A Novel Distributed SDN-Secured Architecture for the IoT

Carlos GONZALEZ, Olivier FLAUZAC and Florent NOLOT
University of Reims Champagne-Ardenne, CReSTIC, Reims-France
Email: carlos.gonzalez-santamaria@etudiant.univ-reims.fr,
olivier.flauzac,florent.nolot@univ-reims.fr

Antonio JARA
University of Applied Sciences Western Switzerland
Sierre (Switzerland)
Email: jara@ieee.org

***Abstract*—Due to their rapid evolution, mobile devices demand for more dynamic and flexible networking services. A major challenges of future mobile networks is the increased mobile traffic. With the recent upcoming technologies of network programmability like Software-Defined Network (SDN), it may be integrated to create a new communication platform for Internet of Things (IoT). In this work, we present how to determine the effectiveness of an approach to build a new secured network architecture based on SDN and clusters. Our proposed scheme is a starting point for some experiments providing perspective over SDN deployment in a cluster environment. With this aim in mind, we suggest a routing protocol that manages routing tasks over Cluster-SDN. By using network virtualization and OpenFlow technologies to generate virtual nodes, we simulate a prototype system controlled by SDN. With our testbed, we are able to manage 500 things. We can analyze every OpenFlow messages and we have discovered that with a particular flow, the things can exchange information unlike the routing principle.**

***Index Terms*—SDN, Openflow, Cluster, IoT, Ad-hoc.**

I. Introduction

Nowadays, network technology allows mobile devices and hence users to be always connected at any time in any place. With the explosion of connected devices, services and access technologies, mobile networks are constantly increasing coverage and the needs to support a wide range of high data demand has become an issue to be solved. Currently, there are more devices connected to the internet than humans in the world [1], and these generate an enormous amount of traffic (i.e., voice, video, data, etc.). All of these factors increase considerably the cost pressure on mobile network operators, due to the emerging mobile devices and application [2]. A potential solution to reduce Capital Expenditure (CAPEX) and Operational Expenditures(OPEX) costs [3] is to use network sharing, in which mobile network operators share the same network infrastructure.

One of the greatest challenges concerns the security of devices, since it will include every objects or devices able to connect to wireless or wired networks. Devices such as in the military, industries, simple home sensors, medical devices, cars, airplanes and other devices, are some examples wherein security threats can pose risks to human life.

Due to absence of explicit quality-of-service (QoS) control mechanisms for mobile devices, quality-of-experience (QoE) can be guaranteed only locating servers closely to user terminals. In this context, with the latest Internet evolution, billions of devices will be connected to other devices. However, many of these devices will build many ad-hoc networks in which there are no security system to control packets exchanged, no simple solution exist to control the packet exchanges between nodes on the network. If one object is infected by a malicious applications, it may compromise other connected devices.

Over the past few years, the research community has focused on the new networking paradigm, the Software Defined Networking (SDN). The SDN offers many opportunities to protect the network in a more efficient and flexible way [4]. 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 provides them with the appropriate forwarding decisions. To communicate with the Controller, the network devices can use different protocols. The most commonly used protocol for communication between the SDN controller and network devices is OpenFlow [5]. OpenFlow defines control messages which enable the SDN controller to establish a secure connection with the network devices, read their current state, and install forwarding instructions. Furthermore, OpenFlow enables fast reaction to security threats, granular traffic filtering, and dynamic security policy deployment. The flow rules (forwarding decisions) can be dynamically modified in order to adapt to different network changes. Moreover, OpenFlow was initially developed to run experimental protocols in production networks. Afterwards, it was widely deployed in campus networks, data centers, etc.

Current IoT devices are vulnerable to a range of attacks, one of the most important is based upon the plain text meta-data. The benefits of employing SDN techniques in IoT environment can make the IoT much simpler to protect, manage and reconfigure. We provide an overview of how the IoT devices may be handle in ad-hoc networks and cluster scenarios by the SDN paradigm.

In this article, we present a model to control and secure information exchanges for the ad-hoc devices, based on the SDN architectures. Firstly, the proposed model was designed to establish and secure both ad-hoc and mobile networks, in order to include objects such as: sensors, tablets, smart phone, etc. Secondly, we extended the proposed architecture, and explain how flows can be routed between controllers. We demonstrate the implementation of our prototype system and evaluation results. The major contributions of this paper are:

2325-2944/16 $31.00 © 2016 IEEE
DOI 10.1109/DCOSS.2016.22

- A novel exploration of the SDN architecture to interconnect ad-hoc and mobile users applications.
- An introduction to the concept of the SDNCH cluster head to distribute routing functions and security rules to each edge controller.
- An introduction to an implementation model with an Opendaylight controller to manage and monitor traffic from the end-users in ad-hoc networks.

