NMOP Working Group
Internet-Draft
Intended status: Standards Track
Expires: 18 August 2026

T. Hu
CMCC
L. M. Contreras
Telefonica
Q. Wu
Huawei
N. Davis
Ciena
C. Feng

14 February 2026

A YANG Data Model for Network Incident Management
draft-ietf-nmop-network-incident-yang-08

Abstract

This document defines a YANG Module for the network incident lifecycle management. This YANG module is meant to provide a standard way to report, diagnose, and help reduce troubleshooting tickets and resolve network incidents for the sake of network service health and probable cause analysis.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 18 August 2026.

Table of Contents

1. Introduction

[RFC8969] defines a framework for Automating Service and Network Management with YANG [RFC7950] to full life cycle network management. A set of YANG data models have already been developed in IETF for network performance monitoring and fault monitoring, e.g., a YANG data model for alarm management [RFC8632] defines a standard interface for alarm management. A data model for Network and VPN Service Performance Monitoring [RFC9375] defines a standard interface for network performance management. In addition, distributed tracing mechanism defined in [W3C-Trace-Context] can be used to analyze and debug operations, such as configuration transactions, across multiple distributed systems.

However, these YANG data models for network maintenance are based on specific data source information and manage alarms and performance metrics data separately at different layers in various separate management systems. In addition, the frequency and quantity of alarms and performance metrics data reported to Operating Support System (OSS) have increased dramatically (in many cases multiple orders of magnitude) with the growth of service types and complexity and greatly overwhelm OSS platforms [TMF724A]; with existing known

dependency relationships between metric, alarm, and events at each layer (e.g., packet layer or optical layer), it is possible to compress series of alarms (see Section 3.5.3 of [RFC8632]) into fewer network incidents and there are many solutions in the market today that essentially do this to some degree. However, conventional solutions such as data compression are time-consuming and labor-intensive, usually rely on maintenance engineers' experience for data analysis, which, in many cases, result in low processing efficiency, inaccurate probable cause identification and duplicated tickets. It is also difficult to assess the impact of alarms, performance metrics and other anomaly data on network services without known relation across layers of the entire network topology data or the relation with other network topology data.

To address these challenges, a network-wide incident-centric solution is specified to establish the global view on dependency relationships with both network service and network topology at various different layers, which not only can be used at a specific layer in one domain but also can be used to span across layers for multi-layer network troubleshooting.

A network incident refers to an undesired occurrence, such as an unexpected interruption of a network service, degradation of a network service quality, or sub-health of a network service [I-D.ietf-nmop-terminology][TMF724A]. Different data sources, including alarms, metrics, and other anomaly information, can be correlated and combined into one or a few network incidents, regardless of layer, informed by correlation analysis and service impact assessment. For example, if the protocol-related interface fails to work properly, a large amount of alarms may be reported to the upper-layer management system. Although a lot of network services may be affected by the interface, only one aggregated network incident pertaining to the abnormal interface will be reported. A network incident may also be raised through the analysis of some network performance metrics, for example, as described in SAIN [RFC9417], network services can be decomposed to several sub-services, specific metrics can be monitored for each sub-service. Therefore symptoms will occur if services/sub-services are unhealthy (after analyzing metrics), in addition, these symptoms may give rise to a network incident when it causes degradation of the network services.

In addition, Artificial Intelligence (AI) and Machine Learning (ML) are key technologies in the processing of large amounts of data with complex data correlations (see Section 6.1 of [I-D.irtf-nmrg-ai-challenges]). For example, Neural Network Algorithm or Hierarchy Aggregation Algorithm can be used to replace manual alarm data correlation. Through online and offline self-learning, these algorithms can be continuously optimized to improve the efficiency of fault diagnosis.

This document defines a YANG data model for network incident lifecycle management, which improves troubleshooting efficiency, and improves network automation [RFC8969] with RPC operations in this YANG module.

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The following terms are defined in [RFC8632], [RFC9543],[I-D.ietf-nmop-terminology] and are not redefined here:

* Alarm

* Resource

* Event

* Problem

* Incident

* Anomaly

* Cause

* Characteristic

* SLA (Service Level Agreement)

* SLO (Service Level Objective)

The following terms are defined in this document:

Service Impact Assessment: A process that uses algorithmic techniques (e.g., machine learning, automated reasoning, conformance checking, graph traversal, among others) to evaluate whether the network service has been impacted by the network incident and map the network incident to one or a set of network service, which can reduce the volume of fault/alarms reporting, facilitate troubleshooting, and assure network service performance and availability.

Incident Management: Lifecycle management of network incidents, including network incident identification, reporting, acknowledgement, diagnosis, and resolution. Different from the traditional fault management, it takes various different data sources including alarms, metrics, and other anomaly information and aggregates them into one or a few network incidents irrespective of layer through data correlation analysis and the Service Impact Assessment. One fault on the network device can cause multiple network incidents, e.g., multiple service offerings that are dependent on that device and take different route will go down, e.g., suffer increased latency as redundant routes become more congested. A network incident might impact one or a set of network services. The network incident can also been seen as customer incident [TMF724A] when service SLA [RFC9543] associated with one specific network service and network incident has been affected. How customer incident is translated from the network incident is beyond the scope of this document.

Incident Management System: An entity that implements network incident management. It includes (but not limited to) Incident Server and Incident Client.

Incident Server: An entity that is responsible for detecting and reporting one network incident, performing network incident diagnosis, resolution and prediction in specific domain, etc.

Incident Client: An entity that can manage network incidents based on global view on network topology data correlation. For example, it can receive network incident notifications, query the information of network incidents, instruct an Incident Server to diagnose, help resolve, etc. In addition, it can trigger issue tickets and involve repair crew to fix the problem.

Incident Handler: An entity that can receive network incident

notification, store and query the information of network incidents for data analysis. Unlike the Incident Client, it does not control the incident server and cannot instruct it to perform network incident diagnosis or resolution.

Probable Root Cause: If removing a factor completely resolves the ongoing incident (specifically, regarding network outage or service impairments and their associated subsequent failures and symptoms) and prevents the problem from recurring, then such factor is considered as a probable root cause of a problem.

Since one Fault may give rise to another Fault or Problem, a probable root cause is commonly meant to describe the original event or combination of circumstances that is the foundation of all related Faults.

Conversely, a causal factor is a contributing action that influences the outcome of the incident or event but is not the probable cause.

3. Sample Use Cases

3.1. Incident-Based Trouble Tickets Dispatching

Usually, the dispatching of trouble tickets in a network is mostly based on alarms data analysis and often requires operators' maintenance engineers. These operators' maintenance engineers are responsible for monitoring and detecting and correlating alarms, e.g., that alarms at both endpoints of a specific tunnel or at both optical and IP layers which are associated with the same network fault. Therefore, they can correlate these alarms to the same trouble ticket, which offers a low level of automation. If there are more alarms, then the human costs for network maintenance are increased accordingly.

Some operators preconfigure accept-lists and adopt some coarse granularity data correlation rules for the alarm management. This approach seems to improve fault management automation. However, some trouble tickets might be missed if the filtering conditions are too restrictive. If the filtering conditions are not restrictive, it might end up with multiple trouble tickets being dispatched to the same network fault. It is hard to achieve a perfect balance between the network management automation and duplicated trouble tickets under the conventional working situations.

With the help of the Network Incident Management, massive alarms can be aggregated into a few network incidents based on service impact asessment, so the number of trouble tickets will be reduced. At the same time, the efficiency of network troubleshooting can be largely improved, which addresses the pain point of traditional trouble ticket dispatching.

3.2. Incident Derivation from L3VPN Service Unavailability

The Service Attachment Points (SAPs) defined in [RFC9408] represent the network reference points where network services can be delivered or are being delivered to customers.

SLOs [RFC9543] can be used to characterize the ability of a particular set of nodes to communicate according to certain measurable expectations [I-D.ietf-ippm-pam]. For example, an SLA might state that any given SLO applies to at least a certain percentage of packets, allowing for a certain level of packet loss and exceeding packet delay threshold to take place. For example, an SLA might establish a multi-tiered SLO of end-to-end latency as follows:

* Not to exceed 30 ms for any packet.

* Not to exceed 25 ms for 99.999% of packets.

* Not to exceed 20 ms for 99% of packets.

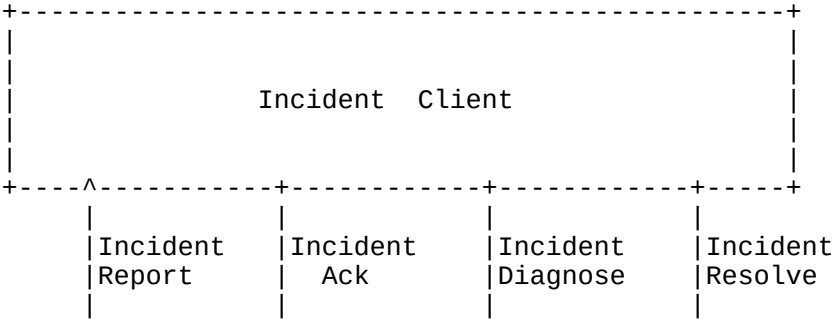
This SLA information can be bound with two SAPs or multiple SAPs defined in [RFC9408], so that the service orchestration layer can use these interfaces to commit the delivery of a service on specific point-to-point service topology or point to multi-point topology. When a given SLO threshold is violated, a network incident (or customer incident [TMF724A] associated with an L3VPN service may be derived.

3.3. Multi-layer Fault Demarcation

When a fault occurs in a network that contains both packet-layer devices and optical-layer devices, it may cause correlative faults in both layers, i.e., packet layer and optical layer. Specifically, fault propagation could be classified into three typical types. First, faults occurring at a packet-layer device might further cause fault (e.g., Wavelength Division Multiplexing (WDM) client fault) at an optical-layer device. Second, faults occurring at an optical-layer device might further cause fault (e.g., Layer 3 link down) at a packet- layer device. Third, faults occurring at the inter-layer link between a packet-layer device and an optical-layer device might further cause faults at both devices. Multiple operation teams are usually needed to first analyse a large amount of alarms (triggered by the above-mentioned faults) from single network layer (either packet layer or optical layer) independently, then cooperate to locate the Probable Root Cause through manually analyzing multi-layer topology data and service data, thus fault demarcation becomes more complex and time-consuming in multi-layer scenario than in single-layer scenario.

With the help of Network Incident Management, the management systems first automatically analyze Probable Root Cause of the alarms at each layer and report corresponding network incidents to the multi-layer, multi-domain management system, then such management system comprehensively analyzes the topology relationship and service relationship between the Probable Root Causes of both layers. The inner relationship among the alarms will be identified and finally the Probable Root Cause will be located among multiple layers. By cooperating with the integrated Optical time-domain reflectometer (OTDR) embedded within the network device, we can determine the target optical exchange station before site visits. Therefore, the overall fault demarcation process is simplified and automated, the analysis result could be reported and visualized in time. In this case, operation teams only have to confirm the analyzing result and dispatch site engineers to perform relevant maintenance actions (e.g., splice fiber) based on the Probable Root Cause.

