Semantic Metadata Annotation for Network Anomaly Detection
draft-netana-nmop-network-anomaly-semantics-01

Experiment: Network Anomaly Lifecycle
draft-netana-nmop-network-anomaly-lifecycle-01

Helps to annotate operational data, refine outlier detection, supports supervised and semi-supervised machine learning development, enables data exchange among network operators, vendors and academia, and make anomalies for humans apprehensible

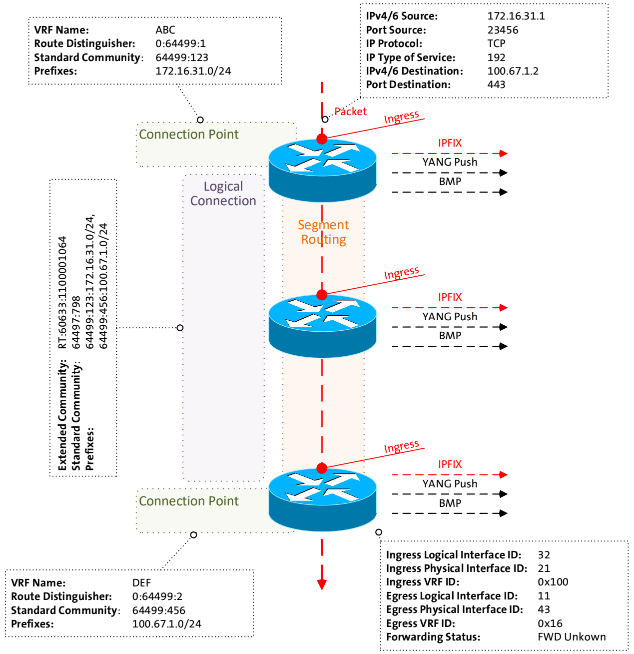
thomas.graf@swisscom.com
wanting.du@swisscom.com
alex.huang-feng@insa-lyon.fr
vincenzo.riccobene@huawei-partners.com
antonio.roberto@huawei.com

15. March 2024

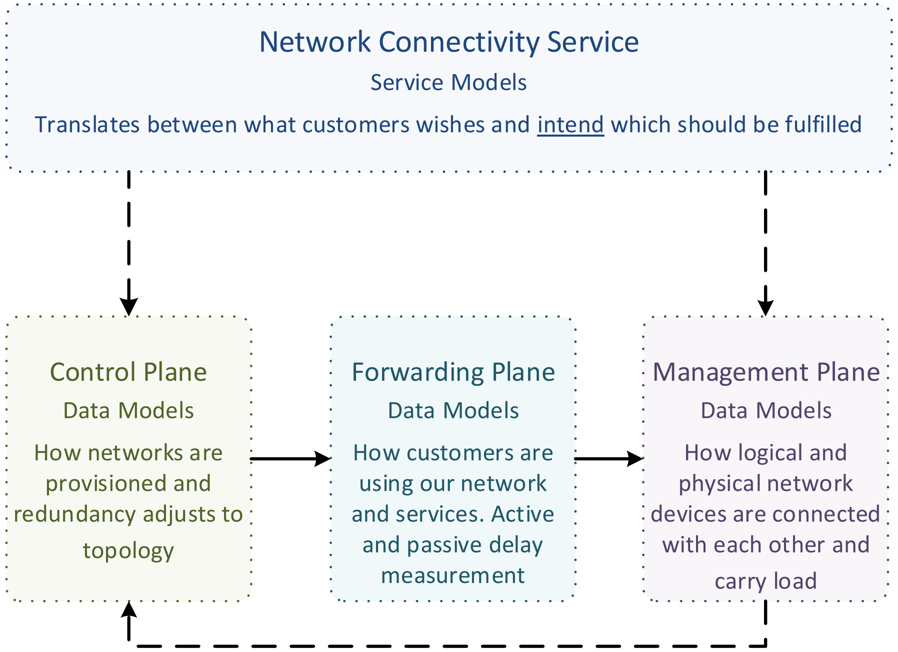
What to monitor

Which operational metrics are collected

« Network operators connect customers in routing tables called VPN's »

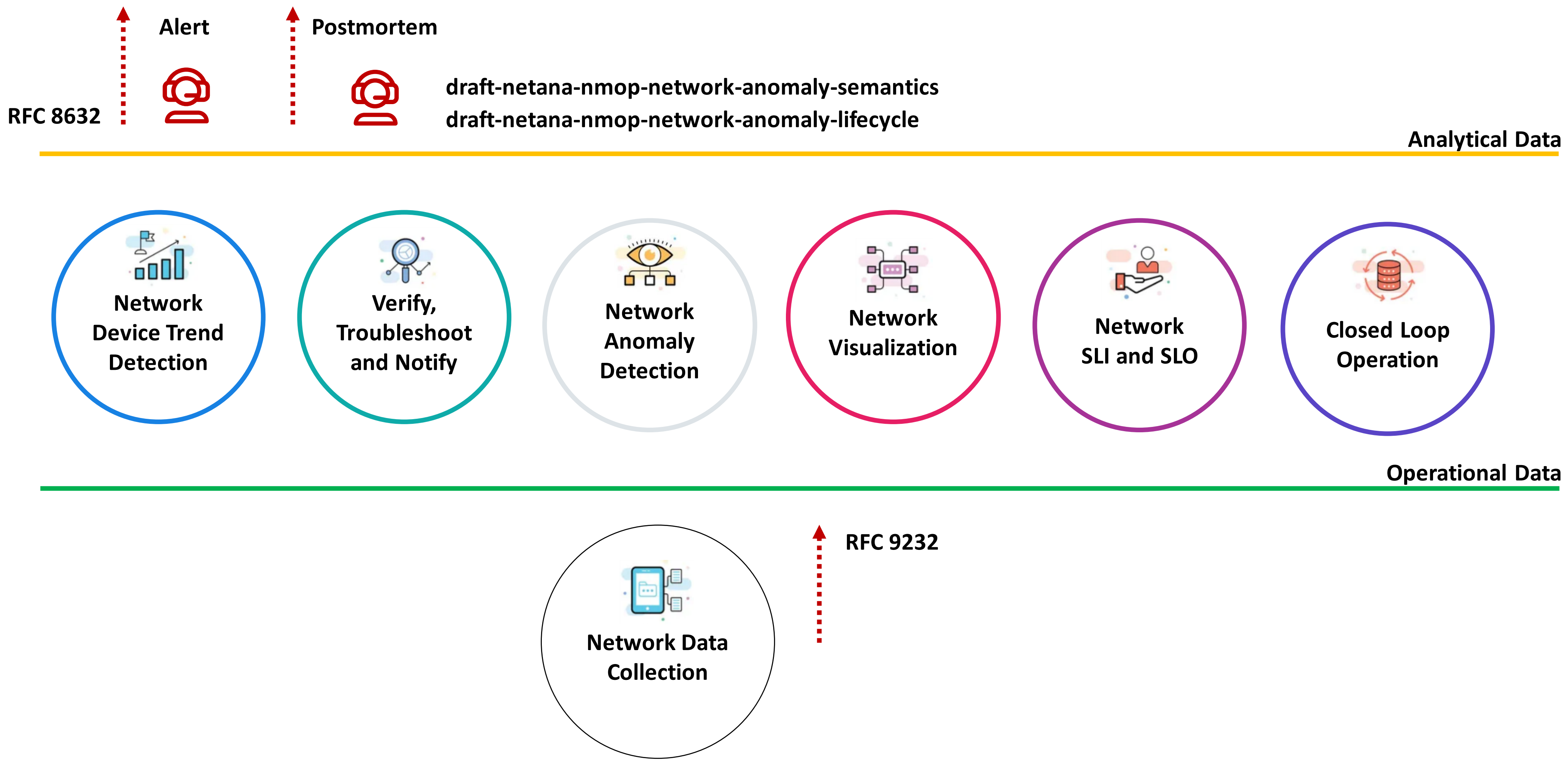


« Network Telemetry (RFC 9232) describes how to collect data from all 3 network planes efficiently »



