sensors

MDPI

Article

Secure Route-Obfuscation Mechanism with Information-Theoretic Security for Internet of Things

Abid Rauf [1,*], Zhaohong Wang [2], Hasan Sajid [3] and Muhammad Ali Tahir [1]

[1] National University of Sciences and Technology (NUST), School of Electrical Engineering and Computer Science, Islamabad 44000, Pakistan; ali.tahir@seecs.edu.pk
[2] Department of Electrical and Computer Engineering, California State University, Chico, CA 90802, USA; zwang25@csuchico.edu
[3] National University of Sciences and Technology (NUST), School of Mechanical and Manufacturing Engineering, Islamabad 44000, Pakistan; hasan.sajid@smme.nust.edu.pk
* Correspondence: abid.rauf@seecs.edu.pk

Received: 30 June 2020; Accepted: 25 July 2020; Published: 29 July 2020

Abstract: As accessibility of networked devices becomes more and more ubiquitous, groundbreaking applications of the Internet of Things (IoT) find their place in many aspects of our society. The exploitation of these devices is the main reason for the cyberattacks in IoT networks. Security design is still an open problem and a crucial step in making IoT applications successful. In dicey environments, such as e-health, smart grid, and smart cities, real-time commands must reach the end devices in the scale of milliseconds. Traditional public-key cryptosystem, albeit necessary in the context of general Internet security, falls short in establishing new session keys in the scale of milliseconds for critical messages. In this paper, a systematic perspective for securing IoT communication, specifically satisfying the real-time constraint against certain adversaries in realistic settings. First, at the network layer, we propose a secret random route computation scheme using the software-defined network (SDN) based on a capability scheme using the network actions. The computed routes are random in the eyes of the eavesdropper. Second, at the application layer, the source breaks command messages into secret shares and sends them through the network to the destination. Only the legitimate destination device can reconstruct the command. The secret sharing scheme is efficient compared to PKI and comes with information-theoretic security against adversaries. Our proof formalizes the notion of security of the proposed scheme, and our simulations validate our design.

Keywords: information-theoretic security; multipath routing; Internet of things

1. Introduction

The fast expansion of network-enabled devices makes our lives more convenient than ever. From smart homes to smart cities, an upgrade of network-connected devices is everywhere. Thus, an era of the IoT has come. Some IoT applications are time-critical. For example, in critical infrastructures such as the smart grid, the utility should deliver controlling commands to end-user devices at the scale of milliseconds [1]. In IoT surveillance systems, the detection of intruders should also trigger the alert in milliseconds. Mainly, concerns are raised over the security of the IoT [2].

IoT devices can range from medical devices, smart home sensors, autonomous vehicles as well as neuclear technology. Due to the light nature of IoT, it is vulnerable to cyber-attacks as communication is mostly done without confidentiality and authenticity.

An essential aspect of the success of IoT applications is information security. The receiver node needs to authenticate the sender node, and the messages themselves should be appropriately encrypted

against illegitimate adversaries. In the context of general internet communication, the public key infrastructure (PKI) has become the de facto security mechanism for authentication and encryption. However, it is not always a good fit for PKI in IoT due to its light nature—computing resources and energy storage. First, nodes in IoT usually have limited computing resources compared to general-purpose computers. Running PKI algorithms will deplete the computing resources as well as battery energy. Second, the encryption of public-key cryptosystems expands the size of messages to thousands of times. Consequently, much-expanded ciphertext increases the use of bandwidth. In turn, the inefficiency in the communication and decryption makes PKI so slow that it cannot satisfy real-time requirements of critical IoT applications [1].

With the advent of quantum computers, mathematically hard problems that are almost impossible for classical digital computers become much easier. Mathematical hard problem based cryptographic primitives such as the PKI schemes, are vulnerable to computing machines with high power [3]. With the rise of quantum computing platforms, the continuous usage of conventional Public key primitives established hundreds of years ago, has opened doors for attackers to crack the implementation [4]. A classical encryption can exponentially decrypted using the quantum computers [4].

Considering the ever more critical IoT applications in the critical domains with time constraints on message passing, we propose a secure message delivery framework with information-theoretic security in this paper. Unlike the PKI framework, we endeavor to guarantee real-time message passing for time-critical scenarios and to relieve the assumption on the computational power of adversaries.

Latest advancements in the field of computer networks has introduced a novel generation of software-defined network (SDN), known as the "SDN Controller", which permits to manipulate the general performance of the network [5].

The facilitation of strategies by the controller such as, responses to security threats, fine-grained traffic filters, and effective depolyment of security policies on the fly. Our proposed framework makes use of SDN to mitigate the vulnerabilities. The controller is central and has visibility over its domain. Controllers can manage the security of the IoT architecture.

In Figure 1 shows SDN architecture and highlight security vulnerabilities. The SDN controller is the key part of the control plane, and it is responsible to develop interfaces to other SDN planes. The SDN controller interconnect Application Programming Interface (API) with application plane through Northbound Interface and enables the communication between switch and controller through a Southbound interface in data plane. Open Networking Foundation maintained the essential protocol, i.e., OpenFlow used the Southbound interface. Use of SDN technology in IoT has open the door of deployment of services by decoupling of business and data planes. Due to the fully distributed nature of the control, new services like IoT in a fully optimized manner are deployed.

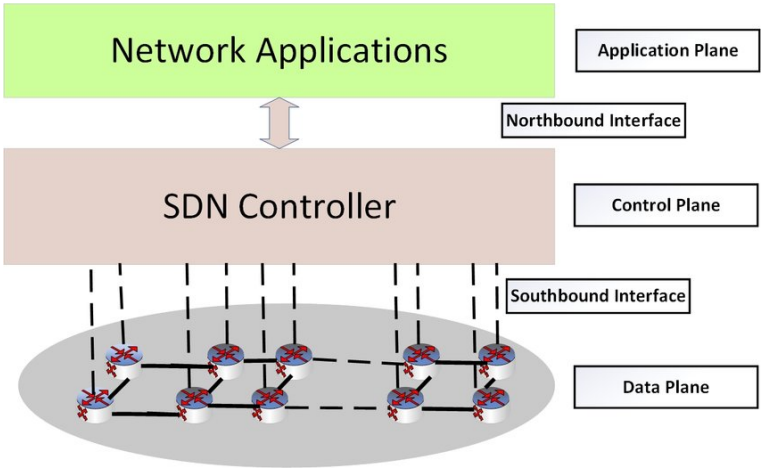

Figure 1. Software-defined network (SDN) architecture.

Due to mixture of various components in the SDN architecture, and communication between these components, not only the efficiency and reliability, security is a very important aspect. Malicious intruders attack the weaknesses, by means of taking control of the controller in an SDN network. The change of routes by changing the flow table entries of devices leads to the Man in the Middle Attack (MITM).

With all the above observations, our unique contribution in this domain is as follows. Firstly, we have proposed a framework with SDN controlling secret shares is the first novel framework utilizing secret schemes for traffic obfuscation. We make the routing paths random from time to time employing SDN control. Consequently, the routes become unpredictable in the eyes of an eavesdropper. Intuitively, the routes look random to the eavesdropper—the security relates to randomly guessing the ever-changing correct route out of a large set of routes. Second, we turn to information-theoretic secure encryption schemes, in contrast to the public key scheme which is not suitable for real-time applications. Therefore, intercepting shares less than the threshold will never reveal the original message. The formal proof is summarized in the paper. Third, using the well-established Shamir's secret sharing scheme, even if a share is intercepted and altered by an adversary, the receiver node will detect the alteration at the reconstruction stage. Furthermore, we proposed a node capability-based mechanism that supports the routing algorithm in maintaining efficiency. Our framework comes with security proof and simulation to validate our design.

Next, we formulate our design problem as well as the adversary model.

Problem Statement and Adversarial Modeling

We consider IoT scenarios that are organized as IP networks. Any end-users are associated with end node devices, i.e., the IoT devices. The communication between nodes is through routers and gateways. Refer to Figure 1, the routers, and gateways form the backbone of the IoT network. The communication in the IoT scenario happens as follows. The administrator or an end node may be sending real-time controlling commands to another node. The end node will find one of the nearby routers to relay its message. The routers in between form necessary routes to further deliver the message to its destination. The SDN controller is responsible for figuring out possible routes for the messages to pass through routers.

