL 1	**P1**	**P2**
L= 0.5	0.968	0.155
L = 0.6	0.968	0.174
L = 0.7	0.968	0.193
L= 0.8	0.968	0.285
	N = 5	

Table 5. Probability of detecting of Random-Substitute eavesdropping when Lucky is 0.5.

MODEL 1	**P1**	**P2**
N = 4	0.938	0.145
N = 5	0.969	0.155
N = 6	0.984	0.065
N = 7	0.992	0.067
N = 8	0.996	0.028
	LUCKY = 0.5	

Table 6. Probability of detecting of Random-Substitute eavesdropping when N is 5.

MODEL 1	**P1**	**P2**
L= 0.5	0.969	0.155
L = 0.6	0.969	0.174
L = 0.7	0.969	0.193
L= 0.8	0.969	0.285
	N = 5	

5.2. Modeling and Analysis of Classical Phase in Scyther

In this section, formal analysis of the classical part in the proposed signcryption scheme is presented which involves generating and exchanging encrypted data and digital signature altogether. We have used Scyther for the formal analysis of the non-quantum part of the proposed scheme. One of the properties we are testing is the secrecy of the quantum key and how it is affecting the secrecy of the encrypted data transferred.

Scyther is a tool that verifies traditional security and authentication protocols [44]. Using Scyther, two main properties are analyzed: Secrecy and Authentication [44].

Secrecy: The following assumptions are made [44]:

- The sender or the receiver is communicating with a trusted party.
- The sender and the receiver are communicating over an untrusted channel.

Authentication: The four factors that are assumed for the system are as follows [44]:

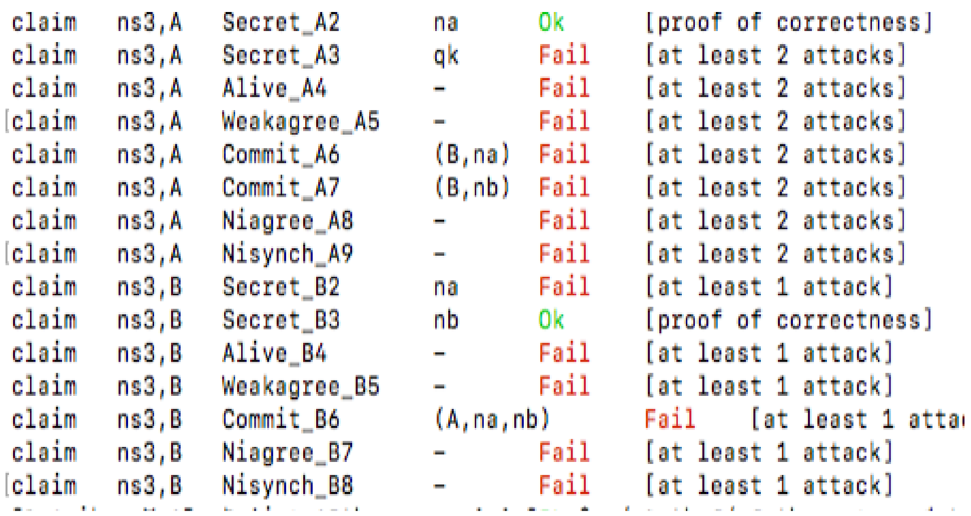
- Aliveness: There is at least one communication partner in the network.
- Synchronization: The intended party is aware of the authenticity of the other party to which it is communicating with.
- The protocol is executing.
- Message Agreement: The message sent by the sender is intact and not tampered. Thus, it has been exchanged as expected.

Furthermore, in the proposed signcryption model, we have used two keys. One, the quantum key is denoted as qk. The qk is used for the encryption of messages. Two, sk denotes the private key in the model. It is used to generate the digital signature. It provides authentication to the scheme. Both qk and sk are secret and private. Figures 4 and 5 exhibit the verification results of a simple authentication protocol and the proposed signcryption scheme, respectively.

With Scyther as the security analysis tool, we have conducted a formal analysis of the signcrypted communication with the quantum and private keys. We have performed verification for the classical phase of the proposed scheme to provide motivation to move forward with the implementation phase. Figure 5 provides the result generated by Scyther when the quantum key is secret and not public. As per the result, the protocol success-fully claims and passes the tests for security properties, which are mainly based on the following [45].

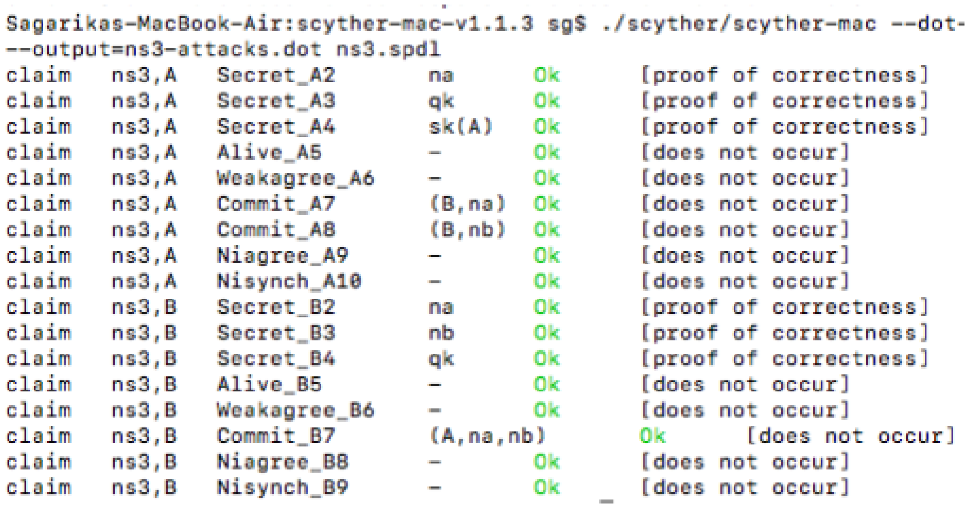
- Secrecy of the keys and the cipher.
- Commitment and aliveness of the two parties.
- Synchronization of the communication between two parties.
- Weak agreement property tests spoofing or man-in-the-middle (MiM) attack between the two parties. A weak agreement between two roles means there is no third-party spoofing or launching a MiM attack [45].

We performed verification for the classical phase of the proposed scheme to provide motivation to move forward toward the implementation phase.



```
claim   ns3,A   Secret_A2      na      Ok     [proof of correctness]
claim   ns3,A   Secret_A3      qk      Fail   [at least 2 attacks]
claim   ns3,A   Alive_A4       -       Fail   [at least 2 attacks]
[claim  ns3,A   Weakagree_A5   -       Fail   [at least 2 attacks]
claim   ns3,A   Commit_A6      (B,na)  Fail   [at least 2 attacks]
claim   ns3,A   Commit_A7      (B,nb)  Fail   [at least 2 attacks]
claim   ns3,A   Niagree_A8     -       Fail   [at least 2 attacks]
[claim  ns3,A   Nisynch_A9     -       Fail   [at least 2 attacks]
claim   ns3,B   Secret_B2      na      Fail   [at least 1 attack]
claim   ns3,B   Secret_B3      nb      Ok     [proof of correctness]
claim   ns3,B   Alive_B4       -       Fail   [at least 1 attack]
claim   ns3,B   Weakagree_B5   -       Fail   [at least 1 attack]
claim   ns3,B   Commit_B6      (A,na,nb)      Fail    [at least 1 atta
claim   ns3,B   Niagree_B7     -       Fail   [at least 1 attack]
[claim  ns3,B   Nisynch_B8     -       Fail   [at least 1 attack]
```

