## **SAI IPSEC API Proposal**

## **High Level Design Document**

# Intel IPSEC SAI proposal in addition to Arista PR #1206

# **Scope**

This document defines the technical specifications for the API used to support RFC-4303 IPsec ESP tunnel mode and RFC-3948 UDP-encapsulated IPsec ESP tunnel mode in Open Compute Project Switch Abstraction Interface (SAI). This API only supports the IPsec data path security using GCM-AES ciphersuite. The key exchange protocol is outside the scope of this document and SAI IPsec API.

# **Overview**

IPsec secures packets between a pair of end-points identified by a pair of IP addresses. The switch becomes such an end-point in case it originates and terminates tunnels with another end-point. So, IPsec in a switch is always associated with tunnel origination and termination. IPSEC inline offload in a host enables crypto encryption and decryption operations to the SmartNIC hardware.

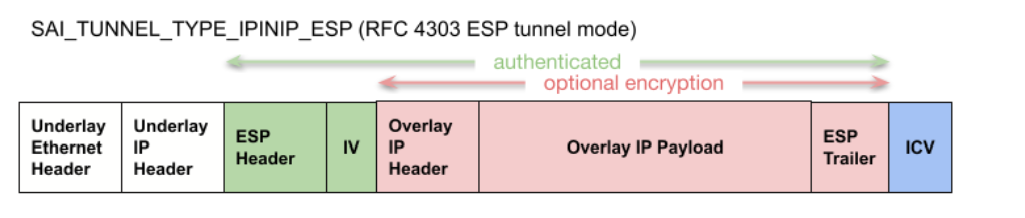
**Update to Initial proposal**

Issue: Interoperability between Arista Switch and other NICs that will follow the standard. This issue is directional. If the packets are coming from the Switch towards the NIC, NIC will have no issues decrypting or parsing but if the packets are being sent by the NIC towards the Switch they will not have the necessary hint, in which case the Switch will not be able to parse the packets after decryption in order to terminate the VxLAN tunnel on the switch.

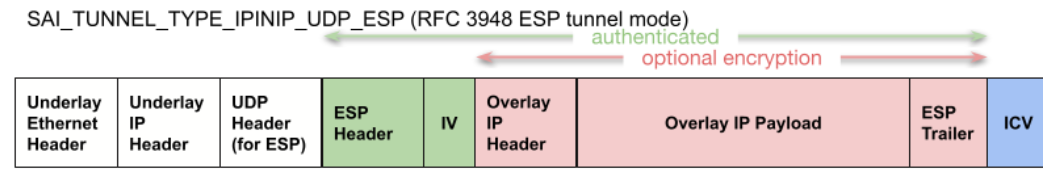
 In terms of APIs, this causes no changes.

**Tunnel Cases**

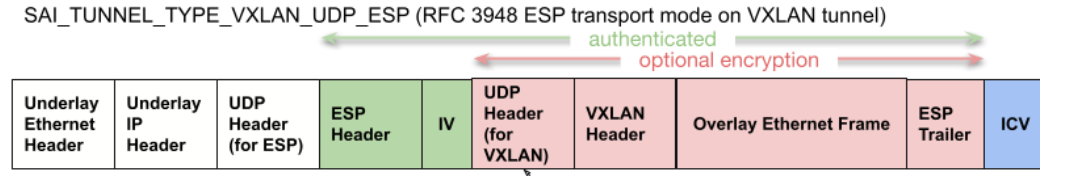
**IPinIP - no NAT traversal**



**ESP packets inside UDP packets for traversing Network Address Translators (NATs) (see**[**[RFC3715]**](https://tools.ietf.org/html/rfc3715#section-2.2)



**VXLAN packet inside of UDP ESP packets for nat traversing**



 SmartNIC can extract next header post decryption from the ESP Trailer and then terminate UDP + Vxlan tunnel. So there is no need to rely on Source Port first Nibble in the Overlay UDP header to convey the presence of Overlay UDP-VxLAN tunnel or Overlay IP. And use the Next-Header field in the ESP trailer to identify this information.

The outer UDP header will have Destination port indicating the presence of ESP header.

The source port will be used for Entropy (typically copies the hash over internal Layer3-Layer-4 fields and is reflected in the source port.

No passing of hint in the Overlay UDP Source port for inner parsing will be provided by the SNIC.

## **Counters**

## Need to expose Global Counter type

* + SAI\_IPSEC\_ATTR\_STATS\_MODE (Read only or Read & clear) - Clarified
* We maintain two level of counters. One at global level or aggregated for ingress and egress and one for each SA. We need a way of differentiating between the two.
* Recommend to add an attribute for supporting both LOCAL and GLOBAL counters.

SAI\_IPSEC\_ATTR\_STATS\_TYPE\_SA , SAI\_IPSEC\_ATTR\_STATS\_TYPE\_GLOBAL

GLOBAL counters support needs an extra attribute in addition to those defined for the per SA stats in sai\_ipsec\_sa\_stat\_

SAI\_IPSEC\_SA\_STAT\_INVALID\_SA

## **Events for soft/hard age limit**

Notifications to control plane can happen because of various reasons like soft ageing limit, hard ageing limit, Sequence number rollover, SA Domain invalid (Intel) for each direction of traffic.

* + sai\_create\_ipsec\_event\_fn create\_ipsec\_event;

|  |
| --- |
| typedef sai\_status\_t (\*sai\_create\_event\_fn)( |
|  | \_Out\_ sai\_object\_id\_t \*ipsec\_event\_id, |
|  | \_In\_ sai\_object\_id\_t switch\_id, |
|  | \_In\_ uint32\_t attr\_count, |
|  | \_In\_ const sai\_attribute\_t \*attr\_list); |

## **SA Domain programming:**

* 1. SA Domain is specified in the IPSEC SA entry. The data plane packets accessing the SA entry will also have domain ID. The SA domain check ensures that only the permitted data packets that originate from the same domain ID (tenant network) can process the packets. Packets which do not match are dropped.
  2. Our understanding is there can be multiple SA domains, one per Control plane or one per Tenant.
  3. The SA domain check is only enforced on Egress. It is not implemented in ingress since the Function or domain has yet to be determined in the lower portion of the packet pipeline
  4. In the Ingress Direction NIC uses outer header fields to identify the SA Domain.
  5. Add Domain id to SA creation SA

ipsec\_attr\_list[7].id = SAI\_IPSEC\_SA\_ATTR\_IPSEC\_DOMAIN;

ipsec\_attr\_list[7].value.domain = ipsec\_domain\_1;