While the scenario is general IoT, we focus on information-theoretic security against adversaries modeled as follows. We assume that an adversary may intrude into one of the routing paths in the IoT scenario. They may passively spy on the messages sent through the communication lines and may attempt active assaults such as the replay assaults and DoS assaults, and they may actively tamper encrypted communication. The adversary mainly tries to decrypt the messages, and to deduce the network's process by finding the opportunities of launching sophisticated attacks.

The overall idea of our proposed framework is to obfuscate traffic patterns to the adversary and encrypting messages, so the adversary cannot decipher the message if the adversary only has access to a few routing paths. We do no longer assume the adversaries' computational capability. The reason is that adversaries with quantum computing power should be taken into consideration in the design of securing light weighted IoT devices.

In Section 2 we have reviewed all the important aspects related to our design. Section 3 describes our detailed proposed framework. Section 4 provides security proof for our framework. Section 5 presents the simulations that validate our design. Finally, Section 6 concludes the paper.

2. Related Work

Our framework will utilize several primitive building blocks. The first one is the SDN controlled routing. The area of routing in SDN has been aggressively explored. The SDN controller can collect link states from SDN-enabled routers to form a complete interpretation of the network. [6]. Therefore, the SDN controller helps to compute the shortest routing paths for nodes in the network [7]. The SDN

could be deployed to secure the entire network because it has the role of resource allocation, especially the placement of secure-related middleboxes [8].

A work that is particularly relevant to our work is the routing paths construction in SDN. To address the issue of SDN routing, several studies has been carried out. Shin et al. proposed four basic routing algorithms to maximize the security in utilizing the resources in SDN and the idea of Network Services Virtualization (NSV) [9]. Shen et al. proposed a data structure to abate the tree and cost of recovery [10]. A unique cost prototype, which simulates the usage expenses, hyperlink resources, and nodes, to maximize the throughput in the network by Huang et al. [11]. As compared to them, we have focused more on the security using obfuscation of routing. Furthermore, the information-theoretic proof authenticates that the scheme is secure against the adversaries with unlimited power.

A segment routing algorithm is proposed by Lee and Sheu [12] with segment routing, which focuses on balancing the traffic load on different routes and reducing the size of the network packet. Lee et al [13] proposed a routing aware algorithm. Huang et al. [14] and Wan et al. [15] proposed an assessment system that focused on SDN routing. SDN based routing called RouteGuardian, was proposed by Wang et al. [16], which considered the competences of the SDN switch nodes. RouteGuardian supports dynamic routing, but the efficiency of the proposed scheme has high time complexity.

There are various proposals suggesting to change network configurations dynamically to mislead the attacker. Previously more focus was on changing the network architecture to obscure traffic [17], applying cryptographic primitives on packets, and randomly changing the IP addresses and TCP/UDP ports. The attack surface of SDN was discovered by Yoon et al. [18]. As compared to the work done in past, our work's prime focus is security and obfuscation based upon random routing path construction and information-theoretic security.

PKI is a mechanism developed involving the trust of third party mediation. High computation cost near to exponentiation is required to generate and verify the signature. Fouda et al. [19] proposed Diffie–Hellman along with a hashing-based authentication. In [20] among smart meters, a cryptographic identity based mutual authentication mechanism by Nicanfar et al. As, all these proposed schemes make use of PKI, they require exponential computational cost. In the potential threat of quantum computer adversary, all number-theoretic-based public key cryptosystems will lose their security. Researchers have worked to find alternative public-key primitives, such as lattice-based cryptography and code-based cryptography [21–24].

Alternatively, people use information-theoretic secure schemes to secure messages. In fact, messages are broken into secret shares, and a legitimate receiver of the message should collect enough shares before he/she could reconstruct the original messages [25,26].

3. Proposed Framework

Our proposed framework for IoT scenarios with timing constraints is described and illustrated in this section. We have described the frequently used symbols throughout this section in Table 1. The entities of our framework comprise three roles: nodes, routers, and SDN controller. The nodes, denoted as n_i, are IoT devices. They are resources limited in carrying out complicated cryptographic operations. The routers and the SDN controller are the backbone entities of the framework, similar to any IP networks.

Table 1. Commonly used symbols.

Symbol	Description
$S1...S12$	Routers
n_i	IoT nodes
T	SDN Topology
s	Input Message Packet
t^i	Node reliability status
β_i	Reliability Path List
τ_γ	Percentage of nodes in the path
t^1	Packet transfer probability by a node
t^2	Probability of a node not breaking a route
K	Total number of paths
γ_i	Route
d	Cost of the link
e	Link between a pair of router
C	Cost between a pair of routers
V	Set of nodes in the Topology
Γ	Set of neighbour routers
t	Number of shortest routes / shares
SSS	Shamir Secret Sharing
α_i	Secret Coefficients
λ	Session count
n_{total}	Packet count in λ sessions
n_{broken}	Broken session count in last λ sessions
n_γ	Node count in the path γ
n_{nodes}	Node count in the Network
H	Set of potential routers from among available routes
R_γ	Reliability of the path γ_i

The routers, denoted in lower-case letters as r, t, u, v, etc., are SDN-enabled and they send link states to the SDN controller, denoted as SDN. Therefore, the SDN controller gets the network topology and capability in terms of reliability of the switches as input, and it will compute routing paths for nodes [6]. The overall architecture of our proposed scheme is detailed in Figure 2.

Our framework will take the reliability of the switches discussed in Sections 3.1 and 3.2, which is determined by considering the actions of the network nodes proposed by [27]. The SDN topology shown in Figure 3 comprises of three logical and physical entities: controller, hosts, and switches.

Our first measure to protect the network traffic happens at the network layer. The SDN controller collects routing status information from the SDN-enabled routers. Then, the controller calculates multiple routes according to Algorithm 1. To obfuscate the traffic in the eye of the adversary, the SDN controller generates several routes for destination and receiver nodes in the network. It can randomly assign costs on each link so that each time the computed routes would be different. Without loss of generality, we denote any pair of communicating nodes as n_i and n_j. The SDN controller also decides the number of routes, denoted as $\gamma_1, \gamma_2, \gamma_3$, etc., between the pair. The randomization of the routes and the number of routes all increase the difficultly for the adversary to intercept and eavesdrop the messages between nodes n_i and n_j. The formal proof is summarized in Section 4.

To clarify the description of the routes computation, we suppose that the source end node n_i has its next-hop router to be r and the last router to the receiver end node n_j to be t. Router r could be one of the nearby routers available to n_i. Therefore, we can view the source as r and the destination as t. The job of the SDN controller is to identify a number of routes between r and t. The proposed algorithm is built upon the shortest paths pair routing proposed in [28]. The notations are as follows, H is the set of potential routers from among available routes, $\Gamma(u)$ is the set of neighbor routers of u, K represents routes, r represents source, t represents the destination, u and v are intermediate routers in the route from r to t and v is the predecessor of u, $d(x)$ stands for the cost of the link to router x, e stands for the link between a pair of routers (v, u), and $C(v, u)$ represents cost between routers v and u.

The method of calculating the routing paths are very different from the traditional networks. Because, SDN provides an upper-level vision of the network, that is the reason for controlling the network through software programming. The controller is strong enough to control the flow of packets in the network. With random assignments of link costs by the controller, each time the computed routes would be different. For this purpose, the controller must consider the capability of the network for the optimal delivery of packets across the nodes.

