פרויקטים בתקשורת מחשבים - 236340 - סמסטר אביב 2016

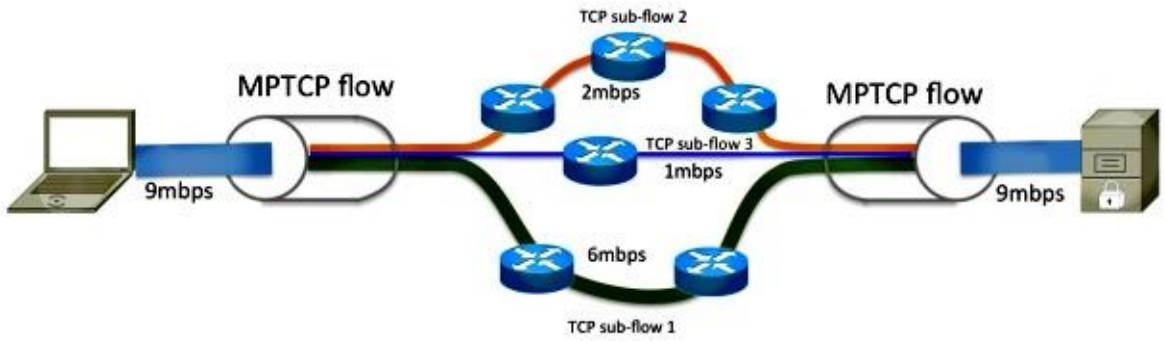
MultiPath TCP (MPTCP)

Abstract:

MultiPath TCP (MPTCP) is an ongoing effort of the Internet Engineering Task Force's (IETF) Multipath TCP working group, which aims at allowing a Transmission Control Protocol (TCP) connection to use multiple paths to maximize TCP traffic throughput and increase redundancy.

MPTCP is useful particularly in wireless networks such as Wi-Fi and mobile networks. In addition, data centers can gain performance and redundancy benefits of such implementation.

This TCP extension is currently implemented in various OSs such us: Linux Kernel, Android, Apple IOS and more.



Goals:

Implement MPTCP on ns-3 (Network Simulator open source project). You will need to demo improved TCP behavior in ideal environment with no packet loss, congestion cases as well as connections flapping. In addition – compare results with 1 TCP connection using the whole bandwidth.

Requirements:

C++, Internet Networking Course

Robust Header Compression (ROHC)

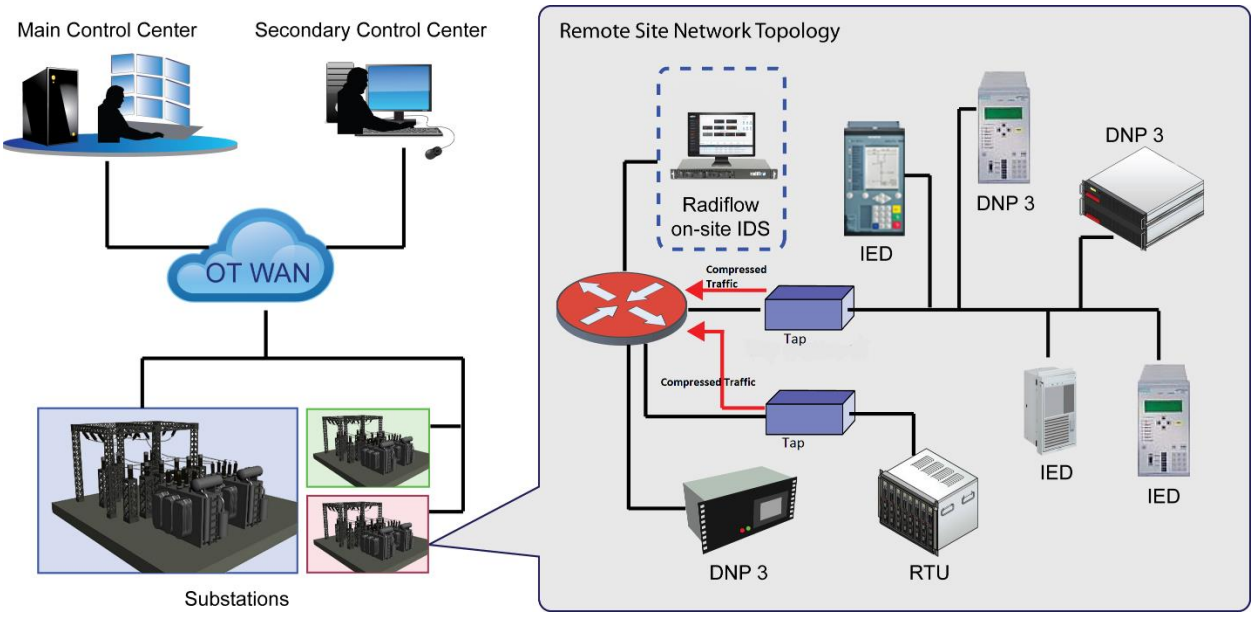
Abstract:

