2015 29th International Conference on Advanced Information Networking and Applications Workshops

SDN based architecture for IoT and improvement of the security

Olivier FLAUZAC and Carlos GONZÁLEZ and Abdelhak HACHANI and Florent NOLOT
University of Reims Champagne-Ardenne, CReSTIC
Reims, France
Email: olivier.flauzac@univ-reims.fr, carlos.gonzalez-santamaria@etudiant.univ-reims.fr,
hachani.abdelhak@gmail.com, florent.nolot@univ-reims.fr

***Abstract*—With the exponential growth of devices connected to the Internet, security networks as one of the hardest challenge for network managers. Maintaining and securing such large scale and heterogeneous network is a challenging task. In this context, the new networking paradigm, the Software Defined Networking (SDN), introduces many opportunities and provides the potential to overcome those challenges. In this article, we first present a new SDN based architecture for networking with or without infrastructure, that we call an SDN domain. A single domain includes wired network, wireless network and Ad-Hoc networks. Next, we propose a second architecture to include sensor networks in an SDN-based network and in a domain. Third, we interconnect multiple domains and we describe how we can enhanced the security of each domain and how to distribute the security rules in order not to compromise the security of one domain. Finally, we propose a new secure and distributed architecture for IoT (Internet of Things).**

***Index Terms*—SDN, Internet of Things (IoT), Ad-Hoc networks, Security.**

I. Introduction

The protection of data transmission has been an issue even long before the creation of the Internet. However, with the latest Internet evolution, billions of devices will be connecting to users using both wired and wireless infrastructure. As a result, it exposes users and network resources to many potential attacks. A special concern will be dedicated to the security of the Internet of Things (IoT), since it will include every object or device with networking capabilities. Objects can include simple home sensors, medical devices, cars, airplanes and even nuclear reactors and other things, which can poses risks to human life.

Traditional security mechanisms like Firewalling, Intrusion Detection and Prevention Systems are deployed at the Internet edge. Those mechanisms are used to protect the network from external attacks. Such mechanisms are no longer enough to secure the next generation Internet. The borderless architecture of the IoT raises additional concerns over network access control and software verification.

In Ad-hoc network for IoT does not exist simple solution to control the exchanges between each node. For instance, if one thing is corrupted by a virus, this treat can propagate itself in the network without any control. Moreover, anyone can connect his things on the network. Details of network access control implementation based for IoT devices can be found in [19].

The new networking paradigm, the Software Defined Networking (SDN), offers many opportunities to protect the network in a more efficient and flexible way [1]. In SDN architectures, network devices do not make forwarding decisions. Instead of that, network devices communicate with a special node, called the SDN controller, in order to provide them with the appropriate forwarding decisions. To communicate with the Controller, the network devices can use different protocols. The most used protocol for the communication between the SDN controller and the network devices is the OpenFlow [3]. OpenFlow defines control messages that enable the SDN controller to establish a secure connection with the network devices, read their current state, and install forwarding instructions. Furthermore, OpenFlow provides granular and flexible traffic management, using twelve fields in the packets header to match the network traffic. The flow rules (forwarding decisions) can be dynamically modified in order to adapt to different network changes. Moreover, OpenFlow was initially developed to enable researches and run experimental protocol in the production networks. Afterwards, it was widely deployed in campus networks, data centers, etc.

In this article, we present a security model for the IoT based on the SDN architectures. Firstly, the proposed security model was designed to establish and secure both wired and wireless network infrastructure. Secondly, we extended the proposed architecture in order to include Ad-Hoc networks and network object things such as: sensors, tablets, smart phone, etc. Our main contributions are as follows:

- To the best of our knowledge, this is the first effort that uses the SDN architecture to tackle security issues in the IoT.
- Inspired by existing Network Access Control and security techniques, we design a secure SDN-based architecture for the IoT.
- Based on a Grid of Security paradigm [9], we enhance security policies exchange and deployment between SDN control domains.

Our security model is discussed later in this article, and we conclude with the outline of our vision for the SDN based security on solutions for the IoT.

978-1-4799-1775-4/15 $31.00 © 2015 IEEE
DOI 10.1109/WAINA.2015.110

II. The Software Defined Network

Software-Defined Networking (SDN) has emerged as a new paradigm for enabling innovation in networking research and development. The control and data planes are decoupled, network intelligence and state are logically centralized. A new device called controller connects to the switch through a secured OpenFlow channel and manages this switch via the OpenFlow protocol.

