JUNIPER NETWORKS | Engineering Simplicity

DAY ONE: MACsec UP AND RUNNING

By Chris Hellberg & Pradeep Chalicheemala

DAY ONE: MACsec UP AND RUNNING

While MACsec is a relatively new technology, Juniper has invested in it heavily by integrating MACsec encryption in its core ASICs and making the technology available on a wide variety of MX, PTX, QFX and EX Series platforms. With *Day One: MACsec Up and Running*, you can discover which platforms and hardware combinations support different types of MACsec and you'll be able to get up and running within a matter of hours in a basic lab setting. Learn how to configure MACsec on Juniper equipment, verify that it is working, and troubleshoot existing configurations. The technology is relatively simple so little prior knowledge is needed to strengthen and secure your network environment.

"This book is very timely because Juniper has fully committed to MACsec technology to enable the confidentiality and integrity of our customers' data. We have now implemented full line-rate MACsec encryption engines on most of our home grown ASICs and MACsec will soon be available on a majority of our products."

Santhosh Somasekharan Nair, Senior Engineer, MACsec Software Lead, Juniper Networks

IT'S DAY ONE AND YOU HAVE A JOB TO DO, SO LEARN HOW TO:

- Understand at a very high level where MACsec is used and how MACsec is different from other security protocols.
- Understand which Juniper Networks devices support MACsec and select the Junos OS software version to suit your MACsec needs.
- Configure MACsec on an interface between two directly connected Juniper Networks devices that support MACsec.
- Learn about Static PSK mode and PSK Keychains.
- Avoid common pitfalls when interoperating MACsec between devices from different vendors.
- Configure MACsec connectivity over a WAN and potential issues.
- Monitor the statistics of MACsec packets and sessions.

ISBN 978-1-941441-84-8
51600
9 781941 441848

Day One: MACsec Up and Running

by Chris Hellberg and Pradeep Chalicheemala

JUNIPER NETWORKS

Published by Juniper Networks Books
Author: Chris Hellberg & Pradeep Chalicheemala
Technical Reviewers: Ryan Barnes, Muthu Balasubramanian, Ben Holler, Santhosh Somasekharan Nair, Dmitry Shokarev
Editor in Chief: Patrick Ames
Copyeditor: Nancy Koerbel

ISBN: 978-1-941441-84-8 (print)
Printed in the USA by Vervante Corporation.

ISBN: 978-1-941441-83-1 (ebook)

Version History: v1, Febraury 2019
2 3 4 5 6 7 8 9 10

http://www.juniper.net/dayone

About the Authors
Chris Hellberg is a Systems Engineer at Juniper Networks focusing on financial services customers. He was previously a Professional Services consultant in Europe, Asia-Pacific and the United States. He holds an MBA and a Masters in Advanced Finance.

Pradeep Chalicheemala is a Product Manager for Cloud Customer Segment at Juniper Networks. Before his current role, Pradeep held ASIC design engineering roles at Juniper and Brocade. He holds an MBA degree from UC Berkeley. Pradeep also holds Masters and Bachelors degrees in Engineering.

Authors' Acknowledgments
CH: Thanks to Patrick and reviewers for their help. I dedicate my scribblings to my wife, Cody.
PC: I thank my Product, Sales and Engineering colleagues who helped us finish this book in a relatively short span. Special thanks to Patrick and the Juniper publishing team. To my wife Gowthami, I dedicate my work.

Feedback? Comments? Error reports? Email them to dayone@juniper.net.

Welcome to Day One

This book is part of the *Day One* library, produced and published by Juniper Networks Books.

Day One books cover the Junos OS and Juniper Networks networking essentials with straightforward explanations, step-by-step instructions, and practical examples that are easy to follow. You can obtain the books from various sources:

- Download a free PDF edition at http://www.juniper.net/dayone.
- Many of the library's books are available on the Juniper app: Junos Genius.
- Get the ebook edition for iPhones and iPads from the iBooks Store. Search for *Juniper Networks Books* or the title of this book.
- Get the ebook edition for any device that runs the Kindle app (Android, Kindle, iPad, PC, or Mac) by opening your device's Kindle app and going to the Amazon Kindle Store. Search for *Juniper Networks Books* or the title of this book.
- Purchase the paper edition at Vervante Corporation (www.vervante.com) for between $15-$40, depending on page length.
- Note that most mobile devices can also view PDF files.

Key MACsec Resources

MACsec is still a relatively new technology but the Juniper TechLibrary has MACsec information for Junos and for individual hardware platforms. This book is not a substitute for that body of work, so you should take the time to review the documentation: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/understanding_media_access_control_security_qfx_ex.html.

What You Need to Know Before Reading This Book

The authors have made a few assumptions about the general knowledge level of the reader:

- You need to be conversant with Junos OS CLI and configuration.
- You will need access to two Juniper Networks devices that support MACsec and be able to install a Junos OS Software version that supports MACsec features being discussed in this book. (See Chapter 1 for details.)

The *Day One* library has several books on Junos for beginners, for intermediate network administrators, and for advanced-level engineers and architects. See https://www.juniper.net/dayone.

What You Will Learn by Reading This Book

This book will help you to:

- Understand, at a very high level, where MACsec is used and how MACsec is different from other security protocols.
- Understand which Juniper Networks devices support MACsec, and select the Junos OS software version to suit your MACsec needs.
- Configure MACsec on an interface between two directly connected Juniper Networks devices that support MACsec.
- Learn about Static PSK mode and PSK Keychains.
- Avoid common pitfalls when interoperating MACsec between devices from different vendors.
- Choose between 128-bit AES encryption and 256-bit AES encryption and configure optional Fail Open mode.
- Configure MACsec connectivity over a WAN and address potential issues.
- Monitor the statistics of MACsec packets and sessions.

Chapter 1

Introduction

MACsec provides simple yet robust Layer 2 network security to encrypt and protect a single point-to-point Ethernet link. The end device can be an Ethernet switch, a router, or even a host. MACsec is defined by IEEE standard 802.1AE and MACsec encryption is implemented using dedicated hardware and chipsets. Additionally, MACsec security keys and sessions are managed by the MACsec Key Agreement (MKA) protocol software stack running on the Junos OS. This protocol is defined in IEEE standard 802.1X (specifically 802.1X-2010).

You enable MACsec at the logical interface level without needing to know about the underlying traffic flow (i.e. independent of any MAC or IP address information). Every packet on the wire (with the exception of MACsec MKA control packets and a few other Layer 2 control packets) is fully encrypted excluding the source and destination MAC addresses.

Why MACsec?

MACsec is best used to secure device-to-device connections in a colocation center or a data center environment, building-to-building links in a city, or even city-to-city links traversing long distances including undersea or terrestrial cables. You can also encrypt Layer 2 connections via a provider WAN with MACsec. MACsec is increasingly used (as we write this book in early 2019) by governments across the world, big cloud providers in the USA, and the financial industry. Service providers are in the early stages of considering MACsec to provide last mile security for end users and to secure mobile backhaul traffic.

By using dedicated hardware-based encryption engines, you can run MACsec on your interfaces at full duplex, line-rate without suffering any performance or throughput penalties (after accounting for additional MACsec headers). If deploying MACsec is simple, standard, comes included with increasing number of Juniper products, and has near-zero performance penalty, why wouldn't you secure and authenticate your links? This *Day One* book will help you hit the ground running.

Juniper Networks Hardware Supporting MACsec

MACsec is supported on MX, PTX, and ACX series routers. MACsec is also supported on some EX and QFX series switches. Please see Tables 1.1-1.5 for a full list of Juniper Networks hardware that currently supports MACsec.

Table 1.1 is a quick reference table. You can find a more up-to-date version of this information in the data sheets of the corresponding products. Also note that specific feature support will vary based on the capability of the hardware listed. For example, 256-bit AES encryption is not available on MIC-3D-20GE-SFP-E or MIC-3D-20GE-SFP-EH interface cards due to lack of 256-bit encryption capable Hardware.

Table 1.1 Juniper MX Routers Supporting MACsec

Series	Product	Line card	Interface Card
MX	MX10003	MX10003-LC2103 (All variants)	JNP-MIC1-MACSEC
	MX5, MX10, MX40	-	MIC-3D-20GE-SFP-E MIC-3D-20GE-SFP-EH
	MX80, MX104	-	MIC-3D-20GE-SFP-E MIC-3D-20GE-SFP-EH MIC-MACSEC-20GE
	MX240, MX480, MX960	MX-MPC1E (All variants) MX-MPC2E (All variants) MX-MPC3E (All variants) MPC2E-3D-NG (All variants) MPC3E-3D-NG (All variants)	MIC-3D-20GE-SFP-E MIC-3D-20GE-SFP-EH MIC-MACSEC-20GE
	MX240, MX480, MX960	MPC7E-10G MPC10E (All variants)	-
	MX2008, MX2010, MX2020	MX2K-MPC8E (All variants) MX2K-MPC9E (All variants)	MIC-MACSEC-MRATE

Table 1.2 Juniper PTX Routers Supporting MACsec

Note: Table includes products that have yet to start production

Series	Product	Line card	Interface Card
PTX	PTX10008 PTX10016	PTX10K-LC1104 (All variants) PTX10K-LC1105 (All variants)	
	PTX10001	-	-
	PTX10003	-	-

Table 1.3 Juniper ACX Routers Supporting MACsec

Note: Table includes products that have yet to start production

Series	Product	Line card	Interface Card
ACX	ACX6360 ACX5448-M	-	-

Table 1.4 Juniper EX Switches Supporting MACsec

Series	Product	Line card	Interface Card
EX	EX3400 EX4300-48MP EX4300 EX4550 EX4600	-	-
	EX9204 EX9208 EX9214	EX9200-40F-M EX9200-40XS	-

Table 1.5 Juniper QFX Switches Supporting MACsec

Note: Table includes products that have yet to start production

Series	Product	Line card	Interface Card
PTX	QFX10008 QFX10016	QFX10000-30C-M QFX10000-12C-DWDM	
	QFX5100	-	-

Juniper Networks Software Supporting MACsec

Junos OS support for MACsec was introduced in stages starting in 2014, with most of the advanced feature support being introduced in late 2017 and 2018 Junos OS releases. Please refer to the release-specific technical documentation for additional information on specific feature availability on various platforms.

If you plan on enabling 256-bit AES encryption, you must use one of the following Junos OS versions (or a later release).

