

|  |
| --- |
|  |

Switch Abstraction Interface

Change Proposal

**Contents**

[1 Overview 1](#_Toc43567199)

[2 Specification 2](#_Toc43567200)

[2.1 Changes to saimirror.h 2](#_Toc43567201)

[2.2 Changes to saisamplepacket.h 2](#_Toc43567202)

[2.3 Changes to saiport.h 3](#_Toc43567203)

[3 Examples 3](#_Toc43567204)

[3.1 SFlow tunel encapsulation using mirroring 3](#_Toc43567205)

[3.2 Configure sampling rate on a mirror session 4](#_Toc43567206)

[3.3 Configure sampling rate on a port 4](#_Toc43567207)

License

© 2014 Microsoft Corporation, Dell Inc., Facebook, Inc, Broadcom Corporation, Intel Corporation, Mellanox Technologies Ltd.

As of September 9, 2014, the following persons or entities have made this Specification available under the Open Web Foundation Final Specification Agreement (OWFa 1.0), which is available at <http://www.openwebfoundation.org/legal/the-owf-1-0-agreements/owfa-1-0>

Microsoft Corporation, Dell Inc., Facebook, Inc, Intel Corporation, Mellanox Technologies Ltd.

You can review the signed copies of the Open Web Foundation Agreement Version 1.0 for this Specification at <http://opencompute.org/licensing/>, which may also include additional parties to those listed above.

Your use of this Specification may be subject to other third party rights. THIS SPECIFICATION IS PROVIDED "AS IS." The contributors expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, noninfringement, fitness for a particular purpose, or title, related to the Specification. The entire risk as to implementing or otherwise using the Specification is assumed by the Specification implementer and user. IN NO EVENT WILL ANY PARTY BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE FOLLOWING IS A LIST OF MERELY REFERENCED TECHNOLOGY: Microprocessor technology, semiconductor manufacturing technology, operating system technology (including without limitation networking operating system technology), emulation technology, graphics technology, video technology, integrated circuit packaging technology and the like, compiler technologies, object oriented technology, optical/RF communications technology including chip I/O and driver technology, bus technology, memory chip technology (including, without limitation, NAND memory, NOR memory, resistive RAM (RRAM), seek scan probe (SSP) memory, nonvolatile memory (including without limitation, memory based on chalcogenide materials, phase change memory (PCM), one or more stacked layers of memory cells, embedded PCM memories, non-volatile cache memory, solid state drives, SRAM, embedded DRAM, ferro-electric memory, and polymer memory)) and/or health-related and medical technology. IMPLEMENTATION OF THESE TECHNOLOGIES MAY BE SUBJECT TO THEIR OWN LEGAL TERMS.

# Overview

In the current SAI model, packets can either be sampled to CPU using samplepacket or can be mirrored to a remote destination using ERSPAN encapsulation. This proposal is to sample packets and mirror to a remote destination using sflow encapsulation.

**SFLOW Encapsulation:**

This can be achieved in two ways:

1. Using only mirror
2. Using mirror and samplepacket

**1. SFlow tunel encapsulation using mirroring:**  
This is very similar to l3 gre tunnel encapsulation for mirrored packets. Following is the sequence to create a mirror and sample it into the SFLOW tunnel.

* Mirror session type is SAI\_MIRROR\_SESSION\_TYPE\_SFLOW.
* Set UDP source port using SAI\_MIRROR\_SESSION\_ATTR\_UDP\_SRC\_PORT
* Set UDP dst port using SAI\_MIRROR\_SESSION\_ATTR\_UDP\_DST\_PORT
* Set Sampling rate using SAI\_MIRROR\_SESSION\_ATTR\_SAMPLE\_RATE

Other mirror parameters like Src IP, Dst IP, TOS and TTL can be used to formulate the IP Header.

**2. Sampling using mirror and samplepacket:**  
The sampling rate can be configured independently of the mirroring session which provides more flexibility. For instance, a mirroring session can be configured to create a SFLOW tunnel. Every port can have a different sample packet object with different sampling rate but can share the same mirroring session. A new sample\_packet\_type\_t enum has been added to the samplepacket.h to select a mirror as a sampling destination. All the mirror session object will be selected when the sample object is attached to a SAI port object via SAI\_PORT\_ATTR\_INGRESS\_SAMPLEPACKET\_ENABLE

or SAI\_PORT\_ATTR\_EGRESS\_SAMPLEPACKET\_ENABLE port attributes.