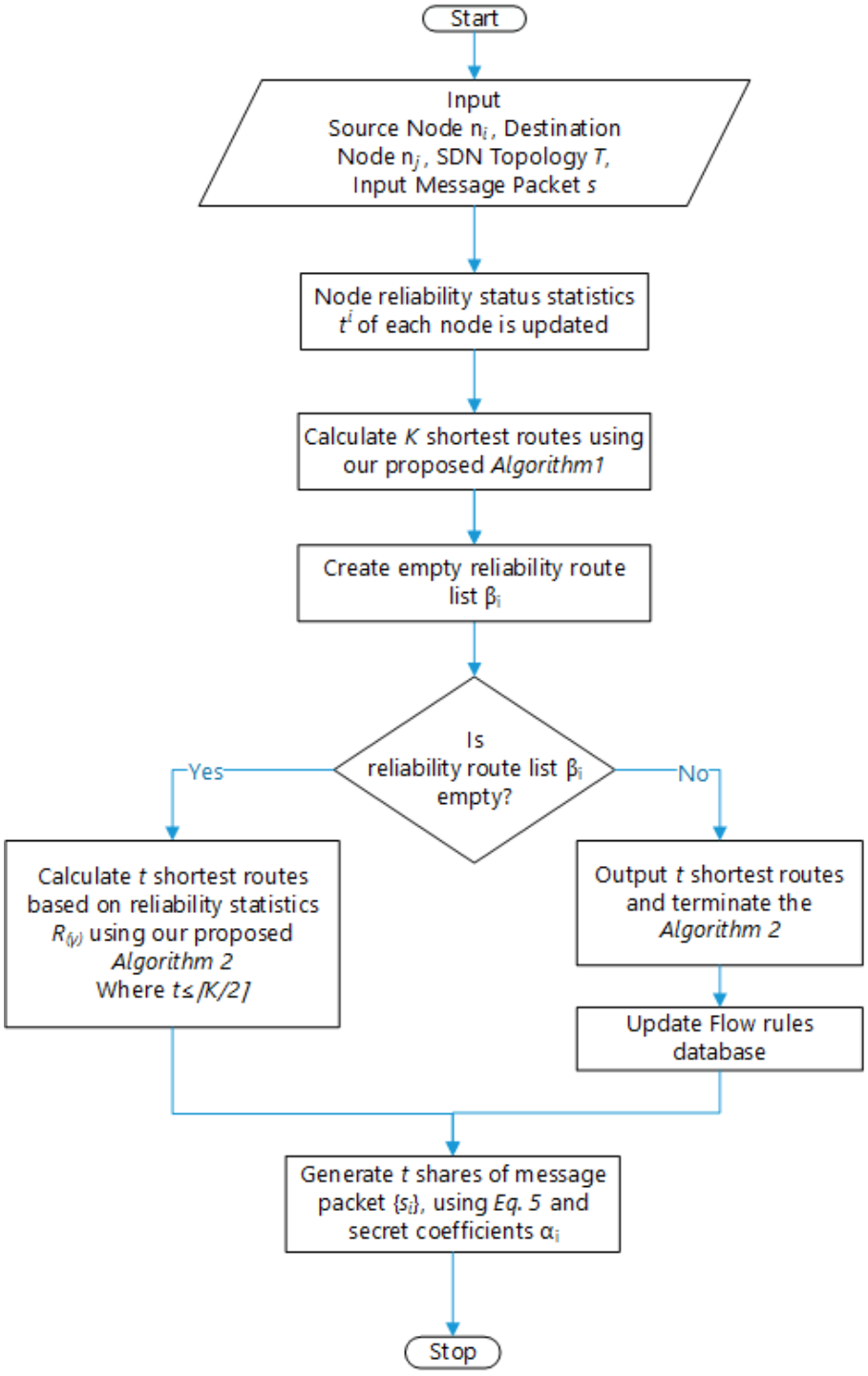

Figure 2. Flow chart for the mechanism of Obfuscation of routing paths in IoT-SDN.

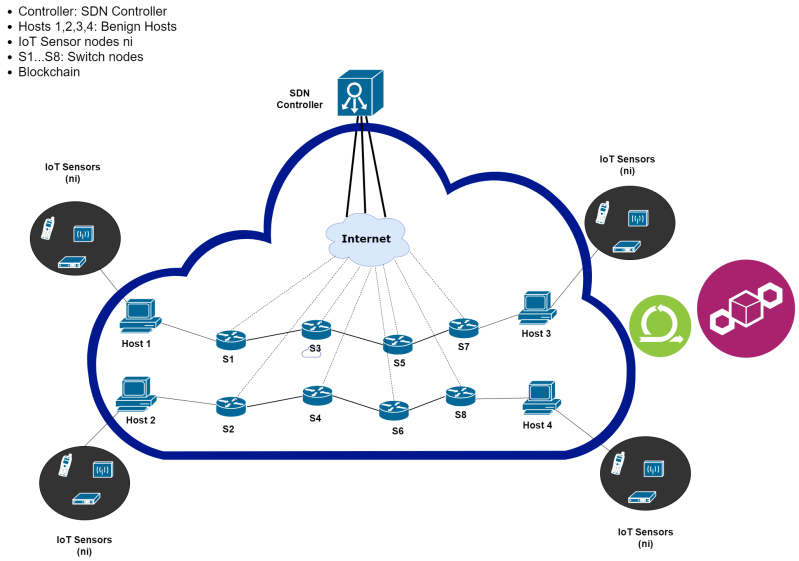

Figure 3. Routing path construction in IoT-SDN.

Algorithm 1: SDN Multiple Routes Computation.

Data: source r, destination t
Result: routes γ_i for $i \in \{1, ..., K\}$
initialization;
repeat
  K = random(seed);
until $K \geq 2$;
while $i < K$ **do**
  initialize vertice set V;
  $H \leftarrow V$;
  while $H \neq \emptyset$ **do**
    $u \leftarrow min\{d(x)\}, \forall x \in H$;
    Update $H \leftarrow u$;
    if $u = t$ **then**
      exit;
    else
      if $\Gamma(u) \neq \emptyset$ **then**
        foreach $v \in \Gamma(u)$ **do**
          if $v \neq w(u)$ **then**
            Replace $w(u)$ with the router of the lower cost;
          end
        end
      end
    end
  end
  Output γ_i;
  foreach $e_i = (u, v)_i \in \gamma_i$ **do**
    Update $e_i \leftarrow \overleftarrow{(v, u)}$;
    Update $C(u, v) \leftarrow -C(u, v)$;
  end
end

The routers send reliability statistics to the controller after a fixed small period. The controller will generate multiple paths based on these observations. It will output the probability of how a node will behave soon based on its performance factors.

3.1. Reliability of a Network Node in Terms of Probability

In order to calculate the reliability, we consider two of the three actions for a node $t_i^j, j = 1, ..., k$, i.e., $t_i^{(1)}$ (Packet transfer probability by a node) and $t_i^{(2)}$ (the probability of a node not breaking a route) of a network node proposed by Mahmoud et al. [27]. In this study, we will introduce another factor τ (the percentage of nodes in the path) to compute the nodes' reliability.

$t_i^{(1)}$: Packet transfer probabilty by a node: n_{relay}: count of transmitted packets in λ sessions.

$$t_i^{(1)} = \frac{n_{relay}}{n_{total}} \tag{1}$$

$t_i^{(2)}$: The probability of node not breaking a route:

$$t_i^{(2)} = 1 - \frac{n_{broken}}{\lambda} \tag{2}$$

τ_γ: Percentage of nodes in the path: Let n_γ be the total number of nodes in the path γ. Let n_{nodes} be the total network nodes.

$$\tau_\gamma = \frac{n_\gamma}{n_{nodes}} \tag{3}$$

3.2. Reliability of Network Path

By aggregating the reliability properties of a network nodes discussed in Section 3.1, the following equation will compute the reliability of the shortest routing path in the network. As all the actions described are independent from each other, the product of the probabilities can be used as an aggregate measure. Let $R_{(\gamma)}$ is the reliability of the path $\gamma_i, i = 1, 2, ..., k$.

$$R_{(\gamma)} = \left[\left\{ \sum_{j=1}^{k} \left(\prod_{\{i=1,2,...,n_\gamma\}} t_i^j \right) \right\} + (1 - \tau_\gamma) \right] / 3 \tag{4}$$

The reliability $R_{(\gamma)}$ is the probability of sending a packet based on the real-time parameters of the network path nodes.

