Correlating different network topology layers in heterogeneous environments

Dennis Pellikaan, Diederik Vandevenne

UNIVERSITY OF AMSTERDAM

SURF SARA

February 11, 2013

Abstract

Network Topology Discovery is the process to automatically generate a network topology. For example, by detecting the connected devices by sending messages on the wire. This paper focuses on the usage of the Link Layer Discovery Protocol for the discovery of the physical (Ethernet) topology and how to correlate that information with topology information of other layers. Both a theoretical and a partial practical approach of correlating the different layers is covered.

In a modern network, protocols and technologies that do virtualization, aggregation and pruning of links and devices must be taken into account to be able to correctly correlate the different topologies.

It is relatively easy to create a topology based on the information that LLDP provides. Tests in the production network of SURFsara showed that LLDP is configured on most of the devices which make it a useful protocol to do topology discovery in heterogeneous networks.

The Simple Network Management Protocol is used to retrieve all topology information, which has proven to be a reliable method. The information found in the different Management Information Bases (MIB) can be linked back to a single interface index value, which makes it possible to correlate the information with other layers. Vendors tend to use proprietary solutions and they often place that information in a vendor-specific MIB. Although within this research the main focus is on the usage of standardized MIBs, vendor-specific MIBs should allow for the mapping of protocol information to the interface index.

Contents

List of Tables

List of Figures

1 Introduction

Network Topology Discovery is the process of automatically generating a network topology. For example, by detecting the connected devices by sending messages on the wire. It has many applications such as inventory management, monitoring, visualization and path finding.

Models that describe the working of computer networks often distinguish different functional layers. As a consequence, Network Topology Discovery can be done on different layers of the network. The physical topology, for example, gives an overview of the physical interconnections between all devices in the network. The logical topology displays the data flow between devices according to the protocols that are used on the different functional layers. Examples are Ethernet, which operates on the Open Systems Interconnection (OSI) Data Link layer, or Internet Protocol (IP), which operates on the OSI Network layer.

RFC 2922 [7] describes a Physical Topology Management Information Base (PTOPO-MIB) that can contain information about the physical topology of the network. However, it does not describe a method to do topology discovery and how to fill the PTOPO-MIB. Most manufactures of network devices have their own proprietary protocols such as Nortel Discovery Protocol (NDP), Cisco Discovery Protocol (CDP) and Foundry Discovery Protocol (FDP) for this purpose. Despite of the existence of a standardized way to store physical topology information, manufactures commonly use the proprietary management information base (MIB) objects that are associated with the previous mentioned protocols.

In 2005, the first version of the IEEE 802.1AB [20] standard was ratified and subsequently updated in 2009. This open standard describes a protocol called Link Layer Discovery Protocol (LLDP) that is meant to facilitate multivendor interoperability to discover the physical topology of IEEE 802 local area network (LAN) environments and to make this information available in a standardized way. Although LLDP has its own MIB objects, the PTOPO-MIB is supported by LLDP for backward compatibility reasons. Manufacturers are adopting LLDP, but it is unclear if enough devices already support it to use it in practice.

1.1 Related work

A lot of research has already been done on Network Topology Discovery. Different methods to do IP topology discovery are compared in [19]. When supported by all network devices, information gathered with Simple Network Management Protocol (SNMP) gives the most accurate and fastest results. Another, even faster approach that uses Open Shortest Path First (OSPF) information gathered with SNMP is described in [21].

Topology discovery of Ethernet networks is sometimes closely related to what is called physical topology discovery and those two terms are often interchanged. However, many methods [5] [6] [10] discover only the active Ethernet topology which does not give a complete picture of the whole physical network. Although the terms are interchanged, one should realize the differences between physical topology discovery and topology discovery of Ethernet networks.

OSI layer one and two devices are transparent to their surroundings. They forward network packets but cannot be addressed directly. Because of this and because the protocols that are designed for physical topology discovery are often proprietary, most research on physical topology discovery and Ethernet topology discovery are based on information gathered from the Address Forwarding Tables (AFT) of switches [5] [6] [10] [12] [13]. Another approach, described in [8], circumvents the stated problems by using software agents on end devices to discover the topology of the Ethernet network.

LLDP has not really caught the attention of researchers. Perhaps this is because it is relatively new and cannot be used in heterogeneous networks with older devices. Also the correlation between topology discovery information on different layers is hardly done. However, there is a paper [18] that describes how the layer two and layer three topologies can be combined.

Another thing that stands out is that most research on topology discovery is based on the use of relative simple networks based on Ethernet and IP. However, current networks use many other protocols and technologies that influences the network topology. Examples are Link Aggregation, Virtual Local Area Network (VLAN) and Virtual Routing and Forwarding (VRF).

1.2 Research question

What are the challenges with the correlation of physical topology information based on LLDP and logical topology information?

1.3 Working hypothesis

LLDP is mature enough and widely implemented which makes it a useful protocol for topology discovery in modern heterogeneous network environments.

All information needed to correlate the different network topology layers is available in the Management Information Base (MIB) of network devices.

1.4 Outline

Before discussing the theory behind topology discovery, we first look at the definitions in section 2. In section 3 LLDP is studied more closely. Within this research the reference topology is created with LLDP. Especially of interest are the situations where LLDP could produce wrong topology information. In section 4 the logical topology layers are looked at more closely in a theoretical manner. In section 5 the information structure is discussed on how to store the topology information that is retrieved from the MIB and how that can be correlated. Whereas in section 5 the information structure is discussed, in section 6 we look at how the information can be extracted from the management information base and how that information can be interpreted. An overview of our findings can be found in section 7, followed by a conclusion in section 8.

2 Topology Discovery

2.1 Physical topology definition

As already touched in section 1.1, the terms physical topology and Ethernet topology are often interchanged. Dependent on ones viewpoint and the context that is set, those two can be closely related. This is the case with Ethernet switches that are connected with each other via normal Ethernet links. However, when lower layer technologies are introduced, such as Wavelength-division multiplexing (WDM) which is a technology to multiplex multiple optical signals onto one fiber, the physical topology and Ethernet topology are not that similar anymore. The physical locations of devices and cables can also be seen as physical topology information. This is also information that is not part of the Ethernet topology. To avoid confusion, the definition of physical topology that is used in the context of this paper is more clearly explained in this section.

LLDP is used in this paper as a basis to correlate network topology information. The IEEE 802.11AB [20] standard that describes LLDP does use the term physical topology quite often when it explains its goals and capabilities. This is maybe the most important reason to not try to avoid this term within this paper. However, it might be clear that the topology that is discovered with LLDP does not encompass all possible definitions of physical topology. Whenever the term physical topology is used within this paper, it must be put in the context of LLDP. This is why the definition of physical topology defined in the IEEE 802.11AB [20] standard and stated below is applicable within this paper.

Physical network topology: *The identification of systems, of IEEE 802 LAN stations that compose each system, and of the IEEE 802 LAN stations that attach to the same IEEE 802 LAN.*

2.2 Network layers

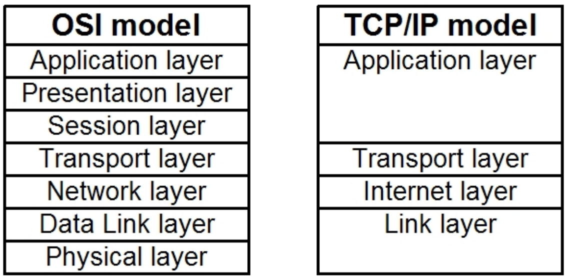
Computer networks can be complex. To make this complexity more manageable, they can be divided in functional layers. Network protocols operate on one or more functional layers and can be stacked on top of each other. They can be seen as building blocks that implement specific functions that can be combined with other protocols to create powerful applications. IP, for example, is not very useful on its own. However, combined with other protocols, it forms the foundation of the internet.

