Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2016

J. Schoenwaelder
V. Bajpai
Jacobs University Bremen

July 3, 2015

A YANG Data Model for LMAP Measurement Agents
draft-ietf-lmap-yang-01.txt

Abstract

This document defines a data model for Large-Scale Measurement Platforms (LMAP). The data model is defined using the YANG data modeling language.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 4, 2016.

Table of Contents

1. Introduction

This document defines a data model for Large-Scale Measurement Platforms (LMAP) [I-D.ietf-lmap-framework]. The data model is defined using the YANG [RFC6020] data modeling language. It aims to be consistent with the LMAP Information Model [I-D.ietf-lmap-information-model].

1.1. Terminology

This document uses the LMAP terminology defined in [I-D.ietf-lmap-framework].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.2. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.

- Abbreviations before data node names: "rw" means configuration (read-write), and "ro" means state data (read-only).

- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Data Model Overview

The tree diagram below shows the structure of the configuration model.

```
module: ietf-lmap-control
   +--rw lmap
      +--rw agent
      |  +--rw agent-id?             yang:uuid
      |  +--rw device-id?            inet:uri
      |  +--rw group-id?             string
      |  +--rw report-agent-id?      boolean
      |  +--rw controller-timeout?   uint32
      +--rw schedules
      |  +--rw schedule* [name]
      |     +--rw name              string
      |     +--rw event             -> /lmap/events/event/name
      |     +--rw action* [name]
      |     |  +--rw name           string
      |     |  +--rw task           -> /lmap/tasks/task/name
      |     |  +--rw option* [name]
      |     |  |  +--rw name     string
      |     |  |  +--rw value?   string
      |     |  +--rw destination*   -> /lmap/schedules/schedule/name
      |     +--rw execution-mode    enumeration
      +--rw suppression
      |  +--rw enabled?        boolean
      |  +--rw stop-running?   boolean
      |  +--rw start?          yang:date-and-time
      |  +--rw end?            yang:date-and-time
      |  +--rw task*           -> /lmap/tasks/task/name
      |  +--rw schedule*       -> /lmap/schedules/schedule/name
      +--rw tasks
      |  +--rw task* [name]
      |     +--rw name                   string
      |     +--rw (task-identification)
      |     |  +--:(registry)
      |     |  |  +--rw registry*             inet:uri
```

```
|     |  +--:(program)
|     |     +--rw program?              string
|     +--rw option* [name]
|     |  +--rw name     string
|     |  +--rw value?   string
|     +--rw tag*                    string
|     +--rw suppress-by-default?   boolean
+--rw events
   +--rw event* [name]
      +--rw name             string
      +--rw (event-type)?
      |  +--:(periodic)
      |  |  +--rw periodic
      |  |     +--rw interval    uint32
      |  |     +--rw start?      yang:date-and-time
      |  |     +--rw end?        yang:date-and-time
      |  +--:(calendar)
      |  |  +--rw calendar
      |  |     +--rw month*             union
      |  |     +--rw weekday*           union
      |  |     +--rw day-of-month*      union
      |  |     +--rw hour*              union
      |  |     +--rw minute*            union
      |  |     +--rw second*            union
      |  |     +--rw timezone-offset?   timezone-offset
      |  |     +--rw start?             yang:date-and-time
      |  |     +--rw end?               yang:date-and-time
      |  +--:(one-off)
      |  |  +--rw one-off-time     yang:date-and-time
      |  +--:(immediate)
      |  |  +--rw immediate        empty
      |  +--:(startup)
      |     +--rw startup          empty
      +--rw random-spread?   int32
```

The tree diagram below shows the structure of the state model.

```
module: ietf-lmap-control
   +--ro lmap-state
      +--ro agent
      |  +--ro agent-id        yang:uuid
      |  +--ro device-id       inet:uri
      |  +--ro hardware        string
      |  +--ro firmware        string
      |  +--ro version         string
      |  +--ro last-started    yang:date-and-time
      +--ro tasks
         +--ro task* [name]
            +--ro name                      string
            +--ro (task-identification)
            |  +--:(registry)
            |  |  +--ro registry*                inet:uri
            |  +--:(program)
            |     +--ro program?                 string
            +--ro last-completion?          yang:date-and-time
            +--ro last-status?              status-code
            +--ro last-message?             string
            +--ro last-failed-completion?   yang:date-and-time
            +--ro last-failed-status?       status-code
            +--ro last-failed-message?      string
```

The tree diagram below shows the structure of the notification (reporting) model.

```
module: ietf-lmap-report
notifications:
   +---n report
      +--ro date        yang:date-and-time
      +--ro agent-id?   yang:uuid
      +--ro group-id?   string
      +--ro task* [name]
      |  +--ro name                   string
      |  +--ro (task-identification)
      |  |  +--:(registry)
      |  |  |  +--ro registry?             inet:uri
      |  |  +--:(program)
      |  |     +--ro program?              string
      |  +--ro option* [name]
      |  |  +--ro name     string
      |  |  +--ro value?   string
      |  +--ro tag*                   string
      |  +--ro suppress-by-default?   boolean
      +--ro header
      |  +--ro column*   string
      +--ro row*
         +--ro start       yang:date-and-time
         +--ro end?        yang:date-and-time
         +--ro conflict*   string
         +--ro value*      string
```

3. Relationship to the Information Model

The LMAP information model [I-D.ietf-lmap-information-model] is devided into six sections. They are mapped into the YANG data model as explained below:

- Pre-Configuration Information: This is not modeled explicitly since it is a subset of the configuration information.

