Current Trends of Topology Discovery in OpenFlow-based Software Defined Networks

Leonardo Ochoa-Aday, Cristina Cervelló-Pastor, *Member, IEEE*, and Adriana Fernández-Fernández

***Abstract*—The explosion of Internet services such as video on demand, big data, server virtualization and cloud services is among the trends driving the networking industry to change traditional network architectures to more flexible and dynamic schemes.**

Software Defined Networking is an emerging network architecture that could address the needs of services providers and networks operator. This new technology consist in decoupling the control plane from the data plane, enabling to centralize control functions in a concentrated or distributed platform. It also creates an abstraction between the network infrastructure and network applications that allows to design more flexible and programmable networks. However, in order to both services and network applications can run properly, a global and updated view of the network is required at every moment.

This paper attempts to address the main protocols and approaches of the topology discovery service provided by the controller in a single administrative domain. Also the procedure of topology discovery in a network composed by non-OpenFlow and OpenFlow switches are presented. In addition, attention is focused on Layer 2 discovery protocols LLDP and BDDP and major limitations of these procedures are discussed.

Keywords: SDN, OpenFlow, topology discovery, LLDP, BDDP.

I. Introduction

Currently the high demand for services (such as big data, cloud services, video traffic, among others) across large-scale networks with multi-domains as Internet, generates a large amount of revenue for service providers and network operators.

To be able to address these high demands of users efficiently, dynamic mechanisms of autoconfiguration of network elements according to the new policies or business requirements are needed, setting an almost impossible challenge for existing IP networks. In addition, to deploy policies of high-level network, the operators need to configure each elements of the network and often via specific low-level commands from manufacturer. This is because the plane that decides how to handle the traffic (control plane) and the plane that forwards traffic in accordance with decisions of the control plane (forwarding plane) are vertically integrated in a single network device [1]. This feature of current networks greatly hampers innovation and flexibility in network infrastructure.

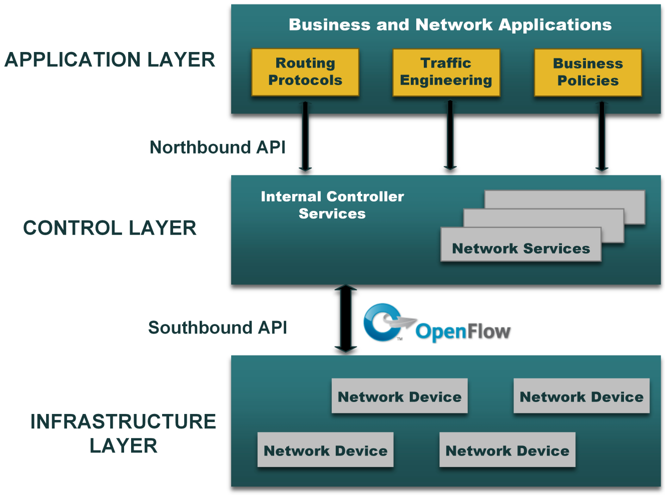

Figure 1. Layered architecture of the SDN.

In this context, Software Defined Networking (SDN) is an emerging network architecture that could address the needs of data centers, campus networks and requirements of services providers in carrier environments [2]. This new paradigm proposes decoupling the control plane of the forwarding plane, by centralizing intelligence, state of the network and control functions in an entity called the controller or Network Operating System (NOS). NOS creates an abstraction layer between underlying network infrastructure and business applications while maintaining a global network view at each instant. This new architecture lets operators to make dynamic configurations and innovations in the network via applications programmed in the top of the SDN architecture. As a result, network operators and service provider can increase automation and network control, allowing them to build flexible and programmable networks that adapt easily to the dynamic needs of business and end users.

The logical architecture scheme in layers proposed by SDN is shown in Fig. 1. The infrastructure layer or data plane composed by forwarding devices is situated at the bottom. Those physical devices are now simple forwarding elements without control or a software to take independent decisions.

The Network Operating System runs in the control layer, i.e. a software platform on commodity server technology that provides the essential resources and abstractions to facilitate the programming of forwarding devices based on a logically centralized approach [1]. In the paper we focus on topology discovery, which is a critical service provided by the control layer for the proper functioning of applications and network services.

