NMOP
Internet-Draft
Intended status: Experimental
Expires: 22 April 2025

V. Riccobene
A. Roberto
Huawei
T. Graf
W. Du
Swisscom
A. Huang Feng
INSA-Lyon
19 October 2024

An Experiment: Network Anomaly Lifecycle
draft-netana-nmop-network-anomaly-lifecycle-04

Abstract

Network Anomaly Detection is the act of detecting problems in the network. Accurately detect problems is very challenging for network operators in production networks. Good results require a lot of expertise and knowledge around both the implied network technologies and the connectivity services provided to customers, apart from a proper monitoring infrastructure. In order to facilitate network anomaly detection, novel techniques are being introduced, including programmatical, rule-based and AI-based, with the promise of improving scalability and the hope to keep a high detection accuracy. To guarantee acceptable results, the process needs to be properly designed, adopting well-defined stages to accurately collect evidence of anomalies, validate their relevancy and improve the detection systems over time, iteratively.

This document describes a well-defined approach on managing the lifecycle process of a network anomaly detection system, spanning across the recording of its output and its iterative refinement, in order to facilitate network engineers to interact with the network anomaly detection system, enable the "human-in-the-loop" paradigm and refine the detection abilities over time. The major contributions of this document are: the definition of three key stages of the lifecycle process, the definition of a state machine for each anomaly annotation on the system and the definition of YANG data models describing a comprehensive format for the anomaly labels, allowing a ~~well-structured~~well-structured exchange of those between all the interested actors.

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 22 April 2025.

Table of Contents

1. Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the Network Management Operations ~~and Management Area Working Group~~ Working Group mailing list (nmop@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/nmop/.

Source for this draft and an issue tracker can be found at https://github.com/network-analytics/draft-netana-nmop-network-anomaly-lifecycle.

2. Status of this document

This document is experimental. The main goal of this document is to propose an iterative lifecycle process to network anomaly detection by proposing a data model for metadata to be addressed at different lifecycle stages.

The experiment consists of verifying whether the approach is usable in real use case scenarios to support proper refinement and adjustments of network anomaly detection algorithms. The experiment can be deemed successful if validated at least with an open-source implementation ~~sucessfully~~successfully applied with real networks.

3. Introduction

In [I-D.ietf-nmop-terminology] ~~A~~ a network anomaly is defined as "an unusual or unexpected event or pattern in network data in the forwarding plane, control plane, or management plane that deviates from the normal, expected behavior".

A network problem is defined as "~~A~~ a state regarded as undesirable and may require remedial action" (see [I-D.ietf-nmop-terminology]).

The main objective of a network anomaly detection system is to identify Relevant States of the network (defined as states that have relevancy for network operators, according to [I-D.ietf-nmop-terminology]~~.~~), as they could lead to problems or might be clear indications of problem already happening.

Commenté [MB1]: Refers to what?

~~That said, i~~It is still remarkably difficult to gain a full understanding and a complete perspective of "if" and "how" a relevant state is actually an indication of a problem or it is just unexpected, but has no impact on service or end users. Anomaly detection providers should aim at increasing accuracy, by minimizing false positives and false negatives. Moreover, the behaviour of the network naturally changes over time, when more connectivity services are deployed, more customers on-boarded, devices are upgraded or replaced, and therefore it is almost impossible to build an anomaly detection system that remains accurate over time.

Commenté [MB2]: ?

Commenté [MB3]: It should be dynamic in nature as the network it tracks :-)

This opens up to the necessity of further validating notified relevant states to check if ~~the~~ a detected Symptom~~s~~ ~~are~~ is actually impacting connectivity services: this might require different actors (both human and algorithmic) to act during the process and refine their understanding across the network anomaly lifecycle.

Finally, once validation has happened, this might lead to refinements to the logic that is used by the detection, so that this process can improve the detection accuracy over time.