A frequently used model to describe the working of computer networks is the OSI reference model [3]. The OSI model contains seven layers. Each layer has a defined set of functions. Each layer can only interact with the layer directly above and beneath it and with the equal level layer of another system. A layer provides a service to the layer above it and depends on the service provided by the layer beneath it. Network protocols situated on a specific layer implement the functionality that is defined for that layer.

Another model that is often compared with the OSI model is the Transmission Control Protocol/Internet Protocol (TCP/IP) model [9]. In contrast to the OSI reference model, the TCP/IP model is an implementation model that was created to describe how the already implemented TCP/IP protocol stack worked. The OSI reference model was created to be used as a framework to build the OSI protocol stack. The TCP/IP model has only four layers which are less restrictive than the seven layers of the OSI model. The layers of the OSI and TCP/IP models are displayed in figure 1.

Although the OSI and TCP/IP models are great to learn about networking, they have some limitations which make them often not suitable to create a detailed view of all functional layers of complex networks. Protocols do not

Figure 1: The OSI and TCP/IP models



OSI model
Application layer
Presentation layer
Session layer
Transport layer
Network layer
Data Link layer
Physical layer

TCP/IP model
Application layer
Transport layer
Internet layer
Link layer

always correctly fit one of the defined layers and can be stacked in various ways that do not follow the strict layer hierarchy.

Computer networks are also often depicted in the form of graphs. The basic elements of a graph, vertices and edges, are too limited to display all functional layers of complex networks. However, by introducing labels, colors and clustered elements, more detail can be given. The downside of graphs is that they can become cluttered quite rapidly which defeats its purpose of creating a clear overview of the whole network.

To overcome the limitations of basic graphs and the OSI and TCP/IP models to represent computer networks, the ITU-T created the ITU-T G.800 recommendation. This recommendation defines functional elements that can be used as building blocks to represent complex multi layer networks. The ITU-T G.800 recommendation does not have a fixed number of layers as the OSI model does, but describes adaptation and termination functions that are used to connect different layers.

3 Topology based on LLDP

The link layer discovery protocol allows devices attached to an IEEE 802 LAN to advertise its system's information and capabilities to other devices on the same LAN. The information fields in an LLDP frame are contained in an LLDP Data Unit (LLDPDU) as a sequence of variable length elements, that each include type, length, and value fields (known as TLVs). Table 1 describes each TLV element. As show in table 2, each LLDPDU contains the following four TLVs, it may contain zero or more optional TLVs, and is followed by an TLV indicating the end of the LLDPDU. The concatenation of the chassis ID and the port ID form the MAC service access point (MSAP). The MSAP is an identifier to identify a port/agent to an associated device.

Table 1: Type, length and value (TLV)

Field	**Description**
Type	Identifies what kind of information is being sent
Length	Indicates the length of the information string in octets.
Value	Value is the actual information that needs to be sent

Table 2: LLDP Data Unit

Chassis ID TLV
Port ID TLV
Time to Live TLV
Optional TLVs
End of LLDPDU TLV

As mentioned before, all TLVs are placed in a single LLDPDU. The LLDPDU is then transmitted to its neighbour. For each port their is an instance of an LLDP agent. When there is more than one media access control (MAC) address configured to a port, then there is one instance of an agent for each MAC address. The IEEE 802.1AB Standard recommends an interval of 30 seconds between each transmission, but this may be changed, depending on the device.

3.1 LLPD operational modes

LLDP is a unidirectional protocol. An LLDP agent can transmit information associated with its MSAP. An LLDP agent can also receive information of a system associated with a remote MSAP. LLDP allows the transmitter and the receiver to be separately enabled. This allows a device, for example, to receive LLDP information of its neighbours, but does not transmit its local information to its neighbours.

3.2 Management Information Base

LLDP needs a place to store the information it receives from its neighbours and this is provided by means of the MIB. LLDP uses well defined MIB objects and they are well structured. The MIB information can be retrieved using the LLDP-MIB [20, sec. 11.5]. The LLDP MIB is divided into two major parts. Firstly, there is mandatory basic MIB. This MIB holds all mandatory information of the local device and from its neighbours. Secondly, there is the organizational part, which may contain zero or more organizational specific MIBs.

3.3 Uniquely identifying each device

When describing a topology it is imported that each device can uniquely be identified within the topology. LLDP does not provide a (globally) unique identifier to identify a device. However, LLDP mandates the use of a chassis ID field with the soul purpose of being the chassis identifier [20, p. 5], and that makes it the best choice to identify each device. There are 7 chassis ID subtypes and each subtype indicates the basis of the chassis ID. Table 3 gives an overview of the possible subtypes and their references. An important observation that needs to be made is, depending on the vendor, that the administrator sometimes can decide which subtype is used, and with certain subtypes it is also possible to fill in the information that the chassis ID holds. Because of this reason it is important that within an administrative domain the subtype is chosen carefully to be certain that the chassis ID is unique.

Table 3: Chassis ID subtype enumeration

ID subtype	ID basis	Reference
0	Reserved	
1	Chassis component	EntPhysicalAlias when entPhysClass has a value of 'chassis(3)' (IETF RFC 4133)
2	Interface alias	IfAlias (IETF RFC 2863)
3	Port component	EntPhysicalAlias when entPhysicalClass has a value 'port(10)' or 'backplane(4)' (IETF RFC 4133)
4	MAC address	MAC address (IEEE Std 802)
5	Network address	networkAddress [a]
6	Interface name	ifName (IETF RFC 2863)
7	Locally assigned	local [b]
8-255	Reserved	

[a] networkAddress is an octet string that identifies a particular network address family and an associated network address that are encoded in network octet order. An IP address, for example, would be encoded with the first octet containing the IANA Address Family Numbers enumeration value for the specific address type and octets 2 through n containing the address value (for example, the encoding for C0-00-02-0A would indicate the IPv4 address 192.0.2.10).

[b] local is an alpha-numeric string and is locally assigned.

IEEE Std. 802.1AB-2009

3.4 Multiple neighbours on a single port

LLDP supports multiple neighbours on a single port. The total amount of neighbours is not infinite and this needs to be taken into consideration when describing the topology. LLDP can also falsely interpret other devices as being neighbours.

3.4.1 Unmanaged network switches

LLDP supports three destination MAC addresses [20, p. 19]. Table 4 gives an overview of these address.

When LLDP sends a LLDPDU to one of these addresses and the destination device is aware of these protocols, than these address are only forwarded when allowed by the specifications. The MAC address 01-80-C2-00-00-03, which is associate with the nearest bridge, is the most commonly used and is recognized by all types of bridges. However if a device (e.g. NON 802.1D layer two switch) does not recognize the destination address as reserved, it may very well forward the address to all its other ports. 802.1D [2] is the IEEE MAC Bridges standard, which includes bridging, STP and others. Unmanaged switches typically do not support this standard. When multiple devices are interconnected through such a device then this could easily lead to confusing when one looks at the neighbouring information found in the LLDP MIB. Figure 2 shows how three devices are all connected to a NON IEEE 802.1D switch. All three devices both send and receive LLDP information. Because the switch forwards the LLDPDU

Table 4: MAC addressess used by LLDP

Nearest bridge	01-80-C2-00-00-0E	Propagation constrained to a single physical link; stopped by all types of bridge
Nearest non-TPMR bridge	01-80-C2-00-00-03	Propagation constrained by all bridges other than TPMRs; intended for use within bridged networks
Nearest Customer Bridge	01-80-C2-00-00-00	Propagation constrained by customer bridges; this gives the same coverage as a customer

IEEE Std. 802.1AB-2009

MAC to all other ports, each device now sees all the other devices connected to the same switch. From an LLDP perspective, all devices are now directly connected and are seen as each others neighbours, as seen in figure 3.