4. Network Incident Management Architecture



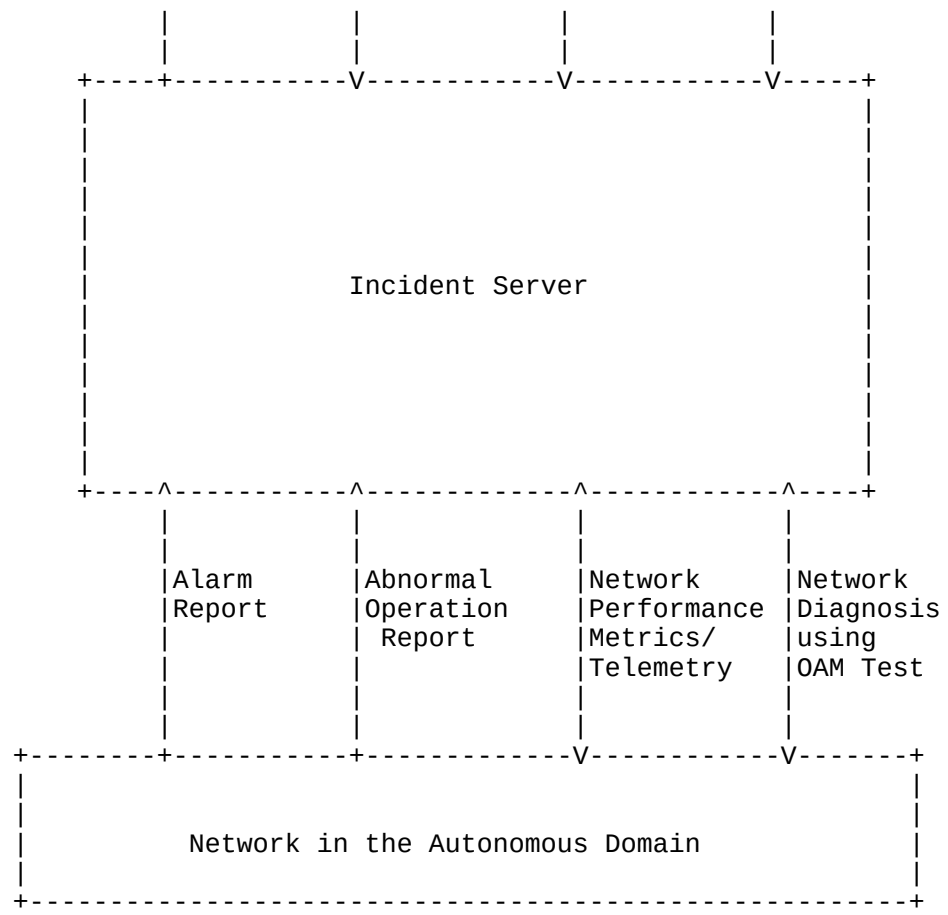


Figure 1: Network Incident Management Architecture

Figure 1 illustrates the Network Incident Management architecture. Two key components for the Incident Management are the Incident Client and the Incident Server.

The Incident Server can be deployed in network operation platforms, network analytic platforms, controllers [RFC8969] in each domain and provides functionality such as network incident identification, report, diagnosis, resolution, or querying for the network incident lifecycle management.

The Incident Client can be deployed within a single domain as the Incident Server or across domains with the global view of network data. It can be deployed either in the same network operation platforms, network analytic platforms, controllers as the Incident Server within a single domain, or at the upper-layer network operation platforms, network analytic platforms or controllers (i.e.,multi-domain controllers), to invoke the functionalities provided by the Incident Server in each domain to meet business requirements of the fault management.

A typical workflow of network incident lifecycle management is as follows:

* Some alarm or abnormal operations, network performance metrics, network diagnosis information [I-D.ietf-opsawg-scheduling-oam-tests] are reported from the network to the Incident Server. The Incident Server receives these alarms/abnormal operations/metrics and try to analyze the correlation of them, e.g., generate a symptom if some metrics are evaluated as unhealthy, the Probable Root Cause can be detected based on the data correlation analysis. If a network incident is identified, the "incident report" notification will be reported to

the Incident Client. The impact of network services will be further analyzed and will update the network incident if the network service is impacted.

* Incident Client receives the network incident from the "incident report" notification reported by Incident Server, and acknowledges it with the subsequent "incident ack" rpc operation. The Incident Client may further invoke the "incident diagnose" rpc to diagnose this network incident to find the Probable Root Causes.

* If the Probable Root Causes have been found, the Incident Client can resolve this network incident by invoking the 'incident resolve' rpc operation to ask the Incident Server to resolve it, or dispatching a troubleshooting ticket or using other network functions (routing calculation, configuration, etc.) without being known by the Incident Server.

* In case of the 'incident resolve' rpc operation invoked by the Incident Client, the Incident Server will monitor the status of the network incident and update the status of network incident to 'cleared' if the incident can be fixed. For more detailed workflow, please refer to section 5.3.

4.1. Interworking with Alarm Management

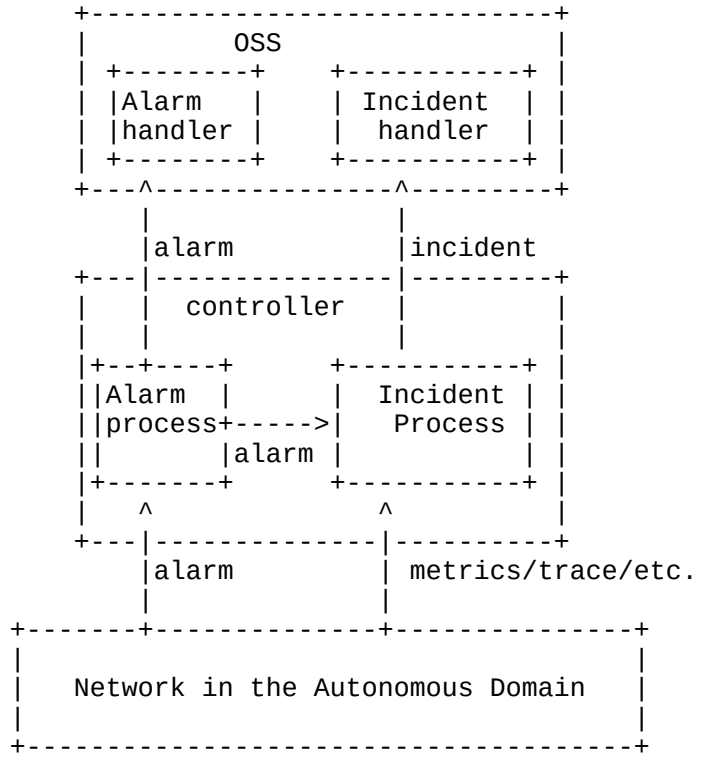


Figure 2: Interworking with Alarm Management

A YANG model for the alarm management [RFC8632] defines a standard interface to manage the lifecycle of alarms. Alarms represent the undesirable state of network resources [I-D.ietf-nmop-terminology], The alarm data model also defines the Probable Root Causes and impacted services fields, but there may be insufficient information to determine them at lower layer system (mainly in devices level), so alarms do not always tell the status of network services or necessarily point to the Probable Root Causes of problems. As described in [RFC8632], the alarm management acts as a starting point for high-level fault management. While Network Incident Management often works at the network level, so it is possible to have enough information to perform data correlation and Service Impact

Assessment. Alarms can work as one of data sources of Network Incident Management and may be aggregated into a few network incidents by the correlation analysis, network service impact and Probable Root Causes may be determined during the incident process.

Network Incident also contains some related alarms, if needed users can query the information of alarms by alarm management interface [RFC8632]. In some cases, e.g., cutover scenario, the Incident Server may use alarm management interface [RFC8632] to shelve some alarms.

Alarm management may keep the original process, alarms are reported from network to network controller or network analytic platform and then reported to upper-layer system (e.g., the alarm handler within the OSS).

Similarly, the network incident is reported from the network to the network controller or network analytic platform and then reported to the upper-layer system (e.g., Incident Handler within the OSS). Upper-layer system may store these network incidents and provide the information for fault analysis (e.g., deeper customer incident analysis based on network incident).

Different from alarm management, incident process within the controller comprising both Incident Client and Incident Server functionalities provides not only network incident reporting but also diagnosis and resolution functions, it’s possible to support self-healing and may be helpful for single-domain closed-loop control.

Incident Management is not a substitute for alarm management. Instead, they can work together to implement fault management.

4.2. Interworking with SAIN

SAIN [RFC9417] defines an architecture of network service assurance.

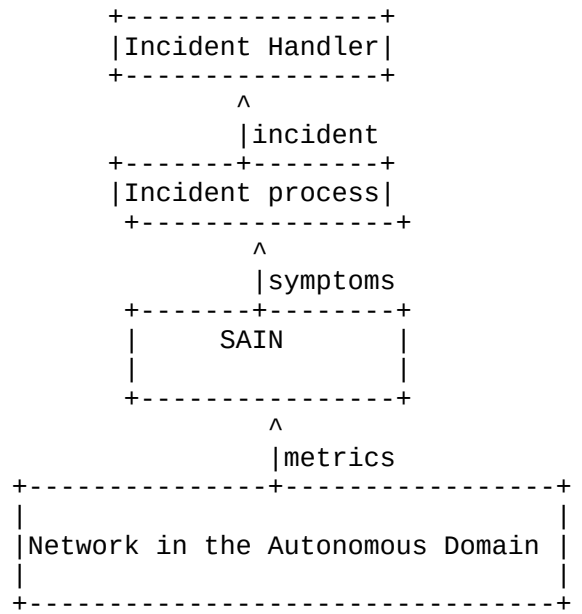


Figure 3: Interworking with SAIN

A network service can be decomposed into some sub-services, and specific metrics can be monitored for sub-services. For example, a tunnel service can be decomposed into some peer tunnel interface sub-services and IP connectivity sub-service. If some metrics are evaluated to indicate unhealthy for specific sub-service, some symptoms will be present. Incident process comprising both Incident

Client and Incident Server functionalities may identify the network incident based on symptoms, and then report it to Incident Handler within the Operation Support System (OSS). So, SAIN can be one way to identify network incident, services, sub-services and metrics can be preconfigured via APIs defined by service assurance YANG model [RFC9418] and the network incident will be reported if symptoms match certain condition or characteristic considered as an indication of a problem or potential problem.

4.3. Relationship with RFC8969

[RFC8969] defines a framework for network automation using YANG, this framework breaks down YANG modules into three layers, service layer, network layer and device layer, and contains service deployment, service optimization/assurance, and service diagnosis. Network incident works at the network layer and aggregates alarms, metrics and other information from device layer, it’s helpful to provide service assurance. And the network incident diagnosis may be one way of service diagnosis.

