Integrity Enhancing Protocols: Performance and Recommendations for Nuclear Systems

Prepared for
US Department of Energy

Romuald Valme[1], Adam Beauchaine[1], Minami Tanaka[1], Christopher Lamb[1]

[1]Sandia National Laboratories

September 2024
SAND2024-12102

Telephone:	(865) 576-8401
Facsimile:	(865) 576-5728
E-Mail:	reports@osti.gov
Online ordering:	http://www.osti.gov/scitech

Telephone:	(800) 553-6847
Facsimile:	(703) 605-6900
E-Mail:	orders@ntis.gov
Online order:	https://classic.ntis.gov/help/order-methods/

ABSTRACT

In today's communication landscape there are multiple technologies and protocols used for communication between end devices. Within security paradigms for these protocols, integrity management is a common goal of system designers. Communication protocols focused on maintaining message integrity can provide assurance that some received data has not been altered or tampered with. While integrity is often coupled with confidentiality in protocol design, this analysis focuses on an evaluation of only integrity protocols. This report outlines various ways message integrity may be preserved with respect to high performance operational technology (OT) systems. It describes a series of experiments and an evaluation framework used to evaluate the performance of the identified integrity approaches regarding common system design goals. Finally, it addresses the testing environment utilized and closes the report with a summary of experimental results.

ACKNOWLEDGEMENTS

The team wishes to acknowledge Department of Energy office of nuclear energy (DOE-NE) for supplying financial support and resources to this project. This project was funded by DOE-NE under contract number NT0111000/234155. The team also wishes to thank Ben Cipiti, Katya Leblanc, and Dan Warner for their advocation, guidance, and review of this project.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

This page left blank

EXECUTIVE SUMMARY

Information and communication data are ubiquitous in modern digital systems. This data often contains critical instructions and commands for devices in Operational Technology (OT) environments. OT hardware is highly utilized in Advanced Reactor (AR) systems. The devices in AR systems leverage various protocols to carry critical data for control, read, and actuate actions within a reactor. The integrity of this data is critical for the safety and operation of these systems. Adversaries are aware of this and may target these devices by distorting the instructions being sent across the network. To prevent these adversaries from causing harm, systems that verify the integrity of the data being transmitted are necessary. Integrity Enhancing Protocols (IEP) can accomplish this task through a variety of algorithms. Throughout this paper, these protocols and the diversity of approaches that exist are explored.

Section one introduces integrity protocols and their use cases. This section outlines the motivation for these protocols and describes how OT devices and advanced nuclear reactor environments benefit from the incorporation of these technologies. With the appropriate algorithm, data can be sent in clear text with a short verifying message tag appended to it. This would result in less resource utilization and cost for a reactor system than combined approaches in which integrity is provided alongside confidentiality or authenticity.

Section two begins a detailed description of various categories of integrity protocols. Many features of these protocols are explored such as their pros and cons and implementation. This section outlines the underlying cryptographic techniques that underly many protocols. These topics include digital signature, public key cryptography, hashing, and message authentication codes (MACs). After this each category of protocol is discussed. In each category various protocols and corresponding details are outlined.

Section three dives into the methodology for testing protocols. It outlines the novel framework developed to analyze protocols. It also reviews the criteria used to select protocols to test. The framework is based on four metrics that were decided on based on literature reviews of evaluation metrics. These metrics are protocol running time, endpoint storage, hardware performance, and NIST security level. This section describes these metrics in greater detail. Many of these protocols exist in network environments with constant communications traffic. Details on how these environments can be implemented are also given. Advanced reactors benefit from the evaluation of protocols through a robust framework such as the one presented in section three.

Section four describes how the chosen protocols were implemented for testing. Here implementation questions regarding hardware and libraries are addressed. What benefits and advantages are derived from the team's selection in comparison to others. Testing tool implementation and data retrieval strategies are discussed. Finally, the virtualization platform and network topological schemes are discussed.

Section five provides an analysis for the collected data. In this section the team describes key findings from the chosen implemented protocols. The results show the lightweight nature of protocols such as ECDSA and MACs with their faster key generation time and lower CPU usage compared to standard algorithms such as RSA. This result can also be viewed in the table 1. This table shows a compiled form of averaged results per protocol. Timing metrics are recorded in MS, whereas spatial metrics are recorded in bytes. CPU and RAM measurements are recorded as

percentages of total usage while the program was active i.e., CPU 2.3 = 2.3% usage over a testing operation.

Out of all the algorithms sampled, MAC had the highest performance for strict integrity provision. This performance can be attributed to the absence of a public key cryptosystem in favor of a shared secret architecture. Reactor environments which mostly require data integrity can save on resources and costs by avoiding the usage of public key algorithms. The range of performance in post-quantum algorithms is seen by the faster round trip time of the lattice-based Dilithium in comparison with the hash-based Sphincs+ algorithm. By incorporating algorithms such as Dilithium AR systems looking for quantum resistance would optimize CPU and memory usage thus saving operating costs. The findings provided in the results section give operators and engineers the information they need to properly implement integrity assurances tailored to their environment and without the additional complexity of using confidentiality or authenticity measures. Integrity protocols can also function as an active component of a Defensive Cyber Security Architecture (DCSA), integrating into secure elements and devices throughout a network.

ECDSA (secp256k1)							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
4461.545	72.49505	548.6535	416	48	1.261276	6.969479	806.3168317
ECDSA (Brainpool)							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
12698.62	151.1782	1738.059	416	48	1.836975	8.952559	2009.128713
RSA							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
77162.01	269.7624	403.7129	72	48	4.438073	7.04877	941.4356436
Dillithium							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
226.7327	271.703	287.9109	72	48	1.558841	5.973637	1249.19802
MAC							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
78.71287	3.980198	17.85149	24	40	1.474	5.973583	825.6732673
Sphincs+							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
9299.03	326.198	5933.525	72	48	7.865583	6.878221	6635.990099
ElGamal							
KeyGen (MS)	**Encap (MS)**	**Decap (MS)**	**Key Size (Bytes)**	**CS Size (Bytes)**	**CPU (%)**	**RAM (%)**	**RTT (MS)**
2131484	205.0792	867.6634	72	48	19.31755	6.70472	1278.257426

Table 1: Average Protocol Scores for Framework Components

ACRONYMS AND TERMS

Acronym/Term	Definition
IoT	Internet of Things
OT	Operational Technology
AR	Advanced Reactor
MAC	Message Authentication Code
RAM	Random Access Memory
NIST	National Institute of Standards and Technology
DCSA	Defensive Cyber Security Architecture
TPM	Trusted Platform Module
GCM	Galois Counter Mode
RSA	Rivest–Shamir–Adleman (Public Key Cryptosystem)
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standards
PQC	Post Quantum Cryptography
CPU	Central Processing Unit
KVM	Kernel-based Virtual Machine
QEMU	Quick Emulator
VM	Virtual Machine
VCPU	Virtual Central Processing Unity
AES	Advanced Encryption Standard
SHA	Secure Hashing Algorithm

1. INTRODUCTION

In recent years, the complexity of operational technology (OT) systems has grown considerably. This has coincided with a rise in cybersecurity threats targeting this new, larger attack surfaces of OT environments. Within some disciplines which incorporate OT, focus on cybersecurity infrastructure is often limited, leading to OT environments with outdated security provisions. [1] These environments can often be unprepared to meet the challenges posed by cyber-attacks, based on the speed at which these attacks are evolving [2]. Within OT security infrastructure, the paradigm of integrity has been noted as particularly important, given the capacity for false data injection to trigger unwanted, potentially dangerous events if physical behavior is impacted [2]. Integrity is often coupled with additional security goals, such as authenticity as is seen in digital signatures. Confidentiality guaranteeing ciphers may additionally include integrity as a provided service. This report focuses uniquely on integrity-guaranteeing protocols, while this includes those that offer additional security provisions, integrity is the solely evaluated goal. This is due to the theoretical low cost of integrity implementation, as well as the benefits to OT systems that come with such provisions [2].

System security engineers need a clearer understanding of the current landscape of integrity guaranteeing protocols for OT infrastructure. They need an analysis of current approaches to integrity and what performance and security impacts such approaches have on OT systems. This report provides a clear background of modern integrity protocols and how they are implemented within OT systems. It documents the most relevant techniques and practices and how they may be applied within OT systems with respect to protocol performance, technical complexity, and implementation costs. It substantiates these observations with real result data from unique protocol implementations, using metrics designed for OT systems.

Such implementations will be highly valuable in OT environments supporting Advanced Reactors (AR) systems. Protocol performance is of high relevance within AR systems, due to the time-critical nature of applications receiving real-time physics data. Due to low theoretical overheads of integrity-guaranteeing protocols, AR systems could benefit from a "low-cost" layer of an important security provision. This report aims to provide actionable conclusions on implementation guidelines for AR systems, among a broader OT paradigm.

To explore the landscape of integrity-guaranteeing protocols and strategies, this report discusses the history and growing need for integrity within modern OT system architectures. It identifies and explores current standards of integrity protocols, as well as novel approaches being adopted within modern secure systems. It additionally discusses technical and implementation details of each of the integrity systems or paradigms presented, as well as their capacity to integrate into existing OT systems. The diversity of integrity protocols presented allows for a plethora of variations in architecture. Designers can introduce systems that focus primarily on integrity, where data is passed in clear text with a message tag appended which with the appropriate algorithm can verify the integrity of the data. This design can reduce resource utilization and cost. It proposes the usage of a test environment to evaluate performance of a smaller subgroup of protocols determined as most relevant to OT systems. It records and highlights the performance factors of this subgroup using a novel protocol analytic tool and presents visualized views of the resulting data outputs. Finally, this report highlights various attributes and features that separate integrity protocols, such as integrity at rest or integrity in transit.

Evaluated protocols are referred to by public key cryptographic scheme names. Within the context of this work, all public key signature schemes are assumed to reference digital signature algorithms. For example, an evaluation of RSA refers to the RSA digital signature implementation. Due to the limitation of analysis to integrity protocols, alternative uses for public key architecture, such as secret establishment or one time encryption, are not discussed in this work. Protocols that do not leverage public key schemes are specified when mentioned to avoid confusion. Additionally, due to the separation of integrity and confidentiality for experimentation, all message data is transmitted in clear text during experimentation.

2. BACKGROUND

The background of this work discusses integrity guaranteeing tools and techniques, as well as prior work which incorporates them. It discusses the history and motivations behind integrity in modern secure system design, as well as cryptographic approaches for integrity management, and how they may be used to guarantee integrity of information systems. For each of these approaches, special attention is given to their history and potential implementations within an OT system, considering to the unique requirements relating to performance and compatibility that OT environments possess. Each of these techniques has been selected based on relevance to OT integrity management challenges. Additional discussion is included regarding techniques applied to both data in transit as well as storage, including digital signatures, message authentication codes, hashing, data checksums, and Galois counter mode for block ciphers. Finally, implementation strategies for these techniques within existing systems are discussed, with a specific focus given to OT systems and infrastructure.