The controller can add, update, and delete flow entries, both reactively in response to packets and proactively with predefined rules. Moreover, SDN enables fast reaction to security threats, granular traffic filtering, and dynamic security policies deployment.

The SDN architecture provides a programmatic interface inside the controller. SDN allows that the network control operations:

- run on top of one or multiple server platforms with higher performances,
- use vendor independent hardware and an open operating system,
- are able to communicate with other operating systems or control platforms using standard protocols.

In terms of security, SDN architecture extends the security perimeter to the network access end point devices (access switches, wireless access points, etc.), by pushing security policies to those devices [2].

After establishing connection with the OpenFlow switches, the SDN controller builds a global network view based on the information received via the OpenFlow protocol. Furthermore, the SDN controller can:

- perform network discovery, using the Link Layer Discovery Protocol (LLDP),
- collect statistics about network traffic using a special field in the flow rules earlier installed by the controller.

A. Simple Point of Failure

Previous works, [10], [11], [12], [13], the authors have developed framework and security applications for SDN. The main issue of their works is the presence of a single point of failure which is the controller. Using only one controller, a Denial of Service (DoS) can occur. Furthermore, security threats are another drawback. If an attacker compromises the SDN controller, then he has full control over the network. So, with one controller a potential risk exists to the entire network. In addition hardware and software failures also pose risks to a single controller system.

A solution using multiple controllers [14] can increase trustworthiness and fault tolerance. When a controller fails, another SDN controller can take control to avoid system failures.

B. Synchronized Multiple Controllers

In an SDN network, switches communicate with their controller via standard TLS or TCP connections. The Open Daylight Controller [14] supports a Cluster-based High Availability model. The controller has a global view of the network, and it can easily ensure that the network is in optimal condition. There is increased network performance with multiple controllers, because each controller has a partial view of the network, and the controllers have to collaborate and exchange informations with each other. The interaction between the Controller(s) and the Open-Flow enabled switches is essentially to have one Openflow switch multi-homed to multiple controllers. If one of the controllers goes down, another is ready to control the switch.

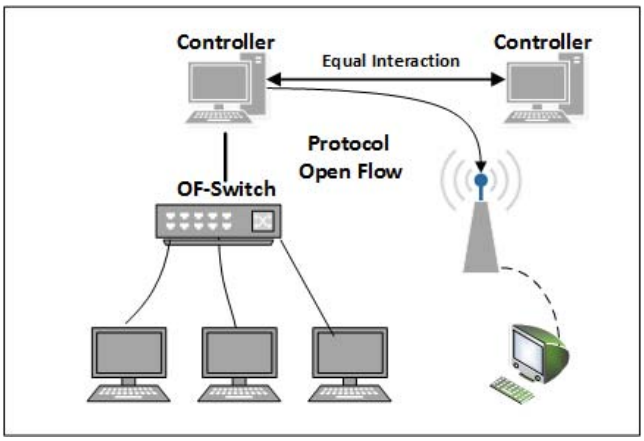

Fig. 1. Equal interaction SDN Networks.

Since version 1.2, Openflow specify two modes of operation when multiple controllers exist in the network:

- Equal interaction (Fig. 1): in this case all controllers have read/write access to the switch, which means they have to synchronize in order not to step on each other's feet.
- Master/Slave interaction: in this case there will be one master and multiple slaves (there could be still multiple equals as well)

III. SDN Based Ad-Hoc Architecture

The equal interaction is the default comportment of a controller. It has full access to the switch and all controllers have the same rules. Based on this assumption, we propose Multiple SDN Controller architecture for Ad-Hoc Networks.

