ACTOR: Alarm Correlation and Ticketing for Open ROADM

Tianliang Zhang*†, Wendell Liu†, Balagangadhar Bathula†‡, and Andrea Fumagalli*
* Open Networking Advanced Research (OpNeAR) Lab, UT Dallas, TX, USA
† AT&T Labs, 200 Laurel Avenue South, Middletown, NJ, USA
‡ bb4341@att.com

***Abstract*—The Open ROADM ecosystem enables greater flexibility in deploying optical networks through centralized SDN management and intelligent service orchestration. However, since the maintenance signaling is yet to be standardized and implemented within each device, it is difficult to identify the root cause of issues and manage the network effectively when faults propagate in the network. To tackle this problem, this paper presents a proof-of-concept implementation of an alarm correlation and ticketing system for the multi-vendor Open ROADM ecosystem. The proposed system uses a graph-based method to identify the root cause of alarms and generate tickets for network operations teams. The results of our laboratory tests demonstrate the effectiveness of the proposed system in managing alarms in the multi-vendor Open ROADM ecosystem.**

***Index Terms*—Open ROADM, multi-vendor, SDN, alarm correlation, graph-based**

I. Introduction

The Open ROADM multi-source agreement (MSA) [1] defines YANG data models to represent the Reconfigurable Optical Add/Drop Multiplexer (ROADM) and transponder devices in a multi-vendor optical transport network. Interoperability specifications are defined to overcome the vendor lock-in problem in hardware, software, control, and management. The optical transport network composed of Open ROADM compliant devices can therefore be deployed and controlled by an open source SDN controller. On the one hand, the typical functionalities offered by an Open ROADM compliant SDN controller (e.g., TransportPCE [2]) include device configuration, service provisioning, and alarm/fault and performance monitoring (PM) through its south-bound non-proprietary APIs. On the other, the interaction with a hierarchical controller (or orchestrator) and other external applications (e.g., user interface, network operational platform [3]) through its non-proprietary north-bound APIs increases network awareness and resource utilization effectiveness.

As any transport network, an Open ROADM network requires a fault management and ticketing system. Open ROADM devices are designed to detect faults or anomalies and report them to the fault management system via alarm notification. A single fault typically generates multiple alarm notifications in the network due to a cascade effect involving a multitude of physical components, which are all concurrently affected by the same root cause, defined as the highest-level cause that sets in motion the entire cause-and-effect reaction, which ultimately leads to the alarm notification(s). Identifying in real-time the root cause of many concurrent alarm notifications is particularly challenging in disaggregated solutions such as Open ROADM.

Consider for instance a conventional single-vendor integrated transport system. In this integrated system each device performs a significant amount of alarm correlation with reporting suppression via hierarchical relationship within the device and probable cause transformation via maintenance signaling (e.g., payload mismatch indication or PMI, forward defect indication or FDI, backward defect indication or BDI [4]) at the network level. Since the alarm behavior is consistent and predictable, known a-priory alarm patterns can be used to isolate the root cause.

In contrast, each network element/device (e.g., xponder, ROADM) in an Open ROADM system is disaggregated from the others and so are its functions. Consequently, (proprietary) maintenance signaling between these elements is disabled. For example, the OTN photonic layer maintenance signaling is disabled between ROADM devices to enable ROADM line system interoperability between different vendors by means of the Open ROADM device data plane object model [1]. While on the one hand, each element is still expected to conform to the maintenance signaling insertion and detection — as specified by the relevant standards — on the other the modality in which these maintenance signaling events are reported is not subject to standardization and varies from vendor to vendor. To complicate matters further, the extent of alarm correlation and suppression achieved by each network element through resource hierarchical relationship is implementation specific. The fault management system of an Open ROADM network must therefore tolerate subtle differences in implementation among different vendors. Although alarm correlation in a partially disaggregated multi-vendor optical network has been addressed in [5], this problem still needs to be addressed in a fully disaggregated optical network.

This paper presents for the first time a proof-of-concept implementation of an alarm correlation and ticketing system for a multi-vendor optical network based on Open ROADM MSA. A two-dimensional graph-based method is used in the correlation engine to model the devices and network layout and subsequently generate the root cause of the alarm(s) along with recommended corrective actions. The graph is easy to create and maintain thanks to the standardization of Open ROADM MSA and the method is easy to implement

without complex mathematical computation. The concept is experimentally verified over an Open ROADM testbed in a validation lab.