2.1. History

In Cryptography, integrity ensures that the message received is the same as what the sender transmitted. It guarantees that the data has not been altered by some nefarious actor or system error. The usage of the term integrity was pioneered and formally defined in the paper *A Comparison of Commercial and Military Security Policies* [3]. Integrity provisions are derived from the classical "subject-object" model of access control, in which constrained data assets must have a dedicated, controlled transformation procedure to be written to. This is combined with regular checks of data infrastructure to ensure no illegitimate changes have been made to data assets. The paper also defines unconstrained data assets as those which may be modified with simple read/write primitive operations, noting that allowing for data alterations in this manner represents a threat to integrity.

There are various techniques that can be leveraged to provide integrity in modern systems. One of these techniques is the digital signature, which also provides authenticity protection. Digital signatures allow you to verify that a message has been sent by the expected party. In 1976 Whitfield Diffie and Martin Hellman devised the idea of a digital signature. This idea however was incomplete. It was missing an asymmetric encryption algorithm. A year later Ronald Rivest, Adi Shamir, and Len Adleman developed the RSA algorithm which would enable the production of a rudimentary digital signature leading to the first digital signature used, in Lotus Notes, in 1988.

2.2. Public Key Cryptography and Digital Signatures

Public-key cryptography enables digital signatures. In public-key cryptography everyone has their own unique encryption and decryption key. During communication someone would encrypt a message to a receiver with the receiver's public key which only the receiver could decrypt with their private key. However, in digital signatures the process is reversed. The sender's private key is used to encrypt or sign a message which then everyone can validate the origin of using the sender's public decryption key. The figure below shows an example communication involving a digital signature between two different parties Alice and Bob. The figure shows how Bob must sign the message x using his private key. It also shows how the public key can be used to verify the source of the message.

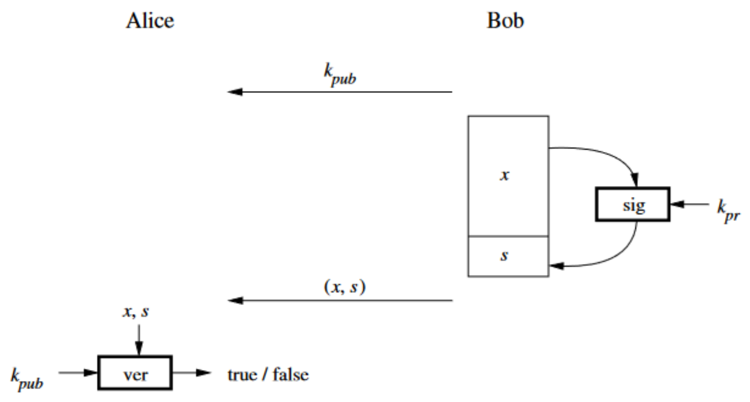


Figure 1: Digital Signature Process [4]

Digital signatures ensure data integrity. If even a single bit of the message in transit is altered by an adversary, the decrypted result will be vastly different. These algorithms typically have multiple phases. The phases typically include key generation, key distribution, document hashing, document signing, and signature verification. In key generation the public key algorithm is selected, and the public and private key are generated based on various parameters. The keys are then securely delivered and installed onto their corresponding systems. The above figure shows the message x being signed, but the length of the message may pose issues in transmission. It is not scalable to have the entire message x being signed and transmitted.

To mitigate this problem, one must select and utilize a hashing algorithm. The hashed document is then encrypted using the private key in a process known as signing and once delivered to the receiver is decrypted using the decryption key through a process known as verification. The goal of a digital signature is to guarantee data integrity and authenticity. The data cannot have been tampered with and must have originated from the holder of the private key.

2.3. Hashing and Message Authentication Codes (MACs)

As seen in the architecture of digital signatures, the usage of cryptographic hash functions within integrity preserving communication schemes has been widespread. Any known hash function shared between two communication participants may serve this purpose in a simple manner. By hashing the original message and appending the output to the data sent, the receiver may apply the same hash function on the original message and compare the results. If the results are, assuming a hash function of sufficient preimage resistance, the recipient may verify the integrity of the data. Cryptographic hashing functions do not innately provide an authentication equivalent mechanism, or encryption in transit. As such, they are typically only used as components in secure communication schemes.

Message authentication codes (MAC) refers to a system of message authentication. Through the usage of a shared secret key **k**, MAC implements a signing algorithm that outputs a unique tag **t** on the input of **k** and message **m.** This tag is then appended to the original message. Because a trusted recipient also has access to the shared secret key **k**, they may perform an identical procedure to attain the tag **t**, which verifies the sender is a trusted keyholder.

MAC is commonly leveraged in combination with cryptographic hash functions to provide authentication and integrity in secure communication. Such schemes are referred to as hash-based message authentication codes (HMAC). Within an HMAC system, both parties derive two secret keys **k1** and **k2** from the original key **k**. Two rounds of message concatenation and hashing are then performed, with **k1** and **k2** being used for each round key. The first round of the algorithm generates an internal hash that is then passed as input to the next round, the output of this round creates the final HMAC code as shown below in figure 2.

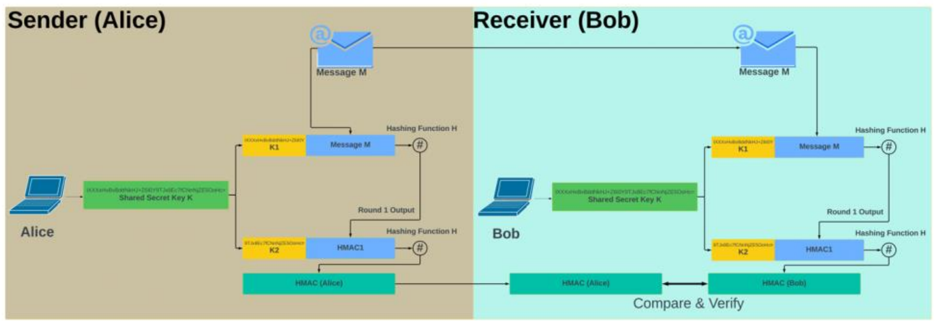


Figure 2: HMAC Algorithm

The repeated rounds of HMAC allow for strong resistance to hashing length extension attacks. This procedure is repeated by the recipient, thereby guaranteeing the integrity of data and identity of the sender. The authentication and hashing algorithm must be repeated on a per transmission basis by both parties for any instance of communication. This has caused several challenges in the implementation of HMAC on resource constrained systems, such as OT environments. Techniques in message construction, or in incorporating MAC directly into transfer mechanisms have been effective in addressing these concerns, shared secret cryptosystem integrity models are discussed below in section 2.5. This report has selected MACs for evaluation as seen in section 3.1.

2.4. Checksums

A cryptographic checksum allows for generalized integrity measurements within a variety of systems. As a general-case definition, checksum refers to a small function that outputs a distinct value based on its input, typically at the binary storage level. This is usually a cryptographic hashing function, but other techniques, such as modular summation of bits, may be used. (Fletcher's Algorithm) Regardless of algorithm, checksum schemes must demonstrate significant diffusion capabilities to be considered successful. Diffusion refers to the capability for some cryptographic algorithm to significantly alter ciphertext output when only marginal changes are made to its plaintext input. Such schemes must also provide pre-image resistance, making it challenging for an adversary to construct a selected valid output, even after being given some input-output pair.

The usage of cryptographic hash functions as checksums appears in a variety of recent schemes. They can be leveraged in providing integrity measurements to remote internet files [5], and providing on-disk integrity measurement of a file system [6]. In both cases, endpoints seeking to establish integrity must have some secure database of trusted checksum outputs. These may be compared with new checksum outputs upon starting a workflow and accessing a checksum-guarded file. This comparison allows for the detection of binary level file modifications thanks to the diffusion model of cryptographic hash functions. In the case of filesystem checksum schemes, filesystem objects may be modified to store a set of pointers to block checksum values associated with the filesystem object. Multi-party checksum schemes are introduced on a per-transaction basis, to allow for the measurement of the data sent in one network transaction. Examples of such schemes include message authentication codes (MAC), and hash-based message authentication codes (HMAC).

Current checksum approaches are related to best practices in cryptographic hashing functions. Current used schemes include the SHA-2 and SHA-3 families of hashing functions. Legacy protocols such as MD5 and SHA-1 are no longer used, due to their small collision space and vulnerability to length extension attacks. In all cases, the usage of checksums for integrity verification may be a computationally expensive task, especially when used to check every individual file-system block. [6] The storage of checksum outputs may also present difficulties, particularly in lightweight systems often found in OT environments. This report addresses these concerns and provides a specified list of integrity measurement approaches, for filesystem data as well as network transactions.

2.5. Block Cipher Integrity Models

Among existing communication standards, there have been measures to implement integrity verification directly into secure multi-party communication models. These include shared secret or symmetric cryptosystems such as AES. Previously discussed approaches such as MAC or HMAC may be simply added after ciphertext construction and function in a standard manner for symmetric cryptosystems, but this significantly increases system complexity and is less computationally efficient when compared to incorporating integrity measurements directly into the cipher suite. This has given rise to integrity guaranteeing modes of operation for symmetric block ciphers, which leverage the natural block encryption mechanisms to create a MAC for an entire multi-block message as a component of the encryption operation.

The most widely leveraged of these techniques is the Galois Counter Mode (GCM) mode of operation [7]. A mode of operation refers to a method of linking block encryption operations within a block cipher, such as AES. By leveraging galois field arithmetical operations across a set of encrypted blocks, GCM may generate a MAC for an entire message sent between parties, regardless of the length/number of blocks used. GCM leverages a standard initialization vector (IV) for block randomization, as well as an additional shared secret key **K2.** This key is used to perform multiplicative galois field arithmetic operations across the full set of encrypted blocks to produce a MAC for the IV and ciphertext values. If either of these is modified in the encryption or in transit, receiver verification will fail. This enables a single verification procedure to be incorporated as part of a large data transmission consisting of many blocks.

This MAC is generated in a similar manner to AES finite field arithmetic. On a given **input key K2** and **irreducible polynomial P**, the first block ciphertext output is multiplied in GF(128) mod **P**. This resulting output is then XORed with the ciphertext of the second block, with this process being repeated until reaching the end of the block cipher output, and subsequently multiplied again by **K2.** This process is repeated until the final round at the end of the block chain, where two final XOR and multiply operations are performed for the length of the message and the IV value to produce the MAC. This process is visualized below in figure 3.

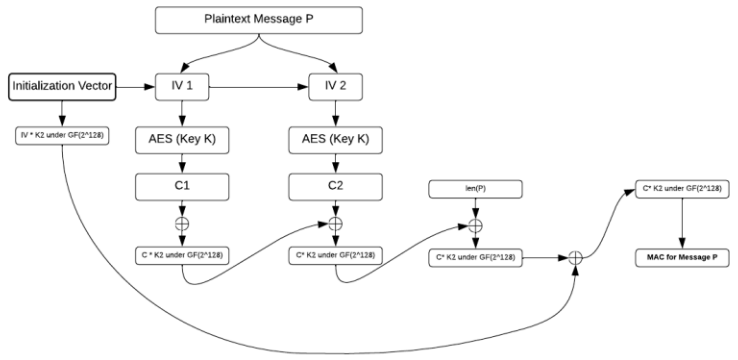


Figure 3: GCM Operation within the AES Cryptosystem

2.6. Existing Approaches

The review of the current state of the art when it comes to integrity enhancing protocols is necessary for later comparisons of this work. The main concepts and ideas of protocols are presented from their corresponding research papers. The origin, history, and structure of protocols are presented. The nature and finer details of these protocols will be uncovered through diagrams and tables. These protocols are put in environments where they are utilized to protect critical assets and data because of this the security and vulnerabilities are conferred. Many of these protocols are used today in various applications, these are described as well as how they could be used in energy facilities.