Deploying Intrusion Detection Systems (IDS) requires copying the network traffic from several sensing points in the network using Tap device and sending the traffic to the IDS server. In Supervisory Control And Data Acquisition (SCADA) system, this mission can be challenging, due to the lower bandwidth of the communication channels on the remote site between the Tap device and the IDS Server.

A solution for reducing the bandwidth is to compress the sniffed traffic before sending it to IDS server. The traffic will be then decompressed and forwarded to the analyzing application running on the IDS server.

The compression is done both on the IP header and on the payload, using standard lossless compression method for the payload and RObust Header Compression (ROHC) for the packet header.

.



Goals:

Implement ROHC on a Client-Server topology or on Network Simulator (ns-3 or Mininet) and test it with both UDP and TCP traffic showing that traffic bandwidth is reduced and not corrupted due to the Compress-Decompress. In addition – need to demonstrate correct handling cases of traffic loss.

Guided by:

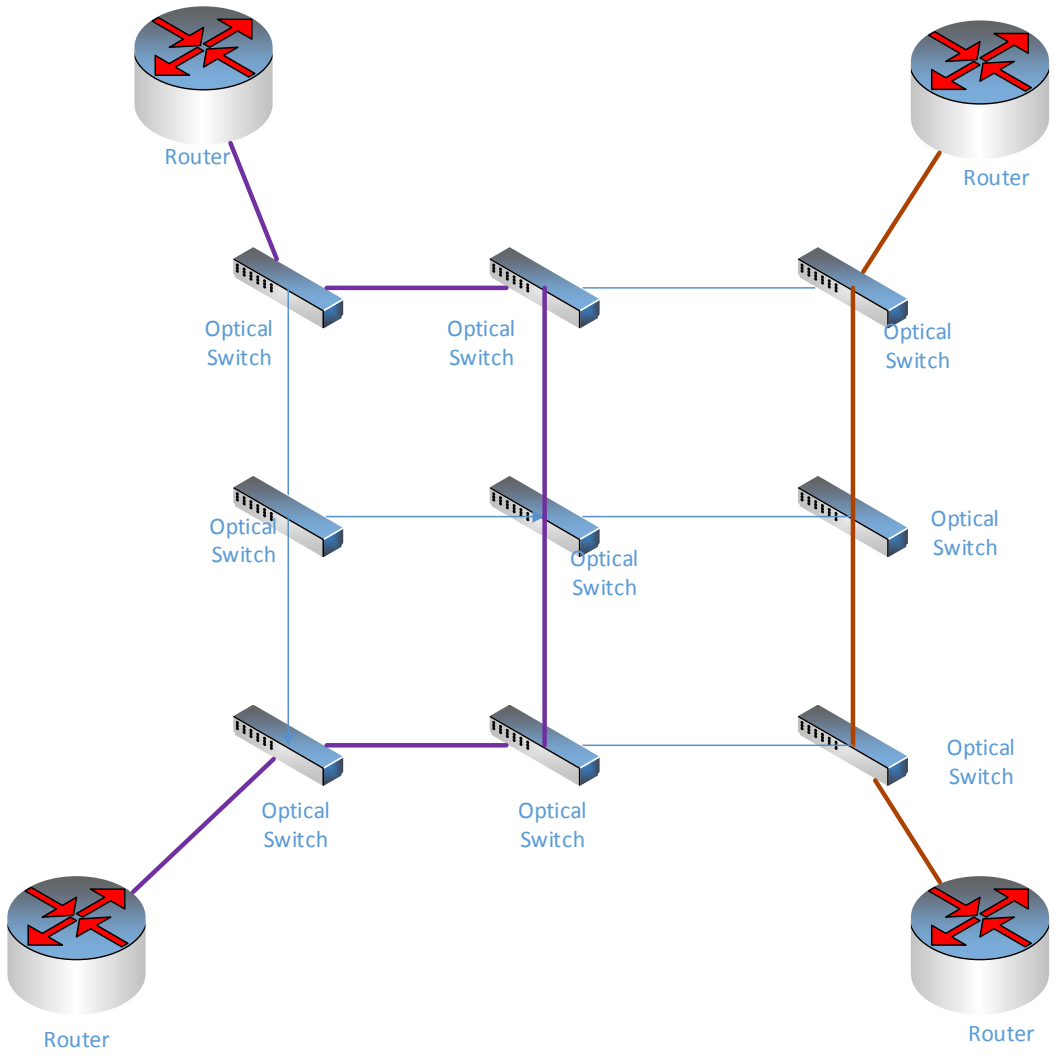
radiflow
Secure Your Assets

Restorable Logical Topology using Cross-Layer Optimization

Abstract:

