An Architecture for a Network Anomaly Detection Framework

draft-netana-nmop-network-anomaly-architecture-00

Motivation and architecture of a Network Anomaly Detection Framework
and the relationships to other documents describing
network symptom semantics and network incident lifecycle

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

25. July 2024

Data Mesh organizes Data in Organizations

Enables Network Analytics use cases

Alert

Postmortem

RFC 8632

draft-netana-nmop-network-anomaly-semantics

draft-netana-nmop-network-anomaly-lifecycle

Analytical Data

draft-netana-nmop-yang-message-broker-integration

Network Device Trend Detection

Verify, Troubleshoot and Notify

Network Anomaly Detection

Network Visualization

Network SLI and SLO

Closed Loop Operation

draft-netana-nmop-network-anomaly-architecture

Operational Data

draft-netana-nmop-yang-message-broker-integration

Network Data Collection

Network Telemetry (RFC 9232)

IPFIX (RFC 7011, RFC 9487, RFC 9160, draft-ietf-opsawg-ipfix-on-path-telemetry)

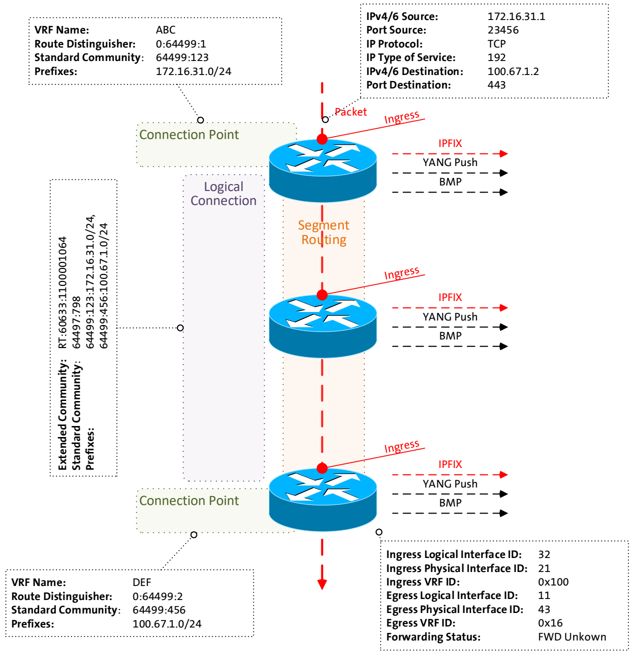
BMP (RFC 7854, RFC 8671, RFC 9069, draft-ietf-grow-bmp-tlv, draft-ietf-grow-bmp-path-marking-tlv, draft-lucente-grow-bmp-rel)

YANG-Push (RFC 8639, RFC 8641, draft-ietf-netconf-udp-notif, draft-ietf-netconf-distributed-notif, draft-ahuang-netconf-notif-yang, draft-ietf-netconf-yang-notifications-versioning, draft-tgraf-netconf-notif-sequencing, draft-tgraf-netconf-yang-push-observation-time)

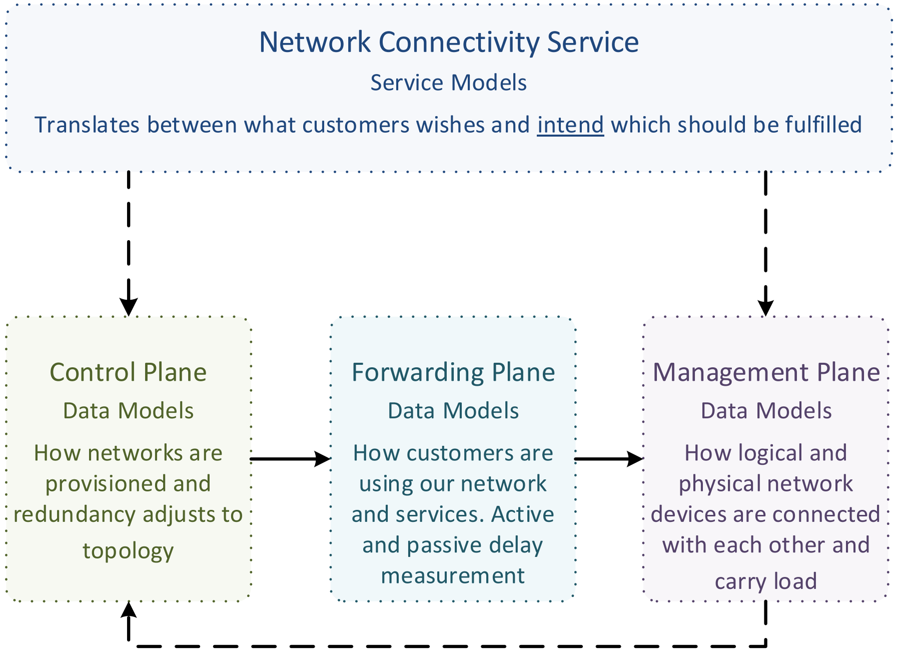
What to monitor

Which metrics are collected

« Network operators connect customers in routing tables called Connectivity Services »

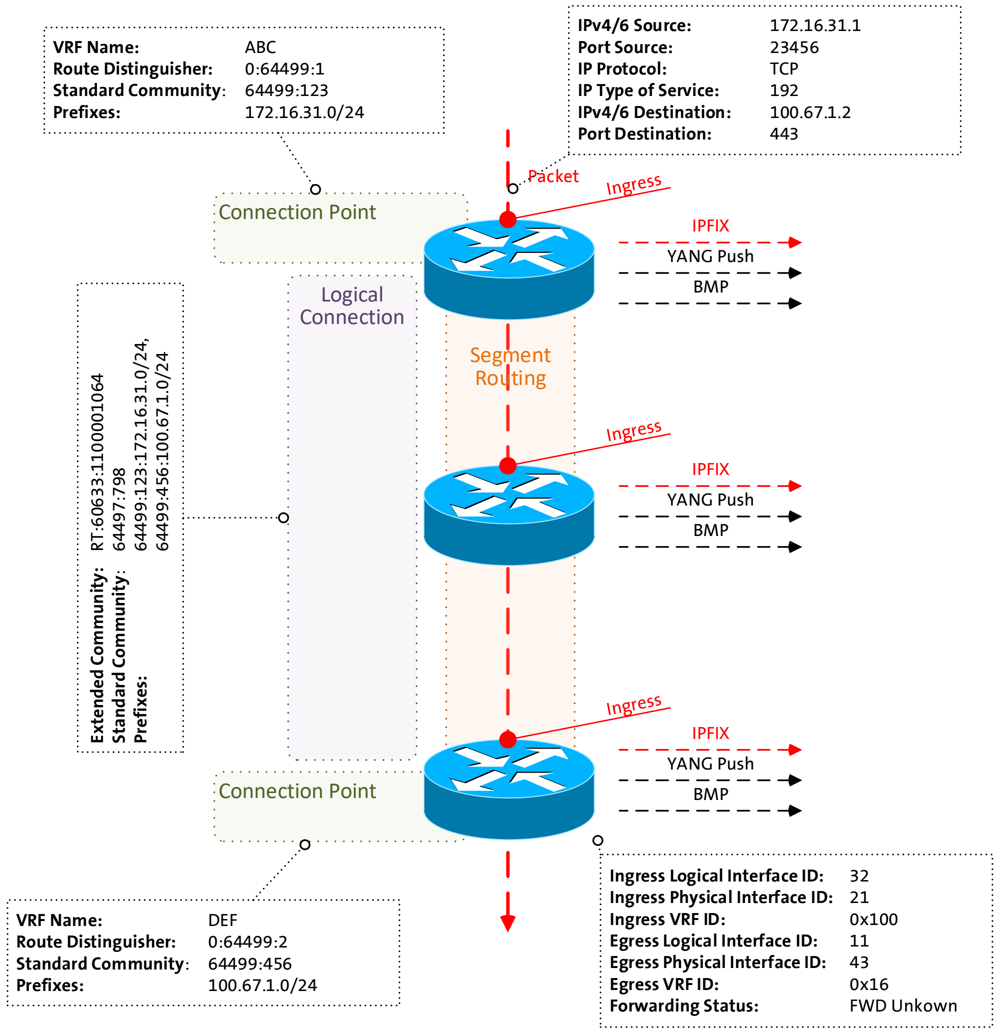


« Network Telemetry (RFC 9232) describes how to collect data from all 3 network planes efficiently »



Example: Monitoring L3 VPN's with IPFIX, BMP and YANG Push

From Connectivity Service to Realtime Network Analytics



- **Connectivity Service perspective,** Connection Points are connected through Logical Connections.
- **From a BGP control-plane perspective,** IPv4/6 unicast prefixes in VRF's are tagged with BGP standard communities.
  - One BGP standard community to identify the Logical Connection. One BGP standard community to identify each Connection Point.
  - When IPv4/6 prefixes are exported from VRF's, a BGP route-distinguisher, BGP extended community route-targets and a SRv6 VPN SID for the IPv6 next-hop are allocated.
- **From a forwarding plane perspective,** when IPv4/6 unicast traffic is received from the edge at the SRv6 PE, a lookup is performed, the SRv6 VPN SID is obtained and IPv6 next-hop is added when forwarded to the core.
- **Swisscom collects** MPLS and SRv6 provider data plane, IPv4/6 unicast customer data-plane in IPFIX and at provider edge BGP VPNv4/6 unicast **in production** to perform real-time data correlation.

Segment Routing IPv6 Encapsulation

RFC 9487 provides IPFIX Visibility

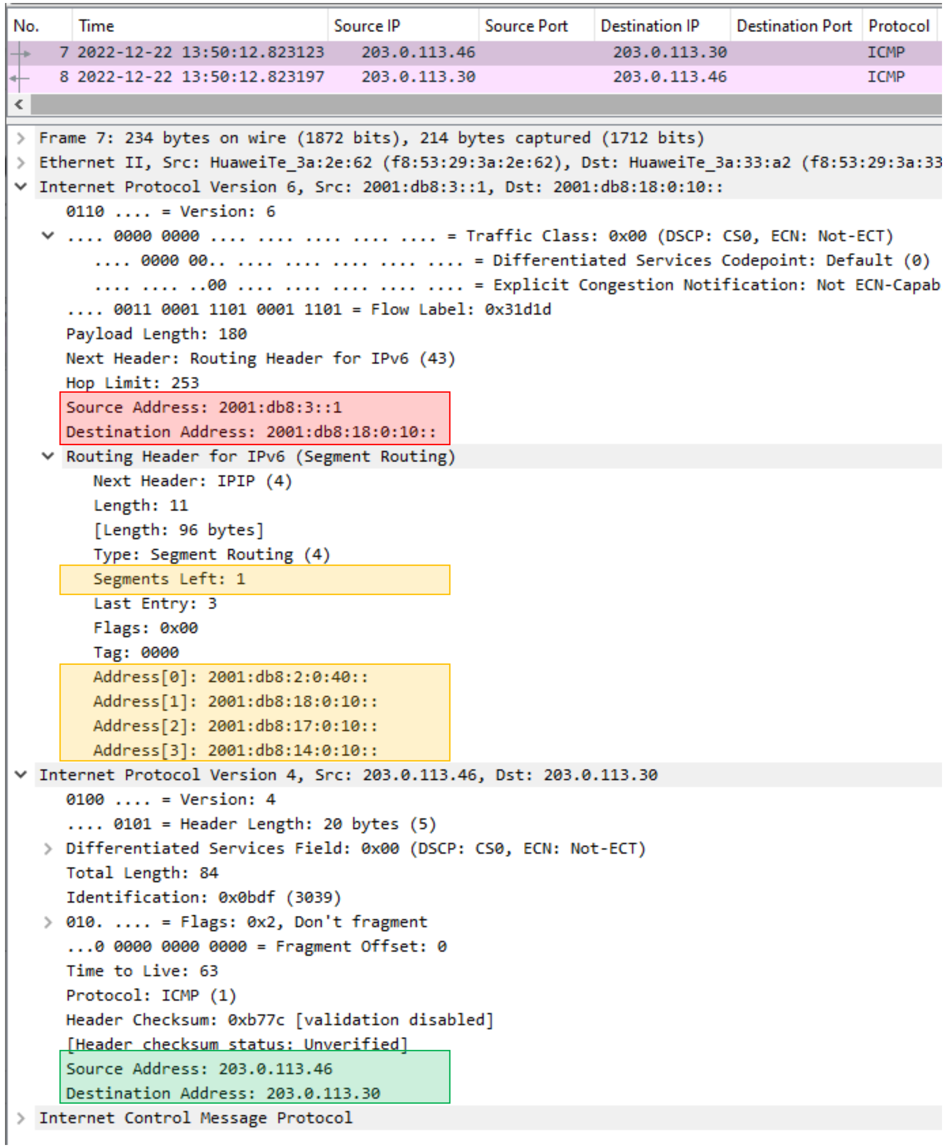
› Provider data-plane

Divided into an IPv6 and Segment Routing Header.