- 17.4R1-S4
- 17.4R2
- 18.1R3
- 18.2R2
- 18.3R1

TIP As Juniper continues to provide MACsec on an increasing number of products, use the Feature Explorer to obtain the most up-to-date information: https://apps.juniper.net/feature-explorer/.

MACsec Licensing

To activate MACsec, a feature license is required on the MPC7E-10G line card on the MX and on all EX and QFX products. At the time of this writing, no other MX / PTX products have additional licensing requirements to use MACsec. As of Junos OS 18.1R1, all EX and QFX platforms fail to activate MACsec configuration if a license is not present.

If you have additional licensing questions or would like a trial license, please contact your sales representative, or visit: https://www.juniper.net/us/en/how-to-buy/.

Chapter 2

Your First MACsec Link

Configuring MACsec on a Juniper Networks device involves two primary steps on each side of the link:

1. Define a MACsec profile called a *connectivity association*.
2. Assign the MACsec profile to an interface.

Step One: Configure a Connectivity Association

A connectivity association (CA) is a set of MACsec attributes that are used by interfaces to create inbound and outbound MACsec secure channels (or connections in simple terms) through which encrypted bi-directional traffic flows. There are two attributes that you must enter as part of defining a CA. They are `security-mode` and a `pre-shared-key` (PSK). We will just stick to those two attributes for now and ignore the others.

Let's set up our first connectivity association and give it a name: `CA_Basic`:

```
[edit]
user@# set security macsec connectivity-association CA_basic security-mode static-cak
```

NOTE For most MACsec deployments, `security-mode static-cak` is the mode to use.

Now let's program a key. Programming the key involves two items: a *connectivity association key name* (CKN) and the actual *connectivity association key* (CAK):

```
[edit]
user@# set security macsec connectivity-association CA_basic pre-shared-key ckn abcd1234abcd1234

user@# set security macsec connectivity-association CA_basic pre-shared-key cak baba1212baba1212
baba1212baba1212
```

NOTE 32 hex characters were programmed for CAK. 32 hex (16 Octets) is the required CAK length to use for 128-bit AES encryption. If you configure 256-bit encryption later (recommended) it is strongly recommended that you use a 256-bit CAK.

NOTE 16 hex characters (8 Octets) were programmed for CKN. Junos OS has a requirement that the CKN needs to be an even number of hex characters from two to 64 characters (no partial Octets). However, we recommend programming a full-length CKN, which is all 32 octets. While not elegant, it will prevent any interoperability issues that may result from using shorter than full-length CKNs. Such interoperability issues are typically due to incompatible padding mechanisms used by different vendors.

WARNING Use different values for CKN and CAK – think of CKN as a username, and CAK as password. Would you pick a password that is same as your username?

Once the minimum required attributes are programmed for the `connectivity-association CA_basic` let's use a show command to see how it looks:

```
[edit] user@# show security macsec connectivity-association CA_basic

security-mode static-cak;
pre-shared-key {
    ckn abcd1234abcd1234abcd1234abcd1234;
    cak "$9$tYyI0ORhclvMXFnSrKvLXGDjkfTF39AtOJGDk.mTQEcSrM8xNdws4BIhreKx7DiH.
Tz369t0BZUjqPfn69Ap0RhylKMLxhc"; ## SECRET-DATA
}
```

This looks good. Note that the CKN is stored in plain text, whereas CAK is stored in encrypted format, which is how you would expect to see your password stored. It's a simple reversible obfuscation, but there is an even safer way to store your CAK that you'll encounter in Chapter 5.

Step Two: Assign the Connectivity Association to an Interface

Now that the connection profile is ready, let's assign this profile to the interface you want to secure:

```
[edit] user@# set security macsec interfaces xe-1/0/7 connectivity-association CA_basic
```

Both steps are complete and a MACsec profile assigned to an interface. You can commit your changes.

Now let's verify by running the following CLI command to see if MACsec has been successfully configured. Note that by default we picked the GCM-AES-128 in our lab:

```
user@> show security macsec connections
    Interface name: xe-1/0/7
        CA name: CA_basic
        Cipher suite: GCM-AES-128   Encryption: on
        Key server offset: 0        Include SCI: no
        Replay protect: off         Replay window: 0
```

There are no security associations (sessions) formed yet due to the lack of presence of the Inbound/Outbound Secure Channel output (which we'll see in a later step). For that to happen, you need to configure the same MACsec profile on the other system. In the other system, go ahead and configure exactly the same steps, except for the interface name. Commit your changes.

Assuming you exactly followed the above steps on both ends of a link, and assuming the link was up, you should now have a live MACsec connection. To verify, use the following command on either system:

```
user@> show security macsec connections

    Interface name: xe-1/0/13
        CA name: CA_basic
        Cipher suite: GCM-AES-128 Encryption: on
        Key server offset: 0       Include SCI: no
        Replay protect: off         Replay window: 0
          Outbound secure channels
            SC Id: 54:1E:56:B3:CA:A2/1
            Outgoing packet number: 1
            Secure associations
            AN: 0 Status: inuse Create time: 00:10:02
          Inbound secure channels
            SC Id: 54:1E:56:B4:0A:B7/1
            Secure associations
            AN: 0 Status: inuse Create time: 00:10:02
```

Here you can see an interface with one outbound secure channel and one inbound secure channel listed. Congratulations. You have just configured your first MACsec connection.

Before starting the next chapter, there are a few spots within Junos to check your MACsec statistics. One example is the `show interfaces detail` output here:

```
user1@device> show interfaces xe-0/0/11 statistics detail | find MACSec
  MACSec statistics:
    Output
        Secure Channel Transmitted
        Protected Packets               : 0
        Encrypted Packets               : 52
        Protected Bytes                 : 2600
        Encrypted Bytes                 : 624
     Input
        Secure Channel Received
        Accepted Packets                : 22146239
        Validated Bytes                 : 1107055634
        Decrypted Bytes                 : 26872207470
  Interface transmit statistics: Disabled
```

Try it. If you're not seeing the session come up as expected, jump to Chapter 6 for some troubleshooting guidelines.

Chapter 3

Keychains and Key Transitions

The simplest way to establish a MACsec session is to use a PSK within a connectivity association. The downside with this method is that rolling over from one key to the next is not hitless. The solution is to use an authentication keychain.

Keychains for Hitless Key Rollover

Even if you do not have a policy to change keys at regular intervals and only plan to configure and use one key, using a chain from the start makes life easier when you do happen to need to change the key.

NOTE Soon you can make key transitions hitless even for PSKs, (i.e., a single PSK, not just keychains) starting in Junos OS 19.2R1. This will be part of additional MACsec software features that are planned for 2019 (see Chapter 7 for more details).

Configuration

An authentication keychain is a set of one or more keys and key names (called *CAK* and *CKN*, respectively). The only piece of information required is a key start time. Here is a simple one-key configuration showing the keychain and the attachment of the chain within the connectivity association:

```
set security authentication-key-chains key-
chain chain1 key 1 secret 37c9c2c45ddd012aa5bc8ef284aa23ff6729ee2e4acb66e91fe34ba2cd9fe314
set security authentication-key-chains key-chain chain1 key 1 key-name deadbeefcaf4
set security authentication-key-chains key-chain chain1 key 1 start-time "2018-11-27.10:36:00 -0500"
```

Then you apply the keychain to the connectivity association:

```
set security macsec connectivity-association CA_basic pre-shared-key-chain chain1
```

When you need to change the existing pre-shared-key, simply add a new key to the existing chain with a start time in the future:

```
set security authentication-key-chains key-chain chain1 key 2 secret
37c9c2c45ddd012aa5bc8ef284aa23ff6729ee2e4acb66e91fe34ba2cd9fe315
set security authentication-key-chains key-chain chain1 key 2 key-name deadbeefcaf5
set security authentication-key-chains key-chain chain1 key 2 start-time "2018-11-27.10:36:00 -0500"
```

Or:

```
security {
    authentication-key-chains {
        key-chain chain1 {
            key 1 {
                secret "$8$aes256-gcm$hmac-sha2-256$100$Ue0Pb8XNWS0$eCaw1UJ9G+vVwRmqwYLdaQ$BaD6gJFrr
pa7MaSw+fzpoQ$wvoBxojHkjwUoeQDU+3+xcFUjBePhWEfXbaQC4KTFCFI3qO2a2kiZozPh6RpM0YENNqjOJAelqi1H+3+HWKN
Ew"; ## SECRET-DATA
                key-name deadbeefcaf4;
                start-time "2018-11-27.10:36:00 -0500";
            }
        key 2 {
            secret "$8$aes256-gcm$hmac-sha2-256$100$iGUBVt258dE$dvtAIZ4xRBGvwFu6F21VHQ$7qmtNkNnnr8sm
0032pDVmQ$RPbhgWw+veVX5cmPBr0nVdBpGVTmahJ6/+64VvVT0pt7E+BLdKKVG2f9r5ehAbZq+DheaffI8NkmkNtc5qkq
sg"; ## SECRET-DATA
            key-name deadbeefcaf8;
            start-time "2018-11-8.11:56:00 -0500";
            }
        }
    }
}
```

NOTE The timezone suffix on the key start time is optional.

Verification

The key (pun intended) to verifying that your key has rolled over is to check the association numbers (ANs) in either the MKA sessions or MACsec connections output. Within a connectivity association between two peers, there are two unidirectional transmit secure channels. A secure channel has a sequence of security associations (SAs), which roll over from one to the next.

A rollover might be time-based, such as changing from one key to the next, or it might be event-based, such as when the packet counters for a given SA approach wraparound.

NOTE Non-eXtended Packet Numbering (XPN) packet counters are 32 bits, which can wrap within a few seconds on a high-speed link. XPN counters are 64 bits and take much longer to wrap, even on high-speed links. For this reason it's strongly recommended to use XPN for link speeds higher than 40g.

If an endpoint was able to gracefully roll over from one key to the next, the AN will increment by one for both the incoming and the outgoing channel and the create time will reset. If the rollover was unsuccessful, you will see the AN back at zero.