Figure 4. Verification results of a simple authentication protocol.



```
Sagarikas-MacBook-Air:scyther-mac-v1.1.3 sg$ ./scyther/scyther-mac --dot-
--output=ns3-attacks.dot ns3.spdl
claim   ns3,A   Secret_A2      na      Ok     [proof of correctness]
claim   ns3,A   Secret_A3      qk      Ok     [proof of correctness]
claim   ns3,A   Secret_A4      sk(A)   Ok     [proof of correctness]
claim   ns3,A   Alive_A5       -       Ok     [does not occur]
claim   ns3,A   Weakagree_A6   -       Ok     [does not occur]
claim   ns3,A   Commit_A7      (B,na)  Ok     [does not occur]
claim   ns3,A   Commit_A8      (B,nb)  Ok     [does not occur]
claim   ns3,A   Niagree_A9     -       Ok     [does not occur]
claim   ns3,A   Nisynch_A10    -       Ok     [does not occur]
claim   ns3,B   Secret_B2      na      Ok     [proof of correctness]
claim   ns3,B   Secret_B3      nb      Ok     [proof of correctness]
claim   ns3,B   Secret_B4      qk      Ok     [proof of correctness]
claim   ns3,B   Alive_B5       -       Ok     [does not occur]
claim   ns3,B   Weakagree_B6   -       Ok     [does not occur]
claim   ns3,B   Commit_B7      (A,na,nb)      Ok      [does not occur]
claim   ns3,B   Niagree_B8     -       Ok     [does not occur]
claim   ns3,B   Nisynch_B9     -       Ok     [does not occur]
```

Figure 5. Verification results of the proposed Signcryption scheme.

6. Experimental Results

We executed the proposed scheme in Python 3.6. We simulated the quantum channel with noise by designing a binary symmetric channel (BSC). In BSC, the sender sends a bit with value being either 0 or 1. The receiver receives that bit. However, there is a small probability that the bit is flipped in the channel [46].

To generate qubits and to measure their state, the Quantum Information Toolkit (QIT) has been used [47]. To implement the basis, two types of operators, namely, Pauli X operator and Hadamard operator, have been used [48].

Pauli X operator: It acts on a single qubit and flips its gate. It maps |0 > to |1 > and |1 > to |0 >.

Hadamard Operator: It provides the property of the Hadamard quantum gate. When it applies on a qubit with state |0 > or |1 >, there is an equal probability that the outcome state is either |0 > or |1 >. Furthermore, if the Hadamard gate applies twice on the same qubit, the final state is always the same as the initial state.

6.1. Comparative Analysis between 128-Bit BB84 vs. 256-BB84 Protocol

In evaluation testing, the simulation splits into two groups:

Group1: It involves performing the proposed scheme on 128-bit initial or raw key.
Group2: It involves performing the proposed scheme on 256-bit initial or raw key.

The simulation parameters of each group are as follows:

- Error rate;
- Sifted key size;
- Final key size;
- Execution time;
- Digital signature size;
- Time to generate a raw key.

Figure 6 illustrates the behavior of the scheme with 128-bit and 256-bit raw keys in terms of error rate. The coefficient of variation of this parameter for the 128-bit raw key is 0.507, and that of 256-bit is 0.502.

Figure 7 shows the behavior of the scheme with 128-bit and 256-bit raw keys in terms of sifted key size. The coefficient of variation of this parameter for the 128-bit and 256-bit is 0.082 and 0.062.

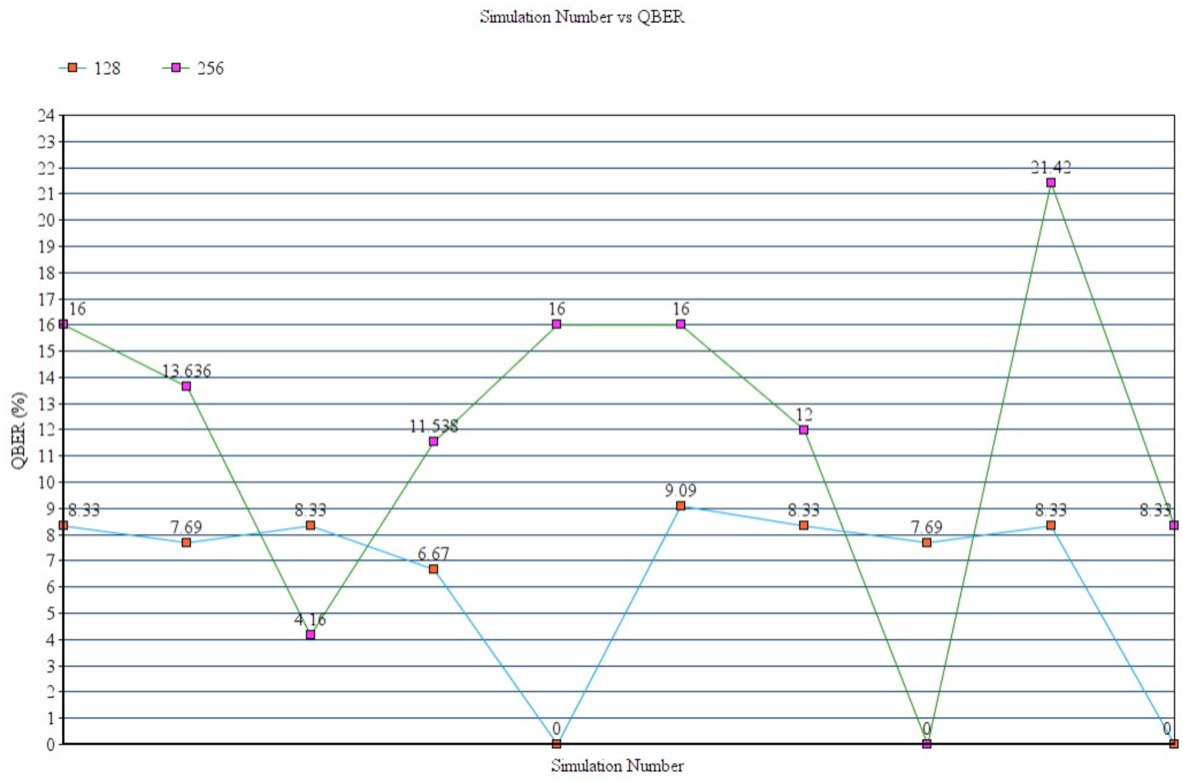


Figure 6. Simulation number vs. QBER.