The IPv6 header shows from which PE to which next-hop it is being forwarded. The Segment Routing Header the list of segments this packet needs to pass through and points to the active segment.

› Customer data-plane

This is what we receive from the customer and encapsulate for transport through the SRv6 core.



BMP – Address Family Agnostic

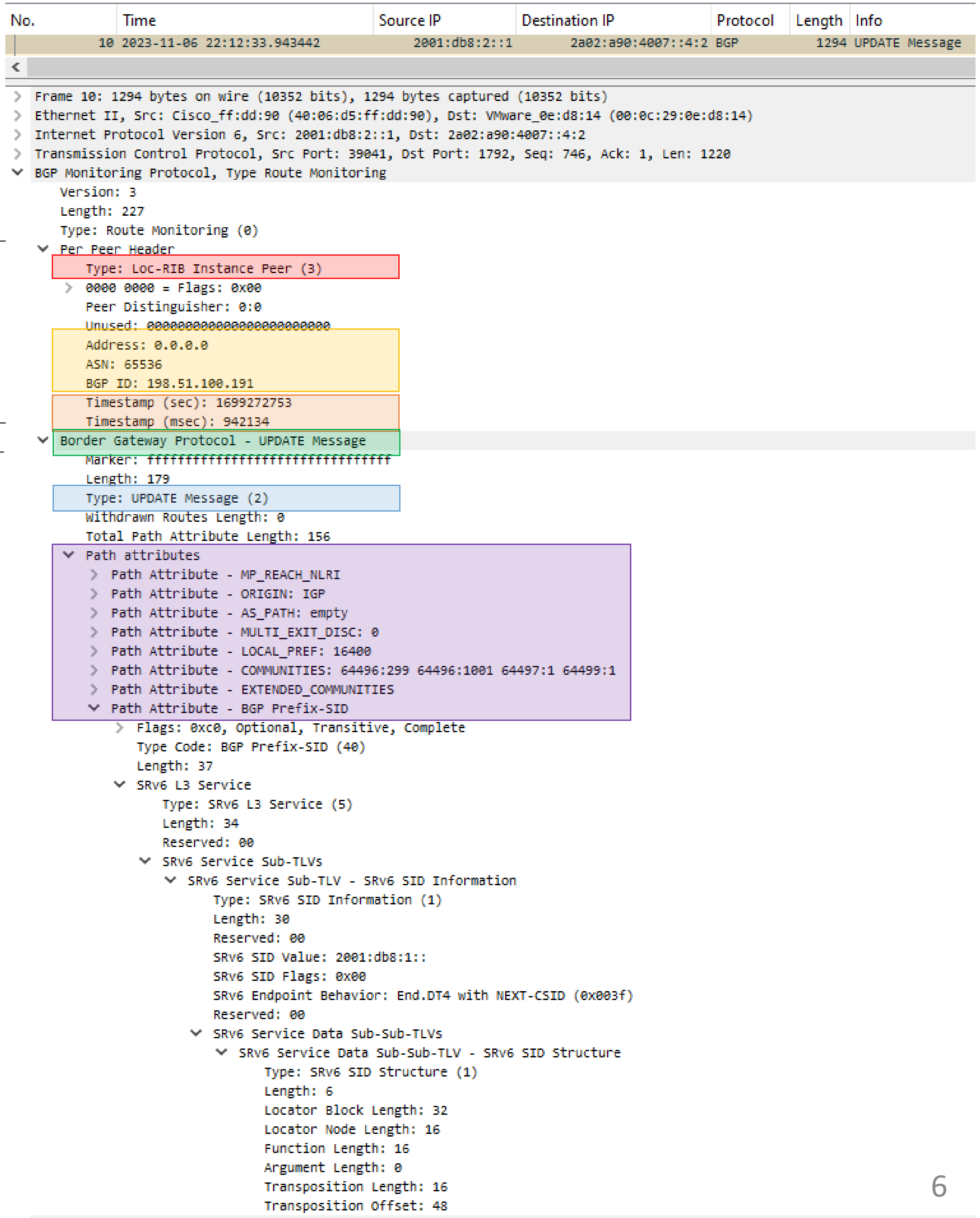
RFC 9069 provides Local RIB Visibility

› BMP Per Peer Header

Shows at which RIB (Adj-RIB In, Local or Adj-RIB Out, Pre or Post Policy) and from which Peering the BGP PDU at which time was obtained.

› Encapsulated BGP PDU

Shows the encapsulated BGP PDU. In case of BMP route-monitoring, it describes whether it was a topology update or withdrawal and for BGP community, NLRI and BGP Prefix SID path attributes.

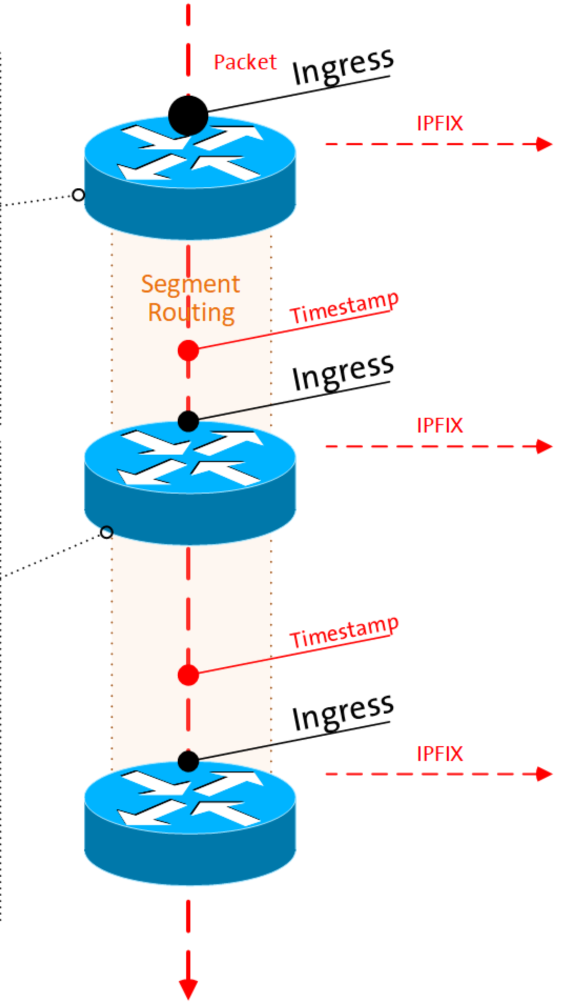


Measure On-Path Delay with Network Context

With draft-ietf-opsawg-ipfix-on-path-telemetry

IPv4/6 Source:	172.16.31.1
Port Source:	23456
IP Protocol:	TCP
IP Type of Service:	192
IPv4/6 Destination:	100.67.1.2
Port Destination:	443
Ingress Logical Interface ID:	32
Ingress Physical Interface ID:	21
Ingress VRF ID:	0x100
Egress Logical Interface ID:	11
Egress Physical Interface ID:	43
Egress VRF ID:	0x16
Forwarding Status:	FWD Unkown

IPv4/6 Source:	172.16.31.1
Port Source:	23456
IP Protocol:	TCP
IP Type of Service:	192
IPv4/6 Destination:	100.67.1.2
Port Destination:	443
Ingress Logical Interface ID:	32
Ingress Physical Interface ID:	21
Ingress VRF ID:	0x16
Egress Logical Interface ID:	11
Egress Physical Interface ID:	43
Egress VRF ID:	0x16
Forwarding Status:	FWD Unkown
SID List:	17001, 34002
Delay Min	1
Delay Sum	5
Delay Max	7



- Packets are **captured** ingress with an **optional sampler**, data plane dimensions **extracted**, **enriched** with management and control plane dimensions and added with a unique **flow ID** to a flow cache on the node for aggregation.
- **A direct export marking bit and optionally a timestamp is added** to the packet when entering the OAM domain by leveraging Enhanced Alternate Marking (RFC 9341, draft-zhou-ippm-enhanced-alternate-marking) or IOAM (RFC 9378) Direct Export or E2E Option Type.
- Each subsequent packet for the same flow increases byte and packet count. Each new flow creates a new flow ID in the flow cache.
- **At each node** in transit or only at the decapsulation node, **delay is calculated** by comparing the observation timestamp in the packet and when packet is received. **Delay is populated into the flow cache together with packet and byte count** as defined in draft-ietf-opsawg-ipfix-on-path-telemetry.

IPFIX provides statistical data plane visibility

Comprehensive IPv6 data plane coverage

Provider IPv6 Data Plane

- IPv6 Source and Destination Address
- IPv6 Hop-by-Hop Options Header IOAM DEX or Alternate Marking
- IPv6 Destination Options Header IOAM E2E or Alternate Marking
- SRv6 Segment Routing Header

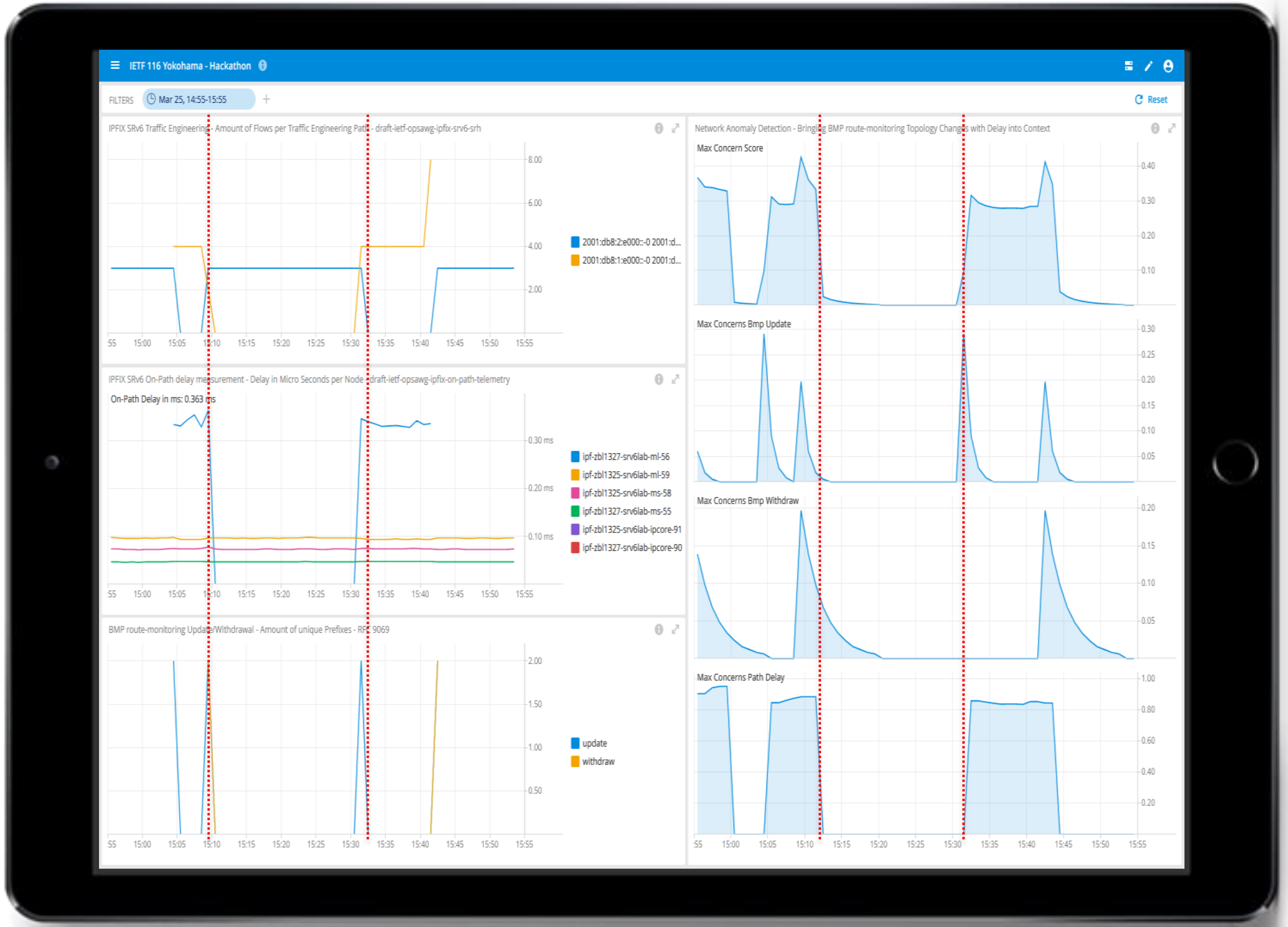
Customer IPv6/4 Data Plane

- IPv4/6 Source and Destination Address, Transport Potocol and Port

Hop-by-Hop processed

Endpoint processed

- **draft-ietf-opsawg-ipfix-on-path-telemetry** defines OAM agnostic on-path delay IPFIX entities.
- **draft-ietf-opsawg-ipfix-alt-mark** defines Alternate Marking IPFIX entities.
- **draft-spiegel-ippm-ioam-rawexport** defines IOAM IPFIX entities.
- **RFC 9487** defines Segment Routing Header IPFIX entities



Network Observability

System Demo

Detect L3 VPN Topology changes and its Forwarding Plane impact in Near Real-Time

What does Network Anomaly Detection mean

Monitor changes, called outliers, in networks

Network Anomaly Detection

For Connectivity Services, Network Anomaly Detection **constantly monitors and detects any network or device topology change**, 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 in 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.