- Configuration Information: This is modeled in the /lmap/agent subtree and the /lmap/schedules and /lmap/tasks subtrees described below. Some items have been left out because they are expected to be dealt with by the underlying protocol.

- Instruction Information: This is modeled in the /lmap/suppression subtree and the /lmap/schedules and /lmap/tasks subtrees described below.

- Logging Information: Some of the logging information, in particular 'success/failure/warning messages in response to information updates from the Controller', will be handled by the

protocol used to manipulate the lmap specific configuration. [[CREF1: It needs to be discussed whether we can rely on informal syslog messages that can be accessed via protocols such RFC 5277 or whether we want to define specific notifications in the YANG data model. --JS]]

- Capability and Status Information: Some of the status information is modeled in the /lmap-state/agent subtree. Information about network interfaces can be obtained from the interfaces YANG data model [RFC7223]. The list of supported tasks is modeled in the /lmap-state/tasks subtree including information about the last execution and the last failed execution.

- Reporting Information: This is modeled by the report notification.

These six sections are build on the following common information objects:

- Schedules: This is modeled in the /lmap/schedules subtree.

- Channels: Channels are not modeled since the NETCONF and RESTCONF server configuration data model [I-D.ietf-netconf-server-model] already provides a mechanism to configure NETCONF and RESTCONF server channels.

- Task Configurations: This is modeled in the /lmap/tasks subtree.

- Event Information: This is modeled in the /lmap/events subtree.

