D1.2 OpenFlow

December 2011
Version 1.1
Ronald van der Pol

SARA

1 Introduction

OpenFlow [1] is a new network technology and an example of Software Defined Networking (SDN) which could revolutionize the way we do networking in the near future. In SDN the control plane and data plane of network equipment is separated. The control plane is run externally on commodity servers and the network equipment is only responsible for data plane forwarding. The forwarding tables are programmed via the OpenFlow API.

OpenFlow was developed several years ago at Stanford University after concluding that networks had become a critical infrastructure and network innovation on that same infrastructure was hampered more and more. As a solution the idea of virtualizing the network in a production part and an experimental part was proposed. This led to the OpenFlow project.

Section 2 is an overview of OpenFlow. Section 3 explains the concept of *network virtualization*. Section 4 gives the status of OpenFlow developments as of December 2011. The equipent present at the LigthHouse OpenFlow testbed is decribed in section 5. Section 6 described the OpenFlow demo at SC11 which was held in November 2011 in Seattle. Finally, some conclusions and what OpenFlow means for SURFnet and the connected organizations is given in section 7.

2 Overview of OpenFlow

OpenFlow is an open standard for Software Defined Networking (SDN) in which the control plane and data plane are separated. OpenFlow switches perform the dataplane function and OpenFlow controllers implement the control plane intelligence and communicate with the OpenFlow switch via the OpenFlow protocol (see figure 1).

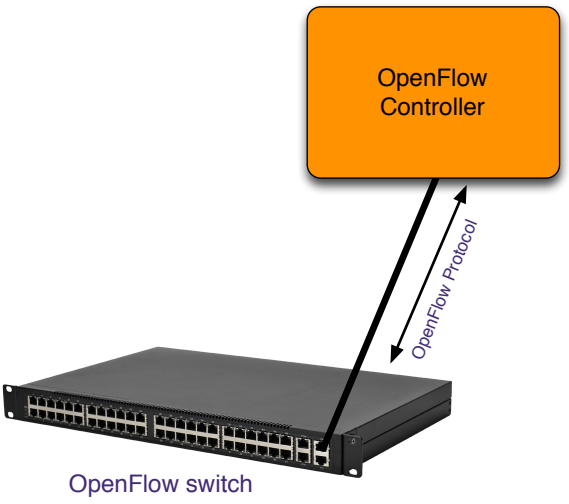


Fig. 1. OpenFlow Switch and Controller

The OpenFlow infrastructure consists of OpenFlow switches and external OpenFlow controllers. These controllers are software daemons running on servers and communicating with OpenFlow switches via the OpenFlow API. The flow-tables[1] of these OpenFlow switches can be programmed via the OpenFlow API. OpenFlow is separating the data and control plane, where the intelligence is implemented in the OpenFlow controllers. These controllers can run on much more powerful hardware than what is available on routers or switches. The switches can be relatively simple and cheap because they only need to do the forwarding.

2.1 Flow Matching

The OpenFlow API allows for the insertion, modification and deletion of flow entries in the switch. These flow entries are similar to Access Control Lists and consist of a match and a corresponding action. Currently, most switches and controllers implement version 1.0.0 [3] of the OpenFlow specification. In this version packets are matched according to any combination the following fields:

- ingress port
- Ethernet source/destination address
- Ethertype
- VLAN ID
- VLAN priority
- IPv4 source/destination address
- IPv4 protocol
- IPv4 ToS
- TCP/UDP source/destination port

Wildcard matching for a field is supported, which means that any value for that field results in a match. If a packet matches none of the entries, the packet is encapsulated and sent to the controller. The current version has no support for IPv6. For each flow match there are zero or more actions associated with it. If a match has no action associated with it, the packet is dropped. A match can result is multiple packets if there is more than one forwarding action associated with that match. Every switch must implement the following actions:

Required Actions	Description
Forward - PORT	Forward packet to physical port.
Forward - ALL	Send packet to all interfaces, including the incoming interface.
Forward - CONTROLLER	Encapsulate and send the packet to the controller.
Forward - LOCAL	Send the packet to the local networking stack.
Forward - TABLE	Send packet-out message according to flow table actions.
Forward - IN-PORT	Send packet out on the input port.
Drop	Implemented as a flow match with no action.