Performing network anomaly detection is a process that requires a continuous learning and continuous improvement. Relevant states are detected by aggregating and understanding Symptoms, then validated, confirming that Symptoms actually impacted connectivity services impacting and eventually need to be further analyzed by performing postmortem analysis to identify any potential adjustment to improve the detection capability. Each of these steps represents an opportunity to learn and refine the process, and since

implementations of these steps might also be provided by different parties and/or products, this document also contributes a formal data model to capture and exchange Symptom information across the lifecycle.

a mis en forme : Surlignage

4. Terminology

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.

This document makes use of the terms defined in [I-D.ietf-nmop-terminology].

* State
* Problem
* Event
* Alarm
* Symptom

The following terms are used as defined in [RFC9417].

* Metric
* Intent

The following terms are defined in this document.

* Annotator: Is a human or an algorithm which produces metadata by describing anomalies with Symptoms.
* False Positive: Is a detected anomaly which has been identified during the postmortem to be not anomalous.
* False Negative: Is anomalous but has ~~been~~ not been identified by the anomaly detection system.

5. Defining Desired States

The above definitions of network problem provide the scope for what to be looking for when detecting network anomalies. Concepts like "desirable state" and "required state" are introduced. This poses the attention on a significant problem that network operators have to face: the definition of what is to be considered "desirable" or "undesirable".

It is not always easy to detect if a network is operating in an undesired state at a given point in time. To approach this, network operators can rely on different methodologies, more or less deterministic and more or less sensitive: on the one side, the definition of intents (including Service Level Objectives

and Service Level Agreements) which approaches the problem top-down; on the other side, the definition of Symptoms, by mean of solutions like SAIN [RFC9417], [RFC9418] and [I-D.~~netana~~ietf-nmop-network-anomaly-architecture], which approaches the problem bottom-up. At the center of these approaches, there are the so-called Symptoms, explaining what is not working as expected in the network, sometimes also providing hints towards issues and their causes.

One of the more deterministic approaches is to rely on Symptoms based on measurable service-based KPIs, for example, by using Service Level Indicators, Objectives, and Agreements:

Service Level Agreement (SLA): ~~An SLA is~~ an agreement between parties that a service provider makes to its customers on the behavior of the provided service. SLAs are a tool to define exactly what customers can expect out of the service ~~provided~~ delivered to them. In many cases, SLA breaches also come with contractual penalties.

Service Level Objectives (SLOs): ~~An SLO is~~ a threshold above which the service provider acts to prevent a breach of an SLA. SLOs are a tool for service providers to know when they should start becoming concerned about a service not behaving as expected. SLOs are rarely connected to penalties as they usually are internal metrics for the service providers.

Service Level Indicators (SLIs): ~~An SLI is~~ an observable metric that describes the state of a monitored subsystem. SLIs are a tool to gain measurable visibility about the behavior of a subsystem in the network. SLIs usually differ from SLOs as SLOs are usually expressed as thresholds, while SLIs would often be expressed e.g. as percentages.

However, the definition of these KPIs turns out to be very challenging in some cases, as accurate KPIs could require computationally expensive techniques to be collected or substantial modifications to existing network protocols.

Alternative methodologies rely on Symptoms as the way to generate analytical data out of operational data. For instance:

SAIN introduces the definition and exposure of Symptoms as a mechanism for detecting those concerning behaviors in more deterministic ways. Moreover, the concept of "impact score" has been introduced by SAIN, to indicate what is the expected degree of impact that a given Symptom will have on the services relying on the related subservice to which the Symptom is attached.

Daisy introduces the concept of concern score to indicate what is the degree of concern that a given Symptom could cause a degradation for a connectivity service.

In general, defining boundaries between desirable vs. undesirable in an accurate fashion requires continuous iterations and improvements coming from all the stages of the network anomaly detection lifecycle, by which network engineers can transfer what they learn

Commenté [MB4]: Is this implemented/deployed?

Commenté [MB5]: You may see a more precise def in rfc9543.

Commenté [MB6]: RFC9543 has the following def: «Service Level Indicator (SLI): A quantifiable measure of an aspect of the performance of a network. For example, it may be a measure of throughput in bits per second, or it may be a measure of latency in milliseconds. »

Commenté [MB7]: An example to cite here?

through the process into new Symptom definitions and, ultimately, into refinements of the detection algorithms.