Figure 2: Topology in reality

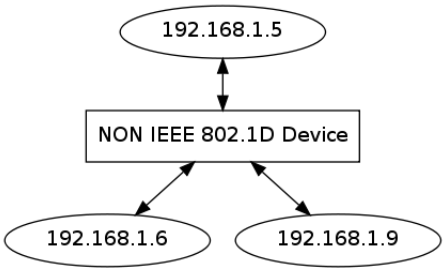
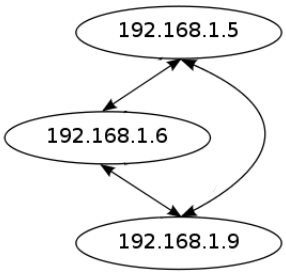


Figure 3: Topology as seen by LLDP



3.4.2 Number of neighbours

According to the IEEE 802.1AB-2009 Standard [20], the amount of space needed in the LLDP remote system MIB is beyond its scope. It may not always be possible to add a neighbour to the system MIB with limited memory. The standard describes the following three possible methods of handling this type of situation, it is up to the vendor on how to implement a solution to this problem:

a) Ignore and not process the new neighbour's information.

b) Delete the information from the oldest neighbour(s) until there is sufficient memory available to store the new neighbour's information.

c) Randomly delete neighbours until there is sufficient memory available to store the new neighbour's information.

Although the implementation of the solution to this problem lies outside the scope of the IEEE 802.1AB-2009 Standard, it is necessary for the implementation to keep track of the situation when it occurs. LLDP does this by properly updating the variables tooManyNeighbors and tooManyNeighborsTimer. The variable tooManyNeighbors identifies when there is insufficient space in the LLDP remote system MIB to store information from all active neighbours. The variable tooManyNeighborsTimer indicates the minimum time this condition exists.

3.5 LLDP and security

Security has not been the main topic of this research, but it does have an important role when automated topology discovery is been used. The issue with security can be divided into two parts:

1. SNMP versions prior to SNMPv3
2. LLDP data between devices

SNMP versions prior to SNMPv3 Older versions of SNMP do not provide adequate security. SNMPv3 is needed to ensure that sufficient cryptography and authentication is used for access to a specific MIB. Using older versions of SNMP than SNMPv3 could lead to leakage of sensitive information, unprivileged access and alteration of data on the wire. Although LLDP itself is not responsible for the choice of the SNMP version used, it should be taken into consideration.

LLDP data between devices When a device that supports LLDP is enabled, it simply starts transmitting system information to all neighbours it is connected to. This makes LLDP very easy to implement, since it requires little or no configuration. LLDP makes a few assumption regarding the security and depends on the network administrator to address these assumptions. First, the LLDPDU packets are sent in the clear. LLDP makes no use of encryption or authentication. It simply sends its information on the wire. Second, LLDP does not verify the source of the LLDP information it receives from its neighbours. This could allow LLDP to process wrong topology information or hide certain events of happening. The latter could potentially lead to bigger problems when topology discovery is used for monitoring purposes.

A more complete overview of security considerations that need to be made when using LLDP can be found in the IEEE 802.1AB Standard [20, sec. 11.4]

4 Logical Topologies

In this section, different logical topologies will be discussed and compared with each other and with the physical topology that is discovered with LLDP. The main topology layers discussed are the Ethernet topology and the IP topology which respectively correspond to OSI layer two and OSI layer three. However there are a lot of other protocols and technologies that operate on the same layers and do have (a big) influence on the topology. One way to deal with this is to create more (sub)layers but this does not always work very well.

Another way to deal with this is to classify these protocols and technologies according to their core behavior [11]: virtualization, aggregation or pruning.

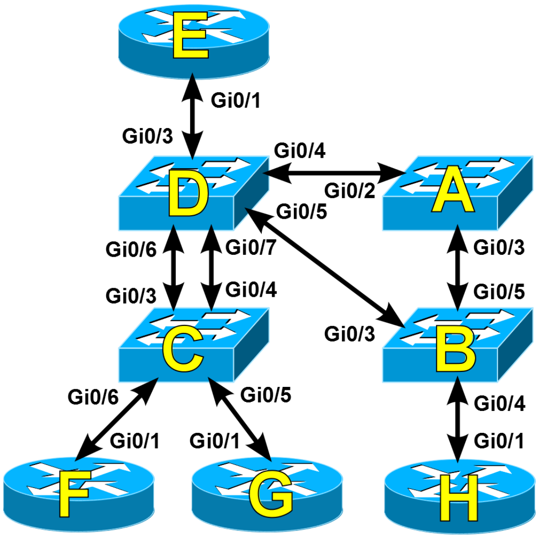
These core behaviors can operate on links or devices and on different OSI layers. Table 5 gives an overview of the classification of some protocols and technologies that influence the topology in certain ways.

Table 5: Classification of technologies and protocols that influence the topology

Protocol / Technology	**Behavior**	**Device / Link**	**OSI Layer**
LACP	Aggregation	Link	Layer two
STP	Pruning	Link	Layer two
VLAN	Virtualization	Device	Layer two
VLAN tagging (802.1Q)	Virtualization	Link	Layer two
VRF	Virtualization	Device	Layer three

4.1 The reference network

Figure 4: The physical topology discovered by LLDP



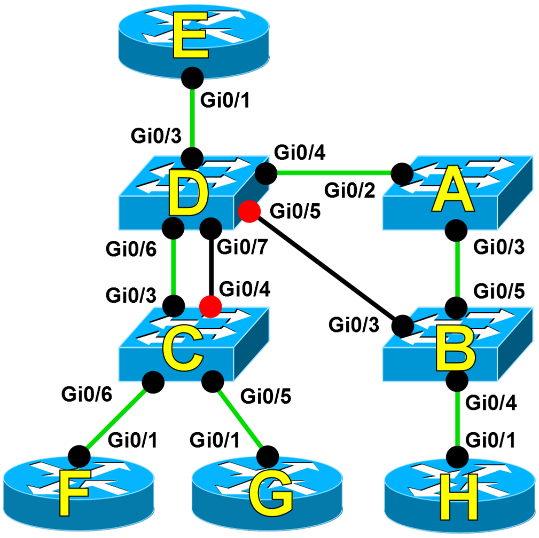
The reference network that will be used within this section is displayed in figure 4. It shows the devices and interconnections that are discovered by LLDP. The double arrows depict the LLDP messages that are sent by each device. The names of the interfaces that connect the devices are also shown. The letter on each device represents the chassis ID (MAC address) of that device. Devices A, B, C and D are layer two switches. Devices E, F, G and H are routers.

4.2 Ethernet and the Spanning Tree Protocol

Ethernet is a family of protocols that is the standard MAC in wired LANs. In modern Ethernet LANs, end-devices are interconnected with each other through switches, which are in essence multi-port bridges. Ethernet works on the physical and data link layers of the OSI model. An Ethernet LAN is also called a broadcast domain because all devices in an Ethernet LAN can reach each other by sending a packet to a special broadcast destination address. Devices that do not forward Ethernet frames and operate on the network layer and above can be seen as end-devices in an Ethernet LAN. Routers form boundaries between Ethernet LANs and do not forward Ethernet frames.

End-devices in an Ethernet LAN communicate with each other by their MAC address. A switch learns the MAC addresses that are associated to a specific port by looking at the source MAC address of Ethernet frames that are arriving at that port and stores this information in an Address Forwarding Table (AFT). When the destination MAC address of an Ethernet frame is not yet stored in the AFT, a switch does not know how to forward that Ethernet frame. In this case, a switch will act like a hub and flood the Ethernet frame out of every port except for the port it arrived on. Ethernet and the bridging operation are respectively described in IEEE 802.3 and IEEE 802.1D.