Here is an example of a host with a secure channel whose association number is 0:

```
lab@nyc-mx10k3-1-re1# run show security macsec connections
    Interface name: et-0/1/6
        CA name: static-psk2
        Cipher suite: GCM-AES-256   Encryption: on
        Key server offset: 0        Include SCI: no
        Replay protect: off         Replay window: 0
          Outbound secure channels
            SC Id: D8:18:D3:94:4C:77/1
            Outgoing packet number: 311
            Secure associations
            AN: 0 Status: inuse Create time: 00:06:37
          Inbound secure channels
            SC Id: D8:18:D3:90:E3:97/1
            Secure associations
            AN: 0 Status: inuse Create time: 00:06:37
```

After the key rollover takes place, notice the new AN, a reset of the create time and packet numbers:

```
lab@nyc-mx10k3-1-re1# run show security macsec connections
    Interface name: et-0/1/6
        CA name: static-psk2
        Cipher suite: GCM-AES-256   Encryption: on
        Key server offset: 0        Include SCI: no
        Replay protect: off         Replay window: 0
          Outbound secure channels
            SC Id: D8:18:D3:94:4C:77/1
            Outgoing packet number: 2
            Secure associations
            AN: 1 Status: inuse Create time: 00:01:13
          Inbound secure channels
            SC Id: D8:18:D3:90:E3:97/1
            Secure associations
            AN: 1 Status: inuse Create time: 00:01:13
```

You can see similar output by looking at the MACsec Key Agreement status and checking to ensure that the ANs have incremented:

```
lab@nyc-mx10k3-1-re1# run show security mka sessions
  Interface name: et-0/1/6
      Member identifier: 016ADB2E7B54B35E4077CB5F
      CAK name: DEADBEEFCAF8
      Transmit interval: 2000(ms)
      Outbound SCI: D8:18:D3:94:4C:77/1
      Message number: 32          Key number: 0
      Key server: no              Key server priority: 16
      Latest SAK AN: 1            Latest SAK KI: 38E2A81962057A929D5D9036/1
      Previous SAK AN: 0          Previous SAK KI: 693800E7340E5C7B1FD94852/1
      Peer list
       1. Member identifier: 38E2A81962057A929D5D9036 (live)
          Message number: 41 Hold time: 5500 (ms)
          SCI: D8:18:D3:90:E3:97/1
          Lowest acceptable PN: 1
```

NOTE The CAK name (`DEADBEEFCAF8`) also points to Key 2 indicating successful rollover.

Troubleshooting

As with most parts of MACsec configuration, keychains do not have many parameters to tweak. Typically, MACsec key rollover issues arise from not having the same date and time set on both devices of the link. But even with a small clock mismatch (recommended variation is less than one minute), both sides will gracefully roll over. MACsec by itself has limited key management capabilities. It is the MKA that handles key updates from one secure association to the next.

To enable MKA debugging, you need to enable traceoptions at the interface level as follows:

```
set security macsec interfaces et-0/1/6 traceoptions file macsec-intf
set security macsec interfaces et-0/1/6 traceoptions file size 10m
set security macsec interfaces et-0/1/6 traceoptions flag keys
set security macsec interfaces et-0/1/6 traceoptions flag mka-packets
```

When a key rollover occurs, the key server initiates the procedure. The two peers will have elected a single key server when bringing up the MKA session. You can see which of the two peers is the key server from looking at the MKA output – if you have not changed the default key server priority, the device with the lower MAC address becomes the key server.

To illustrate how MKA works and how you can troubleshoot key exchange and rollover, let's create a scenario with two hosts, one of which has a clock time that is a few minutes faster than the other. The device with the slower clock time is the key server. The following output is from the non-key-server and shows the receipt of a protocol data unit (PDU) from the key server with the AN of 2:

```
Dec  6 09:15:46.619146 MKA actor #1 received MKPDU, SCI D8:18:D3:90:E3:97/1, MI BB:1A:D2:25:71:AF:E2
:58:7E:64:EE:0B, MN 2
Dec  6 09:15:46.619152   Basic parameter set:
Dec  6 09:15:46.619158     MKA version: 1
Dec  6 09:15:46.619163     Key server priority: 16
Dec  6 09:15:46.619168     Key server: 1
Dec  6 09:15:46.619174     MACsec desired: 1
Dec  6 09:15:46.619179     MACsec capability: integrity and confidentiality with offsets
Dec  6 09:15:46.619187     SCI: D8:18:D3:90:E3:97/1
Dec  6 09:15:46.619195     MI: BB:1A:D2:25:71:AF:E2:58:7E:64:EE:0B
Dec  6 09:15:46.619201     MN: 2
Dec  6 09:15:46.619206     Algorithm agility: 802.1X-2009
Dec  6 09:15:46.619213     CKN: DE:AD:BE:EF:CA:F4
Dec  6 09:15:46.619218   SAK use parameter set:
Dec  6 09:15:46.619224     Latest key association number: 2
Dec  6 09:15:46.619229     Latest key tx: 1
Dec  6 09:15:46.619234     Latest key rx: 1
Dec  6 09:15:46.619240     Old key association number: 1
Dec  6 09:15:46.619245     Old key tx: 0
Dec  6 09:15:46.619250     Old key rx: 0
Dec  6 09:15:46.619255     Plain tx: 0
Dec  6 09:15:46.619261     Plain rx: 0
Dec  6 09:15:46.619266     Delay protect: 0
Dec  6 09:15:46.619275     Latest key identifier: AE:67:9B:0E:78:BD:88:E3:E0:0D:BD:85-1
Dec  6 09:15:46.619281     Latest key lowest acceptable packet number: 22
Dec  6 09:15:46.619290     Old key identifier: B2:32:3F:8E:48:0A:97:51:14:55:BD:F5-1
Dec  6 09:15:46.619295     Old key lowest acceptable packet number: 5
Dec  6 09:15:46.619301   Live peer:
Dec  6 09:15:46.619309     Member identifier: 2A:34:2D:BA:69:78:28:FC:86:F0:ED:CD
Dec  6 09:15:46.619315     Message number: 53
Dec  6 09:15:46.619324   ICV: 30:6D:0A:EF:EB:E5:48:EE:E9:1B:F3:5B:60:DB:FD:46
```

At this stage, the non-key-server expects to have already rolled over its key and started using AN 3, but it is unable to because key changes must be initiated from the key server. Remember that the key server's clock is behind. The non-key-server replies with AN 2:

```
Dec  6 09:15:46.880798 MKA actor #0 sending MKPDU, SCI D8:18:D3:94:4C:77/1, MI FB:72:26:72:9F:F8:83:
4E:44:69:46:A5, MN 318
Dec  6 09:15:46.880814   Basic parameter set:
Dec  6 09:15:46.880820     MKA version: 1
Dec  6 09:15:46.880826     Key server priority: 16
Dec  6 09:15:46.880831     Key server: 0
Dec  6 09:15:46.880837     MACsec desired: 1
Dec  6 09:15:46.880842     MACsec capability: integrity and confidentiality with offsets
Dec  6 09:15:46.880850     SCI: D8:18:D3:94:4C:77/1
Dec  6 09:15:46.880858     MI: FB:72:26:72:9F:F8:83:4E:44:69:46:A5
Dec  6 09:15:46.880864     MN: 318
Dec  6 09:15:46.880869     Algorithm agility: 802.1X-2009
Dec  6 09:15:46.880876     CKN: DE:AD:BE:EF:CA:F6
Dec  6 09:15:46.880882   SAK use parameter set:
Dec  6 09:15:46.880887     Latest key association number: 2
Dec  6 09:15:46.880892     Latest key tx: 1
Dec  6 09:15:46.880898     Latest key rx: 1
Dec  6 09:15:46.880903     Old key association number: 1
Dec  6 09:15:46.880908     Old key tx: 0
Dec  6 09:15:46.880914     Old key rx: 0
```

```
Dec  6 09:15:46.880919     Plain tx: 0
Dec  6 09:15:46.880924     Plain rx: 0
Dec  6 09:15:46.880930     Delay protect: 0
Dec  6 09:15:46.880939     Latest key identifier: AE:67:9B:0E:78:BD:88:E3:E0:0D:BD:85-1
Dec  6 09:15:46.880945     Latest key lowest acceptable packet number: 23
Dec  6 09:15:46.880954     Old key identifier: B2:32:3F:8E:48:0A:97:51:14:55:BD:F5-1
Dec  6 09:15:46.880959     Old key lowest acceptable packet number: 5
Dec  6 09:15:46.880965   Live peer:
Dec  6 09:15:46.880973     Member identifier: AE:67:9B:0E:78:BD:88:E3:E0:0D:BD:85
Dec  6 09:15:46.880979     Message number: 395
Dec  6 09:15:46.880988   ICV: 14:F7:D6:00:72:D5:1B:9C:8F:CA:09:20:CE:F5:D6:BE
```

Next, the key-server's clock indicates that it is time to roll over the key and sends back a new AN of 3 with details of the SAK:

```
Dec  6 09:15:47.120379 MKA actor #0 received MKPDU, SCI D8:18:D3:90:E3:97/1, MI AE:67:9B:0E:78:BD:88:E
3:E0:0D:BD:85, MN 396
Dec  6 09:15:47.120385   Basic parameter set:
Dec  6 09:15:47.120391     MKA version: 1
Dec  6 09:15:47.120397     Key server priority: 16
Dec  6 09:15:47.120402     Key server: 1
Dec  6 09:15:47.120407     MACsec desired: 1
Dec  6 09:15:47.120413     MACsec capability: integrity and confidentiality with offsets
Dec  6 09:15:47.120421     SCI: D8:18:D3:90:E3:97/1
Dec  6 09:15:47.120429     MI: AE:67:9B:0E:78:BD:88:E3:E0:0D:BD:85
Dec  6 09:15:47.120435     MN: 396
Dec  6 09:15:47.120440     Algorithm agility: 802.1X-2009
Dec  6 09:15:47.120447     CKN: DE:AD:BE:EF:CA:F6
Dec  6 09:15:47.120452   SAK use parameter set:
Dec  6 09:15:47.120458     Latest key association number: 3
Dec  6 09:15:47.120463     Latest key tx: 0
Dec  6 09:15:47.120468     Latest key rx: 1
Dec  6 09:15:47.120474     Old key association number: 2
Dec  6 09:15:47.120479     Old key tx: 1
Dec  6 09:15:47.120484     Old key rx: 1
Dec  6 09:15:47.120490     Plain tx: 0
Dec  6 09:15:47.120495     Plain rx: 0
Dec  6 09:15:47.120500     Delay protect: 0
Dec  6 09:15:47.120509     Latest key identifier: BB:1A:D2:25:71:AF:E2:58:7E:64:EE:0B-1
Dec  6 09:15:47.120515     Latest key lowest acceptable packet number: 0
Dec  6 09:15:47.120524     Old key identifier: AE:67:9B:0E:78:BD:88:E3:E0:0D:BD:85-1
Dec  6 09:15:47.120530     Old key lowest acceptable packet number: 22
Dec  6 09:15:47.120535   Live peer:
Dec  6 09:15:47.120543     Member identifier: FB:72:26:72:9F:F8:83:4E:44:69:46:A5
Dec  6 09:15:47.120549     Message number: 318
Dec  6 09:15:47.120558   ICV: 6E:C8:EE:C8:67:5F:75:74:A1:97:74:D3:65:A1:AF:4F
```