Today's communication networks consist of routers and optical switches in a logical topology overlaid on an optical physical infrastructure. The routers are connected to each other via logical links called LightPaths, where each LightPath is established over one or more optical fibers and optical switches connecting these fibers.

The design of LightPaths involves selection of both logical link and logical path, i.e decision which pairs of routers will be connected by the LightPaths and how to route each LightPath across the optical network. Such design requires cross-layer optimization that will maximize the end-to-end traffic throughput in case of a physical failure in order to guarantee the restorability of the logical topology.



Goals:

For a given network topology of routers and optical switches – build the topology using network simulator (ns-3 or Mininet) and implement cross layer optimization algorithm that will find both the shortest path between each of the routers and will ensure restorable topology in case of a link failure.

Requirements:

C++ for ns-3

Python for Mininet

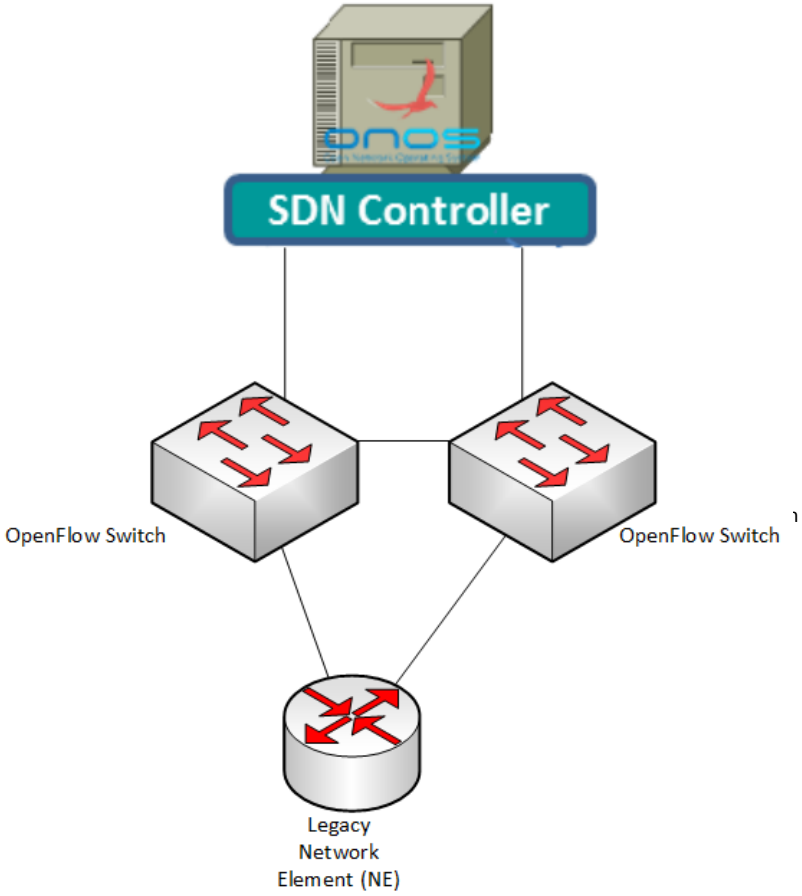
.

Fault Management (FM) in Open Network Operating System (ONOS)

Abstract:

Open Network Operating System (ONOS) is an innovative and rapidly growing open source project that already enable service providers to build real Software Defined Networks (SDN). The deployment of SDN will require ONOS to manage legacy Network Element (NE) in an Hybrid environment.

One of the required management capabilities from ONOS is to support Fault Management in the NE. When a fault or event occurs, the NE typically sends a notification to the network operator via Simple Network Management Protocol (SNMP). The network operator may also (or alternatively) poll the NE to retrieve this information. An alarm is a persistent indication of a fault that clears only when the triggering condition has been resolved.



Goals:

ONOS will provide support for such Fault Alarms. It requires implementation in the ONOS project of basic SNMP V2 functionality (both polling and traps handling), as well as Fault Management application that handles the NE's alarm states.

Perquisite Requirements:

Java