II. SDN Cluster architecture for Ad-Hoc Network

The Open Daylight Controller [25] uses a Model Driven-Service Abstraction Layer (MD-SAL) clustering solution, providing fault-tolerant decentralized peer-to-peer controllers in a cluster deployment. There is an increase of network performance with multiple controllers, because each controller has a partial view of the network and they have to collaborate and exchange information with each other. This cluster model provides a high-level of fault tolerance: active/active or active/passive deployment modes. The possible operation modes of the controller in cluster deployment is explained below:

- In active/active mode, should the configured primary fail, should there be partial network partition of the primary controller's links or should there be full network partition (if monitoring is enabled), the passive secondary will be promoted to the active primary controller role as part of the failover. During a full network partition, both segment of the network may be independently managed.
- In active/passive mode, in the event of the configured primary's failure or partial network partition or failure of the primary controller's links, the passive secondary will be promoted to the active primary controller role as part of the failover. In the event of a full network partition (if monitoring is enabled) the passive secondary will not be promoted to the primary role but will instead suspend core controller functionality. The switches on the secondary's segment of the network will go unmanaged.

Having multiple controllers [25] provides trustworthiness and fault tolerance. If one of the controllers goes down, another SDN controller can take control to avoid system failure. Based on the approach of active/active, we suggest multiple SDN controller architecture for Ad-Hoc Networks. An Ad-Hoc SDN-based architecture involves:

- The legacy interfaces: the physical layer;
- The programmable layer: SDN-compatible virtual switch and an SDN controller operating systems;
- Their applications: the OS layer.

The proposed 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 active/active mode, they will have no need to be concerned about nodes liability for misbehaving users connecting through them [24]. Ad-Hoc users will connect with other nodes through their embedded SDN-compatible switch. At the same time, the SDN controller, in active/active mode, can enhance the security and connectivity between the nodes.

Many solutions have been proposed by different authors [10], [11], [12] in the SDN and Wireless Sensors Network (WSN) field. To the best of our knowledge, there is no work about WSN in the SDN context with cluster architecture. In this paper we propose clustered software defined sensor networks. Clustering consists in organizing the network into groups of nodes, in a hierarchical structure. Each cluster is managed by a cluster head. To develop this architecture we place the SDN controller in the cluster head. Different clustering solutions have been proposed in the literature. Some interesting solutions introduce building 1-hop clusters [14], [15], [16], [17]. In those solutions, each node is at a most a distance of 1 from the cluster head, and the maximum diameter of each cluster is 2. Other solutions build k-hop clusters [18], [19], [22]. In k-hop cluster solutions, each node can be located at a distance at most of k from the cluster head and the maximum diameter of clusters is 2k.

III. SDN for mobile networks

In this section, we introduce the concept of SDCSN (Software Defined Clustered Sensor Networks) and explain the different technologies for their realization. In traditional network architectures, the control and data plane of network devices are compactly coupled. The control plane provides functions to build a forwarding/routing decisions to send frames data link layer (L2) and packets network Layer (L3). The data plane decides what to do with frame or packets, by carrying out the commands of the control plane through forwarding tables, routing tables, ARP tables, amongst others. With this principle, the network nodes must compute the path to the destination after exchanging routing information. As a result, any process requires many successive operations compared to the SDN architecture where the routing decisions are made only by the controller. Network devices will only handle packets based on the flow table entries received from the controller via the OpenFlow protocol. With this in view, the packets can be matched against fourteen required header match fields which provide a higher level of granularity than the traditional forwarding technique.

Today, the network protocols and equipment are not designed to support high levels of scalability or high amounts of traffic and mobility. Dong et al. [26] have proposed a rule management in SDN-enabled mobile access networks. The authors set the ternary content addressable memory (TCAM), to add a caching algorithm which manages the forwarding rules cache. This algorithm for SDN-enabled mobile access networks take into consideration user mobility when an user moves from one base station to another. The proposal is to have an increase caching algorithm for hit ratio than traditional algorithms. Besides, there exists some papers [29], [30] of software-defined approach for mobile networks environment, enabling mobility management via SDN mechanisms. These papers had the focus of determining the integration of SDN and mobile nodes. Our system proposes not only to have an SDN mobile nodes integration, but also to have a secure cluster SDN-based architecture for mobile and Ad-hoc networks.