Even though the key server sends the AN of 3, the *non-key-server* sends two more MKA PDUs to the key server before sending a third with the new AN of 3, completing the bidirectional rollover. During this time, packets can be decoded with either of the two keys until both ends confirm they are using the new SA:

```
Dec  6 09:15:49.459000 MKA actor #1 sending MKPDU, SCI D8:18:D3:94:4C:77/1, MI 2A:34:2D:BA:69:78:28:
FC:86:F0:ED:CD, MN 55
Dec  6 09:15:49.459014   Basic parameter set:
Dec  6 09:15:49.459020     MKA version: 1
Dec  6 09:15:49.459025     Key server priority: 16
```

```
Dec  6 09:15:49.459030     Key server: 0
Dec  6 09:15:49.459036     MACsec desired: 1
Dec  6 09:15:49.459041     MACsec capability: integrity and confidentiality with offsets
Dec  6 09:15:49.459049     SCI: D8:18:D3:94:4C:77/1
Dec  6 09:15:49.459057     MI: 2A:34:2D:BA:69:78:28:FC:86:F0:ED:CD
Dec  6 09:15:49.459063     MN: 55
Dec  6 09:15:49.459068     Algorithm agility: 802.1X-2009
Dec  6 09:15:49.459075     CKN: DE:AD:BE:EF:CA:F4
Dec  6 09:15:49.459094   SAK use parameter set:
x
Dec  6 09:15:49.459106     Latest key tx: 0
Dec  6 09:15:49.459111     Latest key rx: 1
Dec  6 09:15:49.459117     Old key association number: 2
Dec  6 09:15:49.459122     Old key tx: 1
Dec  6 09:15:49.459127     Old key rx: 1
Dec  6 09:15:49.459133     Plain tx: 0
Dec  6 09:15:49.459138     Plain rx: 0
Dec  6 09:15:49.459143     Delay protect: 0
Dec  6 09:15:49.459152     Latest key identifier: BB:1A:D2:25:71:AF:E2:58:7E:64:EE:0B-1
Dec  6 09:15:49.459158     Latest key lowest acceptable packet number: 0
Dec  6 09:15:49.459167     Old key identifier: AE:67:9B:0E:78:BD:88:E3:E0:0D:BD:85-1
Dec  6 09:15:49.459173     Old key lowest acceptable packet number: 23
Dec  6 09:15:49.459178   Live peer:
Dec  6 09:15:49.459187     Member identifier: BB:1A:D2:25:71:AF:E2:58:7E:64:EE:0B
Dec  6 09:15:49.459192     Message number: 3
Dec  6 09:15:49.459201   ICV: 21:95:2B:7B:8C:CE:99:63:5A:0E:72:CA:5C:07:03:90
```

Chapter 4

Configuring MACsec Advanced Features

Congratulations on setting up your link. You are now ready to move to additional options that you can configure on MACsec. While these are called "advanced" options, configuring them is very simple. This chapter shows you how to choose a cipher suite that is different from the default, how to turn on MACsec in integrity only mode without encryption, how to skip a few bytes at the top of the packet from being encrypted, and what optional settings might come in handy when interoperating with other vendors.

Choosing a Cipher Suite

You saw that by default Juniper devices pick `gcm-aes-128` as the cipher-suite. However, there are a total of four cipher suites available to choose from (they go under `connectivity-association`):

```
[edit] user@# set security macsec connectivity-association CA_basic cipher-suite gcm-aes-xpn-256
```

- `gcm-aes-128`: Default mode, enables 128-bit AES encryption.
- `gcm-aes-xpn-128`: Enables 128-bit AES encryption with XPN. XPN is a must for 40GbE and 100GbE interfaces, and it is recommended for 10G interfaces, but is not needed for 1G interfaces.
- `gcm-aes-256`: Enables 256-bit AES encryption. This is not supported on the earlier versions of products, especially MIC-3D-20GE-SFP-E on the MX Series routers. Also, most EX Series products do not support 256-bit encryption.

- `gcm-aes-xpn-256`: Enables 256-bit AES encryption with XPN. XPN is a must for 40GbE and 100GbE interfaces, and it is recommended for 10G interfaces, but is not needed for 1G interfaces. Do ensure that the remote device *also supports* 256-bit encryption.

CAUTION If you configure `xpn` mode on one end and `non-xpn` mode on the other end, you will still see the link being established. However, the behavior and SAK rollovers will be unpredictable.

Should-secure (Fail Open Mode)

If for any reason the MACsec MKA session does not come up between two endpoints, or if the session goes down due to keys getting deleted on one side, you can configure `should-secure` to ensure traffic continues to flow in clear-text. You explicitly enable this at the interface level. This setting goes under `connectivity-association/mka` hierarchy:

```
[edit] user@# set security macsec connectivity-association CA_basic mka should-secure
```

NOTE On MX devices, `must-secure` is the default behavior.

CAUTION Using this feature and behavior is dependent on your organization's security policy and traffic availability goals. Consider the sensitivity of the traffic you are attempting to secure with MACsec. If something interferes with the encrypted status of the link, would you prefer the traffic is dropped versus being potentially exposed in cleartext on the wire? If your customer prefers uptime over security, then `should-secure` would be your best choice, otherwise `must-secure` is your friend.

Include-sci

When you enable MACsec, each packet on the wire gets a security tag, aka SecTag, aka MACsec header. This security tag is usually 8 bytes. If you enable `include-sci`, an additional 8-byte Secure Channel Identifier (SCI) tag is also included in every packet. This tag is required for group connectivity with three or more endpoints, but is not required for point-to point-connections (which is the subject of this book). Some older EX devices, as well as some Cisco devices, always enable this tag. To interoperate with such a device, you must the enable this tag on your device as well. This setting goes under `connectivity-association`:

```
[edit] user@# set security macsec connectivity-association CA_basic include-sci
```

Transmit-interval

The `transmit-interval` sets how often two MACsec endpoints exchange MKA PDUs. The recommended setting is 6000 (six seconds), while the default is 2000 (two seconds) because MKA is handled by the routing engine.

This configuration is under the `connectivity-association/mka` hierarchy:

```
[edit] user@# set security macsec connectivity-association MACSEC_CA_256 mka transmit-interval 4000
```

NOTE Similar to other settings, `transmit-interval` should be set to the same value on both endpoints.

No-encryption

You can use MACsec and MKA only to authenticate an endpoint and guarantee integrity of the link, but chose not to encrypt the packets, which is useful if you need the unencrypted payload to be visible when carrying MACsec over multiple hops such as bump-in-the-wire firewalls or content inspection. This is called *integrity only* mode. If you turn on encryption as well (the default) then it is called *confidentiality mode*.

You can disable encryption by using a *Captain Obvious* attribute called `no-encryption`. This setting also goes under `connectivity-association`:

```
[edit] user@# set security macsec connectivity-association CA_basic no-encryption
```

By the way, if you are using encryption mode or integrity-only mode use the following command to verify:

```
user@> show security macsec statistics
```

You will see output similar to the following showing that encryption is turned off:

```
Secure Channel transmitted
        Encrypted packets: 0
        Encrypted bytes:   0
        Protected packets: 1000
        Protected bytes:   86000
```

Offset

Let's say you wish to have a middle ground between having full confidentiality (encryption on) and integrity only (encryption off) mode. You can achieve this by optionally exposing a set number of bytes of the payload (think IPv4 and IPv6 TCPIP headers and ports) and then encrypting the rest. This could be for an intermediate load balancer activity or for load distribution at the host, in the case of a switch to host links. This is possible by using the `offset` feature. The only two values provided are "30" and "50" bytes:

```
set security macsec connectivity-association MACSEC_CA_256 offset 30
```

Again, one way to ensure `offset` is configured properly is to view the MACsec statistics:

```
user@> show security macsec statistics

        Encrypted packets: 1
        Encrypted bytes:   56
        Protected packets: 0
        Protected bytes:   30
```

You can see that one packet got encrypted and that 30 bytes are correctly skipped from encryption.

Chapter 5

Best Practices and Features

This chapter explains some of the MACsec features and knobs. You should be fully UP right now, so let's get running. And, later, Chapter 7 reviews some of the future capabilities coming to a Juniper Networks device near you.

Keychains for Hitless Key Rollover

As explained previously, the simplest way to establish a MACsec session is to use a PSK within a connectivity association. The downside with a single PSK is that a switch to a new key is not hitless. The solution is to use an *authentication keychain*, even if you do not have a policy to change keys at regular intervals.

An authentication keychain is a set of one or more keys and key names (CAK and CKN, respectively, if you were configuring a PSK). The third required piece of information is a *key start time*. It is probably worth re-stating that with each key a `start-time` is defined but not an `end-time`. Therefore, once your keychain reaches the last defined key – it simply continues to use the last defined key, ad infinitum, until a new one is added on both sides of the link.

The following is a simple one-key configuration showing the keychain and the attachment of the chain within the connectivity association:

```
set security authentication-key-chains key-chain chain1 key 1 secret 37c9c2c45ddd012aa5bc8ef284aa23f
f6729ee2e4acb66e91fe34ba2cd9fe314
set security authentication-key-chains key-chain chain1 key 1 key-name deadbeefcaf4
set security authentication-key-chains key-chain chain1 key 1 start-time "2018-11-27.10:36:00 -0500"

lab@ nyc-mx10k3-1-re1# show
security {
    authentication-key-chains {
        key-chain chain1 {
            key 1 {
                secret "$8$aes256-gcm$hmac-sha2-256$100$Ue0Pb8XNWS0$eCaw1UJ9G+vVwRmqwYLdaQ$BaD6gJFrr
pa7MaSw+fzpoQ$wvoBxojHkjwUoeQDU+3+xcFUjBePhWEfXbaQC4KTFCFI3qO2a2kiZozPh6RpM0YENNqjOJAelqi1H+3+HWKN
Ew"; ## SECRET-DATA
                key-name deadbeefcaf4;
                start-time "2018-11-27.10:36:00 -0500";
            }
        }
    }
}
```

NOTE Refer to Chapter 3 for complete details about using keychains for managing your MACsec keys.