6. Lifecycle of a Network Anomaly

The lifecycle of a network anomaly can be articulated in three phases, structured as a loop: Detection, Validation, Refinement.

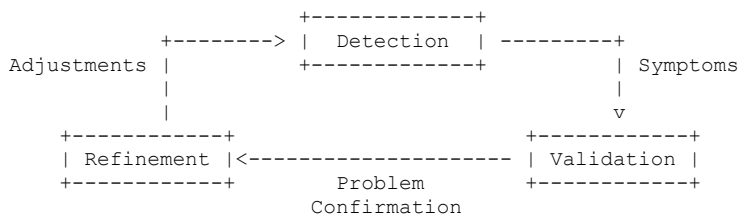


Figure 1: Anomaly Detection Refinement Lifecycle

Each of these phases can either be performed by a network expert or an algorithm or complementing each other.

The network anomaly metadata is generated by an annotator, which can be either a human expert or an algorithm. The annotator can produce the metadata for a network anomaly, for each stage of the cycle and even multiple versions for the same stage. In each version of the network anomaly metadata, the annotator indicates the list of Symptoms that are part of the network anomaly taken into account. The iterative process is about the identification of the right set of Symptoms.

6.1. Network Anomaly Detection

The Network Anomaly Detection stage is about the continuous monitoring of the network through Network Telemetry [RFC9232] and the identification of Symptoms. One of the main requirements that operator have on network anomaly detection systems is the high accuracy. This means having a small number of false negatives, Symptoms causing connectivity service impact are not missed, and false positives, Symptoms that are actually innocuous are not picked up.

As the detection stage is becoming more and more automated for production networks, the identified Symptoms might point towards three potential kinds of behaviors:

i. those that are surely corresponding to an impact on connectivity services, (e.g., the breach of an SLO),

ii. those that will cause problems in the future (e.g., rising trends on a timeseries metric hitting towards saturation),

iii. those or which the impact to ~~connectivuty~~connectivity services cannot be confirmed (e.g., sudden increase/decrease of timeseries metrics, anomalous amounts of log entries, etc.).

The first category requires immediate intervention (a.k.a. the problem is "confirmed"), the second one provides pointers towards early signs of ~~an~~ a problem potentially happening in the near future (a.k.a. the problem is "forecasted"), and the third one requires some analysis to confirm if the detected Symptom requires any attention or immediate intervention (a.k.a. the problem is "potential"). As part of the iterative improvement required in this stage, one that is very relevant is the gradual conversion of the third category into one of the first two, which would make the network anomaly detection system more deterministic. The main objective is to reduce uncertainty around the raised alarms by refining the detection algorithms. This can be achieved by either generating new Symptom definitions, adjusting the weights of automated algorithms or other similar approaches.

6.2. Network Anomaly Validation

The key objective for the validation stage is clearly to decide if the detected Symptoms are signaling a real problem (a.k.a. requires action) or if they are to be treated as false positives (a.k.a. suppressing the alarm). For those Symptoms surely having impact on connectivity services, 100% confidence on the fact that a network problem is happening can be assumed. For the other two categories, "forecasted" and "potential", further analysis and validation is required.

6.3. Network Anomaly Refinement

After validation of a problem, the service provider performs troubleshooting and resolution of the problem. Although the network might be back in a desired state at this point, network operators can perform detailed postmortem analysis of network problems with the objective to identify useful adjustments to the prevention and detection mechanisms (for instance improving or extending the definition of SLIs and SLOs, refining concern/impact scores, etc.), and improving the accuracy of the validation stage (e.g. automating parts of the validation, implementing automated root cause analysis and automation for remediation actions). In this stage of the lifecycle it is assumed that the problem is under analysis.

After the adjustments are performed to the network anomaly detection methods, the cycle starts again, by "replaying" the network anomaly and checking if there is any measurable improvement in the ability to detect problems by using the updated method.