II. Graph-based Service Model

Each direction (A→Z and Z→A) of a service in the network is assigned a two-dimensional directed graph, wherein the hierarchical dimension models the containment relationship between the service and underlying resources, including higher order trails (example in Fig. 1), while the horizontal dimension models the service/link trail (example in Fig. 2). In this paper, the terms "node", "link", and "connection" refer to physical or logical entities in a network, while "arc" and "vertex" are used to describe elements of a graph.

A. Data Acquisition

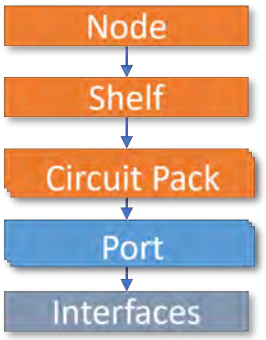
Open ROADM network elements (NE) are monitored by an SDN controller in real-time (TransportPCE is used as the controller of the Open ROADM testbed in this work). TransportPCE aggregates Open ROADM NE's operational data, performance monitoring (PMs), and alarms through NETCONF protocol/API, and keeps the aggregated datastore up-to-date by subscribing to the NE's *openroadm* and *netconf* change and alarm-notification streams. Once connected to a NE, TransportPCE automatically creates and saves the port mapping of logical connection points to external physical ports in the datastore. Network information with respect to various links between NEs and inside each NE is also stored in the TransportPCE datastore.

B. Hierarchical Dimension

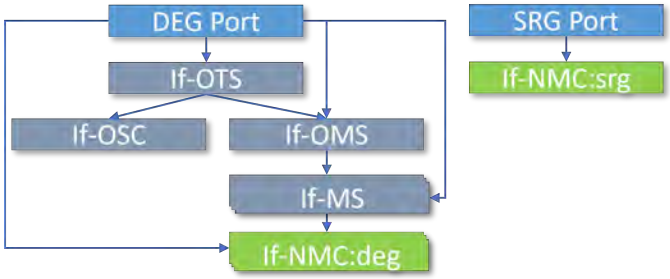
As the Open ROADM device model is clearly defined in the MSA and its operational data is retrieved by TransportPCE, hierarchical relationships between a *containing entity* and one (or more) *contained entity (entities)* in a NE are easily represented in a tree-like graph, as shown in Fig. 1. Examples include a node *containing* shelves, a shelf *containing* circuit pack(s), a circuit pack *containing* port(s), a port *containing* other ports, a port *containing* interface(s), an interface *containing* other interfaces, etc. A failing contained resource should not cause faults in the containing resource, e.g., port state does not cause circuit-pack faults, interface does not cause port faults. In other words, the fault always propagates from the containing resource to the contained resource, not the other way around.

C. Horizontal Dimension

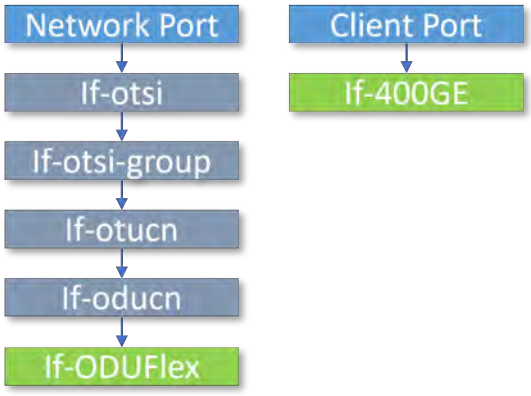
The horizontal dimension of the graph represents *supporting relationships* between a *supported* logical circuit and an ordered list of *supporting* ports, interfaces, logical connections, and internal/physical/external links at that signal layer. Directed arcs in the graph are traced and created from the service routing list of all NEs' external interfaces (i.e., client, network, shared risk group (SRG) and degree (DEG)), and device internal connectivity retrieved by get-connection-port-trail RPC as defined in the Open ROADM MSA.



(a) General device level containment structure



(b) ROADM interfaces contained by degree (DEG) and shared risk group (SRG) port



(c) Xponder (e.g., 400G transponder) interfaces contained by network and client port

Fig. 1: Hierarchical representation for entities in a NE (Orange: device level entities, Blue: ports, Grey: interfaces, and Green: interfaces carrying signal without containing entities)

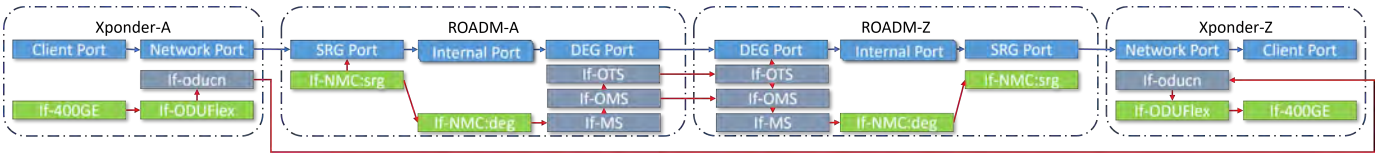


Fig. 2: A one-hop 400G link/service horizontal representation

A one-hop 400G service is represented using the horizontal dimension in Fig. 2. The blue arcs represent the signal flow as it traverses ordered supporting ports, and the red arcs represent the ordered list of supporting interfaces and logical connections at the signal layer[1].