4. YANG Modules

The modules import definitions from [RFC6991] and [RFC7223].

```
<CODE BEGINS> file "ietf-lmap-control@2015-07-03.yang"
module ietf-lmap-control {

  namespace "urn:ietf:params:xml:ns:yang:ietf-lmap-control";
  prefix "lmapc";

  import ietf-yang-types {
    prefix yang;
  }
  import ietf-inet-types {
    prefix inet;
  }

  organization
```

```
       "IETF Large-Scale Measurement Platforms Working Group";

     contact
       "WG Web:   <http://tools.ietf.org/wg/lmap/>
        WG List:  <mailto:lmap@ietf.org>

        Editor:   Juergen Schoenwaelder
                  <j.schoenwaelder@jacobs-university.de>

        Editor:   Vaibhav Bajpai
                  <v.bajpai@jacobs-university.de>";

     description
       "This module defines a data model for controlling measurement
        agents that are part of a Large-Scale Measurement Platform
        (LMAP).";

     revision "2015-07-03" {
       description
        "Initial version";
       reference
        "RFC XXX: A YANG Data Model for LMAP Measurement Agents";
     }

     /*
      * Typedefs
      */

     typedef weekday {
       type enumeration {
         enum sunday {
           value 0;
           description "Sunday of the week";
         }
         enum monday {
           value 1;
           description "Monday of the week";
         }
         enum tuesday {
           value 2;
           description "Tuesday of the week";
         }
         enum wednesday {
           value 3;
           description "Wednesday of the week";
         }
         enum thursday {
           value 4;
```

```
          description "Thursday of the week";
        }
        enum friday {
          value 5;
          description "Friday of the week";
        }
        enum saturday {
          value 6;
          description "Saturday of the week";
        }
      }
      description
       "A type modeling the weekdays in the Greco-Roman
        tradition.";
    }

    typedef month {
      type enumeration {
        enum january {
          value 1;
          description "January of the Julian and Gregorian calendar";
        }
        enum february {
          value 2;
          description "February of the Julian and Gregorian calendar";
        }
        enum march {
          value 3;
          description "March of the Julian and Gregorian calendar";
        }
        enum april {
          value 4;
          description "April of the Julian and Gregorian calendar";
        }
        enum may {
          value 5;
          description "May of the Julian and Gregorian calendar";
        }
        enum june {
          value 6;
          description "June of the Julian and Gregorian calendar";
        }
        enum july {
          value 7;
          description "July of the Julian and Gregorian calendar";
        }
        enum august {
          value 8;
```

```
          description "August of the Julian and Gregorian calendar";
        }
        enum september {
          value 9;
          description "September of the Julian and Gregorian calendar";
        }
        enum october {
          value 10;
          description "October of the Julian and Gregorian calendar";
        }
        enum november {
          value 11;
          description "November of the Julian and Gregorian calendar";
        }
        enum december {
          value 12;
          description "December of the Julian and Gregorian calendar";
        }
      }
      description
        "A type modeling the month in the Julian and Gregorian
         tradition.";
    }

    typedef wildcard {
      type string { pattern '\*'; }
      description
        "A wildcard for calendar scheduling entries.";
    }

    typedef status-code {
      type int32;
      description
        "A status code returned by the execution of a task. Note that
         the actual range is implementation dependent but it should
         be portable to use values in the range 0..127 for regular
         exit codes. By convention, 0 indicates successful termination.
         Negative values may be used to indicate abnormal termination
         due to a signal; the absolute value may identify the signal
         number in this case.";
    }

    typedef timezone-offset {
      type string {
        pattern 'Z|[\+\-]\d{2}:\d{2}';
      }
      description
        "A timezone-offset as it is use in the yang:date-and-time
```

```
          type. The value Z is equivalent to +00:00. The value -00:00
          indicates and unknown time-offset.";
      }

      /*
       * Groupings
       */

      grouping timing-start-end-grouping {
        description
          "A grouping that provides start and end times for
           timing objects.";
        leaf start {
          type yang:date-and-time;
          description
            "The date and time when the timing object
             starts to create triggers.";
        }
        leaf end {
          type yang:date-and-time;
          description
            "The date and time when the timing object
             stops to create triggers.

             It is generally a good idea to always configure
             an end time and to refresh the configuration
             of timing object as needed to ensure that agents
             that loose connectivity to their controller
             do not continue their tasks forever.";
        }
      }

      grouping task-options-grouping {
        description
          "A list of options of a task. Each option is a name/value
           pair (where the value may be absent).";

        list option {
          key "name";
          ordered-by user;

          description
            "A list of options passed to the task. It is a list of
             key / value pairs and may be used to model options.
             Options may be used to identify the role of a task
             or to pass a channel name to a task.";

          // XXX This is kind of broken since making the option name
```

```
        // XXX a key means that a certain option may only appear once.
        // XXX This is not workable since some tests require a list of
        // XXX targets. Turning this into a leaf-list of args also
        // XXX does not work since YANG requires leaf-list values to
        // XXX be unique. Oops.

        leaf name {
          type string;
          description
            "The name of the option.";
        }

        leaf value {
          type string;
          description
            "The value of the option.";
        }
      }
    }

    grouping task-grouping {
      description
        "A grouping that defines the configuration of a task.";

      list task {
        key name;
        description
          "The list of tasks configured on the LMAP agent.";

        leaf name {
          type string;
          description
            "The unique name of a task.";
        }

        choice task-identification {
          mandatory true;
          description
            "Information that identifies the task.";

          leaf-list registry {
            type inet:uri;
            description
              "A registry entry identifying the metrics a measurement
               task supports.";
          }

          leaf program {
```

```
            type string;
            description
              "The (local) program to invoke in order to execute
                 the task.";
          }
        }

        uses task-options-grouping {
          description
            "The list of task specific options.";
        }

        leaf-list tag {
          type string;
          description
            "A tag contains additional information that is passed
             with the result record to the collector. A tag can be
             used to carry the Measurement Cycle ID.";
        }

        leaf suppress-by-default {
          type boolean;
          default true;
          description
            "Indicates whether the task will be suppressed by
             a default supression.";
        }
      }
    }

    /*
     * Configuration data nodes
     */

    container lmap {
      description
        "Configuration of the LMAP agent.";

      /*
       * Common Information Objects: Configuration
       */

      container agent {
        description
          "Configuration of parameters affecting the whole
           measurement agent.";

        leaf agent-id {
```

```
        type yang:uuid;
        description
          "The agent-id identifies a measurement agent with
           a very low probability of collision. In certain
           deployments, the agent-id may be considered
           sensitive and hence this object is optional.";
      }

      leaf device-id {
        type inet:uri;
        description
          "The device-id identifies a property of the
           device running the measurement agent. In certain
           deployments, the device-id may be considered
           sensitive and hence this object is optional.";
      }

      leaf group-id {
        type string;
        description
          "The group-id identifies a group of measurement
           agents. In certain deployments, the group-id
           may be considered less sensitive than the
           agent-id.";
      }

      leaf report-agent-id {
        type boolean;
        default false;
        // XXX: write a must expression that requires
        // group-id to be configured when this is true?
        description