4.4. Relationship with Trace Context

W3C defines a common trace context [W3C-Trace-Context] for distributed system tracing, [I-D.ietf-netconf-trace-ctx-extension] defines a netconf extension for [W3C-Trace-Context] and [I-D.ietf-netconf-configuration-tracing] defines a mechanism for configuration tracing. If some errors occur when services are deploying, it’s very easy to identify these errors by distributed system tracing, and a network incident should be reported.

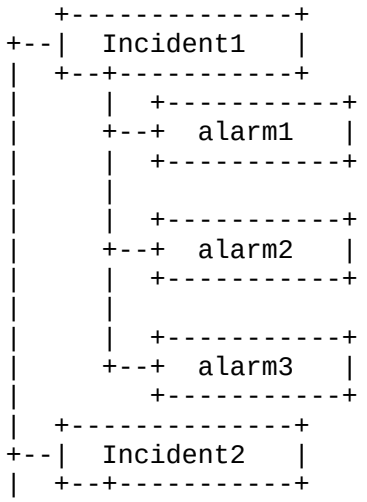
4.5. Relationship with network anomaly architecture

Network anomaly architecture [I-D.ietf-nmop-network-anomaly-architecture] focuses on improving supervised and semi-supervised machine learning systems and evaluating anomaly detection algorithms and technologies. It also can be used to monitor network changes holistically by monitoring all 3 network planes simultaneously and detect whether that change is service disruptive. In case of disruptive changes, the anomaly can be upgraded into the network incident which trigger troubleshooting tickets generation.

5. Functional Interface Requirements between the Client and the Server

5.1. Incident Identification

As depicted in Figure 4, multiple alarms, metrics, or hybrid can be aggregated into a network incident after analysis.



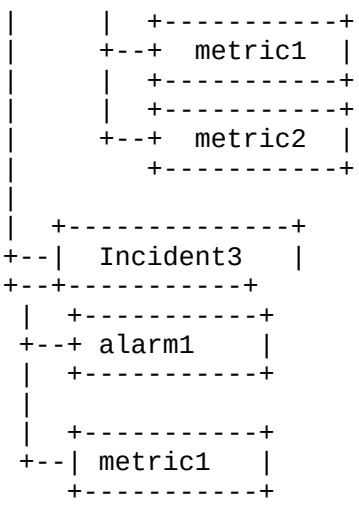


Figure 4: Incident Identification

The Network Incident Management server MUST be capable of identifying network incidents. Multiple alarms, metrics and other information are reported to Incident Server, and the server must analyze it and find out the correlations of them, if the correlation match the network incident rules, network incident will be identified, and reported to the client. If the network incident is repeated many times, the problem needs to be raised. Service Impact Assessment SHOULD be performed if a network incident is identified, and the content of network incident SHOULD be updated if impacted network services are detected.

AI/ML may be used to identify the network incident. Expert system and online learning can help AI to identify the correlation of alarms, metrics and other information by time-base correlation algorithm, topology-based correlation algorithm, etc. For example, if the interface is down, then many protocol alarms will be reported, AI will think these alarms have some correlations. These new correlations will be put into the knowledge base, and the network incident will be identified faster according to knowledge base next time.

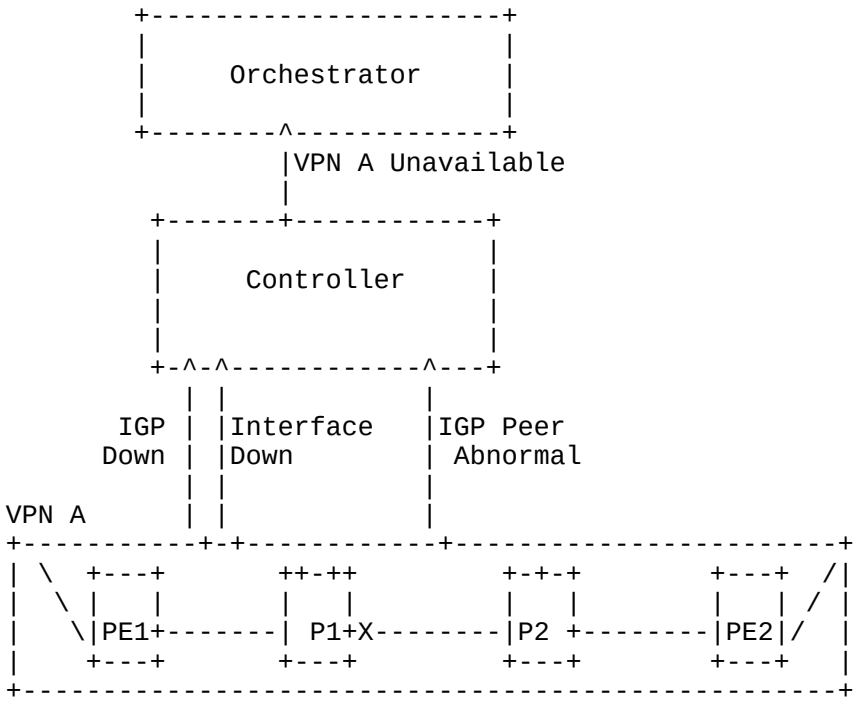


Figure 5: Example 1 of Network Incident Identification

As described in Figure 5, vpn a is deployed from PE1 to PE2, if a interface of P1 is going down, many alarms are triggered, such as interface down, igp down, and igp peer abnormal from P2.

These alarms are aggregated and analyzed by the controller/incident server, and then the network incident 'vpn unavailable' is triggered by the controller/Incident Server. If the network incident 'vpn unavailable' is repeated, the problem can be raised.

Note that Incident Server within the controller can rely on data correlation technology such as service impact asessment and data analytic component to evaluate the real effect on the relevant service and understand whether lower level or device level network anomaly, e.g., igp down, has impact on the service.

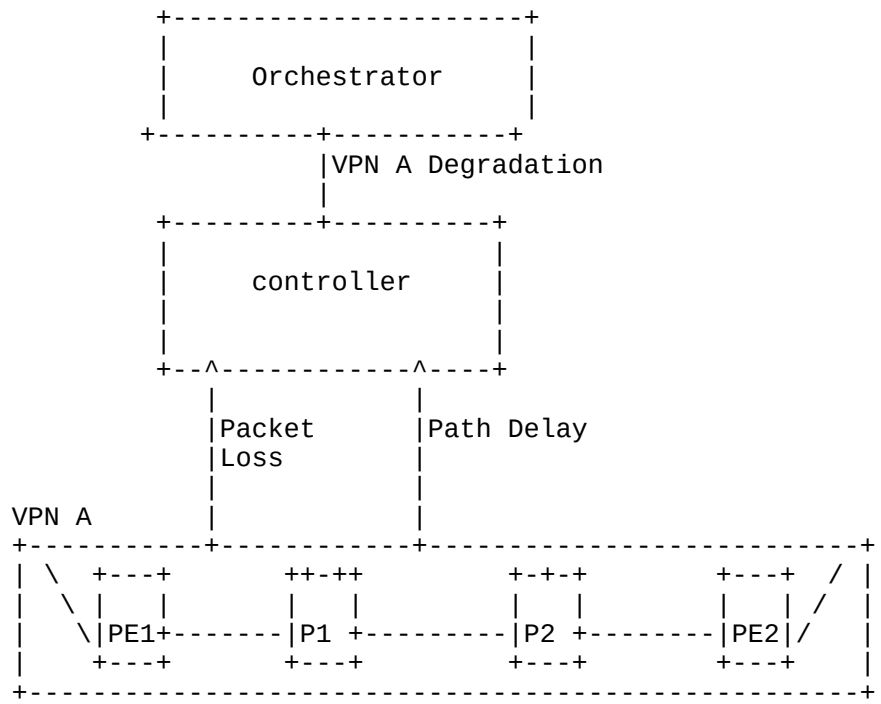


Figure 6: Example 2 of Network Incident Identification

As described in Figure 6, controller collect the network metrics from network elements, it finds the packet loss of P1 and the path delay of P2 exceed the thresholds, a network incident 'VPN A degradation' may be triggered after the Service Impact Assessment.

5.2. Incident Diagnosis

After a network incident is reported to the network Incident Client, the Incident Client MAY diagnose the incident to determine the Probable Root Cause. Some diagnosis operations may affect the running network services. The Incident Client can choose not to perform that diagnosis operation after determining the impact is not trivial. The Incident Server can also perform self-diagnosis. However, the self-diagnosis MUST NOT affect the running network services. Possible diagnosis methods include link reachability detection, link quality detection, alarm/log analysis, and short-term fine-grained monitoring of network quality metrics, etc.

5.3. Incident Resolution

After the Probable Root Cause is diagnosed, the Incident Client MAY resolve the network incident. The Incident Client MAY choose resolve

the network incident by invoking other functions, such as routing calculation function, configuration function, dispatching a ticket or asking the server to resolve it. Generally, the Incident Client would attempt to directly resolve the probable cause. If the Probable Root Cause cannot be resolved, an alternative solution SHOULD be required. For example, if a network incident caused by a physical component failure, it cannot be automatically resolved, the standby link can be used to bypass the faulty component.

Incident Server will monitor the status of the network incident, if the faults are fixed, the Incident Server will update the status of network incident to 'cleared', and report the updated network incident to the client.

Network incident resolution may affect the running network services. The client can choose not to perform those operations after determining the impact is not trivial.

6. Incident Data Model Concepts

6.1. Identifying the Incident Instance

An incident id is used as an identifier of an incident instance, if an incident instance is identified, a new incident ID is created. The incident id MUST be unique in the whole system.

6.2. The Incident Lifecycle

The network incident model clearly separates network incident instance lifecycle from operator incident lifecycle.

o Network incident instance lifecycle: The network incident instrumentation that controls whether a network incident is raised, updated, or cleared.

o Operator incident lifecycle: Operators acting upon the network incident with rpcs like acknowledged, diagnosed and resolved.

6.2.1. Network Incident Instance Lifecycle

From a network incident instance perspective, a network incident can have the following lifecycle: 'raised', 'updated', 'cleared'. When a network incident instance is first generated, the status is 'raised'. If the status changes after the network incident instance is generated, (for example, self-diagnosis, diagnosis command issued by the client, or any other condition causes the status to change but does not reach the 'cleared' level) , the status changes to 'updated'. When a network incident is successfully resolved, the status changes to 'cleared'.

6.2.2. Operator Incident Lifecycle

Operators can act upon network incident with network incident rpcs. From an operator perspective, the lifecycle of a network incident instance includes 'acknowledged', 'diagnosed', and 'resolved'.