Figure 5: The active Ethernet topology



As already described in section 1.1, most research on topology discovery of Ethernet LANs focuses on the AFT [5] [6] [10] [12] [13]. The use of AFTs to create the topology of an Ethernet LAN has some downsides. One downside is that those algorithms cannot guarantee a correct topology in every situation. Another downside is that AFTs are often not complete. It is proven that topology discovery with incomplete AFTs is an NP-hard problem [13].

Figure 5 shows the active Ethernet topology of the reference network. The green links between the devices form the spanning tree. The red dots are interfaces (Gi0/4 on switch C and Gi0/5 on switch D) that are blocked by Spanning Tree Protocol (STP) to prevent loops within the Ethernet LAN.

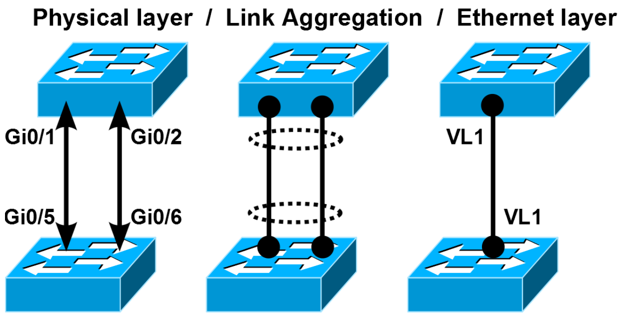
Because of the way LLDP works, the physical topology discovered by LLDP and the active Ethernet topology are not very different. This is why LLDP can be used as a basis to discover the active Ethernet topology without the need for reading out the AFTs of switches.

Without the help of some other protocol, the chance of Ethernet frames endlessly looping through an Ethernet LAN is very big when there are multiple paths to the same destination. To prevent this, STP has been invented. STP creates a spanning tree in the Ethernet LAN by blocking one interface of each link that forms a redundant path. The active Ethernet topology, which shows the data flow on this layer, can be created by combining STP information with the topology discovered by LLDP. STP and an improved version called Rapid Spanning Tree Protocol (RSTP) are described in IEEE 802.1D.

4.3 Link Aggregation

Link Aggregation [1] is a form of inverse multiplexing where several links are combined to create a higher bandwidth virtual link. A virtual interface is created on the devices on both sides of the links and associated with the interfaces that one wants to aggregate. LLDP will still discover all physical links between the devices but STP will use the virtual interface to calculate the spanning tree. This means that only the virtual, aggregated link should be visible in the active Ethernet topology. Although it might look like a small issue, it can be important with automated topology discovery. Besides the fact that the higher capacity aggregated link will not be displayed, the resulting topology will appear to contain one or more loops because the aggregated interfaces do not have STP information.

Figure 6: Topology differences with Link Aggregation



An example of the topology differences when Link Aggregation is introduced is shown in figure 6. The first pair of switches displays the physical topology with the double arrows that depict the LLDP messages that are exchanged. The second pair of switches displays the aggregation of the two links with the two dotted circles. The third pair of switches displays the active Ethernet topology. It shows only one (aggregated) link. Note that the interface name is changed to 'VL1' to emphasize the fact that it is a virtual link.

4.4 Virtual Local Area Networks

An Ethernet LAN consists of one broadcast domain. This can have consequences in terms of performance and security. The use of VLANs [4] is a way to partition an Ethernet LAN in multiple isolated broadcast domains. There are two ways in which this virtualization is achieved. The first one is by assigning interfaces to a specific VLAN. Each switch stores a table with an interface to VLAN mapping. When an Ethernet frame arrives on an interface, it can only be forwarded through another interface (and its associated physical port) that is a member of the same VLAN. This means that Ethernet frames belonging to only one VLAN are allowed on a specific link. Note that VLAN membership of an interface and the corresponding link is only locally maintained. It is very well possible that an interface on a switch is a member of VLAN 10 while the interface on the other side of the link is a member of VLAN 20. This might be a configuration error or a way to do VLAN translation.

Another way to isolate each VLAN is by adding a VLAN tag to an Ethernet frame. In this manner, more VLANs can share the same link. A port that is associated with a link that carries tagged Ethernet frames is often called a trunk port. In the same manner, a port that is a member of a specific VLAN is called an access port. With tagged Ethernet frames, the VLAN membership of the (virtual) link is maintained. When a tagged Ethernet frame arrives on a trunk port, the tag is removed and the Ethernet frame is forwarded through an interface (and its associated physical port) that is a member of the same VLAN. This also includes other trunk ports that are configured to carry tagged Ethernet frames associated with that VLAN.

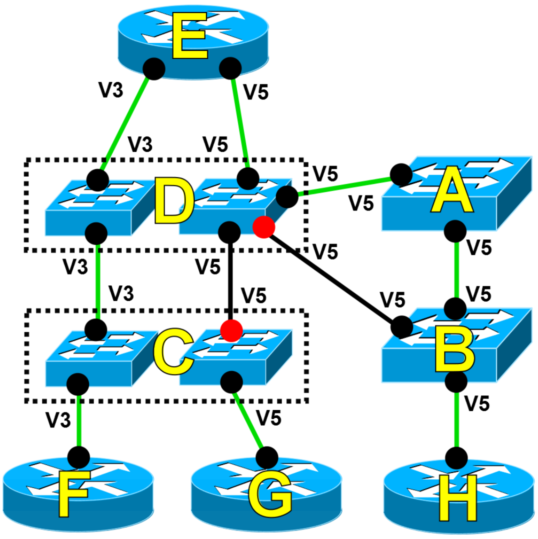
The concept of VLANs introduces other concepts that are related to VLANs. The best way to think about VLANs is that it creates multiple separate Ethernet LANs. This means that a switch that is configured with multiple VLANs is actually a group of virtual switches, one for each VLAN. Access ports that are a member of a specific VLAN are connected with the virtual switch that is associated with that VLAN. A trunk port is connected with every virtual switch that is associated with the VLANs it is configured for.

A router interface can also be configured to support a trunk link, although in a different way as with a trunk port on a switch. A virtual interface, also often called a sub-interface, should be created for each VLAN that is configured on the trunk. These sub-interfaces can be configured with an IP address and accept the tagged Ethernet frames that are associated with the VLAN they are configured for. In this way, a router is able to route packets between different VLANs.

Another device that can do inter-VLAN routing is a layer three switch or, more general, a multi layer switch. This device combines switch and router functionality. Besides a virtual switch for each VLAN, it also contains a virtual router. This virtual router has one virtual interface for each VLAN that is attached to the virtual switches associated to those VLANs. In Cisco parlance, these virtual interfaces are called Switch Virtual Interface (SVI)s. In Juniper parlance they are called Routed VLAN Interface (RVI)s. SVIs and RVIs can be configured with an IP address but may not be confused with a sub-interface on a router. A sub-interface is associated with one physical port while an SVI or RVI is associated with all ports that belong to a specific VLAN.

Just like with a normal Ethernet LAN, loops can occur within VLANs and a protocol is needed to prevent this from happening. STP and RSTP can work together with VLANs but can create some problems when all VLANs are not configured to use the same links between switches. After all only one spanning tree is created for all VLANs. To resolve this issue, Cisco developed Per-VLAN Spanning Tree Plus (PVST+) and Rapid PVST+ to create one spanning tree for each VLAN. Multiple Spanning Tree Protocol (MSTP) [4] is an open standard that can create multiple spanning tree instances. Each VLAN can be associated with one spanning tree instance.