How to organize and collaborate with data

The Data Mesh Architecture enables Network Analytics use



What does Network Anomaly Detection mean

Monitor changes

Network Anomaly Detection

For VPNs, Network Anomaly Detection **constantly monitors and detects any network or device topology changes**, along with their associated forwarding consequences for customers as outliers. Notifications are sent to the Network Operation Center before the customer is aware of service disruptions. **It offers operational metrics for in-depth analysis,** allowing to understand on which platform the problem originates and facilitates problem resolution.

Answers

What changed and when, on which connectivity service, and how does it impact the customers?

Focuses

Provides meaningful connectivity service impact information before customer is aware of and support in root-cause analysis.

Data Mesh

Consumes operational real-time Forwarding Plane, Control Plane and Management Plane metrics and produces analytical alerts.

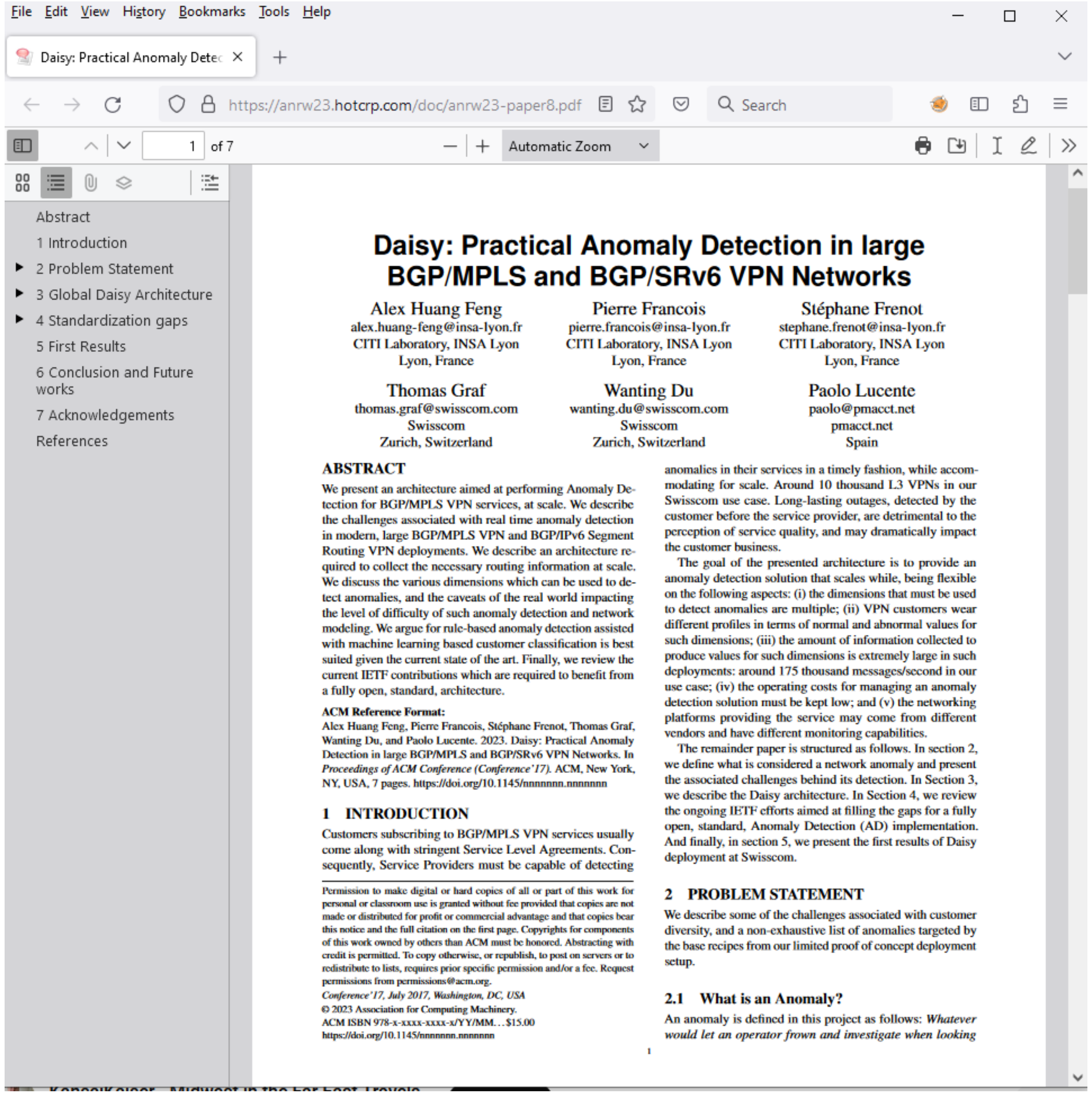
Direction

From connectivity service to network platform.

Presented in ANRW 2023

At IETF 117 San Francisco

« A more detailing paper will be submitted soon to IEEE Transactions on Network and Service Management»



Daisy: Practical Anomaly Detection in large BGP/MPLS and BGP/SRv6 VPN Networks

Alex Huang Feng, alex.huang-feng@insa-lyon.fr, CITI Laboratory, INSA Lyon, Lyon, France
Pierre Francois, pierre.francois@insa-lyon.fr, CITI Laboratory, INSA Lyon, Lyon, France
Stéphane Frenot, stephane.frenot@insa-lyon.fr, CITI Laboratory, INSA Lyon, Lyon, France
Thomas Graf, thomas.graf@swisscom.com, Swisscom, Zurich, Switzerland
Wanting Du, wanting.du@swisscom.com, Swisscom, Zurich, Switzerland
Paolo Lucente, paolo@pmacct.net, pmacct.net, Spain

ABSTRACT

We present an architecture aimed at performing Anomaly Detection for BGP/MPLS VPN services, at scale. We describe the challenges associated with real time anomaly detection in modern, large BGP/MPLS VPN and BGP/IPv6 Segment Routing VPN deployments. We describe an architecture required to collect the necessary routing information at scale. We discuss the various dimensions which can be used to detect anomalies, and the caveats of the real world impacting the level of difficulty of such anomaly detection and network modeling. We argue for rule-based anomaly detection assisted with machine learning based customer classification is best suited given the current state of the art. Finally, we review the current IETF contributions which are required to benefit from a fully open, standard, architecture.