The following actions are optional:

Optional Actions	Description
Forward - NORMAL	Process packet using traditional forwarding path of switch.
Forward - FLOOD	Flood along the spanning tree, not including incoming interface.
Enqueue	Forward through queue attached to a port.
Modify VLAN ID	Replace or add VLAN tag
Modify VLAN Priority	Replace priority field
Strip VLAN tag	Strip VLAN tag
Modify Ethernet address	Replace source/destination Ethernet address
Modify IPv4 address	Replace source/destination IPv4 address
Modify ToS bits	Replace IPv4 ToS field
Modify TCP/UDP port	Replace TCP/UDP source/destination port

[1] Usually a TCAM (Ternary Content Addressable Memory) which is used in traditional Ethernet switches as MAC forwarding table

2.2 OpenFlow Protocol

The OpenFlow protocol between controller and switch consists of three type of messages: contoller to switch (C), asynchronous (A) and symmetric (S). The messages are described in the table below:

Message	Type	Description
Features	C	Request for features supported by switch.
Configuration	C	Request for configuration parameters.
Modify-State	C	Add/modify/delete flow entries or port properties.
Read-State	C	Collect statistics.
Send-Packet	C	Send packets out to a port of the switch.
Barrier	C	Switch sends reply when it has finished all outstanding requests.
Packet-In	A	Data packet was sent from switch to controller.
Flow-Removed	A	Flow entry has expired.
Port-Status	A	Port up/down transition.
Error	A	Error message from switch to controller.
Hello	S	Hello exchange upon connection startup.
Echo	S	Liveness request/reply initiated by switch or controller.
Vendor	S	Used for future additional functionality.

3 Network Virtualization

OpenFlow and SDN are often promoted in combination with network virtualization. In an OpenFlow network there can be one controller per switch or a controller can be attached to multiple switches so that it can manage a whole network. An OpenFlow switch can also be virtualized and each virtual switch (or group of virtual switches) can be managed by its own controller. With OpenFlow networks can be virtualized by giving each application part of an OpenFlow switch (e.g. a couple of ports) and part of the flowtable space. Multiple virtual switches with links between them form what is called a *slice* of the network. In this way the network can be divided into multiple slices (see figure 2). One slice can be used for production traffic, another slice for a

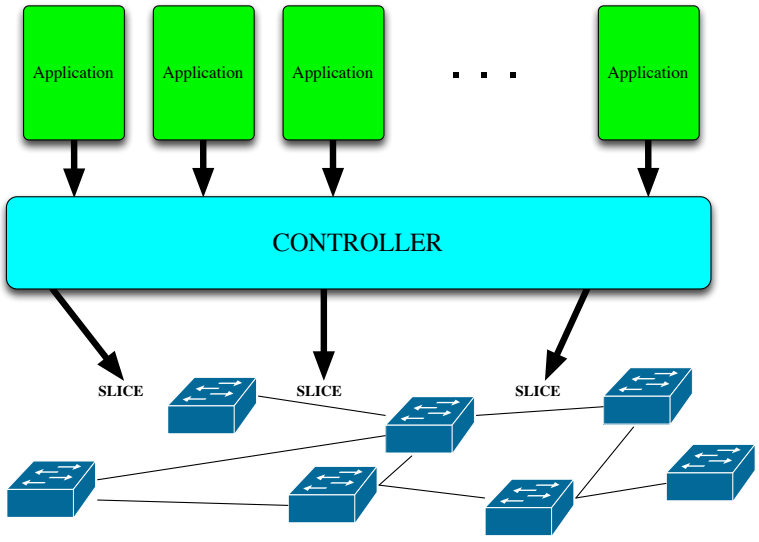


Fig. 2. Creating virtual network slices with OpenFlow

new experimental protocol. This makes network innovation possible again on real hardware with linerate forwarding instead of having to use slow software routers/switches or simulations.

This is the main concept of OpenFlow and SDN: the configuration of the switches is done by software instead of manually by a network administrator. This is far less error-prone, especially when dealing with many switches. The applications only see their part of the physical network. This is similar to virtual machines on a server. The combination of virtual machines connected via their own network slice provides a nice virtual infrastructure for the applications.

