A picture containing logo

Description automatically generated

Switch Abstraction Interface

Change Proposal

|  |  |
| --- | --- |
| **Title** | **SAI Periodic and Timeslice Telemetry** |
| **Authors** | **Jason Bos** |
| **Status** | **In Review** |
| **Type** | **Standards Track** |
| **Created** | **8 Nov 2022** |
| **SAI-Version** | **1.11** |

# Overview

This spec enhances the existing TAM (Telemetry and Monitoring) spec to address three new features: periodic telemetry exports, timeslice telemetry reports, and granular counter subscription.

# Periodic Telemetry

The TAM API provides a methodology to associate telemetry exports to events. This proposal extends that with a capability to push telemetry based on a configured interval rather than in response to an event.

To accomplish this, the event API is extended with a new SAI\_TAM\_EVENT\_TYPE\_PERIODIC event type. The threshold object may be used with an absolute value threshold in time units to define the collection interval:

sai\_attr\_list[0].id = SAI\_TAM\_EVENT\_THRESHOLD\_ATTR\_ABS\_VALUE;

sai\_attr\_list[0].value.u32 = 300;

sai\_attr\_list[1].id = SAI\_TAM\_EVENT\_THRESHOLD\_ATTR\_UNIT;

sai\_attr\_list[1].value.u32 = SAI\_TAM\_EVENT\_THRESHOLD\_UNIT\_NANOSEC;

attr\_count = 3;

**sai\_create\_tam\_event\_threshold\_fn**(

&sai\_tam\_threshold\_obj,

switch\_id,

attr\_count,

sai\_attr\_list);

sai\_attr\_list[0].id = SAI\_TAM\_EVENT\_ATTR\_TYPE;

sai\_attr\_list[0].value.s32 = SAI\_TAM\_EVENT\_TYPE\_PERIODIC;

sai\_attr\_list[1].id = SAI\_TAM\_EVENT\_ATTR\_THRESHOLD;

sai\_attr\_list[1].value.oid = sai\_tam\_threshold\_obj;

/\* use existing attributes for action & collector \*/

sai\_attr\_list[2].id = SAI\_TAM\_EVENT\_ATTR\_ACTION\_LIST;

sai\_attr\_list[2].value.objlist.count = 1;

sai\_attr\_list[2].value.objlist.list[0] = sai\_tam\_event\_action\_obj;

sai\_attr\_list[3].id = SAI\_TAM\_EVENT\_ATTR\_COLLECTOR\_LIST;

sai\_attr\_list[3].value.objlist.count = 1;

sai\_attr\_list[3].value.objlist.list[0] = sai\_tam\_collector\_obj;

attr\_count = 4;

**sai\_create\_tam\_event\_fn**(

&sai\_tam\_event2\_obj,

switch\_id,

attr\_count,

sai\_attr\_list);

Note: If the user specifies an interval shorter than the implementation is capable, it should generate reports as rapidly as possible.

## Timeslice Telemetry

The Timeslice telemetry format is an output report format for exporting one or more counters correlated with collection timestamps. The goal is to provide a compact format to observe multiple counters correlated closely correlated in time.

In implementations, the reports may be generated by the datapath, so the exact format of the packet will be vendor-defined. One possible format, to show the intent:

Header:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Metadata | Time-0 | Time-1 | Time-2 | Time-3 | Time-4 | Time-5 |

Body:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Counter ID 0 | Counter-0-at-Time-0 | Counter-0-at-Time-1 | Counter-0-at-Time-2 | Counter-0-at-Time-3 | Counter-0-at-Time-4 | Counter-0-at-Time-5 |
| Counter ID 1 | Counter-1-at-Time-0 | Counter-1-at-Time-1 | Counter-1-at-Time-2 | Counter-1-at-Time-3 | Counter-1-at-Time-4 | Counter-1-at-Time-5 |
| … |  |  |  |  |  |  |

To configure the format of the report, the TAM report API is extended with a new type and attribute to declare the number of samples per report, i.e. the width of the table.

sai\_attr\_list[0].id = SAI\_TAM\_REPORT\_ATTR\_TYPE;

sai\_attr\_list[0].value.u32 = SAI\_TAM\_REPORT\_TYPE\_TIMESLICE;

sai\_attr\_list[1].id = SAI\_TAM\_REPORT\_ATTR\_NUMBER\_OF\_TIMESLICES;

sai\_attr\_list[1].value.u32 = 6;

attr\_count = 2;

**sai\_create\_tam\_report\_fn**(

&sai\_tam\_report\_obj,

switch\_id,

attr\_count,

sai\_attr\_list);

An implementation should use atomic snapshots when possible. If not, the counters should be collected so the time interval between different snapshots is as consistent as possible across the selected counters. For example, by collecting in the same order across multiple samples.

## Granular counter subscription

The TAM API provides a set of predefined telemetry groups, like SAI\_TAM\_TELEMETRY\_TYPE\_SWITCH or SAI\_TAM\_TELEMETRY\_TYPE\_PORT, and extensibility to the API to define new fixed collections. However, there may be a need for the client to request specific counters rather than a full telemetry dump of the configured objects.

This could be used in combination with the timeslice format, especially with very rapid intervals, to restrict the amount of data pushed to a small number of specific counters.

This proposal extends the SAI API to introduce a stat list type to define a list of enums and associated object types.

typedef struct \_sai\_object\_stat\_id\_t

{

sai\_object\_type\_t object\_type;

sai\_stat\_id\_t stat\_enum;

} sai\_object\_stat\_id\_t;

typedef struct \_sai\_object\_stat\_list\_t

{

uint32\_t count;

sai\_object\_stat\_id\_t \*list;

} sai\_object\_stat\_list\_t;

The telemetry type object is defined to allow the use of this list as an attribute.

/\*\*

\* @brief List of stats to collect

\*

\* @type sai\_object\_stat\_list\_t

\* @flags CREATE\_AND\_SET

\* @default empty

\*/

SAI\_TAM\_TEL\_TYPE\_ATTR\_OBJECT\_STATS,

Configuration Example:

sai\_attr\_list[0].id = SAI\_TAM\_TEL\_TYPE\_ATTR\_TELEMETRY\_TYPE;

sai\_attr\_list[0].value.u32 = SAI\_TAM\_TEL\_TYPE\_ATTR\_OBJECT\_STAT

sai\_attr\_list[1].id = SAI\_TAM\_TEL\_TYPE\_ATTR\_OBJECT\_STATS;

sai\_attr\_list[1].value.object\_stat\_list.count = 1;

sai\_attr\_list[1].value.object\_stat\_list.list[0].object\_type = SAI\_OBJECT\_TYPE\_INGRESS\_PRIORITY\_GROUP;

sai\_attr\_list[1].value.object\_stat\_list.list[0].stat\_enum = SAI\_INGRESS\_PRIORITY\_GROUP\_STAT\_CURR\_OCCUPANCY\_BYTES;

sai\_attr\_list[2].id = SAI\_TAM\_TEL\_TYPE\_ATTR\_REPORT\_ID;

sai\_attr\_list[2].value.oid = sai\_tam\_report\_obj; /\* Report object created earlier and reused \*/

attr\_count = 3;

**sai\_create\_tam\_tel\_type\_fn**(

&sai\_tam\_flow\_tel\_type\_obj,

switch\_id,

attr\_count,

sai\_attr\_list);