Software-Defined Networking Security: Pros and Cons

Citation	Dabbagh, M., Hamdaoui, B., Guizani, M., & Rayes, A. (2015). Software-Defined Networking Security: Pros and Cons. IEEE Communications Magazine, 53(6), 73-79. doi:10.1109/MCOM.2015.7120048
DOI	10.1109/MCOM.2015.7120048
Publisher	IEEE - Institute of Electrical and Electronics Engineers
Version	Accepted Manuscript
Terms of Use	http://cdss.library.oregonstate.edu/sa-termsofuse

Software-Defined Networking Security: Pros and Cons

Mehiar Dabbagh, Bechir Hamdaoui, Mohsen Guizani† and Ammar Rayes‡
Oregon State University, Corvallis, OR 97331, dabbaghm,hamdaoub@onid.orst.edu
† Qatar University, mguizani@ieee.org
‡ Cisco Systems, San Jose, CA 95134, ‡ rayes@cisco.com

***Abstract*—Software-Defined Networking (SDN) is a new networking paradigm that decouples the forwarding and control planes—traditionally being coupled with one another—while adopting a logically centralized architecture aiming to increase network agility and programability. While many efforts are currently being made to standardize this emerging paradigm, careful attentions need to be paid to security at this early design stage too, rather than waiting until the technology becomes mature, thereby potentially avoiding previous pitfalls made when designing the Internet in the 80's. This article focuses on the security aspects of SDN networks. We begin by discussing the new security pros that SDN brings and by showing how some of the long-lasting issues in network security can be addressed by exploiting SDN capabilities. Then, we describe the new security threats that SDN is faced with and discuss possible techniques that can be used to prevent and mitigate such threats.**

I. What's SDN?

Just when you start getting used to the IT buzzwords, such as mobile clouds [1], internet of things [2], and network virtualization [3], the IT people hit you with a new one. Software-Defined Networking (SDN) received a great attention from the academia and industry recently, and quickly became the new buzzword. This new networking paradigm (illustrated in Fig. 1) is based on the idea of decoupling the network's control plane from the forwarding plane, which results in turning traditional network's complicated routing devices into simple switches whose job is merely to follow the policy that is depicted by an intelligent and programmable logically centralized controller. This is different from traditional networks shown in Fig. 2, where routing devices are performing both forwarding and control functions in a distributed fashion using static protocols.

In addition to those core differences that distinguish SDN networks from traditional ones, there are also implementation details that are specific to SDN networks. Unlike traditional networks where networking devices decide how an incoming packet should be handled based solely on its IP destination address, SDN follows a *flow-based* forwarding scheme where multiple header fields depict how the incoming packet should be handled. Furthermore, all network devices in SDN networks are recording traffic statistics, something that was performed by only few devices (if any) in traditional networks.

SDN brings promising opportunities to network management in terms of simplicity, programability and elasticity. These opportunities were quickly recognized by big industrial corporations which started at an early stage funding research projects that aim at developing SDN. Today, the major networking vendors such as Cisco are releasing network infrastructure that supports SDN [4]. Furthermore, the paradigm turned to be just what is needed for managing cloud and data centers. In fact, Google has revealed the use of SDN for managing the networking infrastructure of their data centers [5]. While many efforts are currently made to improve and standardize SDN, we believe that further attention should be paid to security.

In this article, we analyze SDN from a security perspective with the objective of shedding the light on the new security capabilities and the new security threats that are brought by this new paradigm. The remaining is organized as follows. Section II briefly explains how the SDN paradigm works. Section III analyzes the security advantages that SDN brings while highlighting what characteristics differentiate SDN from traditional networks, and how these characteristics can be exploited to improve network security. Section IV discusses the new security issues that SDN faces, and describes what techniques could be used to prevent, mitigate or recover from some of these issues. Section V describes the current state of the SDN standards with respect to security. Finally, Section VI summarizes and concludes the article.

II. How Does SDN Work?

Since the forwarding and the control planes are separated in the SDN paradigm, we explain next how each plane works. Throughout this section, we refer to Fig. 1 for illustration.

