Per-Node Capabilities for Optimum Operational Data Collection

draft-claise-netconf-metadata-for-collection-02

Benoit Claise (Huawei), Munish Nayyar (Cisco), Adithya Reddy Sesani (Cisco)

IETF 111, virtual
History: Automation is as Good as ...

- The number of YANG models
- The toolchain
- The YANG models metadata
  - The per-node capabilities

Why? For closed loop automation
Example and Use Case

• Practical example: an interface counter is not updated more frequently than 30 secs.
  – If poll/stream more frequently, the closed loop automation system could take the wrong action

• Knowing the minimum update frequency is key:
  • To draw the right conclusions
  • To help reduce the load on the devices

• Use Case: closed loop automation (reconfiguring network based on observed, network state changes), service assurance, intent-based networking
  – draft-ietf-opsawg-service-assurance-architecture
  – draft-ietf-opsawg-service-assurance-yang-01
Problem Statement

• Need a way to learn from the servers how granular its telemetry and data can be, to provide post-processing analytics and closed loop automation

• Therefore
  – Need a series of extra information about the node capabilities
  – Specified a YANG module that provides per-node capabilities for optimum operational data collection.
Reminder: ietf-netconf-notification-capabilities

- ietf-system-capabilities: provides a structure that can be used to specify YANG related system capabilities for servers
- ietf-notification-capabilities augments ietf-system-capabilities to specify capabilities related to telemetry
  - on-change-supported, minimum-update-period, supported-update-period, minimum-dampening-period
New Specifications

• New YANG module: ietf-system-node-metadata
• Augments ietf-system-capabilities to publish the metadata information specific to YANG node-identifier
• Provides per-node capabilities for optimum operational data collection
• Provide 2 RPCs for simplified operations
module: ietf-system-node-metadata

augment /sysc:system-capabilities/sysc:datastore-capabilities/
   sysc:per-node-capabilities/sysc:node-selection/sysc:node-selector:
      +--ro minimum-observable-period?                     uint64
      +--ro suggested-observable-period?                   uint64
      +--ro optimized-measurement-point?                   empty {optimized-measurement-point-feature}?
      +--ro corresponding-mib-oid?                        yang:object-identifier-128
      +--ro related-node?                                  yang:node-instance-identifier
RPCs

rpcs:
  +---x get-measurement-metadata
    |   +---w input
    |   |   +---w node-selector? yang:node-instance-identifier
    |   +--ro output
    |       +--ro optimized-measurement-point? yang:node-instance-identifier {optimized-measurement-point-feature}?
    |       |   +--ro computed-observable-period? uint64
    |       +--ro active-measurements* []
    |       +--ro subscribed-measurement-period? uint64
  +---x get-system-node-capabilities
    +---w input
    |   +---w node-selector? yang:node-instance-identifier
    +--ro output
    +--ro node-selector-capability* []
      +--ro node? yang:node-instance-identifier
      +--ro minimum-observable-period? uint64
      +--ro suggested-observable-period? uint64
      +--ro optimized-measurement-point? empty {optimized-measurement-point-feature}?
      +--ro corresponding-mib-oid? yang:object-identifier-128
      +--ro related-node? yang:node-instance-identifier
Open Issues

• Difference between minimum-update-period and minimum-observable-period

• "related-node" should be split into two: "related-config-node" and "related-state-node"?

• Explain how to use the RPC from the client side, along with the different options.

• Expand on the active measurement use case

• nanosecond: an overkill?
Feedback

- Do you recognize the problem statement?
- Should we solve this problem?
- Ask:
  - Read the draft
  - Provide feedback