The open source FlowVisor [2] is an example of such a controller. It sits between OpenFlow controllers and the switches and takes care that each controller has access only to its own slice. The flow table of the switches is strictly subdivided between the controllers too.

Big Switch Networks is a company founded in 2010 by Guido Appenzeller and Kyle Forster. Guido was the head of the Clean Slate Lab at Stanford University. The initial focus of Big Switch is controller software for SDN like depicted in figure 2. In December 2011 the Floodlight [5] controller was released. It is a fork of the Beacon controller with an Apache license and forms the foundation of a commercial (distributed) controller from Big Switch Networks.

4 Current Status

OpenFlow got a huge burst in attention in 2011 and there was a lot of press coverage. It even became difficult to distinguish between reality and hype. There were also several major announcements during 2011: the Open Networking Foundation [4] was set up, Internet2, Indiana University and the Clean Slate Program at Stanford University announced a nation wide OpenFlow production network (NDDI) [6] and several network equipment vendors announced OpenFlow switches.

In March 2011 the nonprofit Open Networking Foundation was set up. The six founding members are Deutsche Telekom, Facebook, Google, Microsoft, Verizon, and Yahoo! At the end of 2011 more than 40 companies had joined, including all major network equipment vendors. A three day summit was organized in October 2011 at Stanford University.

Several OpenFlow networks are deployed or will be available soon. In April 2011 Internet2, Indiana University and the Clean Slate Program at Stanford University announced the NDDI project in which a nation wide OpenFlow network will be build. There are already several large OpenFlow testbeds in the world (like GENI [7] in the USA, JGN-X [8] in Japan and Ofelia [9] in Europe), but NDDI will be the first nation wide OpenFlow production network. Deployment will start in 2012.

Various vendors offer OpenFlow firmware images for their switches. Some are only experimental, others are fully supported production images. Vendors that have production images are NEC [10] with their ProgrammableFlow line of switches, IBM has the BNT RackSwitch G8264 with support for OpenFlow and Pronto Systems [12], a Stanford University startup, sells several OpenFlow switches. OpenFlow was also ported to the 4x 1GE version of the NetFPGA card [13], which was developed by a group at Stanford University. This is a PCI card that can be installed in a server. The card can do hardware forwarding between the four 1GE ports. HP spports OpenFlow on the ProCurve 5400 and 6600 series, but it is for reseacrh only. The image (K.15.05.5001) can be downloaded by customers from their support website. Juniper Networks [14] has added support for OpenFlow in their JunOS SDK. Finally, Cisco announced OpenFlow support for their Nexus switches, starting with the Nexus 3000, but no date has been set yet. The following table gives an overview of the production switches:

Vendor	Type	Description	Price (USD)
NEC	PF5240	48x 1GE + 4x 10GE SFP+	unknown
NEC	PF5820	48x 10GE SFP+ + 4x 40GE	unknown
IBM	G8264	48x 10GE + 4x 40GE	30,000
Pronto Systems	3290	48x 1GE + 4x 10GE SFP+	2,750
Pronto Systems	3780	48x 10GE SFP+	9,500
Pronto Systems	3920	48x 10GE SFP+ + 4x 40GE	11,500
Stanford University	NetFPGA	4x 1GE	599

5 LightHouse OpenFlow Testbed

There is a small OpenFlow testbed in LightHouse at SARA consisting of the following equipment:

- Pronto 3290
- Pronto 3290
- server with 4x 1GE NetFPGA card
- server with 4x 1GE NetFPGA card
- server with NOX controller

There are three OpenFlow images available for the Pronto switches: Indigo [15], Pica8 [16] and Open vSwitch [17]. The Pica8 software was used during our demo (see section 6), but Pronto Systems has indicated that future development will be done with the Open vSwitch software, so we will switch to this in 2012. As OpenFlow contoller we used NOX [18], mainly because it supports C++ and the IEEE 802.1ag implementation is written in C. Floodlight seems like a promising development on the controller front and will be investigated in 2012.

6 SC11 OpenFlow Demo

OpenFlow can also be used to add protocol support to OpenFlow switches. This is especially useful for control and management protocols that do not need hardware forwarding. Once such a protocol is implemented in an OpenFlow controller, every switch that supports OpenFlow has now support for that protocol via its controller. We have demonstrated this concept for IEEE 802.1ag.

