Service Assurance for Intent-based Networking Architecture & YANG Modules for Service Assurance

draft-ietf-opsawg-service-assurance-architecture-01
B. Claise (Huawei), J. Quilbeuf (Huawei), D. Lopez (Telefonica), D. Voyer (Bell Canada), T. Arumugam (Cisco)

draft-ietf-opsawg-service-assurance-yang-01
B. Claise (Huawei), J. Quilbeuf (Huawei), P. Lucente (NTT), P. Fasano (Telecom Italia Mobile), T. Arumugan (Cisco)

IETF 111, OPSAWG
Issues & Proposal

• Issues:
  – When a service degrades, where is the fault? What are the symptoms? What is the root cause?
  – When a network component fails, which services are impacted?

• Service Assurance for Intent-based Networking Architecture proposal:
  – Decompose the problem into smaller components (=subservices)
  – The assurance graph links those subservices to map the service « intent »
  – The subservices are assured independently
  – Infer a service health score

• This complements the end-to-end (synthetic) monitoring
Assurance Graph PoC

List of impacted services:
- sain-tunnel CocaCola
- l2vpn-p2p HSBC
- sain-tunnel-ipv6 RedBull

Symptoms/Root causes:
- Output traffic on interface GigabitEthernet0/0/0/0 in ECMP bundle with(GigabitEthernet0/0/0/1) is not fairly balanced
- Output traffic on interface GigabitEthernet0/0/0/1 in ECMP bundle with(GigabitEthernet0/0/0/0) is not fairly balanced
Open Architecture with YANG Models

= Not Standardized

YANG

YANG
Open and Flexible Architecture

- Open architecture for multi-vendor support
  - How? With a YANG module:
    - Can augment the YANG module
    - Even for vendor-specific subservices

- Open architecture for multi-domains (wireline, wireless, 5G, VIM, etc.)
  - How? By linking domain-specific assurance graph
Architecture Draft: Update in v 01

• Covered the feedback from WG adoption
  – Thanks to Med Boucadair, Greg Mirsky, and others
• A lot of editorial improvements
• Connection with existing IETF work:
  – RFC 7149: Software-defined Networking, A Perspective from with a Service Provider Environment
  – RFC 7665: Service Function Chaining Architecture
  – RFC 8309: Service Models Explained
  – RFC 8969: A Framework for Automating Service and Network Management with YANG
• Stressed early in the document the companion YANG model
Architecture Draft: Update in v 01

- A real DAG
• NTP is not the requirement, a time synchronization mechanism is
  – And NTP is a candidate

3.9. Timing

The SAIN architecture requires **time synchronization**, with Network Time Protocol (NTP) [RFC5905] as a candidate, between all elements: monitored entities, SAIN agents, Service Configuration Orchestrator, the SAIN collector, as well as the SAIN Orchestrator. This **guarantees** the correlations of all symptoms in the system, correlated with the right assurance graph version.
module: ietf-service-assurance
  +--ro assurance-graph-version yang:counter32
  +--ro assurance-graph-last-change yang:date-and-time
  +--rw subservices
    +--rw subservice* [type id]
      +--rw type identityref
      +--rw id string
      +--ro last-change? yang:date-and-time
      +--ro label? string
      +--ro under-maintenance? boolean
      +--rw maintenance-contact string
      +--rw (parameter)?
      |  +--:(service-instance-parameter)
      |     +--rw service-instance-parameter
      |       +--rw service string
      |       +--rw instance-name string
      +--ro health-score? uint8
      +--ro symptoms-history-start? yang:date-and-time
    +--rw symptoms
    |  +--ro symptom* [start-date-time id]
    |     +--ro id string
    |     +--ro health-score-weight? uint8
    |     +--ro description? string
    |     +--ro start-date-time yang:date-and-time
    |     +--ro stop-date-time? yang:date-and-time
    +--rw dependencies
    +--rw dependency* [type id]
      +--rw type -> /subservices/subservice/type
      +--rw id -> /subservices/subservice[type=current()../type]/id
      +--rw dependency-type? identityref
module: ietf-service-assurance
+-ro assurance-graph-version   yang:counter32
+-ro assurance-graph-last-change yang:date-and-time
+-rw subservices
  +-rw subservice* [type id]
    +-rw type identityref
    +-rw id string

