

A Data Manifest for Contextualized Telemetry Data

[draft-claise-opsawg-collected-data-manifest-04](#)

B. Claise (Huawei), J. Quilbeuf (Huawei), D. Lopez (Telefonica), I. Dominguez (UPM), T. Graf (Swisscom)

IETF 114, OPSAWG

Goal & Problem Statement

- End goal: analyze the data, from the data collection system, with the proper context, for anomaly detection and, in the end, closed loop automation
- How were data actually metered, under which circumstances?
- What are the platform characteristics?
- Goal is not to expose new information via YANG but rather to define what needs to be kept as metadata (or Data Manifest) to ensure that the data can still be interpreted correctly even:
 - if the source device is not accessible (from the collection system)
 - If the source device has been updated or has a new configuration

Example: Data Collection Vantage Point

Counter1 = 42
Status = UP

No updates since t1

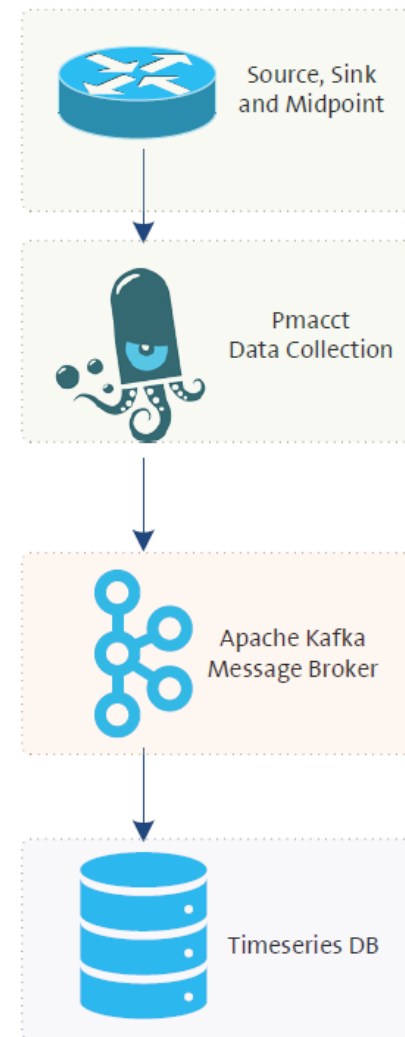
- Problem: Telemetry issue?
- Problem: Sender overloaded?
- Problem: bug?
- No problem: long telemetry cadence?
- No problem: "on-change"?
- No problem: "suppress-redundancy"?



End goal: analyze the data, from the data collection system, with the proper context, for anomaly detection and, in the end, closed loop automation

Proposal: Data Manifest

- Data Manifest composed of 2 YANG models for storing the context:
 - **Platform manifest**: part of the Data Manifest that completely characterizes the platform producing the data.
 - **Data Collection manifest**: part of the Data Manifest that completely characterizes how and when the telemetry was metered.
- “It’s expected that the data manifest is streamed directly from the network equipment, along with telemetry data”
- “MUST be streamed all with the data and stored along with the collected data.”
 - Expected to be streamed from the router directly
 - If not, could be build from the collector
- “In case the data are moved to different place (typically a database), the data manifest MUST follow the collected data.”




Data Collection Manifest

```
module: ietf-collected-data-manifest
  +--ro data-collection
    +--ro mdt-subscriptions* [subscription-id]
      +--ro subscription-id          uint64
      +--ro datastore?              ds:datastore-ref
      +--ro mdt-path-data-manifest* [path]
        +--ro path                  yang:xpath1.0
        +--ro requested-period?     uint64
        +--ro actual-period?        uint64
        +--ro on-change?            boolean
        +--ro suppress-redundancy?  boolean
```

List indexed by subscription ID



Specify datastore per subscription
- monitor diff between config and operational
- monitor candidate



