future internet

Article

A Robust Security Architecture for SDN-Based 5G Networks

Jiaying Yao [1,2], Zhigeng Han [2,3], Muhammad Sohail [1,2] and Liangmin Wang [1,2,*]

1 School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang 212013, China; 2221708060@stmail.ujs.edu.cn (J.Y.); engrsohailaslam@gmail.com (M.S.)
2 Jiangsu Key Laboratory of Security Technology for Industrial Cyberspace, Zhenjiang 212013, China; hanzgnit@126.com
3 School of Information Engineering, Nanjing Audit University, Nanjing 211815, China
* Correspondence: wanglm@ujs.edu.cn

Received: 30 January 2019; Accepted: 26 March 2019; Published: 28 March 2019

Abstract: 5G is the latest generation of cellular mobile communications. Due to its significant advantage in high data rate, reduced latency and massive device connectivity, the 5G network plays a vital role in today's commercial telecommunications networks. However, the 5G network also faces some challenges when used in practice. This is because it consists of various diverse ingredients, termed heterogeneity. The heterogeneity of the 5G network has two consequences: first, it prevents us to use this technology in a uniform way, preventing the wide use of 5G technology; second, it complicates the structure of the 5G network, making it hard to monitor what is going on in a 5G network. To break through this limitation, researchers have worked in this field and design their own protocol, in which software-defined networking (SDN) is one key design concept. By separating control and data plane, SDN can make the 5G network functional and programmable, such that we can handle the heterogeneity in traditional 5G networks. In light of this, we say that SDN-5G network is attractive, but its advantages are not free. The intelligence centralization used in SDN has its own drawbacks when it comes to security. To break through this limitation, we propose a robust security architecture for SDN-based 5G Networks. To find the illegal request from malicious attackers, we add extra cryptographic authentication, termed synchronize secret. The basic idea of our scheme is leveraging preload secrets to differ attacks from regular network communications. The simulation results indicate that our work can completely handle the security problem from SDN with a low disconnect rate of 0.01%, which is much better than that from state of the art.

Keywords: 5G networks; SDN; hash; cryptographic authentication

1. Introduction

5G networks, as a new wireless communication technology, experience a fast development in recent years. As shown in Figure 1, this technique has been widely used in every corner of our daily life. Compared with the outdated commercial 4G (LTE/WiMax) system, this technology has advantages in high data rate, reduced latency, and massive device connectivity, making it a fundamental infrastructure module for wireless communication in the near future. Motivated by its significant advantages, many researchers have designed their own protocols [1–3] to make it fit the requirement of the real applications. In this work, software-defined networking (SDN) is one of the key design concepts. SDN is an approach to improve network performance and monitoring by facilitating network management and enabling programmatically efficient network configuration [4]. By separating data and control planes, SDN enables a wide range of new innovative applications from traffic engineering to data center virtualization, fine-grained access control, and so on [5]. It has a proven advantage in many commercial networks [6,7] and therefore is also a good choice in the field of 5G networks.

Despite these advantages, forming such an SDN-based 5G network is not free, and there remains a lot of challenges when it comes to security. This is because the intelligence centralization of SDN is vulnerable for various attacks. Several research works on SDN have already investigated security applications built upon the SDN controller, with different aims in mind. Distributed Denial of Service (DDoS) detection and mitigation [8,9], as well as botnet [10] and worm propagation [11], are some concrete use-cases of such applications: basically, the idea consists of periodically collecting network statistics from the forwarding plane of the network in a standardized manner (e.g., using OpenFlow), and then apply classification algorithms on those statistics in order to detect any network anomalies. If an anomaly is detected, the application instructs the controller how to reprogram the data plane in order to mitigate it. Another kind of security application leverages the SDN controller by implementing some moving target defense (MTD) algorithms. MTD algorithms are typically used to make any attack on a given system or network more difficult than usual by periodically hiding or changing key properties of that system or network. In traditional networks, implementing MTD algorithms is not a trivial task since it is difficult to build a central authority capable of determining—for each part of the system to be protected—which key properties are hidden or changed. In an SDN network, such tasks become more straightforward thanks to the centrality of the controller. One application can, for example, periodically assign virtual IPs to hosts within the network, and the mapping virtual IP/real IP is then performed by the controller [12]. Another application can simulate some fake opened/closed/filtered ports on random hosts in the network in order to add significant noise during the reconnaissance phase (e.g., scanning) performed by an attacker [13]. Additional value regarding security in SDN enabled networks can also be gained using FlowVisor [14] and FlowChecker [15], respectively. The former tries to use a single hardware forwarding plane sharing multiple separated logical networks. Following this approach, the same hardware resources can be used for production and development purposes as well as separating monitoring, configuration and internet traffic, where each scenario can have its own logical topology which is called slice. In conjunction with this approach, FlowChecker [14] realizes the validation of new OpenFlow rules that are deployed by users using their own slice. SDN controller applications are mostly deployed in large-scale scenarios, which requires comprehensive checks of possible programming errors. A system to do this called NICE was described in 2012 [16]. Introducing an overarching security architecture requires a comprehensive and protracted approach to SDN. Since it was introduced, designers are looking at possible ways to secure SDN that do not compromise scalability. One architecture called SN-SECA (SDN+NFV) Security Architecture [17].

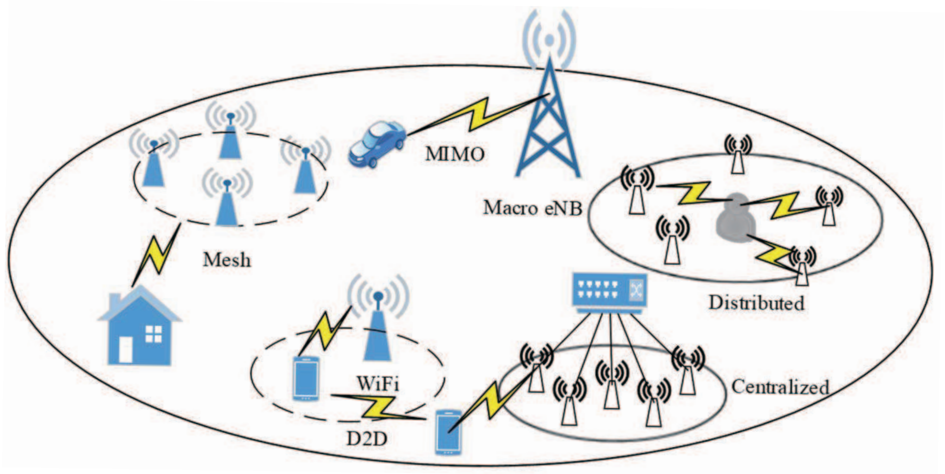


Figure 1. 5G networks.