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 »

Elements of the Architecture

```
        +---------+                     +-------------------+
        |Service  |                     |     Alert and     |
   |--- |Inventory|                     | Problem Management|
   |    |         |                     |      System       |
   |    +---------+                     +-------------------+
   |      |                                      ^     Stream
   |      |                                      |
   |      |       +---------+           +-------------------+
   |      |       | Post-   | Stream    |   Message Broker  |
   |      |       | mortem  | <-------- |  with Analytical  |
   |      |       | System  |           |    Network Data   |
   |      |       +---------+           +-------------------+
   |      |            |                         ^     Stream
   |      |            |                         |
   |      |            |                +-------------------+
   |      | Profile    | Fine           | Alert Aggregation | Store Label
   |      | and        | Tune           | for Anomaly       | ------------|
   |      | Generate   | SDD            | Detection         |             |
   |      | SDD Config | Config         +-------------------+             |
   |      |            |                      ^  ^  ^ Stream             |
   |      v            v                      |  |  |                    ?
   |   +-------------------+            +-------------------+        +---------+
   |   | Service Disruption| Schedule   | Service Disruption| Replay |  Data   |
   |   |     Detection     | ---------> |     Detection     |<------ | Storage |
   |   |   Configuration   | Detection  |                   |        |         |
   |   +-------------------+            +-------------------+        +---------+
   |                                       ^ ^ Stream ^ ^ ^                ^
   |                                       | |        | | |                |
   |                                    +---------+---------+              |
   |                                    | Network |  Data   | Store        |
   |----------------------------------> |  Model  |  Aggr.  | ------------|
                                        |         | Process | Operational Data
                                        +---------+---------+
                                               ^  ^  ^ Stream
                                               |  |  |
                                        +-------------------+
                                        |   Message Broker  |
                                        |  with Operational |
                                        |    Network Data   |
                                        +-------------------+
                                               ^  ^  ^ Stream
Subscribe                       Publish        |  |  |
          +-------------------+         +-------------------+
          | Network Node with | ------> | Network Telemetry |
--------> | Network Telemetry | ------> |  Data Collection  |
          |    Subscription   | ------> |                   |
          +-------------------+         +-------------------+
```

- **Service Inventory** contains list of the connectivity services.
- **Service Disruption Detection** processes aggregated network data to decide whether a service is degraded or not.
- **Service Disruption Detection Configuration** defines the set of approaches that need to be applied to perform SDD.
- **Operational Data Collection** manages network telemetry subscriptions and transforms data into message broker.
- **Operational Data Aggregation** produces data upon which detection of a service disruption can be performed.
- **Network Modeling** establishes knowledge of network relationships.
- **Data Profiling** categorizes nondeterministic customer related data.
- **Detection Strategies** for a profile a detection strategy is defined.
- **Machine Learning** is commonly used to detect outliers or anomalies.
- **Storage** some algorithms may relay on historical (aggregated) operational data to detect anomalies.
- **Alerting** consolidates analytical insights and notifies.
- **Postmortem** refines and stores the network anomaly and symptom labels into the Label Store.
- **Replaying** to validate refined anomaly and symptom labels, historical operational data is replayed.

Semantic Metadata Annotation for Network Anomaly Detection

draft-netana-nmop-network-anomaly-semantics

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

- **Symptom ID and description** uniquely identifies the detected anomaly. **Event ID, start/end-time and confidence/concern-score** uniquely identifies the network event with its start and end time, how confident the system identified the anomaly and how concerned an operator should be.
- **Tags** allows to add customer information.
- **Pattern** describes the identified pattern of the anomaly.
- **Annotator Name, Type,** describes wherever the anomaly was detected by a human or algorithm and uniquely identifies the system who/which detected.

Experiment: Network Anomaly Lifecycle

draft-netana-nmop-network-anomaly-lifecycle

« Network Anomaly Detection is an iterative process that requires continuous improvement »

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 collection and Anomaly Detection system.

Experiment: Network Anomaly Lifecycle

draft-netana-nmop-network-anomaly-lifecycle

```
module: ietf-network-anomaly-metadata
  +--rw network-anomalies
    +--rw network-anomaly* [id version]
      +--rw id                yang:uuid
      +--rw version           uint32
      +--rw description?      string
      +--rw state             identityref
      +--rw annotator
      |  +--rw (annotator-type)
      |  |  +--:(human)
      |  |  |  +--rw human          empty
      |  |  +--:(algorithm)
      |  |     +--rw algorithm      empty
      |  +--rw name?                empty
      +--rw symptoms* [symptom_id]
        +--rw symptom_id      yang:uuid
```

- **ID and Description** uniquely identifies the detected network anomaly (as a container of symptoms).
- **Description and State** provide general information regarding the anomaly and .
- **Annotator** describes the entity that observed the network anomaly: this can be a human or an algorithm (anomaly detection system).
- **Symptoms** provides a list of symptoms (based on ietf-symptom-metadata) that are part of this network anomaly.

Experiment: Network Anomaly Lifecycle

draft-netana-nmop-network-anomaly-lifecycle

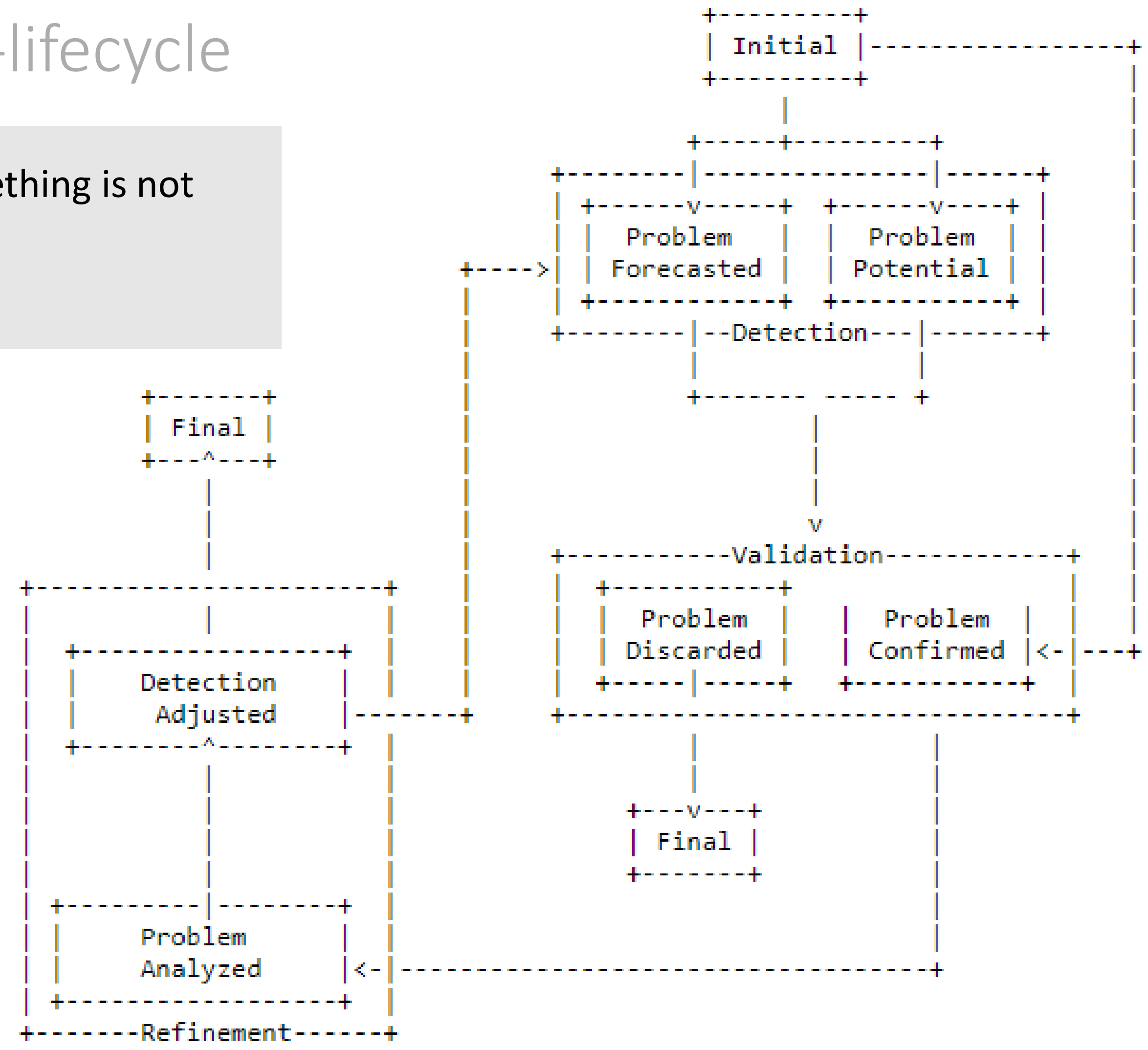
Network Anomaly Detection is the art of understanding when something is not working as expected in the network.
It is an **iterative process** that requires **continuous improvement**

The detection in the different stages can either come from humans (network operation engineers) or algorithms (e.g. rule-based, AI-based).