2.6.1. Multiple Signature Approaches

In circumstances where a document has more than one author there may need to be multiple signatures applied. In [8], the authors describe various types of multi-signature schemes. A sequential approach can be taken where a document is hashed and encrypted to produce the first signature. The resulting signature is then signed in a similar process. This method is iterated until all signers have been represented. A sequential approach is often valuable when there is an ordered hierarchy to the authors [9]. An owner signing a document processing an employee's transfer application may not sign the document unless the employee has signed first. Routing systems have also employed sequential schemes through aggregation. This allows them to reduce memory constraints on nodes that are limited in their resources [10].

There is also a parallel approach where a document is hashed and encrypted by one signer and then the same document is run through the signing process for another signer. Two unique signatures are produced for the same document. One may need this kind of approach in for example an online shop where multiple parties need to sign the same purchase. However, both approaches suffer from different drawbacks. One creates multiple signatures which can be tedious to manage and the other has a complex ordering for its signing and verification. However, there is an approach where you can have multiple signatories in a single digital signature.

Through a modified El Gamal encryption algorithm a group can have multiple signatories with one signature. The private key of each signatory is added to the exponent of the encryption, which is often seen with the Discrete Logarithm Problem (DLP). One does not have to hash their message multiple times, which can be an expensive operation. The speed of the hash function depends on how long the input message is and how many blocks the input fits into or takes to hash. One could instead input their key into the El Gamal algorithm. This approach has been selected for evaluation as seen in section 3.1.

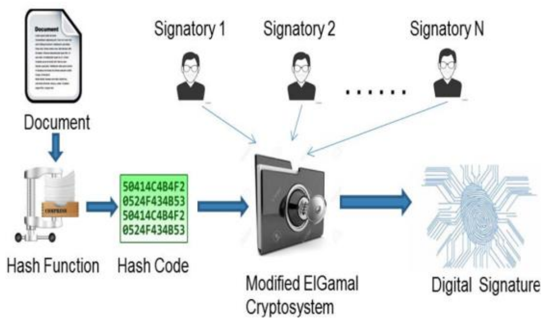


Figure 4: Multisigner Process [8]

2.6.2. *Elliptic Curve (Lightweight) Cryptography*

Within OT systems, cryptography comprises various confidentiality, integrity, and availability protection schemes for secure communication. One category of approaches is often referred to as "lightweight" due to chip area availability, and energy constraints. Due to requiring less chip space (RAM) and memory (ROM), lightweight cryptography provides security solutions for low latency systems with less computing power. Lightweight cryptography plays a critical role in securing OT protocols due to the resource-constrained nature of OT controllers and field devices. These system components often have limited processing power, memory, and energy, making traditional cryptographic algorithms impractical. Lightweight cryptography algorithms are specifically designed to address these constraints while still providing adequate security. Lightweight cryptography is most used today for efficient cryptographic algorithms, secure communication protocols, data encryption and authentication, and device access and access controls. These algorithms are tailored to perform well on resource constrained IoT devices, ensuring that cryptographic operations can be executed without draining the device's limited battery power or overwhelming its processing capabilities. This section provides an overview of classical algorithms used to secure OT device integrity, with a special focus given to elliptic curve cryptography.

Lightweight cryptography is also commonly used to encrypt and authenticate data transmitted between OT devices or stored locally on the devices themselves. For example, lightweight block ciphers are employed to encrypt sensor data before transmission, ensuring that sensitive information remains confidential even if intercepted by malicious actors. Regarding integrity, devices often use lightweight cryptographic protocols like Elliptic Curve Diffie-Hellman (ECDH) or Elliptic Curve Digital Signature Algorithm (ECDSA) for secure key exchange and message integrity. Additionally, lightweight authentication mechanisms such as message authentication codes (MACs) or digital signatures are used to verify integrity and authenticity of data, helping prevent data tampering or spoofing attacks.

The digital signatures used today can be classified according to the underlying mathematical problem which provides the basis for their security including schemes like: (1) Integer Factorization (IF) schemes, which "base their security on the intractability of the integer factorization problem" [11] (i.e. RSA and Rabin signatures schemes), (2) Discrete Logarithm (DL) schemes, which "base their security on the intractability of the (ordinary) discrete logarithm problem in a finite field" [11] (i.e. ElGamal, Schnorr, DSA, and Nyberg-Rueppel), and (3) Elliptic Curve (EC) schemes, which "base their security on the intractability of the elliptic curve discrete logarithm problem". [11]

Elliptic Curve Cryptography (ECC), when used in lightweight algorithms for device authentication, is commonly implemented for key exchange and digital signatures in OT/IoT protocols. This is due to its strong security properties and relatively low computational overhead. ECC can provide equivalent security to traditional public key cryptography algorithms like Rivest-Shamir-Adleman (RSA) and Diffie-Hellman but with smaller key sizes, making it more suitable for resource-constrained devices. This approach is well suited for "lightweight implementations with a minimal security level and with a limited hardware overhead". [12]

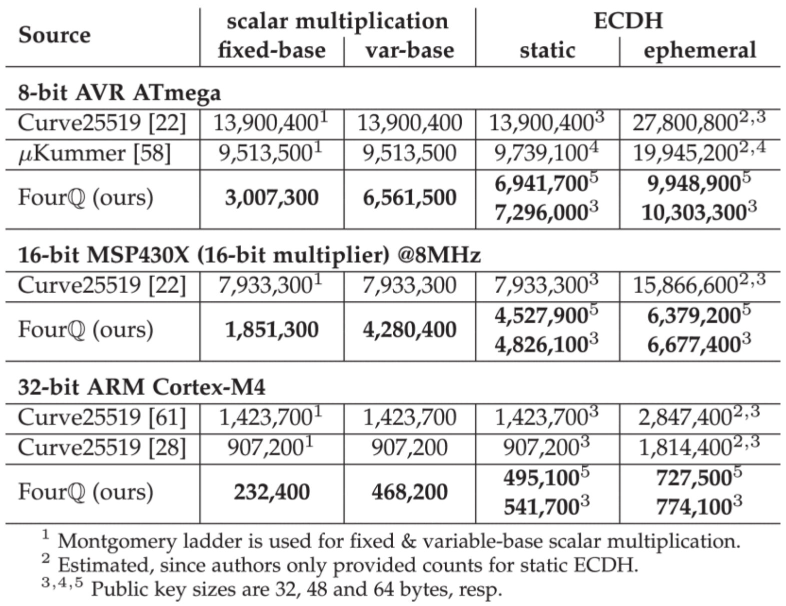
The Elliptic Curve Digital Signature Algorithm (ECDSA) is the "elliptic curve analogue of the Digital Signature Algorithm (DSA)". [11] The DSA can be viewed as a variant of the ElGamal signature scheme with a security "based on the intractability of the discrete logarithm problem in prime-order subgroup of Z*p". [11] ECDSA key pairs are associated with a particular set of domain parameters. The public key is a "random multiple of the base point, while the private key is the integer used to generate the multiple". [11] Before implementing ECDSA, the type of underlying finite field (F_q), polynomial or normal basis field representation, type of elliptic curve and elliptic curve point representation must be considered.

Edward-curve Digital Signature Algorithm (ECDSA) is a modern digital signature algorithm based on elliptic curve cryptography, designed to be efficient and secure. Signature creation for ECDSA is "deterministic in nature whereas ECDSA requires high-quality randomness for every signature to be safe. If it uses low-quality randomness, then an attacker can compute the private key". [13] It is commonly used in securing IoT protocols such as MQTT and CoAP for secure communication, as it offers strong security with smaller key sizes compared to traditional algorithms like RSA.

Elliptic Curve Diffie-Hellman (NIST P-256) is a "prime curve that has been used extensively in critical infrastructure projects, is being used as the Elliptical Curve Digital Signature Algorithm for AS-path signing and verification in the BFPSEC protocol". [14] ECDSA for the curve P-256 delivers 128-bits of security on computationally low-power hardware like IoT devices.

Elliptic Curve Diffie-Hellman (ECDH) for Curve25519 is a "255-bit elliptic curve offering approximately 128-bit classical security". [15] ECC implementations for efficient key exchange over Curve25519 provides secure lightweight public-key cryptography solutions for several applications, primarily for IoT devices.

FourQ is a "high-performance elliptic curve that provides about 128 bits of security and enables efficient and secure scalar multiplications". [16] Implementations based on this curve have shown to achieve the fastest computations of variable-base, fixed-base, and double-scalar multiplications to date on a very large variety of x64 and ARMv7-A processors. Overall, results obtained from different software and hardware platforms consistently report the FourQ is "5 times faster than the standardized NIST curve P-256 and more than 2 times faster than Curve25519". [16] Results for variable-base, fixed-base, static ECDH, and fully ephemeral ECDH key exchange comparing FourQ to Curve25519 is shown in figure 5.



Source	scalar multiplication		ECDH	
	fixed-base	var-base	static	ephemeral
8-bit AVR ATmega				
Curve25519 [22]	13,900,400[1]	13,900,400	13,900,400[3]	27,800,800[2,3]
μKummer [58]	9,513,500[1]	9,513,500	9,739,100[4]	19,945,200[2,4]
Four$\mathbb{Q}$ (ours)	**3,007,300**	**6,561,500**	**6,941,700**[5] **7,296,000**[3]	**9,948,900**[5] **10,303,300**[3]
16-bit MSP430X (16-bit multiplier) @8MHz				
Curve25519 [22]	7,933,300[1]	7,933,300	7,933,300[3]	15,866,600[2,3]
Four$\mathbb{Q}$ (ours)	**1,851,300**	**4,280,400**	**4,527,900**[5] **4,826,100**[3]	**6,379,200**[5] **6,677,400**[3]
32-bit ARM Cortex-M4				
Curve25519 [61]	1,423,700[1]	1,423,700	1,423,700[3]	2,847,400[2,3]
Curve25519 [28]	907,200[1]	907,200	907,200[3]	1,814,400[2,3]
Four$\mathbb{Q}$ (ours)	**232,400**	**468,200**	**495,100**[5] **541,700**[3]	**727,500**[5] **774,100**[3]

[1] Montgomery ladder is used for fixed & variable-base scalar multiplication.
[2] Estimated, since authors only provided counts for static ECDH.
[3,4,5] Public key sizes are 32, 48 and 64 bytes, resp.