          "The 'report-agent-id' controls whether the
           'agent-id' is reported to collectors if the
           'group-id' is configured. If the 'group-id'
           is not configured, the agent-id is always
           reported.";
      }

      leaf controller-timeout {
        type uint32;
        units "seconds";
        description
          "A timer is started after each successful contact
           with a controller. When the timer reaches the
           controller-timeout, all schedules will be
           disabled, i.e., no new actions will be executed
           (and hence no new tasks started). The disabled
```

```
            schedules will be reenabled automatically once
            contact with a controller has been established
            successfully again. Note that this will not affect
            the execution of actions that are essential to
            establish contact with the controller or that
            perform critical housekeeping functions.";
        }
      }

    /*
     * Common Information Objects: Schedules
     */

    container schedules {
      description
        "Configuration of LMAP schedules. Schedules control with
         tasks are executed by the LMAP implementation.";

      list schedule {
        key name;
        description
          "Configuration of a particular schedule.";

        leaf name {
          type string;
          description
            "The locally-unique, administratively assigned name for
             this scheduled task.";
        }

        leaf event {
          type leafref {
            path "/lmap/events/event/name";
          }
          mandatory true;
          description
            "The event source controlling the start of the scheduled
             tasks.";
        }

        list action {
          key name;
          description
            "An action describes a task that is invoked by the
             schedule. Multiple actions are invoked sequentially.";

          leaf name {
            type string;
```

```
        description
          "The unique identifier for this action.";
      }

      leaf task {
        type leafref {
          path "/lmap/tasks/task/name";
        }
        mandatory true;
        description
          "The tasks invoked by this action.";
      }

      uses task-options-grouping {
        description
          "The list of action specific options that are
           appended to the list of task specific options.";
      }

      leaf-list destination {
        type leafref {
          path "/lmap/schedules/schedule/name";
        }
        description
          "A schedule of receiving the output produced by
           this action. A queue is internally used to pass
           results to another schedule. The behaviour of
           an action passing data to its own schedule is
           implementation specific.";
      }
    }

    leaf execution-mode {
      type enumeration {
        enum sequential {
          value 1;
          description
            "The actions of the schedule are executed
     sequentially.";
        }
        enum parallel {
          value 2;
          description
            "The actions of the schedule are executed
             concurrently";
        }
        enum pipelined {
          value 3;
```

```
            description
              "The actions of the schedule are executed in a
               pipelined mode. Output created by an action is
               passed as input to the subsequent action.";
          }
        }
        mandatory true;
        description
          "The execution mode of this schedule determins in
           which order the actions of the schedule are executed.";
      }
    }
  }

  /*
   * Suppression
   */

  container suppression {
    description
      "Suppression information to prevent schedules to start
       certain tasks.";

    leaf enabled {
      type boolean;
      default false;
      description
        "Setting 'enabled' to true will suppress all tasks for
         which suppress-by-default is true.";
    }

    leaf stop-running {
      type boolean;
      default false;
      description
        "Setting 'stop-running' to true will cause running
         tasks to be terminated if suppression is enabled
         (the 'enabled' leaf is set to true). Otherwise,
         running tasks will not be affected if suppression
         is enabled.";
    }

    leaf start {
      type yang:date-and-time;
      description
        "The date and time when supression starts to
         become effective. If not present, supression
         becomes effective immeditately when 'enabled'
```

```
            is set to true.";
        }

        leaf end {
          type yang:date-and-time;
          description
            "The date and time when supression stops to
             be effective. If not present, supression
             continues indefinite until 'enabled' is set
             to false.";
        }

        leaf-list task {
          type leafref {
            path "/lmap/tasks/task/name";
          }
          description
            "A specific task to suppress. If no tasks are
             listed, then all tasks will be suppressed.";
        }

        leaf-list schedule {
          type leafref {
            path "/lmap/schedules/schedule/name";
          }
          description
            "A specific schedule to suppress. If no schedules
             are listed, then all schedules will be suppressed.";
        }

      }

      /*
       * Common Information Objects: Task Configurations
       */

      container tasks {
        description
          "Configuration of LMAP tasks.";

        uses task-grouping;
      }

      /*
       * Common Information Objects: Event Information
       */

      container events {
```

```
        description
          "Configuration of LMAP events.

           Implementations may be forced to delay acting
           upon the occurance of events in the face of local
           constraints. An action triggered by an event
           therefore should not rely on the accuracy
           provided by the scheduler implementation.";

        list event {
          key name;
          description
            "The list of event sources configured on the
             LMAP agent.";

          leaf name {
            type string;
            description
              "The unique name of an event source.";
          }

          choice event-type {
            description
              "Different types of events are handled by
               different branches of this choice. Note that
               this choice can be extended via augmentations.";

            case periodic {
              container periodic {
                description
                  "A periodic timing object triggers periodically
                   according to a regular interval.";

                leaf interval {
                  type uint32;
                  units "milliseconds";
                  mandatory true;
                  description
                    "The number of milliseconds between two triggers
                     generated by this periodic timing object.

                     The execution system must not generate triggers
                     for periodic timing objects that have a interval
                     value of 0. A periodic timing object with an
                     interval of 0 milliseconds will therefore never
                     trigger.";
                }
                uses timing-start-end-grouping;
```

```
      }
    }

    case calendar {
      container calendar {
        description
          "A calendar timing object triggers based on the
           current calendar date and time.";

        leaf-list month {
          type union {
            type month;
            type wildcard;
          }
          description
            "A month at which this calendar timing will
             trigger. The wildcard means all months.";
        }

        leaf-list weekday {
          type union {
            type weekday;
            type wildcard;
          }
          description
            "A weekday at which this calendar timing will
             trigger. The wildcard means all weekdays.";
        }

        leaf-list day-of-month {
          type union {
            type int8 { range "-31..-1 | 1..31"; }
            type wildcard;
          }
          description
            "A day in the month at which this calendar
             timing will trigger. Negative numbers indicate
             days counted backwards from the end of the
             months. The wildcard means all days of a month.";
        }

        leaf-list hour {
          type union {
            type int8 { range "0..23"; }
            type wildcard;
          }
          description
            "An hour at which this calendar timing will
```

```
                    trigger. The wildcard means all hours of a day.";
                }

                leaf-list minute {
                  type union {
                    type int8 { range "0..59"; }
                    type wildcard;
                  }
                  description
                    "A minute at which this calendar timing will
                     trigger. The wildcard means all minutes of
                     an hour.";
                }

                leaf-list second {
                  type union {
                    type int8 { range "0..59"; }
                    type wildcard;