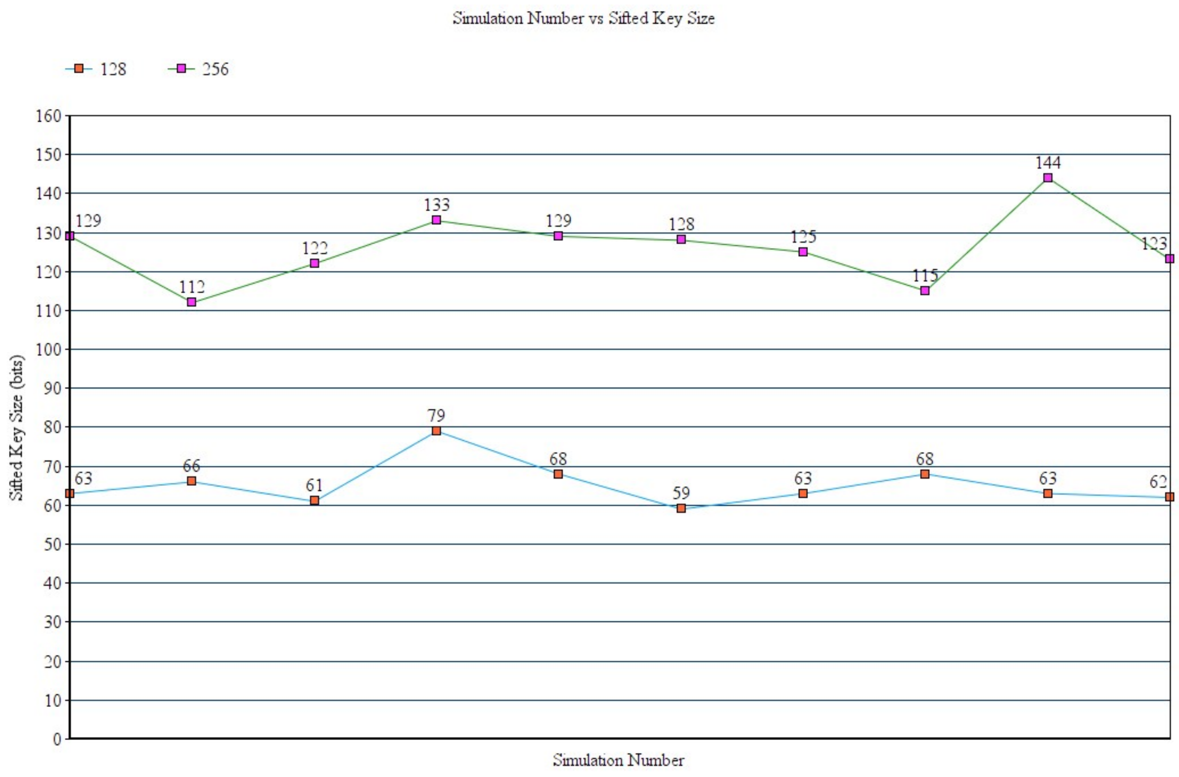


Figure 7. Simulation number vs. sifted key size.

Figure 8 displays the behavior of the scheme with 128-bit and 256-bit raw keys in terms of final key size. The coefficient of variation of this parameter for the 128-bit and 256-bit is 0.024 and 0.018.

Figure 9 illustrates the behavior of the scheme with 128-bit and 256-bit raw keys in terms of digital signature size. The coefficient of variation of this parameter for the 128-bit and 256-bit is 0.0065 and 0.009.

Figure 10 explains the behavior of the scheme with 128-bit and 256-bit raw keys in terms of execution time.

Figure 11 displays the time to generate the raw keys or the initial keys. We observe that the generation time of the former is directly proportional to the generation time of the latter. It is more evident when we compare the mean values of the generation time of each group, as shown in Figure 12. For each data communication, the sender generates a new raw key and obtains the private and secret key. Therefore, the private key and secret key randomly vary in each communication.

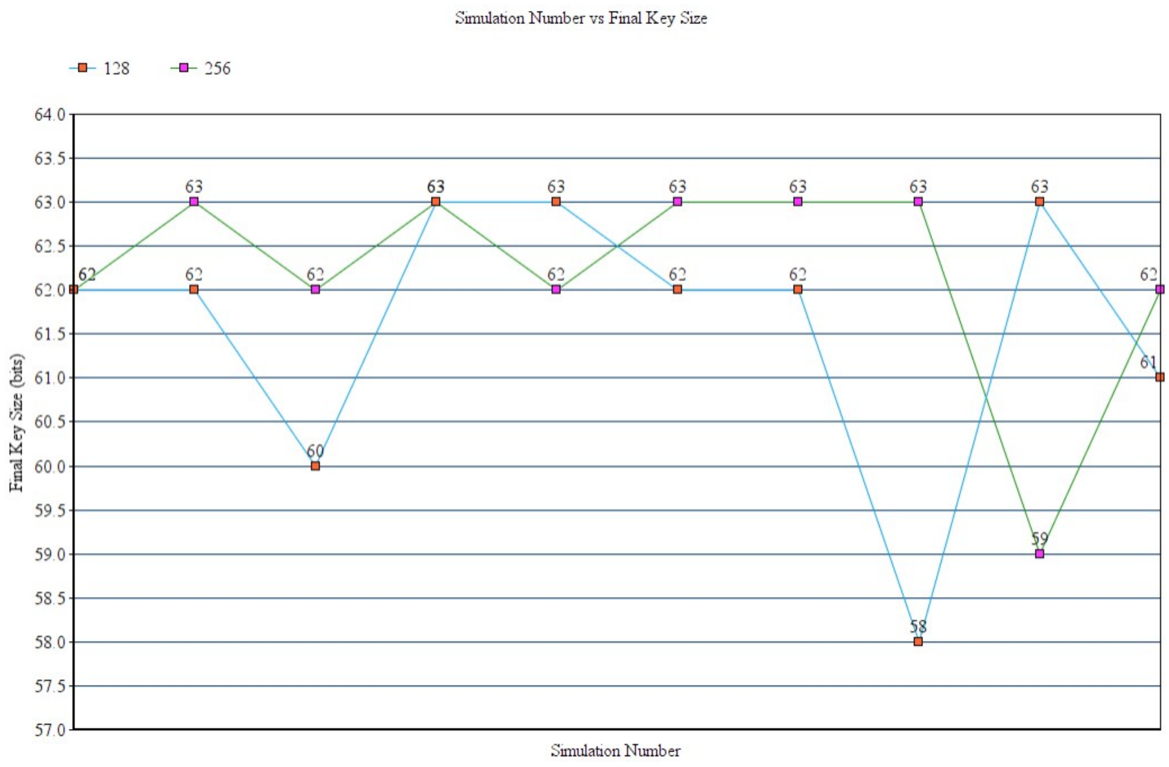


Figure 8. Simulation number vs. final key size.