Figure 5: Results for Variable-Base and Fixed-Base Scalar Multiplication, Static ECDH, and Fully Ephemeral ECDH Key Exchange [16]

An example of a newly designed multiple-time signature scheme that provides lightweight security solutions on FourQ curve is the Signer Efficient Multiple-time Elliptic Curve Signature (SEMECS). SEMECS properties include:

- High computation and energy efficiency at the signer as it only requires "two hash function calls, a single modular multiplication, and modular subtraction to generate a signature". [17]
- Compact private key and signatures as it only requires storing a "32-bit private key (that can be derived from a 16-byte seed with a PRF) and incur an additional 32 Bytes to the message as the signature, for k=128-bit security level". [17]
- Open-source implementation and comprehensive analysis as the original researchers had open-sourced all the implementations for broad testing, benchmarking, and adoption purposes.
- Provable security with a tight reduction as SEMECS has a "tight reduction to the Discrete Logarithm Problem (DLP), without the need for the forking lemma, as a Fiat-Shamir type signatures do". [17]

Although, despite its merits, this newly implemented multiple-time signature scheme has its limitations that are inherent to multiple-time signatures. SEMECS limitations include, but are not limited to, (i) the ability to sign up to a pre-determined k message, but requiring bootstrapping, (ii) requiring resourcefulness of storage as the public key size is linear with respect to k, and (iii) stateful signature scheme relying on previous signature states.

In a recent study analyzing the performance of commonly used public-key key exchange protocols including RSA, Diffie-Hellman, Elliptic Curve Diffie-Hellman (NIST P-256), Curve25519, and FourQ, in terms of key pair generation and secret key exchange agreement, it can be seen in Figure 6 that RSA is "severely impacted by the long bit length required to maintain the security level target" [18] and standard "Diffie-Hellman is a bit slower than the elliptic curve variants, which all seem to be equivalent at this scale". [18]

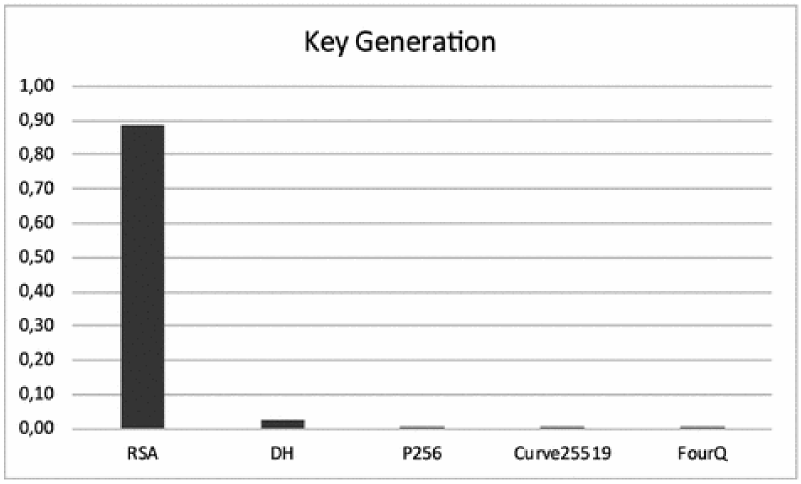


Figure 6: Key Generation Time Required (in seconds) of RSA, DH, P256, Curve2219, and FourQ [18]

When considering only the elliptic curve Diffie-Hellman based algorithms, shown in Figure 7, the standard NIST-256 curve is several times slower than Curve25519 and FourQ.

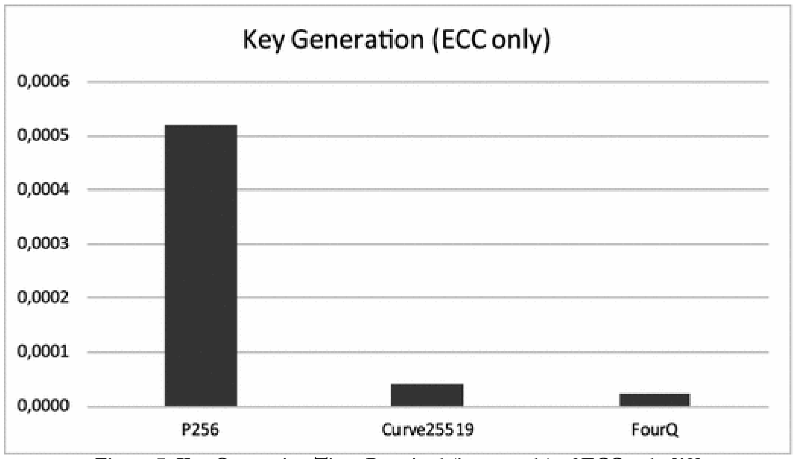


Figure 7: Key Generation Time Required (in seconds) of ECC only [18]

Regarding the secret value exchange operations, Figure 8 demonstrates that the Diffie-Hellman is much slower than alternative algorithms and RSA is marginally slower than the elliptic curve variants.

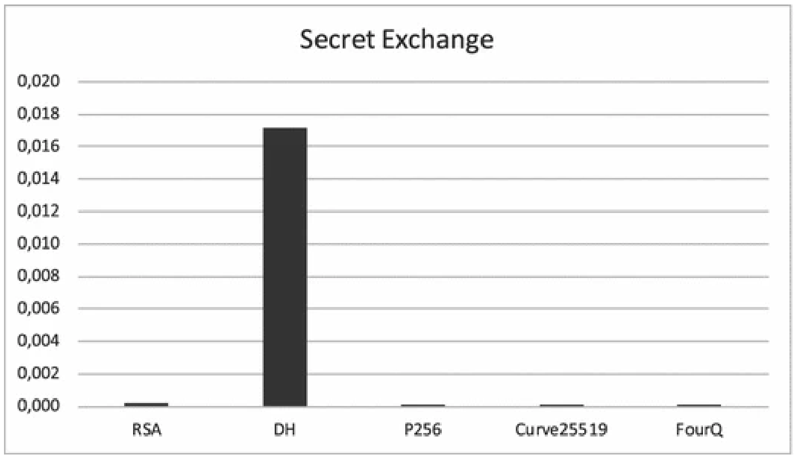


Figure 8: Secret Exchange Time Required (in seconds) of RSA, DH, P256, Curve2219, and FourQ [18]

Focusing only on the elliptic curve Diffie-Hellman variants, shown in Figure 9, again note that there are very "tangible performance benefits with the Curve25519 and, especially, FourQ ECDH schemes". [18]

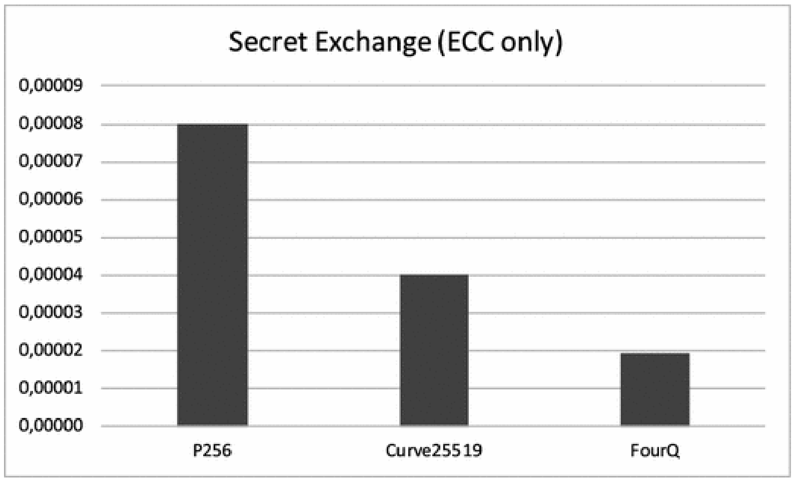


Figure 9: Secret Exchange Time Required (in seconds) of ECC only [18]

The performance of lightweight protocols can be leveraged for OT environments to reduce resource utilization comparable to what is done in IoT systems. Overall, lightweight cryptography is essential for enabling secure and efficient communication and data exchange in the rapidly expanding technology ecosystem. ECDSA algorithms have been selected for evaluation as seen in section 3.1.

2.6.3. Post-Quantum Cryptographic Schemes

The emergence of quantum computing as a cryptanalysis tool has seen continuous interest and research engagement in recent years. [19] Cryptographic schemes designed to resist quantum computing cryptanalysis are referred to as post-quantum and are more computationally complex to initiate and verify than traditional cryptography. The development of such schemes poses novel questions for OT systems in relation to integrity requirements. Since most post-quantum schemes do not alter existing key-based architecture, they may be viewed as "upgrades" to existing protocols if implemented correctly. Post-quantum schemes are primarily lattice based (Falcon, Dilithium, NTRU) or multivariate (LUOV, Picnic, MQDSS). Lattice based schemes may function similarly to classical schemes in practice, but they are based on the hardness of lattice problems, as opposed to the hardness of factoring, discrete log, etc. Lattice based schemes have drawn attention in IoT post-quantum literature due to their comparatively low hardware requirements for endpoints. [20] To introduce the notion of lattice-based schemes, this section briefly details relevant linear algebra concepts.

Consider a set $\boldsymbol{S}$ of n-dimensional real vectors, $\{\boldsymbol{V_1}, \ldots \boldsymbol{V_n}\}$. $\boldsymbol{S}$ may be labeled as linearly independent if for all real numbers $\boldsymbol{A_i}$, the equation $\boldsymbol{A_1} * \boldsymbol{V_1} + \cdots + \boldsymbol{A_n} * \boldsymbol{V_n} = \boldsymbol{0}$ implies that all $\boldsymbol{A_i} = \boldsymbol{0}$. Consider the set of all real linear combinations of these vectors to form a vector subspace $\boldsymbol{C}$ with $\boldsymbol{S}$ as the basis. $\boldsymbol{C} = \{\sum_{i=0}^{n} \boldsymbol{A_i} * \boldsymbol{V_i} : \boldsymbol{A_i} \in \mathbb{R}\}$ A lattice is a vector subspace in which only integer linear combinations of $\boldsymbol{A_i}$ may be used. To build one, simply define $\boldsymbol{L} = \{\sum_{i=0}^{n} \boldsymbol{A_i} * \boldsymbol{V_i} : \boldsymbol{A_i} \in \mathbb{Z}\}$. This discretization of the subspace results in useful properties, such as a definite smallest vector, one may also determine the closest vector to any given vector in the subspace. These properties serve as the groundwork for most lattice-based schemes, when constructed in a high dimension to ensure cryptographic hardness. An example visualization of a closest vector problem may be viewed below in figure 10. Once constructed, keys may be used in identical manner to traditional encryption standards.

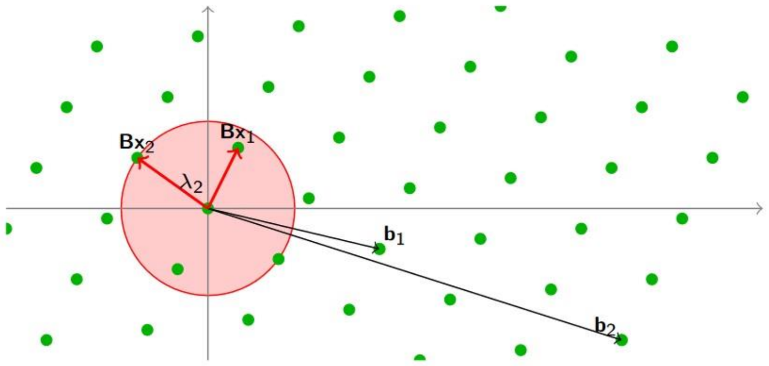


Figure 10: An Example Low Dimension CVP Solution [21]

One such construction mechanism for a public key lattice scheme is the Learning with Errors (LWE) problem. This problem is constructed by introducing some degree of error into a linear system of matrices, with an adversary being required to find the secret vector $\boldsymbol{s}$ when given a matrix $\boldsymbol{A}$ and a vector $\boldsymbol{b}$ within the linear system. $\boldsymbol{As} + \boldsymbol{e} = \boldsymbol{b}.$ Public key encryption is performed by selecting a random small vector $\boldsymbol{r}$ and modulus $\boldsymbol{q}$. Output ciphertext $\boldsymbol{c} = \boldsymbol{rA} + \frac{\boldsymbol{q}}{\boldsymbol{2}} * \ (\boldsymbol{\sigma}, \boldsymbol{0}, \dots, \boldsymbol{0})$. This may subsequently be decrypted by taking the inner product $\boldsymbol{y} = \ \langle \boldsymbol{c}, \boldsymbol{s} \rangle \boldsymbol{mod}(\boldsymbol{q})$, and $\boldsymbol{output\ 0\ if}\ |\boldsymbol{y}| < \frac{\boldsymbol{q}}{\boldsymbol{4}} \boldsymbol{else\ output\ 1}$, this will produce the original plaintext. Such keys can be used in digital signatures or other security applications.