The IEEE 802.1ag [19] standard is a protocol for Ethernet Connectivity Fault Management (CFM). It is a strictly layer 2 protocol and uses only Ethernet MAC addresses and no IP addresses. It offers three types of messages:

- *Continuity Check (CCM)*
  - Detect loss of connectivity
  - Periodic hello messages
  - Sent as multicast Ethernet frames by switch interfaces
  - Interfaces process information, but do not send replies
- *Loopback Message/Reply (LBM/LBR)*
  - Check for reachability
  - Ping to MAC address
  - Sent as unicast Ethernet frames via CLI; interface replies with unicast
  - Similar to IP ping, but at layer 2
- *Link Trace Message/Reply (LTM/LTR)*
  - Path information
  - Traceroute to MAC address
  - Response from MAC address of interfaces in the path
  - Sent as Ethernet multicast frames via CLI; interface replies with unicast
  - Similar to IP traceroute, but at layer 2

This protocol was implemented at SARA [20] and was integrated to the NOX OpenFlow controller for a demo at the SC11 High Performance Computing and Networking conference from 12-18 November 2011 in Seattle [21] [22] [23] [24] [25]. Figure 3 shows the integration of the 802.1ag software in the NOX OpenFlow controller and how it is sending and receiving 802.1ag frames. The frames are transported over the dataplane of the OpenFlow switches. In the demo a large international OpenFlow testbed was set up between Amsterdam, Chicago, Ottawa and Seattle. The links between the sites were monitored using periodic CCM frames. The status of the links was stored in a round-robin database. This status information was published using perfSONAR measurement points and a collector showed the overall status of the network live on a website. It was chosen by the SC11 organization among the top six out of eleven demos.

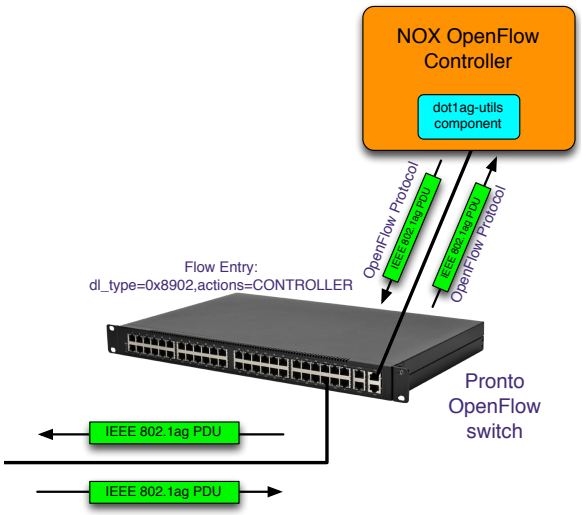


Fig. 3. Integration of 802.1ag in NOX OpenFlow Contoller

7 Conclusion

OpenFlow (or better: software defined networking) has the potential of becoming a disruptive technology in networking. Many network researchers are working on SDN and there are several large OpenFlow networks deployed, even nation wide. Some of these networks are experimental, some are used as production networks (such as the Internet2 NDDI network and networks at the Stanford University campus). Also, all the major network equipment vendors and large companies like Facebook, Google and Yahoo! have jumped on the OpenFlow bandwagon.

On the other hand, OpenFlow is still far from production ready for the average network manager. Production equipment is scarce and documentation is lacking or out of date. The status of various project is also unclear with much speculation about future support. Many see big money in OpenFlow and SDN and several key developers at Stanford University have joined new startup companies leaving the open source projects at Stanford University with few developers left. E.g., there was a rumour that development on the NOX controller had stopped, but this was denied several weeks later. Time will tell. Also, the future support for flowvisor is unclear and the current version still has many bugs. The Beacon contoller has forked in a version maintained by Stanford University with a GPLv2 license and a version maintained by Big Switch Networks with an Apache license.

The standardization of OpenFlow is now taken over by the Open Networking Foundation (ONF). When the standardization was done at Stanford University there was a public mailing list for discussions about the next standard. Currently, the discussions are on a closed mailing list for ONF members only and it costs USD 30,000 per year to become an ONF member.

