LMAP Information Model Issues

Tim Carey (Nokia)

June 12, 2016

LMAP – Information Model Issues

Summary

- In review of draft-ietf-lmap-information-model-09, we realized that this draft:
  - Modified the ma-schedule-obj to include new attributes for schedule end and duration.

- These issues and other issues were noted on the mailing list and discussed during IETF 95. However these this issue remains outstanding.

- In addition – There seems to be a proliferation of events that can cause schedules to be invoked. This mechanism needs further discussion

Summary

- In the latest draft the ma-schedule-obj added 2 attributes: ma-schedule-end and ma-schedule-duration.

- Juergen stated in on the mailing list that these attributes were requested by Al Morton
  - In a discussion with Al yesterday his concern was that the schedule **occurrence** needed to allow for randomness

```
object {
    string              ma-schedule-name;
    ma-event-obj        ma-schedule-start;
   [ma-event-obj    ma-schedule-end;]
   [int                 ma-schedule-duration;]
    ma-action-obj       ma-schedule-actions<0..*>;
    string              ma-schedule-execution-mode;
   [string              ma-schedule-tags<0..*>;]
   [string              ma-schedule-suppression-tags<0..*>;]
} ma-schedule-obj;
```

Problem

- The ma-schedule-start and ma-schedule-end are already typed as events that allow for definition of the event **reoccurrence.**

- Meaning event type already has the start/end/duration/randomness

- **As such the attributes for the end and duration in the schedule are not needed.**

```
object {
   string                ma-event-name;
   union {
       ma-periodic-obj             ma-event-periodic;
       ma-calendar-obj             ma-event-calendar;
       ma-one-off-obj              ma-event-one-off;
       ma-immediate-obj            ma-event-immediate;
       ma-startup-obj              ma-event-startup;
       ma-immediate-obj            ma-event-immediate;
       ma-startup-obj              ma-event-startup;
       ma-controller-lost-obj      ma-event-controller-lost;
       ma-controller-connected-obj ma-event-controller-connected;
   }
   [int                  ma-event-random-spread;]
} ma-event-obj;
```

```
object {
   [datetime         ma-calendar-start;]
   [datetime         ma-calendar-end;]
   [string           ma-calendar-months<0..*>;]
   [string           ma-calendar-days-of-week<0..*>;]
   [string           ma-calendar-days-of-month<0..*>;]
   [string           ma-calendar-hours<0..*>;]
   [string           ma-calendar-minutes<0..*>;]
   [string           ma-calendar-seconds<0..*>;]
   [int              ma-calendar-timezone-offset;]
} ma-calendar-obj;
```

LMAP – Information Model Issues: ma-schedule-obj

Email - Discussion

Lets assume we have a schedule made up of 2 actions (A1, A2) in pipeline mode. The ma-schedule-start would has a periodic event:

> Start April 1, 2016 at 02:00:00

> End May 1, 2016 at 02:00:00

> Interval 86,400 (daily)

> Randomness 3600 (within the hour)

> So the schedule will reoccur every day starting on April 1 beginning at 2:00pm. (T0) The starting period will be random between 2:00am and 3:00am. (T1) This definition is sufficient for invoking A1 (T1) and is what Al considers as Blue times.

>

> As A1 performs a test A1 places bits on the wire (T2) and collects results (T3).

> When A1 completes (T4), A2 is invoked (T5) which places bits on a wire for A2 (T2) and collects results (T3)). Finally the results from the schedule are reported (T6).

> T2&T3: Al considers this time to be part of the definition of the

> metric and are reported as part of the result content (not as an

> actual parameter). This was his Red time

>

> T4&T5: Are the ma-report-[start|end]-time

> T6: Is the ma-report-date

>

> As such – As long as an Event has a defined occurrence (T1=T0 + randomness) all that is necessary is a single parameter to define the occurrence.