Forwarding Plane: This plane is made up of simple switches that are interconnected to form the physical network. The role of these switches is to forward packets based on the control plane's routing policy. In order to achieve this role, each switch maintains a *forwarding table* whose entries are basically forwarding rules that get installed by the control plane. Each rule in the table is made up of three fields: pattern field, counter field and action field. The pattern field defines a *flow* which is basically a set of packet header fields values. Upon the reception of a packet, the switch searches its forwarding table trying to find a rule whose pattern field matches the packet's header values. Once such a rule is found, the rule's counter field gets incremented and the rule's action field gets executed, which could be: i) forward the packet as it is or after modifying some of its header fields to a specific port ii) drop the packet or iii) report the packet back to the controller.

Control Plane: This plane represents the network brain and is responsible for monitoring the network, making routing decisions, and programming the physical network how to behave. It is made up of three main layers: the Network Operating System layer, the Network Abstraction layer and the Application layer. As for the Network Operating System, it is connected to each switch in the network using a duplex link. This layer collects state information from the network switches such as: connectivity

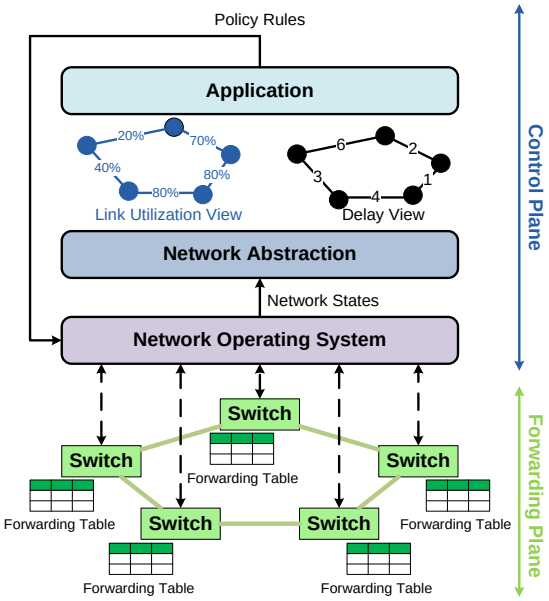


Fig. 1: an SDN network.

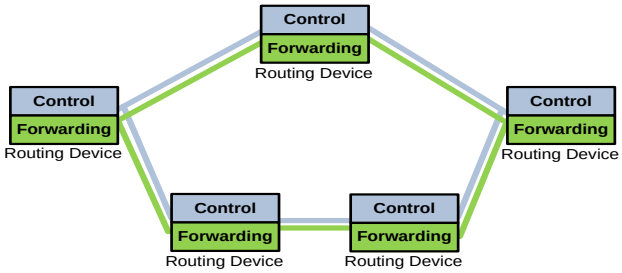


Fig. 2: a traditional network.

states, link delays, link utilization, etc. and passes these state information to the Network Abstraction layer. The role of the Abstraction layer is to extract suitable network representation called views out of these collected data as it has a global view of the entire network. An example of a network representation would be a graph where vertices represent switches and edges represent links, where the edge's weight represents the link's delay or utilization. The Application layer takes one or multiple representations as input, runs a certain algorithm that finds a routing policy that achieves certain objectives such as minimizing delay or avoiding link over-utilization, and returns a set of forwarding rules that get passed back to the Network Operating System which distributes these rules to the network switches.

III. SDN Security Pros

The new structure of the SDN paradigm brings great benefits to the networking scheme. The way the functionalities are abstracted in the SDN paradigm allows writing high-level software applications to manage the network without worrying how to configure the underlying physical network. This is only one of numerous advantages that SDN brings. Our aim is not to discuss those numerous general benefits (e.g. see [6] for further details), but rather to focus on those related to security. We identify three core characteristics that differentiate SDN networks from traditional ones from a security perspective. We discuss next each one of those characteristics explaining why traditional networks lack these characteristics, and how each one can be exploited to improve network security.