3.3. Algorithm Based on Reliability of Network Path

After determining the number of routes K using Algorithm 1 as discussed above, the sender node n_i will run a secret sharing scheme to decompose its message into K shares and send over the network. To demonstrate the idea, we use Shamir's secret sharing (SSS) scheme, denoted as the (t, K) SSS explained as follows [29]. In SSS, the dealer/sender picks a secret s from a finite field F_m, where m is the field's size. The sender then decomposes the secret s into K shares $\{[s]_i^t, i = 1, \ldots, K\}$ such that any subset with less than the threshold $t \leq \lceil K/2 \rceil$ shares cannot give any knowledge to the attacker about the original secret s. Equations (5) and (6) describe the share generation by the sender and secret reconstruction by the legitimate receiver, respectively.

$$[s]_i^t \triangleq \sum_{j=1}^{K-1} \alpha_j i^j + s \bmod m, \tag{5}$$

$$s = \sum_{i \in \Gamma} \gamma_i [s]_i^t \bmod m, \tag{6}$$

Once the receiver receives the secret shares, the receiver will run the reconstruction formula to see the message. If the shares arrive without tampering, the reconstructed message will be meaningful, such as control commands. However, if the adversary corrupted shares, the reconstructed message will be like garbage. Thus, the corrupted message reveals that there must be an active attacker in the routes.

We will propose our network capability based random routing algorithm, in which we use SSS to break the message into K shares and send them using the t routing paths generated by Algorithm 1 using the reliability framework discussed in Sections 3.1 and 3.2.

In the proposed Algorithm 2, using the (t, K) SSS and reliability framework the sender acts as the dealer and breaks his/her secret real-time packets into shares and sends them over the network controlled by the SDN. The routes are random from time to time, and the shares follow the shortest paths determined by the SDN controller and the node capabilities to get to the legitimate receiver. No matter how computationally powerful an adversary is, it cannot break the secret message if it has the number of shares below the threshold.

Algorithm 2: SDN Capability-based secure Route Obfuscating Algorithm.

Data: routes γ_i for $i \in \{1, ..., K\}$, Message s
Result: maximum reliability routes β_i for $i \in \{\gamma_1, ..., \gamma_t\}$

1. Topology of the Network T detected by SDN
2. Status update of the nodes for the path $t_i^j, j = 1, ..., k, k \in Actions$ collected by Controller
3. **while** $i < K$ **do**

   using Equation (4), calculate the reliability $R_{(\gamma)}$.

   end
4. Select the first t paths ranking $R_{(\gamma)}$ from highest to lowest.
5. Divide message s into t secret shares using SSS (t,K)
6. Prepare the routing path information and update the flow tables of the corresponding switches
7. Output β_i for $i \in \{\gamma_1, ..., \gamma_t\}$

In this sense, our framework fits into post-quantum security against the quantum computer adversary. The proposed scheme does increase the path numbers in the networking part, but compared to the PKI, the message size is reduced, and also the computation time is much faster. Moreover, in our case, the security requirement of the user determines the number of shares (t). For high security, t is high near to K, but in case of no security, we can take t to be at a minimum value of 2. We will simulate the efficiency and comparison of our algorithm in Section 5.

Figure 4 shows the example scenario of our secret sharing-based network capability routing algorithm. Table 2 shows the computing routing paths and the reliability from source to destination. The column's of Table 3 show the path information of a route, the second column computes the reliability in terms of reliability $R_{(\gamma)}$ of link paths. For first column $S1$ represents the starting node, whereas S12 represents the destination node.

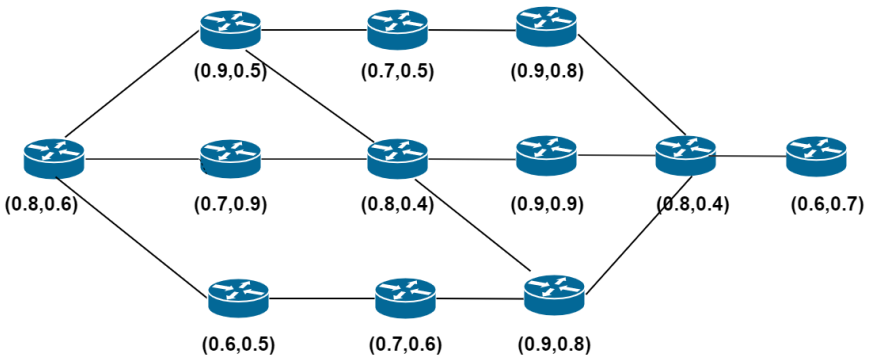

Figure 4. Possible routing path in IoT-SDN and node status.

Table 2. Routing path reliability.

Route	Path	Reliability
γ_1	$S1- > S2- > S6 > S7- > S8- > S12$	0.2617
γ_2	$S1- > S5- > S6 > S11- > S8- > S12$	0.2473
γ_3	$S1- > S2- > S3 > S4- > S8- > S12$	0.2527
γ_4	$S1- > S9- > S10 > S11- > S8- > S12$	0.2285
γ_5	$S1- > S2- > S6 > S11- > S8- > S12$	0.2604
γ_6	$S1- > S9- > S10 > S11- > S6- > S7- > S8- > S12$	0.1508
γ_7	$S1- > S5- > S6 > S2- > S3- > S4- > S8- > S12$	0.1566
γ_8	$S1- > S5- > S6 > S7- > S8- > S12$	0.2493

Table 3. The updates status of nodes in the topology.

Nodes	$t^{(1)}$	$t^{(2)}$
S1	0.8	0.6
S2	0.9	0.5
S3	0.7	0.5
S4	0.9	0.8
S5	0.7	0.9
S6	0.8	0.4
S7	0.9	0.9
S8	0.8	0.4
S9	0.6	0.5
S10	0.7	0.6
S11	0.9	0.8
S12	0.6	0.7

After ranking the paths based on reliability values, first t paths are selected to send the information. The value of t can be adjusted depending upon the user requirements of security. If the security is high, the value of t can be taken large. Otherwise, in the case of no security requirements, t can be as small as permissible in the secret sharing scheme.

Figure 5 illustrates computed reliability status. Figure 5a shows the original topology, which contains twelve SDN-enabled switches. Figure 5b shows the topology showing the shortest path. Figure 5c shows the shortest path determined by using the capability-based routing algorithm proposed by us.

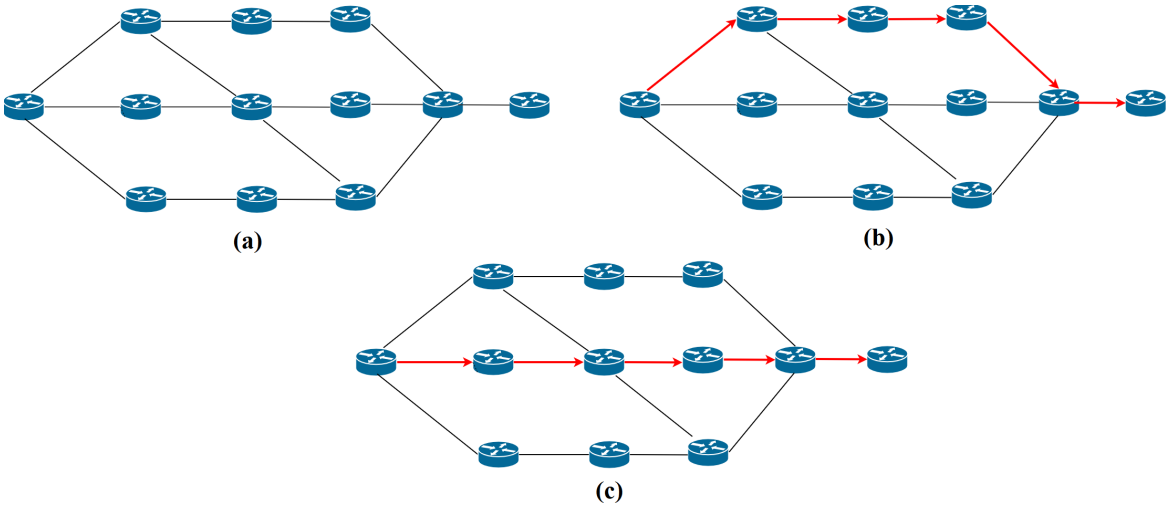

Figure 5. (**a**) Original Topology. (**b**) Shortest Path. (**c**) Shortest path determined using Algorithm 2 in our example scenario.