... 
+-rw (parameter)?
  | +-r:(service-instance-parameter)
  |  | +-rw service-instance-parameter
  |  |  | +-rw service string
  |  |  | +-rw instance-name string
  |  |  |  | +-r:(service-assurance-interface:parameters)
  |  |  |  |  | +-rw service-assurance-interface:parameters
  |  |  |  |  |  | +-rw service-assurance-interface:device string
  |  |  |  |  |  | +-rw service-assurance-interface:interface string
  |  |  |  |  |  |  | +-r:(service-assurance-device:parameters)
  |  |  |  |  |  |  |  | +-rw service-assurance-device:parameters
  |  |  |  |  |  |  |  |  | +-rw service-assurance-device:device string

+-rw dependencies
  +-rw dependency* [type id]
    +-rw type -> /subservices/subservice/type
    +-rw id -> /subservices/subservice[type=current()]/../type]/id
    +-rw dependency-type identityref

Two different subservices (device, and interface)
• “mandatory true” now added for the parameters

```yang
choice parameter {
    description
    "Specify the required parameters per subservice type.";
    container service-instance-parameter {
        when "derived-from-or-self(../type, 'service-assurance:service-instance-idty')";
        description
        "Specify the parameters of a service instance.";
        leaf service {
            type string;
            mandatory true;
            description
            "Name of the service.";
        }
        leaf instance-name {
            type string;
            mandatory true;
            description
            "Name of the instance for that service.";
        }
    }
}
```
YANG Module Draft: Update in v 01

• Added a “parameters” container for interface, to align the structure with the other subservice
  – “parameters” instead of the identity-name

```.yang
augment "/service-assurance:subservices/service-assurance:subservice/service-assurance:parameter" {
  description "Specify the required parameters for the interface-idty subservice type";
  container parameters {
    when "derived-from-or-self(../service-assurance:type, 'interface-idty')";
    description "Required parameters for the interface-idty subservice type";
    leaf device {
      type string;
      mandatory true;
      description "Device supporting the interface.";
    }
    leaf interface {
      type string;
      mandatory true;
      description "Name of the interface.";
    }
  }
}
```
YANG Module Draft: Update in v 01

- Complete set of YANG modules for the architecture draft example
- Added the “IP connectivity” and “IS-IS” subservice
YANG Module Draft: Update in v 01

• New “Appendix A. Example of YANG instances”
  – Validated with yangson

• New “Appendix B. YANG Library for Service Assurance”
YANG Module Draft: Update in v 01

• New section on “guidelines for subservice extension”
  – Module name
  – Module namespace
  – Module prefix
  – Specific identity
  – Parameters
Open Issue, Feedback, and Questions


• Thanks to those who provided/will provide feedback.
Service Assurance for Intent-based Networking Architecture & YANG Modules for Service Assurance

draft-ietf-opsawg-service-assurance-architecture-01
B. Claise (Huawei), J. Quilbeuf (Huawei), D. Lopez (Telefonica), D. Voyer (Bell Canada), T. Arumugam (Cisco)

draft-ietf-opsawg-service-assurance-yang-01
B. Claise (Huawei), J. Quilbeuf (Huawei), P. Lucente (NTT), P. Fasano (Telecom Italia Mobile), T. Arumugan (Cisco)

IETF 111, OPSAWG

BACKUP SLIDES
Flexible Architecture

Could be a single box

Agent could be in or off routers
Global Architecture
Conclusion

• Working prototype
  – still lots of work to do (see previous slides)

• See
  – our work on telemetry
    • https://people.montefiore.uliege.be/bdonnet/telemetry/
  – DxAgent implementation
    • https://github.com/ekorian/dxagent
  – IOAM Agent implementation
    • https://github.com/IurmanJ/ioam-agent
  – CLT
    • https://github.com/IurmanJ/cross-layer-telemetry