Multivariate schemes provide comparable levels of cryptographic hardness to lattice-based schemes but are based on problems involving multivariate polynomial equations. Likely the most ubiquitous example is the Lifted Unbalanced Oil and Vinegar (LUOV) digital signature scheme [22], built around a system of m multivariate quadratic polynomials with n variables, divided into two parts, $\boldsymbol{x_1}, \dots, \boldsymbol{x_v}$: $\boldsymbol{vinegar\ variables}$; $\boldsymbol{x_{v+1}}, \dots, \boldsymbol{x_n}$: $\boldsymbol{oil\ variables.}$ Oil variables are kept public while vinegar variables are kept secret. Signatures are produced through matrix generation on an input of vinegar variables to be collapsed into a linear system, which may be solved through gaussian elimination with oil variables to produce a signature. Additional multivariate schemes may be based on multivariate cubic problems, [23] or separate constructions of quadratic problems. [24]

Within OT systems, performance overhead is a major concern. This makes the implementation of certain post-quantum schemes challenging on individual endpoint devices in industrial information systems. Despite this, research has identified lattice-based schemes as generally more computationally efficient than multivariate schemes. As part of a broad review of the subject, [25] identify lattice-based cryptography through LWE as having several desirable properties for IoT environments, including a strong performance evaluation of ring-LWE schemes on embedded devices. They additionally identify network bandwidth requirements as an area of concern, citing NIST PQC round 2 evaluation metrics. These network performance statistics are displayed below. Additional open-source benchmarking has been performed on the ARM Cortex-M4 processor family, used in embedded devices. These evaluations note CRYSTALS-Kyber and SABER as the most performant schemes among those tested regarding memory stack size and execution clock cycles. Section 3.1 specifies the various algorithms that have been chosen for evaluation. Post-quantum algorithms are among them.

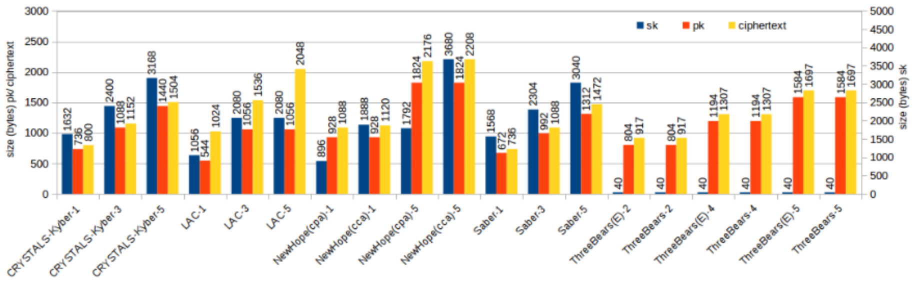


Figure 11: NIST PQC Round 2 Network Bandwidth Performance [26]

3. FRAMEWORK SPECIFICATION AND TESTBED DEVELOPMENT

Given the number of integrity-providing protocols across a wide range of organizational scenarios, a goal of this work is to offer insight into solutions specifically of high relevance to OT environments. Performance analysis of such solutions is particularly relevant, due to the importance of performant system design in real time OT environments. A set of integrity-providing protocols is selected, based on theoretical applicability and industry engagement within the domain of OT. Protocol performance evaluation is a widely studied subject, so increased focus is given to protocols that have not received such evaluative investigations previously. In addition, certain older protocols may not have been evaluated on recent hardware, despite providing similar levels of security to more widely studied solutions. The evaluative process seeks to measure system and network performance of a large set of protocols, and thus chose to leverage a virtualized testbed to implement protocols in a consistent environment. These observations are then analyzed through an OT-specific framework to quantify protocol performance in relation to OT systems. The testing platform leverages a scripted virtual machine (VM) deployment mechanism to isolate protocol differences in a testbed, and record system and network performance impact. The insights gained by the usage of this testbed and protocol selection are intended to enhance understanding of OT system protocol performance and help OT system designers in the construction of performant, integrity-providing systems.

3.1. Selected Protocols for Testbed Evaluation

The chosen protocols are selected primarily on history of implementation in OT environments, or theoretical applicability to OT environments based on existing research. Additionally, more commonly leveraged protocols are evaluated as a point of comparison to newer, less documented methods. Selection is additionally based off applicability to integrity provision of a simple network transaction, most based around digital signatures. This specific focus allows evaluation to be strictly limited to protocols of high relevance in OT communications systems, as well as provide useful metrics for system improvement.

Given the inclusion of digital signatures a high number of integrity-guaranteeing schemes, the main thrust of testing efforts is centered around digital signature algorithms. Elliptic curve digital signature algorithms (ECDSA) and RSA continue to be among the most widely used classical integrity enhancing approaches. Both have been widely tested for system and network performance impact and thus serve as ideal points of comparison for newer approaches. These newer approaches and corresponding testing rationale are detailed inline.

- **Post Quantum Integrity Schemes:** Notable lattice based public key/digital signature schemes are selected from NIST PQC Round 3 finalists, specifically Crystals Dilithium. This selection allows for an analysis of a diverse set of construction mechanisms in lattice-based cryptography.
- **Hash Based Integrity Schemes:** The most widely used hash-based digital signature algorithm is selected, SPHINCS+
- **Multi-Signature Integrity Schemes:** Scenarios are altered to include multi-signature integrity schemes and test the modified ElGamal cryptosystem for integrity provision to multiple endpoint nodes.

To serve as a point of comparison to these newer approaches, the analysis additionally includes several classical protocols previously described. These protocols are subjected to the same evaluative activities as newer approaches and presented alongside them in results, a condensed view of raw average result data is presented in Table 1 in the Executive Summary.

- **Elliptic Curve Cryptography:** The analysis includes two elliptic curve variants, a Koblitz curve (SECP256k1), and a standard Weierstrass curve (BrainpoolP256R1).
- **Message Authentication Codes:** The standard MAC architecture detailed in section 2.3 is implemented.
- **RSA:** The Digital Signature Algorithm leveraging RSA is implemented, with SHA-256 hashing algorithm.

3.2. Operational Technology Evaluative Framework

Existing protocols have been subject to many evaluative efforts in relation to performance, relative security, and scalability. Despite these efforts, evaluation measures often vary in hardware used or offer generalized performance metrics. OT systems often possess specified technologies and have unique requirements for efficient operation. Due to these concerns, this work outlines an OT-relevant model of evaluation for integrity enhancing protocols, building on existing observations and frameworks.

Embedded device performance evaluation has long been a popular topic in the research community, particularly in the field of security. However, these efforts commonly focus on Internet-of-Things (IoT) systems, and typically include IoT platform performance evaluations, physical layer security, and high-level protocol overhead. While these areas present important concerns for traditional IT-based IoT systems, OT architecture presents a separate set of challenges. OT systems are often even more resource constrained than traditional IoT endpoint devices, and traditional system failure states could be unacceptable due to real-time availability requirements. Fleet management of OT devices is far less involved than an IoT platform system, shifting the importance of performance to the endpoint nodes themselves. Drawing on these observations, this work scopes evaluation to OT Programmable Logic Controller based topologies, which account for a significant majority of modern OT systems and allow for straightforward transaction testing within a simulated OT system.

Certain performance metrics prove to be uniquely valuable in the evaluation of OT communication protocols. Running times of key generation, message encapsulation, and de-encapsulation have shown to be historically valuable in protocol evaluation, [27] and are of equal importance to time-critical OT systems. Key management hierarchies [28] have been shown to be an effective manner of visualizing key access and storage requirements in distributed systems. Such techniques may be leveraged in combination with analysis of long-term key storage requirements, which other researchers have noted as being highly variable between protocols. [29] Additional storage requirements, such as cryptosystem storage requirements, or hyperparameter value storage in the case of neural algorithms, must also be accounted for. Short term memory performance is additionally an area of concern and may even cause device failure in certain scenarios. [29] As with running times, memory usage may be subdivided into key generation and communication service requirements. Finally, the NIST security level of a given protocol is included, specifically the NIST PQC Security Levels. [31] The external evaluation of protocols provides value for AR systems, which require clear trusted confirmation of cryptographic hardness in protocol design.

As a result of this analysis, the following metric groups for our evaluative framework are identified:

- **Protocol Running Time:** This component of analysis includes real time, as well as asymptotic analysis for each protocol selected. Running time metrics for digital signature schemes may be further subdivided into key generation and inter device communication and are typically recorded in microseconds.

- **Endpoint Storage Requirements:** This component identifies both key and cryptosystem storage requirements on an endpoint device. Due to variable key lengths between protocols, coupled with often low storage capacities of OT environments, this is a significant metric in measuring the difficulty of practical implementation.
- **Protocol Hardware Performance:** This work leverages standard analytical tools to explore CPU and memory usage for protocol components. Due to the time-critical nature of OT systems, additionally investigate latency overheads or levels of system jitter associated with individual protocols.
- **Existing NIST Security Evaluation Level:** This work includes the relevant NIST security classification levels for each protocol tested, this serves as a relative indicator of overall protocol attack resistance and cryptographic hardness. When coupled with findings on performance, this component offers additional insight into overall protocol usefulness to OT security administrators. While these scores are not provided for all evaluated protocols, NIST classification parameters allows them to be easily derived based on protocol cryptographic hardness. We detail this further in Section **Error! Reference source not found.**.

We leverage a single virtualized testing environment for all evaluated protocols, to ensure observed differences may be accounted for by changes in protocol.

Attribute Group	Attribute	Metric	Notes
Protocol Running time	Key Generation Running Time	Microseconds (µs)	Measured in function call
	Key Generation Asymptotic Worst Case	Big O Notation (O(n))	Derived theoretically
	Key Encapsulation Running Time	Microseconds (µs)	Measured in function call
	Key Decapsulation Running Time	Microseconds (µs)	Measured in function call
Endpoint Storage Requirements	Key Storage Requirements	Bytes	Single key storage
	Cryptosystem Storage Requirements	Bytes	Algorithm component storage requirements
Protocol Hardware Performance	Cryptosystem Average CPU Utilization	Percentage	Average usage over a single transaction
	Cryptosystem Average Memory Usage	Percentage	Average usage over a single transaction
	Network Transmission Time	Milliseconds (ms)	Measured in function call
Existing Security Evaluation Level	NIST PQC Security Project Levels [31]	Integer Scale (1-5)	Protocol security in relation to classical protocols, includes classical and quantum resistance.