A. Global Network View

The fact that the controller in the SDN paradigm has a global network view is perhaps SDN's greatest security advantage over traditional networks. This global network view is attributed to centralization and to the fact that all the elements in the network are collecting and reporting traffic statistics. This is different from traditional distributed networks whose devices require exchanging lots of information and waiting for a convergence time in order to infer partially the state of the remaining parts of the network, and where only few devices (if any) are logging traffic statistics. The security advantages that the global network view brings to the SDN paradigm are the following:

1) **Network-Wide Intrusion Detection.** The global network view allows the SDN controller to run a network-wide Intrusion Detection System (IDS) that analyzes the traffic statistics collected from all the network switches in order to detect malicious traffic. This is different from traditional networks, where IDS is a device that is usually installed on a certain part of the network and thus provides limited detection capabilities as it has limited visibility. After collecting traffic statistics from the SDN switches, there are two ways for an IDS to operate:
   i) *Misuse Detection:* which is based on the idea of building profiles, called signatures, for known attacks. The behavior of the network is monitored constantly and an intrusion is reported if the monitored behavior matches any of those signatures.
   ii) *Anomaly Detection:* The network traffic is profiled when no malicious activity is going on by basically capturing the general characteristics of the packets originating from trusted hosts within the network while running trusted applications. An intrusion is reported if the network behavior deviates significantly from those profiles.

   Each one of those two detection mechanisms has its pros and cons. Anomaly Detection has the ability to identify new attacks and requires less in-depth knowledge about malicious behavior. However, this method produces more false positive alarms compared to Misuse Detection as a deviation from the profiled traffic may not be necessary a malicious behavior but rather a new normal behavior different from the profiled one. Researchers are still investigating improvements for the IDS framework in SDN with special focus on reducing processing overhead, increasing detection accuracy and making the framework adaptive where new normal or malicious behavior can be learned automatically.
2) **Detection of Malicious Switch Behavior.** The global network view not only supports more efficient detection of intrusions originated from malicious traffic, but also helps in detecting network switches' malicious behavior. Consider for example the case where a networking device is dropping some or all incoming packets creating a *black hole*. Identifying which routing device is responsible for

TABLE I: SDN security pros over traditional networks.

SDN Characteristic	Attributed to	Security Use
Global Network View	• Centralization • Traffic Statistics Collection	• Network-Wide Intrusion Detection • Detection of Switch's Malicious Behavior • Network Forensics
Self-Healing Mechanisms	• Conditional Rules • Traffic Statistics Collection	• Reactive Packet Dropping • Reactive Packet Redirection
Increased Control Capabilities	• Flow-based Forwarding Scheme	• Access Control

that is a non-easy task in traditional networks. Probing techniques that are usually used in those networks, such as Traceroute [7], may not be effective for detection as the malicious switch may forward properly only those probes to hide its malicious behavior. Identifying such misbehaving behavior is easier in the SDN paradigm as switches report periodically to the control plane the number of received, forwarded and dropped packets. By analyzing these reports, the misbehaving switch can be identified, or at least the list of suspected switches would be narrowed down. This is done even when the malicious switch claims in its report that all its incoming packets are forwarded properly to its neighbors as the neighboring switches' reports will indicate correctly how many packets were actually received from that switch. However, it is still hard to differentiate packet droppings caused by link errors from those originating from malicious behavior. This is even further challenging when multiple malicious switches collude together to hide their malicious activity.

3) **Network Forensics.** Finally, the fact that the control plane logs the global view of the network over time facilitates performing forensic analysis. One can return back to the logged traffic in order to understand how an undetected attacks was performed, something that is very helpful for developing effective defensive mechanisms against future instances of those attacks. Furthermore, this helps in identifying and isolating compromised hosts in addition to tracing back the person/organization behind those attacks.

B. Self-Healing Mechanisms

Another characteristic that distinguishes SDN from traditional networks is the fact that it is supplied with self-healing mechanisms. *Conditional rules* are an example of such mechanisms that were introduced to the SDN paradigm in [8]. These rules are installed on the switches by the control plane and get activated once a certain condition is met. The condition is usually related to the switch's collected statistics, such as having the number of packets belonging to a certain flow received during a certain period of time exceeds a predefined threshold. The activated rule specifies how the switch should respond when the specified condition is met. These reactions provide automated resiliency against attacks. The reaction specified by the conditional rule could be to drop the packets specified by the rule or to forward those packets through different paths to mitigate the load on certain parts of the network. This provides resiliency against Denial of Service (DoS) attacks targeting network hosts or network links. The reaction could be to modify the destination address of certain packets so that they get delivered to a *honeypot*, which is an isolated and monitored host that is used as a trap to collect further information about malicious activities. This redirection is done in a stealthy way without having the originator of the traffic observe that. This is very useful for detecting *botnets*, which are a set of connected compromised hosts, that are controlled by an attacker and used as a platform to launch distributed attacks against other parts of the network.