In this paper, we target at handling the security problems in SDN-based 5G networks. As we mentioned, the SDN 5G network has two planes: the control plane and the data plane. We add a security module between the two planes. This security will synchronize secrets between the physical layer devices and the SDN controller. This secret used in our scheme is generated by the hash function. During each authentication, the security module will first issue the 5G device for its secret. Since the security module knows the hash function in the device in advance, it can determine whether the reply is correct; the incorrect secrets indicate the request is from an attacker. After a successful authentication, the security module will update the secret seed in the physical layer device to prevent the attacker from decoding the hash function by eavesdropping for a long time. Note that our scheme is low cost and is fully compatible with the SDN architecture, making it competitive when deploying in today's SDN-based applications. Finally, we simulate our scheme under different kinds of attacks. The simulation results show that our work has a disconnect rate of 0.01, which is much lower than the state of art. Our contributions are three fold:

1. In this paper, we propose a security architecture for SDN-based 5G networks. To the best of our knowledge, there is no other work in this field.
2. We detail the security problems of SDN-based 5G network in the communication plane. We propose a low-cost security authentication scheme for these problems.
3. We detail how our scheme raises the security of a current SDN based 5G network. In addition, we compare our work with the state of the art. The simulation results show that our work is robust while it has a low disconnect rate of 0.01%, whose rate is about 80% in the state of the art when the number of attacks is 50,000.

The rest of the paper is organized as follows. Section 2 introduces the background of SDN-based 5G network and its security drawbacks. Then, in Section 3, we give our security architecture in details. After that, the simulation results are provided to demonstrate the superiority of the proposed schemes in Section 4. Finally, Section 5 briefly concludes this manuscript.

2. Background

2.1. SDN Based 5G Network Architecture

In this subsection section, we first illustrate the concept of SDN based network architecture. Here, the SDN is an approach to improve the scalability of the traditional network and to provide the unified interface to the upper applications and terminal devices. In a traditional network architecture, the applications need to program the network hardware devices when communicating with the terminal devices. If the hardware devices are changed (e.g., is provided by a new manufacturer), the original upper applications may not be able to control them since these devices often use a different Application Programming Interface (API), making the network hard to be programmed as a consequence. The SDN architecture is just designed to handle this problem. For better understanding of this technique, we show a typical SDN architecture in Figure 2. As we can see, the SDN architecture is made up of three components: the application layer, the control layer and the data layer. Comparing with traditional network architecture separates the data and control into two planes instead of exposing all of them to the upper application plane. Since the control plane will automatically manage the network hardware devices, this separation can help the application to directly control the network with a unit interface given by the control plane and do not need to consider the hardware changes as we mentioned in the traditional network. Similarly, when the terminal devices in the network attempt to send their message to the applications, the SDN architecture will also provide the unified interface to overcome the bad scalability cased by hardware difference. Considering the significant advantage of SDN architecture, it becomes a key design concept in the 5G Network architecture.

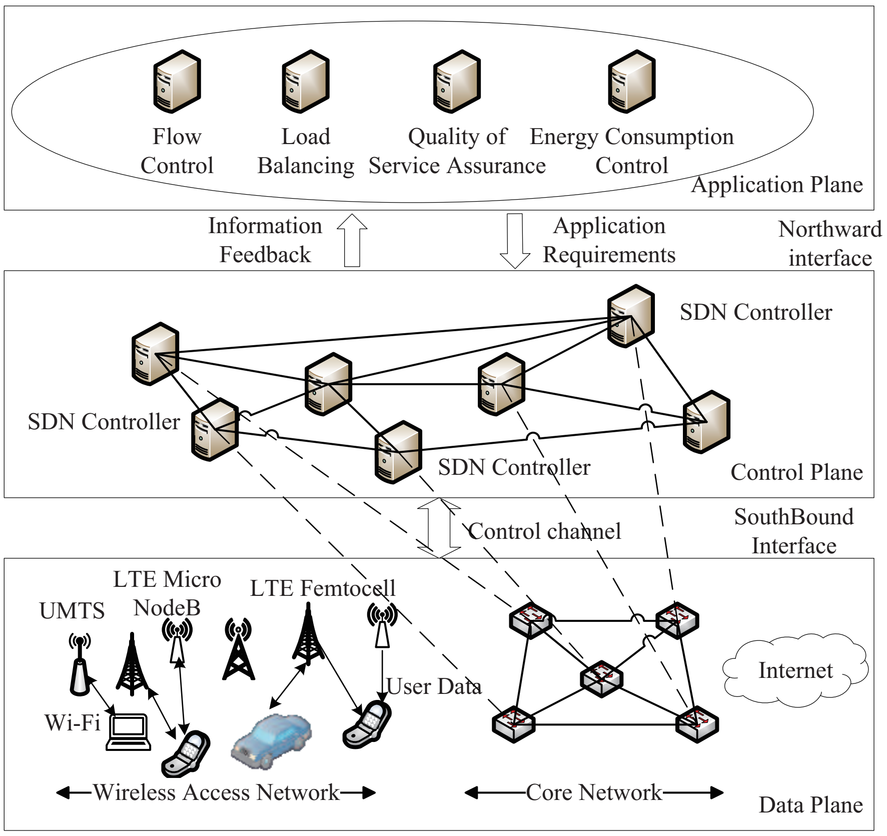


Figure 2. SDN-based architecture.