Time Sync Between End Points

Key transitions are triggered based on system time. While both end points are not required to be running the same clock or using Precision Time Protocol (IEEE 1588), it is strongly recommend you ensure that both ends are using same clock, with a variation under one minute.

NOTE You can check current time using the `show system uptime` command.

Transmit-interval as Six Seconds

A low transmit interval period increases overhead on the link and puts a higher load on the routing engine, which manages the MKA protocol for all MACsec interfaces. A very high transmit interval would mean higher time to converge on new MACsec sessions.

A setting of 6000ms for transmit interval is recommended. Refer to Chapter 4 to learn how to configure `transmit-interval`. The default is 2 seconds.

NOTE Ensure that the same value for `transmit-interval` is programmed on both ends of the link.

Set a Master Password on the Device

Users will recognize that CAKs used in PSK mode or PSK keychains mode are shared secrets and their confidentiality is important. However, they must be stored on the router in order to regularly generate keys (SAKs) to be used for actual encryption.

By default, CAKs are protected using reversible obfuscation and stored in the config file. You can see how they are stored by using the `show security` command. Reversible obfuscation is shown starting at `$9$***`:

```
rootuser@machine1# show security
macsec {
    traceoptions {
        file r0_MACSec.log size 1g;
        flag all;
    }
    connectivity-association CA_basic_cp {
        cipher-suite gcm-aes-256;
        security-mode static-cak;
        mka {
            transmit-interval 2000;
            should-secure;
        }
        pre-shared-key {
            ckn 1234;
            cak "$9$lvcKLx-VwgaZdV"; ## SECRET-DATA
        }
    }
```

While this simple obfuscation is better than plain text, it is not recommended for two reasons. First, the hashes are easily reversible using a multitude of online tools. Second, the fact that they are stored in configuration files means they are at risk of being shared when users copy, ship, or upload configuration files.

To improve the handling of your keys, it's recommended to set a master password on your router.

Starting with Junos 15.1, setting a master password option is supported with the primary goal of hardening shared secrets. When you enable a master password on the router, the secret data (CAKs) are encrypted using the AES-256 algorithm before storing in configuration files. The master password itself is stored separately on the device in a different file:

```
rootuser@machine1# set system master-password plain-text-password Gr3@tp@ss
rootuser@machine1# commit and-quit
rootuser@machine1> configure
rootuser@machine1# show security
regress@narmer# show security
macsec {
    traceoptions {
        file r0_MACSec.log size 1g;
        flag all;
```

```
    }
    connectivity-association CA_basic_cp {
        cipher-suite gcm-aes-256;
        security-mode static-cak;
        mka {
            transmit-interval 2000;
            should-secure;
        }
        pre-shared-key {
            ckn 1234;
            cak "$8$aes256-gcm$hmac-sha2-
256$100$8bc+JG1pCE8$RyYspxLvrGe+kSoruxUqmw$x0jC0+VoGAhTgXaEIGl30w$YghFQw"; ## SECRET-DATA
        }
    }
```

NOTE The AES-256 encrypted passwords in the configuration start with 8 (compared to 9 before).

To check if there is a master password currently configured:

```
rootuser@machine1# show system master-password
password-configured;
```

And to delete a master password from configuration:

```
rootuser@machine1# delete system master-password
```

Using Full-length CAKs and CKNs

It's recommended that CKN length be 64 hex digits (32 octets) irrespective of the cipher suite used (128 bits or 256 bits).

NOTE The length of CKN is flexible although there is a requirement that it has to be between one to 32 octets long, which would mean two to 64 hex digits. An odd number of hex digits is not permitted.

We recommend using full-length CAKs. This means 32 hex digits (16 Octets, 128 bits) for 128-bit AES, and 64 hex digits (32 octets, 256 bits) for 256-bit AES.

NOTE CAK length is extremely important for interoperability with third party MACsec devices because a smaller-than-required CAK could result in incompatible padding of zeroes by a third-party vendor when compared to Juniper.

Extended Packet Numbering (XPN)

We strongly recommend using extended packet numbering mode when using MACsec on 40G and 100G links. This means you must only use `gcm-aes-xpn-128` or `gcm-aes-xpn-256` as your `cipher-suite`.

This is related to extending the life of each SA and thus the lifetime of each SAK. To refresh your memory, you program a CAK as part of your static CAK mode or keychain mode. Once agreed upon by both endpoints, CAK is used to frequently generate a SAK that is actually used for encryption. XPN mode extends the life of each SAK because at 100G link speeds the rollover happens very quickly (< 1 minute), resulting in unnecessary control plane activity.

XPN is the second amendment to the original IEEE MACsec standard. By default, it uses a 32-bit value in the SecTag (MACsec header portion in each packet) for sequential packet numbering. Once this packet numbering is close to rolling over, the endpoints switch to the new SAK. If XPN mode is enabled, use a 64-bit PN field instead of 32 bits, thereby extending the life of SAK.

Carrying MACsec Over Multiple Switches

One of the challenges in transporting MACsec and other link-local Layer 2 protocols, such as LLDP and EAPoL (the transport protocol for MKA), is carrying it over multiple switch hops. The latter case could be a VPLS or pseudowire transport service. In normal situations, LLDP, EAP over LAN (EAPoL), LACP, and other similar protocols, would be locally processed by the switch or router and not propagated further. In our case, we want to extend a multi-access Ethernet network over multiple switch or router hops, so these intermediate devices should not interpret these link-local protocols and should instead transparently forward these control protocols to the next hop rather than locally processing them.

Figure 5.1 shows an EAPoL packet leaving switch A, and hitting aggregation switch B, which drops the packet due to incorrect configuration. Layer 2 protocol tunneling is one technique to forward Ethernet control protocols over a switched domain. The rest of this chapter will use switch hop and provider PE or router interchangeable unless the distinction is important.

Layer 2 Protocol Tunneling

Layer 2 control protocols are identified by their Ether Type and MAC address. In the case of MACsec's MKA protocol, EAP over LAN (EAPoL), this is 0x888e and the multicast address of 01:80:c2:00:00:03, respectively. A switch that you configure to tunnel EAPoL traffic identifies traffic with this destination MAC address and Ether Type and encapsulates it in a simple Ethernet wrapper ,whose destination MAC address is 01:00:0C:CD:CD:D0, and forwards this onto the next switch hop or hops. Looking at Figure 5.1, the next switch receives packets with a non-special destination MAC so the packets are forwarded with no local processing.

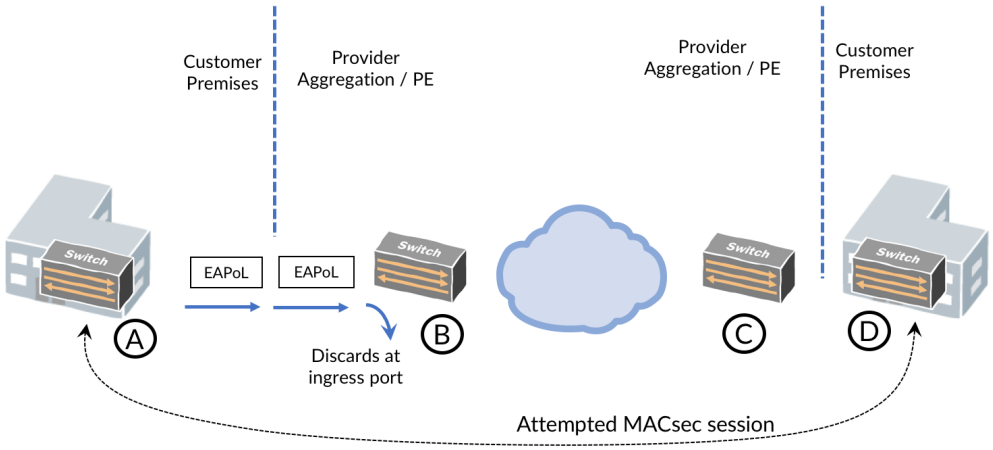

Figure 5.1 *Simple Layer 2 Tunneling Topology*

At the other end of the tunnel, a switch decapsulates the packet, removing the outer Ethernet wrapper, and rereads the destination MAC, which is the EAPOL destination MAC, and forwards the packet out the port towards the MACsec host. This type of tunneling is a rudimentary but effective way to hide Layer 2 control protocols from otherwise nosey intermediate switches. There are no endpoints to define or keepalives, such as in the case of IPSec Dead Peer Detection (DPD).

For most Juniper platforms, the configuration is the same:

```
set protocols layer2-control mac-rewrite interface <interface-name>
```

Under the `interface` hierarchy you add specific protocols you want to tunnel:

```
lab@nyc-ex4300-3# set protocols layer2-control mac-rewrite interface ge-0/0/0 protocol ?
Possible completions:
  <[Enter]>            Execute this command
+ apply-groups         Groups from which to inherit configuration data
+ apply-groups-except  Don't inherit configuration data from these groups
> cdp                  Enable mac rewrite for CDP
> elmi                 Enable mac rewrite for ELMI
> gvrp                 Enable mac rewrite for GVRP
> ieee8021x            Enable mac rewrite for 8021X
> ieee8023ah           Enable mac rewrite for 8023AH
> lacp                 Enable mac rewrite for LACP
> lldp                 Enable mac rewrite for LLDP
> mmrp                 Enable mac rewrite for MMRP
> mvrp                 Enable mac rewrite for MVRP
> stp                  Enable mac rewrite for STP
> udld                 Enable mac rewrite for UDLD
> vstp                 Enable mac rewrite for VSTP
> vtp                  Enable mac rewrite for VTP
  |                    Pipe through a command
{master:0}[edit]
lab@nyc-ex4300-3#
```

Not all platforms have the same tunneling capabilities. At the time of this writing, the EX2300, EX3400, EX4300, and EX4600 can explicitly tunnel while QFX and MX platforms can only tunnel CDP, PVSTP, STP, and VTP.

NOTE Other exceptions are noted here: https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/protocol-edit-protocols-layer2-control.html.