The application layer is situated on top of the architecture, where applications that define control and logic operation of the network are located. In essence, this layer allows

The authors are with the Department of Network Engineering, Universitat Politècnica de Catalunya (UPC), Esteve Terradas, 7, 08860, Castelldefels, Spain, e-mails: {leonardo.ochoa, cristina, adriana.fernandez}@entel.upc.edu.

defining through programmed applications, via northbound APIs, policies and business requirements and then modifying the behavior of forwarding devices based on them.

The controller holds a global view of the network through the topology discovery service. This is crucial for the correct operation of other internal controller services like hosts tracker, network configuration, route planning [3] and for other network applications (e.g. traffic engineering, network monitoring, attack detection, routing protocol, among others [3]) that run on top of the architecture presented.

In this paper, we present an updated and detailed study of existing solutions and limitations of topology discovery in single-domain networks. Compared to recent approaches of topology discovery, this paper tackles the procedure of discovery in OpenFlow-based networks and in hybrid networks composed of traditional and OpenFlow switches (OF switches), which to our knowledge has not been addressed before. Furthermore, this work does a specific study of Layer 2 protocols that perform topology discovery in OpenFlow-based networks. Basically, we present to the best of our knowledge, the most representative solutions and challenges on topology discovery in SDN to date. Also, this paper examines solutions of recent works to optimize and reduce the processing of the controllers during the discovery process.

The remainder of the paper is organized as follows. In Section II an overview of the main aspect of the OpenFlow technical specification and relevant backgrounds of the Layer 2 neighbor discovery protocol LLDP are given. Concepts of topology discovery and administrative OpenFlow domain are defined in Section III. Next, in Section IV we discuss two approaches of topology discovery in single-domain networks. The paper is concluded in Section V, where the possible future work and our final remarks are presented.

II. Preliminary Knowledge

A. LLDP protocol

The Link Layer Discovery Protocol (LLDP) was standardized for the first time in 2005 by the Institute of Electrical and Electronics Engineers (IEEE), under the name IEEE 802.1AB. In 2009 the norm is superseded and is officially standardized a new version updated of the protocol, formally referred to by the IEEE as ”Station and Media Access Control Connectivity Discovery” specified in IEEE standards as document 802.1AB IEEE-2009 [4].

LLDP is a neighbor discovery protocol of a single jump, i.e. it advertises its identity and capabilities and receives the same information from the adjacent switches. Furthermore, it is a vendor neutral protocol that works on Layer 2 of the OSI model and relies on concepts of numerous proprietary discovery protocols: such as Cisco Discovery Protocol (CDP), Nortel Discovery Protocol (NDP), Extreme Discovery Protocol (EDP) and others. Next, the most important features of LLDP protocol used in topology discovery in OpenFlow-based networks will be mentioned.

Each LLDP frame is composed of a header and a payload LLDP data unit called (LLDPDU) as shown in Fig. 2. The default value of the Ethertype field in the header for every LLDP message is 0x88cc, this information is vital to identify efficiently the discovery packets in OpenFlow networks. In the destination MAC field is set a multicast destination MAC address. This address is standardized as ”LLDP Multicast address” [4] and allow traditional switches to identify LLDP discovery packets.

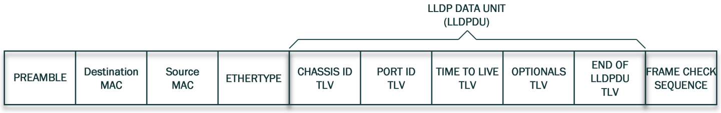

Figure 2. Structure of the LLDP frame.

LLDPDU is composed of optional and mandatory Type Length Value (TLV) structures. The payload starts with three mandatory TLVs, followed by a number of optional TLVs and ends with a special mandatory TLV in which both the type and length fields are zero. Optional TLV includes the Basic set of TLVs and the Organizationally Specific TLVs and can be used to introduce new features of discovery through the LLDP protocol. Optional TLVs is a separate issue and for not doing very long this study are left out of the scope of this paper.

The four mandatory TLVs in the frame making up the core of the operation of the discovery protocol are briefly described below:

1) Chassis ID (Type 1): This TLV structure contains the identifier of the switch that sends the LLDP packet.
2) Port ID (Type 2): This TLV structure contains the identifier of the port through which the LLDP packet is sent.
3) Time to Live (Type 3): This TLV structure contains the value of time in seconds during which the information received in the LLDP packet is going to be valid.
4) End of LLDPDU (Type 4): Special TLV structure that indicates the end of the payload in the LLDP frame.