Table 2: Framework Metrics and Groups

3.3. Testbed Design and Construction

As detailed in section **3.2**, prior approaches have identified numerous metrics of evaluating protocol performance in system testbed scenarios. These approaches also leverage techniques such as hardware reuse allows for easy management of independent variables in an experiment, whether in a virtualized or physical system. Experimental design methodology is largely consistent between prior evaluative efforts. We draw on these established approaches in outlining our tools and techniques for protocol evaluation.

Our proposed design involves the usage of a fully virtualized testbed, which simplifies configurational complexity and improves workflows related to testing multiple protocols, leading to a greater degree of testing efficiency. Performance realism has been noted as a concern in similar experiments, [32] but due to our singular focus on individual protocol performance, this does not present an impediment. Core to our experimental configuration is the usage of the Minimega system [33] as a hypervisor management tool. Minimega was selected as a result of its rapid deployment capabilities, lack of configurational complexity, and data capturing tools. Minimega allows for inspection and `capture of network and file system activities on all virtualized endpoint machines, vastly simplifying performance measurement.

Minimega interacts directly with the KVM hypervisor and QEMU [34] emulation platform, allowing for compatibility with most Debian-based Linux systems. Minimega requires no external software stack or complex initial configuration, allowing for fast, efficient deployment of a wide array of experimental testbed scenarios. OpenVSwitch is leveraged for an internal switching stack, and VMbetter allows for the export of virtual hosts into many common disk image formats. Protonuke, a simple layer 3 traffic generation module, allows for diverse network conditions for experimentation. These features allow for the ease of recording and exporting scientific results, which in turn allows for the replicability of our protocol performance evaluation.

Virtualized system availability is a concern when experimenting with OT systems, as many OT providers have not release emulated versions of their software. However, our scope largely excludes us from this problem, as we specifically endpoint performance. Emulation tools such as QEMU enable the execution of commonly used OT software binaries such as ARM and MIPS, which we may leverage through Minimega. Endpoint emulation allows us to conduct a thorough evaluation of OT performance on certain integrity-enabling protocols. Future work may extend this evaluative platform as OT virtualization technology develops further. Alternatively, future work could leverage a hardware-in-the-loop (HiL) approach to support these devices.

4. PROTOCOL TESTING METHODOLOGY

Performance evaluation was conducted across a range of selected protocols as detailed in section 3.1. To effectively and accurately monitor performance across protocols, a single testing program was constructed, with implementations for each evaluated protocol. This tool was developed with a focus on performance and experimental scalability, formalizing tests as a single network transaction of message, using one of several available protocols. This approach allows for minimization of potential performance differences between publicly available implementations, especially those with graphical or network integration components. This section details both the nature of the testing program, the hardware leveraged in protocol evaluation, and the recorded data from testing.

4.1. Protocol Testing Tool

Given the network transaction focus of the proposed framework, the implemented testing tool is constructed as a network socket program focused on the sending and integrity verification of a single, variable length message. To support this goal, the testing tool is comprised of two components, a client and server program that may run on shared or separate nodes within a testing environment. The client machine implements the key generation, message encapsulation, and message sending components of a transaction, whereas the server machine receives, decapsulates, and verifies the message sent. Encapsulation is used in this context to refer to the preparatory steps taken before the sending of a verifiable message. For digital signatures this includes the encoding of a public key as well as the hex encoding of a signed message. Upon reception of these components, the server performs a message integrity verification and sends a response message to the client containing the success or failure of the integrity verification operation, in addition to the server-side performance metrics.

Both client and server components are implemented in C++ to support performance and protocol implementation requirements. Specifically, the BOTAN library is implemented for its native support of multiple protocols chosen for analysis, including RSA, multiple variants of ECDSA, SPHINCS+, and Dilithium. The open-source nature of BOTAN allows for the verification of protocol implementation, and modification if desired for a given protocol. When applicable, public key information is sent alongside the message for verification. No pre-knowledge concerning keys or cryptographic modules is used unless explicitly specified in protocol design. Performance metric recording is an additional core component of the testing tool, metrics are gathered in accordance with specified framework components. Key generation running time, content encapsulation, content decapsulation, key size, cryptosystem size, CPU usage, RAM usage, and RTT are all measured during program runtime. Results are exported to a CSV file once per run, with subsequent runs appending results to the same file, associated with the protocol chosen for the test. Metrics are gathered from a variety of sources; a separate program thread is launched at the start of testing to record CPU and RAM usage associated with PID using the Linux proc filesystem. Metrics associated with running time are calculated using the C++ chrono library, which records timing intervals for specific algorithmic functions in line with those defined in the framework.

4.2. Testing Conditions and Result Formatting

The initial round of testing was conducted on a single device, in which both client and server components were hosted locally. This approach was chosen for initial testing as to provide a clear baseline of internal protocol performance, before external network conditions were introduced on the virtualized testbed. Future work includes will extend these measurements to the proposed

Minimega testbed. The device leveraged for testing was an Ubuntu 20.04 endpoint system, the components of which include a 12th Gen Intel(R) Core i7-1265U10 Core processor, 16.0 GB of RAM, and 600 GB of system storage. Future testing is likely to include virtualized OT and IoT systems, for increased result applicability to the central theme of this work.

During program runtime, results are stored incrementally in a csv file, in which each protocol run appends a new line. Separate files are generated on a per protocol basis, and not erased until removed in the file system. This allows for the scripting of large-scale testing experiments, in which the testing program is called with command line arguments indicating the message to send and the protocol to use. The core workflow envisioned for analysis is that of a single message being sent and integrity-verified across two network nodes. As such, messages may be variable-length text files that are saved as strings and sent during a transaction. Upon completion of a test or scripted run, csv outputs are immediately viewable and may be imported or analyzed using external tools.

5. CURRENT RESULTS

The current results dataset consists of seven tested protocols: SECP256k1, BrainpoolP256R1, RSA, Dilithium, MAC, Sphincs+, and ElGamal. Two prime field curves are included in the initial round of testing a Koblitz curve (SECP256k1), and a standard Weierstrass curve (BrainpoolP256R1). Dilithium and Sphincs+ serve as two post-quantum implementations. MAC is additionally included as the only non-authenticity verifying protocol. Testing was conducted via a scripted setup as detailed in section 4.2, with 100 runs for each protocol. The resulting CSV files were programmatically allocated into a single dataset, which was imported into Microsoft Excel for analysis and visualization. A compiled form of averaged results per category may be seen in the Executive Summary. Timing metrics are recorded in MS, whereas spatial metrics are recorded in bytes. CPU and RAM measurements are recorded as percentages of total usage while the program was active i.e., CPU 2.3 = 2.3% usage over a testing operation.

Commented [VR1]: I moved this to the executive summary since the table went up there. We should remove this from here.

Immediate observations regarding known protocol strengths are evident from the resulting dataset, for example, in average case analysis, RSA shows to have the longest key generation running times by a significant margin. This constitutes expected behavior, given the poor performance scaling of RSA with larger key sizes. Additionally, note the faster key generation times of the simpler Koblitz curve ECC algorithm compared to the standard implementation. MAC predictable outperforms all others here, as the pre-shared architecture allows for simplistic key generation without public key architecture. While the inclusion of a single non digital signature algorithm may seem out of place, the goal of testing is to measure only integrity provisions across network transactions. Authenticity is often combined with integrity in modern protocol design, leading to the inclusion of protocols which fail to solely provide integrity. When targeted as a unique, single algorithm goal as is done in MAC, results show that performance improves drastically. These results may be visualized below in Figure 12.

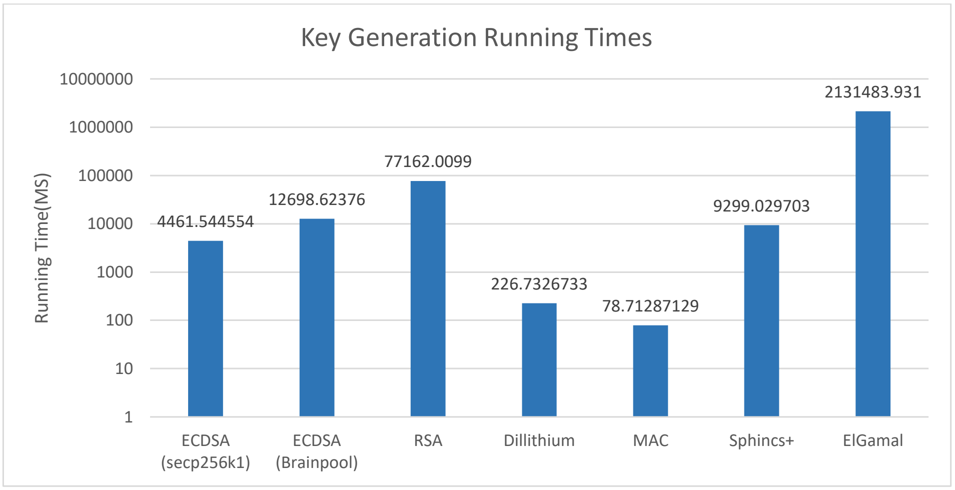


Figure 12: Average Key Generation Running Times

More surprising insights were gained through the usage of framework specific components, such as comparatively high memory usage for the standard Brainpool curve algorithm. As well as the CPU intensiveness of RSA and Sphincs+ in the dataset. These results allow for unique insights into the process of integrity protocol implementation within the OT domain. The high memory usage likely excludes certain variants of elliptic curves, depending on the system. Figure 14 visualize CPU and memory performance metrics. Algorithms with higher CPU requirements, such as Sphincs+ may present performance challenges depending on system scalability needs. An unmeasured statistic was that of signature size for the digital signature algorithms, an area in which Sphincs+ performed particularly poorly. This is reflected in the network transmission time statistics seen above in Table 1. The high storage requirements of Sphincs+ likely make it an untenable solution for many OT systems, as is reflected in figure 15.

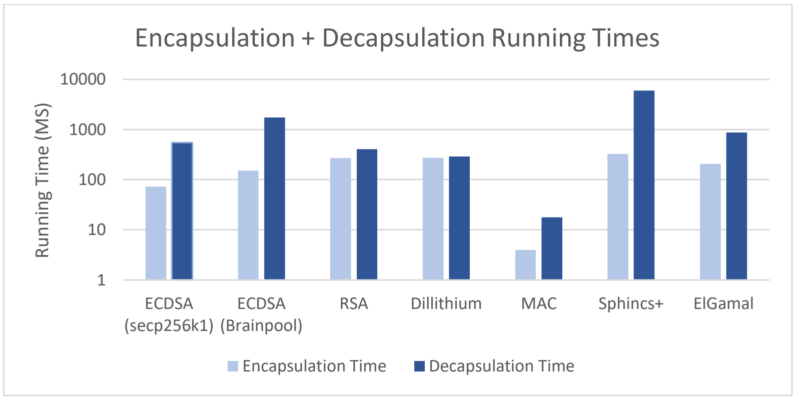


Figure 13: Average Encapsulation & Decapsulation Running Times

Unsurprisingly, one of the best evaluated algorithms was MAC, due to its lack of authenticity provision and simpler key generation procedures. The lack of a signing operation allows for faster key and message transmission and verification, which could be increased further with the inclusion of pre-shared keys, as is done on many existing implementations of MAC. The results of MAC support its usage in OT systems, in which fast network transactions and endpoint performance are prioritized for system operation. The performant results further support the notion of exploring the de-coupling of integrity from other security goals, such as authenticity and confidentiality due to strong protocol performance in high impact metric groups. Figure 16 displays a cumulative distribution of all MAC RTT values, in which performance remains under 1000 MS for upwards of 90% of trials, making the second strongest performing algorithm in this category.

