GMOC Simple Time-Series data transfer API (version 0.1)

Camilo Viecco (cviecco@grnoc.iu.edu)

Introduction

GMOC needs to store and retrieve data for sending time-series data that is not locally generated by GMOC. The goal is for GMOC to store/expose data from multiple sources so that a single unified mechanism to access times-series data for GENI is possible. This API is in addition to the GMOC exchange format [GMOC-EX] which is for non-time-series operational data. This document explains the rational and description for the GMOC time-series API.

Motivation

We need a simple api to move time-series based data (mostly RRD files) from multiple devices so that each site can have a copy of the data.
Problems:

1. RRD files are architecture and compiler dependent.
2. Time series data will not always be RRD.
3. We do not want to expose the file-system of either the sender of receiver of the data
4. We desire some more granular access control mechanisms.
5. A single call should be enough to transfer multiple data sets.
6. There is no meta-data embedded in the rrd files.

Solution

Develop a simple API to allow transfer for generic time-series data. In particular we need to define the file-format for the data transfer. The transport mechanism would be a REST based interface, the error handling should be returned as HTTP error codes.

The data transfer format MUST

1. Include the time-series Data
2. Include meta-data about the time-series data including the units for each column
3. allow for null columns or null values in each column
4. Allow multiple time-series to be expressed in a single call.

e
The data back-end SHOULD:

1. Be able to retrieve arbitrary data periods for the data inserted.
2. Be able to export data in the in the API.
3. Be able to provide a query format so that searching on the time-series meta-data is possible,

This API MUST include running implementations of both the sender and the receiver of the data.

File-Format

The data for the APU is exchanged in an XML file with data stored in a single XML field, very much like KML [OCG-KML] . The Relax-NG schema of the data format Follows.

```
datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"

grammar {

start = element timeseries_data {timeseries-data-content}

timeseries-data-content =
  attribute version {"0.1"},
  attribute time {xsd:integer}?,
  attribute originator {text}?,
  element type_info {type-info-content}+,
  element node_info {host-info-content}+,
  element data_group {data-group-content}+

type-info-content=
  attribute name {text},
  element column_names {text},
  element column_units {text}

host-info-content=
  attribute name{text},
  element location{text},
  element tags{text}

data-group-content =
  attribute name {text},
  attribute interval_type {text}?,
  attribute start {xsd:integer},
  attribute end   {xsd:integer},
  attribute heartbeat {xsd:integer},
  attribute step {xsd:integer},
  element column_names {text},
  element type {text},
  element node_name {text},
  element local_name{text}?,
  element tags {text}?,
  element data{text}

}
```

Example File

For those who like examples better here is an example file:

```
<?xml version="1.0" encoding="UTF-8"?>
<timeseries_data originator="someone" time="1301940403" version="0.1">
    <type_info name="interface">
        <column_names>status,inoctets,inpackets,outoctets,outpackets,inerrors,outerrors</column_names>
        <column_units>NULL,bps,pps,bps,pps,pps,pps</column_units>
    </type_info>
    <type_info name="host_stats">
        <column_names>temp,one_min_cpu,five_min_cpu,mem_total,mem_used,mem_free,mem_cached,disk_total,d
isk_free,io_bus_util,cpu_user,cpu_system,cpu_idle,cpu_wio,cpu_nice,cpu_int</column_names>
        <column_units>deg_celcius,%,%,MB,MB,MB,MB,GB,GB,%,%,%,%,%,%,%</column_units>
    </type_info>
    <node_info name="atla.of.nlr.net">
        <location>atla</location>
        <tags>openflow,nlr</tags>
    </node_info>
    <data_group name="XX" interval_type="open_closed" start="1301940200" end="1301940400"
heartbeat="300" step="100" >
        <column_names>temp,one_min_cpu</column_names>
        <type>host_stats</type>
        <node_name>atla.of.nlr.net</node_name>
        <tags>silly</tags>
        <data>
            1301940100,10.3,5;
            1301940200,10,10;
            1301940300,10,1.1e1;
        </data>
    </data_group>
    <data_group name="XY" interval_type="open_closed" start="1301940200" end="1301940400"
heartbeat="300" step="50">
        <column_names>temp,one_min_cpu</column_names>
        <type>host_stats</type>
        <node_name>atla.of.nlr.net</node_name>
        <tags>openflow</tags>
        <data>
            1301940100,10.3,5;
            1301940200,10,10;
            1301940300,10,1.1e1;
        </data>
    </data_group>
    <data_group name="XZ" interval_type="open_closed" start="1301940200" end="1301940400"
heartbeat="300" step="50">
        <column_names>inoctets,inpackets,outoctets,outpackets</column_names>
        <type>interface</type>
        <node_name>atla.of.nlr.net</node_name>
        <local_name>eth0</local_name>
        <data>
            1301940100,4000,12,5500,10;
            1301940200,8000,20,6000,20;
            1301940300,1000,5,4000,15;
        </data>
    </data_group>
</timeseries_data>
```