In traditional networks, every LLDP frame is sent by the switches that support and have activated this functionality at fixed intervals of time, configured by the network administrator. In OpenFlow-based networks, switches send the LLDP messages to discover the underlying topology by request of the controller. How the discovery process works in OpenFlow networks using the LLDP protocol is explained in detail later.

III. Topology Discovery

Generating automatically a view of the network topology is a critical service that the controller has to ensure for the proper working of other services and network applications. Due to the different functional layers that can model a computer network, the network topology discovery can be performed in different layers of the same. The physical topology provides us a map where one can appreciate nodes distribution and their connections in the network while the logical topology shows the data flow between devices according to the protocols that are used on the different functional layers.

In this paper, we consider as topology discovery the tasks performed by the controller to discover switches, links and hosts in a single OpenFlow administrative domain. It is assumed that the southbound protocol used for communication between the controller and the Forwarding Plane is OpenFlow. In addition, we consider one OpenFlow administrative domain in this study, i.e., a set of OF switches operating under the control of a single Network Operating System (NOS).

IV. Discovery Topology in single-domain networks

The topology discovery in an administrative domain can be performed on a network composed only of OF switches (pure OpenFlow network, see Fig. 3) or in a hybrid network consisting of a mixture of traditional and OF switches (hybrid OpenFlow network, see Fig. 4).

In the next section, we describe how the discovery of switches, links and hosts is performed in an OpenFlow network. First, we describe the case of a network with only OF switches and mention some of the limitations that exist to perform the discovery based only with LLDP packets. Secondly, we present a hybrid network approach where the controllers needs a combination of discovery protocols to know the topology underlying. In addition, the drawbacks of this scheme are shown.

A. Network with only OpenFlow switches

The discovery of existing links between switches in a pure OpenFlow network can be performed with the single-hop neighbor discovery protocol LLDP. The discovery of the links in a network composed only of OpenFlow elements requires no other discovery techniques. This is because a switch that supports the topology discovery approach described below is located at the end of each link.

The switches that support the OpenFlow technical specifications have two major initial configurations that enable discovering the topology. Firstly, every OF switch has initially set the IP address and TCP port of a master controller and a pool of IP addresses of slave controllers to establish a connection as soon as the device is turned on. Secondly the switches have preinstalled flow rules to route directly to the controller via a Packet-In message, any message with 0x88cc EtherType (LLDP packet) received by a different port of the controller.

When the switch is initialized, searches for the master controller in the network and attempts to establish a secure and encrypted connection through TLS (Transport Layer Security) protocol to send and receive configuration messages, flow table entries, among others. The controller as part of the initial handshake sends a message (FEATURE_REQUEST_MESSAGE) to the switch who responds with a message (FEATURE_REPLY_MESSAGE). With this message it informs the controller of relevant parameters for the discovery of the links like the Switch ID, a list of active ports with their respective MAC associates, among others. Until this point, the controller knows exactly the OF switches that are connected in the network and has valuable insights to perform the discovery of the links.

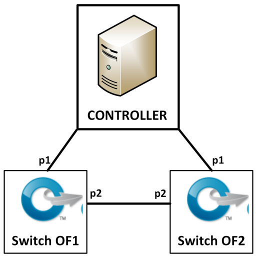

Figure 3. Pure OpenFlow-based network.

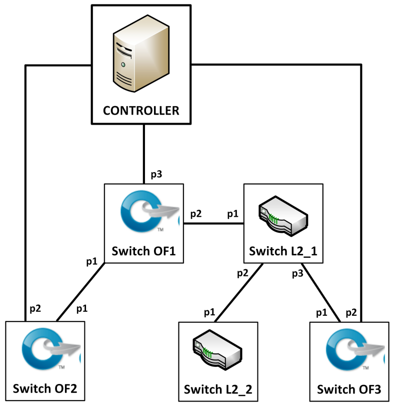

Figure 4. Hybrid OpenFlow-based network.

Based on the information obtained from the initial handshake, the controller knows the exact number of active ports on the OF switches that belong to the administrative domain. The controller generates a Packet-Out message per active port on each switch discovered on the network and encapsulates a LLDP packet inside each generated message. The destination MAC address in the LLDP packet is one of the multicast MAC addresses defined in the IEEE 802.1AB standard.