2.2. Security Problems in SDN-Based 5G Networks

Although the SDN architecture can greatly improve the scalability of the traditional 5G network, the same attributes of centralized control and programmability associated with the SDN platform introduce network security challenges. The communication channel of SDN channel is vulnerable due to the following six reasons [18]:

1. The absence of trust and weak authentication between the applications and the controller, which may lead to a spoofing attack by spoofing northbound API messages.
2. Inappropriate authorization may cause inappropriate or malicious access on the application.
3. The absence of encryption of the traffic between the controller and switches leads to eavesdropping and spoofed southbound communications.
4. The absence of trust and weak authentication at this level may lead to Man-In-The-Middle and spoofing attacks, which makes it easier for the attacker to eavesdrop on flows to see what flows are in use and what traffic is being permitted across the network.
5. Inappropriate authorization that may cause unauthorized access. In fact, if a user requests a service that triggers the controller to create a route and cause packets to traverse that route, it must be ensured that the user is authorized to do so and that the environment can account for that activity.
6. Southbound communication affected by attacks on flow rules mainly in In-band deployments

The security flaw of SDN architecture brings a lot of security problems. In the following, we list three classical attacks [19] to show how a malicious attacker leverages these security flaws to attack the SDN-based network. The first type is IP spoofing. The IP spoofing refers to the progress of using forged IP addresses to inject packets into the SDN networks. This technique is mainly used in the DDOS attacks. In a DDOS attack, the attacker will send the flood of TCP SYN messages or UDP SYN messages or ICMP messages to the victim host using the IP address which can be found using IP spoofing. It causes a table-miss in the flow table in the switch, making the controller become unavailable for other nodes to make routing decisions, reducing the performance of the SDN networks. The second type is the man-in-the-middle (MITM) attack. In a MITM attack, the attacker secretly relays and possibly alters the communication between two parties who believe they are directly communicating with each other. By eavesdropping on the communication messages in the SDN network, the attacker can obtain all details required in conducting a MITM attack since the communication channel in SDN is lack of authentication. The last type of attack is replay attacks. Similar to the MITM attack, the attacker first eavesdrops on the communication channel. With the valid data transmission, the attacker then maliciously or fraudulently repeats or delays it, thereby fooling the honest participant(s) into thinking they have successfully completed the protocol run. Given the bad security condition in SDN architecture, it is urgent to adapt it for practical use.

2.3. Related Work

The methods of SDN security architecture generally fall into four categories. The first type is Denial of Service Mitigation: the key point in protecting networks from being a DOS attack is the recognition of what flows of traffics are malicious as soon as possible because malicious packets are very similar. Traditional methods include preprocessing of traffic log files and sequentially captured packets. Now, this high-overhead practice is not required in SDN because Counters are embedded in each network device in the data. The DOS detection loop consists of three components: streams Collectors, feature extractors, and classifiers. Ref. [9] presents a lightweight method for DDoS attack detection based on traffic flow features, in which the extraction of such information is made with a very low overhead compared to traditional approaches.

The second type is Cloud Security: Cloud computing is becoming a popular paradigm. Many recent new services are based on cloud environments, and a lot of people are using cloud networks.Since many diverse hosts and network configurations coexist in a cloud network, it is essential to protect each of them in the cloud network from threats. Ref. [20] propose a new framework, CLOUDWATCHER, which provides monitoring services for large and dynamic cloud networks. The framework automatically detours network packets to be inspected by pre-installed network security devices. In addition, all these operations can be implemented by writing a simple policy script; thus, a cloud network administrator is able to protect his cloud network easily.

The third type is Topology Protection: protecting network topology is crucial in SDNs since all popular SDN controllers are all subjected to network topology attacks. Any changes to flow behaviors should be flagged for immediate remedy. Ref. [21] proposes a mitigation method for network topology poisoning attacks, and a new SDN controller security extension TopoGuard is demonstrated, which can detect network topology poisoning attacks automatically and in real time. One solution is SPHINX [22], which leverages the novel abstraction of flow graphs. It is very close to the actual network operation to enable incremental verification of all network updates and constraints.

The last type is Policy Enforcement. Stabler et al. propose a reference implementation of an Elastic IP and Security Group service—the OpenFlow. The core of the implementation relies on the integration of an OpenFlow controller (NOX) with the EC2 server. One example is LiveSec. Ref. [23] presents LiveSec, a scalable and flexible security management architecture, which achieves holistic security protection with good scalability and flexibility in large-scale networks. They can apply different sets of policies onto different types of VMs being created dynamically on the cloud, providing protection services similar to firewalls as well as elastic IP service.

Ref. [9] focuses on protecting networks from being DDOS attacks; Ref. [20] focuses on protecting many diverse hosts and network configurations in a cloud network from threats; Refs. [21,22] focuses on protecting network topology is crucial in SDNs since all popular SDN controllers are subjected to network topology attacks. Our solution falls into the fourth category. Ref. [23] focuses on establishing a scalable and flexible security management architecture, which achieves holistic security protection with good scalability and flexibility in large-scale networks. Our solution is different from Ref. [23]. We propose a robust security architecture for SDN-based 5G Networks. To find the illegal request from malicious attackers, we add extra cryptographic authentication, termed as synchronize secret. The basic idea of our scheme is leveraging preload secrets to differ attacks from regular network communications.

3. Robust Security Architecture of SDN-Based 5G Networks

3.1. Basic Idea

Our basic idea is to add authentication that can verify whether the message is from an attacker or a legal user in the communication channel. The concept underlying our security authentication is synchronizing secret. In our scheme, all of the 5G devices will maintain a secret key, which is a value generated by the encryption algorithm. These keys will also be recorded in a back end system. During the communication process, the back end system will check whether the device has the same secret as the its records. Since the attacker can not obtain this secret in advance, its requirement will not pass the authentication process, namely, we have successfully found an SDN security attack. Moreover, as shown in Table 1, the security attacks will occur in two different scenes: the first type occurs in the communication between the SDN control in the control plane and the terminal devices in the data plane, the other type occurs between different terminal devices. In light of this, we propose two different authentication schemes to respectively handle the security problem in the two scenes. In the following, we will give our 5G SDN architecture in details.