This is not the end of the discussion on transporting Layer 2 link-local control protocols over a switched domain. Some platforms and configurations can forward traffic over a Layer 2 point-to-point transport such as a Layer 2 circuit (LDP-signaled service label) or Layer 2 VPN (BGP-signaled service label). Some types of multi-point technologies such as EVPN are possible but depend on the PE platform.

Automatic Layer 2 Protocol Forwarding

A foolproof way to carry Layer 2 control protocols such as MACsec is with a point-to-point MPLS tunnel such as LDP (RFC4447 / Martini L2circuits) or BGP-signaled (RFC4761 / Kompella L2VPN) pseudowires. This strategy has been implemented on the QFX10000 Series and MX Series platforms. The CE-facing port on the PE should be configured with `ethernet-ccc` encapsulation. An example configuration of one end of the LDP-signaled pseudowire is shown here:

```
interfaces {
    et-0/0/1 {
        encapsulation ethernet-ccc;
        unit 0;
    }
}

protocols {
    l2circuit {
        neighbor 192.168.1.2 {
            interface et-0/0/1.0 {
                virtual-circuit-id 100;
            }
        }
    }
}
```

Let's verify the Layer 2 circuit is up and running as follows:

```
lab@nyc-qfx10k2-2# run show l2circuit connections
Layer-2 Circuit Connections:

Legend for connection status (St)
EI -- encapsulation invalid      NP -- interface h/w not present
MM -- mtu mismatch               Dn -- down
```

```
EM -- encapsulation mismatch     VC-Dn -- Virtual circuit Down
CM -- control-word mismatch      Up -- operational
VM -- vlan id mismatch           CF -- Call admission control failure
OL -- no outgoing label          IB -- TDM incompatible bitrate
NC -- intf encaps not CCC/TCC    TM -- TDM misconfiguration
BK -- Backup Connection          ST -- Standby Connection
CB -- rcvd cell-bundle size bad  SP -- Static Pseudowire
LD -- local site signaled down   RS -- remote site standby
RD -- remote site signaled down  HS -- Hot-standby Connection
XX -- unknown

Legend for interface status
Up -- operational
Dn -- down
Neighbor: 192.168.1.1
    Interface                 Type  St     Time last up          # Up trans
    et-0/0/34.0(vc 100)       rmt   Up     Dec 22 18:24:37 2018           1
      Remote PE: 192.168.1.1, Negotiated control-word: Yes (Null)
      Incoming label: 19, Outgoing label: 19
      Negotiated PW status TLV: No
      Local interface: et-0/0/34.0, Status: Up, Encapsulation: ETHERNET
      Flow Label Transmit: No, Flow Label Receive: No

{master:0}[edit]
lab@nyc-qfx10k2-2#
```

Although a multipoint technology such as VPLS or EVPN is an option, you need to be more judicious to use the correct combination of platform and configuration. For instance, the interface and EVPN-MPLS configuration snippet below has been successfully tested on the MX platform. But if you try a similar EVPN configuration on the QFX (using `ethernet-switching` in place of `bridge address family`), the platform will fail to carry the MACsec packets across the EVPN.

You should use the same encapsulation type and routing instance configuration format on the other PE in the EVPN, replacing the interface with your own interface name:

```
interfaces {
    ge-1/0/0 {
        unit 0 {
            family bridge {
                interface-mode access;
                vlan-id 1;
            }
        }
    }
}
routing-instances {
  vpls-vs-1 {
        instance-type virtual-switch;
        interface ge-1/0/0.0;
        route-distinguisher 100:100;
        vrf-target target:1:1;
```

```
        protocols {
            evpn {
                extended-vlan-list 1;
            }
        }
        bridge-domains {
            bd-1 {
                domain-type bridge;
                vlan-id 1;
            }
        }
    }
}
```

MORE? To learn more about troubleshooting and operating EVPN, visit the Juniper TechLibrary: https://www.juniper.net/documentation/en_US/junos/topics/concept/evpns-overview.html. Or, in the *Day One* library: https://www.juniper.net/us/en/training/jnbooks/day-one/proof-concept-labs/using-ethernet-vpns/.

Chapter 6

Troubleshooting and Interoperability

This chapter lists ways to check common misconfigurations between devices. We have included many of the more common errors but do not show them all, since some occur from unusual conditions (such as receiving duplicate Ethernet frames, which may happen during a broadcast loop). Instead, we have selected the most common scenarios you might encounter.

MORE? The Juniper Forums features troubleshooting discussions and insights: forums.juniper.net.

MACsec Protocol Concepts

It's helpful to understand some of the concepts when diving deeper into the machinery of MACsec. Luckily, the protocol workings are relatively simple. A useful way to think of a MACsec-protected link is to split it into a control plane and a data plane. Figure 6.1 illustrates the MKA control plane and the actual user data with MACsec wrappers, the data plane.

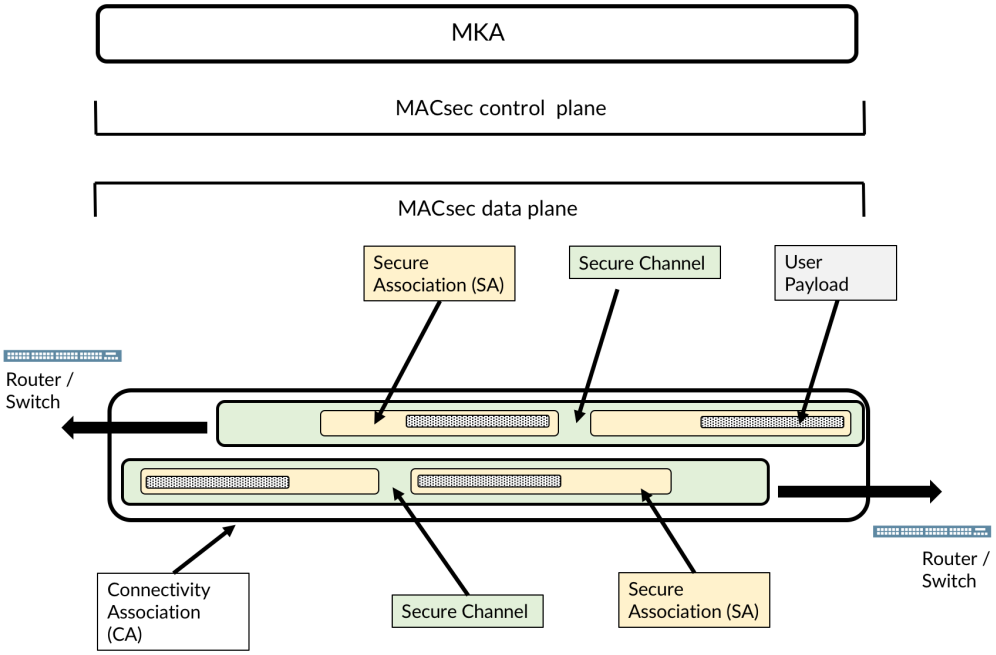

Figure 6.1 *MACsec User and Data Planes*

A connectivity association (CA) is a common MACsec relationship between all hosts on a LAN who share a common key. In a MACsec association with two hosts, a CA contains two unidirectional secure channels, one from each host. Finally, each secure channel has a series of SAs. In Junos OS, SAs last the shorter of either the time until the next key rollover in the chain or the wrap of the packet sequence counter. The operator configures the CAK, which secures the CA and from which Junos OS derives the SAK, which in turn secures each SA.

The initial IEEE MACsec standard (IEEE 802.1AE) describes the data plane functions and excluded key management. An additional IEEE specification (802.1X-2010) provides the mechanism for hosts to leverage the 802.1X specification to manage keys between hosts.

Incorrect CAK

Ethernet packets encrypted with a mismatched CAK will increment the CAK mismatch packets counter, as shown here:

```
user@EX4600A> show security mka statistics
   Interface name: xe-0/0/2
      Received packets:                 362
      Transmitted packets:               364
      Version mismatch packets:           0
      CAK mismatch packets:              50
      ICV mismatch packets:              0
      Duplicate message identifier packets: 0
      Duplicate message number packets:    0
      Duplicate address packets:          0
      Invalid destination address packets:  0
      Formatting error packets:          0
      Old Replayed message number packets:  0
```

Mismatched SCI Setting

We recommend configuring the `include-sci` tag on both ends if you are bringing up a MACsec link between two different hardware devices, such as EX Series vs. MX Series, or a Juniper device and a another vendor's device.

The SCI resides within a connectivity association. Each host tags an SCI onto traffic that it sends to other hosts on the LAN. On point-to-point links, a router has only one other SCI on the link, therefore, an SCI does not serve much purpose on point-to-point links. But on LANs with more than two secure hosts, you need to include an SCI to exchange MACsec packets. Some platforms such as the EX4300 and the MIC-3D-20GE-SFP-E/EH line card force an SCI onto all outgoing traffic regardless, and this setting cannot be disabled. Hence our recommendation to include the tag on both ends.

Interestingly, if an EX4300 receives a packet without an SCI tag, the switch still accepts the packet as valid.

If a device such as an MX receives a packet with an SCI that it is not expecting (therefore, the default state), the router discards the packet and increments the "No SCI" counter shown below:

```
lab@nyc-mx10k3-2-re0> show security macsec statistics interface xe-1/1/5:0 detail
    Secure Channel transmitted
        Encrypted packets: 25
        Encrypted bytes:   4110
        Protected packets: 0
        Protected bytes:   0
    Secure Association transmitted
        Encrypted packets: 25
        Protected packets: 0
    Secure Channel received
        Accepted packets:  0
```

```
        Validated bytes:   0
        Decrypted bytes:   0
    Secure Association received
        Accepted packets:  0
        Validated bytes:   0
        Decrypted bytes:   0
    Error and debug
    Secure Channel transmitted packets
        Untagged: 0, Too long: 0
    Secure Channel received packets
        Control: 139, Tagged miss: 0
        Untagged hit: 0, Untagged: 0
        No tag: 0, Bad tag: 0
        Unknown SCI: 0, No SCI: 25
        Control pass: 0, Control drop: 0
        Uncontrol pass: 0, Uncontrol drop: 0
        Hit dropped: 0, Invalid accept: 0
        Late drop: 0, Delayed accept: 0
        Unchecked: 0, Not valid drop: 0
        Not using SA drop: 0, Unused SA accept: 0
```

Differing Key Sizes

An incorrect key size will show up integrity check value (ICV) errors. The ICV is a variable length field that is dependent on the cipher type. For example, a 256-bit cipher scheme will have a larger ICV than a 128-bit cipher scheme. To find evidence of a cipher scheme mismatch, look in MKA statistics:

```
lab@nyc-mx10k3-2-re0> show security mka statistics
    Interface name: et-1/1/6
        Received packets:                     929
        Transmitted packets:                  930
        Version mismatch packets:             0
        CAK mismatch packets:                 0
        ICV mismatch packets:                 929
        Duplicate message identifier packets: 0
        Duplicate message number packets:     0
        Duplicate address packets:            0
        Invalid destination address packets:  0
        Formatting error packets:             0
        Old Replayed message number packets:  0
```

If you configure one endpoint with XPN, and the other end without XPN, but with the same `key-bit-size`, traffic will still successfully pass and you will not see ICV errors.