→ **Having labels structured well is key**

It is crucial to make sure we can **audit the process** e2e and **involve network engineers** at any stage of the process to validate and provide feedback

→ **Interoperability between teams and between “annotators” is key**



Identified three main stages of the life cycle

“annotator” = entity that produces the label - either human or algorithm

Experiment:
Network Anomaly Lifecycle

Provide labels to ML based on the specific action to perform
(e.g. Active Learning, ML Retraining,)

```
module: ietf-symptom-metadata
       +--ro ifws:symptom
           +--ro ifws:id?                          yang:uuid
           +--ro ifws:event-id?                    yang:uuid
           +--ro ifws:description?                 string
           +--ro ifws:start-time?                  yang:date-and-time
           +--ro ifws:end-time?                    yang:date-and-time
           +--ro ifws:confidence-score?            score
           +--ro ifws:concern-score?               Score
           +--ro (ifws:pattern)?
           |  +--:(ifws:drop)
           |  |  +--ro ifws:drop                   empty
           |  +--:(ifws:spike)
           |  |  +--ro ifws:spike                  empty
           |  +--:(ifws:mean-shift)
           |  |  +--ro ifws:mean-shift             empty
           |  +--:(ifws:seasonality-shift)
           |  |  +--ro ifws:seasonality-shift      empty
           |  +--:(ifws:trend)
           |  |  +--ro ifws:trend                  empty
           |  +--:(ifws:other)
           |     +--ro ifws:other                  string
           +--ro ifws:tags* [key]
           |  +--ro ifws:key       string
           |  +--ro ifws:value     string
           +--ro ifws:annotator
              +--ro  (ifws:annotator-type)
              |  +--:(ifws:human)
              |  |  +--ro ifws:human         empty
              |  +--:(ifws:algorithm)
              |     +--ro ifws:algorithm     empty
              +--ro ifws:name?               string
```

In which state of the anomaly the label was generated

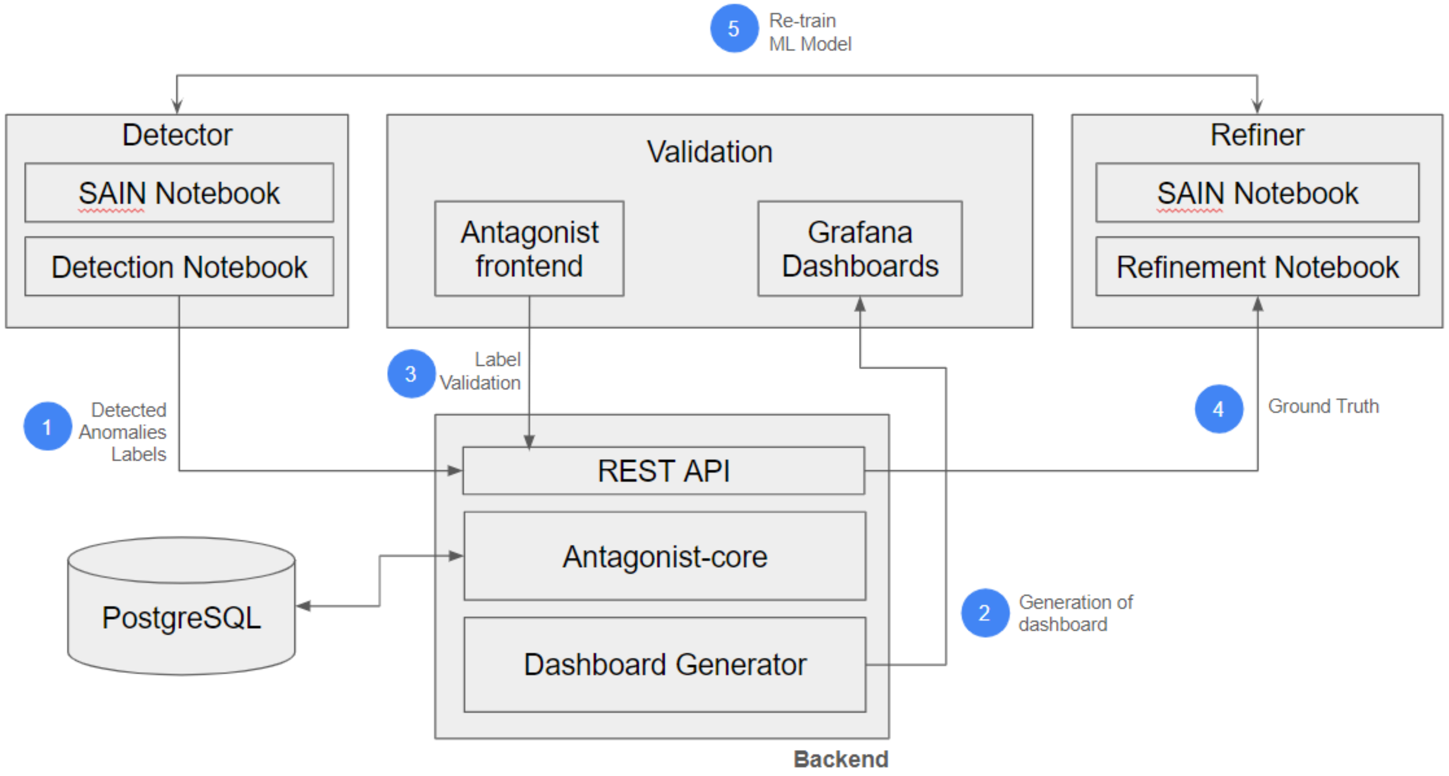
```
module: ietf-network-anomaly-metadata
  +--rw network-anomalies
    +--rw network-anomaly* [id version]
      +--rw id                 yang:uuid
      +--rw version            uint32
      +--rw description?       string
      +--rw state              identityref
      +--rw annotator
      |  +--rw (annotator-type)
      |  |  +--:(human)
      |  |  |  +--rw human         empty
      |  |  +--:(algorithm)
      |  |     +--rw algorithm     empty
      |  +--rw name?               empty
      +--rw symptoms* [symptom_id]
        +--rw symptom_id      yang:uuid
```

Labels can be filtered based on the tags

Which entity generated the label

Experiment: Antagonist

anomaly tagging on historical data



Next Steps:
- Improve scalability
- Integrate and Validate with Swisscom Data

Goals:
- Build a **Label Store** for Network Anomaly detection
- Prove that YANG models contain all the necessary information
- Validate models across a wide range of use-cases

Done so far:
- ✓ Validation with real operational data (Cloud monitoring)
- ✓ Validation with rule-based Network Anomaly Detector (SAIN RFC9417/RFC9418)
- ✓ Validation with a ML-based Network Anomaly Detector (Autoencoder)
- ✓ Add support for Re-training of ML-based models
- ✓ Add partial support for Metadata Filtering and search
- ✓ YANG model refinements to reflect the results of the coding
- ✓ Automatic dashboard generation

Experiment: Antagonist
Supported Use Cases

Antagonist currently supports some interesting use cases, through which we validated the tool:

- **UC1 - Detection** - Enable detectors to persist network anomaly labels
- **UC2 - Validation -** Enable network Engineers to validate labels, review and compare detection results
- **UC3 - Refinement –** Enable the refinement of detectors
  - **UC3.1** - Improving detection rules (e.g. using SAIN – RFC 9417 / 9418)
  - **UC3.2** - Retraining ML models
- **UC4 - Active Learning** – Allow the network experts to validate only the necessary Labels

UC1 Detection
(1 / 2)

```
import sys
sys.path.append('..')
from demo_anomaly_detector import autoencoder_detector

# If a model has been pre-trained, it will be loaded automatically
anomaly_detector = autoencoder_detector.DemoAnomalyDetector()
```

Instantiate an anomaly detector

```
network_anomalies = anomaly_detector.detect(telemetry_df)
```

Run detection on telemetry data

```
## Send the data to Antagonist
for network_anomaly in network_anomalies:

    # Create network anomaly label
    net_anomaly = {
        "annotator": {
            "name": anomaly_detector.get_model_name(),
            "annotator_type": "algorithm"
        },
        "description": f'Detected Network Anomaly on {machine_id} - {datetime.datetime.fromtimestamp(network_anomaly[0]).strftime("%Y-%m-%d at %H")}',
        "state": "incident-potential",
        "version": 1
    }
    response = requests.post(
        f"http://{ANTAGONIST_HOST}/api/rest/v1/network_anomaly", json=net_anomaly
    )
    response.raise_for_status()
    ni_uuid = response.json()
```

Store detected network anomalies on Antagonist

Store detected symptoms on Antagonist

UC1 Detection
(2 / 2)

```python
# Create network symptoms labels and link with the network incident
for symptom in network_anomaly[2]:
    tags = {
        "machine": machine_id,
        "metric": db.get_metric_names()[symptoám[0]],
        "group": group,
    }

    net_sym = {
        'start-time': datetime.datetime.fromtimestamp(symptom[1]).strftime("%Y-%m-%dT%H:%M:%S"),
        'end-time': datetime.datetime.fromtimestamp(symptom[2]).strftime("%Y-%m-%dT%H:%M:%S"),
        "event-id": ni_uuid,
        "concern-score": symptom[3],
        "confidence-score": symptom[4],
        "description": "Symptom",
        "pattern": "",
        "tags": tags,
        "annotator": {
            "name": f"{anomaly_detector.get_model_name()}",
            "annotator_type": "algorithm"
        }
    }
    # Persist the Symptom
    response = requests.post(
        f"http://{ANTAGONIST_HOST}/api/rest/v1/symptom", json=net_sym
    )
    response.raise_for_status()
    symptom_uuid = response.json()
    # Link the Symptom to the network anomaly
    sym_to_net = {"symptom-id": symptom_uuid, "incident-id": ni_uuid}
    response = requests.post(
        f"http://{ANTAGONIST_HOST}/api/rest/v1/network_anomaly/symptom", json=sym_to_net
    )
    response.raise_for_status()
```