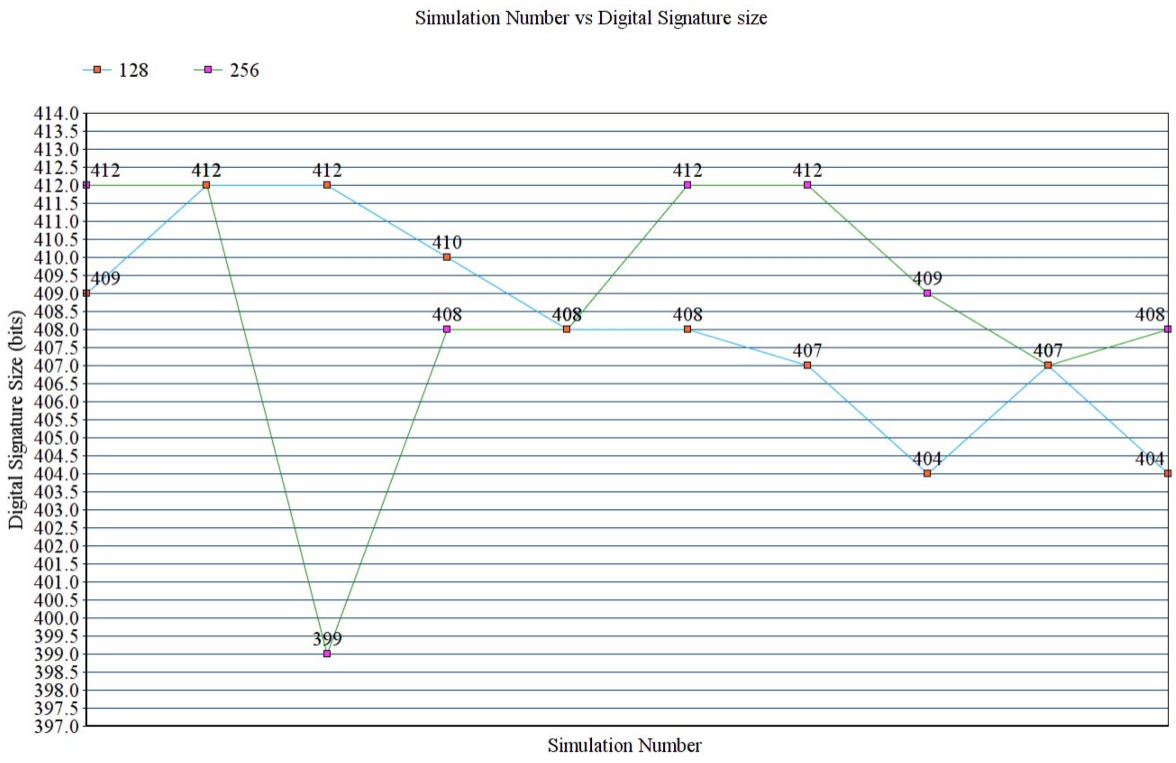


Figure 9. Simulation number vs. digital signature size.

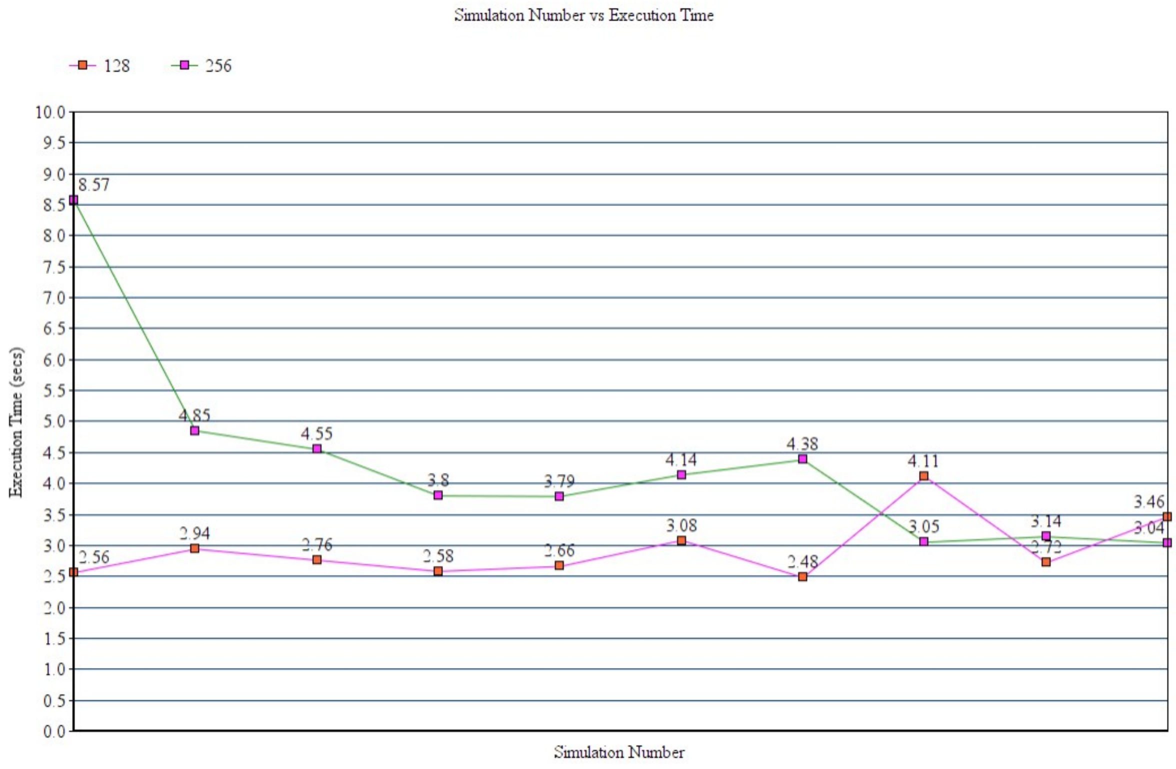


Figure 10. Simulation number vs. execution time.

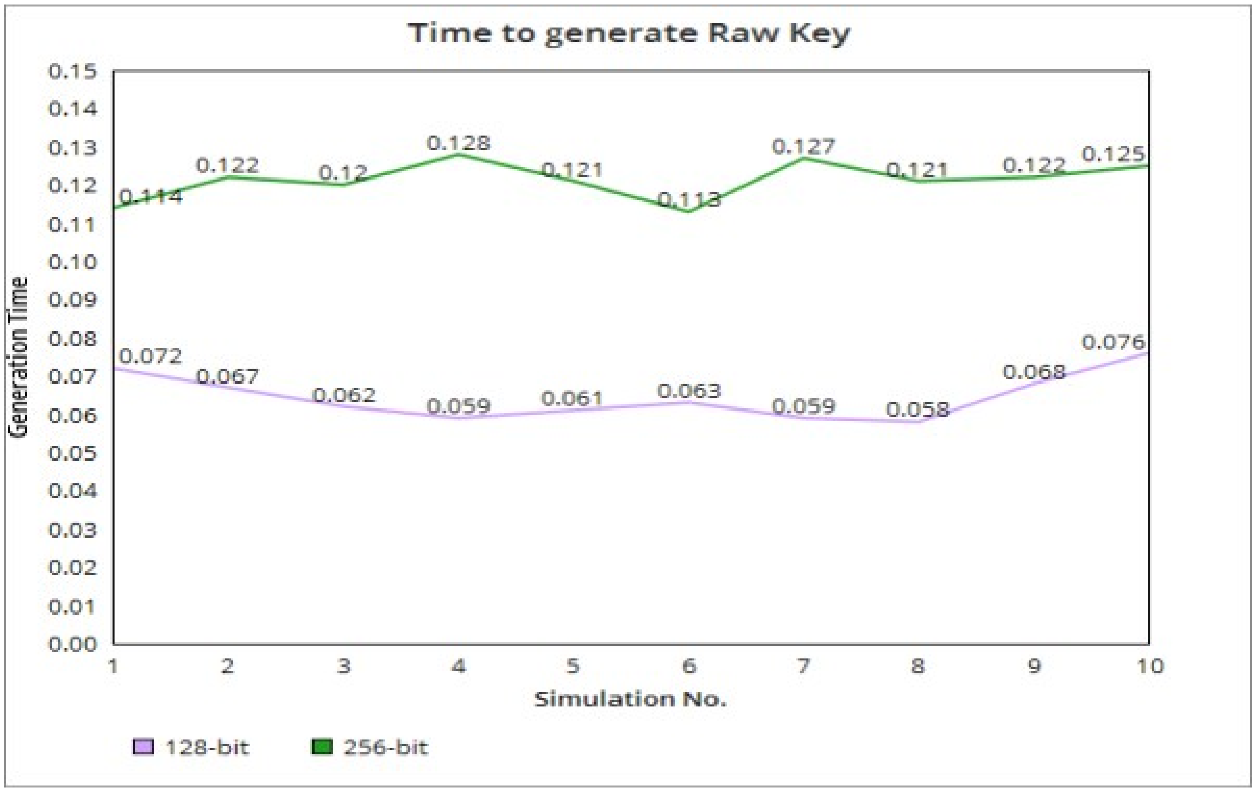


Figure 11. Simulation number vs. time to generate raw key (generation time).