LMAP – Information Model Issues: ma-schedule-obj

Email - Discussion

Lets assume we have a schedule made up of 2 actions (A1, A2) in pipeline mode. The ma-schedule-start would has a periodic event:

> Start April 1, 2016 at 02:00:00

> End May 1, 2016 at 02:00:00

> Interval 86,400 (daily)

> Randomness 3600 (within the hour)

> So the schedule will reoccur every day starting on April 1 beginning at 2:00pm. (T0) The starting period will be random between 2:00am and 3:00am. (T1) This definition is sufficient for invoking A1 (T1) and is what Al considers as Blue times.

>

> As A1 performs a test A1 places bits on the wire (T2) and collects results (T3).

> When A1 completes (T4), A2 is invoked (T5) which places bits on a wire for A2 (T2) and collects results (T3)). Finally the results from the schedule are reported (T6).

> T2&T3: Al considers this time to be part of the definition of the

> metric and are reported as part of the result content (not as an

> actual parameter). This was his Red time

>

> T4&T5: Are the ma-report-[start|end]-time

> T6: Is the ma-report-date

>

> As such – As long as an Event has a defined occurrence (T1=T0 + randomness) all that is necessary is a single parameter to define the occurrence.

LMAP – Information Model Issues: ma-schedule-obj

Email – Discussion – Ron Stanza

> The exact definition of the metric is still unclear to me since I have not seen an example of a test with average complexity using metrics with LMAP from beginning to end. By average complexity I mean a test that allows independent configuration of multiple simultaneous data flows and produces multiple report formats during the test duration.

> The tests I mentioned previously (RFC-2544, Y.1564, Y.1731 and TWAMP) minimally require this type of support and are the main tests used in Ethernet service testing today. None-the-less, my understanding of

> the metric is that it defines the format of the packet stream and the

> methodology of the transmit frequency. These are two examples from

> https://tools.ietf.org/html/draft-ietf-ippm-initial-registry-00

>

> Act_IP_UDP_Poisson_UDP-Payload-250_One-way_Delay_Std_Dev

> Act_IP_UDP_Periodic-var_UDP-Payload-142_One-way_Delay_Mean

>

> It does not appear to define the time period over which the measurement was taken. This is good because for most tests the measurement period is a user defined input parameter. Performance monitoring tests will report the metric repeatedly during the test execution. Referring to the example in my prior email, a TWAMP test would produce a group of metrics for each data stream every 1, 5 or 15 minutes depending on the test configuration. It is possible the test could be configured to produce multiple reports - for example at a short interval for troubleshooting in addition to one of the intervals previously mentioned for general service monitoring. Thus, the TWAMP test produces instances of metrics for an unspecified amount of time - typically until a user manually stops the test.

>

> In summary, from my perspective, the duration of the test is outside the scope of the metric definition….but maybe I just don't understand the metric definition well enough.

LMAP – Information Model Issues: ma-schedule-obj

Email – Discussion – Ron Stanza

> The exact definition of the metric is still unclear to me since I have not seen an example of a test with average complexity using metrics with LMAP from beginning to end. By average complexity I mean a test that allows independent configuration of multiple simultaneous data flows and produces multiple report formats during the test duration.

> The tests I mentioned previously (RFC-2544, Y.1564, Y.1731 and TWAMP) minimally require this type of support and are the main tests used in Ethernet service testing today. None-the-less, my understanding of

> the metric is that it defines the format of the packet stream and the

> methodology of the transmit frequency. These are two examples from

> https://tools.ietf.org/html/draft-ietf-ippm-initial-registry-00

>

> Act_IP_UDP_Poisson_UDP-Payload-250_One-way_Delay_Std_Dev

> Act_IP_UDP_Periodic-var_UDP-Payload-142_One-way_Delay_Mean

>

> It does not appear to define the time period over which the measurement was taken. This is good because for most tests the measurement period is a user defined input parameter. Performance monitoring tests will report the metric repeatedly during the test execution. Referring to the example in my prior email, a TWAMP test would produce a group of metrics for each data stream every 1, 5 or 15 minutes depending on the test configuration. It is possible the test could be configured to produce multiple reports - for example at a short interval for troubleshooting in addition to one of the intervals previously mentioned for general service monitoring. Thus, the TWAMP test produces instances of metrics for an unspecified amount of time - typically until a user manually stops the test.