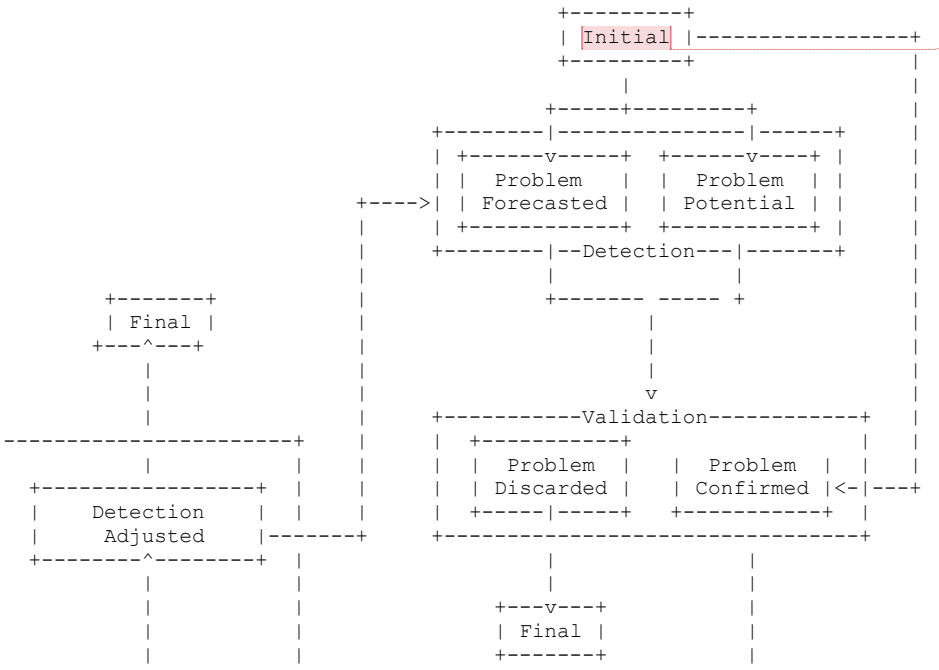
7. Network Anomaly State Machine

In the context of this document, from a network anomaly detection point of view a network problem is defined as a collection of interrelated Symptoms, as specified in [I-D.netana-nmop-network-anomaly-semantics].

The understanding of a network problem can change over time. Moreover, multiple actors are involved in the process of refining this understanding in the different phases.

From this perspective, a problem can be refined according to the

following states (Figure 2).



Commenté [MB8]: Where in hierarchy new problems are added to capitalize on recent incidents, etc.?

Figure 2: Network Anomaly State Machine

The knowledge gained at each stage is codified as a list of anomaly labels that can be stored on a Label Store (see Section 9.1 for a reference).

8. Network Anomaly Data Model

~~In this section, the proposed data model is described.~~

8.1. Overview of the Data Model for the Relevant State and all the related entities

```
module: ietf-relevant-state
+--rw relevant-state
  +--rw id                              yang:uuid
  +--rw description?                    string
  +--rw start-time                      yang:date-and-time
  +--rw end-time?                       yang:date-and-time
  +--rw anomalies* [id version]
      +--rw id                                yang:uuid
      +--rw version                         yang:counter32
      +--rw state                           identityref
      +--rw description?                      string
      +--rw start-time                        yang:date-and-time
      +--rw end-time?                         yang:date-and-time
      +--rw confidence-score                  score
      +--rw (pattern)?
      |  +--:(drop)
      |  |  +--rw drop?                       empty
      |  +--:(spike)
      |  |  +--rw spike?                      empty
      |  +--:(mean-shift)
      |  |  +--rw mean-shift?                 empty
      |  +--:(seasonality-shift)
      |  |  +--rw seasonality-shift?               empty
      |  +--:(trend)
      |  |  +--rw trend?                      empty
      |  +--:(other)
      |     +--rw other?                      string
      +--rw annotator!
      |  +--rw name                           string
      |  +--rw (annotator-type)?
      |     +--:(human)
      |     |  +--rw human?                   empty
      |     +--:(algorithm)
      |        +--rw algorithm?               empty
      +--rw symptom!
      |  +--rw id                               yang:uuid
      |  +--rw concern-score                    score
      +--rw service!
```