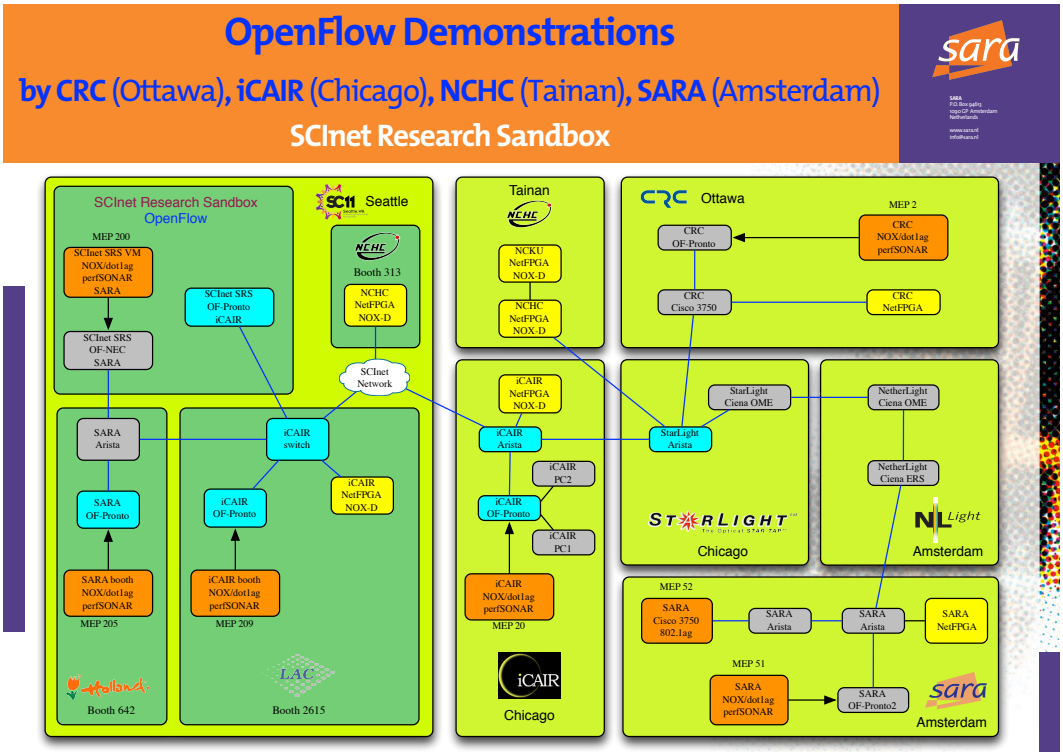
But many people think that SDN is the future of networking. OpenFlow is currently the best bet because it is standardized and supported by several hardware switches. For SURFnet and its connecting organizations it is important to follow the developments with respect to SDN and investigate how it can improve current day networking. This is similar to following developments in multicast, IPv6 and DNSSEC in the past. In 2012 SURFnet will setup a testbed and interested organizations can use it to get familiar with OpenFlow features.

References

1. McKeown, N., Anderson, T., Balakrishnan, H., Purulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.
OpenFlow: Enabling Innovation in Campus Networks