When a network incident instance is generated, the operator SHOULD acknowledge the network incident with 'incident-acknowledge' rpc. And then the operator attempts to diagnose the network incident with 'incident-diagnose' rpc (for example, find out the Probable Root Cause and affected components). Diagnosis is not mandatory. If the Probable Root Cause and affected components are known when the network incident is generated, diagnosis is not required. After locating the Probable Root Cause and affected components, operator can try to resolve the network incident by invoking 'incident-resolve' rpc.

7. Incident Data Model Design

7.1. Overview

There is one YANG module in the "ietf-incident" model, which defines technology independent abstraction of network incident construct for alarm, log, performance metrics, etc. The information reported in the network incident include Probable Root Cause, priority, impact, suggestion, etc.

At the top of "ietf-incident" module is the Network Incident. Network incident is represented as a list and indexed by "incident-id". Each Network Incident is associated with a network service instance, domain and sources. Under sources, there is one or more sources. Each source corresponds to node defined in the network topology model and network resource in the network device, e.g., interface. In addition, "ietf-incident" supports one general notification to report network incident state changes and three rpcs to manage the network incidents.

```
module: ietf-incident
  +--ro incidents
     +--ro incident* [name type incident-id]
        +--ro incident-no         uint64
        +--ro name                string
        +--ro type                identityref
        +--ro incident-id?        string
        +--ro service-instance*   string
        +--ro domain              identityref
        +--ro priority            incident-priority
        +--ro status?             enumeration
        +--ro ack-status?         enumeration
        +--ro category            identityref
        +--ro detail?             string
        +--ro resolve-advice?     String
        +--ro sources
        ...
        +--ro probable-causes
        ...
        +--ro probable-events
        ...
        +--ro events
        ...
        +--ro raise-time? yang:date-and-time
        +--ro occur-time? yang:date-and-time
        +--ro clear-time? yang:date-and-time
        +--ro ack-time? yang:date-and-time
        +--ro last-updated? yang:date-and-time
rpcs:
  +---x incident-acknowledge
  ...
  +---x incident-diagnose
  ...
  +---x incident-resolve

notifications:
  +---n incident-notification
         +--ro incident-no?
                          -> /inc:incidents/inc:incident/inc:incident-no
         ...
         +--ro time? yang:date-and-time
```

7.2. Incident Notifications

notifications:

```
  +---n incident-notification
     +--ro incident-no?        incident-ref
     +--ro name                string
     +--ro type                identityref
     +--ro incident-id?        string
     +--ro service-instance*   string
     +--ro domain              identityref
     +--ro priority            incident-priority
     +--ro status?             enumeration
     +--ro ack-status?         enumeration
     +--ro category            identityref
     +--ro detail?             string
     +--ro resolve-advice?     string
     +--ro sources
     |  +--ro source* [node-ref]
     |     +--ro node-ref       leafref
     |     +--ro network-ref?   leafref
     |     +--ro resource* [name]
     |        +--ro name    al:resource
     +--ro probable-causes
     |  +--ro probable-cause* [node-ref]
     |     +--ro node-ref       leafref
     |     +--ro network-ref?   leafref
     |     +--ro resource* [name]
     |     |  +--ro name          al:resource
     |     |  +--ro cause-name?   identityref
     |     |  +--ro detail?       string
     |     +--ro cause-name?    identityref
     |     +--ro detail?        string
     +--ro probable-events
     |  +--ro probable-event* [type event-id]
     |     +--ro type        leafref
     |     +--ro event-id    leafref
     +--ro events
     |  +--ro event* [type event-id]
     |     +--ro type           identityref
     |     +--ro event-id       string
     |     +--ro (event-type-info)?
     |        +--:(alarm)
     |           +--ro alarm
     |              +--ro resource?               leafref
     |              +--ro alarm-type-id?          leafref
     |              +--ro alarm-type-qualifier?   leafref
     +--ro time?              yang:date-and-time
```

A general notification, incident-notification, is provided here. When a network incident instance is identified, the notification will be sent. After a notification is generated, if the incident server performs self diagnosis or the Incident Client uses the interfaces provided by the Incident Server to deliver diagnosis and resolution actions, the notification update behavior is triggered, for example, the Probable Root Cause objects and affected objects are updated. When a network incident is successfully resolved, the status of the network incident would be set to ’cleared’.

7.3. Incident Acknowledge

```
  +---x incident-acknowledge
  |  +---w input
  |  |  +---w incident-no*
  |  |          -> /inc:incidents/inc:incident/inc:incident-no
```

After an incident is generated, updated, or cleared, the operator needs to confirm the incident to ensure that the client knows the incident.

In some scenarios where automatic diagnosis and resolution are supported, the status of an incident may be updated multiple times or even automatically resolved. Therefore the incident-acknowledge rpc can confirm multiple incidents at a time.

7.4. Incident Diagnose

```
+---x incident-diagnose
|  +---w input
|  |  +---w incident-no*
|  |          -> /inc:incidents/inc:incident/inc:incident-no
```

After a network incident is generated, network incident diagnose rpc can be used to diagnose the network incident and locate the Probable Root Causes. On-demand Diagnosis can be performed on some detection tasks, such as bfd detection, flow detection, telemetry collection, short-term threshold alarm, configuration error check, or test packet injection.

After the on-demand diagnosis is performed successfully, a separate network incident update notification will be triggered to report the latest status of the network incident asynchronously.

7.5. Incident Resolution

```
+---x incident-resolve
 +---w input
 |  +---w incident-no*
 |          -> /inc:incidents/inc:incident/inc:incident-no
```

After the Probable Root Causes and impacts are determined, incident-resolve rpc can be used to resolve the incident (if the server can resolve it). How to resolve an incident instance is out of the scope of this document.

Network incident resolve rpc allows multiple network incident instances to be resolved at a time. If a network incident instance is successfully resolved, a separate notification will be triggered to update the network incident status to 'cleared'. If the network incident content is changed during this process, a notification update will be triggered.

7.6. RPC Failure

If the RPC fails, the RPC error response MUST indicate the reason for the failure. The structures defined in this document MUST encode specific errors and be inserted in the error response to indicate the reason for the failure.

The tree diagram [RFC8340] for structures is defined as follows:

```
  structure incident-acknowledge-error-info:
    +-- incident-acknowledge-error-info
       +-- incident-no?   uint64
       +-- reason?        identityref
       +-- description?   string
  structure incident-diagnose-error-info:
    +-- incident-diagnose-error-info
       +-- incident-no?   uint64
       +-- reason?        identityref
       +-- description?   string
  structure incident-resolve-error-info:
    +-- incident-resolve-error-info
       +-- incident-no?   uint64
       +-- reason?        identityref
       +-- description?   string
```

Valid errors that can occur for each structure defined in this document are described as follows:

```
incident-acknowledge-error-info
-----------------------------------
repeated-acknowledge

incident-diagnose-error-info
-----------------------------------
probable-cause-unlocated
permission-denied
operation-timeout
resource-unavailable

incident-resolve-error-info
-----------------------------------
probable-cause-unresolved
permission-denied
operation-timeout
resource-unavailable
```