Figure 7: VLAN topology



With a simple Ethernet LAN, one has only to combine the topology discovered by LLDP with information about STP or RSTP and Link Aggregation to create a correct topology, while one should take account of many more elements when a more complex Ethernet LAN that consists of two or more VLANs is used. All access ports on switches should be related to a VLAN. The same is true for SVIs and RVIs. VLAN membership of trunk ports and sub-interfaces on routers should be identified. Sub-interfaces should also be related to a physical port. Spanning tree information derived from STP, RSTP, MSTP or any other similar protocol should be associated with each interface associated with an access port. Trunk ports on switches should receive spanning tree information for every VLAN that it is associated with.

Figure 7 shows the topology of the reference network when VLANs are introduced. The reference network is a relative simple example and contains only two VLANs, no multi layer switches and only one trunk link between switch D and router E. It should be clear that more complex topologies can be created. The dotted rectangles represent the borders of a physical device that consists of several virtual devices. VLAN 3 and 5 on switch C do use different links to connect to switch D. Because STP is used and the interface on switch C that connects VLAN 5 with switch D is blocked as a result, router G is disconnected from the rest of the network.

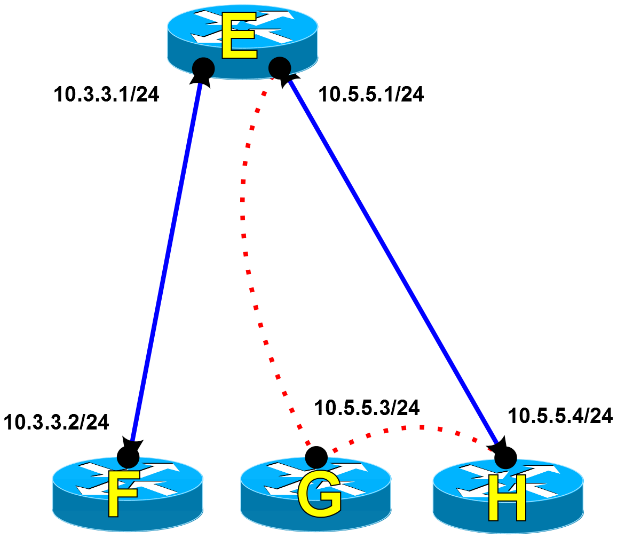
4.5 Internet Protocol

The IP protocol (both version 4 and 6) works on the network layer of the OSI model. End-devices communicate with each other by their IP addresses. Routers and multi layer switches forward IP packets to the destination IP address of a packet by looking up the next hop neighbour with the shortest route to the destination in their routing table. Besides the IP address of the next hop neighbour, routers and multi layer switches should also know which local interface to use to forward the packet. The next hop IP address should be also be translated to a MAC address for the packet to be able to cross the local LAN and reach the neighbour.

To build an IP topology, the routing table can be consulted to read out the next hop IP addresses and the corresponding local interfaces. However, not all routes refer to a next hop neighbour. There are four different types of routes that can be distinguished: direct, indirect, invalid and other. Only the indirect route types refer to a valid neighbour. The direct routes refer to the local device.

The relationship between neighbours on the IP layer is in comparison with LLDP not always as solid. With dynamic routing protocols there is active communication between neighbouring routers but routes can also be configured manually. In this case, one cannot be sure if the neighbour exists and if communication with that neighbour is possible. IP neighbour relationships are also unidirectional. Return traffic does not have to take the same path back and it could very well be that a router does not have a route to another layer three device from which he receives traffic from.

Figure 8: IP topology



To identify a neighbour, one cannot rely on the next hop IP address alone. In some cases, one IP address is assigned to multiple devices. With only an IP address, it is also not possible to determine if the potential neighbour is connected to the same LAN. A solution for this could be to read out the Address Resolution Protocol (ARP) table to translate the IP address to a MAC address and use the combination of these two addresses to check if the neighbour relationship really exists. However, the ARP table is not a reliable source because the information in this table can be aged out when the communication has stopped for a while. Another solution is to make use of neighbour tables build by routing protocols such as OSPF but this does not work for static routes. A more reliable solution is to do path finding on a lower layer to check if the potential neighbour is connected to the same LAN.

When the above issues are taken care of it is relatively easy to correlate the IP topology with the active Ethernet topology. The interfaces of the neighbouring routers that are configured with an IP address should also have been discovered with LLDP. Some extra logic is needed when VLANs are used because multi layer switches and routers may use virtual interfaces that are not discovered with LLDP. The SVI or RVI from a multi layer switch should be traced back to a physical interface that is connected with the same LAN as the IP neighbour. The same applies for sub-interfaces on routers.

Figure 9: Complete topology

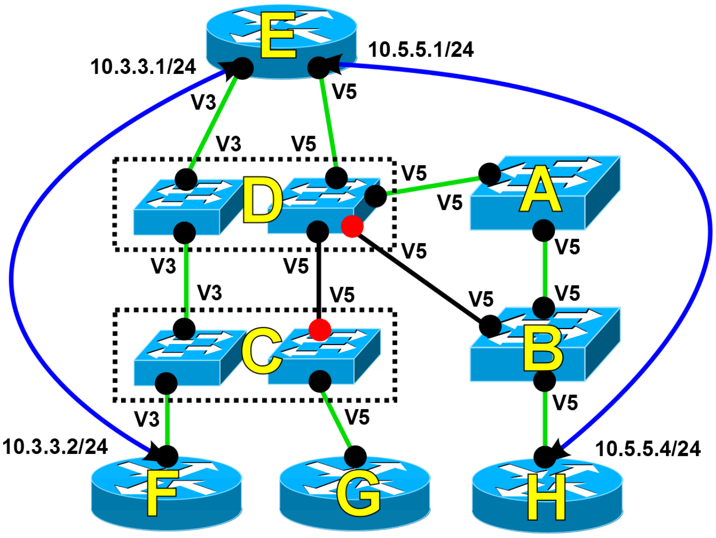
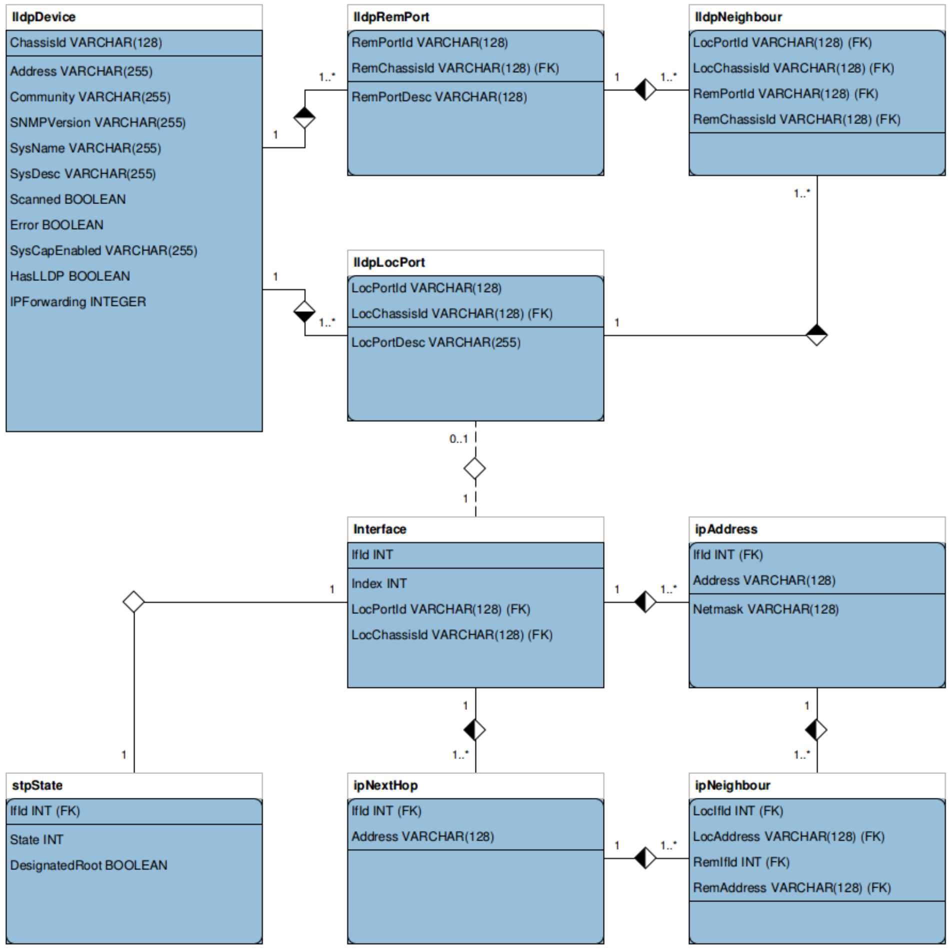