Notice how multiple data point are represented in a single line, each line separated by a semi-colon and each value in the line separated by a comma.

Type-info

The type-info sections describe how data should be aggregated together, the available columns for each data series of the types and the units for each column. The column names must be a list of comma separated names. This list is a superset of all the column names for the time-series of the same type.

Host-info

The host info section include basic data about hosts. It is assumed that each time-series data reports data about a host (for link based data is on the host where the data is measured). Hosts must include a location name and can include a arbitrary tags. However see the tags section for choosing your tags.

Data-group

The data group section includes data about the actual time-series. All the data of the time series is inserted in a single xml field, but this filed must be formatted as a set of semi-colon separated lines, where each line is a comma separated field.

Tags:

Tags are used to categorize the various time-series or hosts. While users can use arbitrary tags for their data series and hosts, each data consumer SHOULD specify which tags are reserved for special use. The rationale for this suggestion is that consumers of data would like to prevent 'pollution' of the tag name-spaces. By 'blessing' some of the tags, the blessed tags can be treated specially by the consumer for better data classification.

For GMOC the following tags are 'blessed' and will appear under the GMOC TAG hierarchy. The sets of blessed tags appear in the GMOC blessed tags table.

TAG location	'Blessed' Values
Host	myplc, of, rtr,switch, edge-node
Time-series	backbone,commodity, peer

Table 1: GMOC blessed Tags

All other tags will be places in the custom sub-tree of the categories.