                  }
                  description
                    "A second at which this calendar timing will
                     trigger. The wildcard means all seconds of
                     a minute.";
                }

                leaf timezone-offset {
                  type timezone-offset;
                  description
                    "The timezone in which this calendar timing
                     object will be evaluated.";
                }
                uses timing-start-end-grouping;
              }
            }

            case one-off {
              leaf one-off-time {
                type yang:date-and-time;
                mandatory true;
                description
                  "This one-off timing object triggers once at the
                   configured one-off-time.";
              }
            }

            case immediate {
              leaf immediate {
                type empty;
```

```
                mandatory true;
                description
                  "This immediate event object triggers immediately
                   when it is configured.";
              }
            }

            case startup {
              leaf startup {
                type empty;
                mandatory true;
                description
                  "This startup event object triggers whenever the
                   LMAP agent (re)starts.";
              }
            }
          }

          leaf random-spread {
            type int32;
            units milliseconds;
            description
              "This optional leaf adds a random spread to the
               computation of the trigger.";
          }
        }
      }
    }

    /*
     * The state subtree provides information about the capabilities
     * and the current status of the MA.
     */

    container lmap-state {
      config false;
      description
        "A tree exporting state information about the LMAP agent.";

      container agent {
        description
          "Operations state of the measurement agent.";

        leaf agent-id {
          type yang:uuid;
          mandatory true;
          description
            "The agent-id identifies a measurement agent with
```

```
            a very low probability of collision. In certain
            deployments, the agent-id may be considered
            sensitive and hence this object is optional.";
        }

        leaf device-id {
          type inet:uri;
          mandatory true;
          description
            "The device-id identifies a property of the
             device running the measurement agent. In certain
             deployments, the device-id may be considered
             sensitive and hence this object is optional.";
        }
        leaf hardware {
          type string;
          mandatory true;
          description
            "A short description of the hardware the measurement
             agent is running on. This should include the version
             number of the hardware";
        }
        leaf firmware {
          type string;
          mandatory true;
          description
            "A short description of the firmware the measurement
             agent is running on. This should include the version
             number of the firmware.";
        }
        leaf version {
          type string;
          mandatory true;
          description
            "A short description of the software implementing the
             measurement agent. This should include the version
             number of the measurement agent software.";
        }
        leaf last-started {
          type yang:date-and-time;
          mandatory true;
          description
            "The date and time the measurement agent last started.";
        }
      }

      container tasks {
        description
```

```
      "Available LMAP tasks, including information about their
       last execution and their last failed execution.";

    list task {
      key name;
      description
        "The list of tasks available on the LMAP agent.";

      leaf name {
        type string;
        description
          "The unique name of a task.";
      }

      choice task-identification {
        mandatory true;
        description
          "Information that identifies the task.";

        leaf-list registry {
          type inet:uri;
          description
            "The registry entry identifying the configured task.";
        }

        leaf program {
          type string;
          description
            "The (local) program to invoke in order to execute
             the task.";
        }
      }

      leaf last-completion {
        type yang:date-and-time;
        description
          "The date and time of the last completion of this task.";
      }

      leaf last-status {
        type status-code;
        description
          "The status code returned by the last execution of
           this task.";
      }

      leaf last-message {
        type string;
```

```
            description
              "The status message produced by the last execution
               of this task.";
          }

          leaf last-failed-completion {
            type yang:date-and-time;
            description
              "The date and time of the last failed invocation
               of this task.";
          }

          leaf last-failed-status {
            type status-code;
            description
              "The status code returned by the last failed execution
               of this task. ";
          }

          leaf last-failed-message {
            type string;
            description
              "The status message produced by the last failed
               execution of this task.";
          }
        }
      }
    }
  }
  <CODE ENDS>



  <CODE BEGINS> file "ietf-lmap-report@2015-07-03.yang"
  module ietf-lmap-report {

    namespace "urn:ietf:params:xml:ns:yang:ietf-lmap-report";
    prefix "lmapr";

    import ietf-yang-types {
      prefix yang;
    }
    import ietf-lmap-control {
      prefix lmapc;
    }

    organization
      "IETF Large-Scale Measurement Platforms Working Group";
```

```
     contact
       "WG Web:   <http://tools.ietf.org/wg/lmap/>
        WG List:  <mailto:lmap@ietf.org>

        Editor:   Juergen Schoenwaelder
                  <j.schoenwaelder@jacobs-university.de>

        Editor:   Vaibhav Bajpai
                  <v.bajpai@jacobs-university.de>";

     description
       "This module defines a data model for reporting results from
        measurement agents that are part of a Large-Scale Measurement
        Platform (LMAP).";

     revision "2015-04-10" {
       description
        "Initial version";
       reference
        "RFC XXX: A YANG Data Model for LMAP Measurement Agents";
     }

     notification report {
       description
         "The result record produced by a certain task.";

       leaf date {
         type yang:date-and-time;
         mandatory true;
         description
           "The date and time when this report was sent.";
         // XXX It is unclear why this is useful. The receiver can
         // XXX easily timestamp when data was received.
       }

       leaf agent-id {
         type yang:uuid;
         description
           "The agent-id of the agent from which this
            report originates.";
       }

       leaf group-id {
         type string;
         description
           "The group-id of the agent from which this
            report originates.";
       }
```

```
      uses lmapc:task-grouping;
      // XXX We would prefer to just send a configuration version
      // XXX number such that the configuration can be identified
      // XXX that was active XXX when the report was generated. It
      // XXX would be nice to have a generic configuration version
      // XXX number that we could reuse. If this works out, we can
      // XXX inline the grouping as well.

      // XXX Note that task-grouping contains stuff that is not
      // XXX relevant here, perhaps using a grouping is not helpful
      // XXX here. It is also missing stuff, such as the options
      // XXX in the schedule entry.

      container header {
        description
          "The header of the result records.";

        leaf-list column {
          type string;
          description
            "A header of a column in the result rows.";
        }
      }

      list row {
        description
          "The rows of the result record.";

        leaf start {
          type yang:date-and-time;
          mandatory true;
          description
            "The date and time when the measurement producing
             this result row started.";
        }

        leaf end {
          type yang:date-and-time;
          description
            "The date and time when the measurement producing
             this result row stopped.";
        }

        leaf-list conflict {
          type string;
          description
            "The name of a task overlapping with the execution
             of the task that has produced this result record.";
```

```
        }

        leaf-list value {
          type string;
          description
            "The value of a cell in the result.";
          // XXX We could make this a union / choice in order to a
          // XXX more compact encoding of numbers in certain encodings
          // XXX (e.g. JSON)
        }
      }
    }
  }
<CODE ENDS>
```