Decoding MACsec Statistics

When you look at MACsec statistics, you'll see two similar-seeming types of statistics under the secure channel: `Protected bytes/packets` and `Secured bytes/packets`. This difference stems from whether traffic is both encrypted and signed with an ICV (encrypted) or just signed with an ICV (protected).

By default, Junos OS encrypts and signs traffic. It then increments the `Encrypted bytes/packets` for traffic it sends. For received packets it increments the `Accepted packets` and `Decrypted bytes` counters.

For traffic that is only signed (a packet that is unencrypted but still has an ICV), Junos OS signs the packet then increments the protected packets/bytes for `sent` packets and increments the accepted packets/validated bytes for `received` packets. To set Junos OS to only sign, but not encrypt, traffic, you need to include the following configuration under your connectivity association:

```
[edit]
user@# set security macsec connectivity-association <connectivity-association-name> no-encryption
```

NOTE Asymetric configuration of encryption and protection is possible. With one end of the link set to no-encryption (i.e., integrity only) while the other side is set to encrypt, routers will successfully process traffic in both directions.

You will notice two types of statistics when looking at MACsec statistics: *secure channel* and *secure association*. A secure channel consists of one or more consecutive secure associations, for example when using keychains with multiple keys, you would rollover to a second secure association and a new key but maintain the same secure channel.

The next section explains in more detail how to troubleshoot key rollovers.

Keychains and MKA

A keychain is a way to switch a set of peers to a new pre-shared key in a hitless fashion. Ideally the clocks of both systems would be synchronized with NTP, however this is not a strict requirement. MKA protocol is tolerant in minor time variations between the end points. We recommend the time difference between the two end points to be less than a minute.

Upgrading Junos and Config Validation

If you are trying to upgrade from one Junos version to another and you notice config validation checks failing at the MACsec portion, please check to see if you are using `master password` on the router. While we recommend using `master password` as a MACsec best practice, having it enabled will cause validation checks to fail during an OS upgrade. This is a harmless issue that will be resolved in future Junos releases. For now, please use `no-validate` option if you are MACsec user and are upgrading from one OS version to another.

```
user@> request system software add junos-mx-17.4R2S4.tgz no-validate
```

Missing MACsec Option

If you can't find the `macsec` option under the `set security` hierarchy, you are probably using a limited version of Junos. You need a full version of Junos to be able to configure `macsec`.

MACsec Overheads

As discussed previously, there is a packet size increase as a result of the additional encapsulation of the original packet as it leaves the switch/router and is encapsulated with MACsec. This overhead is typically negligible (24B or 32B if the `include-sci` option is used, which adds an additional 8 octets) for an average packet distribution.

Let's consider a scenario where an inbound interface is running at 100% line rate on a 10G ingress port, and must transit a 10G MACsec encrypted link before reaching its destination. At 80byte packet sizes, the MACsec overhead added on the core-facing MACsec enabled link can account for ~20% of the original packet. This roughly translates to ~80% maximum theoretical input throughput as the other 20% is consumed by the MACsec header on the wire of the core link. AAs packet sizes approach 1500 bytes this begins to be negligible, but must be considered depending on the average expected packet size on your network requiring encryption. This will be highly dependent on your network and the types of traffic that traverse it.

Given the above scenario there are two ways to approach troubleshooting such an issue:

1. The `show interface statistics <interface>` command will show you the current pps/bps rate on a given interface calculated without any additional overheads. So, you might find on a 10g interface that you are only able to pass 8G of traffic at small packet sizes:

```
user@> show interfaces statistics xe-0/0/6 | match "Input rate|Output rate"
  Input rate     : 1720 bps (2 pps)
  Output rate    : 1592 bps (1 pps)
```

2. The `show interface queue <interface>` command includes additional MACsec overhead, so the bps outputs in this command shows you the original packet (including MACsec headers) in its calculated output. In the example in scenario 1, that would likely be closer to your theoretical 10G throughput number after adding the associated MACsec overhead:

```
user@>show interfaces queue xe-0/0/6
Physical interface: xe-0/0/6, Enabled, Physical link is Up
  Interface index: 178, SNMP ifIndex: 538
Forwarding classes: 16 supported, 4 in use
Egress queues: 8 supported, 4 in use
Queue: 0, Forwarding classes: scavenger
  Queued:
    Packets              :                     0                     0 pps
    Bytes                :                     0                     0 bps
  Transmitted:
    Packets              :                     0                     0 pps
    Bytes                :                     0                     0 bps
    Tail-dropped packets :                     0                     0 pps
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    RED-dropped packets  :                     0                     0 pps
     Low                 :                     0                     0 pps
     Medium-low          :                     0                     0 pps
     Medium-high         :                     0                     0 pps
     High                :                     0                     0 bps
    RED-dropped bytes    :                     0                     0 bps
     Low                 :                     0                     0 bps
     Medium-low          :                     0                     0 bps
     Medium-high         :                     0                     0 bps
     High                :                     0                     0 bps
  Queue-depth bytes      :
    Average              :                     0
    Current              :                     0
    Peak                 :                     0
    Maximum              :              18874368
```

MACsec Interoperability Checklist

To recap: For a quick check before trying to establish a MACsec session between Juniper and a third-party device, please refer to the following checklist (also handy for Juniper-to-Juniper configuration).

- ✓ Ensure the link is up before configuring MACsec.
- ✓ Ensure both ends are using the same cipher-suite. If using 256-bit AES, ensure both systems support 256-bit encryption.
- ✓ If using 256-bit encryption, ensure that you are using a Junos OS version that supports 256-bit AES.
- ✓ Ensure that you are using xpn mode on both ends if 40G or 100G.
- ✓ Ensure same CAK, CKN are configured on both ends.
- ✓ Ensure both end points are using full length CKNs ie. 16 Bytes long.
- ✓ Ensure both end points are using full length CAKs. 8 Bytes for 128-bit AES mode and 16 bytes for 256-bit AES mode.

- ✓ Ensure "encryption" is enabled on both ends. It is enabled by default on Juniper gear. Note that MACsec can also run in integrity-only mode.
- ✓ Turn on `include-sci` tag option on both ends. This is because some devices always include this tag irrespective of the setting.
- ✓ Make sure both ends have their system time in sync. Recommended variation between the clocks is less than one minute.

Chapter 7

What's Next?

As this book was being written there were additional MACsec features in development or under consideration, and these will be covered in a second edition. Check the *Day One Library* for new editions at https://www.juniper.net/books. Here's a sneak peek of what we couldn't write about quite yet, and what features are coming soon.

WAN and VLAN (IFL) MACsec

You will be able to establish MACsec sessions for VLANs (IFLs) instead of physical interfaces (IFD) today. This feature, in combination with VLAN tags in the clear, will enable point-to-multipoint VLAN connections over service provider WANs. Figure 7.1 depicts how the packet structure of a WAN MACsec or VLAN-in-the-clear MACsec is encoded on the wire. The VLAN tag is now unencrypted, leaving the VLAN tag and the two MAC address fields in the clear, allowing intermediate switches to switch, also based on the VLAN tag.

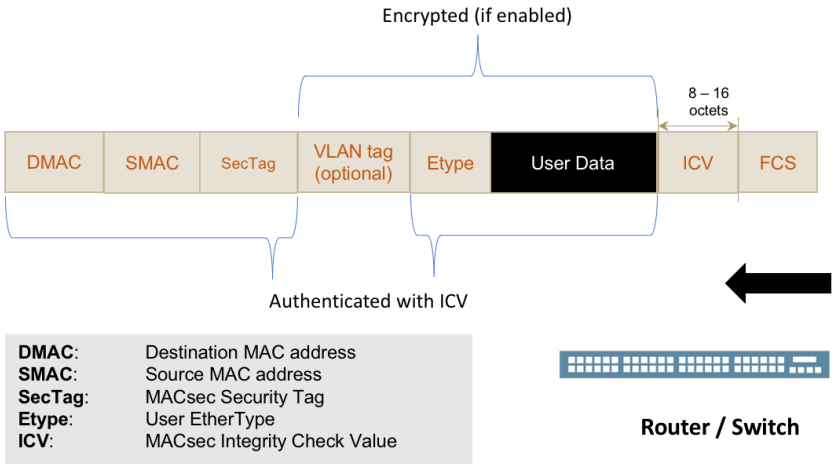

Figure 7.1 *VLAN-in-the-clear Packet Structure*

Also, 802.1x (MKA) control packets could get trapped by an intermediate device preventing the sessions from getting established. Junos OS will be able to change the MAC destination address of MKA packets ability to specify a unicast MAC DA. This will help mitigate the intermediate trap events.

Fallback Sessions

If you're using static CAK security mode as well as hitless PSK keychains security mode and the session doesn't establish for any reason or gets torn down due to one endpoint restarting or due to mismatching keys, today you only have two options: either to send traffic in clear text (`fail-open` / `should-secure mode`) or to bring down the link (`must-secure`).

In the near future, you'll have an ability to provision a fallback CAK on each connectivity association. If the same fallback CAK is programmed on both ends of the link, then when a primary key mismatch occurs, or if the primary session doesn't get established for any reason, then it will use the fallback CAK to establish a fully-encrypted MACsec session.

SAK Rollover Timers

Today, SAKs are switched when the PNs rollover or are about to rollover. In cases where there is little secure traffic flowing on the session, the SAK rollover could take longer than desired. In some cases, this could be days or months depending on traffic patterns.

While we have the ability to seamlessly switch CAKs (reminder: random SAKs are derived from CAKs every time the packet number field rollovers), the typical recommendation as well as user usage results in a CAK life time for several days to months.