A. SDN architecture for Ad-Hoc Network

In this section, we present an SDN based architecture for Ad-Hoc Network. We propose that a node in Ad-Hoc networks can be viewed (Fig. 2) as a combination of

- legacy interfaces : the physical layer,
- programmable layer : SDN compatible virtual switch and an SDN controller
- operating system and their applications : the OS layer

In this model, all legacy interfaces are connected to a virtual switch, and this switch is controlled by an SDN controller, integrated on the node. Since all controllers of each node operate in equal interaction, they will have no need to be concerned about nodes liability for the misbehaving users connecting through them, as in [8]. Ad-Hoc users will connect with other

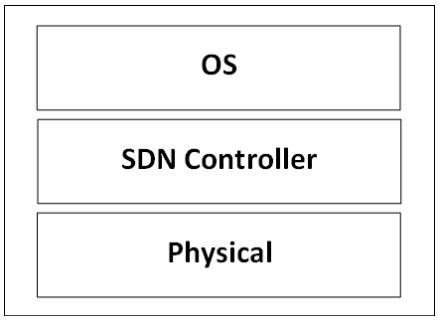

Fig. 2. A node in Ad-Hoc network.

nodes through their embedded SDN-compatible switch. At the same time, the SDN controller, in equal interaction, can enhance the security and connectivity between the nodes.

One of the advantages of this new SDN based Ad-Hoc network architecture is its compatibility with SDN legacy network. Since each node in the Ad-Hoc network has an embedded SDN-compatible switch and an SDN controller, we can interconnect the Ad-Hoc network to the legacy network to construct an Extended SDN domain (Fig. 3). Moreover, as all controllers of the extended SDN domain are in equal interaction, all rules will be synchronized.

In a most recent work like [7], the SDN domain is limited to the network with infrastructure. In this configuration, Ad-Hoc users have to connect through other nodes (Network gateway) directly connected to the SDN domain. In our proposed architecture, the SDN domain is extended in order to include all Ad-Hoc devices. Our proposed solution consists of deploying an OpenFlow software switch, such as Open vSwitch [3] in each Ad-Hoc node. This configuration enables Ad-Hoc nodes to connect to the network as part of the SDN domain, so we can apply the same security rules as for the SDN domain users. As show in Figure 3, the proposed architecture supports networks with or without infrastructure.

B. Distributed Ad-Hoc Control Plane

As each Ad-Hoc node has its own SDN controller, the SDN control plane has to manage the evolution of each SDN virtual switch on each Ad-Hoc device. When a new Ad-Hoc device connects itself or leaves the network, we can have many exchanged messages in order to synchronize all the rules. In order to ensure scalability and fault tolerance, a distributed SDN architecture is preferred, with multiple controllers as in [8]. To ensure that, we dynamically add new controllers to the Ad-Hoc network area and authorize special nodes to run control operation, as illustrated in Figure 4. The new controllers will share the same network global view. However, their functions and SDN management domain will be limited to a small Ad-Hoc area. Furthermore those controllers will be responsible for monitoring the behavior of the software switches, since they are deployed at the user side.

Our proposed distributed network access control architecture enables faster response to network changes. Moreover, it

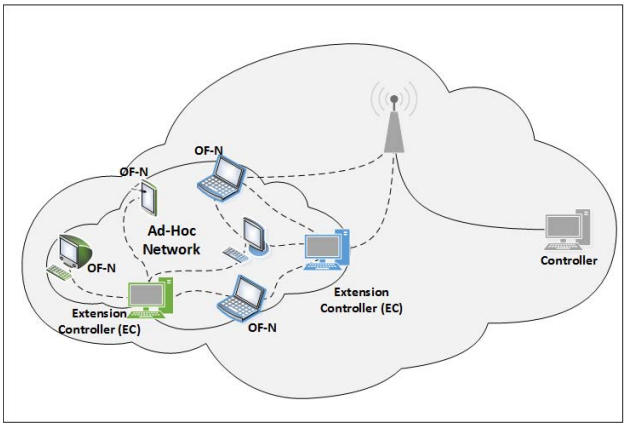

Fig. 4. Distributed Ad-Hoc Control Plane.

reacts to attacks occurring in the SDN extended domain, while sharing the traffic load management with the root controller. As mentioned earlier, control functions of Ad-Hoc controllers will be limited and adapted to the available resources of the hosting Ad-Hoc device. We intend to extend the SDN domain even more, to include smart object like: tablets, smart phones, mobile vehicles, etc. by developing a framework that integrates OpenFlow software switches into those devices.

IV. SDN based architecture for IoT

The traditional network protocols and equipment are not designed to support high levels of scalability, high amounts of traffic and mobility. Some authors [15] have proposed architecture for IoT but to the best of our knowledge. Besides, there exits some papers [20], [21] of software-defined approach for the IoT environment, but it don't propose security mechanism for Ad-hoc network.