3.4. Secret Coefficient Exchange

There is one remaining operational step in the proposed framework: exchanging the secret coefficients between the sender and the receiver. We briefly discuss typical approaches and leave the implementation decisions to practitioners. One approach is to pre-install the secret coefficients on the IoT devices, similar to what the industry has been doing for smartphones. The basic set-up allows both parties to agree upon their secret coefficients offline as part of the system setup process. Another approach is to utilize the blockchain such as Hyperledger Fabric [30,31] as a preprocessing platform. The community of the IoT devices will maintain the blockchain. The blockchain serves as the decentralized moderator among all parties. The secret coefficients of the shares are exchanged through blockchain smart contracts where the sender and the receiver both have access. Before any communication begins, parties exchange and agree on the coefficients. Later, the parties may periodically update the secret coefficients throughout the system's lifetime. The running of the smart contracts helps parties achieve pre-agreed information between the sender and the receiver regarding what sets of coefficients to use and at what time. Therefore, a mechanism of time synchronization is assumed in the set-up. The security of the blockchain is implied by the specific blockchain and its design, out of the paper's scope.

4. Security Proof and Discussion

As the security context is related to the size of the information received by the communicating parties, therefore its perfect to use Shannon's principle of entropy results [32]; we will present the security proof of our mechanism. If the shares are secure in transmission from source to destination, then our scheme is secure against the adversary. We have used the following definitions and properties of entropy (for details on information theory see [33]).

Let us assume the r.v.'s as X, Y and Z.

$$H(X) = -\sum_{x \in \chi} P(\mathbf{X} = x) \log_2 P(\mathbf{X} = x) \tag{7}$$

$$H(X,Y) = -\sum_{x \in \chi} \sum_{y \in Y} P(\mathbf{X} = x, \mathbf{Y} = y) \log_2 P(\mathbf{X} = x, \mathbf{Y} = y) \tag{8}$$

$$H(X|Y) = \sum_{y \in Y} P(\mathbf{Y} = y) H(X|Y = y), \tag{9}$$

$$H(X|Y=y) = -\sum_{x\in\chi} P(\mathbf{X}=x|\mathbf{Y}=y) log_2 P(\mathbf{X}=x|\mathbf{Y}=y) \tag{10}$$

To demonstrate our proof, we had used the following properties:

$$\begin{aligned} H(X) &\geq H(X|Y), && (11)\\ H(X,Y) &= H(X)+H(Y|X), && (12)\\ &= H(Y)+H(X|Y), && (13)\\ 0 \leq H(X) &\leq \log_2 \chi, && (14)\\ If\ H(Y|Z) &= 0\ then\ H(X|Y) \geq H(X|Z), && (15)\\ If\ H(Y|Z) &= 0\ then\ H(X|Y,Z) = H(X|Z) && (16) \end{aligned}$$

In order to generate t secret shares ($1 < t \leq$ k) the SDN controller generates a sharing polynomial $f \in \mathbb{K}_{K-1}$[Y], where $\mathbb{K}$ is finite field $\mathbb{K} = \mathbb{F}_p$, where p is prime.

$$f = \alpha_0 + \alpha_1 Y + ... + \alpha_{k-1} Y^{k-1} \tag{17}$$

where $\alpha_0 = s$ (input packet) and $\alpha_i \in \mathbb{K}$ $(i = 1, .., k-1)$ are selected randomly. It is assumed that no less that t shares, $(j_i \in [K], i \in t)$, $f(j_1), ..., f(j_t)$, of the data packet be able to decode it. The following r.v's are considered in rest of the proof.

- S: Secret s
- f_i: Share $f(i)(i \in [K])$
- A_r: Polynomial coefficient $\alpha_r (0 \leqslant r \leqslant k-1)$

The following are the set notations used in the proof.

- $\bar{I}$: $\{j_1, j_2..., j_t\}$ denotes t shares
- $\bar{f_k}$: $\{f_{i_1}, f_{i_2}, ..., f_{i_k}\}$ denotes k r.v. shares
- $\bar{A_k}$: $\{A_0, A_1..., A_{k-1}\}$ denotes k r.v. A
- $\bar{A}_{k-1}$: $\{A_1, A_2..., A_{k-1}\}$ denotes $k-1$ r.v. A

The sets $\bar{I}$ and $\bar{f_k}$ are bijective according to the SSS scheme.

To prove the unconditional security, we will use the proof model of Christian et al. based on entropy [34] . We need to prove the following two results.

(**1**) To prove the correctness of our scheme i.e., any k or more shares can be pooled together to discover the secret s, i.e.,

$$for\ t \geq k,\ H(S|\bar{f_t}) = 0. \tag{18}$$

(**2**) To prove that adversary did not have any information about s, i.e.,

$$for\ t \leq k-1,\ H(S|\bar{f_t}) = H(S). \tag{19}$$

4.1. Correctness

Theorem 1. *Under the route obfuscation and secret-sharing construction, our proposed scheme is correct.*

Proof. Given k points $(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)$ of $\mathbb{K}^2$, a polynomial is,

$$f = \alpha_0 + \alpha_1 Y + ... + \alpha_{k-1} Y^{k-1} \tag{20}$$

of $\mathbb{K}_{K-1}[Y] \ni f(x_i) = y_i$ for $i = 1, 2, ..., k$. The k coefficients $\alpha_0, ..., \alpha_{k-1}$ are determined by $(j_1, f(j_1)), ..., (j_k, f(j_k))$. It follows from information theory,

$$H(\bar{A}_k \mid \bar{f}_k) = 0 \tag{21}$$

Furthermore, as the polynomial f and shares, uniquely calculate the coefficients $\alpha_0, ..., \alpha_{k-1}$. It follows from information theory,

$$H(\bar{f}_k \mid \bar{A}_k) = 0 \tag{22}$$

using (21), (22) and (15),

$$H(S|\bar{A}_k) = H(S|\bar{f}_k) \tag{23}$$

Also because $\alpha_0 = s$, as illustrated before

$$H(S|\bar{A}_k) = H(S|S, \bar{A}_{k-1}) = 0. \tag{24}$$

Therefor, H($S \mid \bar{f}_k$) = 0.By (11), the following holds

$$H(S|\bar{f}_t) \leq H(S|\bar{f}_k) = 0. \tag{25}$$

by using (14),

$$H(S|\bar{f}_t) = 0. \tag{26}$$

This result shows the information-theoretic correctness of our route-obfuscation mechanism. □

Theorem 2. *The proposed route obfuscation and message delivery is perfectly secure.*

Proof. From (12) and (13), joint entropy of $\bar{A}_k$ and $\bar{f}_0 \cup \bar{f}_k$ is

$$H(\bar{A}_k, \bar{f}_0 \cup \bar{f}_{k-1}) = H(\bar{f}_0 \cup \bar{f}_{k-1}) + H(\bar{A}_k \mid \bar{f}_0 \cup \bar{f}_{k-1}) \tag{27}$$

from (21),

$$= H(\bar{f}_0 \cup \bar{f}_{k-1}). \tag{28}$$

Similarly, from (12) and (13),

$$H(\bar{A}_k, \bar{f}_0 \cup \bar{f}_{k-1}) = H(\bar{A}_k) + H(\bar{f}_0 \cup \bar{f}_{k-1} \mid \bar{A}_k) \tag{29}$$

from (22),

$$= H(\bar{A}_k). \tag{30}$$

Hence,

$$H(\bar{f}_0 \cup \bar{f}_{k-1}) = H(\bar{A}_k), \tag{31}$$

as $\bar{f}_0 = \alpha_0 = s$,

$$H(S, \bar{f}_{k-1}) = H(S, \bar{A}_{k-1}) \tag{32}$$

Using (12), (13), Equation (32) can be written as

$$H(\mathbf{S}) + H(\bar{f}_0 \cup \bar{f}_{k-1}|\mathbf{S}) = H(\bar{A}_{k-1}) + H(\mathbf{S}|\bar{A}_{k-1}) \tag{33}$$