ACM Reference Format:
Alex Huang Feng, Pierre Francois, Stéphane Frenot, Thomas Graf, Wanting Du, and Paolo Lucente. 2023. Daisy: Practical Anomaly Detection in large BGP/MPLS and BGP/SRv6 VPN Networks. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Customers subscribing to BGP/MPLS VPN services usually come along with stringent Service Level Agreements. Consequently, Service Providers must be capable of detecting anomalies in their services in a timely fashion, while accommodating for scale. Around 10 thousand L3 VPNs in our Swisscom use case. Long-lasting outages, detected by the customer before the service provider, are detrimental to the perception of service quality, and may dramatically impact the customer business.

The goal of the presented architecture is to provide an anomaly detection solution that scales while, being flexible on the following aspects: (i) the dimensions that must be used to detect anomalies are multiple; (ii) VPN customers wear different profiles in terms of normal and abnormal values for such dimensions; (iii) the amount of information collected to produce values for such dimensions is extremely large in such deployments: around 175 thousand messages/second in our use case; (iv) the operating costs for managing an anomaly detection solution must be kept low; and (v) the networking platforms providing the service may come from different vendors and have different monitoring capabilities.

The remainder paper is structured as follows. In section 2, we define what is considered a network anomaly and present the associated challenges behind its detection. In Section 3, we describe the Daisy architecture. In Section 4, we review the ongoing IETF efforts aimed at filling the gaps for a fully open, standard, Anomaly Detection (AD) implementation. And finally, in section 5, we present the first results of Daisy deployment at Swisscom.

2 PROBLEM STATEMENT

We describe some of the challenges associated with customer diversity, and a non-exhaustive list of anomalies targeted by the base recipes from our limited proof of concept deployment setup.

2.1 What is an Anomaly?

An anomaly is defined in this project as follows: *Whatever would let an operator frown and investigate when looking*

1

What our motivation is

Automate learn and improve

From network incidents postmortems we network operators **learn and improve** so does network anomaly detection and supervised and semi-supervised machine learning.

The more network incidents are observed, the more we can improve. With more incidents the **postmortem process needs be automated, let's get organized** first by defining human and machine-readable metadata semantics and annotate operational and analytical data.

Let's get further organized by exchanging standardized labeled network incident data among network operators, vendors and academia to **collaborate on academic research**.

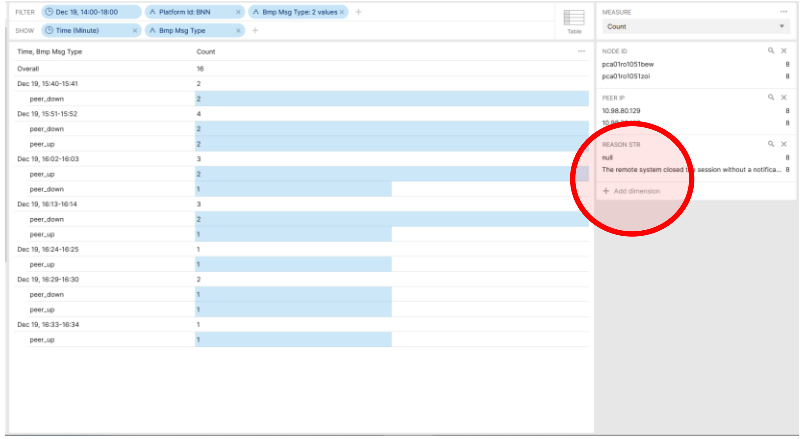
« The community working on Network Anomaly Detection is probably the only group wishing for more network incidents »

Postmortem, Maximum Prefix BGP Peer State Change

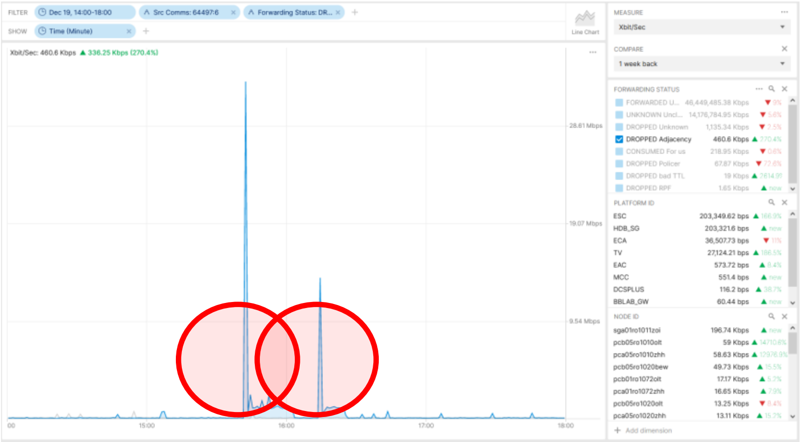
SBInfo-028166, PBI000000193943, INC000012284550

Missing Traffic 64497:6

Flow Count Drop 64497:6



BMP Peer State Change 64497:6



Traffic Drop 64497:6

IPFIX configured on PE and Inter-AS Option A ASBR nodes.

Traffic Drop with Reason Code Adjacency at TV was unrelated.