>

> In summary, from my perspective, the duration of the test is outside the scope of the metric definition….but maybe I just don't understand the metric definition well enough.

LMAP – Information Model Issues: ma-schedule-obj

Email – Discussion – Al Morton with Tims Reply

> The Registry Entries* for the Metrics include Run-time Parameters for the Start time and End Time, but one of them could be interpreted as a

> Duration if specified correctly (the Start time must be all zeros, then the Stop time is interpreted as a duration, AND the actual start

> and stop time are controlled through "other means"). See https://tools.ietf.org/html/draft-ietf-ippm-initial-registry-00#section-4.3.5

>

> "Other means" would be the Schedule object and the Event object. As I understand it both schedule and event objects have start time and stop time specifications available, and Event has Duration specification available which would be useful for recurring events (periodic or calendar).

>

> <TAC> The schedule's timer is sufficient to state when the initial action(s) are invoked. I say actions because the execution mode for the schedule's action's could be parallel. In the case when the schedule's execution mode is sequential then it would be sufficient to state when the first action is invoked.

> What the schedule's timer doesn't do is specify the duration of the individual action(s) or when the second action is executed in relation to the completion of the first action. What we would need to do is either: **Pass the duration of action based on the options (T0 = zero, Tf=some duration) or we could put a timer on the action – which would provide the external control that would allow the start to be better controlled in the context of the overall schedule.**

>

> Eventually, the MA has to report the results of the metric/measurement.

> That's when the actual (RED) start and end times when packets were actually flowing between measurement points are reported, according to the Registry Entry.

- See: https://tools.ietf.org/html/draft-ietf-ippm-initial-registry-00#section-4.4.2

Resolution

- Remove the 2 attributes: ma-schedule-end and ma-schedule-duration.

- Realize that the ma-schedule-start is really the definition of the schedule **occurrence**
  - We can rename the ma-schedule-start to ma-schedule-occurrence

```
object {
    string              ma-schedule-name;
    ma-event-obj        ma-schedule-start;
   [ma-event-obj        ma-schedule-end;]
   [int                 ma-schedule-duration;]
    ma-action-obj       ma-schedule-actions<0..*>;
    string              ma-schedule-execution-mode;
   [string              ma-schedule-tags<0..*>;]
   [string              ma-schedule-suppression-tags<0..*>;]
} ma-schedule-obj;
```

Summary

- The ma-event-obj has become a place where **occurrences** of events are defined.
- The intent is that these occurrences would trigger actions of schedules to be invoked.
- In some cases – periodic, calendar events actually contain a **reoccurrence** definition as part of the event itself
- As such the occurrence event has been overloaded with the reoccurence definition

```
object {
   string              ma-event-name;
   union {
       ma-periodic-obj             ma-event-periodic;
       ma-calendar-obj             ma-event-calendar;
       ma-one-off-obj              ma-event-one-off;
       ma-immediate-obj            ma-event-immediate;
       ma-startup-obj              ma-event-startup;
       ma-immediate-obj            ma-event-immediate;
       ma-startup-obj              ma-event-startup;
       ma-controller-lost-obj      ma-event-controller-lost;
       ma-controller-connected-obj ma-event-controller-connected;
   }
   [int                ma-event-random-spread;]
} ma-event-obj;
```

Resolution

- Create a separate obj (ma-schedule-reoccurrence and add the periodic, calendar, one-off, immediate and random-spread to that object.

- Assign the ma-schedule-reoccurrence to the ma-schedule-object's ma-schedule-occurrence attribute

- Rename the ma-schedule-**occurrence** attribution to ma-schedule-**reoccurrence**

- Add a new type of event: ma-event-schedule-occurrence and document it