# Specification

## Changes to saimirror.h

typedef enum \_sai\_mirror\_sessio\_type\_t {

. . .

/\*\*

\* **@**briefSFLOW Encapsulation | L2 Ethernet header | IP header | UDP header | Original mirrored packet

\*/

SAI\_ERSPAN\_ENCAPSULATION\_TYPE\_MIRROR\_SFLOW\_TUNNEL,

} sai\_mirror\_session\_type\_t;

typedef enum \_sai\_mirror\_session\_attr\_t {

. . .

/\*\*

\* @brief UDP source port. UDP source port to encapsulate

\* SFLOW packets

\*

\* @type sai\_uint16\_t

\* @flags MANDATORY\_ON\_CREATE | CREATE\_AND\_SET

\* @isvlan false

\* @condition SAI\_MIRROR\_SESSION\_ATTR\_TYPE ==

SAI\_MIRROR\_SESSION\_TYPE\_SFLOW

\*/

SAI\_MIRROR\_SESSION\_ATTR\_UDP\_SRC\_PORT,

/\*\*

\* @brief UDP destination port. UDP destination port to

\* encapsulate SFLOW packets

\*

\* @type sai\_uint16\_t

\* @flags MANDATORY\_ON\_CREATE | CREATE\_AND\_SET

\* @isvlan false

\* @condition SAI\_MIRROR\_SESSION\_ATTR\_TYPE ==

SAI\_MIRROR\_SESSION\_TYPE\_SFLOW

\*/

SAI\_MIRROR\_SESSION\_ATTR\_UDP\_DST\_PORT,

} sai\_mirror\_session\_attr\_t;

## Changes to saisamplepacket.h

typedef enum \_sai\_samplepacket\_type\_t

{

. . .

/\*\* Copy the sample packets using mirror session \*/

SAI\_SAMPLEPACKET\_TYPE\_MIRROR\_SESSION,

} sai\_samplepacket\_type\_t;

## Changes to saiport.h

typedef enum \_sai\_port\_attr\_t {

/\*\*

\* @brief Enable/Disable Samplepacket session

\*

\* Enable sample ingress mirroring by assigning list of mirror object ids

\* Disable sample ingress mirroring by assigning object\_count as 0 in

\* object list

\*

\* @type sai\_object\_list\_t

\* @flags CREATE\_AND\_SET

\* @objects SAI\_OBJECT\_TYPE\_MIRROR\_SESSION

\* @allownull true

\* @default SAI\_NULL\_OBJECT\_ID

\*/

SAI\_PORT\_ATTR\_INGRESS\_SAMPLE\_MIRROR\_SESSION,

/\*\*

\* @brief Enable/Disable Samplepacket session

\*

\* Enable sample egress mirroring by assigning list of mirror object ids

\* Disable sample egress mirroring by assigning object\_count as 0 in

\* object list

\*

\* @type sai\_object\_list\_t

\* @flags CREATE\_AND\_SET

\* @objects SAI\_OBJECT\_TYPE\_MIRROR\_SESSION

\* @allownull true

\* @default SAI\_NULL\_OBJECT\_ID

\*/

SAI\_PORT\_ATTR\_EGRESS\_SAMPLE\_MIRROR\_SESSION,

} sai\_port\_attr\_t;