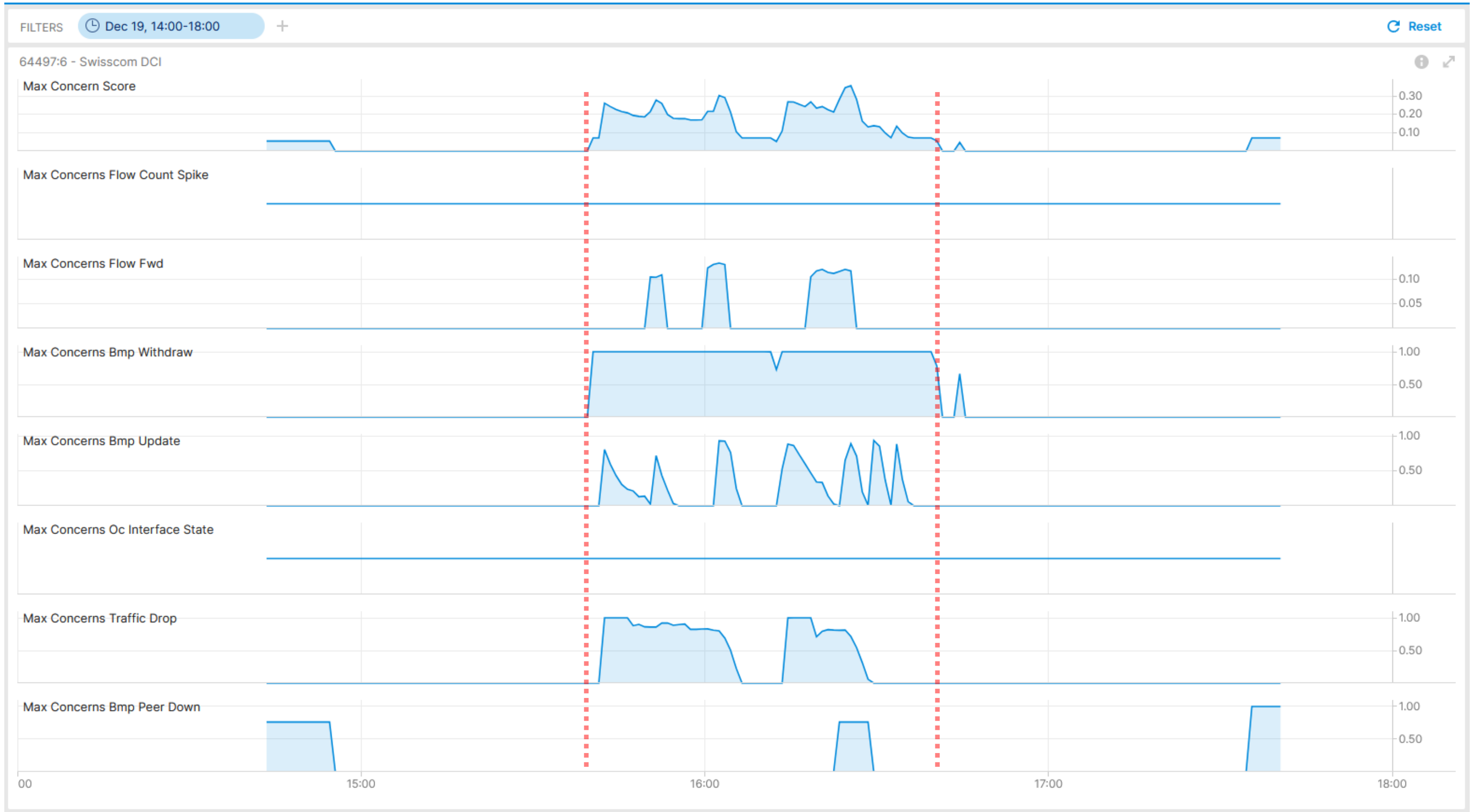
BMP ADJ-RIB In pre-policy on BGP VPNv4 /6 and IPv4/6 VRF unicast peers configured on MPLS PE's. BMP ADJ-RIB In pre-policy on BGP VPNv4 /6 on Route Reflectors.

BMP peer_down reports that it is type 4 (Remote system closed, no data) instead of type 1 (Local system closed, NOTIFICATION PDU follows) due to CSCwi61922.

Postmortem, Maximum Prefix BGP Peer State Change

SBInfo-028166, INC000012284550, Bright Lights Live

Max Concern Score: **0.36**
Traffic Drop: **1.0**
Missing Traffic: **0.13**
BMP Update/Withdraw: **1.0**
BMP Peer Down: **0.76**



Cosmos Bright Lights Anomaly Detection – 64497:6 SC-DCI

BMP route-monitoring Update/Withdraw recognized topology change.

BMP peer Down recognized peering state change delayed due to potential data processing lag.

Interface Down/Up check did not apply.

Traffic Drop check recognized forwarding drop.

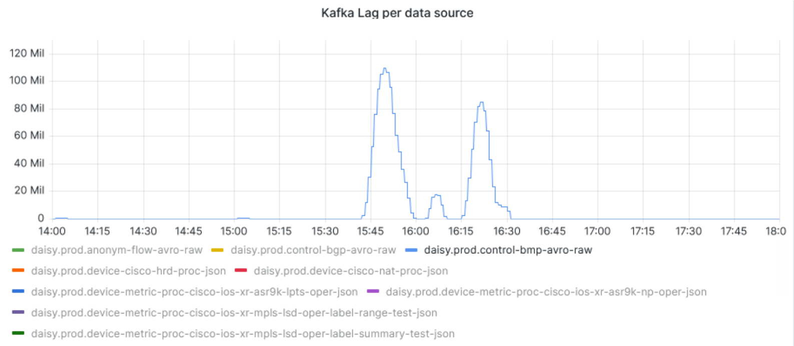
Missing Traffic recognized that connectivity is impaired.

Flow Count Spike did not apply.

Overall: 4 out of 6 checks have detected a customer impact inside of monitoring domain. Works as designed.

Postmortem
What to do next?

- **Record incident in Cosmos Bright Lights lab. -> Done!**
- **Analyze why (TSDB ingestion delay?) not all BMP peer_down where being recognized by BMP peer_down check.**



What went well?

Anomaly Detection rules detected outage based on BMP update/withdrawal and peer_down, IPFIX flow count drop, traffic drop and missing traffic. Works as designed.

What could be improved?

Consider to implement capacity management and trend detection analytical use case for BGP max prefix configured peers, BGP Local RIB path count and BGP process memory.

draft-ietf-grow-bmp-rel authors considering to support two reason code TLV's for prefixes crossing the warning and the maximum threshold.

draft-msri-grow-bmp-bgp-rib-stats authors contacted at GROW to consider another BMP statistics definition describing how many percent of the configured maximum prefix count has been reached.

Similar as we are draft-ietf-grow-bmp-path-marking-tlv how the BGP path will be installed into the RIB, we could add as a TLV also the local allocated MPLS label from the Label FIB.

BMP peer_down reason code is 4 instead of 1 on Cisco IOS XR. Addressed and confirmed in SR 696692110. CSCwi61922 bugfix verified.

BGP notification sub-code support in NetGauze verified.

What is a symptom and how to categorize them

From action to reason to cause

Action: Which action the network node performed for a packet in the forwarding plane, a path or adjacency in the control plane or state or statistical changes in the management plane.

Reason: For each reason one or more actions describing why this action was used. From drop unreachable, administered, and corrupt in forwarding plane, to reachability withdraw and adjacency teared down in control plane, to Interface down, errors or discard in management plane.

Cause: For each reason one or more causes describes why the action was chosen. From missing next-hop and link-layer information in forwarding plane, to reachability withdrawn due to peer down or path no longer redistributed.

« Symptoms are categorized in which plane they have been observed, their action, reason and cause »

Outliers in Anomaly Detection

From global to contextual to collective

Global outliers: An outlier is considered "global" if its behavior is outside the entirety of the considered data set.

Contextual outliers: An outlier is considered "contextual" if its behavior is within a normal (expected) range, but it would not be expected based on some context. Context can be defined as a function of multiple parameters, such as time, location, etc.

Collective outliers: An outlier is considered "collective" if the behavior of each single data point that are part of the anomaly are within expected ranges (so they are not anomalous, it's either a contextual or a global sense), but the group taking all the data points together, is.