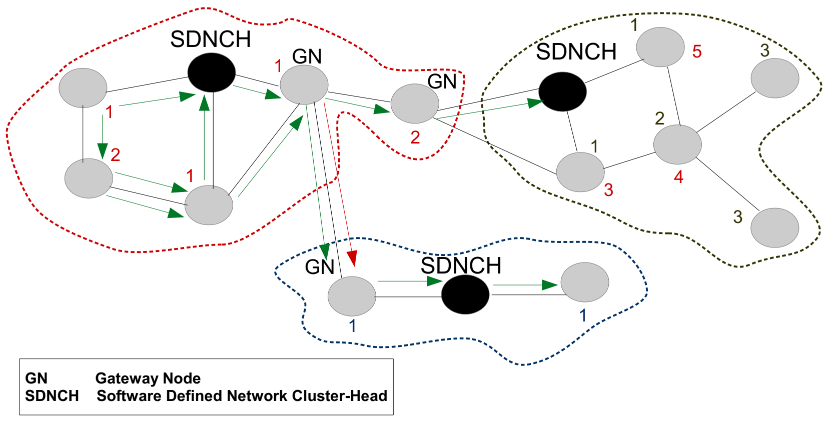

Fig. 1. Data Communication Software-defined wireless sensor networks

If our approach is for sensor networks, we assume that each device, with low resources, can be associated to one neighbor node which has the SDN capability.

The WSN may contain hundreds or thousands of sensor nodes. Normally, a large network cannot operate efficiently without some organized structure. For this reason, we suggest to cluster the network and assume that each cluster head is a controller. Each node can be in one of the following states [7], [9], [22]: Simple Node (SN), Gateway Node (GN) or Cluster Head (CH). In our approach, Cluster Heads (CH) in SDCSN architecture are called SDN Cluster Heads (SDNCH). Each cluster is called an SDN Domain.

In each SDN domain, the SDNCH [10] is in charge of managing the operation of the sensor nodes (Fig. 1). With this clustering approach the collected information about the environment is processed in the domain by nodes and will be routed to the SDNCH. Moreover, the controller is the most powerful node on the cluster. By using the active/active interaction between SDNCH, it will have a full access to the switch and the same flow rules. Based on this approach, we can set configuration parameters, store data and aggregate the collected information in the domain. We can also send information to the sink or to some other SDNCH. If a SDNCH is disabled, the entire domain becomes inaccessible temporarily and based on self-stabilizing clustering for WSNs [22] a new SDNCH can be selected. Each sensor node exchanges information with its neighbors or 2-hop neighbors. Under the assumption of graph connectivity, information generated at one sensor can reach the SDNCH by routing it through the network. The gateway is engaged in aggregating and transmitting the data from entire sensor node domains to the other domains. Thus, when a gateway goes down, the communication between domains will be disconnected and the associated sensors to SDNCH are isolated.

The SDNCH plays the role of coordinator in every domain. It makes a decision, modifies the flow tables of the corresponding switches by sending messages to each one of them. The controller uses a multipath-routing between SDNCHs. This process allows all of the devices in the domain to address the network load to be balanced among several alternate paths. Consequently the chances of congestion are reduced. Every SDNCH acts as a temporary base station within its domain, and it is capable of communicating with other SDNCHs. A Domain is therefore composed of a SDNCH, gateways and sensor nodes.

- The SDN Cluster Head (SDNCH) is the coordinator of the domain.
- Gateway is a bridge node between sensor nodes and SDNCH.
- Sensor Nodes (SN) are groups of nodes in a domain, together with their gateway nodes.

In the base station architecture proposed by Gante et al. [11], an SDN controller is combined with WSN and the controller needs to know the topology of the entire network. SDN has a higher potential to develop forwarding decisions of SN based on the rules set generated by the controller, thus permitting a better cooperation among SDNCH and SNs [6], [11], [12], [13].

Certainly, QoS could be an efficient way for sensor nodes to function in this architecture using the concept of meters tables. This consists of meter entries, defining per-flow meters [21]. The per-flow meters permit Openflow to implement different QoS operations, such as rate-limiting, and can be combined with per-port queues to implement complex QoS frameworks (ex. DiffServ). Moreover, SDN controllers [11] can reduce energy consumption by different sensor nodes, making the best routing decision and injecting these in the nodes flow tables. With the network management controlled by the SDN, the routing decisions and policies have low convergence time in comparison with routing protocols.