8. Network Incident Management YANG Module

This module imports types defined in [RFC6991], [RFC8345], [RFC8632],[RFC8791].

```
<CODE BEGINS> file "ietf-incident@2025-09-16.yang"
module ietf-incident {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-incident";
  prefix inc;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }
  import ietf-alarms {
    prefix al;
    reference
      "RFC 8632: A YANG Data Model for Alarm Management";
  }
  import ietf-network {
    prefix nw;
    reference
      "RFC 8345: A YANG Data Model for Network Topologies";
  }
  import ietf-yang-structure-ext {
    prefix sx;
    reference
          "RFC 8791: YANG Data Structure Extensions";

  }
  organization
    "IETF NMOP Working Group";
  contact
    "WG Web:   <https://datatracker.ietf.org/wg/nmop/>;
     WG List:  <mailto:nmop@ietf.org>

     Author:   Chong Feng
               <mailto:fengchonglly@gmail.com>
     Author:   Tong Hu
               <mailto:hutong@cmhi.chinamobile.com>
     Author:   Luis Miguel Contreras Murillo
               <mailto:luismiguel.contrerasmurillo@telefonica.com>
```

```
   Author :  Qin Wu
             <mailto:bill.wu@huawei.com>
   Author:   Nigel Davis
             <mailto:ndavis@ciena.com>";
description
  "This module defines the interfaces for incident management
   lifecycle.

   This module is intended for the following use cases:
   * incident lifecycle management:
     - incident report: report incident instance to client
                        when an incident instance is detected.
     - incident acknowledge: acknowledge an incident instance.
     - incident diagnose: diagnose an incident instance.
     - incident resolve: resolve an incident instance.

   Copyright (c) 2024 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Revised BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (https://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX
   (https://www.rfc-editor.org/info/rfcXXXX); ; see the RFC
   itself for full legal notices.

   The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
   NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
   'MAY', and 'OPTIONAL' in this document are to be interpreted as
   described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
   they appear in all capitals, as shown here. ";

revision 2025-09-16 {
  description
    "Merge incident yang with incident type yang
     and fix broken ref.";
  reference
    "RFC XXX: YANG module for network incident management.";
}

// Identities

identity incident-domain {
  description
    "The abstract identity to indicate the domain of
     an incident.";
}

identity single-domain {
  base incident-domain;
  description
    "Single domain.";
}

identity access {
  base single-domain;
  description
    "access domain.";
}

identity ran {
  base access;
```

```
        description
          "Radio access network domain.";
      }

      identity transport {
        base single-domain;
        description
          "Transport domain.";
      }

      identity otn {
        base transport;
        description
          "Optical transport network domain.";
        reference
          "RFC9376: Applicability of GMPLS for beyond 100 Gbit/s Optical Transport Network
";

      }

      identity ip {
        base single-domain;
        description
          "Ip domain.";
            reference
              "RFC1136: Administrative Domains and Routing Domains A Model for Routing in
the Internet";
      }

      identity ptn {
        base ip;
        description
          "Packet transport network domain.";
          reference
           "RFC6373: MPLS Transport Profile (MPLS-TP) Control Plane Framework";

      }

      identity cross-domain {
        base incident-domain;
        description
          "Cross domain.";
      }

      identity incident-category {
        description
          "The abstract identity for incident category.";
      }

      identity device {
        base incident-category;
        description
          "Device category.";
          reference
          "RFC8348: A YANG Data Model for Hardware Management";

      }

      identity power-environment {
        base device;
        description
          "Power environment category.";
          reference
          "RFC8348: A YANG Data Model for Hardware Management";

      }
```

```
identity device-hardware {
  base device;
  description
    "Device hardware category.";
    reference
    "RFC8348: A YANG Data Model for Hardware Management";

}

identity device-software {
  base device;
  description
    "Device software category";
    reference
    "RFC8348: A YANG Data Model for Hardware Management";

}

identity line-card {
  base device-hardware;
  description
    "Line card category.";
    reference
    "RFC8348: A YANG Data Model for Hardware Management";

}

identity maintenance {
  base incident-category;
  description
    "Maintenance category.";
}

identity network {
  base incident-category;
  description
    "Network category.";
}

identity protocol {
  base incident-category;
  description
    "Protocol category.";
}

identity overlay {
  base incident-category;
  description
    "Overlay category";
}

identity vm {
  base incident-category;
  description
    "VM category.";
}

identity event-type {
  description
    "The abstract identity for Event type";
  reference
    "RFCXXXX: Some Key Terms for Network Fault and Problem Management";

}
```

```
identity alarm {
  base event-type;
  description
    "Alarm event type.";
  reference
    "RFC8632: A YANG Data Model for Alarm Management";

}

identity notif {
  base event-type;
  description
    "Notification event type.";
      reference
        "RFC5277:NETCONF Event Notifications";
}

identity log {
  base event-type;
  description
    "Log event type.";
      reference
        "RFC5424: The Syslog Protocol";
}

identity kpi {
  base event-type;
  description
    "KPI event type.";
  reference
    "RFC2330: Framework for IP Performance Metrics";
}

identity unknown {
  base event-type;
  description
    "Unknown event type.";
}

identity incident-class {
  description
    "The abstract identity for Incident category.";
}

identity problem {
  base incident-class;
  description
    "It indicates the class of the incident is a problem
    (i.e.,cause of the incident) for example an interface
    fails to work.";
  reference
    "RFCXXXX: Some Key Terms for Network Fault and Problem Management";

}

identity sla-violation {
  base incident-class;
  description
    "It indicates the class of the incident is a sla
     violation, for example high CPU rate may cause
     a fault in the future.";
}

identity acknowledge-error {
  description
    "Base identity for the problem found while attempting
```

```
     to fulfill an 'incident-acknowledge' RPC request.";
}

identity diagnose-error {
  description
    "Base identity for the problem found while attempting
     to fulfill an 'incident-diagnose' RPC request.";
}

identity resolve-error {
  description
    "Base identity for the problem found while attempting
     to fulfill an 'incident-resolve' RPC request.";
}

identity repeated-acknowledge {
  base acknowledge-error;
  description
    "The incident referred to has already been acknowledged.";
}

identity probable-cause-unlocated {
  base diagnose-error;
  description
    "Fail to locate the probable causes when performing the
     diagnosis operation. The detailed reason MUST be included
     in the 'description'.";
}

identity probable-cause-unresolved {
  base resolve-error;
  description
    "Fail to resolve the probable causes when performing the
     resolution operation. The detailed reason MUST be included
     in the 'description'";
}

identity permission-denied {
  base diagnose-error;
  base resolve-error;
  description
    "The permission required for performing specific
     detection/resolution task is not granted.";
}

identity operation-timeout {
  base diagnose-error;
  base resolve-error;
  description
    "The diagnosis/resolution time exceeds the preset time.";
}

identity resource-unavailable {
  base diagnose-error;
  base resolve-error;
  description
    "The resource is unavailable to perform
     the diagnosis/resolution operation.";
}

identity cause-name {
  description
    "Base identity for the cause name.";
}

// Typedefs
```

```
typedef incident-priority {
  type enumeration {
    enum critical {
      description
        "The incident MUST be handled immediately.";
    }
    enum high {
      description
        "The incident should be handled as soon as
         possible.";
    }
    enum medium {
      description
        "Network services are not affected, or the
         services are slightly affected,but corrective
         measures need to be taken.";
    }
    enum low {
      description
        "Potential or imminent service-affecting
         incidents are detected,but services are
         not affected currently.";
    }
  }
  description
    "Define the priority of incident.";
}

typedef incident-ref {
  type leafref {
    path "/inc:incidents/inc:incident/inc:incident-no";
  }
  description
    "Reference a network incident.";
}

// Groupings

grouping probable-cause-info {
  description
    "The information of probable cause.";
  leaf cause-name {
    type identityref {
      base cause-name;
    }
    description
      "The name of cause.";
  }
  leaf detail {
    type string;
    description
      "The detail information of the cause.";
  }
}

grouping resources-info {
  description
    "The grouping which defines the network
     resources of a node.";
  uses nw:node-ref;
  list resource {
    key "name";
    description
      "The resources of a network node.";
    leaf name {
```

```
      type al:resource;
      description
        "Network resource name.";
    }
  }
}

grouping incident-time-info {
  description
    "The grouping defines incident time information.";
  leaf raise-time {
    type yang:date-and-time;
    description
      "The time when an incident instance is raised.";
  }
  leaf occur-time {
    type yang:date-and-time;
    description
      "The time when an incident instance occurs.
       It's the occur time of the first event during
       incident detection.";
  }
  leaf clear-time {
    type yang:date-and-time;
    description
      "The time when an incident instance is
       resolved.";
  }
  leaf ack-time {
    type yang:date-and-time;
    description
      "The time when an incident instance is
       acknowledged.";
  }
  leaf last-updated {
    type yang:date-and-time;
    description
      "The latest time when an incident instance is
       updated";
  }
}

grouping incident-info {
  description
    "The grouping defines the information of an
     incident.";
  leaf name {
    type string;
    mandatory true;
    description
      "The name of an incident.";
  }
  leaf type {
    type identityref {
      base incident-class;
    }
    mandatory true;
    description
      "The type of an incident.";
  }
  leaf incident-id {
    type string;
    description
      "The unique qualifier of an incident instance type.
      This leaf is used when the 'type' leaf cannot
      uniquely identify the incident instance type.  Normally,
```

```
    this is not the case, and this leaf is the empty string.";
}
leaf-list service-instance {
  type string;
  description
    "The related network service instances of
     the incident instance.";
}
leaf domain {
  type identityref {
    base incident-domain;
  }
  mandatory true;
  description
    "The domain of an incident.";
}
leaf priority {
  type incident-priority;
  mandatory true;
  description
    "The priority of an incident instance.";
}
leaf status {
  type enumeration {
    enum raised {
      description
        "An incident instance is raised.";
    }
    enum updated {
      description
        "The information of an incident instance
         is updated.";
    }
    enum cleared {
      description
        "An incident is cleared.";
    }
  }
  default "raised";
  description
    "The status of an incident instance.";
}
leaf ack-status {
  type enumeration {
    enum acknowledged {
      description
        "The incident has been acknowledged by user.";
    }
    enum unacknowledged {
      description
        "The incident hasn’t been acknowledged.";
    }
  }
  default "unacknowledged";
  description
    "The acknowledge status of an incident.";
}
leaf category {
  type identityref {
    base incident-category;
  }
  mandatory true;
  description
    "The category of an incident.";
}
leaf detail {
```

```
  type string;
  description
    "Detailed information of this incident.";
}
leaf resolve-advice {
  type string;
  description
    "The advice to resolve this incident.";
}
container sources {
  description
    "The source components.";
  list source {
    key "node-ref";
    min-elements 1;
    description
      "The source components of incident.";
    uses resources-info;
  }
}
container probable-causes {
  description
    "The probable cause objects.";
  list probable-cause {
    key "node-ref";
    description
      "The probable causes of incident.";
    uses resources-info {
      augment "resource" {
        description
          "Augment probable cause information.";
        //if probable cause object is a resource of a node
        uses probable-cause-info;
      }
    }
    //if probable cause object is a node
    uses probable-cause-info;
  }
}
container probable-events {
  description
    "The probable cause related events of the incident.";
  list probable-event {
    key "type event-id";
    description
      "The probable cause related event of the incident.";
    leaf type {
      type leafref {
        path "../../../events/event/type";
      }
      description
        "The event type.";
    }
    leaf event-id {
      type leafref {
        path "../../../events/event[type = current()/../type]"
           + "/event-id";
      }
      description
        "The event identifier, such as uuid,
         sequence number, etc.";
    }
  }
}
container events {
  description
```

```
  "Related events.";
list event {
  key "type event-id";
  description
    "Related events.";
  leaf type {
    type identityref {
      base event-type;
    }
    description
      "Event type.";
  }
  leaf event-id {
    type string;
    description
      "The event identifier, such as uuid,
       sequence number, etc.";
  }
  choice event-type-info {
    description
      "Event type information.";
    case alarm {
      when "derived-from-or-self(type, 'alarm')" {
        description
          "Only applies when type is alarm.";
      }
      container alarm {
        description
          "Alarm type event.";
        leaf resource {
          type leafref {
            path "/al:alarms/al:alarm-list/al:alarm"
               + "/al:resource";
          }
          description
            "Network resource.";
          reference
            "RFC 8632: A YANG Data Model for Alarm
             Management";
        }
        leaf alarm-type-id {
          type leafref {
            path "/al:alarms/al:alarm-list/al:alarm"
               + "[al:resource = current()/../resource]"
               + "/al:alarm-type-id";
          }
          description
            "Alarm type id";
          reference
            "RFC 8632: A YANG Data Model for Alarm
              Management";
        }
        leaf alarm-type-qualifier {
          type leafref {
            path "/al:alarms/al:alarm-list/al:alarm"
               + "[al:resource = current()/../resource]"
               + "[al:alarm-type-id = current()/.."
               + "/alarm-type-id]/al:alarm-type-qualifier";
          }
          description
            "Alarm type qualifier";
          reference
            "RFC 8632: A YANG Data Model for Alarm
             Management";
        }
      }
```

```
        }
      }
    }
  }
}

// RPCs

rpc incident-acknowledge {
  description
    "This rpc can be used to acknowledge the specified
     incidents.";
  input {
    leaf-list incident-no {
      type incident-ref;
      description
        "The identifier of an incident instance.";
    }
  }
}

rpc incident-diagnose {
  description
    "This rpc can be used to diagnose the specified
     incidents. The result of diagnosis will be reported
     by incident notification.";
  input {
    leaf-list incident-no {
      type incident-ref;
      description
        "The identifier of an incident instance.";
    }
  }
}

rpc incident-resolve {
  description
    "This rpc can be used to resolve the specified
     incidents. The result of resolution will be reported
     by incident notification.";
  input {
    leaf-list incident-no {
      type incident-ref;
      description
        "The identifier of an incident instance.";
    }
  }
}

sx:structure incident-acknowledge-error-info {
  container incident-acknowledge-error-info {
    description
      "This structure data MAY be inserted in the RPC error
       response to indicate the reason for the
       incident acknowledge failure.";
    leaf incident-no {
      type uint64;
      description
        "Indicates the incident identifier that
         fails the operation.";
    }
    leaf reason {
      type identityref {
        base acknowledge-error;
      }
      description
```

```
        "Indicates the reason why the operation is failed.";
    }
    leaf description {
      type string;
      description
        "Indicates the detailed description about the failure.";
    }
  }
}
sx:structure incident-diagnose-error-info {
  container incident-diagnose-error-info {
    description
      "This structure data MAY be inserted in the RPC error
       response to indicate the reason for the
       incident diagnose failure.";
    leaf incident-no {
      type uint64;
      description
        "Indicates the incident identifier that
         fails the operation.";
    }
    leaf reason {
      type identityref {
        base diagnose-error;
      }
      description
        "Indicates the reason why the operation is failed.";
    }
    leaf description {
      type string;
      description
        "Indicates the detailed description about the failure.";
    }
  }
}
sx:structure incident-resolve-error-info {
  container incident-resolve-error-info {
    description
      "This structure data MAY be inserted in the RPC error
       response to indicate the reason for the
       incident resolution failure.";
    leaf incident-no {
      type uint64;
      description
        "Indicates the incident identifier that
         fails the operation.";
    }
    leaf reason {
      type identityref {
        base resolve-error;
      }
      description
        "Indicates the reason why the operation is failed.";
    }
    leaf description {
      type string;
      description
        "Indicates the detailed description about the failure.";
    }
  }
}

// Notifications

notification incident-notification {
  description
```

```
      "Incident notification. It will be triggered when
       the incident is raised, updated or cleared.";
    leaf incident-no {
      type incident-ref;
      description
        "The identifier of an incident instance.";
    }
    uses incident-info;
    leaf time {
      type yang:date-and-time;
      description
        "Occuring time of an incident instance.";
    }
  }

  // Data definitions

  container incidents {
    config false;
    description
      "The information of incidents.";
    list incident {
      key "name type incident-id";
      description
        "The information of incident.";
      leaf incident-no {
        type uint64;
        mandatory true;
        description
        "The unique sequence number of the incident instance.";
      }
      uses incident-info;
      uses incident-time-info;
    }
  }
}
<CODE ENDS>
```