« Collective outliers are important because networks are connected. Through different planes interconnected symptoms from various angles can be observed »

Annotate Operational Data

YANG Module

```
module: ietf-symptom-semantic-metadata
  +--rw symptom
    +--rw id          yang:uuid
    +--rw event-id    yang:uuid
    +--rw description                  string
    +--rw start-timeyang:date-and-time
    +--rw end-time    yang:date-and-time
    +--rw confidence-score             float
    +--rw concern-score?               float
    +--rw tags* [key]
    |  +--rw key       string
    |  +--rw value     string
    +--rw (pattern)?
    |  +--:(drop)
    |  |  +--rw dropempty
    |  +--:(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 source
        +--rw (source-type)
        |  +--:(human)
        |  |  +--rw human          empty
        |  +--:(algorithm)
        |     +--rw algorithm      empty
        +--rw name?                string
```

- **Symptoms** describe what changed in the network for what reason and cause with which concern score from when to when.
- **Tags** describes in which network plane, which action, reason and cause was observed.
- **Pattern** describes the measurement pattern over time of the time series data.
- **Source** describes which system **observed** the outlier. A human or a network anomaly detection system.

Experiment: Network Anomaly Lifecycle

What is the Motivation?

Network anomaly detection is about **identifying behaviours** that provide **evidence** of service consumers experiencing a **degradation**.

Network Operators often implement a **continuous review process**, in order to **iteratively collect and incorporate more and more network and service knowledge** into the methodology, to **improve** (reducing False Positives and False Negatives) and validate the detection, e.g. by performing post-mortem analysis.

We see the need to **provide a well-defined lifecycle for the refinement of network anomaly detection**, as this can open up to a **more structured cooperation between different actors** involved in different stages of the lifecycle, including customer service operators, network engineers, Data Scientists, AI algorithms, etc.

This proposed draft describe an **experiment**: verifying whether the approach is usable in real use case scenarios to support proper refinement and adjustments of network anomaly detection algorithms.

Network Anomaly Lifecycle

draft-netana-nmop-network-anomaly-lifecycle

4. Lifecycle of a Network Anomaly

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

```
                        +-------------+
             +--------> |  Detection  | ---------+
 Adjustments |          +-------------+          | Symptoms
             |                                   |
             |                                   v
     +------------+                      +------------+
     | Refinement |<-------------------- | Validation |
     +------------+                      +------------+
                          Incident
                        Confirmation
```

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.

Detection: The Network Anomaly Detection stage is about the continuous monitoring of the network through Network Telemetry [RFC9232] and the identification of symptoms.

Validation: Decides if the detected symptoms are signaling a real incident or if they are to be treated as false positives.

Refinement: Network operator performs detailed postmortem analysis of the network incident, collected Network Telemetry data and detected anomaly with the objective to identify useful adjustments in the Network Telemetry data and Anomaly Detection system.

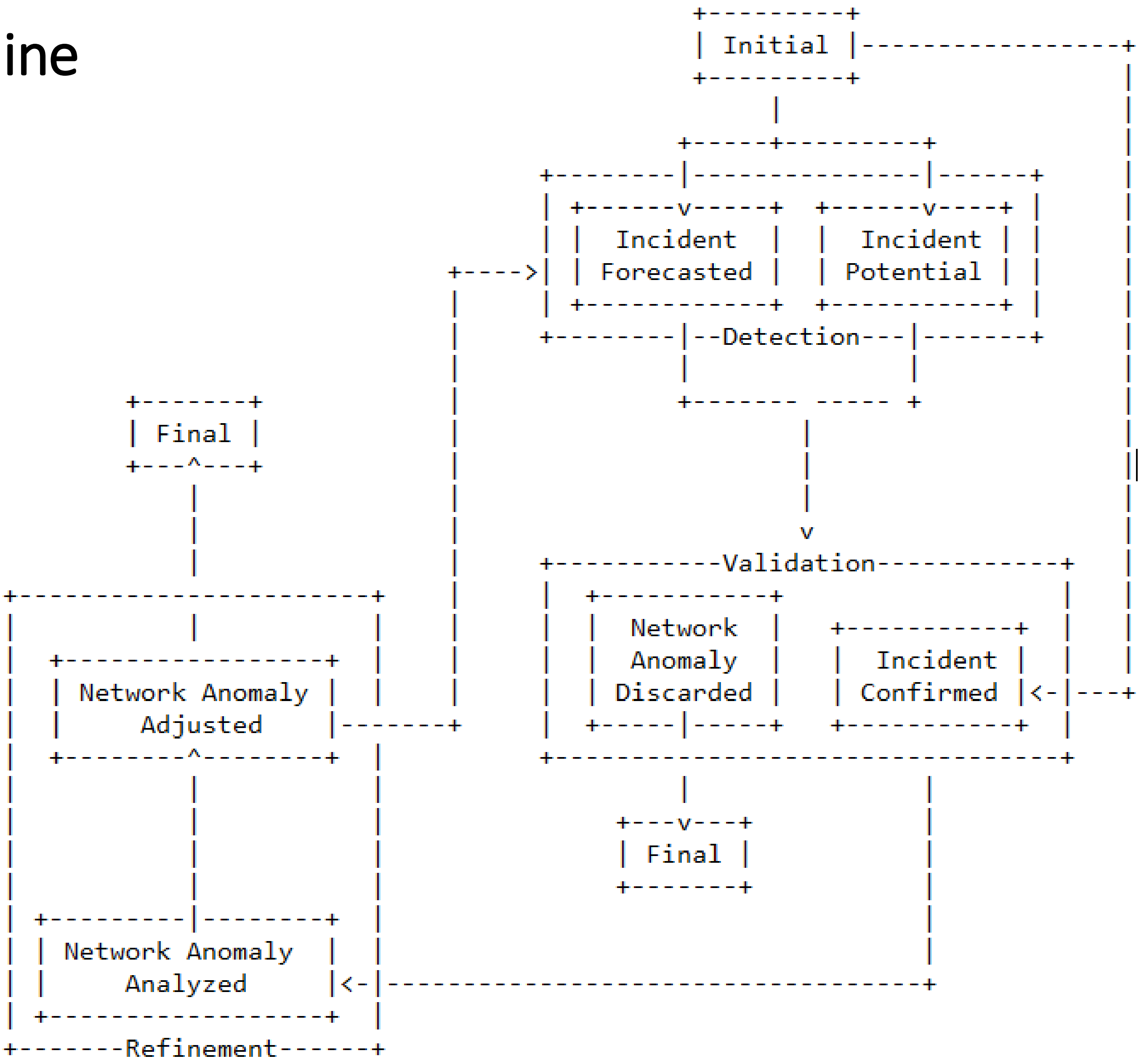
Network Anomaly State Machine

Incident Relationships

Incident Forecasted: A potential network incident is predicted in the future by the Network Anomaly Detection system.

Incident Potential: A potential network incident has been detected by the Network Anomaly Detection system.

Incident Confirmed: A potential network incident has been confirmed in the postmortem validation.