The coefficients $\alpha_0, ..., \alpha_{k-1}$ are chosen randomly in $\mathbb{K}$. As a result,

$$H(\bar{A}_{k-1}) = \log_2 |\mathbb{K}|^{k-1}. \tag{34}$$

As coefficients are independent from the secret s, i.e., $\mathrm{H}(\bar{A}_{k-1}) = \mathrm{H}(S)$, therefore

$$\mathrm{H}(\bar{f}_{k-1} \mid S) = \log_2 |\mathbb{K}|^{k-1}.$$

using (11),

$$H(\bar{f}_{k-1}) \geq H(\bar{f}_{k-1} \mid S), \tag{35}$$

$$H(\bar{f}_{k-1}) \geq \log_2 |\mathbb{K}|^{k-1} \tag{36}$$

using (14),

$$H(\bar{f}_{k-1}) \leq \log_2 |\mathbb{K}|^{k-1}, \tag{37}$$

therefore,

$$H(\bar{f}_{k-1}) = \log_2 |\mathbb{K}|^{k-1}, \tag{38}$$

from (12) and (13), we can write Equation (32) as

$$H(\bar{f}_{k-1}) + H(\mathbf{S}|\bar{f}_{k-1}) = H(\bar{A}_{k-1}) + H(\mathbf{S}|\bar{A}_{k-1}) \tag{39}$$

As

$$H(\bar{f}_{k-1}) = H\bar{A}_{k-1}) = \log_2 |\mathbb{K}|^{k-1} \tag{40}$$

$$H(\mathbf{S}|\bar{A}_{k-1}) = H(\mathbf{S}), \tag{41}$$

hence,

$$H(\mathbf{S}|\bar{f}_{k-1}) = H(\mathbf{S}). \tag{42}$$

Using (11)

$$\begin{aligned} H(\mathbf{S}) &\geq H(\mathbf{S}|\bar{f}_t) && (43)\\ &\geq H(\mathbf{S}|\bar{f}_{k-1}) && (44)\\ &= H(\mathbf{S}). && (45) \end{aligned}$$

This implies,

$$H(\mathbf{S}|\bar{f}_t) = H(\mathbf{S}). \tag{46}$$

This proves that our route obfuscation and message delivery scheme is perfectly secure in information-theoretic sense. □

4.2. Discussion

In this section briefly discuss the possible mitigation of the attacks using our scheme.

- Our scheme is a plausible way to guard against MITM attacks. In southbound communication, Transport Layer Security (TLS) is used to protect the network from malicious hosts. Open Flow suggests secure southbound communication through TLS. Systems that do not use TLS are prone to MITM attacks. Many types of MITM attacks exist in the literature. Eavesdropping and packet replaying are some of the kind of MITM attacks. This is the most common attack on privacy. Attackers can easily discover the content of the interaction by snooping the data. To avoid the latency of using TLS support, we use the SSS scheme. Our scheme divides the information into shares, which makes it impossible for adversary to get any use full information out of the packet. If any malicious host replays the packets, the destination node will not be able to decode and result in dropping the fake malicious packet. Moreover, the destination node can predict the presence of an intruder in the network path.

- Our scheme can also help detect the adversarial or fake nodes in the network by analyzing the data. Packets are replayed or sent by nodes that do not allow the destination node to decode the packets, thereby identifying node disrupting network and further being blocked.

5. Simulations

This section, analyses the result of our proposed scheme. We prototyped our methodology and the proposed mechanism, and its usefulness and execution cost is assessed in this section. To emulate our network, We adopt Mininet [35] (version 2.3.0), an open source environment, which is prominently utilized for imitating OpenFlow network environments. We generated different network topologies listed in Table 4. One of the topologies has 12 nodes, see Figure 5. We simulated the network environment in python on Ubuntu 18.04 LTS, using our workstation with configuration (Intel(R) Core TM i7-4500U CPU @ 1.80 GHz 2.40 GHz). For our scheme, Ryu [36] (version 4.24), an open source, free SDN controller supporting python language, is selected. The virtual switch use Open vSwitch (OvS) [37](version 2.10.0). Open vSwitch, often abbreviated as OVS, is an implementation requires virtual environments. The purpose of OVS is providing a switching stack for hardware virtualization environments at the same time as supporting more than one protocols and requirements utilized in networks. Table 5 summarizes the software that was used to implement our proposed mechanism.

Table 4. Network configuration.

Network	No. of Nodes	Number of links	Avg. Min. Nodes in a path	Avg. Max. Nodes in a path
Network1	10	15	4	7
Network2	20	28	7	13
Network3	30	39	6	12
Network4	40	53	8	19
Network5	50	71	5	21
Network6	100	112	7	27

Table 5. Experimental and evaluation software.

Software Application	Operation	Version release
Mininet	Emulator for Network	2.3.0
Ryu	SDN Controller	4.24
OpenFlow	Communications Protocol	1.3
Linux	Operating System	Ubuntu 18.04 64bit
VM VirtualBox Oracle	Virtualizer machine	5.2.4

We added two modules by extending the Ryu controller: (i): Evaluating the nodes using the capability mechanism. (ii): Security level according to the real-time security requirement of the receiver.

We will evaluate our secret route-obfuscation algorithm's performance by calculating the overhead introduced compared to the shortest path algorithm and the time complexity. To evaluate the overhead introduced while splitting the message into shares, as illustrated in Algorithm 2, we compare the results with the shortest path algorithm using Algorithm 1 in our implementation.

5.1. t Paths Using Algorithm 2

Figure 6 illustrates the overhead of our security mechanism over the shortest path. In this study, the network performance is examined according to the number of routers. However, the positions of the hosts and routers are kept constant. We simulated K-shortest path using Algorithm 1 and then choose t paths using Algorithm 2, where $t \leq \lceil K/2 \rceil$. To probe the effect of the size of network, we consider SDN IoT networks with 10, 20, 30, 40, 50, and 100 nodes. The X-axis represents the network

size and *Y*-axis represents time in milliseconds to construct *K* paths and selecting best *t* paths. Results indicate that the efficiency of calculating and choosing the paths is acceptable when *t* is small.

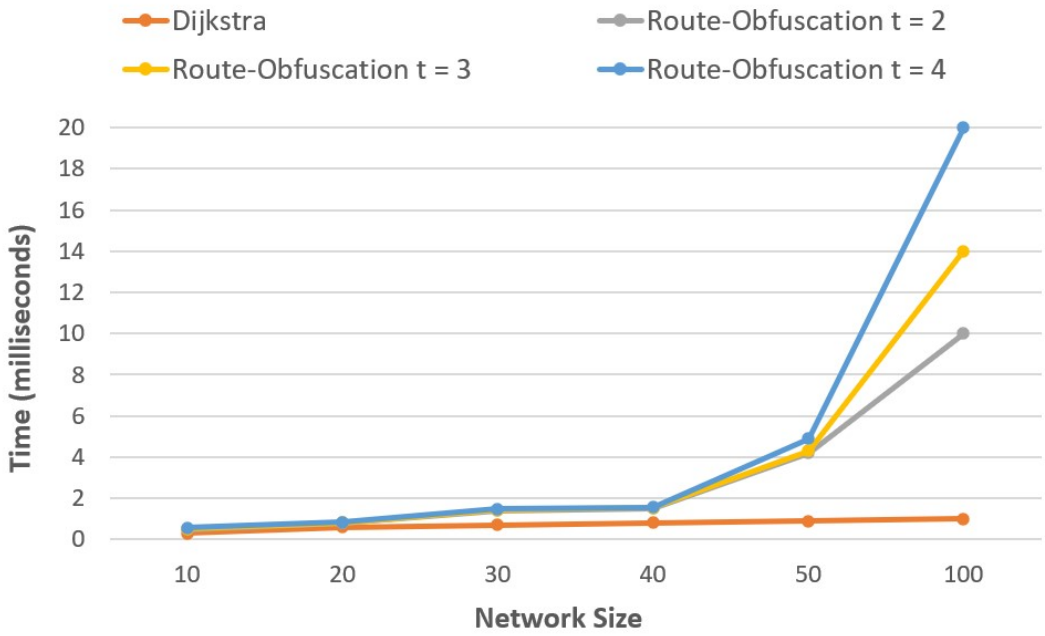

Figure 6. Time complexity with different network nodes.