5. Security Considerations

TBD

6. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registrations have been made.

```
      URI: urn:ietf:params:xml:ns:yang:ietf-lmap-control
      Registrant Contact: The IESG.
      XML: N/A; the requested URI is an XML namespace.

      URI: urn:ietf:params:xml:ns:yang:ietf-lmap-report
      Registrant Contact: The IESG.
      XML: N/A; the requested URI is an XML namespace.
```

This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

```
name: ietf-lmap-control
namespace: urn:ietf:params:xml:ns:yang:ietf-lmap-control
prefix: lmapc
reference: RFC XXXX

name: ietf-lmap-report
namespace: urn:ietf:params:xml:ns:yang:ietf-lmap-report
prefix: lmapr
reference: RFC XXXX
```

7. Acknowledgements

Juergen Schoenwaelder and Vaibhav Bajpai work in part on the Leone research project, which receives funding from the European Union Seventh Framework Programme [FP7/2007-2013] under grant agreement number 317647.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6991] Schoenwaelder, J., "Common YANG Data Types", RFC 6991, July 2013.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, May 2014.

8.2. Informative References

[I-D.ietf-lmap-framework] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A framework for Large-Scale Measurement of Broadband Performance (LMAP)", draft-ietf-lmap-framework-12 (work in progress), March 2015.

[I-D.ietf-lmap-information-model] Burbridge, T., Eardley, P., Bagnulo, M., and J. Schoenwaelder, "Information Model for Large-Scale Measurement Platforms (LMAP)", draft-ietf-lmap-information-model-06 (work in progress), July 2015.

[I-D.ietf-netconf-server-model]
Watsen, K. and J. Schoenwaelder, "NETCONF Server and RESTCONF Server Configuration Models", draft-ietf-netconf-server-model-06 (work in progress), February 2015.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

Appendix A. Open Issues

- The task-options-grouping needs fixing, see the comments inline.
- Should results be shipped via notifications or would it be more practical to have a collector provding a RESTCONF server and data is simply written via posts to suitable resources? This seems to be the way current platforms work. If we do this, we also need to think about how RESTCONF client credentials are configured.