Considering that the network comprises of a set of OF switches S interconnected by a set of links L, the total of Packet-Out messages (see Eq. 1) that the controller sends out to the network to discover all existing links between OF switches with P active ports is:

$$\text{TOTAL}_{\text{PACKET-OUT}} = \sum_{i=1}^{S} P_i \qquad (1)$$

The Packet-Out message will also install the corresponding flow entries in the switch to route each LLDP packet through the port indicated in the TLV field of the LLDP message payload. When an OF switch receives a LLDP message sent by the controller, it forwards the message by the appropriate Port ID (TLV) to adjacent switches. As described above in Section II, the Chassis ID and the Port ID are included in the LLDP packet among the mandatory parameters of the payload, describing the switch that sent the message LLDP. Upon receiving the messages by a port that are not the controller port, the adjacent switches encapsulate the packet within a Packet-In message addressed to the controller. Meta-data is included in the message such as Switch ID, Port ID where the LLDP packet is received, among others.

Messages exchanged in one way between the controller and OF switches to discover the link between Switch OF1 and Switch OF2 are shown in a diagram in Fig. 5. Similarly this exchange of messages occurs in the opposite direction. After receiving those Packet-In, the controller is able to discover a link between two OF switches, based on the information contained in the LLDPDU and the data collected in the meta-data. The controller stores this information in the database and thereby updates the status of the network topology.

This process is repeated for every OF switch available on the network. The entire process of the topology discovery is performed periodically with a fixed start cycle that has a typical default value of 5 seconds. In this approach, the amount of Packet-In received (see Eq. 2) is double the number of links existing on the domain.

$$\text{TOTAL}_{\text{PACKET-IN}} = 2 \cdot L \tag{2}$$

The topology discovery in OpenFlow domains currently is not standardized. The use of LLDP protocol in discovery mechanism was inherited from the first implementation of a OpenFlow controller (NOX) [5].

An interesting approach to reduce processing cost due to topology discovery in the controller with values above 45% is presented in [6]. This solution is based on developments of the OpenFlow technical specification and proposes a simple modification to the controller behavior. Greater efficiency in the process of topology discovery based on LLDP protocol in pure OpenFlow network is achieved.

The main limitation of the LLDP protocol is that it only discovers links between adjacent switches, which has a serious impact in hybrid OpenFlow networks. This makes the LLDP packet misses when there are links with switches that do not support OpenFlow, i.e. the controller cannot discover all the links on its administrative domain if there exist switches that works traditionally. Therefore, a different approach is required to discover the topology in a network composed of traditional and OF switches.

B. Network with Traditional and OpenFlow switches

The controller of an administrative domain is the responsible for discovering the OF nodes and links that are set between these nodes through elements that do not support the OpenFlow protocol on the network. The mechanism to discover OF switches in a hybrid OpenFlow domain does not differ from the discussed above. The controller during the initial handshake with OF switches discovers their location and capabilities. In case of traditional switches, the controller is not able to discover these nodes, because there is not an initial handshake between them. This is why the controller is not going to know necessary parameters of traditional nodes such as its identity, capabilities or location on the network.

In hybrid OpenFlow networks where there are traditional switches, two types of connections between switches that support the OpenFlow technical specification may fundamentally exist.

1) *Direct links between OF switches*. This connection is a direct link between two active ports of OF switches, as shown in Fig. 4 between the switch OF1 port p1 and switch OF2 port p1. The discovery process of these links is performed by the controller in the same manner that was explained in the previous section.

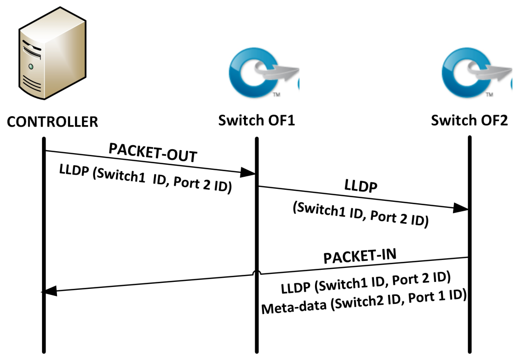
CONTROLLER
Switch OF1
Switch OF2
PACKET-OUT
LLDP (Switch1 ID, Port 2 ID)
LLDP
(Switch1 ID, Port 2 ID)
PACKET-IN
LLDP (Switch1 ID, Port 2 ID)
Meta-data (Switch2 ID, Port 1 ID)

Figure 5. Messages exchanged between the controller and OF switches in LLDP-based topology discovery.