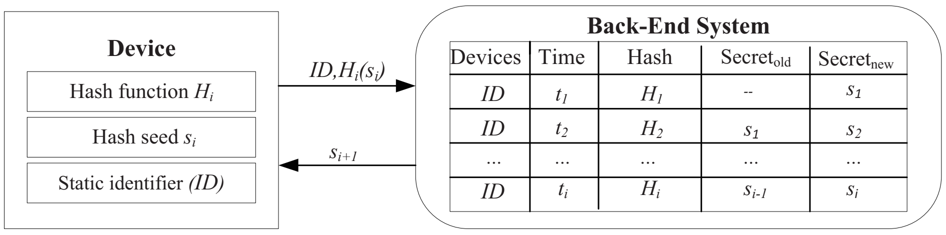
Table 1. Security problems in 5G networks.

Channel Types	**IP Spoofing**	**MITM Attack**	**Replay Attack**
Control channel	√	√	√
Data channel	√	√	√

3.2. Security Architecture

In this subsection, we will demonstrate our security architecture, which is illustrated in Figure 3. In this figure, we can see that the authentication process has two ends. The left end is the SDN devices, which is a switch of an SDN control. The secret store in each device consists of three parts: a unique ID, a preload hash table H_i given by the back-end and a synchronized secret s_i. The secret is a random number that is changed every time the device requests building a communication channel. During an authentication, the back-end first verifies the device's ID. If this number is valid, the back-end will then check the hash value from H_i by using s_i as hash seed. If this number matches the hash value given by the hash table and seed stored for this device in the back-end, the device passes the authentication and then builds a communication channel. After the authentication, the back-end will generate a new secret and load it into the device. An attacker can pass the authentication only after (1) eavesdropping the secret s_{i+1} in the former authentication and (2) knowing the hash function preload in the devices. We say that meeting the two requirements is difficult since, if the device has a hash value from an outdated secret, it indicates that this request is from an attacker.

Next, we will detail how to add this authentication in the SDN-based 5G architecture. Figure 4 illustrates our security architecture. As we can see, we add the security gateways (SGW) and security entry (SecE) in the SDN architecture. SGW refers to the gateway that serves as the point of decryption and encryption for the network. SGW can relay authentication messages between SE and data plane switch (DPS). SecE refers to the back-end in Figure 3. It will maintain the hash tables and hash seeds of all devices and provide authentication when building the communication channel. Here, we note that we use the SGW to relay the authentication message instead of letting the SecE communicate with the 5G devices directly. This SGW can combine the messages from devices and then transmit these messages to the SecE. In such a way, we can lower the load of SecE, accelerating the authentication process as a consequence. Since we add the SGW, the authentication process is a little different from the basic authentication scheme illustrated in Figure 3. In the following, we will respectively detail the authentication in the control channel and the authentication in the data channel.



Devices	Time	Hash	Secret$_{old}$	Secret$_{new}$
ID	t_1	H_1	--	s_1
ID	t_2	H_2	s_1	s_2
...	...	...	...	...
ID	t_i	H_i	s_{i-1}	s_i

Figure 3. Illustration of synchronous secrets.

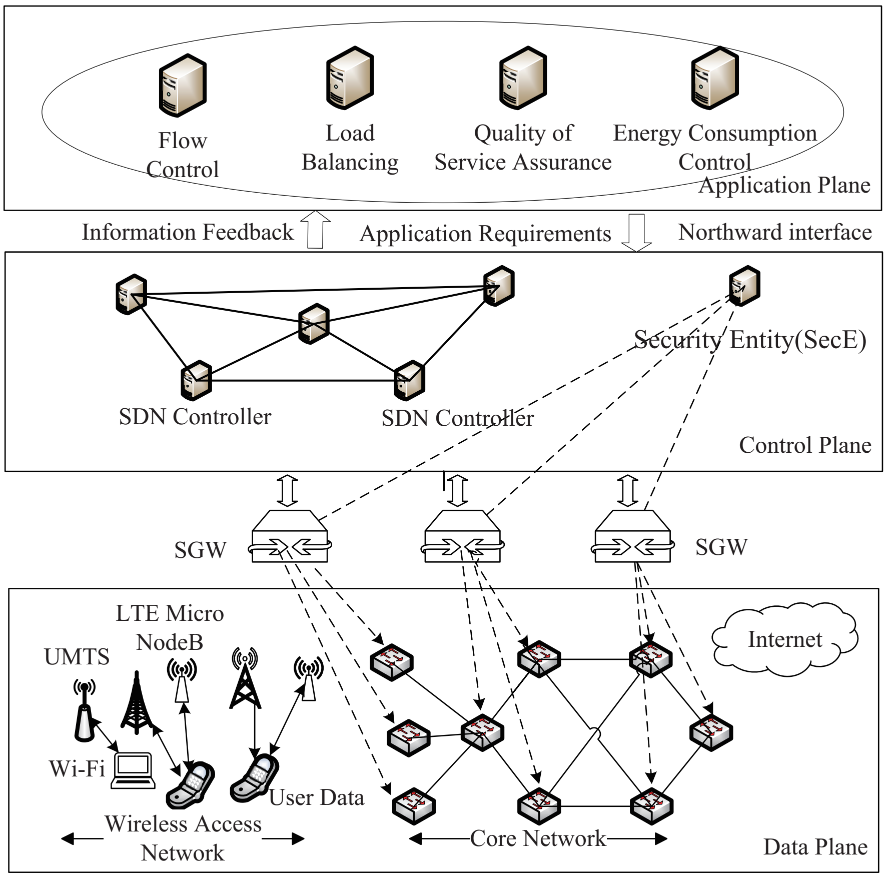


Figure 4. Robust security architecture.

We first introduce our authentication scheme in the control channel. The authentication process is illustrated in Figure 5. As we can see, the DPS first transmit an authentication request M_1 to SGW, M1 consists of two parts: the device's hash value and ID. After receiving M_1, the SGW will combine the current authentication requests, and then send the combination *REQ* to the SecE. After that, SecE verifies the hash values in *REQ* and then sends the authentication messages *ACK* to the SGW. Finally, the SGW will relay the authentication messages to the DPS. If the DPS passes this authentication, it will update its secret and build the control channel with SDN controllers. On the contrary, if the DPS doesn't pass this authentication, it will do nothing or request again.

We then introduce our authentication scheme in the data channel. The authentication process is illustrated in Figure 6. As we can see, the data channel is built between DPS1 and DPS2. Thus, we need to verify both of them to guarantee the security of this channel. In this figure, DPS1 first sends authentication request M_1 to DPS2, and M_1 is made up of the device's ID and associated hash value. After receiving the request from DPS1, DPS2 will send authentication request M_2 to the SGW.