To deploy this architecture, SDNCH not only has to manage the domain network. It also has to monitor and efficiently secure the domain to prevent outside and inside attacks. The emergence of the next generation Internet requires high level

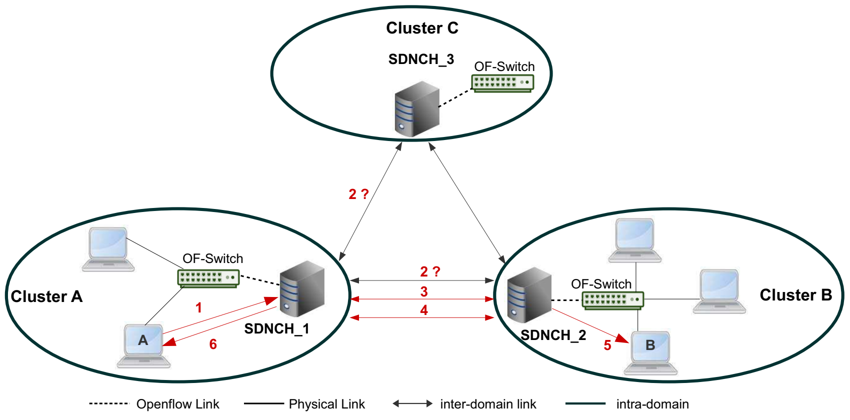

Fig. 2. Distributed Routing Cluster for SDN

security. Many works have studied network security using the SDN controller or installing security policies into OpenFlow switches, either by implementing firewalls, IPS, or IDS [8], [20], [23]. Our purpose is to achieve maximum synchronization of SDNCH in a security perimeter enabling granular control over network access and continuous monitoring of SNs. Moreover, this approach acts as a security guard on the edge of the domain to ensure the domain safety. For example, if the service is forest fire detection, SNs can identify threats by smoke, ultraviolet and temperature sensors and achieve optimal transmission to the SDNCH.

Therefore, the proposed cluster can not solve problems with routing processes in distributed SDN-architecture. But by combining routing protocols and SDN, a new effective controller-routing method is proposed in the next section.

IV. Routing Protocol for Distributed Cluster SDN

For the clustering architecture, we propose an SDN controller in each cluster. This controller manages and controls all traffic of nodes connected only in its intra-domain. In this environment, the controllers communicate with others via an inter-domain link. Previous work [27], [28] 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 on each SDNCH.

The deployment of this architecture is based on the perspective of an Opendaylight MD-SAL Akka-based clustering solution. To select an appropriate path between nodes connected on the cluster, the process of routing flows has been indicated in (Fig. 2).

The main process flow is as follows:

1) Node A is joined to node B, node A sends a request to the SDNCH1;
2) SDNCH1 sends the same request to the neighbor controllers connected on the network;
3) The ones which know node B send a replies to SDNCH1, for this example SDNCH2;
4) The flow may now be installed on the SDNCH1 and node A can use the communication path via SDNCH1 for taking out of cluster the packet;
5) The routing information between SDNCH1 and SDNCH2 being set up on the inter-domain path;
6) The messages will be exchanged between both nodes.

We focus in the proposal of a new routing protocol for a distributed cluster SDN, which can be used to support SDN-based inter-domain collaboration. The goal is to allow an automated routing setup of inter-domain clusters. As a result, a node can interact with other nodes located in different clusters through an inter-cluster routing protocol.

V. Experiments

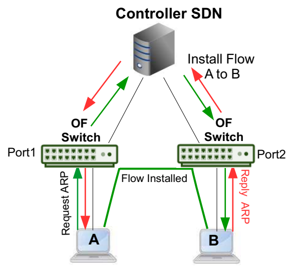

Fig. 3. Openflow ARP Request

We implemented an Opendaylight SDN controller [25] using OpenFlow 1.3 protocol. According to the OpenFlow protocol, an OpenFlow switch forwards a packet to the controller if it does or does not have a matching flow rule for that packet. When host A wants to send an IP packet over Ethernet to host B by IP address, it needs an ARP reply from B. When a new flow from host A arrives at an OpenFlow switch, the switch forwards the packet to the controller SDN. The controller plays the role of ARP proxy. If there is not