Figure 8 displays the IP topology of the reference network. Both routers F and H are connected with router E and the other way around. This is depicted by the blue lines with the double arrows. There is also a red dotted line drawn between router G and E and router G and H. This is done because of the problem with the blocked port that has already been shown in picture 7. When a dynamic routing protocol is used on all routers, router G and E and router G and H would not be neighbours. When static routes are configured between the mentioned routers the result would be different. Theoretically, router G and E and router G and H are neighbours in that case because they have routes to each other. However, they would not be able to forward traffic to each other because one interface on the path is blocked. This is a nice example on how those layers interact with each other and on how a lower layer affects a higher layer. Figure 9 displays a combination of the OSI layer two and three topologies.

5 Correlation

This section discusses how the information gathered by SNMP from the MIB can be structured into a data structure that allows for correlating the different layers. The data structured is created in the form of an Entity Relationship Diagram (ERD). The principal behind this ERD is that the information model can be extended with other protocol information. To accomplish this, a database was designed to provide a basis on which future protocols and information could, relative easy, be added in the form of tables. When the ERD is implemented in a database and the tables are populated with their corresponding data, the correlations can be made by means of the Structured Query Language (SQL). Figure 10 shows the ERD of the database as it is implemented. In the following sections the ERD is further explained. The next chapter discusses how the information found in the MIB can be retrieved and interpreted.

Figure 10: Database design



5.1 LLDP

The four top tables, all starting with *lldp*, hold the minimal information to describe the topology based on the LLDP information. Each LLDP device has zero or more local ports and zero or more remote ports. Both are stored in the *lldpLocPort* and *lldpRemPort* tables respectively. LLDP only fills local port information when there is an active link connected to it. Remote port information is only filled when a neighbour sends LLDP information on that link. The *lldpNeighbour* table is populated with links between local ports and remote ports. Each local and report port is, of course, related to a device. The reason why the *lldpLocPort* and *lldpRemPort* are separated, is because it is possible for a device to see its neighbours, but that it is not possible to read out that neighbour's MIB directly. This could be because SNMP read access is not allowed to that particular device, but it does send LLDP information to its neighbours to which there is read access to the LLDP MIB.

When it is possible to read out the local port information from a device, it is possible to find the associate interface index. Therefore the *Interface* table can be filled based on the correlation made between the LLDP local port and the interface index.

5.2 STP and IP

With LLDP filling the Interface table, it is now possible to correlate other protocol information to the interface index. Both the STP MIB and the IP MIB have object identifier (OID)s for mapping their local MIB indexes to their associate interface indexes.

5.3 Interface

For different layers and protocols to be correlated, it is imported how these are linked within the device and how this information is available in the MIB. According to RFC 1213 [16], each interface, be it virtual, physical etc, should be assigned a unique number and it must stay the same from one re-initialization to the next re-initialization. It is important to clarify that the interface index does not necessarily correspond to a physical port on the device. Extra steps need to be made in order to make this relation.

In the current database design the *Interface* table is filled based on the correlation between the LLDP local port and the interface index. The information model is designed such, that is should be possible to fill the *Interface* table by means of other methods. This method makes this model more flexible and allows for future extensions.

6 Implementation

This section addresses some of the solutions and challenges of describing a topology based on the information that is gathered from the SNMP MIB. Considering the complexity of how protocols interact, it is of paramount importance to have a true unique identifier that is linked to an interface. The interface index, as described in RFC 2863 [15], is used for this purpose. The interface index holds a central position for linking different kinds of information from different MIBs to that of a single interface.

Describing a topology, solely based on information gathered via SNMP can be a daunting task. Standardization has led to some well defined guidelines for vendors to follow. But as soon as more complex protocols are used, many vendors return to proprietary solutions. These protocols are not included in the implementation, but of main importance is that the implementation of these protocols should always allow for a solution to be mapped to the interface index.

The correlations between the different protocols and the interface index are only shown for the minimal information required. General information like system name or interface name, to name a few, follow the same procedure like the steps described below. For a detailed overview of the available MIBs used in the proof of concept (PoC) [1], can be found in appendix B.

6.1 Reading out SNMP data

The PoC implementation of the automated topology discovery makes use of the standard SNMP tools provided in the Debian package repository. The following command syntax is used to extract the needed information:

```
snmpwalk -On -Oe -PR -Ih -v VERSION -c COMMUNITY_STRING ADDRESS OID
```

6.2 Mapping LLDP to the interface index

Each LLDP OID needs to be linked together using the MIB index. This MIB index is only relevant within the context of a specific MIB and on a specific device. Each OID returns zero or more results, depending on the OID, and concatenates the MIB index to the LLDP OID. Other information can also be concatenated to the LLDP OID, therefore it is important to understand the structure of all relevant OIDs.

6.2.1 Port ID subtype

Linking the LLDP port ID with an interface index need be done in three steps. First it is important to look at the subtype of the port ID. This information can be retrieved using the *LldpPortIdSubtype* MIB object, which returns a list of all LLDP interfaces on which LLDP is active. Table 6 shows an example of port ID subtype 5 [20, p. 28] , which refers to the interface name.

Table 6: LLDP port ID subtype

Base OID	MIB index	Type	Value
.1.0.8802.1.1.2.1.3.7.1.2	.24	INTEGER	5

6.2.2 Port ID

The next step is to retrieve the port ID and linking this to the subtype. The *lldpLocPortId* OID returns a list of all interfaces with the corresponding MIB index and the port ID value. Depending on the port ID subtype, the SNMP string returned can contain different data types. Table 7 shows an example of a port ID with a string value.

Table 7: LLDP port ID

Base OID	MIB index	Type	Value
.1.0.8802.1.1.2.1.3.7.1.3.1	.24	STRING	Gi0/24

[1] https://github.com/SNE-RP1/Topology_Discovery

6.2.3 Interface index

Finally, the port ID can be mapped to the interface index, as shown in table 8. Depending on the port ID subtype, a different OID needs to be queried in order to get the interface index. Continuing on the examples provide in table 6 and 7, a lookup for the interface index can be done using the MIB OID *ifName* from the IF-MIB [15]. The structure slightly differs than that of the of the information returned by the LLDP OIDs. The interface name is returned in the value field and the interface index is concatenated to the base OID.

Table 8: Mapping the LLDP port ID to the interface index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.31.1.1.1.1	.10124	STRING	Gi0/24

With these three steps it is possible to map the LLDP port ID to the interface index. The interface index is needed to correlate other layers with LLDP. In some cases the port ID has a locally assigned value, therefore there is no guarantee that the port ID can always be mapped to the interface index. This would have a serious drawback, since it would then be very hard to correlate LLDP with anything else.