2) *Links between OF switches across traditional switches*. This is a non-direct connection between two active ports of OpenFlow nodes but they are still in the same broadcast domain, as shown in Fig. 4 between the switch OF1 port p2 and switch OF3 port p1. In this paper, we will name this type of links multi-hop connections or indirect links indiscriminately. Then, we explain why the controller requires a different approach to find these links and how a specific solution programmed in current OpenFlow controllers works.

The LLDP-based links discovery mechanism is not applicable in this scenario because between two OF switches there may exist 1 or N traditional switches. The LLDP protocol is a mechanism of a single-hop, so that the LLDP packets would be processed and dropped automatically by a traditional switch in an indirect link. Therefore, the controller shall use a different approach in the topology discovery mechanism, basing its strategy on the use of broadcast protocols of network discovery. In this way, the controller could at least discover the links between two ports of OF switches that are not connected directly but that are in the same broadcast domain.

Currently, to make discovering links in a network composed of traditional and OF switches, some of the major OpenSource controllers utilize a combination of Layer 2 topology discovery protocols like LLDP and BDDP (Broadcast Domain Discovery Protocol).

The BDDP protocol is a specific solution programmed in OpenSource controllers such as Floodlight and OpenDayLight (ODL), for discovering of multi-hop links in a hybrid OpenFlow network [7],[8]. This approach, at time of writing this paper, does not constitute a standard, but it is of interest to discover links with traditional switches using the idea of this new protocol.

The BDDP messages present the same structure of LLDP packets, is a frame composed of mandatory and optional TLV structures as was shown in the above section. The key difference is in the field of destination MAC address of the frame header. Thus, this field has a broadcast address (ff:ff:ff:ff:ff:ff) in contrast to multicast addresses used by the LLDP protocol. This feature allows that traditional switches can forward BDDP packets by addressing the problem of single-hop discovery protocol. In this approach, multi-hop links can be discovered using traditional switches belonging to the same broadcast domain that OF switches. Another major difference is the EtherType field in the BDDP header. This protocol uses a different value to the one used in LLDP messages. Generally, 0x8999 is the value utilized in BDDP frames.

In the initial handshake, as mentioned above, the controller obtains the exact amount of active ports of OF switches that belong to its administrative domain. To discover indirect links through traditional switches, the controller encapsulates per every active port of each switch, one BDDP message inside one Packet-Out that sends to the network. With the sending of the Packet-Out an entry is included in the flow table that instructs each OF switch that receives the message.

After processing the BDDP packet, the OF switch forwards the message to neighboring switches via the port indicated in the TLV field (Port ID). This packet includes the same parameters that in LLDP recognize the switch (Switch ID, Port ID, among others).

The neighbor switch might be a device supporting OpenFlow (Example switch OF2 in Fig. 4) or a traditional switch. In the first case, the packet will do a match with the Ethertype 0x8999 of the message and it will be sent directly to the controller via a Packet-In. Identification information about the port of the neighbor OF switch that received the BDDP message is included in meta-data. Based on this information, the controller is capable of determining a link between the two OF switches, as explained above in Section V index A.

In the case that the neighbor is a traditional switch (Example switch L2_1 in Fig. 4), it will examine the destination MAC address of the packet. Thus, it will notice that this is a broadcast address (ff:ff:ff:ff:ff:ff) and it will flood the packet by all its ports. Suppose that at least one of its neighbors supports OpenFlow, it will send the message via Packet-In to the controller. In this Packet-In message the metadata are included, giving to controller the information needed to finally discover the multi-hop link.

After completing this procedure, the domain controller has a BDDP packet received via a Packet-In with the information necessary to discover an indirect link between two OF switches (example switch OF1 and switch OF3). As an illustrative example to visualize the operation explained, consider that the controller sends in a Packet-Out a BDDP message to the switch OF1. Data like Switch and Port ID of the switch OF1 are included in the payload of the BDDP message. The Switch and Port ID of switch OF3 are contained in the meta-data of the Packet-In. Based on this information the controller learns that port 1 of switch OF3 can be accessed through port 2 of the switch OF1.

In Fig. 6, messages exchanged in one way between the controller and OF switches to discover the indirect link between Switch OF1 and Switch OF3 are shown. In this way the controller discovers a non-direct connection between two active ports of OpenFlow nodes and updates in its database the status of the network topology.