A. SDN Domain

In IoT or in sensor networks, each device cannot have an embedded SDN compatible switch and an SDN controller as we have proposed in the previous section. But, we can assume that each device, with low resource can be associated to one neighbor node which has the SDN capability. In a heterogeneous network as in Figure 3 we have two types of nodes in a domain. If the node has enough of resources, we called it an OF node and if not, we called it a sensor or a smart object. Each domain has its SDN controller which control all traffic in its domain.

B. SDN Domain interconnection

In the proposed architecture with multiple SDN domains, we assume that in each domain, we have one SDN controller or multiple SDN controllers. These controllers manage only the device in its domain. A domain represents an enterprise network or a datacenter.

An SDN-based architecture for the IoT requires heterogeneous interconnection with larges number of SDN domains. In order to achieve such large scale interconnection, we

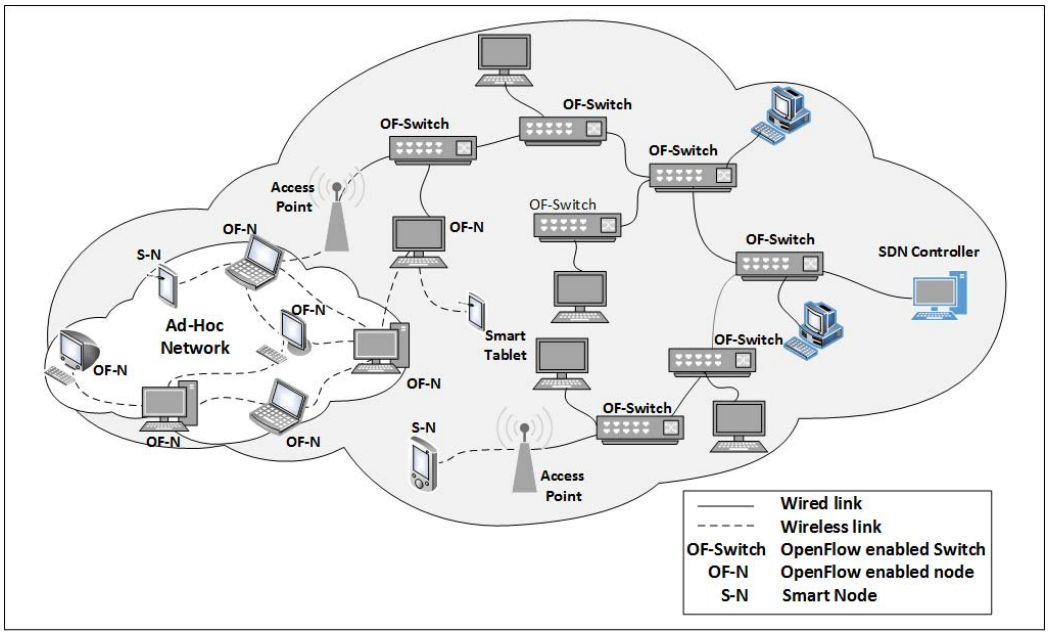

Fig. 3. Extended SDN Domain.

introduce a new type of controller in each domain : the root controller, that we can also called a Border controller. Some authors [16], [17] propose hierarchical architecture for SDN, to optimize and distribute control functions. We propose not to distribute control functions on multiple controllers but to distribute routing functions and security rules on each border controller. Moreover these controllers are responsible for establishing connections and exchanging information with other SDN border controllers.

The development of this architecture is based on the perspective of equal interaction between controllers, using the existing security mechanisms. Each SDN domain has its own security policies and management strategy. To solve potential problems raised by the heterogeneity of the security policies respective to the interconnected SDN domains, we use the Grid of Security concept proposed by Flauzac et al. in [18]. The Grid of Security is a middleware for decentralized enforcement of the network security.

V. Distributed SDN Security Solution

The controller can not only manage the network, but also monitors and efficiently secures the network against outside and inside attacks.

Many works have studied network security using the SDN architecture, either by implementing firewalls, IPS and IDS modules on top of the SDN controller [4], [5], [6], or by installing security policies into OpenFlow switches [2]. The emergence of the next generation Internet architecture, requires even higher security levels, such as authenticating network devices, users and objects connecting to users using both wired and wireless technologies. Furthermore, we need to monitor the behavior of both the users and the objects, establish trust boundaries, and use accounting methods along with software verification. However, existing security mechanisms [2], [4], [5], [6] do not provide these security levels to meet the security needs of next generation Internet architecture.