MDT path typed as XPath



Unsigned ints for periods



New: YANG module versioning in Platform Manifest

```
module: ietf-collected-data-platform-manifest
+--ro platform
  +--ro name?          string
  +--ro vendor?        string
  +--ro software-version? string
  +--ro software-flavor? string
  +--ro os-version?    string
  +--ro os-type?       string
  +--ro yang-library
    +--ro module-set* [name]
      +--ro name          string
      +--ro module* [name]
        +--ro name          yang:yang-identifier
        +--ro revision?     revision-identifier
        +--ro namespace     inet:uri
        +--ro location*     inet:uri
        +--ro submodule* [name]
          +--ro name          yang:yang-identifier
          +--ro revision?     revision-identifier
          +--ro location*     inet:uri
          +--ro revision-label? rev:revision-label
        +--ro feature*      yang:yang-identifier
        +--ro deviation*    -> ../../module/name
        +--ro revision-label? rev:revision-label
      +--ro import-only-module* [name revision]
        +--ro name          yang:yang-identifier
        +--ro revision      union
        +--ro namespace     inet:uri
        +--ro location*     inet:uri
        +--ro submodule* [name]
          +--ro name          yang:yang-identifier
          +--ro revision?     revision-identifier
          +--ro location*     inet:uri
          +--ro revision-label? rev:revision-label
        +--ro revision-label? rev:revision-label
      +--ro schema* [name]
        +--ro name          string
        +--ro module-set*
          -> ../../module-set/name
        +--ro deprecated-nodes-implemented? boolean
        +--ro obsolete-nodes-absent?      boolean
      +--ro datastore* [name]
        +--ro name          ds:datastore-ref
        +--ro schema        -> ../../schema/name
  +--ro packages-set
    +--ro package* [name version]
      +--ro name          -> /pkgs:packages/package/name
      +--ro version       leafref
      +--ro checksum?    leafref
```

Platform info, similar to yangcatalog

Supported YANG modules: YANG-Library

YANG package

New: YANG module versioning in Platform Manifest

```
module: ietf-collected-data-platform-manifest
+--ro platform
  +--ro name?          string
  +--ro vendor?        string
  +--ro software-version? string
  +--ro software-flavor? string
  +--ro os-version?    string
  +--ro os-type?       string
  +--ro yang-library
    +--ro module-set* [name]
      +--ro name          string
      +--ro module* [name]
        +--ro name          yang:yang-identifier
        +--ro revision?    revision-identifier
        +--ro namespace    inet:uri
        +--ro location*    inet:uri
        +--ro submodule* [name]
          +--ro name          yang:yang-identifier
          +--ro revision?    revision-identifier
          +--ro location*    inet:uri
          +--ro revision-label? rev:revision-label
        +--ro feature*    yang:yang-identifier
        +--ro deviation*  -> ../../module/name
        +--ro revision-label? rev:revision-label
      +--ro import-only-module* [name revision]
        +--ro name          yang:yang-identifier
        +--ro revision      union
        +--ro namespace    inet:uri
        +--ro location*    inet:uri
        +--ro submodule* [name]
          +--ro name          yang:yang-identifier
          +--ro revision?    revision-identifier
          +--ro location*    inet:uri
          +--ro revision-label? rev:revision-label
        +--ro revision-label? rev:revision-label
    +--ro schema* [name]
      +--ro name          string
      +--ro module-set*
        +--ro name          string
        +--ro location*    inet:uri
        +--ro revision-label? rev:revision-label
      +--ro deprecated-nodes-implemented? boolean
      +--ro obsolete-nodes-absent?      boolean
    +--ro datastore* [name]
      +--ro name          ds:datastore-ref
      +--ro schema      -> ../../schema/name
  +--ro packages-set
    +--ro package* [name version]
      +--ro name          -> /pkgs:packages/package/name
      +--ro version      leafref
      +--ro checksum?    leafref
```

In the YANG source for platform-manifest:

```
// This is a copy-paste of the augments from the module
// ietf-yang-library-revisions.
// We just changed the path to which the data is added

augment "/p-mf:platform/p-mf:yang-library/p-mf:module-set"
  + "/p-mf:module" {
  description
    "Add a revision label to module information";
```

Copy pasted from ietf-yang-library-revisions

```
augment "/yanglib:yang-library/yanglib:module-set/yanglib:module" {
  description
    "Add a revision label to module information";
```

Copy pasting from ietf-yang-library-revisions does not seem right.

Is there a way to include directly the augmented version of ietf-yang-library?