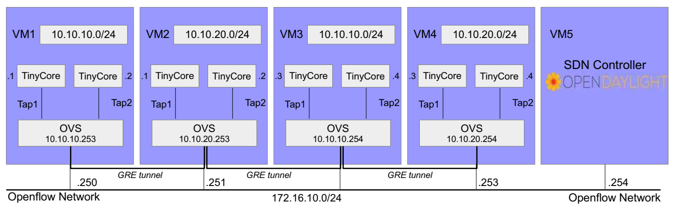

Fig. 4. Network design of the mobile devices based SDN testbed

match with the flow entry installed, the packet will be rejected. Otherwise, an ARP request is allowed to pass toward host B. Upon receiving the ARP reply, the routing in the controller will send the flow to the switch connected to the source host and the switch connected to the destination host. Once, the link is established, a shortest route will be set for the switch. In this way, the switch is in charge of routing the packets between Host A and Host B (Fig. 3).

The implemented testbed in this paper includes a virtualized testing environment combined with vmware vsphere, qemu system, Tinycore, software-based Open vSwitch (OVS) and Opendaylight controller. The experimental platform was developed based on SDN and that enabled users easily to design network topology via web applications. The framework of the testbed would then allocate resources to the experiment, which is able to create routes between nodes, monitor requests sent to the controller and also decide policies delivered to each device. Our approach on the testbed provides an environment for mobile and ad-hoc network deployment. It allows nodes to be connected with others via one SDN-compatible OVS switch and at the same time this switch is controlled by an SDN controller.

For the L2/L3 forwarding decisions, the SDN controller knows the topology and also the devices attached to the topology as well as their identities IP/MAC addresses. Then, programming the switches via an open API, the OpenFlow controller may manage and constantly the Openflow network update upon changes. In L2, the OVS acts as a regular layer-2 learning switch, it automatically creates a learning table based on the source MAC address of incoming frame and places it on an incoming port then either forwards the frame to the appropriate output port destination.

To build a route L3 the forwarding operations at flow or packet level are therefore processed by the controller pushing rules into hardware devices calling the ovs-ofctl add-flow command as well as ARP resolution between hosts and edge switches. If the switch receives a packet that does not match any entry in the flow table, the Openflow switch forwards the packet to the controller by an ARP request. When such request is received by the controller, it will be ignored because the controller does not know the NextHop IP address needed to handle all traffic forwarded to an another subnet. To configure the OVSs to learn the NextHop, an IP addresses is assigned to them. The packets routed will be sent out via the OVS local port to establish the connection between all nodes.

As shown in Fig. 4 we have built a testbed to experiment our presented protocol and it represents a cluster. We have virtualized OpenFlow compatible switchs (OpenVSwitch version 2.5.0) and small Linux (TinyCore with 48Mo of RAM). To evaluate the performance of Openflow, the experimental scenario deployed on the SDN testbed consists of Opendaylight Helium-SR4 and five virtual machines (VM) connected to the same network. The PC running the controller uses Ubuntu 14.04 operating system and was provided with 2 virtual CPUs and 16GB of RAM. Each VM had 8 virtual CPU and 16GB of RAM, using the Debian 8.3 image with OVS pre-installed. We use KVM for machine virtualization over the VMs, to run TinyCore Linux 3.16.6 with 48MB of RAM. From this testbed, we will be able to experiment our protocol with more than 500 devices, in a first attempt. Each used devices is a TinyCore Linux system.

VI. Conclusions

In this paper, we have proposed a cluster management system based on OpenFlow. The system manages communication between clusters by an SDN cluster head managed in an SDN controller. We have also built a prototype system to emulate a real Openflow network. Experiments show that our solution enables the control of physical/virtual networks, topology, traffic, flow table, and services in an SDN controlled network.

For future work, there are many directions that we intend to explore. Our next set of experiments is in progress, wherein we are working to set up a routing cluster protocol dynamic over SDN. This work shows promise for achieving one of the goals of software defined networking which is to provide better routing paths for devices in cluster scenarios. We wish to explore the distributed management of our SDN-based IoT testbed and also carry out performance evaluation in terms of transmission delay, processing workload, the management of massive amounts of topological information and the scalability of the control plane.

VII. Acknowledgements

This work was supported by SENACYT-Panama, Secretaria Nacional de Ciencia, Tecnologia e Innovacion.

References

[1] Evans, Dave. The Internet of Things How the Next Evolution of the Internet Is Changing Everything. CISCO white paper (2011).

[2] Liotou, E.; Tseliou, G.; Samdanis, K.; Tsolkas, D.; Adelantado, F.; Verikoukis, C., *Multi-tenancy for Virtualized Network Functions*, in Quality of Multimedia Experience (QoMEX), 2015 Seventh International Workshop on. pp. 1-2, May 2015.