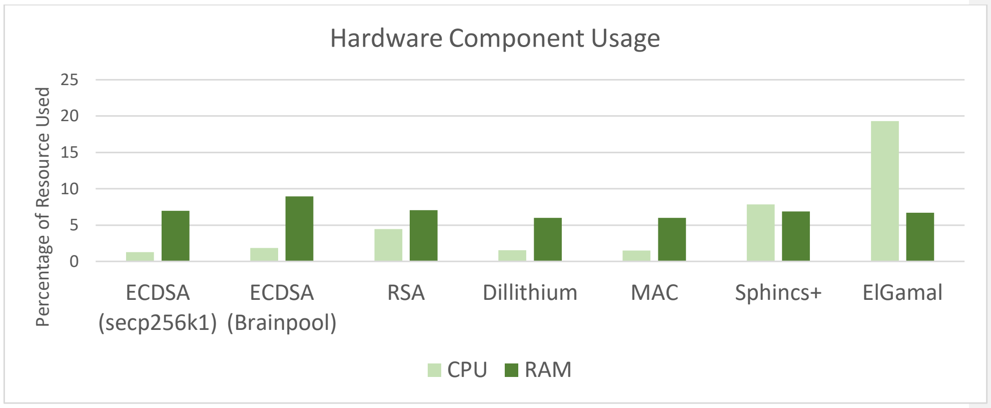


Figure 14: Average Hardware Component Usage

All tests were performed across a local socket connection on the endpoint device, even in these conditions, some degree of transmission variability was observed. Additional network constraints, such as those detailed in the proposed testing environment would provide additional insight into protocol performance in real-time OT systems. Minimega traffic generation using Protonuke could be leveraged to provide real-time network background data. Protocol performance differences are likely to be brought further into view in such a setting, particularly in instances in which retransmission of data packets is required. Key and signature sizes are likely to become even more significant in such a setting.

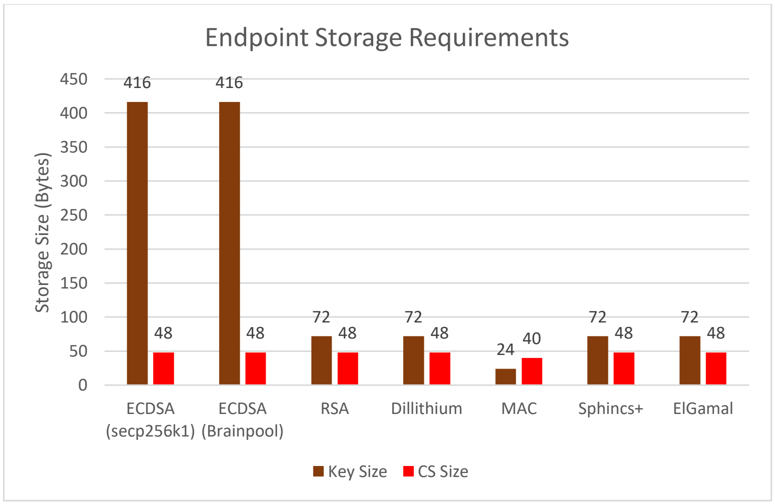


Figure 15: Storage Component Sizes

Existing security quantifiers represent the final part of the framework analysis. NIST PQC Security categories are based off relative hardness to classical cryptographic protocols, specifically AES. Post Quantum resistance is additionally included when applicable. All protocols are evaluated based on the described set of input parameters in section 3.2. Results are predictable based on protocol architecture and initialization specifications. This information is viewable below in Table 3

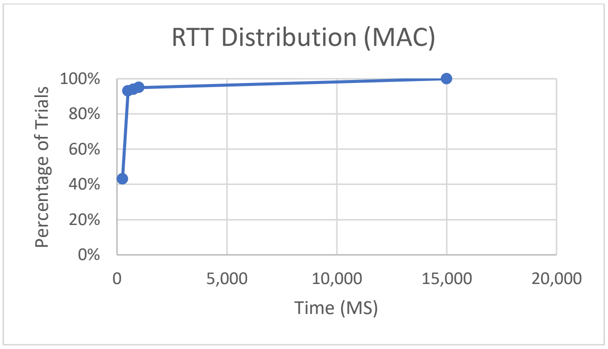


Figure 16: Cumulative Distribution of MAC RTT

Protocol Name	NIST PQC Security Level	Quantum Resistance
ECDSA (SECP256k1)	3(Comparable to AES-192)	None
ECDSA (Brainpool256r1)	3(Comparable to AES-192)	None
RSA	2 (Comparable to AES-128)	None
Dilithium	5 (Comparable to AES-256)	Strong
MAC	2 (Comparable to AES-128)	Weak (Vulnerable to Grover's algorithm)
Sphincs+	5 (Comparable to AES-256)	Strong
ElGamal	2 (Comparable to AES-128)	None

Table 3: NIST PQC Ranking Applied to Selected Algorithms

6. CONCLUSION

This report presents the motivation and current landscape of integrity providing protocols. Integrity provision provides organizational value across a wide range of scenarios, encompassing OT environments. Many protocols are available for use, each containing unique advantages. Section 2 serves as a survey of integrity protocols providing a background of these protocols. Several categories of protocols are described to provide a thorough investigation of what is available. This background is given to provide documentation of characteristics and architecture of protocols and their categories. Section 3 discusses the evaluation framework created for analyzing integrity protocols. The framework consists of various metrics such as protocol running time, CPU, and memory utilization. These metrics help to categorize protocols and provide operators with information on their resource utilization. It additionally gives the protocols chosen for analysis and the reasoning behind their selection.

The team implemented elliptic curve algorithms, MACs, Sphincs+, Crystals-Dilithium, and RSA. In section 4 the testing methodology for this work is presented. The testing program is detailed in this section. The client-server architecture of the system is discussed and allows for the encapsulated data transfer to occur. The hardware used to run the tests are also described to give the reader an understanding of the computational resources the algorithms were run on. Section 5 outlines the evaluation framework results found from implementing various protocols. Depending on the algorithm, integrity, authenticity or confidentiality can be incorporated into a system. Having all three of these features can be resource intensive. The results section provides an analysis of the various selected protocols. The results show that the integrity specific protocols such as MAC take less time for key generation, encapsulation and decapsulation. They also consume less memory and processing power. If the data can be passed in clear text with the incorporated MAC, operating cost can be reduced in comparison to many other protocols. The enhanced security of quantum-resistant protocols such as Sphincs+ can be visualized through the increased resource utilization of these algorithms. This is due to the increased complexity of the trapdoor function necessary to provide quantum-resistance, in addition to large signature sizes generated by such algorithms. Overall, the results showcase the diverse range of integrity protocols and provide operators with a roadmap for what they should choose when devising their networks.

Regarding protocol selection, these results demonstrate superior performance metrics for MAC across multiple evaluative categories related to running times and hardware usage. This suggests that future implementations of resource-constrained devices could be better served to move past the digital certificate model for improved performance within the domain of classical integrity providing protocols. Additionally, within systems that require post-quantum attack resistance, lattice-based cryptosystems offer better performance and storage requirements than a hash-based counterpart. Additionally, multi-signature algorithms tend to perform similarly to other classical digital signature systems on a per-endpoint basis and should likely only be incorporated when specifically needed for certain system architectures. Regarding specific device implementations, further testing would be required to determine device specific protocol suitability. However, this work shows Koblitz curve ECDSA and MAC as performant options for endpoint OT devices. Additionally, Sphincs+ should likely not be used in time sensitive OT environments, due to delay induced by high RTT measurements.

Within the domain of Nuclear AR systems, these results present an actionable framework for performance preservation in AR systems seeking to incorporate data integrity guarantees into component communications. Such systems would be well served to leverage non-public key protocols when available. MAC presents the most compelling choice for classical integrity provision in AR systems. For systems which require resilience to post-quantum attacks, Dilithium and other lattice-based schemes offer comparatively high-performance solutions on OT devices. Finally, these results additionally show several options for integrity and authenticity provisions in the analyzed public key algorithms, which may be used as needed to produce similar observations.

A DCSA is a necessary component of any secure reactor system. These systems must be able to deny parties from performing certain actions such as the alteration of critical data. This can be done through secure elements such as a Trusted Platform Module (TPM), which is able to store keys securely on a device and an integrity protocol, which uses those keys to verify that data has not been altered during communication. The incorporation of integrity protocols into the DCSA prevents an attacker from distorting information being sent across network infrastructure. These tools can enhance the protection of reactor components, functions and processes. Various algorithms are presented in this work that can accomplish this task, but for integrity purposes the results show that MAC provide the best performance.

Future work is likely to incorporate additional protocols or framework metrics. While initial work indicates a set of promising protocols for OT system adoption, further work is needed to calibrate this list for specific scenario optimization. Additional authenticity or confidentiality requirements may be present in some settings, which would require separate validation. Novel framework metrics could be included, to further tailor results to specific environments or operational paradigms. End-to-end network systems often request the retransmission of data at the application layer of the network stack. Evaluation of protocol performance in such re-transmission scenarios presents an OT-Specific evaluative future goal, as retransmission performance may be observed in network and application layer metric gathering. Finally, future deployments of integrity provisions in OT physical systems may be observed, to monitor physical implications of integrity provision being present in OT systems.

7. REFERENCES

[1] M. S. Sonkor and B. G. d. Soto, "Operational Technology on Construction Sites:A Review from the Cybersecurity Perspective," *Journal of Construction Engineering and Management,* 2021.

[2] G. Assenza and R. Setola, "OPERATIONAL TECHNOLOGY CYBERSECURITY: HOW VULNERABLE IS OUR CRITICAL INFRASTRUCTURE?," *INTERNATIONAL SCIENTIFIC JOURNAL OF THE MINISTRY OF DEFENSE OF THE REPUBLIC OF NORTH MACEDONIA,* 2019.

[3] D. D. Clark and D. R. Wilson, "A Comparison of Commercial and Military Computer Security Policies," *IEEE Symposium on Security and Privacy,* 1987.

[4] A. M. Qadir and V. Nurhayat, "A Review Paper on Cryptography," *International Symposium on Digital Forensics and Security,* 2019.

[5] M. C. B. C. M. H. I. B. a. K. H. Alexandre Meylan, "A Study on the Use of Checksums for Integrity Verification of Web Downloads.," *ACM Trans. Priv. Secur. 24,* p. Article 4, 2021.

[6] C. P. W. a. E. Z. Gopalan Sivathanu, "Enhancing File System Integrity Through Checksums," *Technical Report FSL-04-04,* 2004.

[7] D. A. McGrew and J. Viega, "The Galois/Counter Mode of Operation (GCM)," *NIST Special Publication 800-38D,* 2007.