C. Increased Control Capabilities

SDN networks have more control capabilities compared to traditional networks. This is attributed to the adoption of the flow-based scheme in SDN, where multiple header fields define how packets should be handled in the network rather than relying merely on the destination address as in traditional networks.

Thus the SDN controller can have better access control by specifying what types of packets should be carried within the network based on the payload type, the source address or any other header field value. The rules installed by the controller can for example allow only TCP packets that are originating from a certain host to be routed through the network. This helps in limiting malicious traffic from entering to or from originating from any switch in the SDN network. This is different from traditional networks, where networking devices forward packets blindly, leaving access decisions to the receiving end where a firewall usually sits and is required to inspect the payloads of all delivered packets.

IV. SDN Security Threats and Countermeasures

Having explained the security advantages that SDN brings which could be summarized in Table I, we explain now the new security threats and attacks that the SDN paradigm is exposed to. This section divides those attacks and threats into three categories based on what part of the SDN paradigm they target, i.e., the forwarding plane, the control plane or the links connecting the two planes. Countermeasure techniques that could be used to prevent, mitigate or recover from some of those attacks are also described while highlighting the challenges encountered when developing these defensive mechanisms.

A. Forwarding Plane Attacks and Countermeasures

We describe first some of the security threats the SDN's forwarding plane faces, as well as some possible countermeasures.

Switch DoS: Given that current switches have limited storage capacity and that the rules produced by the controller should cover all flows (all header fields possibilities), it becomes clear that it is next to impossible to store all these rules in current switches. A reactive caching mechanism is thus adopted instead

in current SDN implementations where whenever a switch does not find a matching rule for the flow of one of its incoming packets, the packet gets stored temporary on a switch buffer, and a query is sent to the controller asking for the missing rule. Once that rule is received, the packet gets processed based on that rule and the rule gets cached in the switch's forwarding table so that the next packets of that flow get processed directly.

This reactive caching mechanism makes switches vulnerable to a DoS attack where a malicious user floods the switch with packets of large payloads that belong to different flows. The rules of some of these flows may not be cached in the forwarding table which requires sending queries to the control plane. As a result, the switch ends up storing some of those large packets in its buffer waiting for the control responses. This buffer can be filled up quickly, especially if those packets have large payloads, causing legitimate packets belonging to new flows to be dropped as there is no enough space on the buffer to store those packets.

Multiple solutions were proposed to address this attack. One solution is proactive caching, where switches don't wait to receive new packets to request rules, but rather cache *a priori* as many rules as their table can fit. This technique turns out to be very efficient especially when combined with using aggregate rules, where the installed rules cover ranges of header fields rather than single values, which compresses the number of needed rules. Finally, having a low delay on the links that connect switches with the control plane helps processing the buffered packets quickly, making this attack harder to take place. We will discuss later how to maintain low switch-controller communication delay.

Packet Encryption and Tunnel Bypassing: The adoption of the flow-based forwarding scheme allows SDN networks to customize how packets with different payloads are to be treated. This could be used for access control purposes, where the controller can specify what types of packets are allowed to flow inside and through the network. Packets with different payloads must be dropped or sent to the controller for further inspection.

Although the flow-based scheme brings new network management capabilities, it is not clear how this scheme should deal with encrypted packets where not all the packet's header fields are visible to the network switches. In fact, entire packets (headers and payloads) can be concealed from the network switches by creating tunnels that encapsulate an encryption of those packets within other packets. Once those packets are received at the other end of the tunnel, the inner packet gets decrypted, decapsulated and routed based on the policy of the new network that the other end of the tunnel belongs to. These encrypted packets and tunnel connections allow malicious users to skip their network border and bypass their network's access control policy. This can be addressed as in [9] by constructing models to identify the payload type of the encrypted packets based on analyzing traffic statistics such as message length, inter-packet arrival times, etc.

B. Control Plane Attacks and Countermeasures

We now discuss some potential attacks and their countermeasures, arising from the centralized nature of SDN's control plane.