Table 4 shows the network configuration. For each network, we varied the number of routers and calculated the links. The average minimum and the maximum of nodes in the shortest paths are determined.

We replicated our experiments 20 times and calculated the average time complexity for constructing the possible routing paths, as the network size affects the time. When the first transfer starts, the controller takes some time to discover the network's topology and update the flow tables. This time is calculated as an average of 6.3 s. Of course, this time depends on the specification of the computer used. We have removed this time as it is the delay, which could affect the network performance time.

Simulation tests were carried out by taking variable message lengths of 200, 300, 400, and 500 megabytes and shares of the input packet of size 3, 4, and 5 per host pair. While the number of routers is increased, the hosts and IoT nodes' positions are kept the same. Figure 7 (a-d) shows the average time of our implementation. In the graph, network size is plotted on *X*-axis and the average time in milliseconds for our obfuscation route algorithm to calculate the routing paths on *Y*-axis. The graph shows that the execution time of the algorithm increases linearly with increase in network size. Dijkstra's execution time is the lowest, as it is the most basic algorithm used here for comparison. because of more than one path, our algorithm's execution time shows that it is an acceptable trade-off between security and efficiency.

We compared the running time of our algorithm before and after configuring the reliability mechanism to determine the paths. In the graph, network size i.e., nodes is plotted on the *X*-axis and the average time in milliseconds on *Y*-axis by taking message lengths of 200, 300, 400, and 500 MB. Average results are obtained after replicating experiments 20 times each. The results of Figure 8 indicate that the reliability mechanism has chosen the paths efficiently as compared to the one without. Therefore, the feasibility of incorporating the security mechanism can be considered where speed can be compromised over security. As we are choosing the shortest path in our route obfuscation approach, the increase in time is approximately linear.

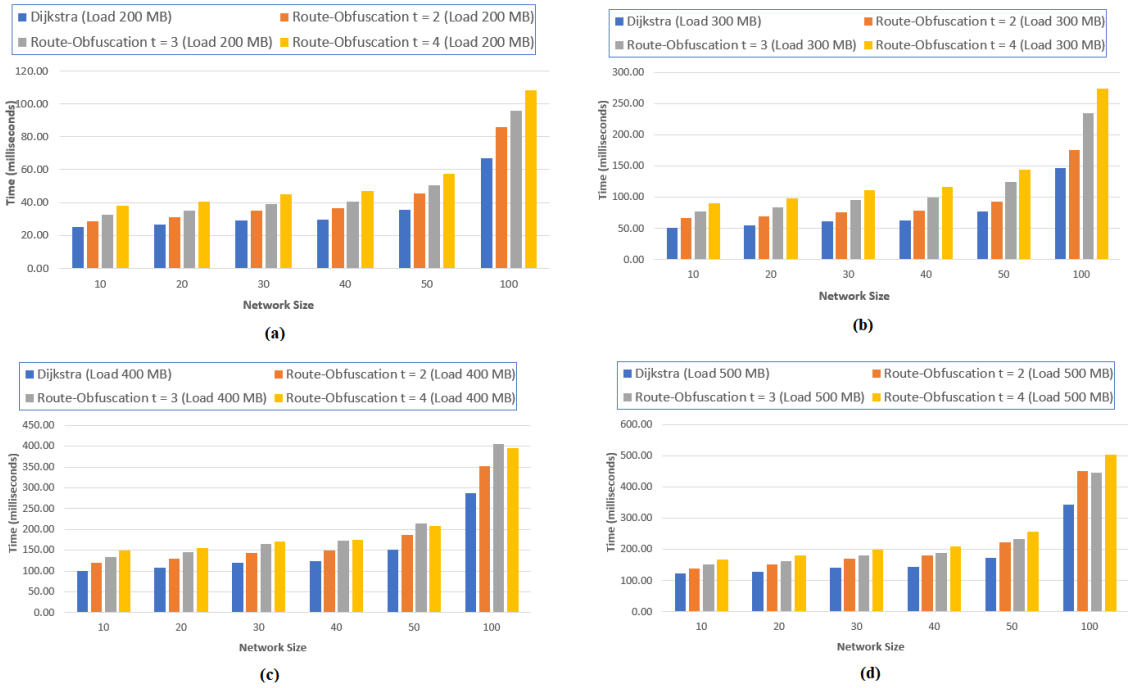

Figure 7. Average execution time of our route obfuscation algorithm. (**a**) Load = 200 MB; (**b**) Load = 300 MB; (**c**) Load = 400 MB; (**d**) Load = 500 MB.

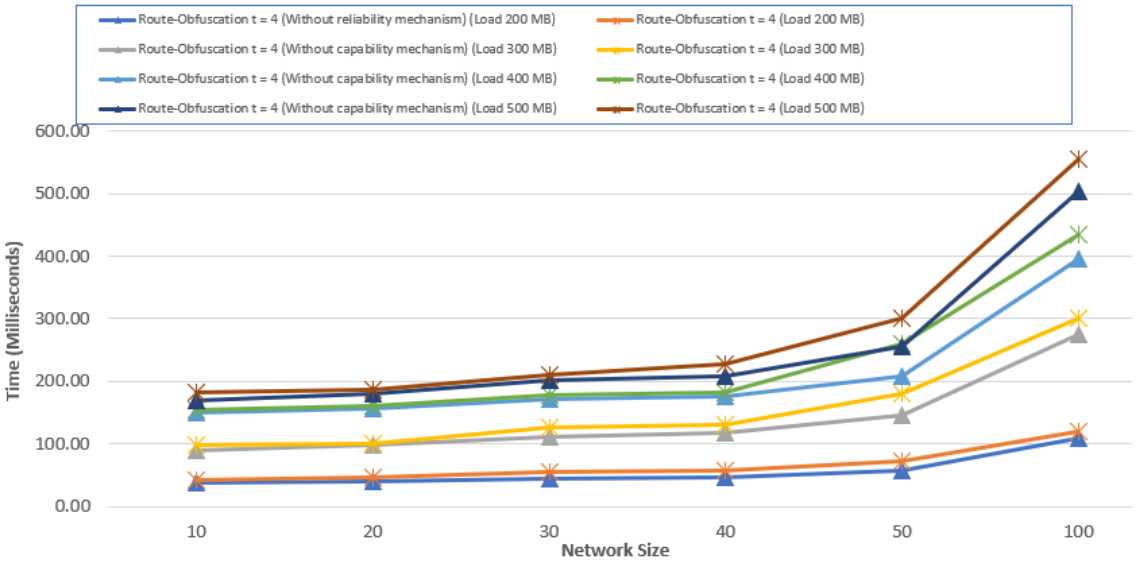

Figure 8. Average execution time comparison of using the route obfuscation algorithm for $t = 4$.

5.2. Time Complexity

Let us consider formal definition for our IoT SDN Network. For dijkstra algorithm the running time complexity is $O(m + n \log n)$, and for our t-shortest path route-obfuscation algorithm is $O(t(m + n \log n))$, where m and n represents nodes and edges respectively.

6. Conclusions

With the threat of upcoming quantum adversaries, cryptography schemes built upon PKI that rely on number-theoretic hard problems are vulnerable. Besides, the limited computing resources and energy storage in the IoT devices make the efficiency of using PKI to establish critical message passing in the scale of milliseconds unrealistic. To overcome both challenges, we propose to

use information-theoretic secret sharing with route-obfuscation to provide security against certain adversaries in realistic settings. Our construction achieves information-theoretic security in message encryption and increases the difficulty for adversaries to intercept messages or modify the messages. Our scheme comes with theoretic proofs and simulation validation. The proposed scheme is also efficient as it has a time complexity of a few milliseconds, validated by the simulation results. In the future, we will investigate the relationship between the number of random routes and time complexity, as well as countermeasures for active adversaries on the routes after the receiver node detects the failure of reconstructed messages.