[3] Medhat, A.M.; Carella, G.; Mwangama, J.; Ventura, N., *An SDN QoE-service for dynamically enhancing the performance of OTT applications*, in Network Softwarization (NetSoft). pp. 1-6, April 2015.

[4] 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.

[5] 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.

[6] T. Luo, H. Tan, T.Q.S. Quek, *Sensor OpenFlow: Enabling Software-Defined Wireless Sensor Networks*, Communications Letters, IEEE, 2012, 16, (11), pp.1896-1899

[7] P. Azad and V. Sharma, *Cluster Head Selection in Wireless Sensor Networks under Fuzzy Environment*, ISRN Sensor Networks, 2013

[8] 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.

[9] M. Ba, O. Flauzac, R. Makhloufi et al.: *A Novel Aggregation Approach Based on Cooperative Agents and Self-Stabilizing Clustering for WSNs*, The Seventh International Conference on Communication Theory, Reliability, and Quality of Service, CTRQ 2014

[10] D. Zeng, T. Miyazaki, Song Guo et al., *Evolution of Software-Defined Sensor Networks*, Mobile Ad-hoc and Sensor Networks (MSN), 2013 IEEE Ninth International Conference, 2013, pp 410-413

[11] A. De Gante, M. Aslan and A. Matrawy, *Smart wireless sensor network management based on software-defined networking*, Communications (QBSC), 2014 27th Biennial Symposium 2014, pp. 71-75

[12] S. Costanzo, L. Galluccio, G. Morabito et al., *Software Defined Wireless Networks: Unbridling SDNs*, Software Defined Networking (EWSDN), European Workshop 2012, pp.1-6

[13] Y. Fan , V. Gondi, J.O. Hallstrom et al.: *OpenFlow-based load balancing for wireless mesh infrastructure*, Consumer Communications and Networking Conference (CCNC), 2014 pp 444-449

[14] N. Mitton, A. Busson, E. Fleury, *Self-organization in large scale ad hoc networks*, in MED-HOC-NET, 2004

[15] O. Flauzac, B.S. Haggar, F. Nolot, *Self-stabilizing clustering algorithm for ad hoc networks*, ICWMC, 2009, pp 24-29

[16] C. Johnen and L. Nguyen, *Self-stabilizing construction of bounded size clusters*, ISPA, 2008, pp 4350

[17] C. Johnen and L. H. Nguyen, *Robust self-stabilizing weight-based clustering algorithm*, TCS, 2009, pp 581594

[18] N. Mitton, E. Fleury, I. Guerin Lassous et al., *Selfstabilization in self-organized multihop wireless networks*, in ICDCSW, 2005, pp. 909915

[19] E. Caron, A. K. Datta, B. Depardon et al., *A selfstabilizing k-clustering algorithm for weighted graphs*, JPDC, 2010, pp 11591173

[20] S. Son , S. Shin, V. Yegneswaran et al., *Model checkinginvariant security properties in openflow*, in Proceedings of the IEEE International Conference on Communications, 2013, pp 1974-1979

[21] OpenFlow Switch Specification, Open Networking Foundation, Available: http://www.opennetworking.org/

[22] M. Ba, O. Flauzac, B. S. Haggar et al, *Self-stabilizing k-hops clustering algorithm for wireless ad hoc networks*, in: 7th ACM ICUIMC (IMCOM), 2013, pp 138

[23] 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.

[24] 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.

[25] Network Functions Virtualization (NFV), Available: *OpenDaylight*, http://www.opendaylight.org/

[26] Mianxiong Dong; He Li; Kaoru Ota; Jiang Xiao, *Rule caching in SDN-enabled mobile access networks*, in Network, IEEE , 2015 , pp.40-45, July-August 2014

[27] 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

[28] 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

[29] Meneses, F.; Corujo, D.; Guimaraes, C.; Aguiar, R.L., *Multiple Flow in Extended SDN Wireless Mobility*. in Software Defined Networks (EWSDN), 20145, 2015 Fourth European Workshop on, vol., no., pp.1-6, Oct 2015.

[30] Costa-Requena, J.; Kantola, R.; Llorente, J.; Ferrer, V.; Manner, J.; Ding, A.Y.; Yanhe Liu; Tarkoma, S., *Software defined 5G mobile backhaul*. in 5G for Ubiquitous Connectivity (5GU), 2014 1st International Conference on , vol., no., pp.258-263, 26-28 Nov. 2014.