9. Security Considerations

The YANG module specified in this document defines a data model that is designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. These YANG-based management protocols (1) have to use a secure transport layer (e.g., SSH [RFC4252], TLS [RFC8446], and QUIC [RFC9000]) and (2) have to use mutual authentication.

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. These are the subtrees and data nodes and their sensitivity/vulnerability:

'/incidents/incident': This list specifies the network incident entries. Unauthorized read access of this list can allow intruders to access network incident information and potentially get a picture of the broken state of the network. Intruders may exploit the vulnerabilities of the network to lead to further negative impact on the network. Care must be taken to ensure that this list is accessed only by authorized users.

Some of the RPC operations in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control access to these operations. These are the operations and their sensitivity/vulnerability:

"incident-diagnose": This RPC operation performs network incident diagnosis and Probable Root Cause locating. If a malicious or buggy client performs an unexpectedly large number of this operation, the result might be an excessive use of system resources [I-D.ietf-nmop-terminology] on the server side as well as network resources. Servers MUST ensure they have sufficient resources to fulfill this request; otherwise, they MUST reject the request using RPC errors defined in section 7.6.

"incident-resolve": This RPC operation is used to resolve the network incident. If a malicious or buggy client performs an unexpectedly large number of this operation, the result might be an excessive use of system resources on the server side as well as network resources. Servers MUST ensure they have sufficient resources to fulfill this request; otherwise, they MUST reject the request without compromise on security of data-at-rest in the server.

10. IANA Considerations

10.1. The "IETF XML" Registry

IANA is requested to register the following URI in the "ns" registry within the "IETF XML Registry" group [RFC3688]:

URI: urn:ietf:params:xml:ns:yang:ietf-incident
Registrant Contact: The IESG.
XML: N/A, the requested URIs are XML namespaces.

10.2. The "YANG Module Names" Registry

IANA is requested to register the following YANG module in the "YANG Module Names" registry [RFC6020] within the "YANG Parameters" registry group.

Name: ietf-incident
Maintained by IANA? N
Namespace: urn:ietf:params:xml:ns:yang:ietf-incident
Prefix: inc
Reference: RFC XXXX

// RFC Ed.: replace XXXX and remove this comment

Acknowledgements

The authors would like to thank Mohamed Boucadair, Robert Wilton, Benoit Claise, Oscar Gonzalez de Dios, Adrian Farrel, Mahesh Jethanandani, Balazs Lengyel, Dhruv Dhody,Bo Wu, Qiufang Ma, Haomian Zheng, YuanYao, Wei Wang, Peng Liu, Zongpeng Du, Zhengqiang Li, Andrew Liu , Joe Clark, Roland Scott, Alex Huang Feng, Kai Gao, Jensen Zhang, Ziyang Xing, Mingshuang Jin, Aihua Guo, Zhidong Yin, Guoxiang Liu, Kaichun Wu for their valuable comments and great input to this work.

References

Normative References

[I-D.ietf-nmop-terminology]
Davis, N., Farrel, A., Graf, T., Wu, Q., and C. Yu, "Some Key Terms for Network Fault and Problem Management", Work in Progress, Internet-Draft, draft-ietf-nmop-terminology-

23, 18 August 2025,
<https://datatracker.ietf.org/doc/html/draft-ietf-nmop-terminology-23>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/rfc/rfc3688>.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/rfc/rfc4252>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/rfc/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/rfc/rfc6241>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/rfc/rfc6991>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/rfc/rfc8040>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/rfc/rfc8341>.

[RFC8345] Clemm, A., Medved, J., Varga, R., Bahadur, N., Ananthakrishnan, H., and X. Liu, "A YANG Data Model for Network Topologies", RFC 8345, DOI 10.17487/RFC8345, March 2018, <https://www.rfc-editor.org/rfc/rfc8345>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

[RFC8632] Vallin, S. and M. Bjorklund, "A YANG Data Model for Alarm Management", RFC 8632, DOI 10.17487/RFC8632, September 2019, <https://www.rfc-editor.org/rfc/rfc8632>.

[RFC8791] Bierman, A., Bj旦rklund, M., and K. Watsen, "YANG Data Structure Extensions", RFC 8791, DOI 10.17487/RFC8791, June 2020, <https://www.rfc-editor.org/rfc/rfc8791>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

Informative References

[BERT] "BERT (language model)", n.d., <https://en.wikipedia.org/wiki/BERT_(language_model)>.

[I-D.ietf-ippm-pam] Mirsky, G., Halpern, J. M., Min, X., Clemm, A., Strassner, J., and J. Fran巽ois, "Precision Availability Metrics for Services Governed by Service Level Objectives (SLOs)", Work in Progress, Internet-Draft, draft-ietf-ippm-pam-09, 1 December 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-ippm-pam-09>.

[I-D.ietf-netconf-configuration-tracing] Quilbeuf, J., Claise, B., Graf, T., Lopez, D., and S. Qiong, "External Trace ID for Configuration Tracing", Work in Progress, Internet-Draft, draft-ietf-netconf-configuration-tracing-06, 3 November 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-configuration-tracing-06>.

[I-D.ietf-netconf-trace-ctx-extension] Gagliano, R., Larsson, K., and J. Lindblad, "NETCONF Extension to support Trace Context propagation", Work in Progress, Internet-Draft, draft-ietf-netconf-trace-ctx-extension-05, 19 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netconf-trace-ctx-extension-05>.

[I-D.ietf-nmop-network-anomaly-architecture] Graf, T., Du, W., Francois, P., and A. H. Feng, "A Framework for a Network Anomaly Detection Architecture", Work in Progress, Internet-Draft, draft-ietf-nmop-network-anomaly-architecture-07, 18 January 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-nmop-network-anomaly-architecture-07>.

[I-D.ietf-opsawg-scheduling-oam-tests] Contreras, L. M., Lopez, V., and Q. Wu, "A YANG Data Model for Network Diagnosis using Scheduled Sequences of OAM Tests", Work in Progress, Internet-Draft, draft-ietf-opsawg-scheduling-oam-tests-03, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-scheduling-oam-tests-03>.

[I-D.irtf-nmrg-ai-challenges] Fran巽ois, J., Clemm, A., Papadimitriou, D., Fernandes, S., and S. Schneider, "Research Challenges in Coupling Artificial Intelligence and Network Management", Work in Progress, Internet-Draft, draft-irtf-nmrg-ai-challenges-05, 18 March 2025, <https://datatracker.ietf.org/doc/html/draft-irtf-nmrg-ai-challenges-05>.

[ITU-T-G-7710] "ITU-T G.7710/Y.1701 - Common equipment management function requirements", 2020, <https://www.itu.int/rec/T-REC-G.7710>.

[ITU-T-X-733] "ITU-T X.733 - Information technology - Open Systems Interconnection - Systems Management - Alarm reporting function", 1999, <https://www.itu.int/rec/T-REC-X.733/fr>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/rfc/rfc7950>.

[RFC8969] Wu, Q., Ed., Boucadair, M., Ed., Lopez, D., Xie, C., and L. Geng, "A Framework for Automating Service and Network Management with YANG", RFC 8969, DOI 10.17487/RFC8969, January 2021, <https://www.rfc-editor.org/rfc/rfc8969>.

[RFC9375] Wu, B., Ed., Wu, Q., Ed., Boucadair, M., Ed., Gonzalez de Dios, O., and B. Wen, "A YANG Data Model for Network and VPN Service Performance Monitoring", RFC 9375, DOI 10.17487/RFC9375, April 2023, <https://www.rfc-editor.org/rfc/rfc9375>.

[RFC9408] Boucadair, M., Ed., Gonzalez de Dios, O., Barguil, S., Wu, Q., and V. Lopez, "A YANG Network Data Model for Service Attachment Points (SAPs)", RFC 9408, DOI 10.17487/RFC9408, June 2023, <https://www.rfc-editor.org/rfc/rfc9408>.

[RFC9417] Claise, B., Quilbeuf, J., Lopez, D., Voyer, D., and T. Arumugam, "Service Assurance for Intent-Based Networking Architecture", RFC 9417, DOI 10.17487/RFC9417, July 2023, <https://www.rfc-editor.org/rfc/rfc9417>.

[RFC9418] Claise, B., Quilbeuf, J., Lucente, P., Fasano, P., and T. Arumugam, "A YANG Data Model for Service Assurance", RFC 9418, DOI 10.17487/RFC9418, July 2023, <https://www.rfc-editor.org/rfc/rfc9418>.

[RFC9543] Farrel, A., Ed., Drake, J., Ed., Rokui, R., Homma, S., Makhijani, K., Contreras, L., and J. Tantsura, "A Framework for Network Slices in Networks Built from IETF Technologies", RFC 9543, DOI 10.17487/RFC9543, March 2024, <https://www.rfc-editor.org/rfc/rfc9543>.

[TMF724A] "Incident Management API Profile v1.0.0", 2023, <https://www.tmforum.org/resources/standard/tmf724a-incident-management-api-profile-v1-0-0/>.

[W3C-Trace-Context] "W3C Recommendation on Trace Context", 2021, <https://www.w3.org/TR/2021/REC-trace-context-1-20211123/>.