White Paper, March 2008
http://www.openflow.org/documents/openflow-wp-latest.pdf
2. The FlowVisor Project
https://openflow.stanford.edu/display/DOCS/Flowvisor
3. OpenFlow Switch Specification
Version 1.0.0 (Wire Protocol 0x01)
December, 2009
http://www.openflow.org/documents/openflow-spec-v1.0.0.pdf
4. Open Networking Foundation
http://www.opennetworking.org/
5. The Floodlight OpenFlow controller
http://floodlight.openflowhub.org/
6. Network Development and Deployment Initiative (NDDI)
http://www.internet2.edu/network/ose/
7. Global Environment for Network Innovations (GENI)
http://www.geni.net/
8. Japan Gigabit Network eXtreme (JGN-X)
http://www.jgn.nict.go.jp/english/info/technologies/openflow.html
9. OpenFlow in Europe: Linking Infrastructure and Applications
http://www.fp7-ofelia.eu/
10. NEC ProgrammableFlow OpenFlow switches
http://www.necam.com/PFlow/
11. IBM BNT RackSwitch G8264
http://www-03.ibm.com/systems/x/options/networking/bnt8264/index.html
12. Pronto OpenFlow switches
http://www.prontosys.net/
13. NetFPGA PCI card
http://netfpga.org/
14. Juniper networks delivers openflow application to enable network programmability and flexibility for customers
http://bit.ly/sWMmaP
15. Indigo - OpenFlow for Hardware Switches
http://www.openflowhub.org/display/Indigo
16. Pica8 OpenFlow firmware
http://www.openflowhub.org/display/Indigo
17. Open vSwitch - an Open Virtual Switch
http://openvswitch.org/
18. NOX - An OpenFlow Controller
http://noxrepo.org/
19. IEEE 802.1ag-2007 - Connectivity Fault Management
http://standards.ieee.org/findstds/standard/802.1ag-2007.html
20. van der Pol, R. Open Source Implementation of the IEEE 802.1ag standard
http://nrg.sara.nl/dot1ag-utils
21. van der Pol, R.
Monitoring and Troubleshooting OpenFlow Slices with an Open Source Implementation of IEEE 802.1ag
SCinet Research Sandbox Experiment Results
SC11 Technical Program
http://sc11.supercomputing.org/schedule/event_detail.php?evid=rsand109 http://nrg.sara.nl/presentations/SC11-SRS-8021ag.pdf
22. van der Pol, R., Boele, S., Dijkstra, F.
OpenFlow Demonstrations by CRC, iCAIR, NHCH and SARA
Poster at SC11, Nov 12-18, 2011, Seattle, USA
http://nrg.sara.nl/presentations/SC11-SRS-8021ag.pdf
23. van der Pol, R., Boele, S., Dijkstra, F.
OpenFlow Demo, IEEE 802.1ag Ethernet OAM
SCInet Research Sandbox
Poster at SC11, Nov 12-18, 2011, Seattle, USA
http://nrg.sara.nl/posters/SC11-openflow.pdf

24. van der Pol, R., Boele, S., Dijkstra, F.
OpenFlow Demo, PerfSONAR Control Plane
Poster at SC11, Nov 12-18, 2011, Seattle, USA
`http://nrg.sara.nl/posters/SC11-controlplane-topology.pdf`
25. van der Pol, R., Boele, S., Dijkstra, F.
OpenFlow Demo, Monitoring of Ethernet OAM
Poster at SC11, Nov 12-18, 2011, Seattle, USA
`http://nrg.sara.nl/posters/SC11-information-flow.pdf`