Inspired by existing Network Access Control and security techniques [8], we design an extended secure SDN-based architecture for the IoT. To explain our architecture, we first present a simple solution in which a controller manages the security of one SDN domain. Second, we can extend this first solution to include multiple controllers with regard to the available resources on each control platform. In addition, we extend the distributed control architecture by interconnecting all SDN domains via border controllers, which leads us close to a secure model for the IoT.

Traditional Ad-Hoc architecture does not provide network access control or global traffic monitoring, due to the absence of the network infrastructure. The architecture proposed in this article overcomes those security limitations and enables dynamic network configuration and security policies deployment.

Our purpose is to achieve maximum synchronization of SDN Controllers in a security perimeter enabling a granular control over network access and continuous monitoring of network endpoints.

In order to secure network access and network resources, the SDN controllers begin by authenticating the network devices. Once the OpenFlow secure connection between the switch and the controller is established, the controller blocks switch ports directly connected to the users. After that, the controller authorizes only user's authentication traffic. Once the user is authenticated, and based on the authorization level of the user, the controller will push the appropriate flow entries to the

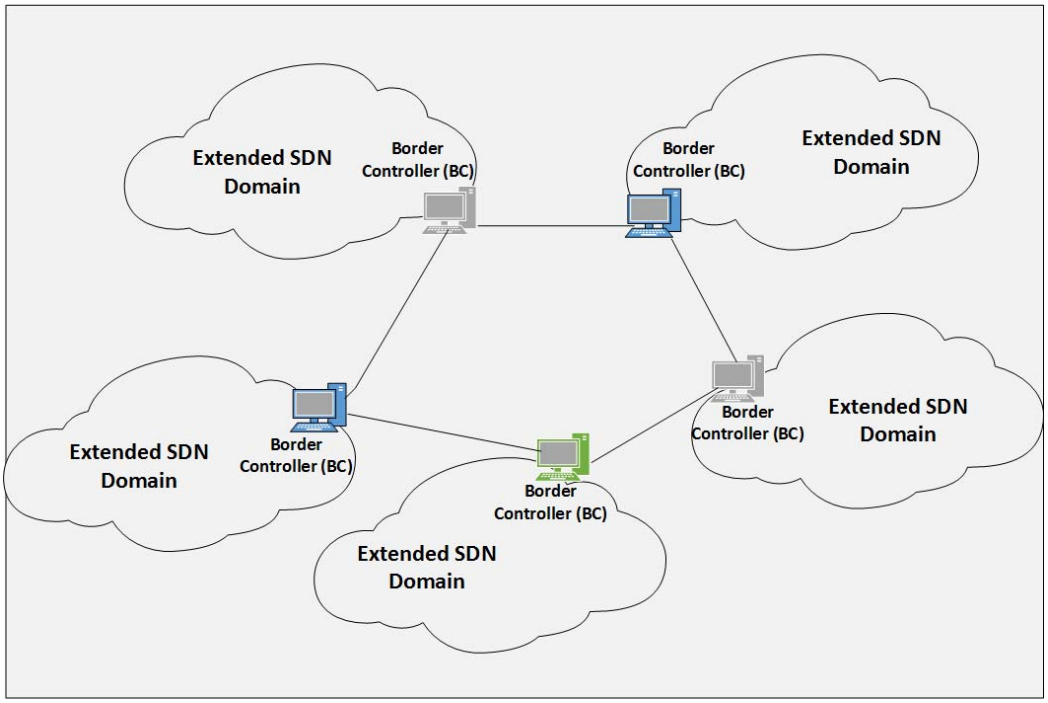

Fig. 5. SDN Domain Interconnection.

software or the hardware access switch. In IoT, we extend this authentication process to devices. Each device has to associate itself with an OpenFlow enable node, each of which is connected to one controller in their domain.

The whole concept of the grid of security network is to extend the SDN domain concept to multiple domains (Fig. 6), and each controller of each domain exchanges their security rules. There are SDN controllers which behave as security guards on the edge of the extended SDN Domain to ensure the network safety. Safety connections between domains could be provisioned and only added to SDN Controllers. Only recognized traffic could be accepted. The controllers know policies in their domain but they don't know policies of the other domains. So, when a node wants to communicate with another node of another domain, the flow has to be forward to the Security Controller, also called the Border Controller. The security controller asks each neighbor controller if it knows the destination of the information (Fig. 6).