Network Anomaly Metadata

YANG Module

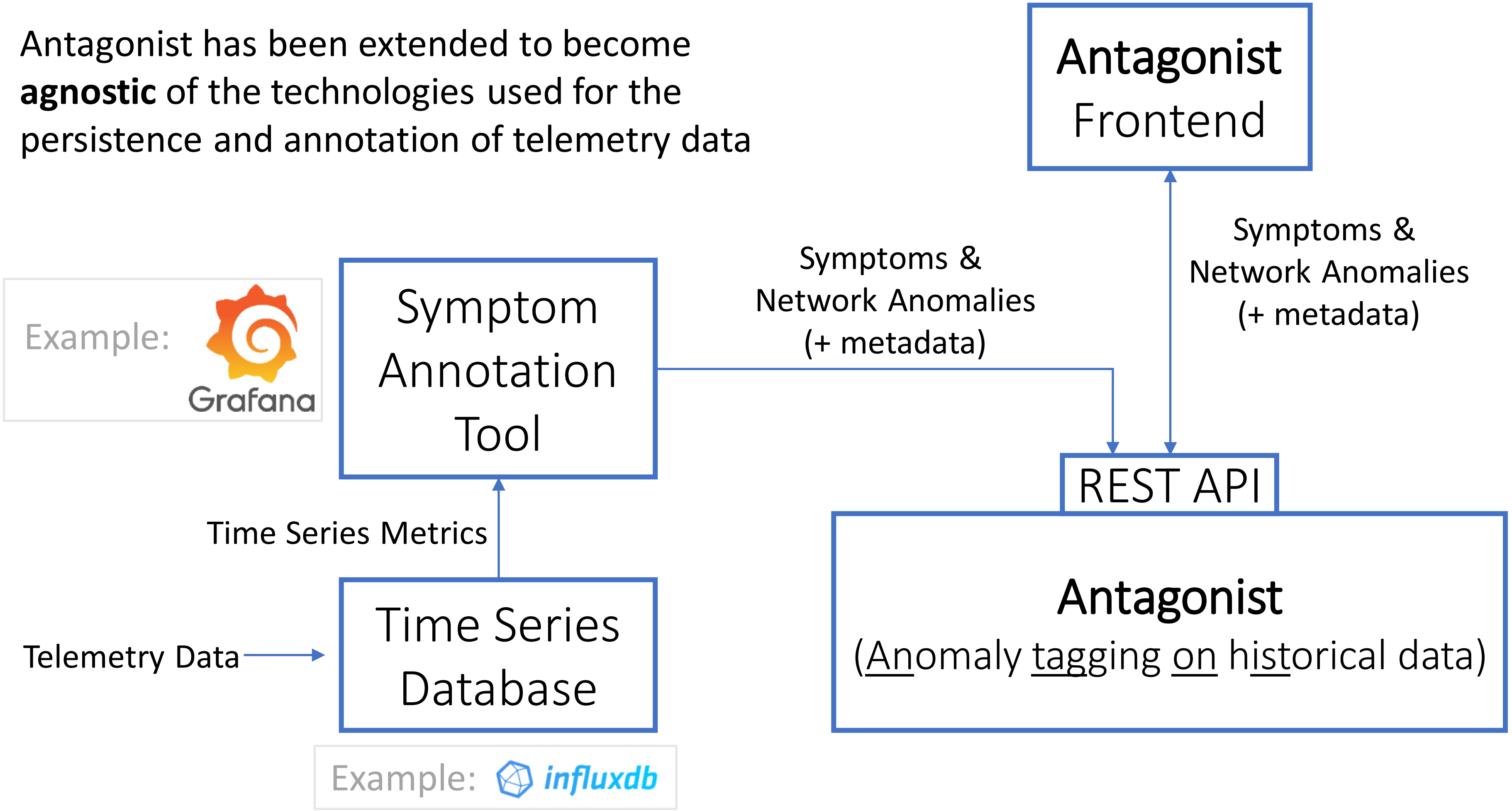
```
module: ietf-network-anomaly-metadata
  +--rw network-anomalies
     +--rw network-anomaly* [id author-name version state]
        +--rw id                yang:uuid
        +--rw description?      string
        +--rw author
        |  +--rw author-name       string
        |  +--rw author-type?      identityref
        |  +--rw algo-version?     uint8
        +--rw version           uint8
        +--rw state             identityref
        +--rw symptoms* [symptom_id]
           +--rw symptom_id      yang:uuid
```

- **ID and Description** uniquely identifies the detected anomaly.
- **Author Name, Type, Version and Algo-Version** describes wherever the anomaly was detected by a human or algorithm and uniquely identifies the system and version who/which detected.
- **State** describes the state of the anomaly (selected among the states defined in the state machine).
- **Symptoms** describes the identified symptoms defined in ietf-symptom-semantic-metadata.

IETF 119 Hackathon - Antagonist

Design and workflow

Antagonist has been extended to become **agnostic** of the technologies used for the persistence and annotation of telemetry data



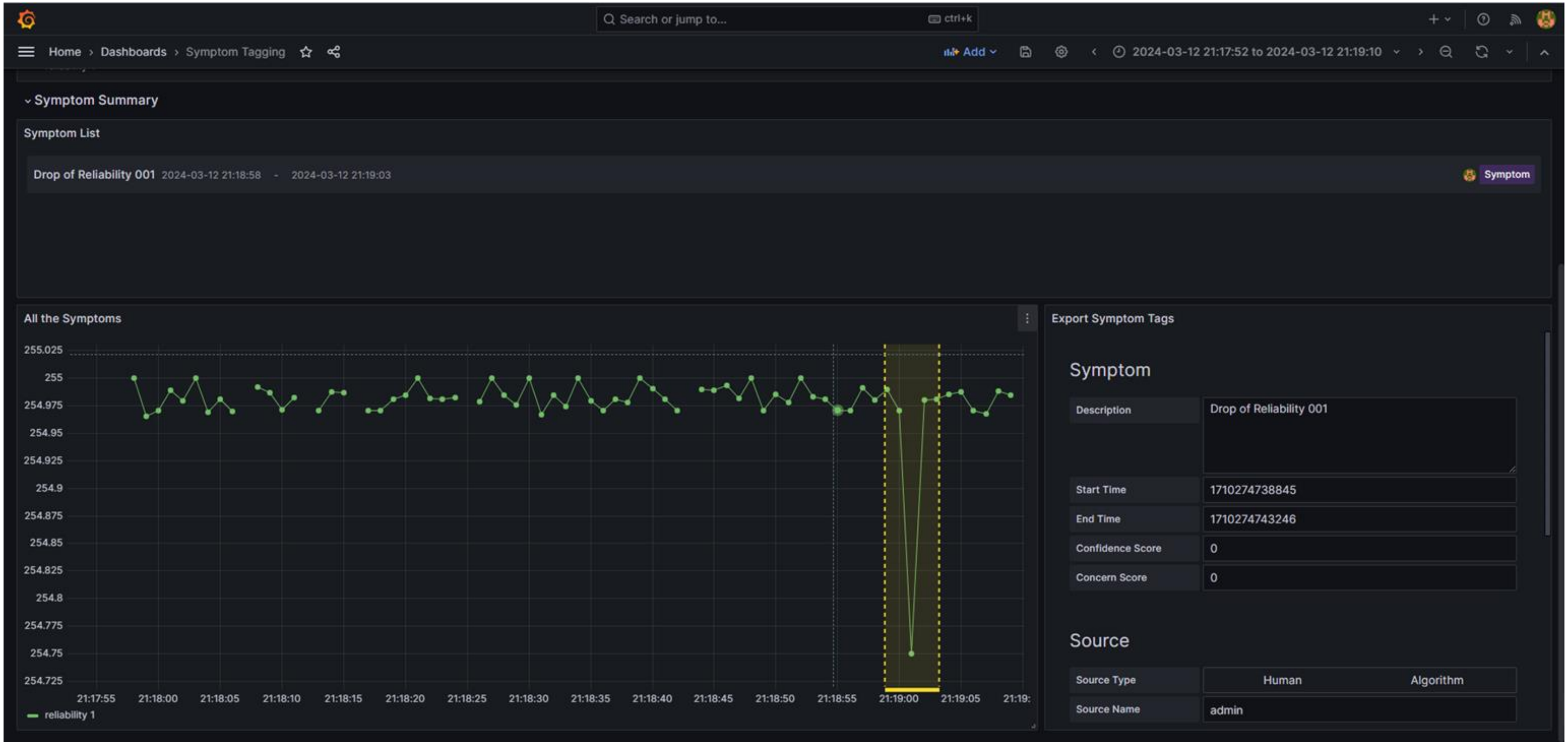
Antagonist exposes a REST API to support **ingestion** and **exposure** of symptoms and network anomaly data and semantic metadata.