[8] A. J. Ordonez and B. D. Gerardo, "Digital Signature with Multiple Signatories Based on Modified ElGamal Cryptosystem," 2018. [Online].

[9] T. Yang and Y. Zhang, "Digital Signature Based on ISRSAC," *ieee signal processing,* 2021.

[10] M. Kenta, Y. Naoto and O. Shingo, "Secure Routing Protocols for Sensor Networks," *2015 IEEE Trustcom/BigDataSE/ISPA,* 2015.

[11] D. Johnson , A. Menezes and S. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)," 31 January 2014. [Online]. Available: https://link.springer.com/article/10.1007/s102070100002. [Accessed 1 January 2024].

[12] I. K. Dutta, B. Ghosh and M. Bayoumi, "Lightweight Cryptography for Internet of Insecure Things: A Survey," IEEE, 14 March 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8666557. [Accessed 1 January 2024].

[13] N. Kumar and S. Aggarwal, "Advances in Computers: Digital Signatures," ScienceDirect, 29 September 2020. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0065245820300590. [Accessed 1 January 2024].

[14] M. Adalier and A. Teknik, "Efficient and Secure Elliptic Curve Cryptography Implementation of Curve P-256," NIST, 2015. [Online]. Available: https://csrc.nist.gov/csrc/media/events/workshop-on-elliptic-curve-cryptography-standards/documents/papers/session6-adalier-mehmet.pdf?ref=https://githubhelp.com. [Accessed 1 January 2024].

[15] M. B. Niasar, R. E. Khatib, R. Azarderakhsh and M. Mozaffari-Kermani, "Fast, Small, and Area-Time Efficient Architectures for Key-Exchange on Curve25519," IEEE, 7 June 2020. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9154488?casa_token=Z8V4mffdHEIAAAAA:fXopJ1HDlzwFDmHtCngvwlhr1feRzsQc9R3PjcNVzNxS_Gxfhg2dEFD-OKKF2u5W8opSj0IN0w. [Accessed 1 January 2024].

[16] Z. Liu, P. Longa, G. C. C. F. Pereira, O. Reparaz and H. Seo, "FourQ on Embedded Devices with Strong Countermeasures Against Side-Channel Attacks," IEEE, 30 January 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8274963?casa_token=bxXvpKYe31YAAAAA:xV8ZWYjI4-mr5B68ADqvaSs9eDKF9hGorpnaxuldglcma-wtIWUMl9gzdM1VmQ87JLS5rIStlQ. [Accessed 1 January 2024].

[17] A. A. Yavuz and M. O. Ozmen, "Ultra Lightweight Multiple-Time Digital Signature for the Internet of Things Devices," IEEE, 15 July 2019. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8762164?casa_token=d0rErVsORCoAAAAA:EdEl73_M4tkQVS1M7kgr9t-LL3DbrAYE8FQ9DnUUpS6DHRsp-dlpFIHj5OWTw_KTnfS2Kp7PgQ. [Accessed 1 January 2024].

[18] R. Alvarezq, J. Santonja and A. Zamora, "Algorithms for Lightweight Key Exchange," Springer Link, 3 November 2016. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-48799-1_58. [Accessed 1 January 2024].

[19] J. H. Faruk, S. Tahora, M. Tasnim, H. Shahriar and N. Sakib, "A Review of Quantum Cybersecurity: Threats, Risks, and Opportunities," *International Conference on AI in Cybersecurity (ICAIC),* 2022.

[20] Z. Liu, K.-K. R. Choo and J. Grossschadl, "Securing Edge Devices in the Post-Quantum Internet of Things Using Lattice-Based Cryptography," *IEEE Communications Magazine,* 2018.

[21] E. Orsini, "Hybrid lattices and the NTWO cryptosystem," Department of Mathematics at University of Trento, University of Trento, 2011.

[22] A. Kipnis, J. Patarin and L. Goubin, "Unbalanced Oil and Vinegar Signature Schemes," *Eurocrypt,* 1999.

[23] N. Kundu, S. K. Debnath, D. Mishra and T. Choudhury, "Post-quantum digital signature scheme based on multivariate cubic problem," *Journal of Information Security and Applications,* 2020.

[24] D. Kales and G. Zaverucha, "Forgery Attacks on MQDSSv2.0," *Microsoft Research Article,* 2019.

[25] B. Liu, X. L. Yu, S. Chen, X. Xu and L. Zhu, "Blockchain Based Data Integrity Service Framework for IoT Data," *IEEE International Conference on Web Services,* 2017.

[26] A. Khalid and M. S, "Lattice-based Cryptography for IoT in A Quantum World: Are We Ready?," *International Workshop on Advances in Sensors and Interfaces,* pp. 194-199, 2019.

[27] F. Lauterbach, P. Burdiak, F. Richter and M. Voznak, "Performance Analysis of Post-Quantum Algorithms," *Telecommunications Forum,* no. 9, 2021.

[28] N. Kumar and A. Mathuria, "Comprehensive Evaluation of Key Management Hierarchies for Outsourced Data," *Cybersecurity,* vol. 2, no. 1, 2019.

[29] A. Fournaris, G. Tasopoulos, M. Brohet and F. Regazzoni, "Running Longer to Slim Down: Post Quantum Cryptography on Memory Constrained Devices," *IEEE International Conference on Omni-layer Intelligent Systems,* 2023.

[30] "Security Requirements for Cryptographic Modules," National Institute of Standards and Technology, 22 March 2019. [Online]. Available: https://csrc.nist.gov/pubs/fips/140-3/final. [Accessed 29 2 2024].

[31] "Post Quantum Cryptography," National Institute of Standards & Technology, 3 January 2017. [Online]. Available: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/evaluation-criteria/security-(evaluation-criteria). [Accessed 29 February 2024].

[32] M. Hibler, R. Ricci, L. Stoller, J. Duerig, S. Guruprasad, T. Stack, K. Webb and J. Lepreau, "Large-scale Virtualization in the Emulab Network Testbed," University of Utah, Salt Lake City, UT, 2008.

[33] J. Crussell, J. Erickson, D. Fritz and J. Floren, "minimega v3.0," Sandia National Labs, Albuquerque, NM, 2015.

[34] N.A., "QEMU: A Generic and Open Source Machine Emulator and Virtualizer," QEMU, 4 March 2024. [Online]. Available: https://www.qemu.org/. [Accessed 5 March 2024].

[35] R. Gennaro, C. Gentry and B. Parno, "Non-Interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers," *IACR,* 2010.

[36] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky and D. Werthimer, "SETI@home: An Experiment in Public-Resource Computing," *Communications of the ACM,* 2001.

[37] S. Goldwasser, S. Micali and C. Rackoff, "The Knowledge Complexity of Interactive Proof Systems," *SIAM Journal on Computing,* 1989.

[38] X. Yu, Z. Yan and A. Vasilakos, "A Survey of Verifiable Computing," *Mobile Networks and Applications,* 2017.

[39] D. Fiore, R. Gennaro and V. Pastro, "Efficienty Verifiable Computation on Encrypted Data," *Communications of the ACM,* 2014.

[40] I. Zikratov, A. Kuzmin, V. Akimenko, V. Niculichev and L. Yalansky, "Ensuring data integrity using blockchain technology," *Conference of Open Innovations Association (FRUCT),* 2017.

[41] E. Reilly, M. Maloney, M. Siegel and G. Falco, "An IoT Integrity-First Communication Protocol via an Ethereum Blockchain Light Client," *International Workshop on Software Engineering Research & Practices for the Internet of Things (SERP4IoT),* 2019.

[42] H. Han, S. Fei, Z. Yan and X. Zhou, "A Survey on Blockchain-Based Integrity Auditing for Cloud Data," *Digital Communications and Networks,* 2022.

[43] C. Stach, C. Gritti, D. Przytarski and B. Mitschang, "Trustworthy, Secure, and Privacy-aware Food Monitoring Enabled by Blockchains and the IoT," *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops),* 2020.

[44] NIST, "Digital Signatures," 4 January 2017. [Online]. Available: https://csrc.nist.gov/projects/digital-signatures.

[45] NIST, "NIST Selects 'Lightweight Cryptography' Algorithms to Protect Small Devices," 7 February 2023. [Online]. Available: https://www.nist.gov/news-events/news/2023/02/nist-selects-lightweight-cryptography-algorithms-protect-small-devices.

[46] C. Kumar, S. S. Prajapati and R. K. Verma, "A Survey of Various Lightweight Cryptography Block Cipfers for IoT Devices," 23 December 2022. [Online]. Available: https://ieeexplore.ieee.org/document/10080556/authors#authors.

[47] S. Jhajharia, S. Mishra and S. Bali, "Public Key Cryptography using Neural Networks and Genetic Algorithms," *IEEE ,* 2013.

[48] X. Hao and W. Ren, "Asymmetric cryptographic functions based on generative adversarial," *elsevier,* pp. 243-253, 2021.

[49] I. Meraouche and S. Dutta, "Learning asymmetric encryption using adversarial neural networks," *Elsevier,* 2023.

[50] A. Jain and J. Singh, "Improved Recurrent Neural Network Schema for Validating," *mathematics,* 2022.

[51] A. Ahmed, F. Abdullatif and T. Hasa, "Generating and Validating DSA Private Keys from Online Face," *International Journal on Advanced Engineering Information Technology,* 2019.

[52] I. Goodfellow and Y. Bengio, Deep Learning, 2019.

[53] H. Badr, "Instant-Hybrid Neural-Cryptography (IHNC) based on fast machine," *Neural Computing and Applications,* 2022.

[54] C. I. Rene, N. Katuk and B. Osman, "A Survey of Cryptographic Algorithms for Lightweight Authenticaiton Schemes in the Internet of Things Environment," IEEE, 09 December 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9970015. [Accessed 1 January 2024].

[55] G. Mao, Y. Liu, W. Dai, G. Li, Z. Zhang, A. H. F. Lam and R. C. C. Cheung, "REALIZE-IoT: RISC-V-Based Efficient and Lightweight Public-Key System for IoT Applications," IEEE, 2020 July 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10189076. [Accessed 1 January 2024].

[56] Y. Song, X. Hu, W. Wang, J. Tian and Z. Wang, "High-Speed and Scalable FPGA Implementation of the Key Generation for the Leighton-Micali Signature Protocol," IEEE, 27 April 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9401177?casa_token=MCdpIwJhw1sAAAAA:SpOc2zFzfnc87hKkgCOaJI3bRskbFSNT8ShPUtbtwe6AGoeeahCBtOTVwzpsvHM7ampEhcjLOg. [Accessed 1 January 2024].

[57] P. Longa and R. Cruz, "Microsoft/FourQlib," GitHub, [Online]. Available: https://github.com/Microsoft/FourQlib. [Accessed 1 January 2024].

[58] "Signer Efficient Multiple-Time Elliptic Curve Signature," Github, [Online]. Available: https://github.com/ozgurozmen/SEMECS. [Accessed 1 January 2024].

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Lon Dawson	8851	ladawso@sandia.gov
Ben Cipiti	8845	bbcipit@sandia.gov
Technical Library	1911	sanddocs@sandia.gov

Email—External

Name	Company Email Address	Company Name
Katya Le Blanc	katya.leblanc@inl.gov	Idaho National Laboratory

This page left blank