Vendor-dependent implementations of alarm correlation and suppression within the device are known to generate inconsistencies in a multi-vendor network solution, e.g., either reporting *lossofsignal* on the interface or *lossoflight* on the port. To cope with these inconsistencies the horizontal dimension of the graph makes use of additional complementing arcs as illustrated in Fig. 2, where the vertex representing ROADM-A `If-NMC:srg` interface has an outgoing arc connecting to its SRG port vertex, and the vertex representing ROADM-Z `If-OTS` interface has an outgoing arc connecting to its DEG port vertex. These additional arcs ensure that even if some vendors implement *lossoflight* on the port and others implement *lossofsignal* on the interface, any related fault propagation can still be traced from the interface in question (e.g., `If-NMC:srg` in ROADM-A) to its neighbor's port (e.g., DEG port in ROADM-Z). Adding these additional arcs solves this potential vendor inconsistency problem and ensures logical fault propagation from the NE to its immediate neighbor.

III. Alarm Correlation and Ticketing

With the two-dimensional graph built for each service in each direction of propagation, the network alarm correlation logic is a simple data structure search problem where the root cause is the most upstream resource (vertex) that reports a failure. Alarms retrieved from NEs are carefully attached to the received vertices (i.e., resources) in the graph. The direction of each alarm is associated with the corresponding service direction of propagation. Last, the hierarchical and horizontal correlation must have a limited life-time so that independent subsequent failures are not masked by the correlation procedure.

A. Root Cause Searching Algorithm

Let $G(V, E)$ be a graph representing a given direction of a service, where V is the set of vertices and E is the set of arcs. Let V represent the set of devices, shelves, circuit-packs, ports and interfaces. An arc $e \in E$ in this graph G represents the hierarchical and/or horizontal dependencies. Let V_a represent the set of vertices with alarms. Let A_i represent the set of alarms for vertex $i \in V_a$. Let T_i represent the set of alarm time-stamps for vertex $i \in V_a$. Let t_{ij} represent the time-stamp of j^{th} alarm at vertex i. Let Δ represent the time window for alarm correlation. Let $A^{'} = \bigcup_{i=1}^{n} A_i$ be the set of all alarms in the graph. By eliminating the alarms in the graph that have an upstream alarm whose time-stamps are in the same time window, the remaining alarms are the root cause(s). The detailed algorithm is described in Algorithm 1.

Algorithm 1 Root Cause Searching

1: **for** $n_i \in V_a$ **do**
2: **for** $n_j \in V_a$ **do**
3: **if** $n_i \neq n_j$ & $\exists path(n_i, n_j)$ & $\exists (t_i, t_j) \in (T_i, T_j), \mid t_i\text{-}t_j \mid \leq \Delta$ **then**
4: $A^{'} = A^{'}$ - a_j, where $a_j \subseteq A_j$ is a subset of alarms that have an upstream alarm on vertex n_j
5: **end if**
6: **end for**
7: **end for**
8: **return** $A^{'}$

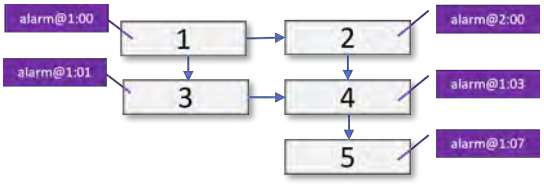
Fig. 3 illustrates with one example the correlation of alarms and the determination of root causes as it is achieved using the two-dimensional graph. The arcs between vertices represent either hierarchical or horizontal relationships. Each vertex reports one alarm with a timestamp. Assume a correlation time window of 5 minutes. Since vertex 3, 4 and 5's alarm reported time is within a 5-minute time window compared to its ancestor's alarm timestamp, these alarms are removed from the graph according to the procedure in Algorithm 1. Because vertex 2's alarm timestamp is more than 5 minutes after its ancestor's (vertex 1) alarm timestamp, and it does not have any other ancestors, vertex 2's alarm is also considered a root alarm in the graph in addition to vertex 1's alarm.