The exposed data can be used as ground-truth.

Source Code: https://github.com/vriccobene/antagonist

IETF 119 Hackathon – Antagonist

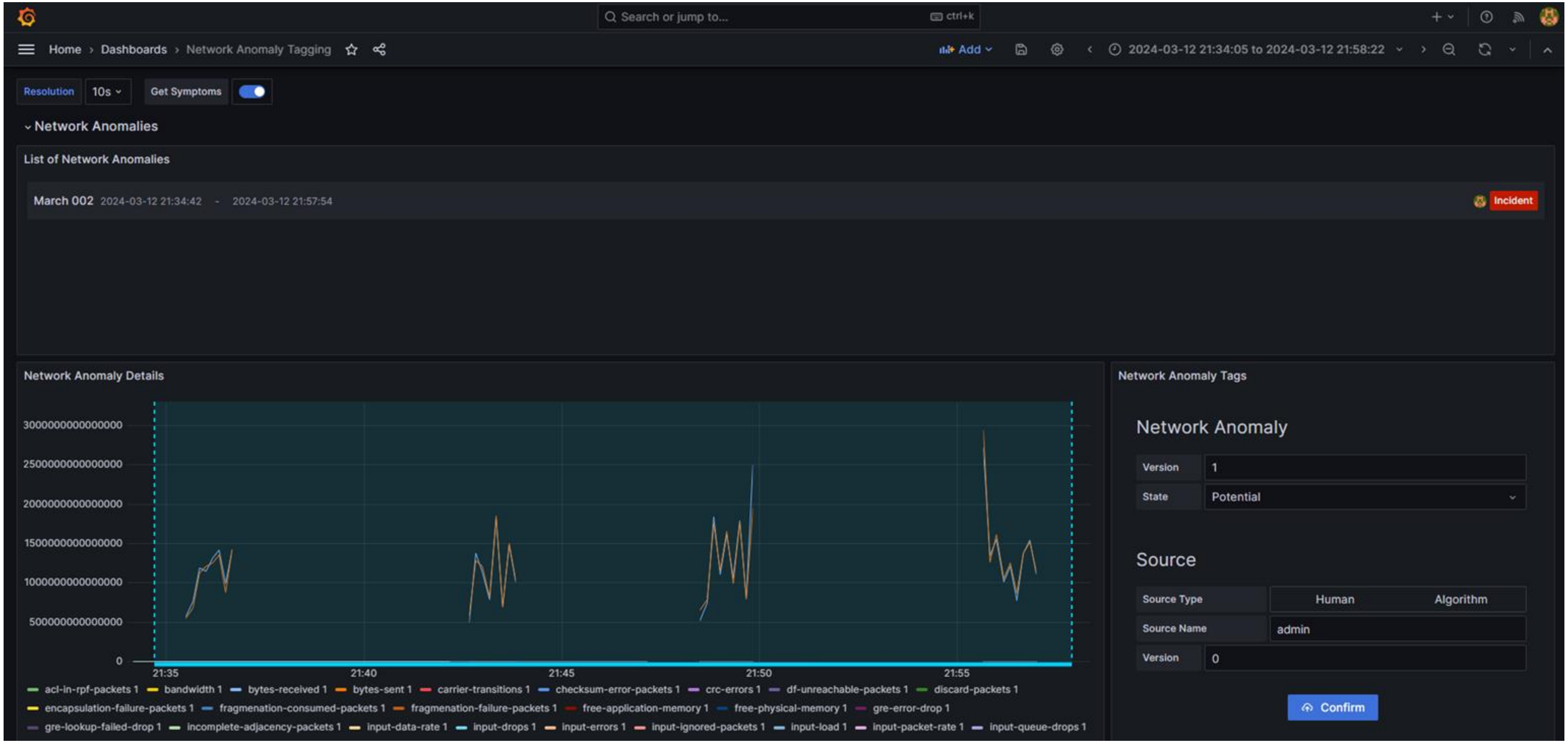
Labelling a Symptom on Time Series



When symptoms are tagged, they get submitted to Antagonist

IETF 119 Hackathon – Antagonist

Labelling a Network Anomalies on Time Series

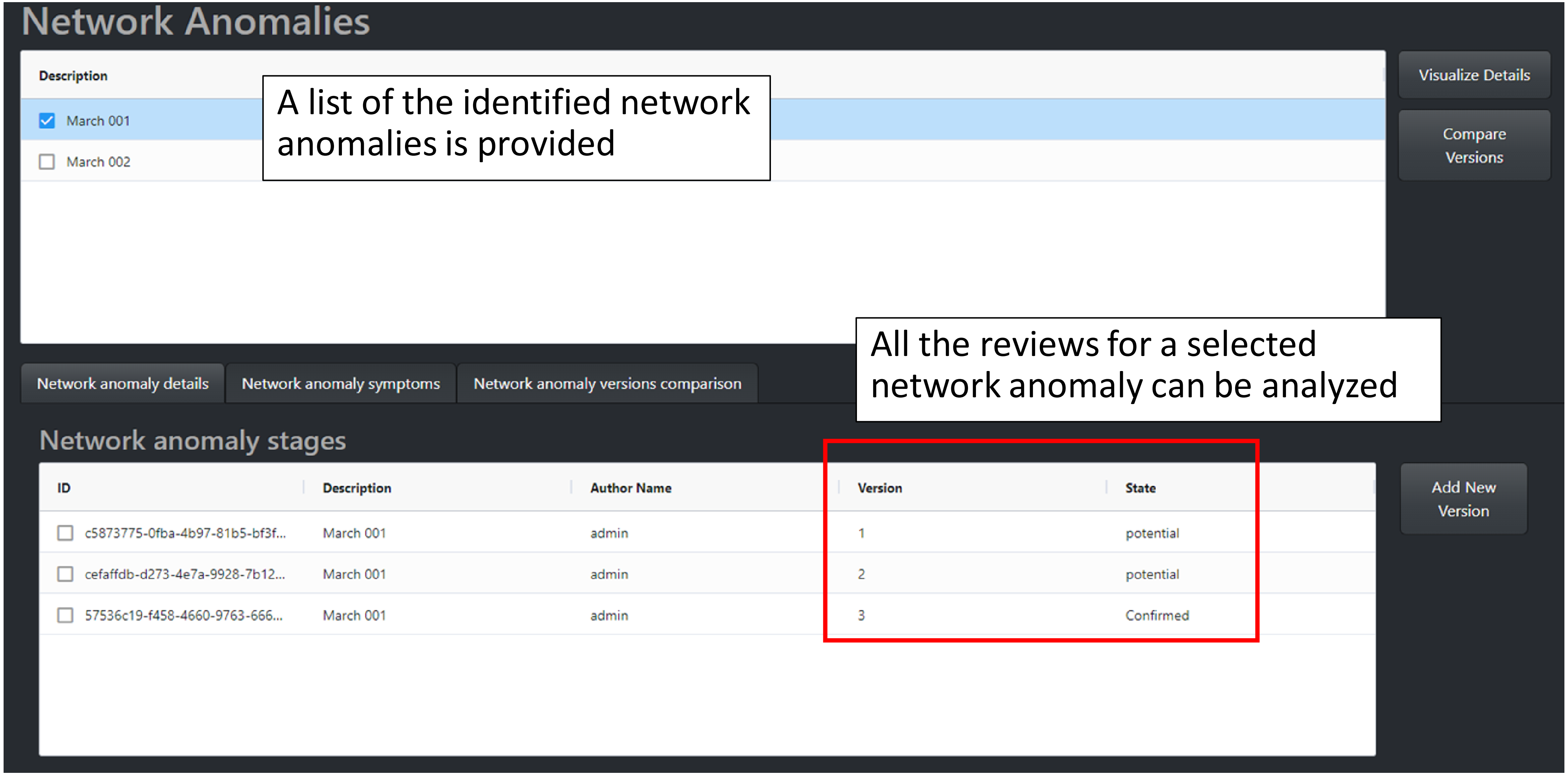


When Network Anomalies are tagged, they get submitted to Antagonist

IETF 119 Hackathon – Antagonist

Labelling a Network Anomalies on Time Series

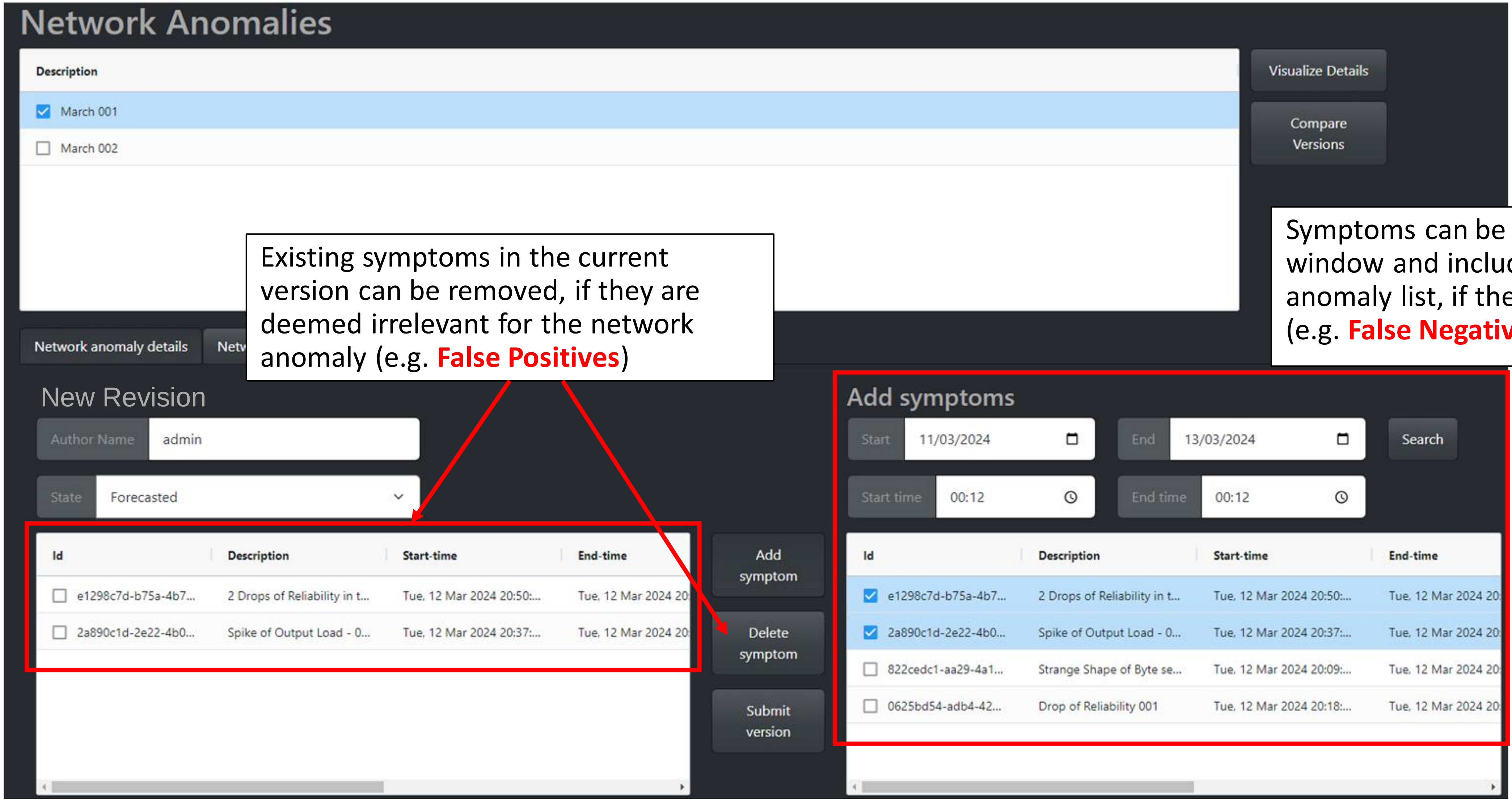
Antagonist can be used to review and analyse network anomalies



IETF 119 Hackathon – Antagonist

Labelling a Network Anomalies on Time Series

Antagonist allows to move the network anomaly forward in its lifecycle, by adding new revisions



Existing symptoms in the current version can be removed, if they are deemed irrelevant for the network anomaly (e.g. **False Positives**)

Symptoms can be retrieved by time window and included in the network anomaly list, if they were missed before (e.g. **False Negatives**)

The information collected by Antagonist can be used by network engineers to review the network anomaly history or can be provided to AI algorithms as additional knowledge for training.