

Per-Node Capabilities for Optimum Operational Data Collection

[draft-claise-netconf-metadata-for-collection-03](#)

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

IETF 113

Automation is as Good as ...



The number of YANG models



The toolchain



The YANG models metadata



The per-node capabilities

YANG-Push Notification Capabilities

[RFC9196]



- **ietf-system-capabilities**: provides a placeholder structure that can be used to discover YANG-related system capabilities for servers.
- **ietf-notification-capabilities** augments "ietf-system-capabilities" to specify capabilities related to "Subscription to YANG Notifications for Datastore Updates" (RFC 8641):
 - Max-nodes-per-update,
 - periodic-notifications-supported,
 - minimum-update-period,
 - supported-update-period,
 - on-change-supported,
 - minimum-dampening-period

This draft

- Provides some additional **per-node capabilities for optimum operational data collection** (as an extension to RFC9196)
- Provide 2 RPCs for simplified operations
- Disclaimer: draft not updated, waiting for feedback before updating

Suggested and computed observable-period

```
leaf suggested-observable-period {
  type uint64;
  units "nanoseconds";
  description
    "The suggested observable period for this node-selector.
    This value represents factory default suggested
    information, only available at implementation time.";
}
```

- Different than the RFC9196 minimum-update-period

```
leaf computed-observable-period {
  type uint64;
  units "nanoseconds";
  description
    "the computed observable period for this node-selector (and
    optimized-measurement-point). The system internally
    dynamically computes the suggested observable period
    (relevant for polling or streaming cadence) which can be
    greater-or-equal to the minimal-observable-period.
    Since this value is dynamic, this metadata is only
    available in a run time environment.";
}
```

- Ex: FIB observable-period depends on the FIB size

Optimized-measurement-point

```
leaf optimized-measurement-point {  
  if-feature "optimized-measurement-point-feature";  
  type empty;  
  description  
    "This node-selector is an optimized measurement point."  
}
```

- In some server design, operational data are usually modeled/structured in a way that the relevant data are grouped together and reside together.
- In most cases, it is more performant to fetch this data together than as individual leaves: data are structured together internally, grouped together, and therefore fetched together.

Corresponding-mib-oid

```
leaf corresponding-mib-oid {  
  type yang:object-identifier-128;  
  description  
    "The object identifier (OID) assigned to a SMIV2 definition,  
    corresponding to this node-selector."  
}
```

- The object identifier (OID) assigned to a SMIV2 definition, corresponding to the node-selector.
- Existing SNMP MIBs based automations can use this information to migrate to more analytics-ready YANG Modeled data

Related-node

```
leaf related-node {  
  type yang:node-instance-identifier;  
  description  
    "In case the node instance is an operational node then the  
    associated node-instance-identifier represents the config  
    leaf directly related to this operational node. In case the  
    node instance is an config node then the associated  
    node-instance-identifier represents the operational leaf  
    directly related to this configuration node. A typical  
    example is the relationship between the admin-status and  
    oper-status, which is impossible to detect automatically in  
    a non-NMDA environment or for non-openconfig YANG modules.  
    The related-node SHOULD NOT reported for NMDA architectures  
    and openconfig YANG modules.";  
}
```

- For non NMDA, non openconfig: oper and config
- Ex: RFC7223 admin-status & oper-status

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 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

- "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

Feedback

- Do you recognize the problem statement?
- Which objects are relevant to you?
- Ask:
 - Read the draft
 - Provide feedback
 - Let's work on it, or forget about it