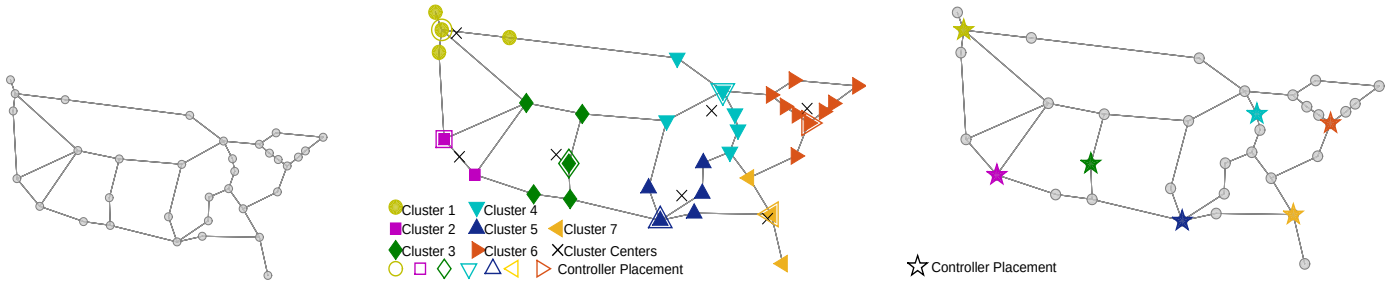
DDoS Attack: The control plane is susceptible to Distributed Denial of Service (DDoS) attacks where multiple compromised hosts distributed in the network may synchronously flood the network switches with packets. Since not all rules will already be available in the switches' tables, many queries will be generated and sent to the controller which ends up utilizing the controller's processing power causing legitimate queries to be delayed or dropped.

One solution to such attacks is *replication*, where multiple physical controllers manage the network rather than a single one. However, the forwarding plane should continue to operate as if a single controller is programming the entire network. This is referred to as *logical centralization* and requires these multiple controllers to be connected to each other via secure links in order to maintain consistent rules all the time and to arrange how they should split up managing the network.

When replication is adopted, each switch will be connected to multiple controllers, and one of those controllers will be selected to be the master for that switch. Queries are directed to the master controller in the general case or to one of the remaining connected controllers when the master fails. The amount of queries generated by the switches, the processing capabilities of the controllers and the switch-controller link delays are all taken into account when assigning which controller should be the master for each switch in the network. This is done with the objective of balancing the load on the controllers while also minimizing the switch-controller communication delays. These assignments need not be static but can rather be dynamic, where a controller with high load can ask one of the remaining light loaded controllers to become the master for some of its assigned switches. This helps in keeping the load balanced among the controllers and provides resiliency against DDoS attacks.

When multiple controllers manage the network, deciding where to deploy these controllers is an important design question from a performance and resiliency perspectives. The distance separating the switch from its master controller is a key factor to be taken into account when making those placements. Keeping this distance short guarantees low delay on the switch-controller communication link, which improves network responsiveness and makes it harder for switch DoS attacks to take place.

The controller placement problem is illustrated using the Internet2 OS3E topology which is made up of a set of interconnected switches as shown in Fig. 3(a), and where k replicated controllers need to be deployed on some network nodes. We are searching for a placement that minimizes the average switch-master delay, where each switch selects the controller reached with the shortest path to be its master. The problem is known to be NP-hard [10], and thus finding global optimal placements becomes computationally infeasible as the size of the network and/or the number of controllers grow. However, suboptimal solutions can still be obtained by clustering the switches into k clusters, with each cluster containing a set of switches that are geographically close to one another. Fig. 3(b) shows the resulting clusters, when using k-Means [11], for the OS3E topology when $k = 7$, where switches belonging to different clusters are marked by different shapes/colors, and cluster centers are marked by 'x'. Next, for each cluster, the node that has the least average shortest-path distance to the remaining cluster members is selected for controller placement as illustrated in Fig. 3(b). We also show in Fig. 3(c) the global optimal controllers placement which was obtained by a brute-force search. Observe that five out



(a) Internet2 OS3E topology. (b) Clustering-based controller placement. (c) Global optimal controller placement.