# Examples

## SFlow tunel encapsulation using mirroring

sai\_api\_query(SAI\_API\_MIRROR, &sai\_mirror\_api);

sai\_object\_id\_t mirror\_erspan\_object;

sai\_attribute\_t attr[15] = {0};

attr[0].id = SAI\_MIRROR\_SESSION\_ATTR\_TC;

attr[0].value.u8 = 7;

attr[1].id = SAI\_MIRROR\_SESSION\_ATTR\_MONITOR\_PORT;

attr[1].value.oid = 9; //Embed in sai\_object\_id port numbering format

attr[2].id = SAI\_MIRROR\_SESSION\_ATTR\_TYPE;

attr[2].value.s32 = SAI\_MIRROR\_TYPE\_SFLOW;

attr[3].id =SAI\_MIRROR\_SESSION\_ATTR\_IPHDR\_VERSION;

attr[3].value.u8 = 0x4;

attr[4].id =SAI\_MIRROR\_SESSION\_ATTR\_TOS;

attr[4].value.u16 = 0x2;

attr[5].id =SAI\_MIRROR\_SESSION\_ATTR\_TTL;

attr[5].value.u8 = 0x2;

attr[6].id =SAI\_MIRROR\_SESSION\_ATTR\_SRC\_IP\_ADDRESS;

attr[6].value.ipaddr.addr.ip4 = 0xa0101002;

attr[6].value.ipaddr.addr\_family = SAI\_IP\_ADDR\_FAMILY\_IPV4;

attr[7].id =SAI\_MIRROR\_SESSION\_ATTR\_DST\_IP\_ADDRESS;

attr[7].value.ipaddr.addr.ip4 = 0xa0101011;

attr[7].value.ipaddr.addr\_family = SAI\_IP\_ADDR\_FAMILY\_IPV4;

attr[8].id =SAI\_MIRROR\_SESSION\_ATTR\_SRC\_MAC\_ADDRESS;

attr[8].value.mac[0] = 0x00;

attr[8].value.mac[1] = 0x01;

attr[8].value.mac[2] = 0x02;

attr[8].value.mac[3] = 0x03;

attr[8].value.mac[4] = 0x04;

attr[8].value.mac[5] = 0x05;

attr[9].id =SAI\_MIRROR\_SESSION\_ATTR\_DST\_MAC\_ADDRESS;

attr[9].value.mac[0] = 0x00;

attr[9].value.mac[1] = 0x11;

attr[9].value.mac[2] = 0x12;

attr[9].value.mac[3] = 0x13;

attr[9].value.mac[4] = 0x14;

attr[9].value.mac[5] = 0x15;

attr[10].id =SAI\_MIRROR\_SESSION\_ATTR\_UDP\_SRC\_PORT;

attr[10].value.u16 = 0xb0;

attr[11].id =SAI\_MIRROR\_SESSION\_ATTR\_UDP\_DST\_PORT;

attr[11].value.u16 = 0xa0;

sai\_mirror\_api->create\_mirror\_session (&mirror\_erspan\_object, 14, attr);

## Configure sampling rate on a mirror session

Configure a mirroring session using 3.1 (or refer to samplepacket object spec).

attr[14].id = SAI\_MIRROR\_SESSION\_ATTR\_SAMPLE\_RATE;

attr[14].value = 0.5;

sai\_mirror\_api->create\_mirror\_session (&mirror\_erspan\_object, 15, attr);

## Configure sampling rate on a port

Configure different type of mirroring sessions (SFLOW, ERSPAN, Local and RSPAN) and associate to a port using SAI\_PORT\_ATTR\_INGRESS\_SAMPLE\_MIRROR\_SESSION (or SAI\_PORT\_ATTR\_EGRESS\_SAMPLE\_MIRROR\_SESSION

//Create a samplepacket object

sai\_api\_query(SAI\_API\_SAMPLEPACKET, &sai\_samplepacket\_api);

sai\_object\_id\_t samplepacket\_object;

sai\_attribute\_t attr[2] = {0};

attr[0].id = SAI\_SAMPLEPACKET\_ATTR\_SAMPLE\_RATE;

attr[0].value.u32 = 0.5;

attr[1].id = SAI\_SAMPLEPACKET\_ATTR\_TYPE;

attr[1].value.s32 = SAI\_SAMPLEPACKET\_TYPE\_MIRROR\_SESSION;

sai\_samplepacket\_api->create\_samplepacket(&samplepacket\_object, 2, attr);

//Associate with port object

sai\_attribute\_t attr;

attr.id = SAI\_PORT\_ATTR\_INGRESS\_SAMPLEPACKET\_ENABLE;

attr.value.oid = samplepacket\_object;

sai\_port\_api->set\_port\_attribute(port\_id, &attr);