Appendix A. Examples of Network Incident Format Representation

A.1. Network Incident Correlated with Specific Network Topology and the Network Service

In this example, we show a nework incident that are associated with the service-instance "optical-svc-A", the node ’D1’, the network topology ’L2-Topo’ and the domain ’FAN’. The Probable Root Cause is also analysed.


```
{
  "incident-no": 56433218,
  "incident-id": "line fault",
  "service-instance": ["optical-svc-A"],
  "domain": "FAN",
  "priority": "critical",
  "occur-time": "2026-03-10T04:01:12Z",
  "clear-time": "2026-03-10T06:01:12Z",
  "ack-time": "2026-03-10T05:01:12Z",
  "last-updated": "2026-03-10T05:31:12Z",
  "ack-status": "unacknowledged",
  "category": "Line",
  "source": [
    {
```

```
        "node-ref": "example:D1",
        "network-ref": "example:L2-topo",
        "resource": [
          {
            "name": "7985e01a-5aad-11ea-b214-286ed488cf99"
          }
        ]
      }
    ],
    "probable-causes": [
      {
        "name": "Feeder fiber great loss change",
        "detail-information": "The connector of the optical fiber
         is contaminated, Or the optical fiber is bent too much.",
        "probable-cause": {
          "network-ref": "example:L2-topo",
          "node-ref": "example:D1",
          "resource": [
            {
              "name": "7985e01a-5aad-11ea-b214-286ed488cf99",
              "cause-name": "ltp",
              "detail": "Frame=0, Slot=6, Subslot=65535, Port=7,
              ODF= ODF001,  Level1Splitter= splitter0025"
            }
          ]
        }
      }
    ],
    "probable-event": [
      {
        "event-id": "8921834",
        "type": "alarm"
      }
    ],
    "events": [
      {
        "event-id": "8921832",
        "type": "alarm"
      },
      {
        "event-id": "8921833",
        "type": "alarm"
      },
      {
        "event-id": "8921834",
        "type": "alarm"
      }
    ]
  }
```


A.2. Network Incident Correlated with Trouble Tickets

In this document, the objective of the Incident Management is to identify probable causes and reduce duplicated tickets.

Traditionally, troubleshooting ticket is created upon critical alert is received, e.g., due to excessive BGP flaps on a particular device by the OSS system. Such troubleshooting ticket will trigger Network Incident Management in the network controller. Therefore normally troubleshooting tickets and network incident are managed by the OSS and the network controller respectively. However Network troubleshooting is sometimes complicated and requires data gathering and analysis from many different tools from the controllers, therefore correlation between troubleshooting ticket and network incident becomes necessary.

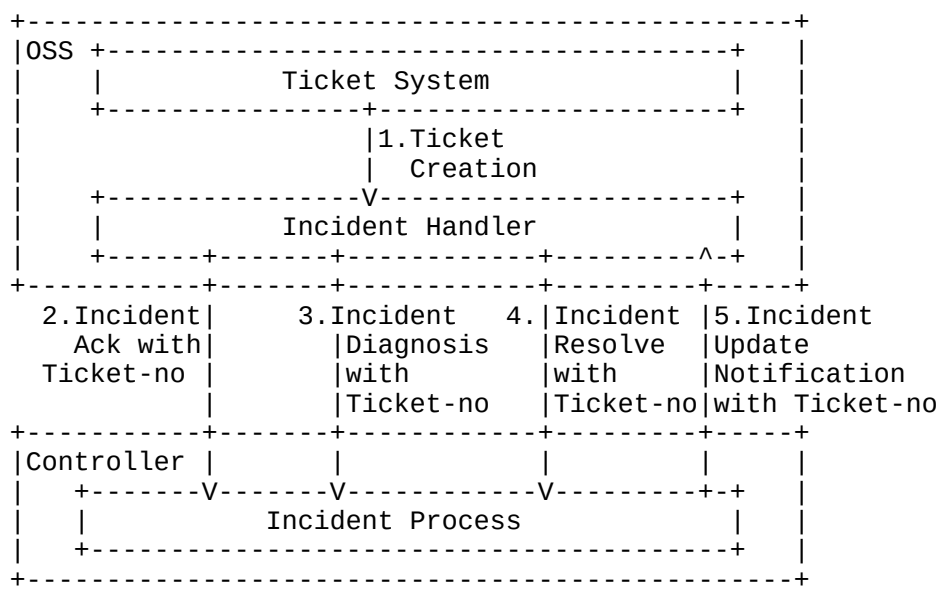


Figure 7: Correlation with troubleshooting tickets

In order to manage the correlation between network incidents and trouble tickets in the YANG data model, three RPCs to manage the network incidents and one notification to report on network incident state changes defined in "ietf-incident" module can be further extended to include "ticket-no" attribute so that such correlation can be carried in the incident update notification and report the upper-layer OSS system. Such correlation can be used by the incident handler in the upper-layer OSS system for further fault demarcation, e.g., identify whether the fault is on the user side or on the network side.

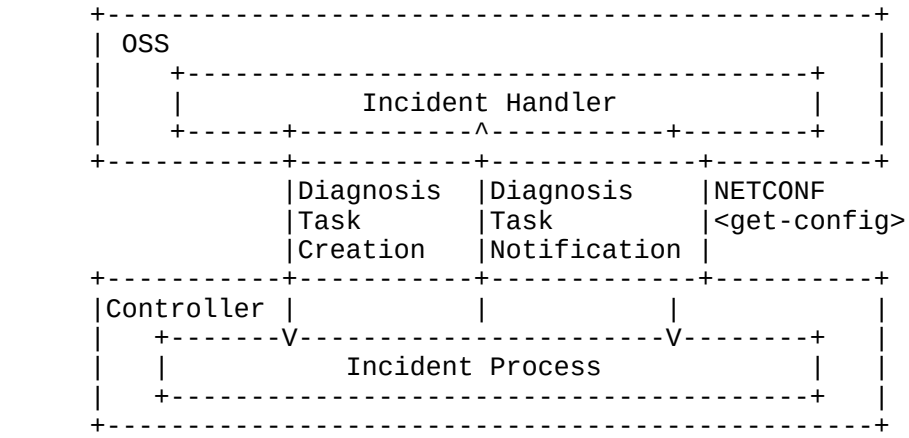
```
rpcs:
 +---x incident-acknowledge
 | +---w input
 |     +---w incident-no* incident-ref
 |     +---w ticket-no? string
 +---x incident-diagnose
 | +---w input
 | |   +---w incident-no* incident-ref
 | |   +---w ticket-no? string
 | +--ro output
 | |   +--ro task-id? string
 +---x incident-resolve
 | +---w input
 |     +---w incident-no* incident-ref
 |     +---w ticket-no? string

 notifications:
 +---n incident-notification
 |   +--ro incident-no? incident-ref
 |   +--ro ticket-no? string
 +--
...
```

A.3. Intent Based Networking with Incident Diagnosis Task List

In this document, the incident-diagnosis RPC defined in "ietf-incident" module can be used to identify Probable Root Causes; and an incident update notification can be triggered to report the diagnosis status if successful.

In some cases, workflows may span a long duration or involve multiple steps task. In such case, intent based networking concept can be

used to support such multiple step task and provide more detailed network diagnosis information.



To do so, the new "diagnosis task creation" RPC can be further defined to support "task-id" attribute in the output parameters and other auxiliary attributes in the input parameters. such RPC can be used to return task-id from the controller. The controller is responsbile for task-id allocation and maintaining task-id list.

```
    +---x diagnose-task-creation
    |  +---w input
    |  |  +---w incident-no?       string
    |  |  +---w ticket-no?         string
    |  |  +---w occur-time?        yang:date-and-time
    |  |  +---w context?           string
    |  |  +---w related-events
    |  |  |  +---w probable-event* []
    |  |  |     +---w type?       leafref
    |  |  |     +---w event-id?   leafref
    |  |  +---w related-objects
    |  |     +---w source* [node-ref]
    |  |        +---w node-ref       leafref
    |  |        +---w network-ref?   leafref
    |  |        +---w resource* [name]
    |  |           +---w name     al:resource
    |  +--ro output
    |     +--ro task-id?   string
```

"ietf-incident" module can be further extended to include "incident-diagnosis-task" list with the following diagnosis information:

o The current status (e.g., created, diagnosing, diagnosed, finished) of each diagnosis task.

o Task start time, end time, diagnosis result (succeeded, failed), failure description, etc.

o Probable Root Causes, probable events, repair recommendations, etc.

so that OSS system can use NETCONF <get-config> operation to look up the diagnosis task detailed information based on such module extension.

```
augment /inc:incidents/inc:incident:
+--ro incident-diagnosis-tasks
|   +--ro incident-diagnosis-task* [task-id]
|   +--ro task-id? String
|   +--ro incident-no* incident-ref
|   +--ro ticket-no? string
|   +--ro start-time? yang:date-and-time
```

```
|   +--ro end-time? yang:date-and-time
|   +--ro task-state? enumeration
|   +--ro diagnosis-result? enumeration
|   +--ro diagnosis-result-description? String
|   +--ro probable-causes leafref //List <RootCause>
...
|   +--ro probable-events leafref //List <Event>
...
|   +-- ro repair-advices
|   +-- ro state enumeration // Incident states such as Creation, Update, Clear
...
```

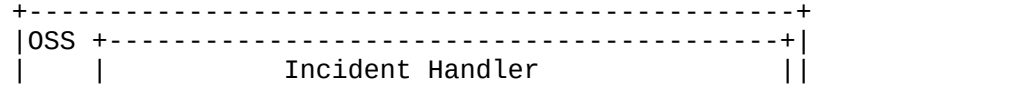
In addition, the new Diagnosis Task Notification can be defined to support Diagnosis Task related attributes reporting.

```
    +---n task-notification
    |  +--ro task-id?                       string
    |  +--ro incident-no?                   string
    |  +--ro ticket-no?                     string
    |  +--ro start-time?                    yang:date-and-time
    |  +--ro end-time?                      yang:date-and-time
    |  +--ro task-state?                    task-state
    |  +--ro diagnosis-result?              diagnosis-result
    |  +--ro diagnosis-result-description?  string
    |  +--ro probable-causes
    |  |  +--ro probable-cause* []
    |  |     +--ro node-ref?      leafref
    |  |     +--ro network-ref?   leafref
    |  |     +--ro resource* [name]
    |  |     |  +--ro name          al:resource
    |  |     |  +--ro cause-name?   identityref
    |  |     |  +--ro detail?       string
    |  |     +--ro cause-name?    identityref
    |  |     +--ro detail?        string
    |  +--ro probable-events
    |  |  +--ro probable-event* []
    |  |     +--ro type?       leafref
    |  |     +--ro event-id?   leafref
    |  +--ro repair-advices?   string
    |  +--ro incident-status?  incident-status-value
```

So that the controller can send diagnosis task notification to the OSS system upon diagnosis task completes and outputs repair suggestion.