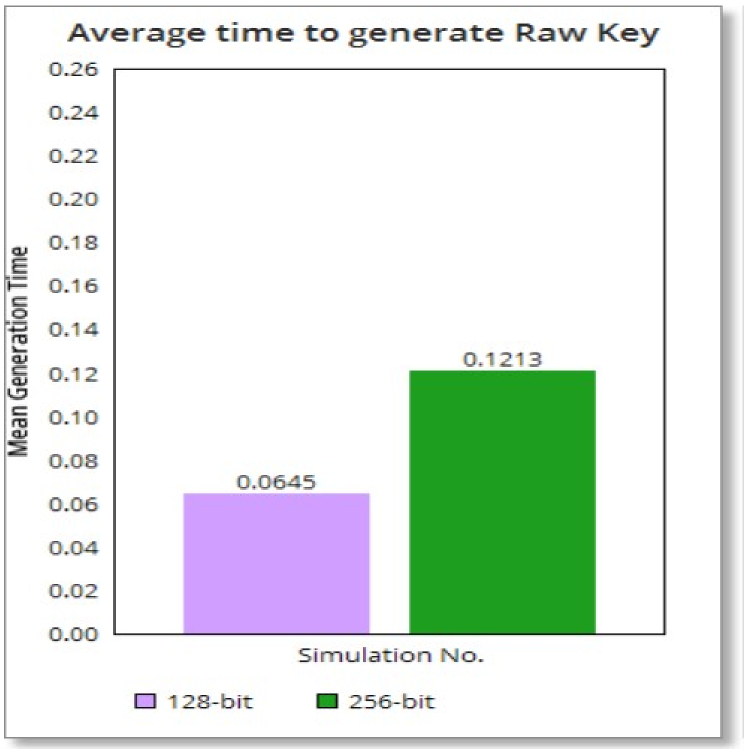


Figure 12. Comparison of Group 1: 128-bit raw key vs. Group 2: 256-bit raw key, using the mean value of generation time of each group.

Furthermore, Figure 13 illustrates the following behaviors.

- The QBER evidently increases as the size of the raw key increases.
- The sifted key size is directly proportional to the raw key size.
- The final key size does not vary when the raw key size varies.
- The digital signature size does not vary when when the raw key size is doubled.
- The execution time significantly changes when the raw key is adjusted.

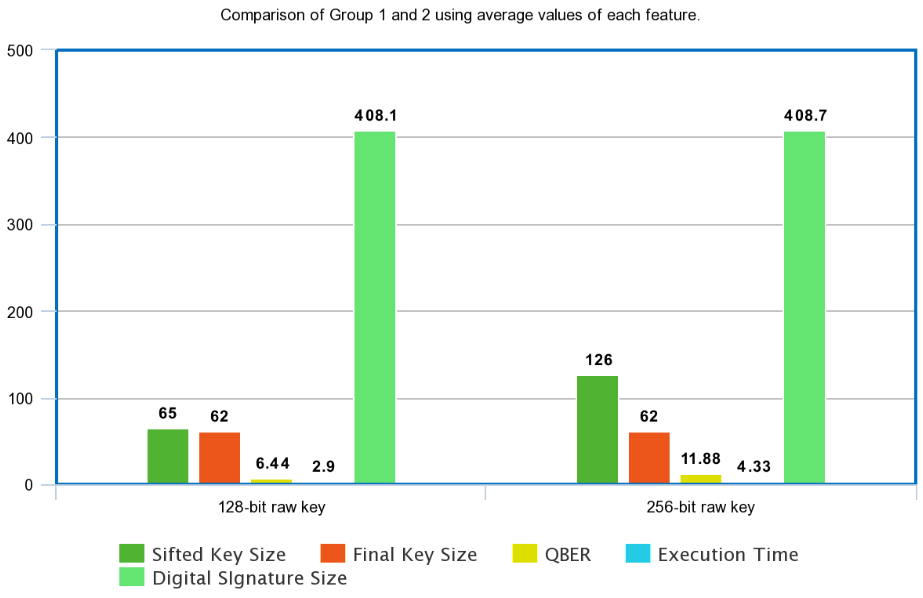


Figure 13. Comparison of Group 1: 128-bit raw key vs. Group 2: 256-bit raw key, using the mean value of each feature.

6.2. Comparative Analysis between AGA-12 vs. Our Proposed Scheme

We have performed the comparative analysis between algorithms used in AGA-12 and our proposed scheme. Since the novelty of our proposed algorithm walks on Quantum Key Distribution protocol, we have used three variables for comparative analysis, mainly, key size, randomness of keys, and execution time.

We executed five simulations of both the algorithms, AGA-12, and our proposed scheme. We have measured the BB84 raw key size, final key size, RSA public key size, and RSA private key size. The average raw key size is 256 bits, and, the final key size of BB84 is approximately 62 bits. The RSA public key size is 1041 bits, and the private key size is around 2048 bits. Figure 14 shows a graph visualizing the difference between the key sizes of each algorithm.

We fed the five keys of BB84 and the RSA algorithm used in AGA-12 to the NIST statistical test suite for randomness. We have analyzed the randomness of the keys based on the appropriate NIST randomness test. Figure 15 shows the percentage of tests passed by both RSA and BB84 keys. We observed that the RSA public key, 1041 bits, passed 98% and the RSA private key, average size 2048 bits, passed 90% of the randomness tests while BB84 raw and final keys have passed 100% of the tests. Moreover, we have also compared the execution time of generating RSA keys and BB84 keys. Figure 16 shows a graph that shows the difference between the execution time to produce keys by the RSA and BB84 algorithms. The mean execution time of generating BB84 keys is 0.014554 s, and that of generating RSA keys is 0.4805578 s.