Similar to the process in the control channel, SGW will then transmit the combination of authentication request REQ to SecE and then get the authentication message ACK. SGW will then send M_3 to DPS2, and M_3 is the authentication message associated with DPS1 and DPS2. Finally, if this request passes the authentication process, DPS2 will send the authentication message to DPS1, building the data channel and then both of the two DPSs will update their own seed. If one of the DPSs doesn't pass this authentication, both of them will do nothing.

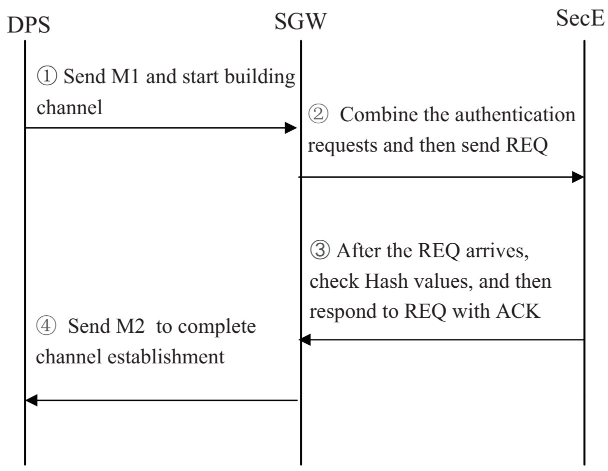

DPS
SGW
SecE
① Send M1 and start building channel
② Combine the authentication requests and then send REQ
③ After the REQ arrives, check Hash values, and then respond to REQ with ACK
④ Send M2 to complete channel establishment

Figure 5. Authorization in Control channel.

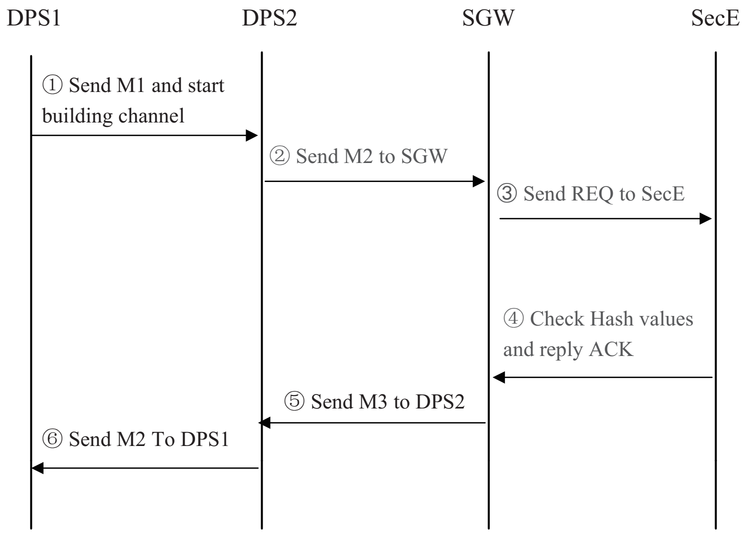


Figure 6. Authorization in Data channel.

4. Security Analysis

In this section, we illustrate how the proposed security architecture protects the SDN-based 5G network under the three kinds of attacks mentioned in Table 1.

4.1. IP Spoofing

In an IP spoofing, we assume that the attacker has known all of the valid data stored in a legal devices, which refer to 5G terminal devices in the data plane or an SDN controller in the control plane. By using the proposed scheme, the system can figure out the messages from the attackers because the secret will change over time; the attacker can only provide an outdated secret, which is not the same as the one stored in the back-end system.

4.2. MITM Attack

In the MITM attack, the attacker secretly relays and possibly alters the communication between two parties who believe they are directly communicating with each other. However, this kind of attack can be detected in our scheme. First, the attacker must hijack both uplink and downlink in an authentication process, causing such kind of attack to be conducted. Second, even if the attacker has successfully conducted an MITM attack, the attacker hijacks a device's communication in the authentication process, making the back-end system fail to update the secret stored in this device; the back-end system can find this difference, which indicates a MITM attack, in the next authentication process of this device.

4.3. Replay Attack

Finally, we will illustrate how the security architecture prevents the replay attack. In a replay attack, the attack will repeat or relay a message from a 5G device by eavesdropping. Since our security authentication will change the secret during each communication, the messages from replay attack will always have an outdated secret, namely, can always be found by our architecture.

5. Performance Evaluation

In the following, we simulate the performance of our scheme under the three kinds of attacks shown in Table 1.

5.1. Simulation Set up

We realize our scheme by Mininet emulator and OpenDaylight as shown in Figure 7. The OpenDaylight is used to perform an SDN controller. The SGW and LSA used in our scheme are modeled by OpenHIP. The hash value in the authentication scheme are generated by SHA-1. For each kind of attack, we test three kinds of settings—*sparse*, *moderate* and *dense*—with respect to the amount of hosts used in the network. There are four hosts in *sparse*, 50 hosts in *moderate* and 500 hosts in *dense*. In this work, we estimate the performance of our work by using the disconnect rate, which is derived as:

$$r = num_{dis}/100, \tag{1}$$

where num_{dis} is the number of disconnects in our simulations. In this paper, we compare the performance of our scheme with the state-of-the-art *TFSv1* [24] and *SDecurity* [25]. The *TFSv1* is widely used in today's commercial SDN Network but is vulnerable to the three kinds of attacks mentioned since it is lack of authentication. *SDecurity* is a typical Software Defined Security Experimental Framework. In our simulation, we assume that the attacker has the prior knowledge of our authentication process, and generate random secrets instead of using the outdated secret obtained by eavesdropping, namely, in this worst case, the attacker can have the chance to break through the security architecture when they guess the right secret key, and we assert that this rate will be much better when deployed in practice.