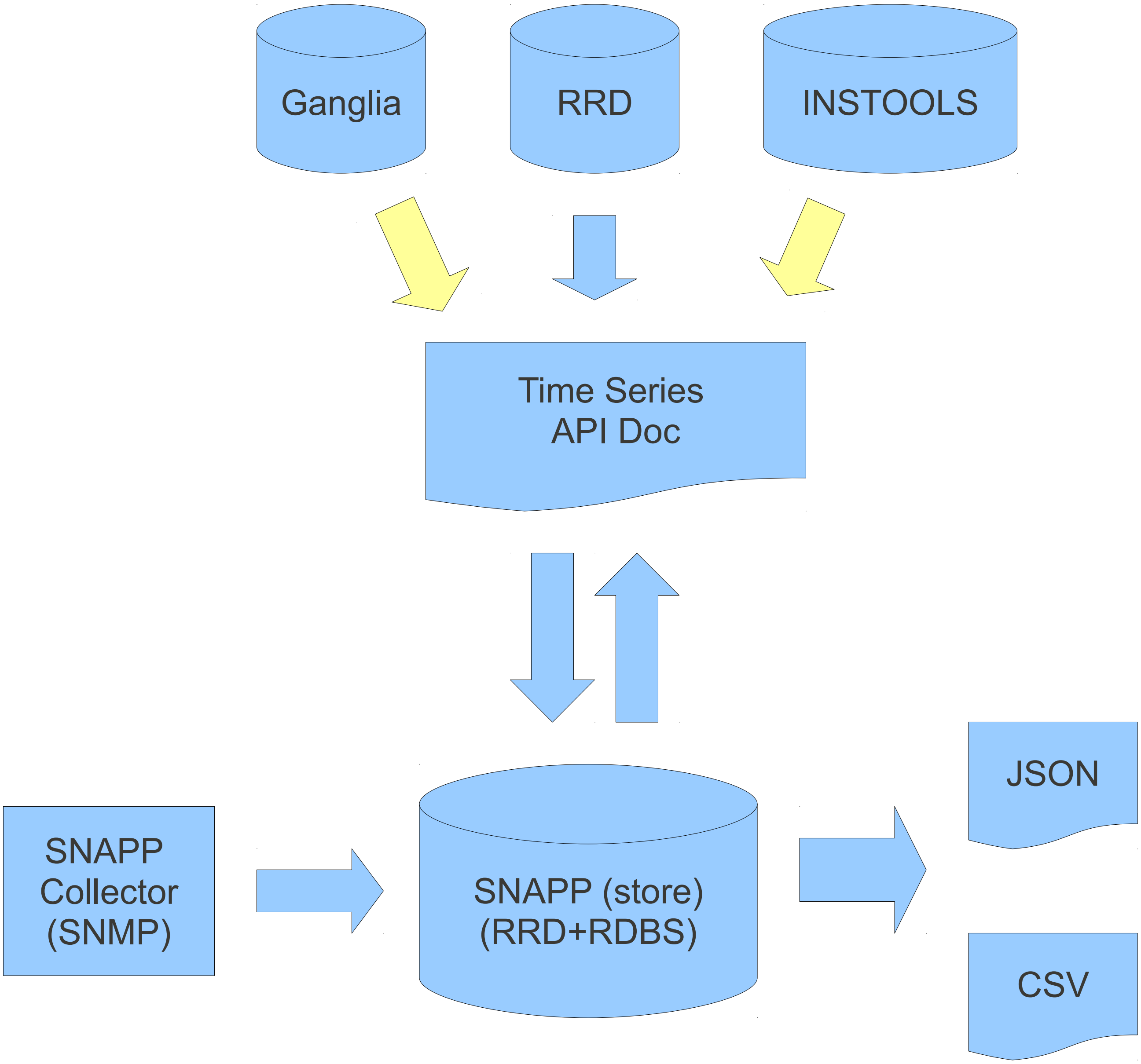
References

[GMOC-EX] GMOC Topology-Entity Data Exchange Format Specification http://gmoc.grnoc.iu.edu/gmoc/index/documents/gmoc-data-exchange-format--initial-document-describing-native-gmoc-data-exchange-format

[OCG-KML] Open Geospatial Consortium KML standard. http://www.opengeospatial.org/standards/kml/

APPENDIX

GMOC Measurement Data Architecture



Inserting data into the GMOC time-series archive could be problematic if authentication and access control are not carefully managed. The data import scripts must be able to both determine correct authentication as well as a well defined policy on what data the authenticated user is allowed to create/update.

It is currently expected to have two types of importers: Slice importers (such as INSTOOLS and On-TIME-Measure) that should be able to create/modify data for the slice and the nodes of their running slice, and Organization importers (such as GPO) that should be able to create/modify data for infrastructure that they manage.

For GEC12 both slice users and organization importers MUST be allowed to proceed following some authentication mechanism. Authorization (the scope of access) for writing would probably not be solved. Authorization for reading SHOULD be ready also be GEC12. Authentication credentials for slices should be done automatically, credentials for organizations could be done manually.

In the picture above, the yellow arrows represents transformation scripts that have not been implemented yet. There is however a utility built to rapidly convert an arbitrary number of files into the format. This is available at: http://gmoc-db.grnoc.iu.edu/sources/.