Set the tags for the Symptoms

Set the tags for the Symptoms

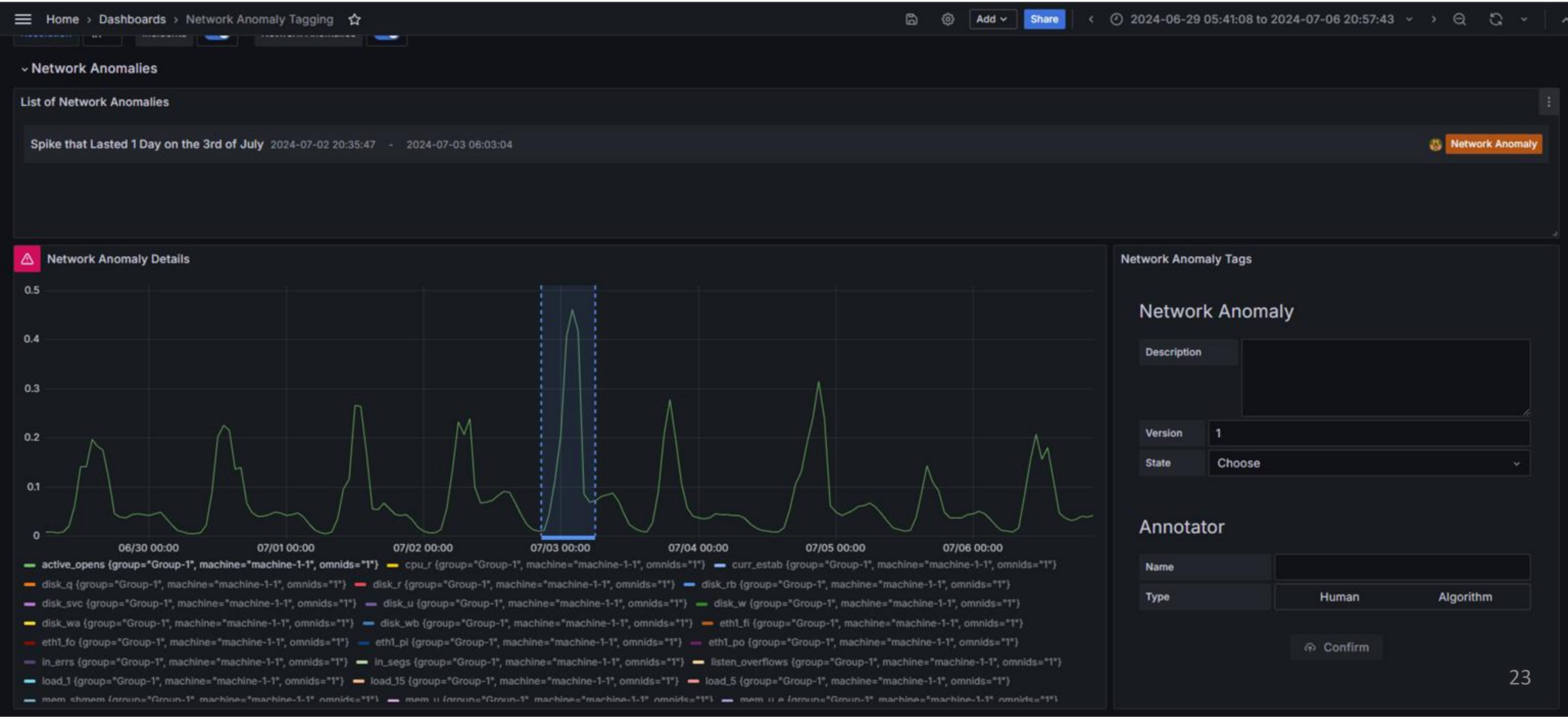
Post the symptom to Antagonist

Connect the network anomaly with the Symptom

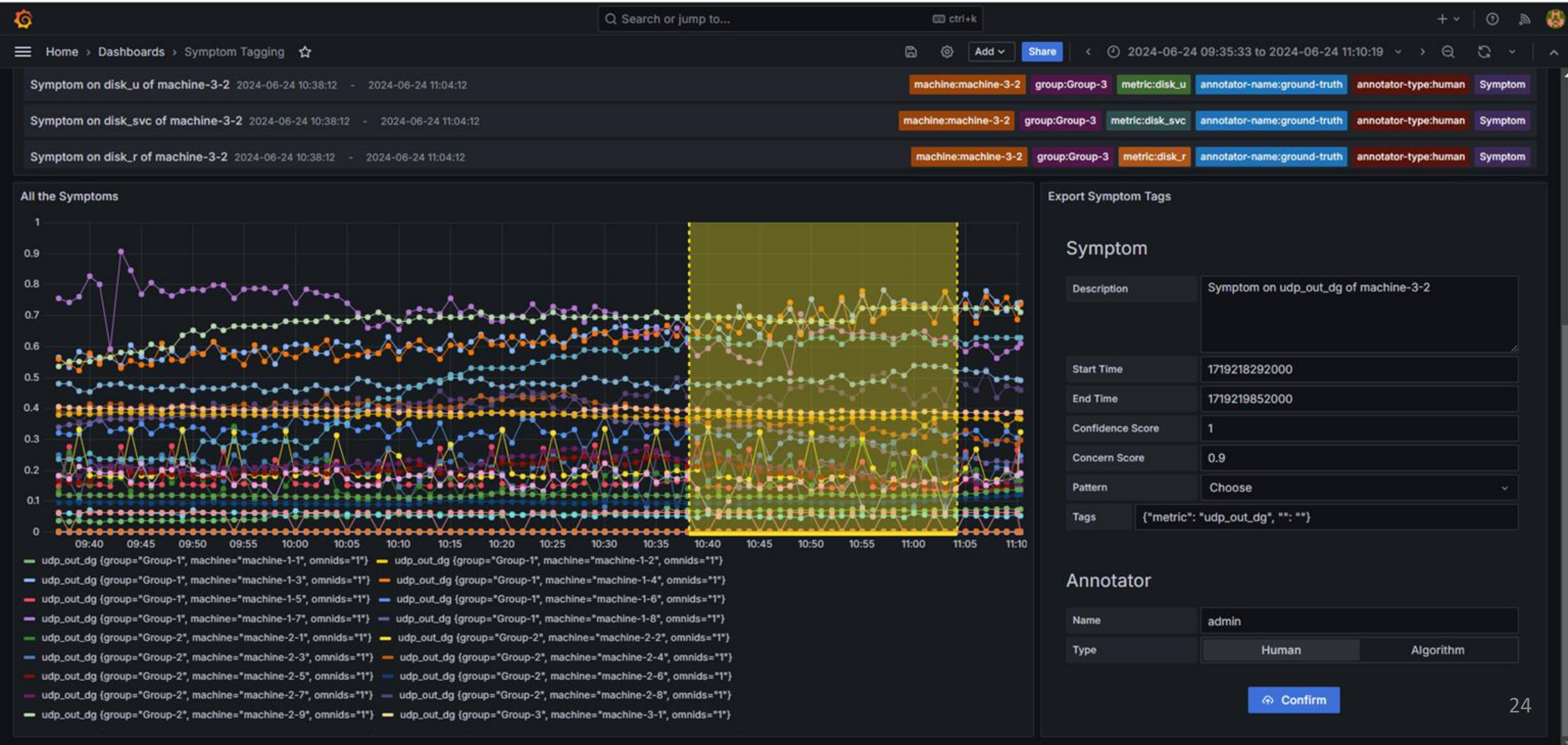
UC2: Validation of Labels on the dashboards

The YANG model contains all the information that we need to automatically generate dashboards

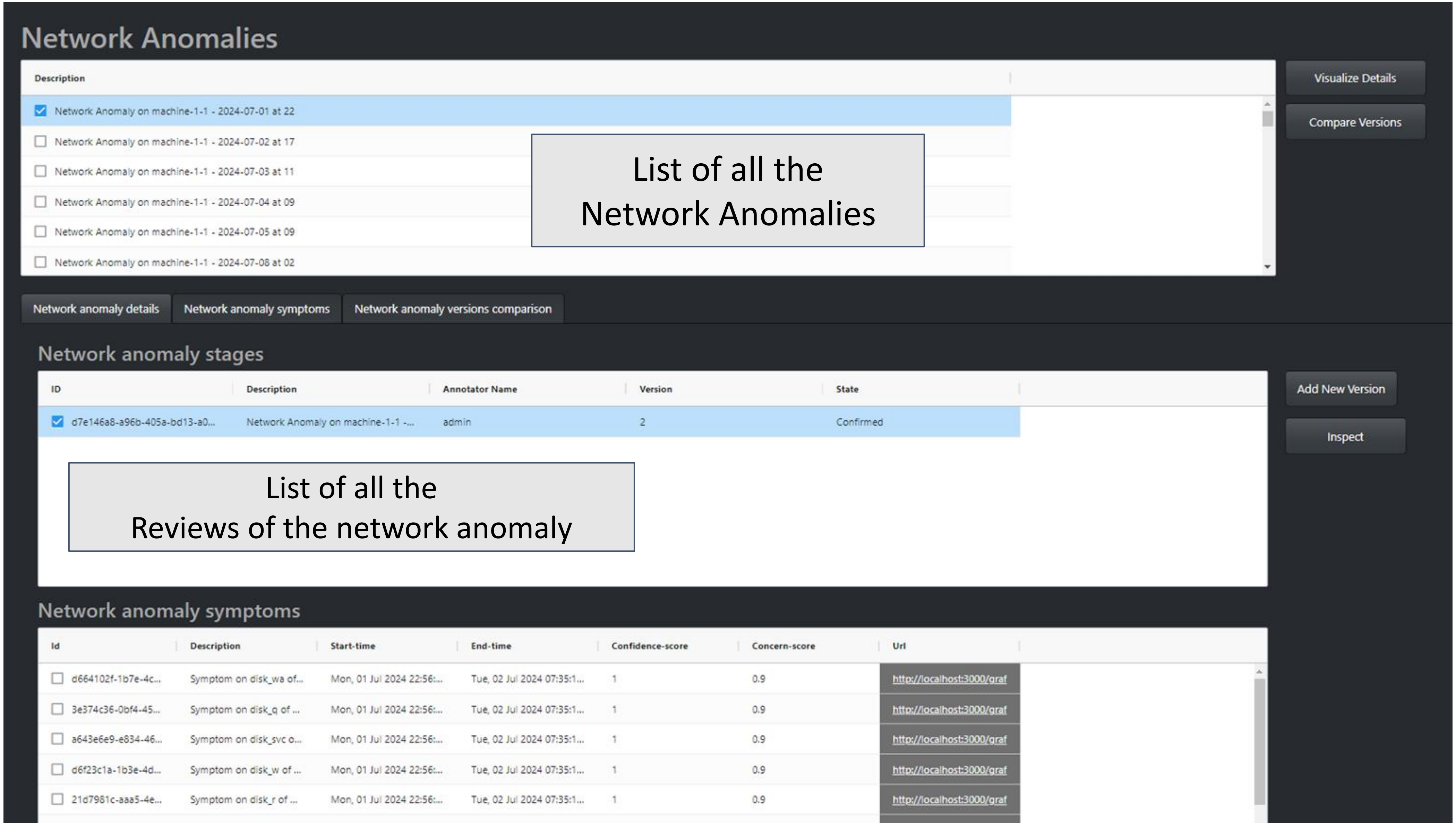
UC2: Validation of Labels on the dashboards - Tagging Network Anomalies



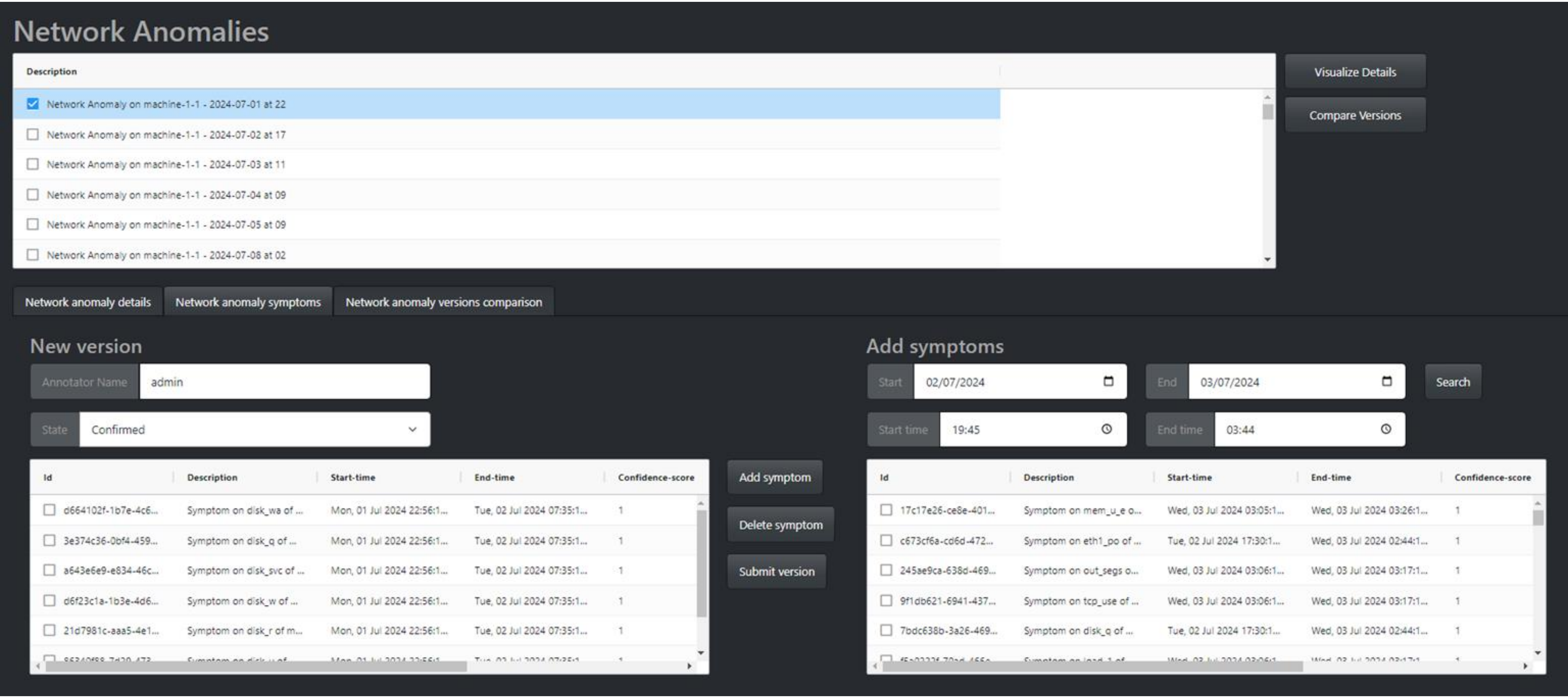
UC2: Validation of Labels on the dashboards - Tagging Symptoms



UC2: Validation of Labels on the dashboards - Analysing and editing network anomalies



UC2: Validation of Labels on the dashboards – Refine network anomalies



False Positives and False Negatives can be managed and corrected easily

UC3.1 – Refinement – Improving Detection rules [SAIN] (1 / 2)

The SAIN Agent defines symptoms and pushes them into the Label Store

```python
symptoms = [
    {
        'start-time': '2024-07-18T18:05:01',
        'end-time': '2024-07-18T23:12:42',
        "concern-score": 0.7,  # impact_score
        "confidence-score": 1,
        "description": "Symptom",
        "pattern": "",
        "tags": {
            "subservice_id": '66409aa8-c0d2-4703-82f6-d18b7aaf3841',
            "subservice_type": 'Device',
            "expression": "device.cpu_util > 90%"
        },
        "annotator": {
            "name": f"SAIN-Agent-{sain_agent_id}",
            "annotator_type": "algorithm"
        }
    },
    {
```

A network anomaly is created to group those symptoms together

```python
anomaly_dict = {
    "description": f'Device Under Stress',
    "state": "problem-potential",
    "version": 1,
    "symptoms": symptoms,
    "annotator": {
        "name": f"{sain_agent_id}",
        "annotator_type": "algorithm",
    }
}
```

UC3.1 – Refinement – Improving Detection rules [SAIN] (1 / 2)

A Network Engineer does the validation and deems the network anomaly to actually be a problem

```
anomaly_dict = {
    "id": network_anomaly_id,
    "description": f'Device Under Stress',
    "state": "problem-confirmed",
    "version": 2,
    "symptoms": symptoms,
    "annotator": {
        "name": f"{sain_user_id}",
        "annotator_type": "human",
    }
}
```