Fig. 3: Illustration of the controller placement decisions for $k = 7$ where the objective is to minimize the average switch-controller distance.

of seven locations are optimal locations, whereas the other two are adjacent to the optimal locations. In fact, the average switch-master distance based on the clustering-based placements is only 1.9% more than the optimal solution.

In addition to switch-controller delays, there are other factors that should be taken into account when making controller placements. For example, the ability to maintain switch-controller connectivity when some links get compromised is an important factor that needs to be accounted for when deciding on where to place controllers. This requires further investigation.

Compromised Controller Attacks: We discussed previously how to make the controller robust against DDoS failures. But we did not discuss the scenario when the attacker, somehow, gains access to the controller, putting all switches controlled by the compromised controller under the mercy of the attacker. The compromised controller can program those switches to drop all incoming traffic, or use them as a platform to launch serious attacks against other targets, such as directing all received packets to a victim so as to deplete its resources.

SDN must have some sort of resiliency against compromised controller attacks. Control replication could be used to resist such attacks. However, this solution will not be efficient if all the controllers are installed on similar platforms as they would all share the same vulnerabilities, allowing the attacker to break into all of them once it succeeds to break into one of them. Diversity among the platforms hosting those controllers is thus an essential condition for this resiliency technique to work. Furthermore, switches no longer query and receive responses from a master controller but rather communicate with all of the controllers that they are connected to. A majority voting technique can be used to decide what rule to follow in case different rules were received from the connected controllers. A switch connected to $2m + 1$ controller in that case would be resilient to up to m compromised controllers. Observe that the larger the number of controllers the switch is connected to, the higher it is resilient to be controlled by a compromised controller. Deciding how many controllers each switch should be connected to can be formulated as an optimization problem as in [12] while taking the processing capacity of those controllers as a constraint and where each switch in the network has a different security requirement (e.g. switches on the edge of the network have lower security requirement compared to those in the core of the network).

C. Forwarding-Control Link Attacks and Countermeasures

Sending unencrypted communication messages on the link connecting the control and forward planes clearly makes the link susceptible to a *man-in-the-middle attack*. The attacker in that case can infer the control policy by eavesdropping the communication exchanged on the link. Even worse, the rules sent from the controller can be tampered or new rules can be fabricated giving the attacker full control over the switch. Thus it is clear that the link layer must be secured against those attacks. Encryption must be used to prevent eavesdropping while the encrypted message need to include some proof of the entity who originated those messages. A timestamp needs also to be included in the encrypted messages in order to prevent *replay attacks*, where a malicious user collects the encrypted rules sent by the controller, and sends them back at a latter point of time causing the switch to return back to use old policy rules.

For convenience, a summary of the security issues that SDN networks are exposed to along with possible countermeasures that we discussed in this section is provided in Table II.

V. SDN Standards: OpenFlow's Robustness to Security Threats

We explained in previous sections the pros and cons of the SDN paradigm from a security perspective. We discuss in this section the current state of the SDN standards in terms of exploiting SDN's security capabilities and of adopting countermeasures against the newly exposed security threats. We focus on OpenFlow as it was the first released SDN standard and as it has gone through multiple revisions before becoming now widely deployed by networking vendors [13]. Our discussion are based on the standard's latest specifications released in December, 2014 [14].

Many of the SDN security advantages are exploited well in OpenFlow as the standard's specifications require switches to collect traffic statistics and to adopt the flow-based forwarding scheme. However, OpenFlow neither enforces switches to support conditional rules nor specifies how such rules should be handled. This limits the SDN's self-healing ability as reactions to malicious traffic are not triggered directly by switches, but could be triggered by generating new rules by the controller later after analyzing the traffic statistics collected by the switches.

As for OpenFlow's countermeasures, the standard supports proactive rule caching though reactive rule caching is more widely used. However, the standard completely ignores how

TABLE II: Summary of the new security issues that SDN networks are exposed to along with possible countermeasures.