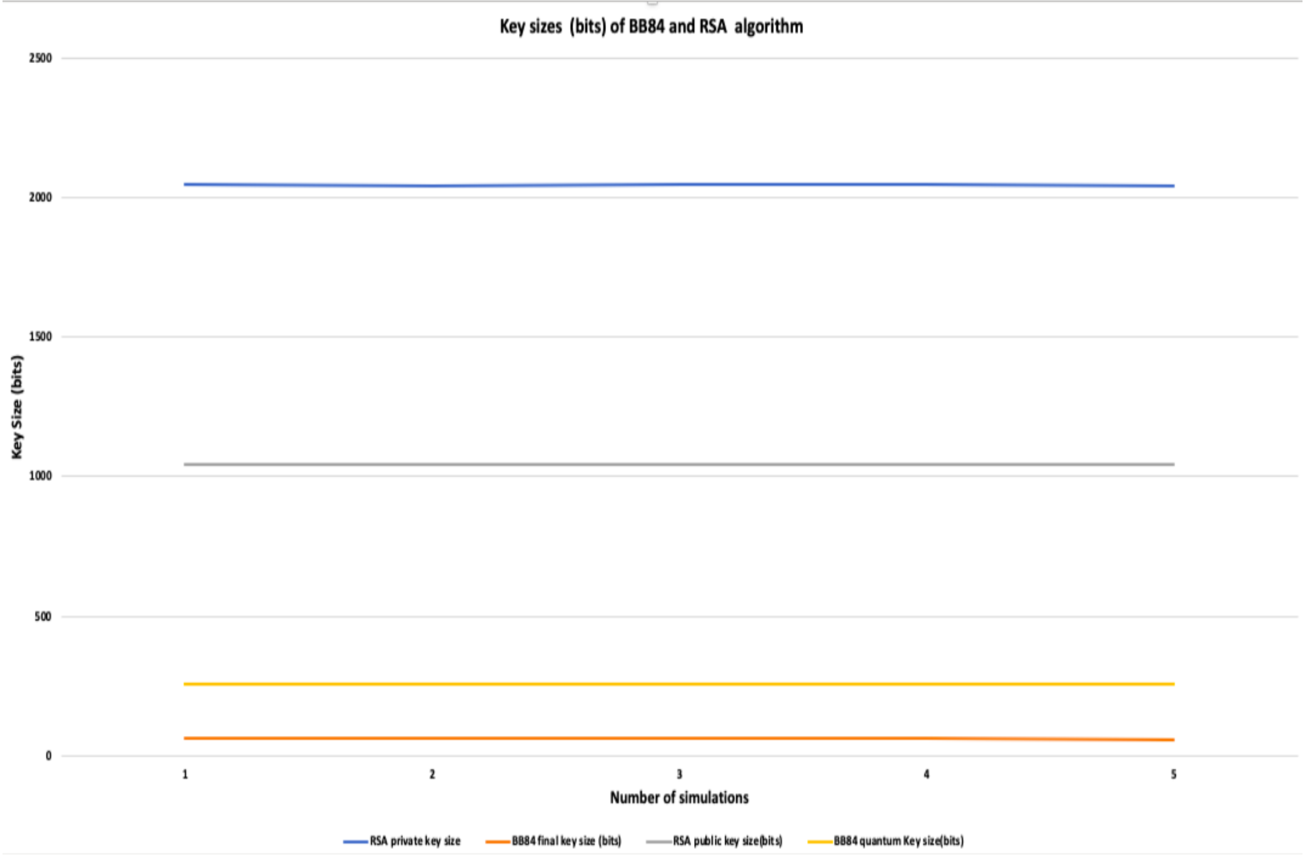


Figure 14. Comparison of RSA keys vs. BB84 keys over 5 simulations.

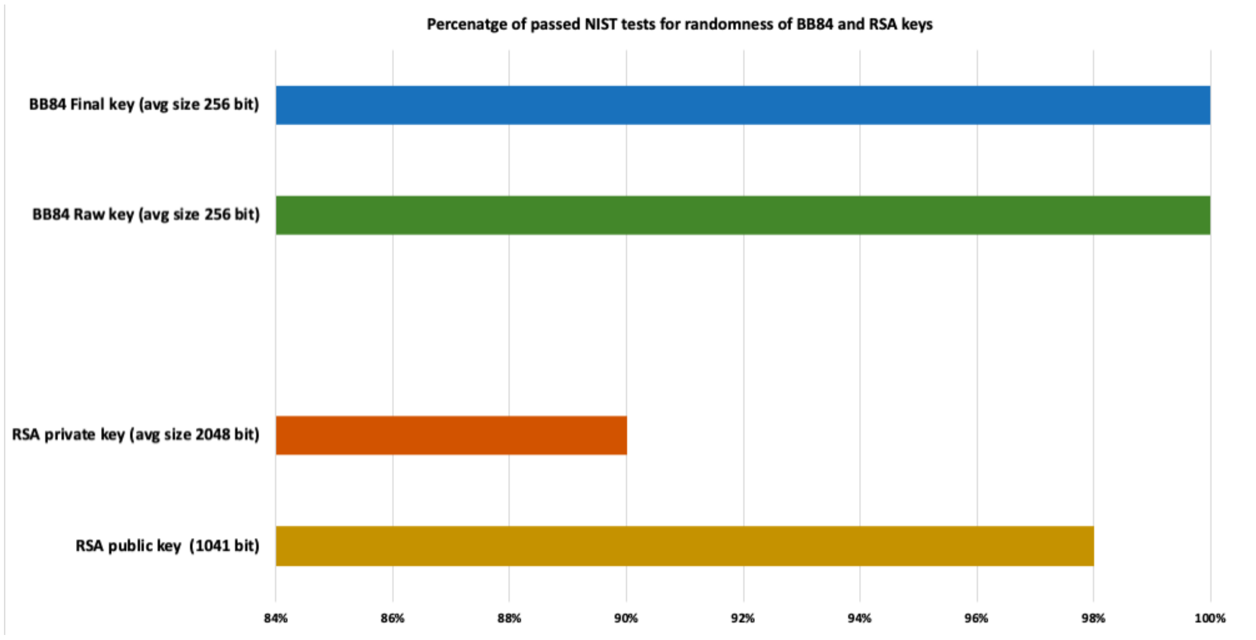


Figure 15. Comparison of percentage of passed NIST tests for randomness of BB84 and RSA keys.

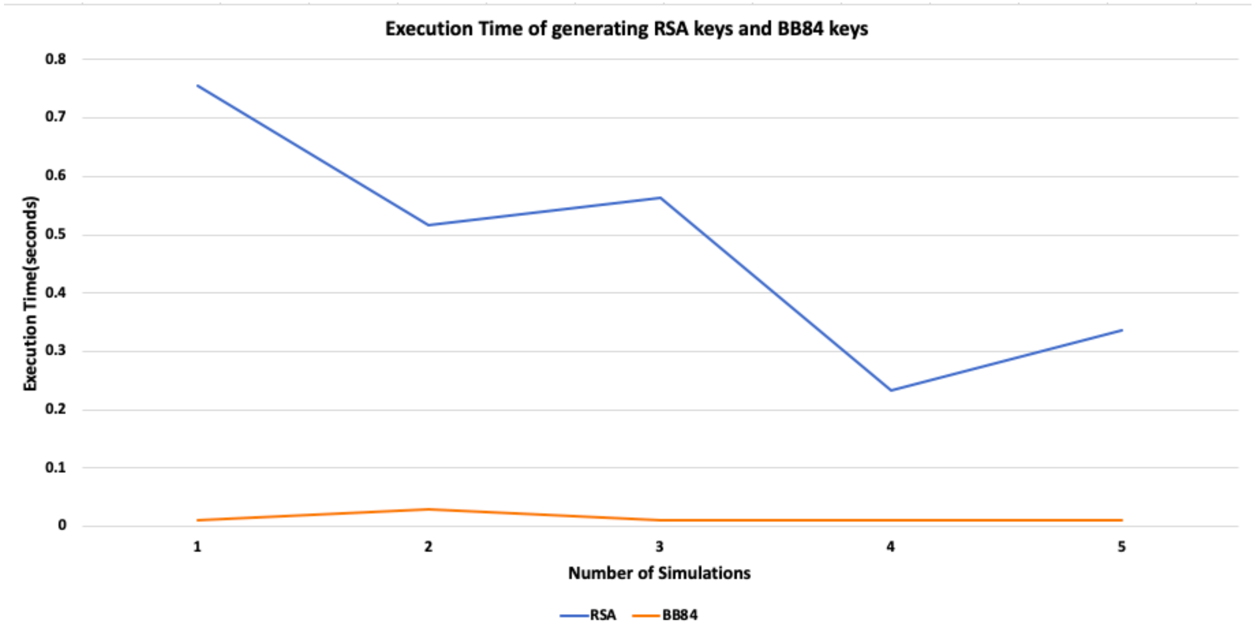


Figure 16. Comparison of execution time of generating RSA keys used in AGA-12 and BB84 keys used in proposed scheme.