New: other updates of YANG modules

- ✓ Fixed YANG validation warnings
- ✓ Fixed YANG tree and module formatting, xml2rfc is happier!
- ✓ Shortened prefixes

New when clauses

```
leaf requested-period {  
  when "../on-change = 'false'";  
  type uint64;  
  description  
    "Requested period, in millisecond, between two successive  
    updates."  
}  
leaf actual-period {  
  when "../on-change = 'false'";  
  type uint64;  
  description  
    "Period in milisecond between two successive updates  
    actually applied by the plattform at configuration time.  
    This period can be larger than the requested period as the  
    platform might adjust it for performance reasons.";
```


Open Questions

- Identified Improvements:
 - Include the source of data somewhere and notably some self-assertiveness
<https://github.com/JeanQuilbeufHuawei/draft-collected-data-manifest/issues/2>
 - Data integrity
 - Use Private Enterprise Number (PEN) from IANA to identify vendors
- Open Questions:
 - Do we want to handle data manifest for non-telemetry values ? SNMP, IPFIX ...?
=> inclined NOT to extend the scope
 - Many ways to identify which YANG modules are supported by a server: Would a pointer to a specific location be more efficient?
 - How to properly specify devices / virtual devices as source?
 - Handle mis-collections ? More Counters ?
 - MDT subscription is a collection of XPathS but NETCONF subscription is a xml filter what to use as key for subscription contents?
<https://github.com/JeanQuilbeufHuawei/draft-collected-data-manifest/issues/9>
 - Should we include encoding? (XML, JSON, CBOR, Protobuf)

Conclusion

- New draft versions (Thanks Med & Tianran & Ignacio & Joe for your review)
- We believe it's a valid problem
- We would like to request WG adoption

Feedback, suggestions, issues, PRs:

<https://github.com/JeanQuilbeufHuawei/draft-collected-data-manifest>

A Data Manifest for Contextualized Telemetry Data

[draft-claise-opsawg-collected-data-manifest-04](#)

B. Claise (Huawei), J. Quilbeuf (Huawei), D. Lopez (Telefonica), I. Dominguez (UPM), T. Graf (Swisscom)

IETF 114, OPSAWG

Platform Manifest

```
module: ietf-collected-data-platform-manifest
  +--ro platform
    +--ro name?          string
    +--ro vendor?       string
    +--ro software-version? string
    +--ro software-flavor? string
    +--ro os-version?   string
    +--ro os-type?      string
    +--ro yang-library
      +--ro module-set* [name]
        +--ro name          string
        +--ro module* [name]
          +--ro name          yang:yang-identifier
          +--ro revision?    revision-identifier
          +--ro namespace    inet:uri
          +--ro location*    inet:uri
          +--ro submodule* [name]
            +--ro name          yang:yang-identifier
            +--ro revision?    revision-identifier
            +--ro location*    inet:uri
            +--ro feature*    yang:yang-identifier
            +--ro deviation*  -> ../../module/name
          +--ro import-only-module* [name revision]
            +--ro name          yang:yang-identifier
            +--ro revision      union
            +--ro namespace    inet:uri
            +--ro location*    inet:uri
            +--ro submodule* [name]
              +--ro name          yang:yang-identifier
              +--ro revision?    revision-identifier
              +--ro location*    inet:uri
        +--ro schema* [name]
          +--ro name          string
          +--ro module-set*  -> ../../module-set/name
        +--ro datastore* [name]
          +--ro name          ds:datastore-ref
          +--ro schema        -> ../../schema/name
      +--ro packages-set
        +--ro package* [name version]
          +--ro name          -> /pkgs:packages/package/name
          +--ro version        -> /pkgs:packages/package[pkgs:name = current()/../name]/version
          +--ro checksum?     -> /pkgs:packages/package[pkgs:name = current()/../name][pkgs:version = current()/../version]/pkg%2checksum
```

New: Vendor

New: Include full yang-library, to have datastores

New: rw->ro for every node, Manifest is NOT configurable

New: Add packages-set

Update Frequency

- Platform manifest:
 - **Update when:** platform changes (i.e. at reboot)
- Data collection manifest:
 - **Update when:** collection condition changes:
 - New subscription
 - Collection period is adjusted based on CPU availability

Collecting Manifests: Use Telemetry

- Platform and data collection manifests are about telemetry
 - we can assume a telemetry system is available
- Event-driven Telemetry/onChange well suited to stream the manifests updates only
- Collectors can choose or not to collect the manifests / header approach would force collector to have the manifests
- Collectors know IDs of their subscription -> YANG key for selecting the data-manifest they need

Mapping Data to Data Manifest

- Collected data needs the following metadata:
 - Source device -> for mapping to platform manifest
 - Subscription ID
 - MDT path
- } For mapping data to data manifest

Specifying how to store metadata for collected data is out-of-scope of this draft.