Targeted Level	Malicious Behavior	Caused by	Possible Countermeasures
Forwarding Plane	Switch DoS	• Limited Forwarding Table Storage Capacity • Enormous Number of Flows • Limited Switch's Buffering Capacity	• Proactive Rule Caching • Rule Aggregation • Increasing Switch's Buffering Capacity • Decreasing Switch-Controller Communication Delay
	Packet Encryption and Tunnel Bypassing	• Invisible Header Fields	• Packet Type Classification Based on Traffic Analysis
Control Plane	DDoS Attack	• Centralization • Limited Forwarding Table Storage Capacity • Enormous Number of Flows	• Controller Replication • Dynamic Master Controller Assignment • Efficient Controller Placement
	Compromised Controller Attacks	• Centralization	• Controller Replication with Diversity • Efficient Controller Assignments
Forwarding-Control Link	Man-in-the-middle Attacks	• Communication Messages Sent in Clear • Lack of Authentication	• Encryption • Use of Digital Signatures
	Replay Attacks	• Communication Messages Sent in Clear • Lack of Timestamping	• Encryption • Timestamp Inclusion in Encrypted Messages

the incoming packets should be handled when some of their headers are hidden due to encryption. Although controller replication is supported in OpenFlow, no specifications were released regarding where to place the replicated controllers or how to make master controller selection [14]. Furthermore, dynamic master control assignment is not directly supported by OpenFlow controllers such as NOX [15]. Finally, one of the major security concerns with OpenFlow is the fact that it leaves it optional of whether or not to encrypt the controller-switch communication channel [14], which makes this channel completely vulnerable to the threats discussed in Section IV-C. We anticipate that future releases of SDN standards will address these raised concerns.

VI. Conclusion

We explained in this article how SDN works and analyzed its pros and cons from a security perspective. In summary, three key characteristics give SDN great security advantages over traditional networks, which are: the global network view, the self-healing mechanisms and the additional control capabilities. While SDN provides promising solutions to many security problems, it is also exposed to new threats and attacks targeting the forwarding plane, the control plane, or the links connecting the two planes. Several preventive and mitigation techniques were also described to address some of those security issues. The current state of the SDN standards was also analyzed with respect to security. We hope that SDN networks will exploit further the paradigm's security advantages while also addressing the new security concerns in the future.

VII. Acknowledgment

This work was made possible by NPRP grant # NPRP 5-319-2-121 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

[1] T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," *IEEE Network*, vol. 27, no. 5, 2013, pp. 12–19.

[2] L. Atzori, A. Iera, and G. Morabito, "From "smart objects" to "social objects": The next evolutionary step of the internet of things," *IEEE Communications Magazine*, vol. 52, no. 1, 2014, pp. 97–105.

[3] T. Taleb, "Toward carrier cloud: Potential, challenges, and solutions," *IEEE Wireless Communications*, vol. 21, no. 3, 2014, pp. 80–91.

[4] "Software-Defined Networking: Why we like it and how we are building on it," *Cisco Inc., White Paper*, 2013.

[5] S. Jain, et al., "B4: Experience with a globally-deployed software defined WAN," in *Proceedings of the ACM SIGCOMM conference*, 2013, pp. 3–14.

[6] M.R. Sama, L. Contreras, J. Kaippallimalil, I. Akiyoshi, Q. Haiyang, and N. Hui, "Software-defined control of the virtualized mobile packet core," *IEEE Communications Magazine*, vol. 53, no. 2, 2015, pp. 107–115.

[7] M. Dabbagh, N. Sayegh, A. Kayssi, I. Elhajj, and A. Chehab, "Fast dynamic internet mapping," *Future Generation Computer Systems*, 2014.

[8] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "AVANT-GUARD: scalable and vigilant switch flow management in software-defined networks," in *Proceedings of the ACM SIGSAC conference on Computer & communications security*, 2013, pp. 413–424.

[9] Z. Fadlullah, T. Taleb, A. Vasilakos, M. Guizani, and N. Kato, "DTRAB: combating against attacks on encrypted protocols through traffic-feature analysis," *IEEE/ACM Transactions on Networking (TON)*, vol. 18, no. 4, 2010, pp. 1234–1247.

[10] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the first ACM workshop on Hot topics in software defined networks*, 2012, pp. 7–12.

[11] J. Han, M. Kamber, and J. Pei, "Data mining: concepts and techniques," 2006, Morgan kaufmann.

[12] H. Li, P. Li, S. Guo, and A. Nayak, "Byzantine-resilient secure software-defined networks with multiple controllers in cloud," *IEEE Transactions on Cloud Computing*, no. 99, 2014, pp. 1–1.