A.4. Multi-Domain Fault Demarcation with Network Incident Management

Take multi-domain fault demarcation as an example, when both base station incident in the RAN network and Network Link incident in the IP network are received and base station incident from user side results from network incident in other domains, the OSS system is unable to find network side problem simply based on base station incident. Therefore incident diagnosis RPC will be invoked with IP address of Base station and incident start time as input and sent to the network controller. The network controller can use network diagnosis related intent based interface to find the corresponding network side port according to the base station IP address, and then further associated with transmission path (current path, historical path) to the base station and current and historical network performance, netowrk resources, and incident status data, to diagnose the Probable Root Cause of the network incident and provide repair suggestions.



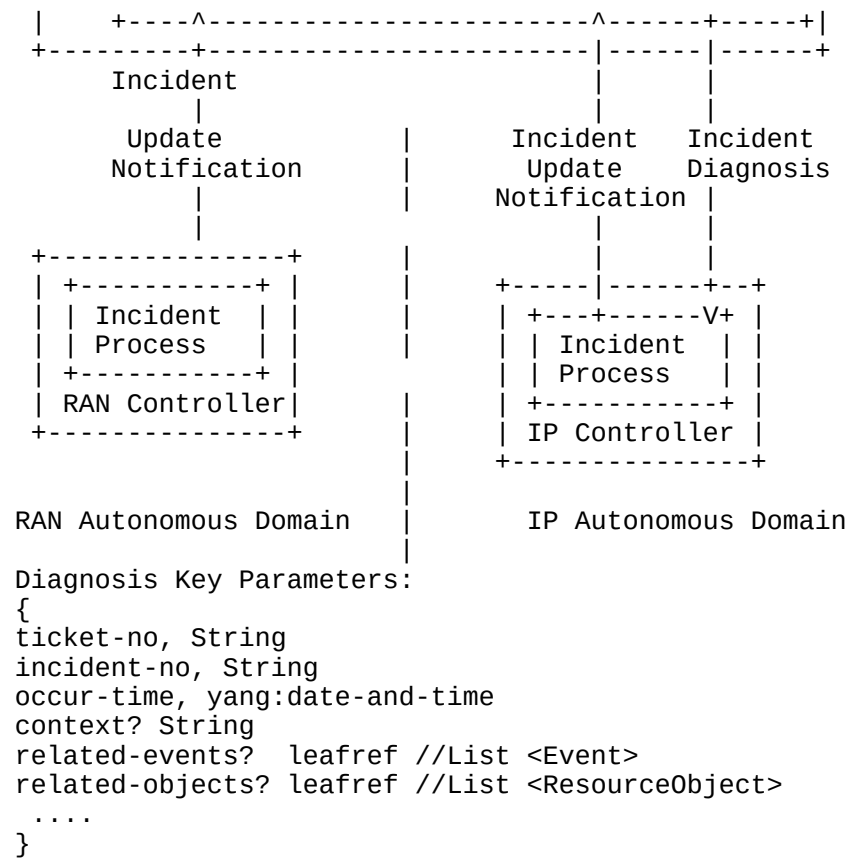


Figure 8: Multi-Domain Fault Demarcation

A.5. Service Complaint triggered Network Diagnosis

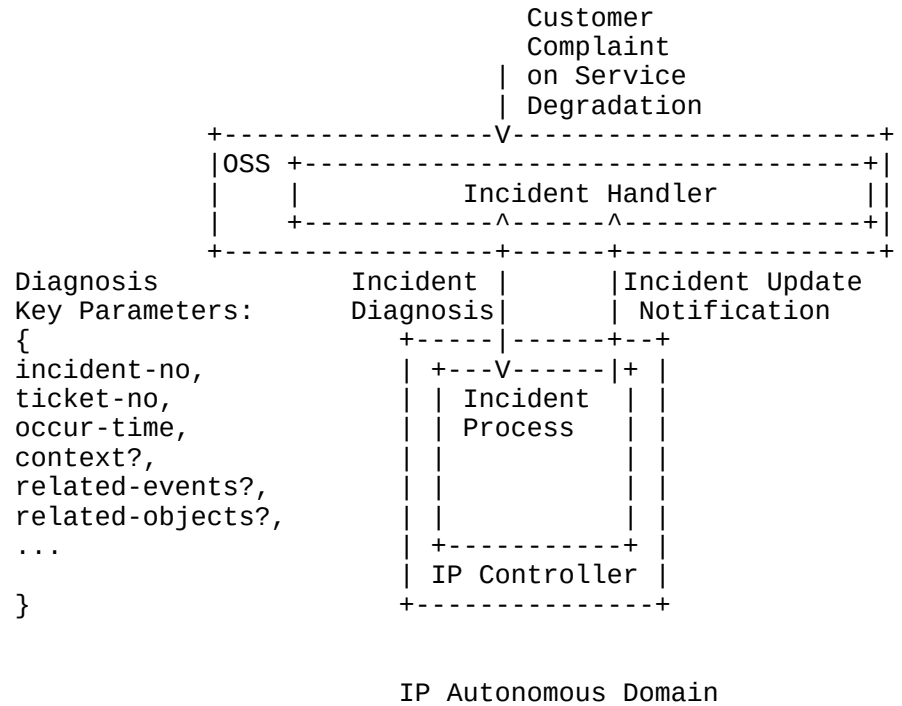


Figure 9: Service Complaint triggered Network Diagnosis

Similarly, in case of service degradation for a lease line service recieving from the customer, the OSS system can request network diagnosis at the network side conducted by the network controller. The network controller can use network diagnosis related intent based interface to find the corresponding network side port based on the

dedicated line service, and then further associate the transmission path (current path, historical path) and current and historical network performance, network resources, and incident status data to diagnose the Probable Root Cause of the fault and provide repair suggestions.

Appendix B. Changes between Revisions

v06 - v07

* Fix Yanglint issue in the YANG data model.

* Align with RFC8407bis section 3.8.3.1 IANA template.

* Align with YANG Module Security Considerations template.

* Probable Root Cause Definition Polishing.

* Tree diagram update for RPC error construct

v05 - v06

* Break down A.3 into 3 sections covering 3 examples.

v04 - v05

* Replace probable cause with probable root cause based on Adrian and Benoit’s suggestion.

* Address editorial comments raised by Aitken Paul.

* YANG Model editorial changes based on Aitken Paul’s comments.

v03 - v04

* Remove constraint of using machine learning for service impact assessment and replace machine learning with algorithmic techniques.

* Replace root cause with probable cause based on IETF 122 NMOP Session Discussion.

* Add two ITU-T references for probable cause definition in the terminologies section.

* Add Lionel Tailhardat from Orange as new contributors based on his input.

* Add two new examples in the Appendix to explore correlation between troubleshooting ticket and incident management and intent based network diagonisis interaction.

v02 - v03

* Cross-checking terminology across NMOP drafts based on Adrian’s comments.

* Align with the Terminology draft based on Thomas’s comments.

* Clarify the relation between the Network Incident, and Customer Incident.

* Add service impact assessment term and its definition.

* Clarify the relation between fault, problem, incident, service.

* Other Editorial changes.

v01 - v02

* Clarify the relation between fault, incident and problem.

* Clarify the relation between fault management and incident management.

* Add clarification text to make draft focus on network level incident management, not be tied with OSS or under the control of OSS.

* Other Editorial changes.

v00 - v01

* Clarify the relationship between incident-no and incident-id.

* Fix Tree Diagram to align with YANG module code change.

* Add json example in the appendix.

* Add failure handling process for rpc error.

* Clarify the relationship between events and cause.

* Clarify synchronous nature of these RPCs.

* Clarify the relationship between inter-layer and inter-domain.

* Refer to terminology draft for terminology alignment.

* Fix pyang compilation issue and yang lint issue.

* Fix Broken ref by using node-ref defined in RFC8345.

* Update YANG data model based on issues raised in issue tracker of the github.

* Shorten the list of authors to 5 based on chairs' comment and move additional authors to top 3 contributors.

* Merge ietf-incident-type.yang into ietf-incident.yang

* Fix enumeration on leaf type

* Clarify the scope in the abstract and introduction and make the scope focus on YANG data model

* Provide text around figure 5 to clarify how the incident server know the real effect on the relevant services.

* Other editorial changes.

v00 (draft-ietf-nmop-network-incident-yang)

* Change draft name from draft-feng-opsawg-incident-management into draft-feng-nmop-netwrok-incident-yang

* Change title into A YANG Data Model for Network Incident Management

* open issues is tracked in https://github.com/billwuqin/network-incident/issues

v03 - v04 (draft-feng-opsawg-incident-management)

* Update incident defintion based on TMF incident API profile specification.

* Update use case on Multi-layer Fault Demarcation based on side meeting discussion and IETF 119 session discussion.

* Update section 5.1 to explain how network incident is generated based on other factors.

* Add one new use cases on Security Events noise reduction based on Situation Awareness.

* Other Editorial changes.

v02 - v03 (draft-feng-opsawg-incident-management)

* Add one new use cases on Incident Generation.

* Add reference to Precision Availability Metric defined in IPPM PAM WG document.

v01 - v02

* A few Editorial change to YANG data models in section 8.

* Add some text to the model design overview.

* Revise sample use cases section to focus on two key use cases.

* Motivation and goal clarification in the introduction section.

v00 - v01 (draft-feng-opsawg-incident-management)

* Modify the introduction.

* Rename incident agent to Incident Server.

* Add the interworking with alarm management.

* Add the interworking with SAIN.

* Add the relationship with RFC8969.

* Add the relationship with observation timestamp and trace context.

* Clarify the incident identification process.

* Modify the work flow of incident diagnosis and resolution.

* Remove identities and typedefs from ietf-incident YANG module, and create a new YANG module called ietf-incident-types.

* Modify ietf-incident YANG module, for example, modify incident-diagnose rpc and incident-resolve rpc.

Contributors

Lionel Tailhardat
Orange
Email: lionel.tailhardat@orange.com

Thomas Graf
Swisscom

Switzerland
Email: thomas.graf@swisscom.com

Zhenqiang Li
CMCC
Email: li_zhenqiang@hotmail.com

Yanlei Zheng
China Unicom
Email: zhengyanlei@chinaunicom.cn

Yunbin Xu
CAICT
Email: xuyunbin@caict.ac.cn

Xing Zhao
CAICT
Email: zhaoxing@caict.ac.cn

Chaode Yu
Huawei
Email: yuchaode@huawei.com

Authors' Addresses

Tong Hu
CMCC
Building A01, 1600 Yuhangtang Road, Wuchang Street, Yuhang District
Hangzhou
311121
China
Email: hutong@cmhi.chinamobile.com

Luis M. Contreras
Telefonica
Madrid
Spain
Email: luismiguel.contrerasmurillo@telefonica.com

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing
210012
China
Email: bill.wu@huawei.com

Nigel Davis
Ciena
Email: ndavis@ciena.com

Chong Feng
Email: fengchonglly@gmail.com