Author Contributions: Conceptualization, A.R. and Z.W.; methodology, A.R. and Z.W.; software, A.R.; validation, A.R., Z.W., H.S., and M.A.T.; writing—original draft preparation, A.R. and Z.W.; writing—review, H.S. and M.A.T.; visualization, A.R.; funding acquisition, A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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

References

1. Esiner, E. F-Pro: A Fast and Flexible Provenance-Aware Message Authentication Scheme for Smart Grid. In Proceedings of the 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), Beijing, China, 21–23 October 2019.
2. Gan, G., Lu, Z., Jiang, J. Internet of things security analysis. In Proceedings of the 2011 International Conference on Internet Technology and Applications, Wuhan, China, 16–18 August 2011.
3. Del Pino, R.; Lyubashevsky, V.; Neven, G.; Seiler, G. Practical quantum-safe voting from lattices. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017.
4. Cesare, C. Encryption faces quantum foe. *Nature* **2015**, *10*, 167. [CrossRef] [PubMed]
5. Open Networking Fundation. Software-defined networking: The new norm for networks. Available online: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf (accessed on 25 July 2020).
6. Wang, J.; Zhai, P.; Zhang, Y.; Shi, L.; Wu, G.; Shi, X.; Zhou, P. Software defined network routing in wireless sensor network. In Proceedings of the Cloud Computing, Security, Privacy in New Computing Environments, Guangzhou, China, 15–16 December 2016; pp. 3–11.
7. Wang, J.; Miao, Y.; Zhou, P.; Hossain, M.S.; Rahman, S.M.M. A software defined network routing in wireless multihop network. *J. Netw. Comput. Appl.* **2017**, *85*, 76–83. [CrossRef]
8. Liu, Y.; Kuang, Y.; Xiao, Y.; Xu, G. SDN-based data transfer security for Internet of Things. *IEEE Internet Things J.* **2017**, *5*, 257–268. [CrossRef]
9. Shin, S.; Wang, H.; Gu, G. A first step toward network security virtualization: From concept to prototype. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 2236–2249. [CrossRef]
10. Shen, S.H.; Huang, L.H.; Yang, D.N.; Chen, W.T. Reliable multicast routing for software-defined networks. In Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM), Kowloon, Hong Kong, 26 April–1 May 2015.
11. Huang, M.; Liang, W.; Xu, Z.; Xu, W.; Guo, S.; Xu, Y. Dynamic routing for network throughput maximization in software-defined networks. In Proceedings of the IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, USA, 10–14 April 2016.
12. Lee, M.-C.; Sheu, J.-P. An efficient routing algorithm based on segment routing in software-defined networking. *Comput. Netw.* **2016**, *103*, 44–55. [CrossRef]
13. Lee, Z.-K.; Lee, G.C.; Oh, H..; Song, H. QoS-aware routing and power control algorithm for multimedia service over multi-hop mobile ad hoc network. *Wirel. Commun. Mob. Comput.* **2012**, *12*, *7*, 567–579 [CrossRef]
14. Huang, H.; Guo, S.; Li, P.; Ye, B.; Stojmenovic, I. Joint optimization of rule placement and traffic engineering for QoS provisioning in software defined network. *IEEE Trans. Comput.* **64**, *12*, 3488–3499. [CrossRef]
15. Wan, K.; Luo, X.F.; Jiang, Y.; XU, K. The Flow-oriented Scheduling Algorithms In SDN System. *Chin. J. Comput.* **2016**, *6*, 1208–1223.

16. Wang, M.; Liu, J.; Mao, J.; Cheng, H.; Chen, J.; Qi, C. RouteGuardian: Constructing secure routing paths in software-defined networking. *Tsinghua Sci. Technol.* **2017**, *22*, 400–412. [CrossRef]
17. Casado, M.; Garfinkel, T.; Akella, A.; Freedman, M.; Boneh, D.; McKeown, N.; Shenker, S. SANE: A Protection Architecture for Enterprise Networks. In Proceedings of the 15th USENIX Security Symposium, Vancouver, B.C., Canada, 31 July–4 August 2006.
18. Yoon, C.; Lee, S. Attacking SDN infrastructure: Are we ready for the next-gen networking. Available online: https://www.blackhat.com/docs/us-16/materials/us-16-Yoon-Attacking-SDN-Infrastructure-Are-We-Ready-For-The-Next-Gen-Networking.pdf (accessed on 25 July 2020).
19. Fouda, M.M.; Fadlullah, Z.M.; Kato, N.; Lu, R.; Shen, X.S. A lightweight message authentication scheme for smart grid communications. *IEEE Trans. Smart Grid* **2011**, *2*, 675–685. [CrossRef]
20. Nicanfar, H.; Jokar, P.; Beznosov, K.; Leung, V.C. Efficient authentication and key management mechanisms for smart grid communications. *IEEE Syst. J.* **2013**, *8*, 629–640. [CrossRef]
21. Bernstein, D.J.; Lange, T. Post-quantum cryptography. *Nature* **2017**, *549*, 188–194. [CrossRef] [PubMed]
22. Paquin, C.; Stebila, D.; Tamvada, G. Benchmarking post-quantum cryptography in tls. In Proceedings of the Eleventh International Conference on Post-Quantum Cryptography, Paris, France, 15–17 April 2020.
23. Nejatollahi, H.; Valencia, F.; Banik, S.; Regazzoni, F.; Cammarota, R.; Dutt, N. Synthesis of flexible accelerators for early adoption of ring-lwe post-quantum cryptography. *ACM Trans. Embed. Comput. Syst.* **2020**, *19*, 1–17. [CrossRef]
24. Andrzejczak, M. The Low-Area FPGA Design for the Post-Quantum Cryptography Proposal Round5. In Proceedings of the 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), Leipzig, Gemany, 1–4 September 2019.
25. Cramer, R.; Damgård, I.B.; Döttling, N.; Fehr, S.; Spini, G. Linear secret sharing schemes from error correcting codes and universal hash functions. In Proceedings of the 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, 26–30 April 2015.
26. Pang, L-J.; Wang, Y.-M. A new (t, n) multi-secret sharing scheme based on Shamir's secret sharing. *Appl. Math. Comput.* **2005**, *10*, 840–848. [CrossRef]
27. Mahmoud, M.M.; Lin, X.; Shen, X. Secure and reliable routing protocols for heterogeneous multihop wireless networks. *IEEE Trans. Parallel Distrib. Syst.* **2013**, *26*, 1140–1153. [CrossRef]
28. Valdés, L.; Ariza, A.; Allende, S.M.; Joya, G. Searching the Shortest Pair of Edge-Disjoint Paths in a Communication Network. A Fuzzy Approach. In Proceedings of the 15th International Work-Conference on Artificial Neural Networks, IWANN 2019, Gran Canaria, Spain, 12–14 June 2019.
29. Shamir, A. How to Share a Secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
30. Yazdinejad, A.; Parizi, R.; Dehghantanha, A.; Zhang, Q.; Choo, K.K. An energy-efficient SDN controller architecture for IoT networks with blockchain-based security. *IEEE Trans. Serv. Comput.* **2020**. [CrossRef]
31. Cachin, C. Architecture of the hyperledger blockchain fabric. Availabe online: https://pdfs.semanticscholar.org/f852/c5f3fe649f8a17ded391df0796677a59927f.pdf (accessed on 25 July 2020).
32. Shannon, C. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
33. Cover, T.M. *Elements of Information Theory*; John Wiley and Sons: Hoboken, NJ, USA, 1999.
34. Corniaux, C.; Ghodosi, H. An entropy-based demonstration of the security of Shamir's secret sharing scheme. In Proceedings of the 2014 International Conference on Information Science, Electronics and Electrical Engineering, Sapporo, Japan, 24–26 April 2014.
35. Mininet. An Instant Virtual Network. Available online: http://mininet.org/ (accessed on 28 June 2020).
36. Ryu. SDN Framework. Available online: https://github.com/faucetsdn/ryu (accessed on 28 June 2020).
37. Open vSwitch. Home Page. Available online: https://www.openvswitch.org/ (accessed on 28 June 2020).