5.2. IP Spoofing

We first show the simulation results of our scheme under IP spoofing. For the sack of clarify, we term our scheme as *Robust* in short. In our simulations, the attacker will eavesdrop the communication channel. Once he achieves a authentication request, it can get the IP of the devices and then generate his own attacks between two adjacent authentication requests from this devices. As we mentioned before, more attempts can give the attacker more opportunities to guess the right secret key. We vary the number of attacks from 0 to 500 and record the associated disconnect rate to study the feasibility of our scheme. The overall simulation result is shown in Figure 8, as we can see that, when the number of attacks reaches 500, the disconnect rate is lower than 0.1%. Then, we compare the performance of *Robust* with two state-of-the-art ones. In Figure 9b, we can see that the disconnect rate of *Robust* keeps increasing as the number of attacks increases. For the state of art *TFSv*1 and *SDSecurity*, the IP spoofing attack is generated by solely using the IP information while not generating the secret key used in our scheme. Taking the case of *moderate* as the example in Figure 9b, we can see that the disconnect of our scheme is extremely low, in the worst case of 500 attacks, the maximum disconnect is still lower than 0.01%, which is much better than the disconnect rate from *TFSv*1 and *SDSecurity*. Taking a joint consideration of *sparse*, *moderate* and *dense*, another important observation is that the amount of hosts will affect the performance of SDN architecture; *SDSecurity* experiences a very efficient loss as the number of hosts increases. On the contrary, the disconnect rate of *Robust* does not vary a lot because the proposed scheme is a lightweight security architect, which fits the large scene very well. Both of the two observations indicate that the performance of the proposed scheme is much better than that from the state of the art under IP spoofing.

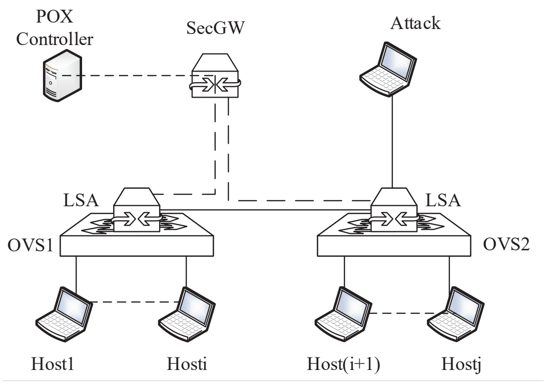


Figure 7. The simulation set up.

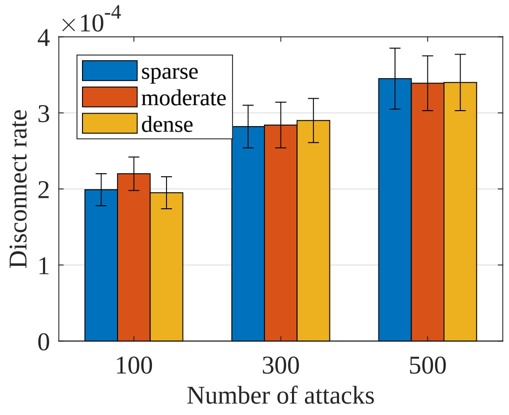


Figure 8. Performance under IP spoofing.

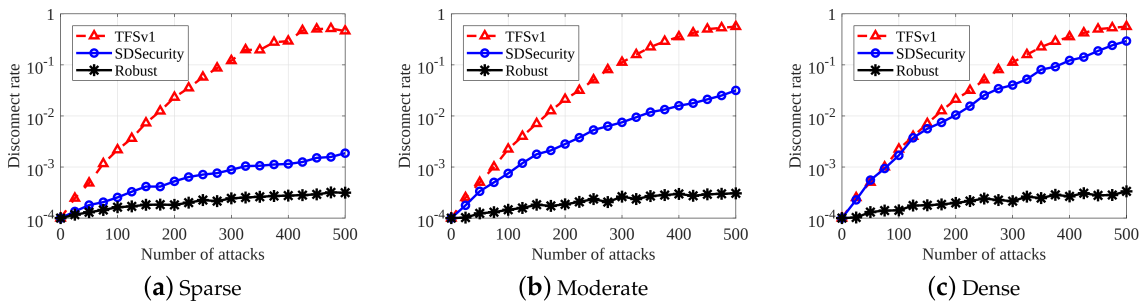


Figure 9. Comparison under IP spoofing. (**a**) sparse; (**b**) moderate; (**c**) dense.

5.3. MITM Attack

In Figures 10 and 11, we show the simulation results under MITM Attack. In an MITM attack, the attacker secretly relays and possibly alters the communication between two parties who believe they are directly communicating with each other. In our simulation, the attacker will hijack the communication channel of a device and then generate his own message to the SecE. Similar to the simulation in IP spoofing, we vary the number of attacks from 0 to 500 and record the associated disconnect rate. The overall simulation result is shown in Figure 10, as we can see that, when the number of attacks reaches 500, the disconnect rate is lower than 0.1%. Then, we compare the performance of *Robust* with two state-of-the-art ones. Taking the case of *moderate* as the example in Figure 11b, *Robust* still has a disconnect lower than 0.01%, which is much better than that from *TFSv*1 and *SDSecurity*. The performance of *TFSv*1 decreases a lot as the number of attacks increases because the communication channel of *TFSv*1 lacks authentication. For the *SDSecurity*, by its performance, still decreases a lot when used in a *dense* scene and therefore can not be used in the large scene. Both of the two observations indicate that our schemes can have better performance when used to detect MITM attacks.

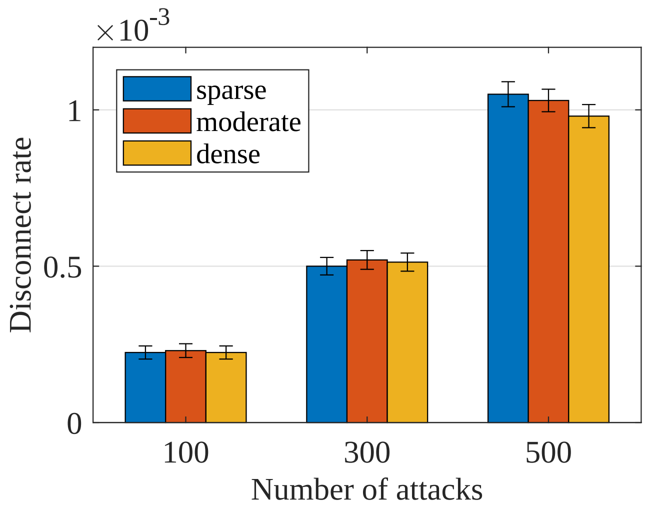


Figure 10. Performance under MITM attack.