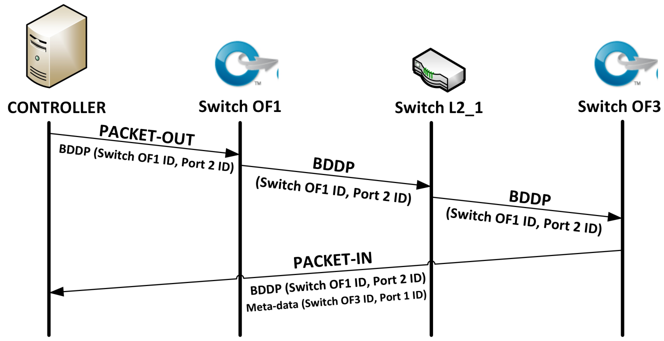
CONTROLLER
Switch OF1
Switch L2_1
Switch OF3
PACKET-OUT
BDDP (Switch OF1 ID, Port 2 ID)
BDDP
(Switch OF1 ID, Port 2 ID)
BDDP
(Switch OF1 ID, Port 2 ID)
PACKET-IN
BDDP (Switch OF1 ID, Port 2 ID)
Meta-data (Switch OF3 ID, Port 1 ID)

Figure 6. Messages exchanged between the controller and OF switches in BDDP-based topology discovery.

Although the use of BDDP protocol in hybrid OpenFlow networks can be an effective solution to discover multi-hop links between OF switches on the underlying topology, it has some limitations. Amongst the main disadvantages of BDDP protocol should be noted that, at the time this paper was written it did not constitute an approved standard. BDDP is a specific solution for topology discovery programmed in the source code of several OpenSource controllers as Floodlight and ODL [7],[8]. Using of broadcast packets to perform the discovery tasks can lead to inefficient and excessive utilization of network resources. Furthermore, as LLDP, BDDP is a link discovery protocol only since it cannot discover the existence of traditional switches in the network.

Therefore, in hybrid OpenFlow networks we propose to use a combination of LLDP and BDDP packets to discover the topology in two phases. In phase 1, the controller performs the LLDP-based mechanism to discover the topology. After receiving all Packet-In sent by OF switches, the controller can identify which active ports connect to traditional switches as those of which does not receive any input. Then, in phase 2, the controller performs the BDDP-based mechanism, by sending BDDP packets only to those ports. In this way, the controller can discover direct and multi-hop links between OF switches and the utilization of network resources due to the BDDP discovery protocol is reduced.

V. Conclusion and Future Works

In this paper, we have presented the current trends of topology discovery in OpenFlow-based networks on a single administrative domain. The topology discovery is a recurring issue in the operation of any computer network that raises the attention of people worldwide. The development of SDN offers a new approach to address this need that differs from the topology discovery in traditional networks because it allows to obtain a global view of the underlying network.

We analyzed in detail LLDP-based topology discovery mechanisms in networks composed only by OF switches and the limitations of this approach were also mentioned. The major differences with the topology discovery in networks composed by traditional and OF switches were discussed. Based on the feature that the multi-hop links are in the same broadcast domain, the controller should use broadcast protocols for discovery the topology in hybrid OF networks. Although the BDDP protocol does not discover traditional switches and its links in OpenFlow networks, it is a powerful tool to discover links between two OF switches that are not directly connected. This approach makes an intensive use of the network resources and is not currently standardized.

In future work, we intend to extend this study to networks with several OpenFlow administrative domains, presenting the main solutions and limitations of the network topology discovery mechanisms in multi-domain scenarios.

Acknowledgment

This work has been supported by the Ministerio de Economía y Competitividad of the Spanish Government under project TEC 2013-47960-C4-1-P.

References

[1] Kreutz, Diego, et al. *"Software-defined networking: A comprehensive survey."* proceedings of the IEEE 103.1, 2015: 14-76.

[2] ONF, *"Open Networking Foundation,"* 2015. [Online]. Available: https://www.opennetworking.org/

[3] Jammal, Manar, et al. *"Software defined networking: State of the art and research challenges."* Computer Networks 72, 2014: 74-98.

[4] *IEEE Standard for Local and Metropolitan Area Networks - Station and Media Access Control Connectivity Discovery.*, IEEE Std 802.1AB, 2009.

[5] Gude, Natasha, et al. *"NOX: towards an operating system for networks."* ACM SIGCOMM Computer Communication Review 38.3, 2008: 105-110.

[6] Pakzad, Farzaneh, et al. *"Efficient topology discovery in software defined networks."* Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on. IEEE, 2014.

[7] Floodlight, 2015. [Online]. Available: http://www.projectfloodlight.org/

[8] ODL, *"OpenDaylight,"* 2015. [Online]. Available: https://www.opendaylight.org/