

|  |
| --- |
|  |

Switch Abstraction Interface

Change Proposal

|  |  |
| --- | --- |
| **Title** | **New Changes for 802.1X** |
| **Authors** | **DELL** |
| **Status** | **In Review** |
| **Type** | **Standards Track** |
| **Created** | **03/04** |
| **SAI-Version** | **V0.9.2** |

**Contents**

[List of Changes i](#_Toc413245278)

[1 Overview 1](#_Toc413245279)

[2 Specification 1](#_Toc413245280)

[2.1 To add Authenticated FDB entry: 1](#_Toc413245281)

[2.2 To remove Authenticated FDB entry: 1](#_Toc413245282)

[2.3 To remove all Authenticated FDB entries on a port: 1](#_Toc413245283)

[2.4 To authenticate unknown MAC by trapping them to CPU 1](#_Toc413245284)

[2.5 To Prevent MAC station moves to 802.1x enabled ports: 2](#_Toc413245285)

[3 Appendix 2](#_Toc413245286)

[3.1 Enable/Disable 802.1x functionality in a port 2](#_Toc413245287)

[3.2 Install authenticated entries 3](#_Toc413245288)

[3.3 Remove authenticated entries 4](#_Toc413245289)

# List of Changes

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Changes | Name | Date |
| 0.9.2 | Proposal for 802.1x |  | 03/04/15 |

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

This document describes the changes that is required in SAI to support 802.1x

# Specification

Using existing FDB and port modules and adding few attributes can help to provide the support for 802.1X.

Below are the requirements.

## To add Authenticated FDB entry:

Existing sai\_create\_fdb\_entry\_fn can be used with **SAI\_FDB\_ENTRY\_STATIC**. This is to add authenticated MAC addresses.

## To remove Authenticated FDB entry:

Existing sai\_remove\_fdb\_entry\_fn API can be used.

## To remove all Authenticated FDB entries on a port:

Modify sai\_flush\_all\_fdb\_entries\_by\_port\_fn as below. This is to remove all the authenticated entries. Currently static entries would not be flushed by this API. To flush them add another parameter.

/\*

\* Routine Description:

\* Remove all FDB entries by port

\*

\* Arguments:

\* [in] port\_id - port id

\* [in] remove\_static – Specifies whether static entries should be deleted too.

\*

\* Return Values:

\* SAI\_STATUS\_SUCCESS on success

\* Failure status code on error

\*/

typedef sai\_status\_t (\*sai\_flush\_all\_fdb\_entries\_by\_port\_fn)(

\_In\_ sai\_port\_id\_t port\_id,

**\_In\_ bool flush\_static\_entries,**

);

## To authenticate unknown MAC by trapping them to CPU

To trap unknown MAC addresses to CPU that are received on a 802.1x enabled port, a new learning mode should be introduced apart from existing two learning modes in sai\_port\_fdb\_learning\_mode\_t.

This new learning mode would specify to trap unknown MAC address to CPU so that they can be authenticated.

typedef enum \_sai\_port\_fdb\_learning\_mode\_t

{

/\* Learning diaable. Do not learn new mac address \*/

SAI\_PORT\_LEARN\_MODE\_DISABLE,

/\* Hardware learning \*/

SAI\_PORT\_LEARN\_MODE\_HW,

**/\* CPU learning \*/**

**SAI\_PORT\_LEARN\_MODE\_CPU,**

} sai\_port\_fdb\_learning\_mode\_t;

## To Prevent MAC station moves to 802.1x enabled ports:

An intrusion can be made when a MAC address get learnt in a non-authenticated port and does station move so that it does not appear as an unknown MAC address. To prevent this two possible solutions can be done

1. Prevent MAC move to the authorized port
2. Restrict port move by having a new port attribute.

Add the below attribute to sai\_port\_attr\_t. The existing sai\_port\_fdb\_learning\_mode\_t values can be used which would specify how MAC move can be handled – Either disabled, learn through HW or learn through CPU.

/\* FDB Learning mode [sai\_port\_fdb\_learning\_mode\_t]

(default to SAI\_PORT\_LEARN\_MODE\_HW) \*/

SAI\_PORT\_ATTR\_FDB\_LEARNING,

**/\* FDB Moving mode [sai\_port\_fdb\_learning\_mode\_t]\*/**

**SAI\_PORT\_ATTR\_FDB\_MOVE,**

…

} sai\_port\_attr\_t

# Appendix

This section is used to describe how to use the above defined SAI specifications to achieve 802.1x functionality.

## Enable/Disable 802.1x functionality in a port

To enable 802.1x functionality in a port, the following calls should be made

*sai\_set\_port\_attribute\_fn*

*ID= SAI\_PORT\_ATTR\_FDB\_LEARNING value = SAI\_PORT\_LEARN\_MODE\_CPU*

*ID = SAI\_PORT\_ATTR\_FDB\_MOVE value = SAI\_PORT\_LEARN\_MODE\_CPU*

This would trap the unknown unicast packets to CPU. It is better to clear the port of any existing MAC addresses. This can be done by normal delete call or with the delete port API with true set for static. It is important to note that if deleting static entries is not desired, then flush\_static\_entries can be set to false.

*sai\_flush\_all\_fdb\_entries\_by\_port\_fn(port\_id,true)*

To disable 802.1x functionality and move back to normal hardware learning the following changes can be made

*sai\_set\_port\_attribute\_fn*

*ID= SAI\_PORT\_ATTR\_FDB\_LEARNING value = SAI\_PORT\_LEARN\_MODE\_HW*

*ID = SAI\_PORT\_ATTR\_FDB\_MOVE value = SAI\_PORT\_LEARN\_MODE\_HW*

## Install authenticated entries

Using step 1 the authenticated entries can be taken to CPU and sent for authentication. Once authenticated is done the installation of authenticated FDB entri can be can be done as follows using the below API

*sai\_create\_fdb\_entry\_fn*

With following parameters

*fdb\_entry – The authenticated FDB entry.*

The following attributes are set

*port – 802.1x enabled port where the FDB entry was identified*

*type - SAI\_FDB\_ENTRY\_STATIC*

*action - SAI\_PACKET\_ACTION\_FORWARD*

## Remove authenticated entries

The authenticated entries can be removed one at a time by using the following API

*sai\_remove\_fdb\_entry\_fn*

*Parameters – FDB Entry.*

All the authenticated entries can be removed at one shot using the following API

*sai\_flush\_all\_fdb\_entries\_by\_port\_fn(port\_id,true)*

where port\_id is the 802.1x enabled port which is desired to be cleared of authenticated entries.