Appendix B. Non-editorial Changes since -06

- A task can now reference multiply registry entries.
- Schedules are triggered by Events instead of Timings; Timings are just one of many possible event sources.
- Actions feed into other Schedules (instead of Actions within other Schedules).
- Removed the notion of multiple task outputs.
- Support for sequential, parallel, and pipelined execution of Actions.

Appendix C. Example Configuration (XML)

```
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <lmap xmlns="urn:ietf:params:xml:ns:yang:ietf-lmap-control">

    <agent>
      <agent-id>550e8400-e29b-41d4-a716-446655440000</agent-id>
      <device-id>urn:dev:mac:0024befffe804ff1</device-id>
      <group-id>wireless measurement at the north-pole</group-id>
      <report-agent-id>true</report-agent-id>
    </agent>

    <schedules>
```

```
      <schedule>
        <name>hourly-schedule</name>
        <event>hourly</event>
        <action>
          <name>icmp-latency-hourly</name>
          <task>icmp-latency-measurement</task>
          <destination>daily</destination>
        </action>
        <action>
          <name>udp-latency-weekdays-hourly</name>
          <task>udp-latency-measurement</task>
          <destination>daily</destination>
        </action>
    <execution-mode>sequential</execution-mode>
      </schedule>

      <schedule>
        <name>daily-schedule</name>
        <event>daily</event>
        <action>
          <name>report-daily</name>
          <task>lmap-reporting-task</task>
          <option>
            <name>channel</name>
            <value>http://collector.example.org/</value>
          </option>
        </action>
    <execution-mode>sequential</execution-mode>
      </schedule>

      <schedule>
        <name>immediate-schedule</name>
        <event>immediate</event>
        <action>
          <name>icmp-latency-immediate</name>
          <task>icmp-latency-measurement</task>
        </action>
        <action>
          <name>report-immediate</name>
          <task>lmap-reporting-task</task>
          <option>
            <name>channel</name>
            <value>http://collector.example.org/</value>
          </option>
        </action>
        <execution-mode>pipelined</execution-mode>
      </schedule>
    </schedules>
```

```
    <suppression>
      <enabled>true</enabled>
      <start>2015-09-02T14:06:11+02:00</start>
      <task>iperf-server</task>
      <schedule>hourly</schedule>
      <schedule>weekdays-hourly</schedule>
    </suppression>

    <tasks>
      <task>
        <name>udp-latency-measurement</name>
        <registry>urn:....</registry>
      </task>
      <task>
        <name>icmp-latency-measurement</name>
        <registry>urn:....</registry>
      </task>
      <task>
        <name>iperf-server</name>
        <program>/usr/bin/iperf</program>
        <option>
          <name>role</name>
          <value>server</value>
        </option>
        <suppress-by-default>false</suppress-by-default>
      </task>
      <task>
        <name>lmap-reporting-task</name>
        <program>/usr/bin/lmap-reporter</program>
      </task>
    </tasks>

    <events>
      <event>
        <name>hourly</name>
        <periodic>
          <interval>3600000</interval>
          <start>2014-09-01T17:44:00+02:00</start>
          <end>2015-09-30T00:00:00+02:00</end>
        </periodic>
      </event>
      <event>
        <name>daily</name>
        <calendar>
          <hour>4</hour>
        </calendar>
      </event>
      <event>
```

```
        <name>once-every-six-hours</name>
        <calendar>
          <hour>0</hour>
          <hour>6</hour>
          <hour>12</hour>
          <hour>18</hour>
          <minute>0</minute>
          <second>0</second>
          <end>2014-09-30T00:00:00+02:00</end>
        </calendar>
        <random-spread>21600000</random-spread>
      </event>
      <event>
        <name>immediate</name>
        <immediate/>
      </event>
      <event>
        <name>startup</name>
        <startup/>
        <random-spread>12345</random-spread>
      </event>
    </events>

  </lmap>

  <lmap-state xmlns="urn:ietf:params:xml:ns:yang:ietf-lmap-control">

    <agent>
      <agent-id>550e8400-e29b-41d4-a716-446655440000</agent-id>
      <device-id>urn:dev:mac:0024befffe804ff1</device-id>
      <hardware>ACME home router</hardware>
      <firmware>OpenWrt version 10.03.1</firmware>
      <version>Measurement Agent Daemon (MAD) 4.2</version>
      <last-started>2015-04-10T17:24:42+02:00</last-started>
    </agent>

    <tasks>
      <task>
        <name>udp-latency-measurement</name>
        <registry>urn:....</registry>
      </task>

      <task>
        <name>icmp-latency-measurement</name>
        <registry>urn:....</registry>
      </task>

      <task>
```

```
        <name>iperf</name>
        <program>iperf</program>
      </task>

      <task>
        <name>lmap-reporting-task</name>
        <program>lmap-reportd</program>
        <last-completion>2015-01-23T12:00:00+01:00</last-completion>
        <last-status>0</last-status>
        <last-message>OK</last-message>
        <last-failed-completion>2015-01-23T03:00:00+01:00</last-failed-
completion>
        <last-failed-status>42</last-failed-status>
        <last-failed-message>connection timed out</last-failed-message>
      </task>
    </tasks>

  </lmap-state>
</data>
```