Commenté [MB9]: Indicate whether this is a network or device model

Commenté [MB10]: Should this be a list?

```
          +--rw id                    yang:uuid

      notifications:
        +---n relevant-state-notification
          +--ro id                          yang:uuid
          +--ro description?                string
          +--ro start-time                  yang:date-and-time
          +--ro end-time?                   yang:date-and-time
          +--ro anomalies* [id version]
              +--ro id                      yang:uuid
              +--ro version                   yang:counter32
              +--ro state                     identityref
              +--ro description?              string
              +--ro start-time                yang:date-and-time
              +--ro end-time?                 yang:date-and-time
              +--ro confidence-score          score
              +--ro (pattern)?
              |  +--:(drop)
              |  |  +--ro drop?                    empty
              |  +--:(spike)
              |  |  +--ro spike?                   empty
              |  +--:(mean-shift)
              |  |  +--ro mean-shift?                      empty
              |  +--:(seasonality-shift)
              |  |  +--ro seasonality-shift?       empty
              |  +--:(trend)
              |  |  +--ro trend?                   empty
              |  +--:(other)
              |     +--ro other?                   string
              +--ro annotator!
              |  +--ro name                             string
              |  +--ro (annotator-type)?
              |     +--:(human)
              |     |  +--ro human?                     empty
              |     +--:(algorithm)
              |        +--ro algorithm?                 empty
              +--ro symptom!
              |  +--ro id                               yang:uuid
              |  +--ro concern-score                    score
              +--ro service!
                +--ro id                                yang:uuid
```

Figure 3: YANG tree diagram for ietf-relevant-state

```
<CODE BEGINS> file "ietf-relevant-state@2024-10-18.yang"
  module ietf-relevant-state {
    yang-version 1.1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-relevant-state";
    prefix rsn;

    import ietf-yang-types {
      prefix yang;
      reference "RFC 6991: Common YANG Data Types";
    }

    organization
      "IETF NMOP Working Group";
```

```
        contact
          "WG Web:   <https://datatracker.ietf.org/wg/nmop/>
          WG List:  <mailto:nmop@ietf.org>

          Authors:  Vincenzo Riccobene
                    <mailto:vincenzo.riccobene@huawei-partners.com>
                    Antonio Roberto
                    <mailto:antonio.roberto@huawei.com>
                    Thomas Graf
                    <mailto:thomas.graf@swisscom.com>
                    Wanting Du
                    <mailto:wanting.du@swisscom.com>
                    Alex Huang Feng
                    <mailto:alex.huang-feng@insa-lyon.fr>";
        description
            "This module defines the relevant-state container and
                notifications to be used by a network anomaly detection
                system. The defined objects can be used to augment
                operational network collected observability data and
            analytical problem data equally. Describing the relevant-state
                    of observed symptoms.

            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; see the RFC
            itself for full legal notices.";

        revision 2024-10-18 {
          description
            "Initial version";
          reference
            "RFCXXXX: Experiment: Network Anomaly Postmortem Lifecycle";
        }

        typedef score {
            type uint8 {
                range "0 .. 100";
            }
        }
        identity network-anomaly-state {

          description
            "Base identity for representing the state of the network
    anomaly";
        }
        identity network-anomaly-detection {
          base network-anomaly-state;
          description
          "A problem reached detection state";
```

Commenté [MB11]: Np need to repeat the prefix of the base identity

```
}
identity network-anomaly-validation {
  base network-anomaly-state;
  description
  "A problem reached validation state";
}
identity network-anomaly-refinement {
  base network-anomaly-state;
  description
  "A problem reached refinement state";
}
  identity problem-forecasted {
  base network-anomaly-detection;
  description
  "A problem has been forecasted, as it is expected that
  the indicated list of symptoms will impact a
      connectivity service in the near future";
}
identity problem-potential {
  base network-anomaly-detection;
  description
  "A problem has been detected with a confidence
  lower than 100%. In order to confirm that this set of
  symptoms are generating connectivity service impact, it
      requires further validation";
}
identity problem-confirmed {
  base network-anomaly-validation;
  description
  "After validation, the problem has been confirmed";
}
identity discarded {
  base network-anomaly-validation;
  description
  "After validation, the network anomaly has been
  discarded, as there is no evindence that it is causing an
  problem";
}
identity analyzed {
  base network-anomaly-refinement;
  description
  "The anomaly detection went through analysis to identify
  potential ways to further improve the detection process in
  for future anomalies";
}
identity adjusted {
  base network-anomaly-refinement;
  description
  "The network anomaly has been solved and analysedanalyzed.
  No further action is required.";
}

grouping relevant-state-grouping {
    leaf id {
        mandatory true;
        type yang:uuid;
        description
            "Unique ID of the fault";
```