6.3 Mapping STP to the interface index

For the mapping of the spanning tree protocol to the interface index, the standardized MIBs BRIDGE-MIB [17] and IF-MIB [15] are used to read out the STP information. The BRIDGE-MIB only provides information on standard STP and RSTP. The reason for this is that there are different variants of STP, such as Cisco's Per-VLAN Spanning Tree (PVST) and Juniper's VLAN Spanning Tree Protocol (VSTP), and these protocols are proprietary. The main drawback of not using the proprietary MIBs is that a lot relevant information is lost. Both PVST and VSTP, provided as examples, are used to incorporate with VLANs, which is important when correlating the Ethernet layer with the IP layer.

Mapping the STP port state information to the interface index can be done in two straightforward steps. First the port state is read, which is linked to an internal STP MIB index. Next the MIB index is mapped to the interface index.

6.3.1 STP port state

The port state can be read with the *dot1dStpPortState* OID of the BRIDGE-MIB [17]. The information returned is the internal MIB index concatenated to the base OID, with the current port state as its value, as shown in table 9 as an example.

Table 9: Mapping the STP port state to the STP MIB index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.17.2.15.1.3	.24	INTEGER	5

6.3.2 Interface index

When the STP MIB index is known, then the interface index can directly be retrieved. In table 10 is shown that MIB index 24 maps to interface index 10124.

Table 10: Mapping STP MIB index to the interface index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.17.1.4.1.2	.24	INTEGER	10124

6.4 Mapping IP to the interface index

Describing the layer three topology can be done by retrieving the local interface IP address(es) assigned to an interface and by retrieving the next hop information from the routing tables. The MIBs as described in RFC 1213 [16] and RFC 4292 [14] are used to retrieve the relevant IP information of each interface. Because the assigned IP addresses and next hop information come from different MIBs, it is important to note that different MIB index numbers may be used, thus each MIB needs to translate its own MIB index to the interface index.

6.4.1 IP Address

The OID *ipAdEntAddr* as defined in RFC 1213 [16] is used to retrieve the IP address associate the a specific interface. As shown in table 11, the internal MIB index corresponds to the actual IP address that is assigned to the interface. Still this four octet number should only be used as an internal reference and not as the actual value. Translating the MIB index number can now be done in one single extra step, as shown in table 12.

Table 11: Mapping IP Address to the IP MIB index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.4.20.1.1	.10.10.10.2	IpAddress	10.10.10.2

Table 12: Mapping IP MIB index to the interface index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.4.20.1.2	.10.10.10.2	IpAddress	523

6.4.2 Next hops

The next hop information gathered with *ipForwardNextHop* MIB object, shows the neighbouring relationship with other IP layer devices. As shown in table 13, the MIB shows a long string, which also holds the netmask for this particular route. In this stage, routing decissions are not part of the topology description and therefore only the value is of importance. In table 14 can be seen that the complete MIB index as discovered above, is needed to retrieve the associated interface index.

Table 13: Mapping the next hop Address to the MIB index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.4.24.4.1.4	.10.10.30.9.255 .255.255.255 .0.10.10.10.8	IpAddress	10.10.10.8

Table 14: Mapping the next hop MIB index to the interface index

Base OID	MIB index	Type	Value
.1.3.6.1.2.1.4.24.4.1.5	.10.10.30.9 .255.255.255.255 .0.10.10.10.8	INTEGER	523

6.5 Mapping the interface index to a physical port

The interface index number itself does not necessarily corresponds with the interface labels that can be read from the physical device. In the past it was possible to get the physical port number by looking up the interface index. With the introduction of the ifTable MIB it has become increasingly more difficult to map an interface index to a physical port. The ifTable MIB is used to describe to interface sub-layers. RFC 2863 [15] provides the *ifName* MIB object as a solution to this problem.

7 Findings

7.1 LLDP usage statistics at SURFsara

Table 15 shows the current state on how LLDP is implemented and supported in the production environment of SURFsara. The rest of the devices did not have SNMP enabled (10 devices), or where devices maintained by external parties or where behind a firewall (29 devices)

We were able to scan a total of 95 devices. From the devices that we were able to be scanned, 10 reported back an error. An error can be caused by a timeout during an SNMP read attempt, wrong SNMP version or community string, or when a device does not support SNMP. 29 other devices were discovered because of neighbouring information provided by LLDP. These devices support LLDP, but were maintained by external parties and could not be read via SNMP. Considering a total of 124 devices, just over 83% supports LLDP.

Table 15: LLDP statistics at SURFsara

	Devices	**SNMP error**	**SNMP support**	**LLDP support**
Allowed	95	10	85	74
External parties	29	-	-	29
Total	124	10	85	103

7.2 State of information found in the MIBs

Describing the topology based on the information found in the MIB requires that all devices support the same MIB, or that all possible MIBs are known and can be implemented. Two variants of MIBs can be distinct. First there are the standardized MIBs and second, there are the proprietary MIBs. Both variants are discussed below.

7.2.1 Standardized MIBs

Standardization organizations like Internet Engineering Task Force (IETF) or Institute of Electrical and Electronics Engineers (IEEE) create standards that vendors can choose to support in their devices. Some of the standards are mandatory, but others are optional. Depending on standards they are not always implemented, which could lead to undesired results.

7.2.2 Proprietary MIBs

Some devices only have a limited set of information in the standard MIBs, while the vendor-specific MIB contains more information. More advanced protocols, like PVST, effectively disables the usage of the standard STP BRIDGE-MIB. This situation also leads to the necessity of usage of proprietary MIBs.

7.3 Identifiers

Depending of the object that is needed to identify, determines the characteristics (e.g. the uniqueness) of an identifier. Below, some of the issues when choosing an identifier are discussed.

7.3.1 Chassis ID

When the physical topology is described by using LLDP as its basis, then the chassis ID is meant to be used as the identifier. Within the administrative domain in which the topology is described, it is important that the chassis IDs are unique within the domain. As mentioned in section 3.3, the chassis ID need not be unique. Some vendors may allow for the administrators to change the subtype of the chassis ID, therefore it is up to the administrator to choose an subtype that guarantees uniqueness.

7.3.2 IP address

The MIB used for learning the IP address(es) and next hop address(es) associated with an interface, provide no further information as to which network segment an IP address belongs. For example, when multiple network domains are taken into consideration, it could be the case that in both networks the same network addressing scheme is used. When the relation from device A with device B needs to be made, then knowing only the next hop address of a device does not suffice. Protocols like OSPF provide extra information as to which segment a network address belongs. Without the use of extra information, then effectively the only solution left is to use layer two path-finding. Only then their is certainty that a device does not have a next hop in a network segment to which their is no layer two path.

7.3.3 Interface

For the correlation of different network layers within the topology, the interface index has a central role. Each interface is given a unique number, which can be used as an identifier for an interface on a particular device. RFC 2863 [15] defines how interfaces should be referenced within the device by means of an interface index and the persitency of the index number. Most devices do support persistent interface index numbers, and by adding or removing an interface, the numbering could change. RFC 2863 does however require that the interface index remains the same during operational of a device. Therefore, it is of importance that correlating various layers and protocol information to the interface index is done within the same operational runtime and that the interface index cannot be trusted after re-initialization (e.g. reboot) of the device.

8 Conclusion

In this paper we looked at the challenges with the correlation of physical topology information based on LLDP and logical topology information.

A device configured with LLDP sends its system information to neighbouring devices on the LAN. As long as all devices in the network support LLDP, it is relatively easy to create a correct topology based on this information. Tests on the production network of SURFsara showed that 83% of the devices had LLDP configured. These results demonstrates the maturity and usefulness of LLDP for topology discovery in heterogeneous network environments.