OpenFlow Demo

IEEE 802.1ag Ethernet OAM
SCInet Research Sandbox

Ronald van der Pol, Sander Boele, Freek Dijkstra

SARA
P.O. Box 94613
1090 GP Amsterdam
Netherlands

www.sara.nl
info@sara.nl

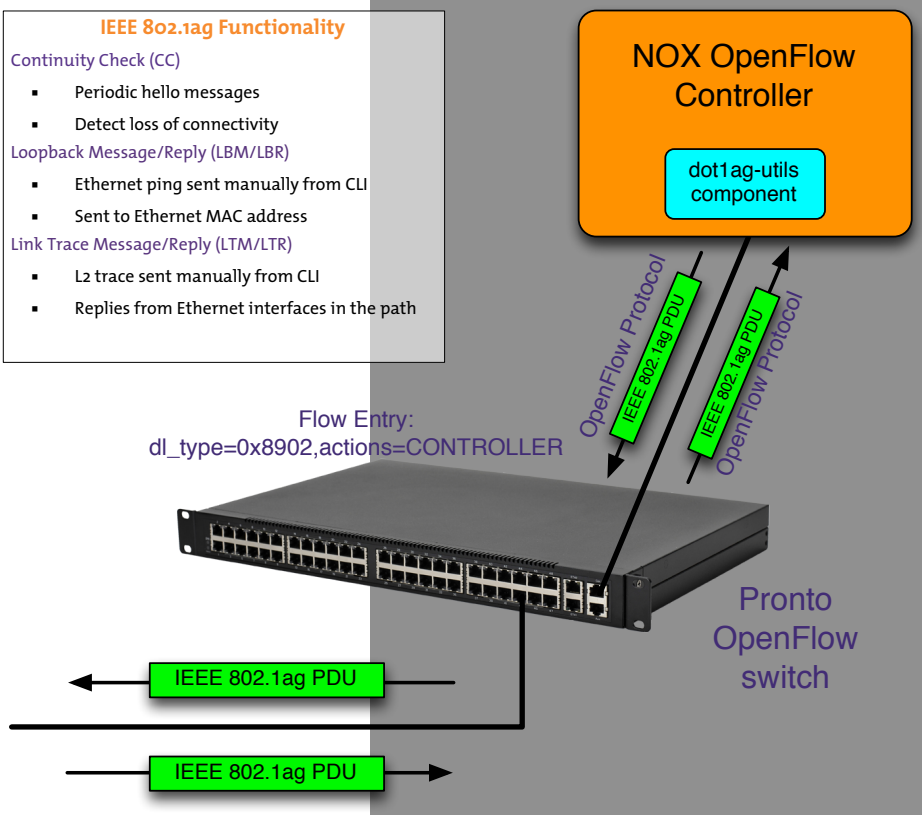
This demo shows how OpenFlow can be used by end-users to easily add new network protocols to OpenFlow switches. SARA implemented the IEEE 802.1ag standard, which is a protocol for Ethernet OAM (Operations, Administration, and Maintenance). The code was added to the NOX OpenFlow controller so that every OpenFlow switch can now support 802.1ag. The detection of link failures with this setup is demonstrated in a multi-domain Ethernet network with sites in Amsterdam, Chicago, Ottawa and Seattle. The 802.1ag implementation is available as open source (BSD license) at http://nrg.sara.nl/dot1ag-utils.

IEEE 802.1ag Functionality

Continuity Check (CC)
- Periodic hello messages
- Detect loss of connectivity

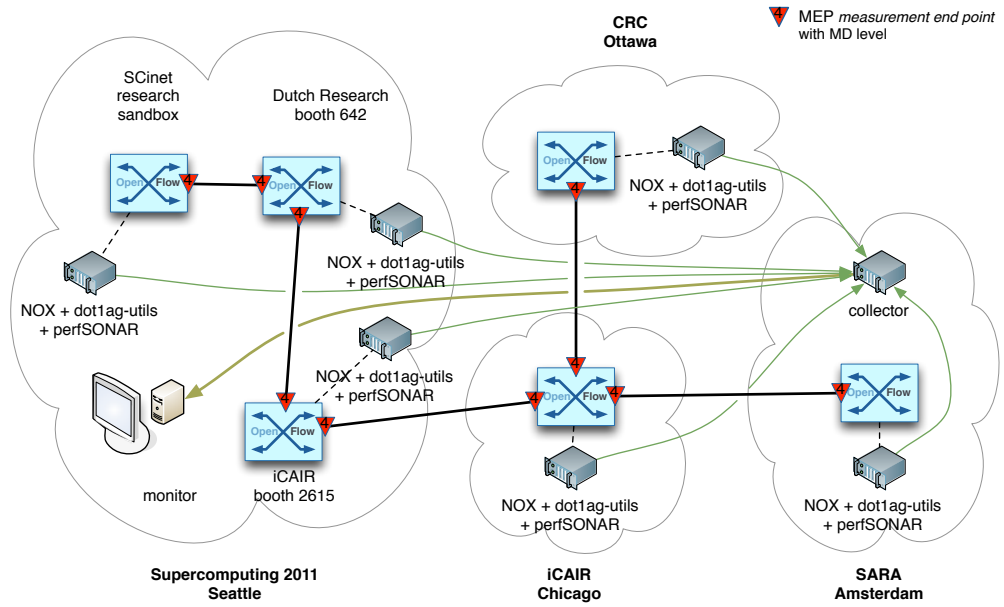
Loopback Message/Reply (LBM/LBR)
- Ethernet ping sent manually from CLI
- Sent to Ethernet MAC address

Link Trace Message/Reply (LTM/LTR)
- L2 trace sent manually from CLI
- Replies from Ethernet interfaces in the path



Sponsored by: SURFNET Gigaport 3

Ethernet Sandbox



Monitoring of Ethernet OAM