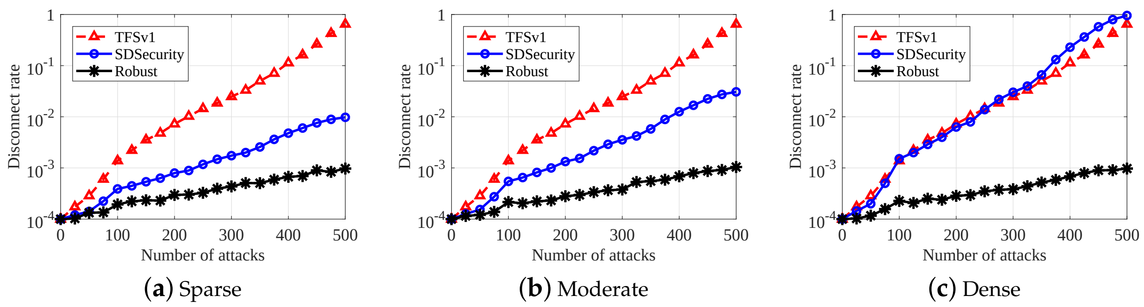


Figure 11. Comparison under MITM attack. (**a**) sparse; (**b**) moderate; (**c**) dense.

5.4. Replay Attack

Finally, we show the simulation results under Replay attack in Figures 12 and 13. In a replay attack, we control the attacker to eavesdrop on the communication channel and record the messages from devices, and then let the attack repeat the associated message to the SecE. We vary the number of attacks from 0 to 500 and record the associated disconnect rate. The overall simulation result is shown in Figure 12, as we can see that, when the number of attacks reaches 500, the disconnect rate is lower than 0.1%. Then, we compare the performance of *Robust* with two state-of-the-art ones. For clarifying, we take the case of *moderate* as an example. As we can see, *Robust* has the lowest disconnect rate under replay attack. In the worst case, our scheme has a disconnect lower than 0.01%, whose value is the lowest one among the three security architectures mentioned. For the two other architectures, this rate is about 75% for *TFSv*1 and 8% for *SDSecurity*. Although the disconnect rate of *SDSecurity* is low in moderate, its performance will decrease a lot with a disconnect rate of 73% in the *dense* scene. These observations indicate our scheme can have better performance when used to detect replay attacks. Considering the outstanding performance of our scheme under the three security attack mentioned in SDN architecture, we say that our scheme can meet the requirement of commercial use.

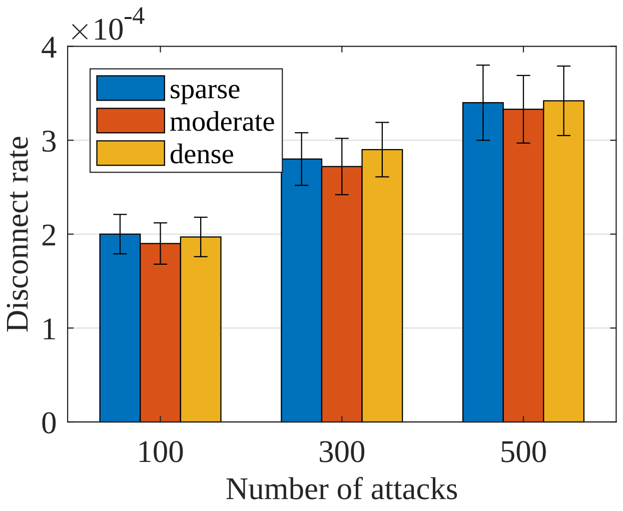


Figure 12. Performance under Replay attack.

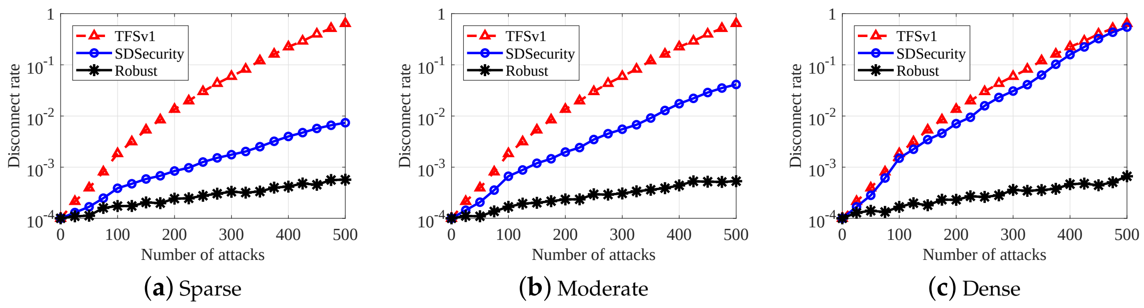


Figure 13. Comparison under replay attack. (**a**) sparse; (**b**) moderate; (**c**) dense.

6. Conclusions

In this paper, we reviewed the principles of SDN-5G architecture, and introduce the associated security problems in detail. For these problems, we propose a novel security architecture called robust, which is based on synchronized secret. This scheme leverages a common secret shared by the back-end system and the network users to avoid illegal service requests from malicious network attackers. Since the communication channel of SDN-5G network is made up of the control channel and the data channel, we design two different authentication methods respectively for the two channels. Extensive simulation results show that our scheme can achieve a disconnect rate less than 0.01% under the three mentioned classical attacks, which rate is much better than that from the state of art.

Author Contributions: Conceptualization, L.W.; Methodology, L.W. and Z.H.; Software, J.Y. and M.S.; Validation, J.Y.; Investigation, J.Y. and L.W.; Data Curation, J.Y.; Writing—Original Draft Preparation, J.Y.; Writing—Review & Editing, Z.H. and M.S.; Visualization, J.Y.; Supervision, L.W.; Project Administration, L.W.; Funding Acquisition, L.W. and Z.H.

Funding: This work was supported by the National Natural Science Foundation of China (Grant No. U1736216), the Natural Science Foundation of Jiangsu Province (BK20151460), and the University Natural Science Foundation of Jiangsu Province (16KJB520021).