6.3. Challenges of Implementing a QKD to SCADA Networks

Our proposed algorithm secures the communication link between RTU and MTU, which is a point-to-point link. A QKD, on the other hand, on a network topology requires two channels, mainly quantum channel (fiber optic) or classical channel (Internet or fiber optic) [4]. Considering hardwired communication, a QKD in the SCADA network topology need $2 \times n$ fiber optics, n being the traditional number of fiber optics [49]. Furthermore, if one of the channels is broken, the entire network shuts down. However, as future research work, a quantum channel can be designed to exchange both qubits and bits of data. Therefore, it will decline the number of fiber optics required in the SCADA network [49].

7. Conclusions

In this paper, we have proposed a novel security scheme which uses the properties of quantum physics to provide the following benefits to SCADA networks.

- It resists not only the attacks of traditional computers but also quantum computers using Shor's algorithm. It also defends against man-in-the-middle attack.
- It is an encryption algorithm which also acts as an intrusion detection system.
- The scheme adds authentication to the communications between units.
- It does not rely on any third party for key generation and authentication.

Our proposed scheme attains all the security goals of integrity, confidentiality, availability, authentication, and non-repudiation. The randomness property of the key and its size enhances the security of the protocol. It uses uncertainty and superposition properties of quantum physics to detect any eavesdropping. Thus, compared to other traditional security schemes, it acts as an encryption as well as an intrusion detection scheme without relying on a third party. Therefore, it reduces the computational cost.

As a part of our future work, we will improve our proposed scheme by mainly working on the digital signature algorithm. We also intend to perform a comparative analysis of the AGA-12 digital signature and our proposed scheme. Additionally, we will conduct a comparative study by implementing various hash functions in the signature scheme and find out which yields the most potent hash. Our proposed scheme focuses on MiM attack and one quantum attack: brute-force attack using Shor's algorithm. However, an eavesdropper can use the properties of quantum mechanics to launch probe attacks on Quantum Key Distribution [50,51]. In our future work, we will focus on these types of attacks, mainly, entangling-probe [50] and Fuchs–Peres–Brandt (FPB) probe attack [45,51]. In future work, we will perform a comparative analysis of our proposed quantum-based signcryption with post-quantum security hybrid signcryption. Wang et al. [52] have proposed a quantum secure signcryption that involves the extension of signcryption to lattice cryptography. However, our algorithm is based on exploiting the fundamentals of quantum computing and extending signcryption to quantum cryptography.

Author Contributions: S.G.: conceptualization, formal analysis, methodology, writing—first draft; M.Z.: project administration, writing—review and editing; B.P.: project administration, writing—review and editing, S.S.: funding acquisition, supervision, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the support in part by the Natural Sciences and Engineering Research Council (NSERC) and industry partners Cistel Technology Inc., through a Collaborative Research Grant.

Data Availability Statement: Not applicable.

Acknowledgments: Thanks to Rohit Joshi of Cistel Technologies for his valuable feedback.

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

References

1. Nader, P.; Honeine, P.; Beauseroy, P. l_p-norms in one-class classification for intrusion detection in SCADA systems. IEEE Trans. Ind. Inform. **2014**, 10, 2308–2317.
2. Saputra, H.; Zhao, Z. Long term key management architecture for SCADA systems. In Proceedings of the 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, 5–8 February 2018; pp. 314–319. https://doi.org/10.1109/WF-IoT.2018.8355183.
3. Choi, D.; Kim, H.; Won, D.; Kim, S. Advanced Key-Management Architecture for Secure SCADA Communications. IEEE Trans. Power Deliv. **2009**, 24, 1154–1163.
4. Ghosh, S.; Sampalli, S. A Survey of Security in SCADA Networks: Current Issues and Future Challenges. IEEE Access **2019**, 7, 135812–135831. https://doi.org/10.1109/ACCESS.2019.2926441.
5. Kang, D.J.; Lee, J.J.; Kim, S.J.; Park, J.H. Analysis on cyber threats to SCADA systems. In Proceedings of the 2009 Transmission & Distribution Conference & Exposition: Asia and Pacific, Seoul, Korea, 26–30 October 2009; IEEE: New York, NY, USA, 2009; pp. 1–4.
6. Lomonaco, S. Shor's quantum factoring algorithm. In Proceedings of the Symposia in Applied Mathematics, San Diego, CA, USA, 4–5 January 2002; Volume 58, pp. 161–180.