It is worth noting that if multiple faults occur in the same time window and there exists a directed path between the elements reporting the alarms in the graph, the method only finds the most upstream resource that reports a failure. This behavior makes sense because network debugging should follow a top-down approach and the fault of the hidden alarm will show up when its upstream fault is resolved.

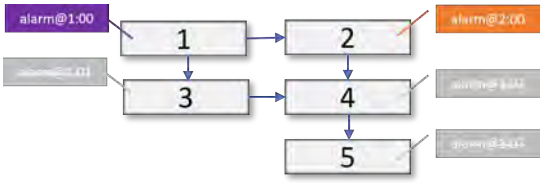
B. Ticketing System

All the remaining alarms $A^{'}$ for each graph will be pushed into a ticketing pool without duplicates, which would happen when multiple services share the same network resources. Port and interface alarms will be ticketed against the link or connection in the network. Alarms at shelf and circuit-pack level will be ticketed directly along with their relevant device

[1]For more information about these supporting interfaces and logical connections, please refer to [6].



(a) Before root cause searching



(b) After root cause searching

Fig. 3: An example of root cause searching with $\Delta = 5$ minutes

node id, resource id, and alarm fields. Corrective actions can be recommended based on a) vendor's device alarm details as a reference, and b) operator's past experience.

IV. Experiment Setup and Results

In order to test the feasibility of the proposed method and algorithm, we set up the testbed shown in Fig. 4. The Open ROADM testbed comprises two 400G transponders from INFINERA (GX Series CHM1R) and CISCO (Bo Series), respectively, and two ROADMs from FUJITSU (1FINITY) and INFINERA(G30 OLS), respectively. A 400G service is provisioned over the testbed. Three types of common network failures are emulated: 1) fiber break on the ROADM to ROADM degree connection, 2) fiber break on the network tail connection between transponder and ROADM SRG, and 3) circuit-pack failure. In all three cases, the root cause or causes (when multiple failures are induced at the same time) are successfully identified and ticketed. It is worth mentioning that multiple failures that exist at the same time could be triggered previously in the same time window or different time windows. The algorithm is able to locate the most upstream cause(s) in the graph in each time window. The running time to build and update the graph, search for the root cause(s) and issue ticket(s) is within a few seconds with Python programming language.

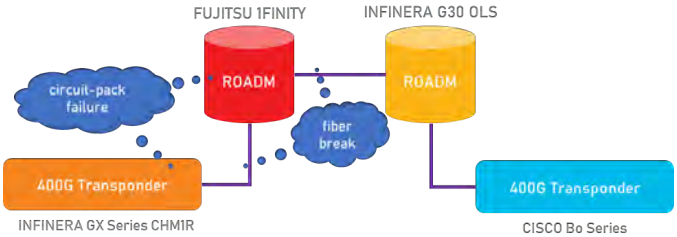


Fig. 4: Experiment setup and failure emulation

V. Summary

An alarm correlation and ticketing system for an Open ROADM disaggregated multi-vendor optical network was introduced in this paper. A two-dimensional graph-based method leveraging containment and supporting relationships among network entities was proposed and experimentally validated. Using this method, tickets with the root cause alarm and recommended corrective actions can be provided by the ticketing system of an Open ROADM disaggregated multi-vendor optical network within a few seconds from the fault occurrence. This work helps us understand details and challenges in doing alarm correlation and ticketing in the Open ROADM ecosystem, and build the ground to apply AI/ML techniques in the future. This proof-of-concept method could be used for the data set collection and labeling, and in the future to train supervised or reinforcement learning models.

References

[1] OpenROADM MSA, "OpenROADM MSA," http://OpenROADM.org, 2022, last retrieved 9/25/2022.

[2] OpenDaylight Project, "Transport PCE," https://docs.opendaylight.org/projects/transportpce/en/latest/user-guide.html#, 2022, last retrieved 9/25/2022.

[3] N. Ellsworth and et.al., "Using Network Operations Platform and Orchestrator to Enhance Programmable OpenROADM Optical Networks," in *2022 Optical Fiber Communications Conference and Exhibition (OFC)*, 2022, p. M3Z.8.

[4] ITU-T, "ITU-T G.798.1," https://www.itu.int/rec/dologin_pub.asp?lang=s&id=T-REC-G.798.1-201104-S!!PDF-E&type=items, 2011.

[5] Q. Pham-Van and et.al., "Demonstration of Alarm Correlation in Partially Disaggregated Optical Networks," in *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, 2020, p. M3Z.6.

[6] "OpenROADM Device (7.1 MSA) Whitepaper," https://0201.nccdn.net/4_2/000/000/072/2aa/open-roadm-msa-release-7-device-white-paper-v1.2.pdf, OpenROADM, 2020.