A Network Engineer in the post-mortem analysis, discovers another symptom that would have helped identifying the root cause faster.
This symptom is added to the list.

```
# Refinement of symptoms
new_symptom = {
    'start-time': '2024-07-18T18:05:01',
    'end-time': '2024-07-18T23:12:42',
    "concern-score": 1,  # impact on health score
    "confidence-score": 1,
    "description": "Symptom",
    "tags": {
        "subservice_id": 'f803f55c-2967-4d24-a81a-f83267dc4239',
        "subservice_type": 'Interface',
        "expression": "interface.receive-packet > 800000"
    },
    "annotator": {
        "name": f"{sain_user_id}",
        "annotator_type": "human"
    }
}

symptoms.append(new_symptom)
anomaly_dict = {
    "id": network_anomaly_id,
    "description": f'Device Under Stress and High Network Traffic',
    "state": "problem-confirmed",
    "version": 3,
    "symptoms": symptoms,
    "annotator": {
        "name": f"{sain_user_id}",
        "annotator_type": "human",
    }
}
```

UC3.2: Refinement (Retraining a ML model)

```python
params = {
    "start_time": start.strftime("%Y/%m/%dT%H:%M"),
    "end_time": end.strftime("%Y/%m/%dT%H:%M"),
    "annotator-type": "human"
}
response = requests.request(
    "GET", f"http://{ANTAGONIST_HOST}/api/rest/v1/symptom",
    headers=headers, data=payload, params=params)
response.raise_for_status()

#symptoms = format_symptoms(response.json(), start, end)
symptoms = response.json()
```

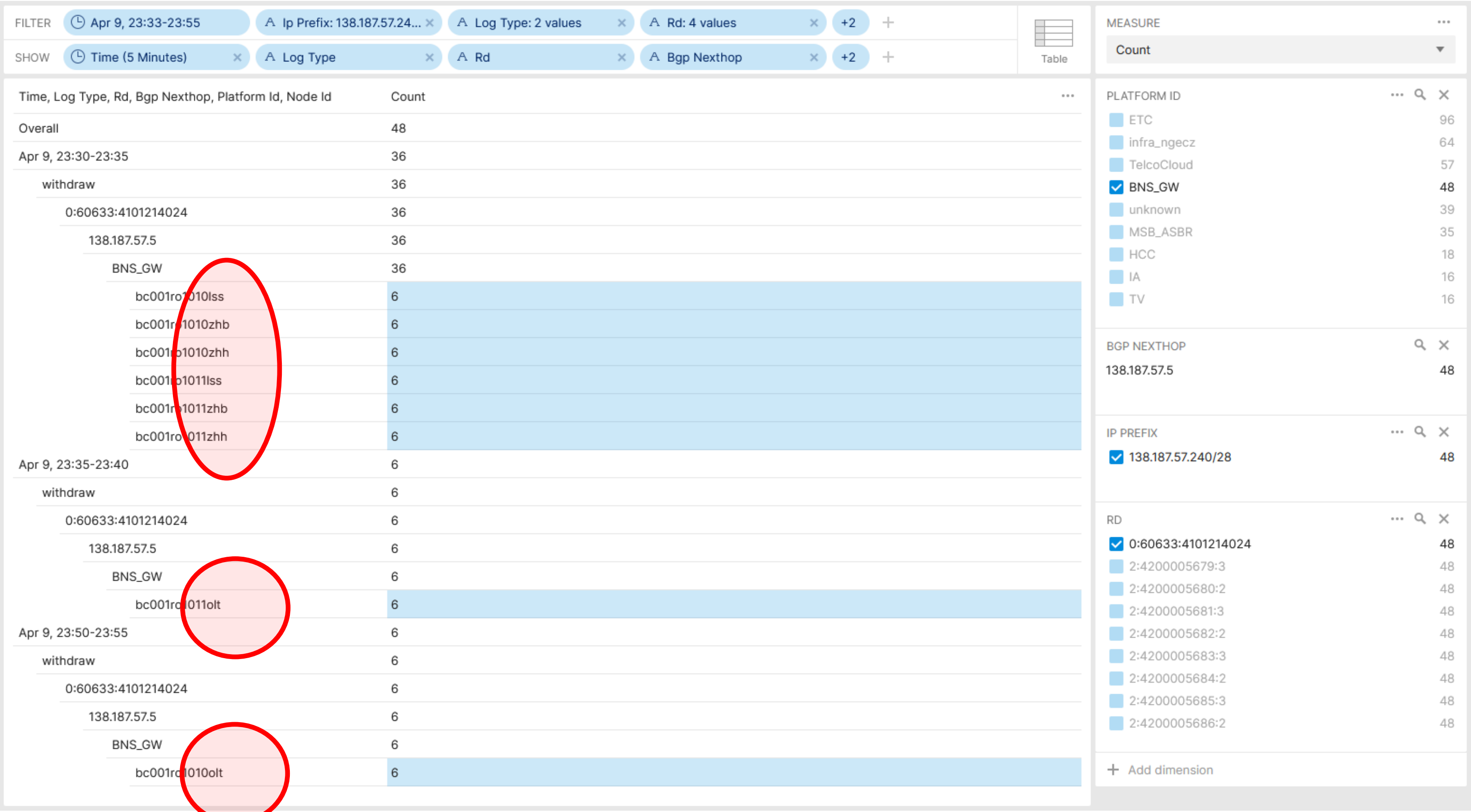
Get the symptom generated by humans

```python
import sys
sys.path.append('..')

from demo_anomaly_detector import autoencoder_detector
new_anomaly_detector = autoencoder_detector.DemoAnomalyDetector()

annotation_df = json_to_df(symptoms)
new_anomaly_detector.train(historical_telemetry_df, annotation_df, force=True)
```

Train a new model (which can then be compared with previous one - or others)

Postmortem, L3 VPN Pilot Migration - Voice Over IP

Post Maintenance Window Analysis



Overall BGP Update/withdrawals Across Swisscom MPLS/SRv6 Cores

Maintenance window was scheduled to start on April 9th 22:00 with a total of 4 migration steps.

At 12:45 CE facing interfaces on first PE **node to be phased out was disabled.** Triggering a VPNv4 withdrawal of 138.187.57.240/28 from 138.187.57.6 Lo0 towards route-reflectors and then to Inter-AS Option B ASBR. BMP route-monitoring and CLI show commands verified successful route propagation.

At 23:02 CE facing interfaces on first PE **node to be migrated to was enabled.** Triggering a VPNv4 update of 138.187.57.240/28 from 138.190.129.180 Lo0 towards route-reflectors and then to Inter-AS option B ASBR. BMP route-monitoring and CLI show commands verified successful route propagation.

SOS

At 23:34 CE facing interfaces on second PE **node to be phased out was disabled.** Triggering a VPNv4 withdrawal of 138.187.57.240/28 from 138.187.57.5 Lo0 towards route-reflectors and then to Inter-AS option B ASBR. BMP route-monitoring and CLI show commands verified successful route propagation to route-reflector **and on two Option B ASBR, but on other six after 20 mins delay.**

Postmortem, L3 VPN Pilot Migration - Voice Over IP

Show command drove to wrong conclusion

```
show route table bgp.l3vpn.0 protocol bgp 138.187.57.240/28 detail

60633:4101214024:138.187.57.240/28 (1 entry, 1 announced)
         BGP    Preference: 170/-101
                Route Distinguisher: 60633:4101214024
                Next hop type: Indirect, Next hop index: 0
                Address: 0x1a963a3c
                Next-hop reference count: 8
                Source: 138.190.128.116
                Protocol next hop: 138.187.57.5
                Label operation: Push 83714
                Label TTL action: prop-ttl
                Load balance label: Label 83714: None;
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Delete Int Ext ProtectionPath ProtectionCand>
                Local AS: 64088.1116 Peer AS: 64088.1116
                Age: 5:28       Metric: 805     Metric2: 4
                Validation State: unverified
                Resolving-AIGP: 4
                Effective metric: 8 (IGP metric plus resolving AIGP)
                Task: BGP_64088.1116.138.190.128.116
                Announcement bits (1): 1-BMP
                AS path: 60633 64088.5 ?
                Communities: 60633:204 60633:208 60633:1002 64497:4965
64499:13338 target:60633:1100006314
                Accepted
                BMP: Pre: withdraw Station: DAISY_BMP_1
                BMP: Pre: withdraw Station: DAISY_BMP_2
                BMP: Station: <unassigned>
                                Color: VPN Label: 83714
                Localpref: 100
                Router ID: 138.190.128.116
                Thread: junos-main
```

```
show route table bgp.l3vpn.0 protocol bgp 138.187.57.240/28 detail

60633:4103214024:138.187.57.240/28 (3 entries, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 60633:4103214024
                Next hop type: Indirect, Next hop index: 0
                Address: 0x1526757c
                Next-hop reference count: 4
                Source: 138.187.57.3
                Protocol next hop: 138.190.128.180
                Label operation: Push 83118
                Label TTL action: prop-ttl
                Load balance label: Label 83118: None;
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                State: <Active Ext ProtectionPath ProtectionCand>
                Local AS: 64088.1116 Peer AS: 60633
                Age: 14:29:45   Metric: 800     Metric2: 4
                Validation State: unverified
                Resolving-AIGP: 4
                Effective metric: 8 (IGP metric plus resolving AIGP)
                Task: BGP_60633.138.187.57.3
                Announcement bits (2): 0-BGP_RT_Background 1-BMP
                AS path: 60633 64088.1180 ?
                Communities: 60633:204 60633:208 60633:1001 60633:1111
64497:4965 64499:13338 target:60633:1100006314
                Accepted
                BMP: Pre: advertise Station: DAISY_BMP_1
                BMP: Pre: advertise Station: DAISY_BMP_2
                                Color: VPN Label: 83118
                Localpref: 100
                Router ID: 138.187.57.3
                Thread: junos-main
```

SOS

Juniper JunOS CLI show command shows that path is for 20min no longer primary active but still as backup path inactive. **Output mislead network engineer to believe that path is still installed.**

Postmortem, L3 VPN Pilot Migration - Voice Over IP

Data collection timestamp drove to wrong conclusion

```
 "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "bew03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.190.128.117",
    "string": "Tue Apr 09 2024 23:52:34"

  "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "bew03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.187.57.4",
    "string": "Tue Apr 09 2024 23:52:34"

  "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "bew03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.187.57.3",
    "string": "Tue Apr 09 2024 23:52:34"

 "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "bew03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.190.128.117",
    "string": "Wed Apr 10 2024 01:04:00"

  "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "zoi03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.187.57.3",
    "string": "Tue Apr 09 2024 23:54:09"

  "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "zoi03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.187.57.4",
    "string": "Tue Apr 09 2024 23:54:17"

  "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "zoi03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.190.128.117",
    "string": "Tue Apr 09 2024 23:54:24"

 "timestamp": "Tue Apr 09 2024 23:34:21",
  "writer_id": "zoi03bmp45c 20240220-1 (45ae4201)",
  "peer_ip": "138.190.128.117",
    "string": "Wed Apr 10 2024 00:54:51"
```

The yellow marked timestamp shows the optional BMP per-peer header observation timestamp.