There is further potential to exploit this architecture; with a security controller in every domain, it can prevent users opening unauthorized new services. Only services authorized by controller can be used for endpoint devices. In order to prevent one user initiating unauthorized communication with another user, when one wants to open one a communication port, he must make a request to the SDN controller. For instance, assuming that one user or device wants to open a web service, the first task is to request to its controller to determinate if a web service exists and can be opened on the network. Then each border controller asks each controller of each domain. If a such service exists on a device, then all messages will be forward to this device.

Furthermore, the Extended SDN controllers periodically monitor and check the flow table entries of the software switches because they are deployed on the user side. In the proposed architecture, we deployed software switches on the users side to enhance the forwarding capabilities of Ad-Hoc devices. Moreover, the deployed software switches allow the SDN controller to apply and enforce the security policies inside the Ad-Hoc area.

VI. Conclusion

In this article we proposed a new SDN based network architectures with distributed controllers. Moreover, our solution can be used in the context of Ad-Hoc network and IoT.

First, we presented a new architecture with multiple SDN controllers in equal interaction. Second, we propose an architecture which is scalable with multiple SDN domains. In each domain we can have networks with or without infrastructure and each controller is responsible only for its domain. The communications between domains is made with special controllers called Border Controllers. The Border Controllers have to work in a new distributed interaction in order to guarantee the independence of each domain in case of failure. We adopt an architecture to guarantee the security of the entire network with the concept of grid of security embedded in each controller to prevent attacks. We work to build this architecture and test it in a real environments.

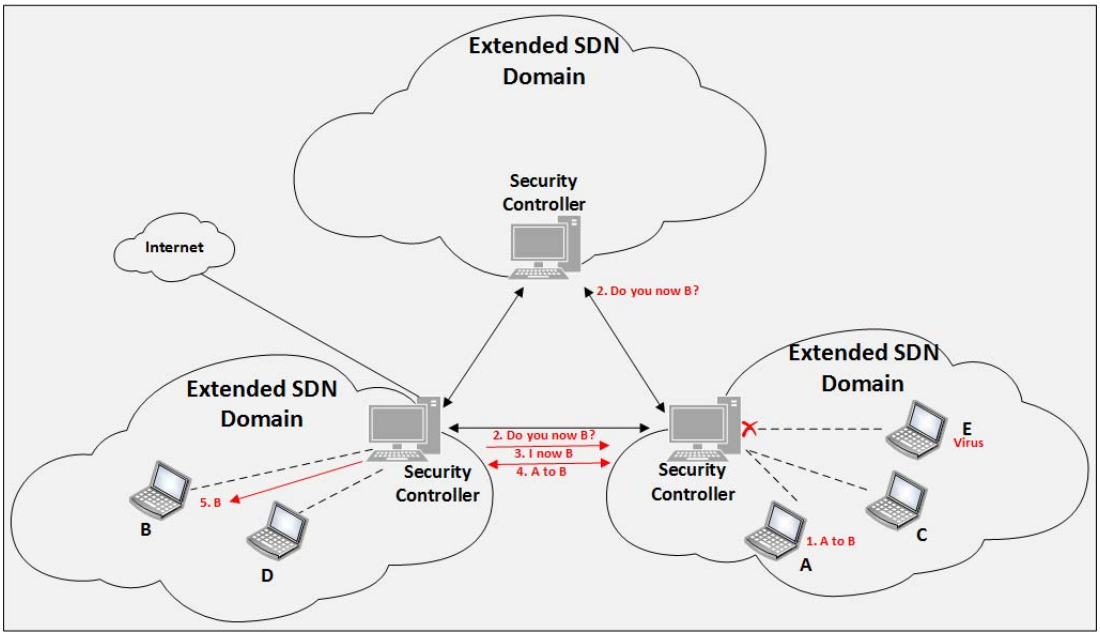

Fig. 6. Grid of Security in SDN Domain.

REFERENCES

[1] A. Tootoonchianand Y. Ganjali, *Hyperflow: A distributed control plane for openflow*, in Proceedings of the Internet Network Management Conference on Research on Enterprise Networking. pp. 3-3, 2010.

[2] B. Nunes, M. Santos, B. de Oliveira, C. Margi, K. Obraczka, and T. Turletti, *Software defined networking enabled capacity sharing in user-centric network*, IEEE Communications Magazine. vol. 52, no. 9, pp. 28-36, 2014.