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

References

1. Fan, K.; Gong, Y.; Liang, C.; Li, H.; Yang, Y. Lightweight and ultralightweight RFID mutual authentication protocol with cache in the reader for IoT in 5G. *Secur. Commun. Netw.* **2016**, *9*, 3095–3104. [CrossRef]
2. Zhang, Z.; Chai, X.; Long, K.; Vasilakos, A.V.; Hanzo, L. Full duplex techniques for 5G networks: Self-interference cancellation, protocol design, and relay selection. *IEEE Commun. Mag.* **2015**, *53*, 128–137. [CrossRef]
3. Ravindran, R.; Chakraborti, A.; Amin, S.O.; Azgin, A.; Wang, G. 5G-ICN: Delivering ICN services over 5G using network slicing. *IEEE Commun. Mag.* **2017**, *55*, 101–107. [CrossRef]
4. Benzekki, K.; El Fergougui, A.; Elbelrhiti Elalaoui, A. Software-defined networking (SDN): A survey. *Secur. Commun. Netw.* **2016**, *9*, 5803–5833. [CrossRef]
5. Casado, M.; Foster, N.; Guha, A. Abstractions for software-defined networks. *Commun. ACM* **2014**, *57*, 86–95. [CrossRef]
6. Felix, A.; Borges, N.; Wu, H.P.; Hanlon, M.; Birk, M.; Tschersich, A. Multi-layer SDN on a commercial network control platform for packet optical networks. In Proceedings of the Optical Fiber Communication Conference, San Francisco, CA, USA, 9–13 March 2014.
7. Kuklinski, S.; Chemouil, P. Network management challenges in software-defined networks. *IEICE Trans. Commun.* **2014**, *97*, 2–9. [CrossRef]
8. Giotis, K.; Argyropoulos, C.; Androulidakis, G.; Kalogeras, D.; Maglaris, V. Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. *Comput. Netw.* **2014**, *62*, 122–136. [CrossRef]
9. Braga, R.; Mota, E.; Passito, A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. In Proceedings of the IEEE Local Computer Networks (LCN), Denver, CO, USA, 11–14 October 2010; pp. 408–415.
10. Feamster, N. Outsourcing home network security. In Proceedings of the ACM SIGCOMM Workshop on Home Networks, New Delhi, India, 30 August–3 September 2010; pp. 37–42.
11. Jin, R.; Wang, B. Malware detection for mobile devices using software-defined networking. In Proceedings of the 2013 Second GENI Research and Educational Experiment Workshop, Salt Lake City, UT, USA, 20–22 March 2013; pp. 81–88.
12. Jafarian, J.H.; Al-Shaer, E.; Duan, Q. Openflow random host mutation: Transparent moving target defense using software defined networking. In Proceedings of the First Workshop on Hot Topics in Software Defined Networks, Helsinki, Finland, 13 August 2012; pp. 127–132.
13. Kampanakis, P.; Perros, H.; Beyene, T. SDN-based solutions for Moving Target Defense network protection. In Proceedings of the IEEE World of Wireless, Mobile and Multimedia Networks (WoWMoM), Boston, MA, USA, 14–17 June 2014; pp. 1–6.
14. Sherwood, R.; Gibb, G.; Yap, K.K.; Appenzeller, G.; Casado, M.; McKeown, N.; Parulkar, G. Flowvisor: A network virtualization layer. *OpenFlow Switch Consortium Tech. Rep* **2009**, *1*, 132.
15. Al-Shaer, E.; Al-Haj, S. FlowChecker: Configuration analysis and verification of federated OpenFlow infrastructures. In Proceedings of the 3rd ACM Workshop on Assurable and Usable Security Configuration, Chicago, IL, USA, 4 October 2010; pp. 37–44.
16. Canini, M.; Venzano, D.; Peresini, P.; Kostic, D.; Rexford, J. A NICE way to test OpenFlow applications. In Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation, San Jose, CA, USA, 25–27 April 2012.
17. Bernardo, D.V.; Chua, B.B. Introduction and analysis of SDN and NFV security architecture (SN-SECA). In Proceedings of the Advanced Information Networking and Applications (AINA), Gwangju, Korea, 25–27 March 2015; pp. 796–801.

18. Feghali, A.; Kilany, R.; Chamoun, M. SDN security problems and solutions analysis. In Proceedings of the IEEE International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS), Paris, France, 22–24 July 2015; pp. 1–5.
19. Alsmadi, I.; Xu, D. Security of Software Defined Networks: Asurvey. *Comput. Secur.* **2015**, *53*, 79–108. [CrossRef]
20. Shin, S.W.; Gu, G. Cloudwatcher: Network security monitoring using openflow in dynamic cloud networks. In Proceedings of the 2012 IEEE International Conference on Network Protocols (ICNP), Austin, TX, USA, 30 October–2 November 2012; pp. 1–6.
21. Hong, S.; Xu, L.; Wang, H.; Gu, G. Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures. In Proceedings of the Network and Distributed System Security Symposium, San Diego, CA, USA, 8–11 February 2015.
22. Dhawan, M.; Poddar, R.; Mahajan, K.S.; Mann, V. SPHINX: Detecting Security Attacks in Software-Defined Networks. In Proceedings of the Network and Distributed System Security Symposium, San Diego, CA, USA, 8–11 February 2015.
23. Wang, K.; Qi, Y.; Yang, B.; Xue, Y.; Li, J. LiveSec: Towards Effective Security Management in Large-Scale Production Networks. In Proceedings of the 2012 32nd International Conference on Distributed Computing Systems Workshops, Macau, China, 18–21 June 2012; pp. 451–460.
24. Kempf, J.; Bellagamba, E.; Kern, A.; Jocha, D.; Takács, A.; Sköldström, P. Scalable fault management for OpenFlow. In Proceedings of the IEEE Communications (ICC), Ottawa, ON, Canada, 10–15 June 2012; pp. 6606–6610.
25. Darabseh, A.; Alayyoub, M.; Jararweh, Y.; Benkhelifa, E.; Vouk, M.A.; Rindos, A. SDSecurity: A Software Defined Security experimental framework. In Proceedings of the 2015 IEEE International Conference on Communication Workshop (ICCW), London, UK, 8–12 June 2015.