Appendix D. Example Configuration (JSON)

```
   {
     "ietf-lmap-control:lmap": {
       "agent": {
         "agent-id": "550e8400-e29b-41d4-a716-446655440000",
         "device-id": "urn:dev:mac:0024befffe804ff1",
         "group-id": "wireless measurement at the north-pole",
         "report-agent-id": true
       },
       "schedules": {
         "schedule": [
           {
             "name": "hourly-schedule",
             "event": "hourly",
             "action": [
               {
                 "name": "icmp-latency-hourly",
                 "task": "icmp-latency-measurement",
                 "destination": [
                   "daily"
                 ]
               },
               {
                 "name": "udp-latency-weekdays-hourly",
                 "task": "udp-latency-measurement",
```

"destination": [

```
                "daily"
              ]
            }
          ],
          "execution-mode": "sequential"
        },
        {
          "name": "daily-schedule",
          "event": "daily",
          "action": [
            {
              "name": "report-daily",
              "task": "lmap-reporting-task",
              "option": [
                {
                  "name": "channel",
                  "value": "http://collector.example.org/"
                }
              ]
            }
          ],
          "execution-mode": "sequential"
        },
        {
          "name": "immediate-schedule",
          "event": "immediate",
          "action": [
            {
              "name": "icmp-latency-immediate",
              "task": "icmp-latency-measurement"
            },
            {
              "name": "report-immediate",
              "task": "lmap-reporting-task",
              "option": [
                {
                  "name": "channel",
                  "value": "http://collector.example.org/"
                }
              ]
            }
          ],
          "execution-mode": "pipelined"
        }
      ]
    },
    "suppression": {
      "enabled": true,
```

```
        "start": "2015-09-02T14:06:11+02:00",
        "task": [
          "iperf-server"
        ],
        "schedule": [
          "hourly",
          "weekdays-hourly"
        ]
      },
      "tasks": {
        "task": [
          {
            "name": "udp-latency-measurement",
            "registry": [
              "urn:...."
            ]
          },
          {
            "name": "icmp-latency-measurement",
            "registry": [
              "urn:...."
            ]
          },
          {
            "name": "iperf-server",
            "program": "/usr/bin/iperf",
            "option": [
              {
                "name": "role",
                "value": "server"
              }
            ],
            "suppress-by-default": false
          },
          {
            "name": "lmap-reporting-task",
            "program": "/usr/bin/lmap-reporter"
          }
        ]
      },
      "events": {
        "event": [
          {
            "name": "hourly",
            "periodic": {
              "interval": 3600000,
              "start": "2014-09-01T17:44:00+02:00",
              "end": "2015-09-30T00:00:00+02:00"
```

```
            }
          },
          {
            "name": "daily",
            "calendar": {
              "hour": [
                4
              ]
            }
          },
          {
            "name": "once-every-six-hours",
            "calendar": {
              "hour": [
                0,
                6,
                12,
                18
              ],
              "minute": [
                0
              ],
              "second": [
                0
              ],
              "end": "2014-09-30T00:00:00+02:00"
            },
            "random-spread": 21600000
          },
          {
            "name": "immediate",
            "immediate": [null]
          },
          {
            "name": "startup",
            "startup": [null],
            "random-spread": 12345
          }
        ]
      }
    },
    "ietf-lmap-control:lmap-state": {
      "agent": {
        "agent-id": "550e8400-e29b-41d4-a716-446655440000",
        "device-id": "urn:dev:mac:0024befffe804ff1",
        "hardware": "ACME home router",
        "firmware": "OpenWrt version 10.03.1",
        "version": "Measurement Agent Daemon (MAD) 4.2",
```

```
        "last-started": "2015-04-10T17:24:42+02:00"
      },
      "tasks": {
        "task": [
          {
            "name": "udp-latency-measurement",
            "registry": [
              "urn:...."
            ]
          },
          {
            "name": "icmp-latency-measurement",
            "registry": [
              "urn:...."
            ]
          },
          {
            "name": "iperf",
            "program": "iperf"
          },
          {
            "name": "lmap-reporting-task",
            "program": "lmap-reportd",
            "last-completion": "2015-01-23T12:00:00+01:00",
            "last-status": 0,
            "last-message": "OK",
            "last-failed-completion": "2015-01-23T03:00:00+01:00",
            "last-failed-status": 42,
            "last-failed-message": "connection timed out"
          }
        ]
      }
    }
  }
```


Authors' Addresses

Juergen Schoenwaelder
Jacobs University Bremen

Email: j.schoenwaelder@jacobs-university.de

Vaibhav Bajpai
Jacobs University Bremen

Email: v.bajpai@jacobs-university.de