The active Ethernet and IP topologies are relatively straightforward to construct. However, in modern networks, protocols and technologies that do virtualization, aggregation and pruning of links and devices must be taken into account to be able to correctly correlate the different topologies.

In the end, all network topology information can be linked to an interface. The interface index is the central piece of information that is used to correlate all topology layers. In theory, there is nothing that prevents the correlation between topology layers. However, there is a roadblock on the practical side that makes it hard to create a tool that works in an heterogeneous network environment. This roadblock has to do with the extraction of the necessary information from the MIB. Basic information can be gathered from standardized objects but manufactures still use a lot of proprietary objects, even when standardized alternatives exist.

Acknowledgements

We would like to thank Freek Dijkstra and Farhad Davani of SURFsara for their support during our research and their infinite amount of enthusiasm. We are also grateful for the equipment SUFRsara has provided us and the possibility to test our software in their production environment.

References

[1] 802.1AX-2008: Standard for Local and metropolitan area networks–Link Aggregation.

[2] IEEE 802.1D Standard. http://standards.ieee.org/getieee802/download/802.1D-1998.pdf.

[3] ISO/IEC 7498-1:1994: Information technology-Open Systems Interconnection-Basic Reference Model: The Basic Model.

[4] IEEE 802.1q: VLAN, 2005.

[5] Yigal Bejerano. Taking the skeletons out of the closets: a simple and efficient topology discovery scheme for large ethernet lans. *IEEE/ACM Trans. Netw.*, 17(5):1385–1398, October 2009.

[6] Yigal Bejerano, Yuri Breitbart, Minos N. Garofalakis, and Rajeev Rastogi. Physical topology discovery for large multi-subnet networks. In *in Proc. IEEE Infocom*, pages 342–352, 2003.

[7] A. Bierman and K. Jones. Physical Topology MIB. RFC 2922 (Informational), September 2000.

[8] Richard Black, Austin Donnelly, and Cedric Fournet. Ethernet topology discovery without network assistance. In *Proceedings of the 12th IEEE International Conference on Network Protocols*, ICNP '04, pages 328–339, Washington, DC, USA, 2004. IEEE Computer Society.

[9] R. Braden. Requirements for internet hosts - communication layers. RFC 1122 (Standard), October 1989. Updated by RFCs 1349, 4379.

[10] Yuri Breitbart, Minos Garofalakis, Ben Jai, Cliff Martin, Rajeev Rastogi, and Avi Silberschatz. Topology discovery in heterogeneous ip networks: the netinventory system. *IEEE/ACM Trans. Netw.*, 12(3):401–414, June 2004.

[11] Freek Dijkstra, Farhad Davani, Diederik Vandevenne, and Dennis Pellikaan. Discussion, February 2013.

[12] Hassan Gobjuka and Yuri Breitbart. Discovering network topology of large multisubnet ethernet networks. In *Proceedings of the 32nd IEEE Conference on Local Computer Networks*, LCN '07, pages 428–435, Washington, DC, USA, 2007. IEEE Computer Society.

[13] Hassan Gobjuka and Yuri J. Breitbart. Ethernet topology discovery for networks with incomplete information. *IEEE/ACM Trans. Netw.*, 18(4):1220–1233, August 2010.

[14] B. Haberman. IP Forwarding Table MIB. RFC 4292 (Proposed Standard), April 2006.

[15] K. McCloghrie and F. Kastenholz. The Interfaces Group MIB. RFC 2863 (Draft Standard), June 2000.

[16] K. McCloghrie and M. T. Rose. Management information base for network management of tcp/ip-based internets:mib-ii, 1991.

[17] K. Norseth and E. Bell. Definitions of Managed Objects for Bridges. RFC 4188 (Proposed Standard), September 2005.

[18] Suman Pandey, Mi-Jung Choi, Sung-Joo Lee, and James W. Hong. Ip network topology discovery using snmp. In *Proceedings of the 23rd international conference on Information Networking*, ICOIN'09, pages 33–37, Piscataway, NJ, USA, 2009. IEEE Press.

[19] R. Sharma R. Siamwalla and R.keshav. Discovering internet topology. 1999.

[20] IEEE Computer Society. Ieee std. 802.1ab-2009, 2009.

[21] Choonho Son, Junsuk Oh, Kyoung-Ho Lee, Kieung Kim, and Jaehyung Yoo. Efficient physical topology discovery for large ospf networks. In *NOMS*, pages 325–330. IEEE, 2008.

A Acronyms and abbreviations

AFT Address Forwarding Table

ARP Address Resolution Protocol

CDP Cisco Discovery Protocol

ERD Entity Relationship Diagram

FDP Foundry Discovery Protocol

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

IP Internet Protocol

LAN local area network

LLDPDU LLDP Data Unit

LLDP Link Layer Discovery Protocol

MAC media access control

MIB management information base

MSAP MAC service access point

MSTP Multiple Spanning Tree Protocol

NDP Nortel Discovery Protocol

OID object identifier

OSI Open Systems Interconnection

OSPF Open Shortest Path First

PoC proof of concept

PTOPO-MIB Physical Topology Management Information Base

PVST+ Per-VLAN Spanning Tree Plus

PVST Per-VLAN Spanning Tree

RSTP Rapid Spanning Tree Protocol

RVI Routed VLAN Interface

SNMP Simple Network Management Protocol

SQL Structured Query Language

STP Spanning Tree Protocol

SVI Switch Virtual Interface

TCP/IP Transmission Control Protocol/Internet Protocol

VLAN Virtual Local Area Network

VRF Virtual Routing and Forwarding

VSTP VLAN Spanning Tree Protocol

WDM Wavelength-division multiplexing

B MIB objects

Table 16: Overview of MIB objects used

MIB name	**Object identifier**	**Object name**
IF-MIB	1.3.6.1.2.1.2.2.1.1	ifIndex
	1.3.6.1.2.1.31.1.1.1.1	ifName
	1.3.6.1.2.1.2.2.1.2	ifDescr
	1.3.6.1.2.1.2.2.1.6	ifPhysAddress
LLDP-MIB	1.0.8802.1.1.2.1.3.2.0	lldpLocChassisId
	1.0.8802.1.1.2.1.3.3.0	lldpLocSysName
	1.0.8802.1.1.2.1.3.4.0	lldpLocSysDesc
	1.0.8802.1.1.2.1.3.7.1.3	lldpLocPortId
	1.0.8802.1.1.2.1.3.7.1.4	lldpLocPortDesc
	1.0.8802.1.1.2.1.3.6.0	lldpLocSysCapEnabled
	1.0.8802.1.1.2.1.4.1.1.7	lldpRemPortId
	1.0.8802.1.1.2.1.4.1.1.8	lldpRemPortDesc
	1.0.8802.1.1.2.1.4.1.1.5	lldpRemChassisId
	1.0.8802.1.1.2.1.4.2.1.4	lldpRemManAddr
BRIDGE-MIB	1.3.6.1.2.1.17.2.15.1.3	dot1dStpPortState
	1.3.6.1.2.1.17.2.5.0	dot1dStpDesignatedRoot
	1.3.6.1.2.1.17.1.4.1.2	dot1dBasePortIfIndex
IP-MIB	1.3.6.1.2.1.4.20.1.1	ipAdEntAddr
	1.3.6.1.2.1.4.20.1.2	ipAdEntIfIndex
	1.3.6.1.2.1.4.20.1.3	ipAdEntNetmask
IP-FORWARD-MIB	1.3.6.1.2.1.4.24.4.1.4	ipCidrRouteNextHop
	1.3.6.1.2.1.4.24.4.1.5	ipCidrRouteIfIndex
	1.3.6.1.2.1.4.24.4.1.6	ipCidrRouterType
SBE-MIB	1.3.6.1.2.1.4.1.0	ipConfiguredFlag