[13] R. Jain and S. Paul, "Network virtualization and software defined networking for cloud computing: a survey," *IEEE Communications Magazine*, vol. 51, no. 11, 2013, pp. 24–31.

[14] "OpenFlow switch specifications version 1.5.0," *Open Networking Foundation*, December 2014.

[15] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," in *Proceedings of ACM SIGCOMM workshop on Hot topics in software defined networking*, 2013, pp. 7–12.

Mehiar Dabbagh received his B.S. degree in Telecommunication Engineering from the University of Aleppo, Syria, in 2010 and the M.S. degree in Electrical and Computer Engineering from the American University of Beirut (AUB), Lebanon, in 2012. During his Master's studies, he worked as a research assistant in Intel-KACST Middle East Energy Efficiency Research Center (MER) at the American University of Beirut (AUB), where he developed techniques for software energy profiling and software energy-awareness. Currently, he is a Ph.D. student in Electrical Engineering and Computer Science at Oregon State University (OSU), where his research focus is on how to make cloud centers more energy efficient. His research interests also include: Cloud Computing, Energy-Aware Computing, Networking, Security and Data Mining.

Bechir Hamdaoui (S'02-M'05-SM'12) is presently an Associate Professor in the School of EECS at Oregon State University. He received the Diploma of Graduate Engineer (1997) from the National School of Engineers at Tunis, Tunisia. He also received M.S. degrees in both ECE (2002) and CS (2004), and the Ph.D. degree in Computer Engineering (2005) all from the University of Wisconsin-Madison. His research interests span various topics in the areas of wireless communications and computer networking systems. He has won the NSF CAREER Award (2009), and is presently an AE for IEEE Transactions on Wireless Communications (2013-present), and Wireless Communications and Computing Journal (2009-present). He also served as an AE for IEEE Transactions on Vehicular Technology (2009-2014) and for Journal of Computer Systems, Networks, and Communications (2007-2009). He served as the program chair for SRC in ACM MobiCom 2011 and many IEEE symposia/workshops, including ICC, IWCMC, and PERCOM. He also served on the TPCs of many conferences, including INFOCOM, ICC, and GLOBECOM. He is a Senior Member of IEEE, IEEE Computer Society, IEEE Communications Society, and IEEE Vehicular Technology Society.

Mohsen Guizani (S'85-M'89-SM'99-F'09) is currently a Professor at the Computer Science & Engineering Department in Qatar University. Qatar. He also served in academic positions at the University of Missouri-Kansas City, University of Colorado-Boulder, Syracuse University and Kuwait University. He received his B.S. (with distinction) and M.S. degrees in Electrical Engineering; M.S. and Ph.D. degrees in Computer Engineering in 1984, 1986, 1987, and 1990, respectively, all from Syracuse University, Syracuse, New York. His research interests include Wireless Communications and Mobile Computing, Computer Networks, Cloud Computing, Cyber Security and Smart Grid. He currently serves on the editorial boards of several international technical journals and the Founder and EiC of "Wireless Communications and Mobile Computing" Journal published by John Wiley. He is the author of nine books and more than 400 publications in refereed journals and conferences (with an h-index=30 according to Google Scholar). He received two best research awards. Dr. Guizani is a Fellow of IEEE, member of IEEE Communication Society, and Senior Member of ACM.

Ammar Rayes Ph.D., is a Distinguished Engineer at Cisco Systems and the Founding President of The International Society of Service Innovation Professionals, www.issip.org. He is currently chairing Cisco Services Research Program. His research areas include: Smart Services, Internet of Everything (IoE), Machine-to-Machine, Smart Analytics and IP strategy. He has authored / co-authored over a hundred papers and patents on advances in communications-related technologies, including a book on Network Modeling and Simulation and another one on ATM switching and network design. He is an Editor-in-Chief for "Advances of Internet of Things" Journal and served as an Associate Editor of ACM "Transactions on Internet Technology" and on the Journal of Wireless Communications and Mobile Computing. He received his BS and MS Degrees in EE from the University of Illinois at Urbana and his Doctor of Science degree in EE from Washington University in St. Louis, Missouri, where he received the Outstanding Graduate Student Award in Telecommunications.