7. Grover, L.K. A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 212–219.
8. Dennis, R. Quantum Computers Are the Most Powerful Tech Threat to Cryptocurrency. 2018. Available online: https://blog.icoalert.com/quantum-computers-are-the-most-powerful-tech-threat-cryptocurrency-will-face (accessed on 14 January 2019).
9. Mavroeidis, V.; Vishi, K.; Zych, M.D.; Jøsang, A. The impact of quantum computing on present cryptography. arXiv **2018**, arXiv:1804.00200.
10. Hosoyamada, A.; Sasaki, Y. Quantum collision attacks on reduced SHA-256 and SHA-512. In Proceedings of the Annual International Cryptology Conference, Online, 16–20 August 2021; Springer: Berlin, Germany, 2021; pp. 616–646.
11. Sibson, P.; Erven, C.; Godfrey, M.; Miki, S.; Yamashita, T.; Fujiwara, M.; Sasaki, M.; Terai, H.; Tanner, M.G.; Natarajan, C.M.; et al. Chip-based quantum key distribution. Nat. Commun. **2017**, 8, 13984.
12. Chandra, S.; Paira, S.; Alam, S.S.; Sanyal, G. A comparative survey of symmetric and asymmetric key cryptography. In Proceedings of the 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE), Hosur, India, 17–18 November 2014; IEEE: New York, NY, USA, 2014; pp. 83–93.
13. Zhang, X.; Dong, Z.Y.; Wang, Z.; Xiao, C.; Luo, F. Quantum cryptography based cyber-physical security technology for smart grids. In Proceedings of the 10th International Conference on Advances in Power System Control, Operation & Management (APSCOM 2015), Hong Kong, China, 8–12 November 2015.
14. Busch, P.; Heinonen, T.; Lahti, P. Heisenberg's uncertainty principle. Phys. Rep. **2007**, 452, 155–176.
15. Sinha, A.; Vijay, A.H.; Sinha, U. On the superposition principle in interference experiments. Sci. Rep. **2015**, 5, 10304.
16. Bužek, V.; Hillery, M. Quantum copying: Beyond the no-cloning theorem. Phys. Rev. A **1996**, 54, 1844.
17. Johnson, J.S.; Grimaila, M.R.; Humphries, J.W.; Baumgartner, G.B. An analysis of error reconciliation protocols used in quantum key distribution systems. J. Def. Model. Simul. **2015**, 12, 217–227.
18. Portugal, R. Quantum Walks and Search Algorithms; Springer: Berlin, Germany, 2013.
19. Hwang, R.J.; Lai, C.H.; Su, F.F. An efficient signcryption scheme with forward secrecy based on elliptic curve. Appl. Math. Comput. **2005**, 167, 870–881.
20. Zaverucha, G.M.; Stinson, D.R. Short one-time signatures. Adv. Math. Commun. **2011**, 5, 473.
21. Yan, H.; Peng, X.; Lin, X.; Jiang, W.; Liu, T.; Guo, H. Efficiency of Winnow protocol in secret key reconciliation. In Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering, Los Angeles, CA, USA, 31 March–2 April 2009; IEEE: New York, NY, USA, 2009; Volume 3, pp. 238–242.
22. Singh, V.; Sharma, N. A Review on Various Error Detection and Correction Methods Used in Communication. Am. Int. J. Res. Sci. Technol. Eng. Math. **2015**, 15, 252–257.
23. Alabady, S.A.; Al-Turjman, F. Low complexity parity check code for futuristic wireless networks applications. IEEE Access **2018**, 6, 18398–18407.
24. Choudhari, S.P.; Chakole, M.B. Reed solomon code for WiMAX network. In Proceedings of the 2017 International Conference on Communication and Signal Processing (ICCSP), Melmaruvathur, India, 6–8 April 2017; IEEE: New York, NY, USA, 2017; pp. 0176–0179.
25. Lu, X.; Feng, D. Quantum digital signature based on quantum one-way functions. In Proceedings of the 7th International Conference on Advanced Communication Technology, ICACT 2005, Phoenix Park, Korea, 21–23 February 2005; IEEE: New York, NY, USA, 2005; Volume 1, pp. 514–517.
26. Abdullah, G.M.; Mehmood, Q.; Khan, C.B.A. Adoption of Lamport signature scheme to implement digital signatures in IoT. In Proceedings of the 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 3–4 March 2018; IEEE: New York, NY, USA, 2018; pp. 1–4.
27. Cleary, F.; Felici, M. Cyber Security and Privacy: 4th Cyber Security and Privacy Innovation Forum, CSP Innovation Forum 2015, Brussels, Belgium April 28–29, 2015, Revised Selected Papers; Springer: Berlin, Germany, 2015; Volume 530.
28. Ponomarev, S.; Atkison, T. Industrial control system network intrusion detection by telemetry analysis. IEEE Trans. Dependable Secur. Comput. **2015**, 13, 252–260.
29. ICS Advisory (ICSA-10-201-01C). Available online: https://www.cisa.gov/uscert/ics/advisories/ICSA-10-201-01C. (accessed on 15 January 2019).
30. Carcano, A.; Coletta, A.; Guglielmi, M.; Masera, M.; Fovino, I.N.; Trombetta, A. A multidimensional critical state analysis for detecting intrusions in SCADA systems. IEEE Trans. Ind. Inform. **2011**, 7, 179–186.
31. Ponomarev, S.; Wallace, N.; Atkison, T. Detection of ssh host spoofing in control systems through network telemetry analysis. In Proceedings of the 9th Annual Cyber and Information Security Research Conference, Oak Ridge, TN, USA, 8–10 April 2014; pp. 21–24.
32. Cekerevac, Z.; Dvorak, Z.; Prigoda, L.; Cekerevac, P. Internet of things and the man-in-the-middle attacks-security and economic risks. MEST J. **2017**, 5, 15–25.
33. Gidney, C.; Ekerå, M. How to factor 2048 bit RSA integers in 8 h using 20 million noisy qubits. arXiv **2019**, arXiv:1905.09749.
34. Karati, A.; Fan, C.I.; Hsu, R.H. Provably Secure and Generalized Signcryption with Public Verifiability for Secure Data Transmission Between Resource-Constrained IoT Devices. IEEE Internet Things J. **2019**, 6, 10431–10440.
35. Fröhlich, B.; Lucamarini, M.; Dynes, J.F.; Comandar, L.C.; Tam, W.W.S.; Plews, A.; Sharpe, A.W.; Yuan, Z.; Shields, A.J. Long-distance quantum key distribution secure against coherent attacks. Optica **2017**, 4, 163–167.

36. Routray, S.K.; Jha, M.K.; Sharma, L.; Nyamangoudar, R.; Javali, A.; Sarkar, S. Quantum cryptography for IoT: APerspective. In Proceedings of the 2017 International Conference on IoT and Application (ICIOT), Nagapattinam, India, 19–20 May 2017; IEEE: New York, NY, USA, 2017; pp. 1–4.
37. Kumar, A.; Garhwal, S. State-of-the-Art Survey of Quantum Cryptography. Arch. Comput. Methods Eng. **2021**, 28, 3831–3868.
38. Sun, S.; Huang, A. A review of security evaluation of practical quantum key distribution system. Entropy **2022**, 24, 260.
39. Bennett, C.H.; Brassard, G. Quantum cryptography: Public key distribution and coin tossing. Theor. Comput. Sci. **2014**, 560, 7–11.
40. Riley, M.; Richardson, I. An Introduction to Reed-Solomon Codes: Principles, Architecture and Implementation.2003. Available online: https://www.cs.cmu.edu/~guyb/realworld/reedsolomon/reed_solomon_codes.html. (accessed on 30 August 2020).
41. Soykan, E.U.; Ersoz, S.D.; Soykan, G. Identity based signcryption for advanced metering infrastructure. In Proceedings of the 2015 3rd International Istanbul Smart Grid Congress and Fair (ICSG), Istanbul Turkey, 29–30 April 2015; IEEE: New York, NY, USA, 2015; pp. 1–5.
42. Papanikolaou, N.K. Techniques for Design and Validation of Quantum Protocols. Master's Thesis, University of Warwick, Coventry, UK, 2005.
43. Kuppam, S. Modelling of Quantum Key Distribution Protocols in Communicating Quantum Processes Language with Verification and Analysis in PRISM. In Proceedings of the SIMULTECH 2018: 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Porto, Portugal, 29–31 July 2018; pp. 75–82.
44. Cremers, C.; Mauw, S. Operational semantics of security protocols. In Scenarios: Models, Transformations and Tools; Springer: Berlin/Heidelberg, Germany, 2005; pp. 66–89.
45. Lowe, G. A hierarchy of authentication specifications. In Proceedings of the 10th Computer Security Foundations Workshop, Rockport, MA, USA, 10–12 June 1997; IEEE: New York, NY, USA, 1997; pp. 31–43.
46. Victoria, U. coding515.pdf—ECE 515 Information Theory Channel Capacity and Coding 1 Information Theory Problems How to Transmit or Store Information as Efficiently. 2016. Available online: https://www.coursehero.com/file/35896396/coding515pdf/ (accessed on 7 June 2019).
47. Quantum Information Toolkit—Quantum Information Toolkit 0.11.0 Documentation. Available online: http://qit.sourceforge.net/docs/html/ (accessed on 7 June 2019).
48. Williams, C.P. Quantum Gates. In Explorations in Quantum Computing; Texts in Computer Science; Springer: London, UK, 2011; pp. 1–5.
49. Ghosh, S.; Zaman, M.; Sakauye, G.; Sampalli, S. An Intrusion Resistant SCADA Framework Based on Quantum and Post-Quantum Scheme. Appl. Sci. **2021**, 11, 2082.
50. Azuma, H. An entangling-probe attack on Shor's algorithm for factorization. J. Mod. Opt. **2018**, 65, 415–422.
51. Shapiro, J.H.; Wong, F.N. Attacking quantum key distribution with single-photon two-qubit quantum logic. Phys. Rev. A **2006**, 73, 012315.
52. Wang, F.; Hu, Y.; Wang, C. Post-quantum secure hybrid signcryption from lattice assumption. Appl. Math. Inf. Sci. **2012**, 6, 23–28.