Commenté [MB12]: Indicate th uniqueness scope.

```
        }
        leaf description {
            type string;
            description
                "Textual description of the fault";
        }
        leaf start-time {
            type yang:date-and-time;
            mandatory true;
            description
                "Date and time indicating the beginning of the fault";
        }
        leaf end-time {
            type yang:date-and-time;
            description
                "Date and time indicating the end of the fault";
        }
    }

    grouping annotator-grouping {
        leaf name {
            mandatory true;
            type string;
        }
        choice annotator-type {
            case human {
                leaf human {
                    type empty;
                }
            }
            case algorithm {

                leaf algorithm {
                    type empty;
                }
            }
        }
    }

    grouping symptom-grouping {
        leaf id {
            type yang:uuid;
            mandatory true;
            description
                "Unique ID of the symptom type";
        }
        leaf concern-score {
            type score;
            mandatory true;
        }
    }

    grouping service-grouping {
        leaf id {
            type yang:uuid;
            mandatory true;
            description
```

Commenté [MB13]: Description is missing

Commenté [MB14]: Idem

Commenté [MB15]: No need to suffix with the type!

```
                    "Unique ID of the connectivity service (or monitored
entity)";
                }
            }

        grouping anomaly~~-grouping~~ {
            list anomalies {
                key "id version";
                leaf id {
                    type yang:uuid;
                    description
                        "Unique ID of the anomaly";
                }
                leaf version {
                    type yang:counter32;
                    description
                          "Version of the problem metadata object.
                         It allows multiple versions of the metadata to be
                         generated in order to support the definition of
                          multiple problem objects from the same source to
                          facilitate improvements overtime";
                }
                leaf state {
                    type identityref {
                       base network-anomaly-state;
                    }
                    mandatory true;
                    description "State of the anomaly.";
                }
                leaf description {
                    type string;
                    description
                        "Textual description of the anomaly";
                }
                leaf start-time {
                    type yang:date-and-time;
                    mandatory true;
                    description
                        "Date and time indicating the beginning of the
anomaly";
                }
                leaf end-time {
                    type yang:date-and-time;
                    description
                        "Date and time indicating the end of the anomaly";
                }
                leaf confidence-score {
                    type score;
                    mandatory true;
                }
                choice pattern {
                    description
                        "Network Plane affected by the symptom";
                    case drop {
                        leaf drop {
                            type empty;
                        }
                    }
```

Commenté [MB16]: Why a grouping is defined here with one single leaf?

Commenté [MB17]: Does this identify an occurrence or a type?

If an occurrence, not sure how the version is useful

Commenté [MB18]: This may not be known. No?

I would remove the mandatory statement

```
                    case spike {
                        leaf spike {
                            type empty;
                        }
                    }
                    case mean-shift {
                        leaf mean-shift {
                            type empty;
                        }
                    }
                    case seasonality-shift {
                        leaf seasonality-shift {
                            type empty;
                        }
                    }
                    case trend {
                        leaf trend {
                            type empty;
                        }
                    }
                    case other {
                        leaf other {
                            type string;
                            description "specify the type";
                        }
                    }
                }
                container annotator {
                    presence "It specifies an annotator for the anomaly";
                    uses annotator-grouping;
                }
                container symptom {
                    presence "It specifies the symptom for the anomaly";
                    uses symptom-grouping;
                }
                container service {
                    presence "It specifies the connectivity service (or
the monitored entity) affected by the anomaly";
                    uses service-grouping;
                }
            }
        }

        notification relevant-state-notification {
            uses relevant-state-grouping;
            uses anomaly-grouping;
        }

        container relevant-state {
            uses relevant-state-grouping;
            uses anomaly-grouping;
        }
      }
   <CODE ENDS>
```

Commenté [MB19]: I would used identities here unless you are sure that this is a definitive list.