The blue marked timestamp shows the timestamp being augmented on the BMP data collection and **being used for the time series database.**

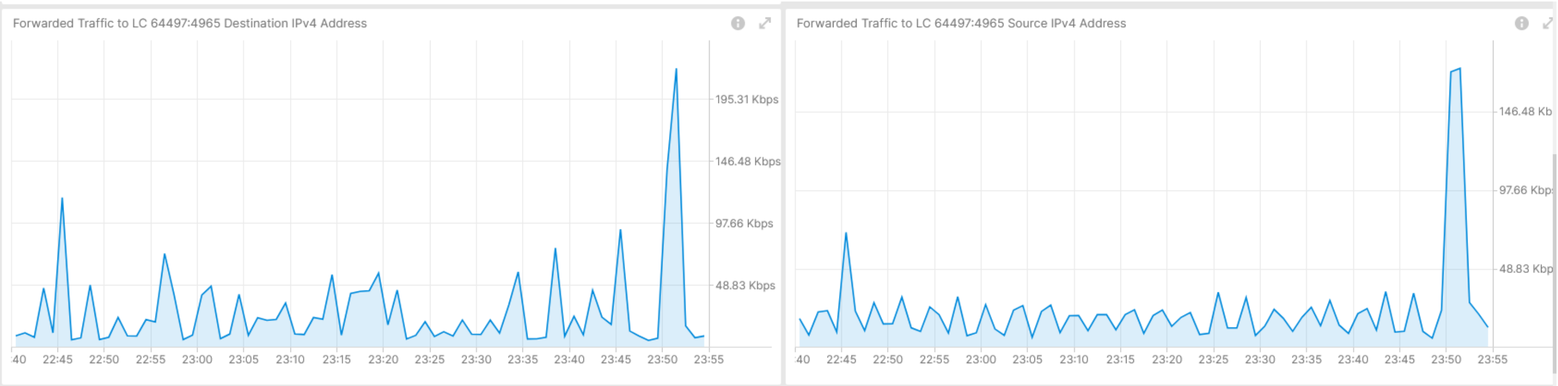
SOS

Because BMP per-peer timestamp is optional, in the time series database ingestion, the data collection augmentation timestamp is used instead. **Leading to false conclusions when the state change was observed.**

Postmortem, L3 VPN Pilot Migration - Voice Over IP

Route-Reflector Peering and L3 VPN Traffic View

BMP Peering Statistics on Route Reflectors



Traffic to Voice over IP Service on affected L3 VPN

IPFIX configured on PE **but not on involved MPLS Inter-AS option B ASBRs due to PR1567039.**

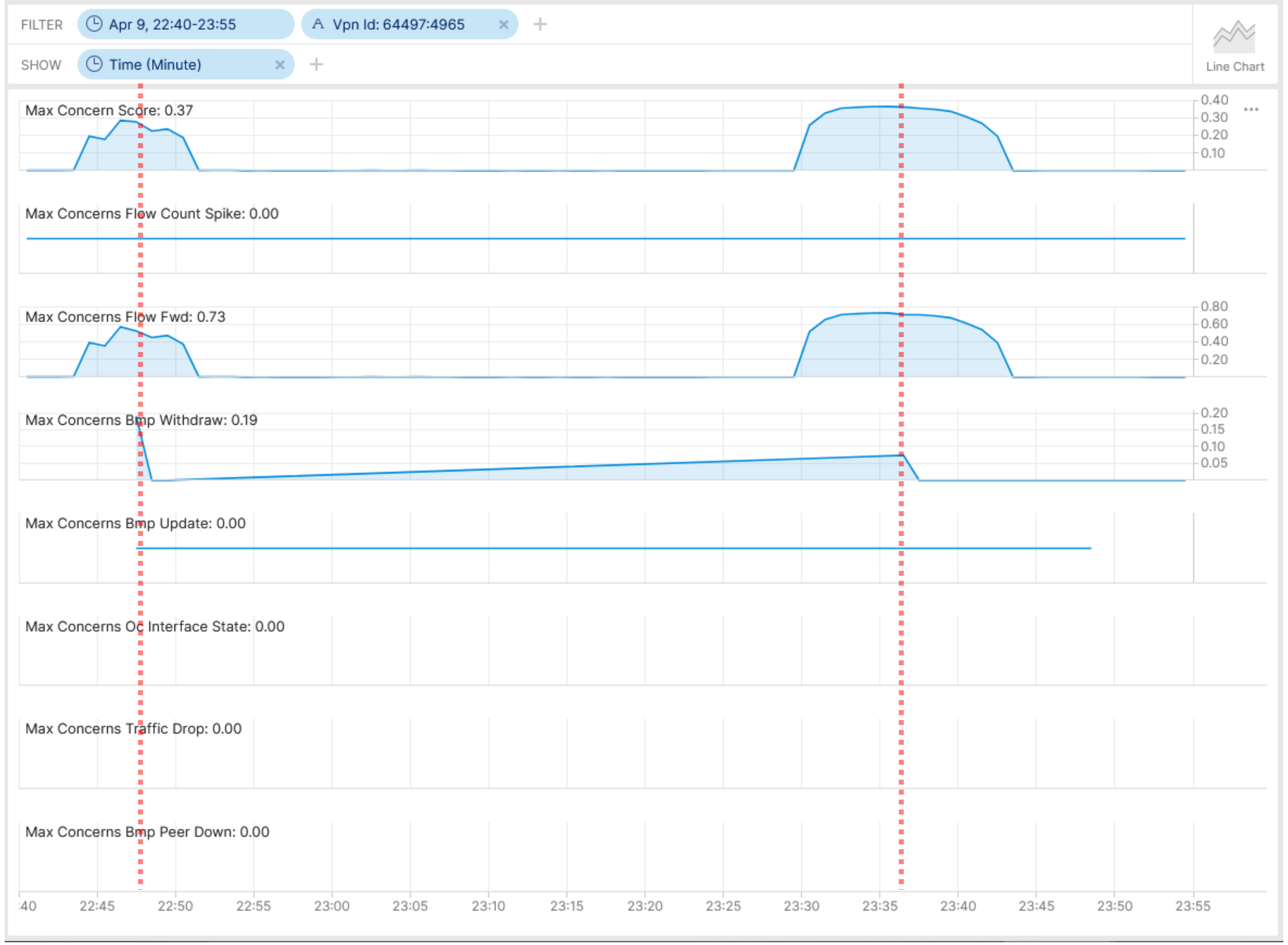
BMP ADJ-RIB In pre-policy on BGP VPNv4 /6 on MPLS PE's. BMP ADJ-RIB In pre-policy on BGP VPNv4 /6 on Route Reflectors and BMP ADJ-RIB In pre-policy and ADJ-RIB Out post-policy on Inter-AS Option B ASBR.

YANG Push on most nodes but not relevant for this use case.

Real-Time Streaming under Development

Postmortem, L3 VPN Pilot Migration - Voice Over IP

64497:4965 - Anomaly Detection - Live



Cosmos Bright Lights Anomaly Detection – 64497:4965

Max Concern Score: **NA**
BMP Withdrawal Score: **0.19**

- **BMP route-monitoring Update/Withdraw check recognize withdrawal.**
- BMP peer Down/Up check did not apply.
- **Interface Down/Up check did not recognize.**
- Traffic Drop spike did not apply.
- Missing Traffic check did not apply. (not fully implemented yet).
- Increased or decreased Flow Count did not apply.
- **Overall: 1 out of 6 checks have detected the BGP topology change. Real-time streaming implementation work in progress as expected.**

Postmortem
What to do next?

- **Support on upcoming maintenance window with verification dashboard and active monitoring.**
  -> Done

What went well?

Work in progress Cosmos Bright Lights real-time streaming Anomaly Detection BMP route-monitoring withdrawal rule detected topology change.

BMP collected metrics are consistent across multiple vendors vs. CLI show output is vendor dependent.

What could be improved?

BMP per-peer observation timestamp should be mandatory. See https://datatracker.ietf.org/doc/html/draft-boucadair-nmop-rfc3535-20years-later-02#section-4.7. **-> To be addressed in GROW/NMOP.**

BMP per-peer header should have an export timestamp. See https://datatracker.ietf.org/doc/html/draft-boucadair-nmop-rfc3535-20years-later-02#section-4.7. **-> To be addressed in GROW/NMOP.**

With RFC 8671 (Support for Adj-RIB-Out in BMP) path propagation could have been observed on route-reflectors.

With draft-lucente-grow-bmp-rel (Logging of routing events in BMP) path drops could been observed on Inter-AS option B ASBRs and route-reflectors.

With draft-ietf-grow-bmp-path-marking-tlv path status changed could have been observed on Inter-AS option B ASBRs.

Clarify why Juniper JunOS delayed BMP export for 20 resp. 80 minutes. Due to fact that the path was still passive in the BGP RIB?

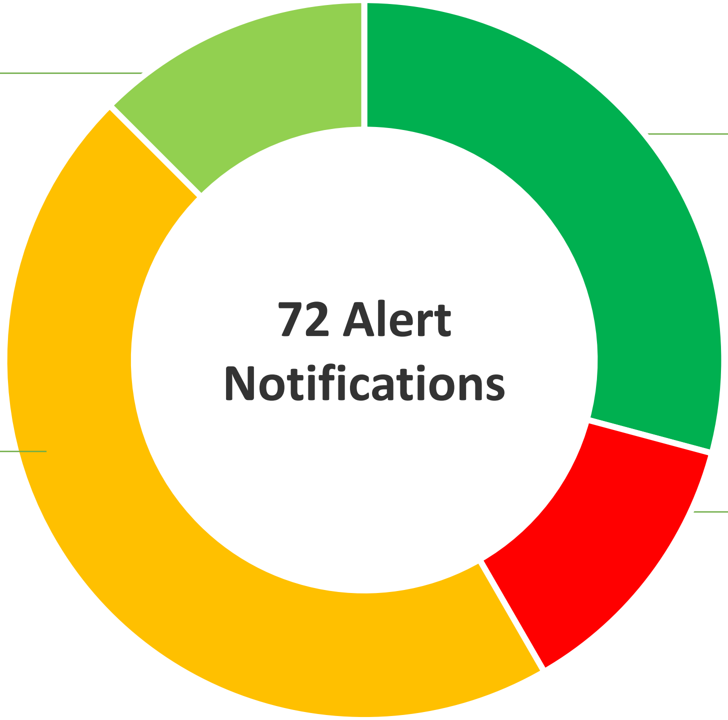
With IPFIX (**deconfigured due to PR1567039**) and support of IE90 ForwardingStatus (**not supported on Juniper JunOS**) forwarding drops could have been observed on Inter-AS option B ASBRs.

Swisscom - Cosmos Bright Lights PoC Summary

After 20 Incidents and 18 Months Time

5
In Replay
Incident Detected

46
Falls Positive due
to Apache Druid
delayed ingestion



15
In Near Real-Time
Incident Detected

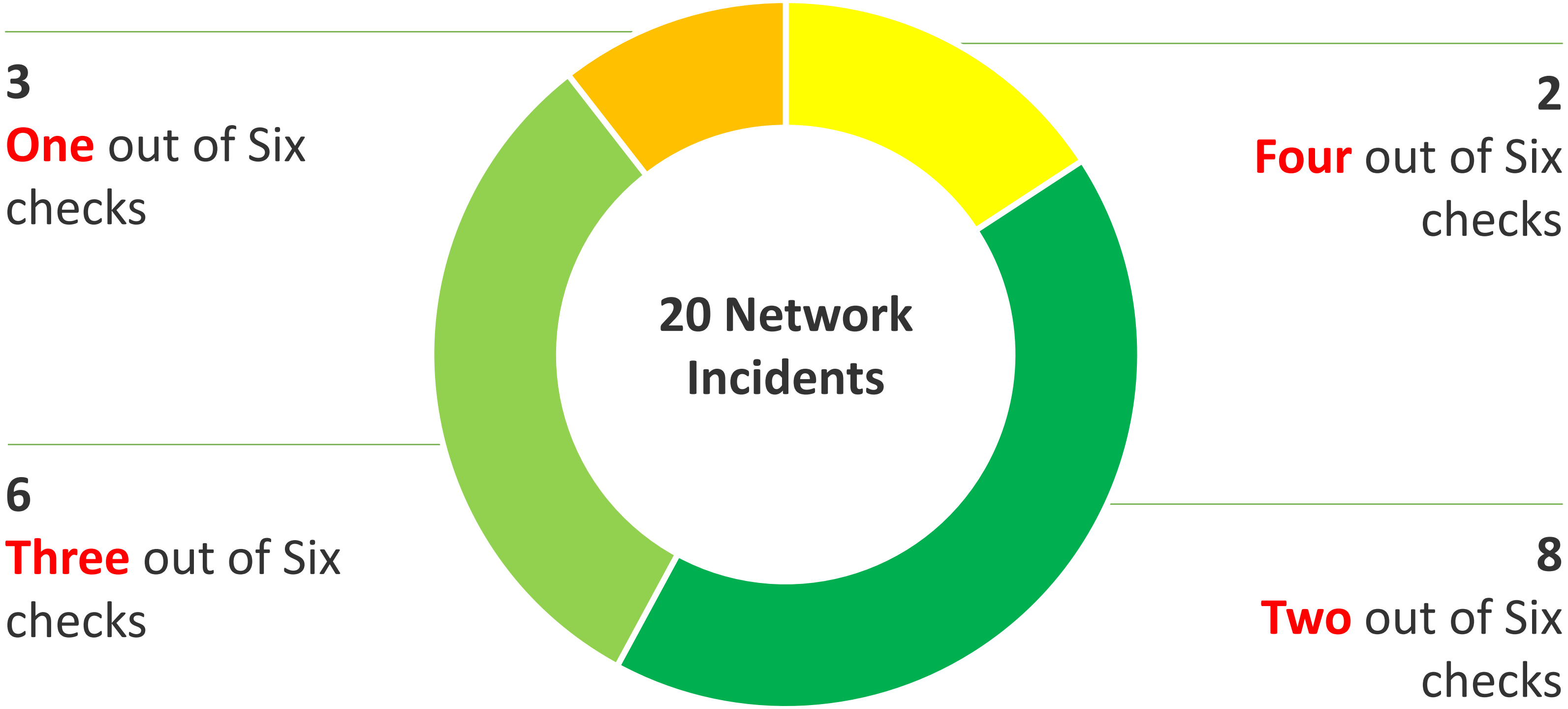
6
Falls Positive due to
previous week data
not comparable

Key Facts in V0 (2023-2024)

- 16 L3 VPNs proactively monitored.
- Individual Service Disruption Detection rule accuracy is beyond 90%. Summed accuracy is beyond 95%.
- Max Concern score ranged between 0.06 and 0.85. In average 0.46.
- In 4 cases additional YANG, in 13 cases additional BMP, in 2 cases Netconf Transaction-ID and 1 case additional L2 IPFIX metrics would have helped to gain more visibility.
- Key observability feature missing: BMP Local RIB with Path Marking.