[3] S. Scott-Hayward, G. OCallaghan, and S. Sezer, *SSDN security: A survey*, in Proceedings of the IEEE SDN for Future Networks and Services. pp. 1-7, 2013.

[4] S. Son, S. Shin, V. Yegneswaran, P. Porras, and G.Gu, *Model checking invariant security properties in openflow*, in Proceedings of the IEEE International Conference on Communications. pp. 1974-1979, 2013.

[5] H. Hu, W. Han, G.-J. Ahn, and Z. Zhao, *Flowguard: Building robust firewalls for software-defined networks*, in Proceedings of the Third Workshop on Hot Topics in Software Defined Networking. pp. 97–102, 2014.

[6] R. Jin and B. Wang, *Malware detection for mobile devices using software-defined networking*, in Proceedings of the Workshop of Research and Educational. pp. 81-88, 2013.

[7] R. Skowyra, S. Bahargam, and A. Bestavros, *Software-defined ids for securing embedded mobile devices*, in Proceedings of the High Performance Extreme Computing Conference, pp. 1-7, 2013.

[8] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker and J. Turner, *Openflow: Enabling innovation in campus networks*, SIGCOMM Computer Communication. vol. 38, no. 2, pp. 69-74, 2008.

[9] A. De Rubertis, L. Mainetti, V. Mighali, L. Patrono, I. Sergi, M. Stefanizzi, and S. Pascali, *Performance evaluation of end-to-end security protocols in an internet of things*, in Proceedings of the 21st International Conference on Software, Telecommunications and Computer Networks. pp. 1-6, 2013.

[10] R. Braga, E. Mota, and A. Passito, *Lightweight DDoS flooding attack detection using NOXIOpenFlow*, in Local Computer Networks (LCN), 2010 IEEE 35th Conference on. IEEE, 2010, pp. 408–415.

[11] H. Jafarian, E. Al-Shaer, and Q. Duan, *Open flow random host mutation: transparent moving target defense using software defined networking*, in Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012, pp. 127–132.

[12] S. Shin, P. Porras, V. Yegneswaran, M. Fong, G. Gu, and M. Tyson, *FRESCO: Modular composable security services for software-defined networks*, in Proceedings of Network and Distributed Security Symposium, 2013.

[13] S. Shin and G. Gu, "CloudWatcher, *Network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?)*, in Network Protocols (lCNP), 2012 20th IEEE International Conference on. IEEE, 2012, pp. 1–6.

[14] Network Functions Virtualization (NFV), *OpenDaylight*, http://www.opendaylight.org/, [Online; accessed 09- Octubre-2014].

[15] P. Diogo and L.P. Reis and N. Vasco Lopes, *Internet of Things: A system's architecture proposal*, in 9th Iberian Conference on Information Systems and Technologies (CISTI), 2014 , pp.1,6, 18-21 June 2014

[16] Shuai Gao; Yujing Zeng; Hongbin Luo; Hongke Zhang, *Scalable area-based hierarchical control plane for software defined information centric networking*, in 23rd International Conference on Computer Communication and Networks (ICCCN), 2014, pp.1,7, 4-7 Aug. 2014

[17] Xu Li; Djukic, P.; Hang Zhang, *Zoning for hierarchical network optimization in software defined networks*, in IEEE Network Operations and Management Symposium (NOMS), 2014, pp.1,8, 5-9 May 2014

[18] O. Flauzac, F. Nolot and C. Rabat; and L. Steffenel, *Grid of security: A new approach of the network security*, in Proceedings Third International Conference on Network and System Security. pp. 67–72, 2009.

[19] P. Sanchez; R. Lopez; A. Skarmeta, *PANATIKI: A Network Access Control Implementation Based on PANA for IoT Devices*. Sensors 2013, 13, 14888-14917.

[20] Zhijing Qin; Denker, G.; Giannelli, C.; Bellavista, P.; Venkatasubramanian, N., *Software Defined Networking architecture for the Internet-of-Things*. in Network Operations and Management Symposium (NOMS), 2014 IEEE , vol., no., pp.1,9, 5-9 May 2014.

[21] P. Martinez, A. Skarmeta, *Empowering the Internet of Things with Software Defined Networking*. in FP7 European research project on the future Internet of Things.