Figure 4: YANG module for ietf-relevant-state

The model has been defined based on a list of requirements, defined

in the following. The data model:

must be semantically consistent from a networking point of view;

must be both human and machine readable;

must allow different detection, validation and refinement solutions to interoperate;

must provide support the "human-in-the-loop" ~~paradigm~~, allowing for network experts to validate and adjust network anomaly labels and detection systems.

The base for the modules is the relevant-state data model. Relevant state is at the root of the data model, with its parameters (ID, description, start-time, end-time) and a collection of anomalies. This allows the relevant state to be considered as a container of anomalies.

Each anomaly is characterized by some intrinsic fields (such as id, version, state, description, start-time, end-time, confidence score and pattern) Particularly the confidence score is a measure of how confident was the detector in considering the given anomaly as an anomalous behaviour.

Each anomaly also ~~include~~includes the symptom and the service container.
These containers are placeholders to represent the information about the symptom (what is ~~exaclty~~exactly happening as anomalous behaviour) and
the connectivity service (what entity is affected by the anomaly). In particular, for what concerns the symptom, a concern score is defined, which has the meaning of expressing how much the anomaly is impacting connectivity services. Based on the intentions of the authors of this data model, the correct use of the symptom and service containers is to augment the model by providing a more accurate implementation of both of them. An example of this is provided in [I-D.netana-nmop-network-anomaly-semantics], where an augmentation of Symptom for connectivity services is proposed.

Also, a list of various actors that can be involved in the process is presented as following:

In the detection stage: the detectors can be Network Engineers and/or Automatic detectors (including Rule-based detectors and ML-based detectors)

In the validation stage: the validators can be Network Engineers manually validating the labels

In the refinement stage: the refiners can be Data Scientists and/or Automatic Refiners (including systems that automatically refine the detection systems, based on the validated labels).

The data model that has been defined is used in two YANG modules: the relevant-state-notification and the ietf-relevant-state: the notification is primarily used by the Network Anomaly Detector, to notify the "Alarm and Problem Management System" and the "Post-mortem

Commenté [MB20]: What does that mean?

Commenté [MB21]: Not sure this adds much value.

Commenté [MB22]: How is this met?

Commenté [MB23]: Should a control knob be supported to let some to be fully automated while other to be subject to validation, etc.?

a mis en forme : Surlignage

System" (see [I-D.~~netana~~ietf-nmop-network-anomaly-architecture]); the container instead is used inside the Post-mortem system to ~~exhance~~exchange anomaly detection ~~lables~~labels between the anomaly detection stages defined above (~~valdation~~validation, refinement, detection).

9. Implementation status

This section provides pointers to existing open source implementations of this draft. Note to the RFC-editor: Please remove this before publishing.

9.1. Antagonist

An ~~open source~~open-source implementation for this draft is called AnTagOnIst (Anomaly Tagging On hIstorical data), and it has been implemented in order to validate the application of the YANG model defined in this draft. Antagonist provides visual support for two important use cases in the scope of this document:

* the generation of a ground truth in relation to symptoms and problems in timeseries data

* the visual validation of results produced by automated network anomaly detection tools.

The ~~open source~~open-source code can be found here: [Antagonist]

As part of the experiment that was conducted with AnTagOnIst, Some main Use Case scenarios have been validated so far:

Exposure of a GUI for human validation of the labels.

Integration with Rule Based anomaly detection systems. In particular the integration with SAIN and Cosmos Bright Lights is ~~on going~~ongoing.

Integration with ML-based detection systems.

10. Security Considerations

The security considerations will have to be updated according to "https://wiki.ietf.org/group/ops/yang-security-guidelines".

11. Acknowledgements

The authors would like to thank xxx for their review and valuable comments.

12. Normative References