Swisscom - Cosmos Bright Lights PoC Detail

Multiple Perspectives increases Accuracy



Key Improvements in V1 (2024)

- >12000 L3 VPNs proactively monitored since June 2024.
- Realtime Streaming eliminates delayed ingestion falls positives and scaling.
- Improved profiling. Compares to multiple previous weeks and discard largest deviation eliminates falls positives.
  -> Work In progress

Key Improvements in V2 (2025)

- Annotate operational and analytical Network Incident data for reproduction.
- Enabling automated workflow. From PowerPoint slide decks to data driven actionable insights.

Network Anomaly Detection Framework

Detect service interruption faster then humans can

Incident and Problem Management:

- Some Key Terms for Network Incident and Problem Management
  draft-ietf-nmop-terminology

Network Anomaly Detection:

- An Architecture for a Network Anomaly Detection Framework
  draft-netana-nmop-network-anomaly-architecture
- Semantic Metadata Annotation for Network Anomaly Detection
  draft-netana-nmop-network-anomaly-semantics
- Experiment: Network Anomaly Lifecycle
  draft-netana-nmop-network-anomaly-lifecycle

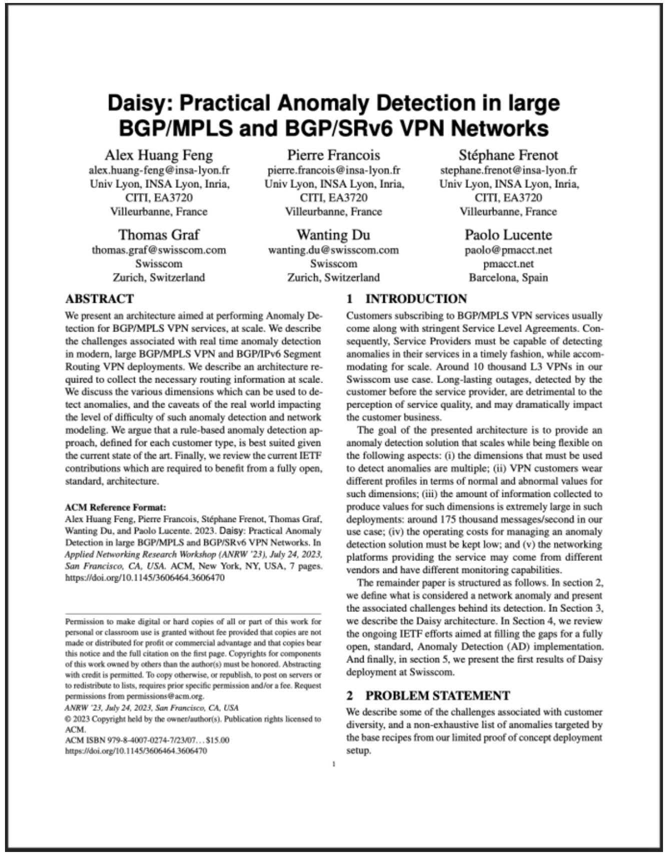
« Addressing the need to detect connectivity service interruption faster than humans can and facilitate collaboration by enabling exchange on labeled data with standardized semantics on a common framework.

Please consider to attend IETF 120 NMOP working group session on Friday 13:00 – 15:00 or go onto the mailing list and contribute to the discussion. »

Relevant Papers for more Details

Practical Anomaly Detection in Internet Services: An ISP centric approach

Alex Huang Feng*, Pierre Francois*, Kensuke Fukuda‡, Wanting Du†, Thomas Graf†, Paolo Lucente§, Stéphane Frénot*



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

Alex Huang Feng, Pierre Francois, Stéphane Frenot, Thomas Graf, Wanting Du, Paolo Lucente

Paper "**Practical Anomaly Detection in Internet Services: An ISP centric approach**"
Published at AnNet Workshop (In conjunction with IEEE NOMS)
Seoul, South Korea (6–10 May 2024)
Open access: https://hal.science/hal-04655324

Paper "**Daisy: Practical Anomaly Detection in large BGP/MPLS and BGP/SRv6 VPN Networks**" published at ACM/IRTF ANRW'23
San Francisco, USA (24 July 2023)
Open access: http://hal.science/hal-04307611


Yang
Kafka

Handling Operational YANG Modelled Data

State of the Union

Nowadays network operators are using **machine and human readable YANG** RFC 7950 to model their configurations and obtain YANG modelled data from their networks.

Network operators organizing their data in a Data Mesh where a message broker such as Apache Kafka facilitates the exchange of messages among data processing components.

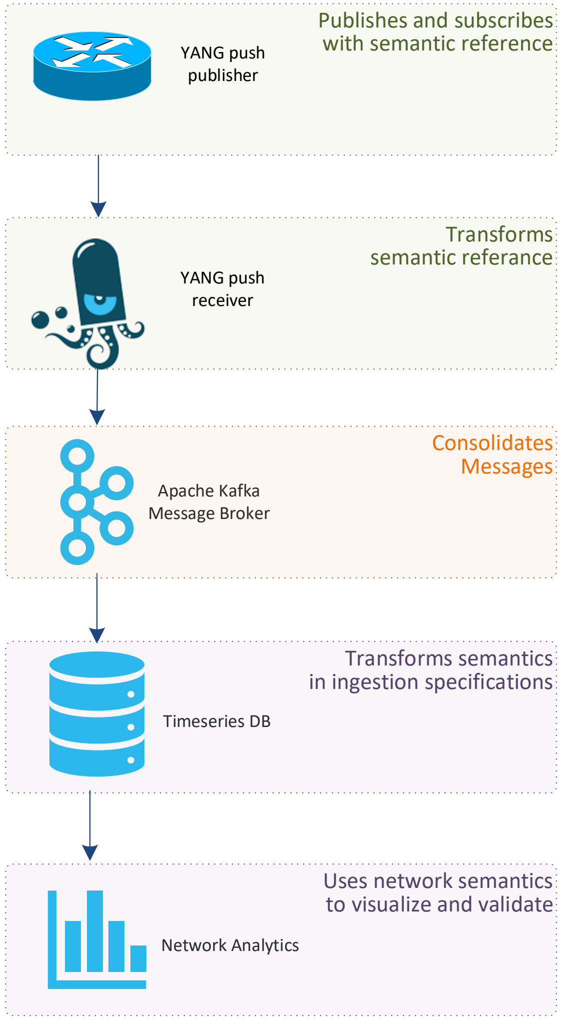
Today, subscribing to a YANG datastore, publishing a YANG modeled notifications message from the network and viewing the data in a time series database, **manual labor is needed to perform data transformation** to make a message broker and its data processing components with YANG notifications interoperable.

« Even though YANG is intend to ease the handling of data, this promise has not yet been fulfilled for Network Telemetry RFC 9232 »

From YANG-Push to Network Analytics

Aiming for an automated data processing pipeline

- **A network operator aims for:**
  - An **automated data processing pipeline** which starts with YANG-Push, consolidates at Data Mesh and ends at Network Analytics.
  - Operational metrics where **IETF defines the semantics.**
  - Analytical metrics where **network operators gain actionable insights.**
- **We achieve this by integrating YANG-Push into Data Mesh to:**
  - Produce metrics from networks **with timestamps when network events were observed.**
  - Hostname, publisher ID and sequence numbers help us to understand **from where metrics were exported and measure its delay and loss.**
  - Forward **metrics unchanged** from networks
  - **Learn semantics** from networks and validate messages.
  - **Control semantic** changes end to end.



Elements of the Architecture
Workflow Diagram

```
+---------------------------------------------------------+
|                  Time Series Database                   |
+---------------------------------------------------------+
                               ^
                               | (12) Ingest Data
                               | According to Schema
+---------------------------------------------------------+
|             Time Series Database Ingestion              |
+---------------------------------------------------------+
(10) Get |  ^                              ^ (9) Validate
  Schema |  |                              | Serialized Message
         |  |                              | Schema on Consumer
         |  |                              |
         |  |                     +-------------------+
         |  |                     |     Message       |
         |  |                     |     Broker        |
         |  |                     +-------------------+
         |  |                              |
         |  | (11) Issue                   | (8) Serialize
         |  |                              | YANG-Push Message
         |  |                              | annotated Schema
         v  | Schema          (6) Post     | ID on Producer
+-------------------+             Schema   +-------------------+
|       YANG        | <--------------      | Data Collection   |
|  Schema Registry  | -------------->      | YANG-Push Receiver|
+-------------------+ (7) Issue            +-------------------+
                      Schema ID     (4) Get |  ^ (3) Receive
                                     Schema |  | YANG-Push
                                            |  | Subscription
                                            |  | Start Message
                                            |  |   ^
                                            |  |   |
                                            |  |   | (5) Publish
                                            |  |   | YANG-Push
                                            |  |   | Message
                                            |  |   | with
                      (1) Discover Notif.   v  |   | Subscr. ID
+-------------------+ Capabilities         +-------------------+
| Manage YANG-Push  | -------------->      |   Network Node    |
|   Subscription    | (2) Subscribe        | YANG-Push Publisher|
+-------------------+ -------------->      +-------------------+
```

Figure 1: End to End Workflow

- **Network Orchestration** subscribes to YANG datastore.
- **Network Node** informs Data Collection on subscription state and publishes YANG metrics with YANG-Push.
- **Data Collection** obtains for each subscription the YANG module dependencies and the YANG modules on the network node, registers it in the YANG Schema Registry and prefixes the forwarded YANG notifications with the obtained schema ID.
- **YANG Schema Registry** issues for a Message Broker subject a schema ID for each new schema tree, compares a new schema tree with an existing and versions it.
- **Time Series Database Ingestion** consumes YANG-Push notifications from Message Broker, obtains schema tree from YANG schema registry, validates YANG notifications against schema and uses schema to populate into database table.

Address YANG Specification and Integration Gaps

Aiming for an automated data processing pipeline

YANG Specifications Gaps:

- YANG model for NETCONF Event Notifications
  draft-ahuang-netconf-notif-yang --------------------------------------→
- Validating anydata in YANG Library context
  draft-aelhassany-anydata-validation --------------------------------→

YANG Integration Gaps:

- Support of Network Observation Timestamping in YANG Notifications
  draft-tgraf-netconf-yang-push-observation-time ------------------------→
- Support of Hostname and Sequencing in YANG Notifications
  draft-tgraf-netconf-notif-sequencing -----------------------------------→
- Support of Versioning in YANG Notifications Subscription
  draft-ietf-netconf-yang-notifications-versioning ------------------------→
- Augmented-by Addition into the IETF-YANG-Library
  draft-lincla-netconf-yang-library-augmentation ------------------------→

« Addressing those gaps are a prerequisite to enable an automated data processing chain as described in draft-ietf-nmop-yang-message-broker-integration. **Please consider to attend IETF 120 NMOP working group session on Friday 13:00 – 15:00 or go onto the mailing list and contribute to the discussion.** »