Therefore, we need a mechanism to force SAK rollovers sooner rather than waiting for packet numbers to rollover.

Junos MACsec has a feature on the roadmap to introduce a timer-based trigger per connectivity association to trigger SAK rollovers.

Hitless GRES and NSR with MACsec

TToday, when a routing engine failure happens and triggers a failover mechanism such as graceful Routing Engine switchover (GRES) or nonstop active routing (NSR), MACsec sessions go down and get re-established once the switchover is complete. This results in brief (a few seconds) traffic interruption, as well as several Layer 2 and Layer 3 protocols going down and then restarting.

This is one roadmap feature that is a high priority; the goal is to be able to do hitless GRES and NSR with MACsec sessions continuing to stay up.

Dynamic CAK and Certificate-based Authentication

Dynamic CAK security mode is used between a switch and a host. While Junos OS currently supports dynamic CAK behavior on the EX Series switches, this feature is not supported on any of MX Series routers.

Juniper Networks is also working to add the support of certificate-based authentication to all of our products, in addition to providing this dynamic CAK feature. Stay tuned.

MACsec Everywhere with Juniper ASICs

Juniper has now integrated 256-bit capable MACsec encryption at full line rate into its future silicon. This includes both the Trio family of ASICs and the Express chipset, which means that a growing number of current and a majority of future generations of routers and switches will ship by default with MACsec capability. This includes native MACsec lines up to 400GE.

Good times are ahead. Or perhaps we should say… safer and secure networks are ahead.

Appendix

MACsec FAQ

What is the difference between a CAK and a CKN?

A CAK is a long hexadecimal key that lasts for the duration of the connectivity association. The CKN is a shorter name you can assign to the key.

Do I need a special hardware or software to do MACsec?

Most EX Series platforms can do MACsec with their built-in hardware. See here: https://www.juniper.net/documentation/en_US/junos/topics/topic-map/understanding_media_access_control_security_qfx_ex.html.

All other QFX and MX platforms need special hardware to process MACsec.

Older EX 14.1-heritage of software needed a special controlled version of Junos OS to function. Later versions of Junos OS code support MACsec in either the domestic or standard version of Junos. The limited image does not support MACsec.

What is MACsec Encryption / Decryption Performance?

Encryption / decryption is implemented in hardware and it is fully line rate after accounting for 16-24 bytes of MACsec header in each packet. MACsec encryption will add a very small additional latency to the system (approximately 0.25-0.5 us).

What are the extra overheads when MACsec is enabled?

16 to 24 bytes (with SCI). Note that in MIC-3D-20GE-SFP-E SCI the tag is always on.

Is it possible to enable encryption on an aggregated link?

Yes, but each physical member link should be configured separately.

Do MX routers transport MACsec packets transparently?

MACSec and MACSec Key Agreement protocol packets are bridged / cross-connected transparently by default. It is possible to set up a filter and discard them if needed.

When I breakout 4x10G, etc., can MACsec be turned on independently on just one link?

Yes, this is supported. No limitations on each individual interface. MACsec behavior doesn't change between a stand-alone interface and a breakout interface.

Does Juniper support router failover features (GRES/NSR) with MACsec on?

Currently GRES / NSR will temporarily impact the traffic. The link will go down for a few seconds and will automatically come back up. This could potentially bring down protocols such as ISIS, BFD, LACP, etc. This support is currently on the MACsec product roadmap for 2019.

Packet Structures

Figure 8.1 shows an Ethernet packet secured by MACsec. There are two aspects to the modification that MACsec makes to the packet. The first is the authentication data which is contained in the ICV field. The ICV field is a sophisticated checksum of the whole original user packet except the ICV itself. This field ensures that no part of the original packet was modified. The second aspect is the optional (but usually enabled) encryption, which includes all fields of the user packet, including the original EtherType, VLAN tags, and user data.

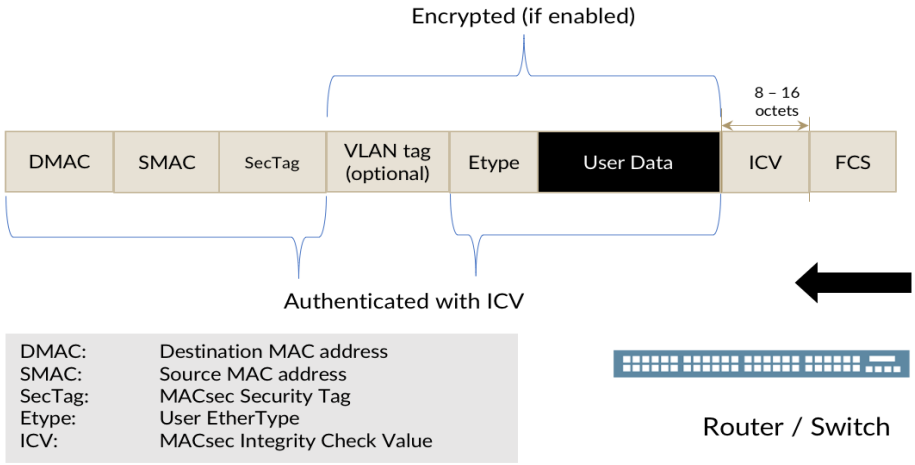

Figure 8.1 *Basic MACsec Packet Structure*

In the packet in Figure 8.2, you can see VLAN tags are included within MACsec's encryption scope. Since intermediate switches are unable to see tagged information, they treat all VLANs as being part of a single broadcast domain. *VLAN-in-the-clear* is an upcoming feature described in Chapter 7, which leaves both MAC addresses and VLANs unencrypted, allowing for more complex switching topologies.

Figure 8.2 explores a more detailed look at the MACsec Security Tag packet structure. The EtherType is a fixed value of 0x88e5. The tag control information field is a six-bit set of fields to indicate encryption data (for example, is it enabled) and secure channel information. The association number is a two-bit field that helps to create a unique secure association identifier. Short Length (SL) helps MACsec decoders to handle payloads of less than 48 bytes. A PN is a 32-bit sequence number for non-XPN environments (64-bit for XPN), which requires establishing a new SA after counter wrapping. SCI is an optional field that is typically used in MACsec environments with more than two hosts in a connectivity association.

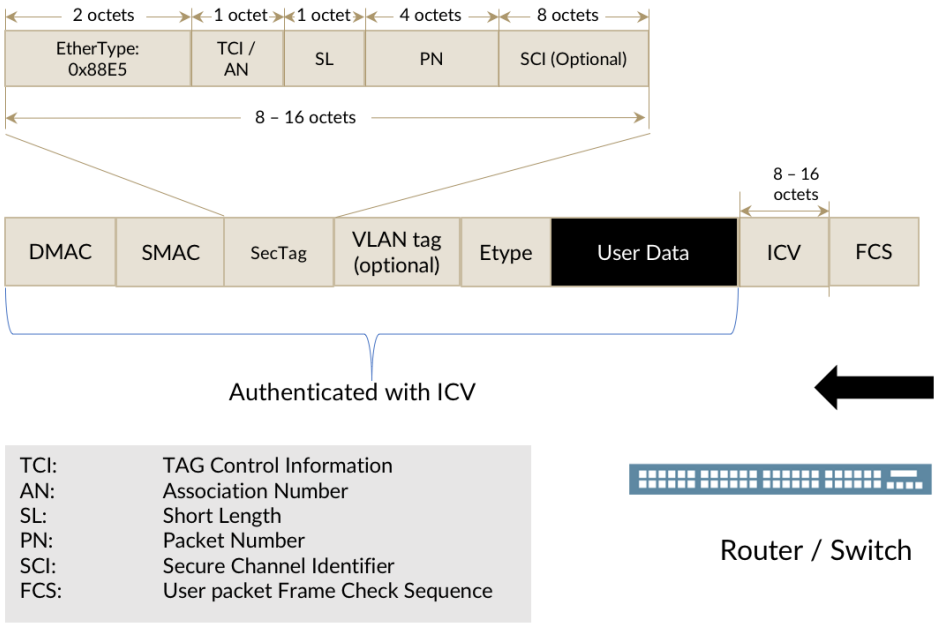

Figure 8.2 *MACsec Packet and Security Tag*

MACsec Speak – Glossary

Here's a glossary review to get you familiar with MACsec concepts, and test your understanding of the technology:

AN Association Number

This is a two-bit value that is concatenated with a Secure Channel Identifier to identify a Secure Association. To easily identify whether a SAK has rolled over, check the AN.

CA Connectivity Association.

A long-term relationship between endpoints that share common MACsec parameters, such as a key. On a router-to-router link, a CA would contain two unidirectional secure channels.

CAK Connectivity Association Key

A long-term key either configured statically or assigned dynamically via RADIUS.

CKN Connectivity Association Key Name

A Connectivity Association Key name (CKN) is a Key Name value in hexadecimal that identifies a CAK. A CKN can be a short (but an even number of characters) set of values to identify a key. Corresponding CKNs are shared by participating endpoints between each other when coming to an agreement on which CAK to use.

ICV Integrity Check Value

A field similar to the CRC, which protects the destination and source MAC addresses of the Ethernet packet and everything in between. The Integrity Check Value (ICV) helps to validate that the MACsec packet in either case of an encrypted or unencrypted MACsec session.

MKA MACsec Key Agreement

A protocol used between MACsec peers to help synchronize keys. MACsec itself does not have much key management capability and leaves it up to other techniques to distribute keys. Some of MKA's most important functions are to securely encrypt and distribute the SAK between peers and handle the rollover of keys. One host within a CA is elected key server (lowest priority wins) to distribute key information. MKA also has a built-in polling interval, which acts as a keepalive to allow for key server redundancy. MKA uses the EAP over LAN Ether Type of 0x888e.

SA Secure Association (SA)

A secure channel has a succession of secure associations (SAs), each with its own secure association number, which is derived from an association number. A secure association key (SAK) is unique for the SA and is derived from the long term CAK. Junos rolls over from the next SA either when using hitless rollovers and a keychain or when packet sequence numbers would roll over. The Junos OS does not periodically renegotiate a SAK.

SAK Secure Association Key

A key that is derived from the Connectivity Association Key and is relevant only for the length of the Secure Association between two peers. The secure Association Key (SAK) is securely distributed between MACsec peers.

SC Secure Channel

A unidirectional channel from one host to one or more hosts on the same broadcast domain. Since most deployments are on point-to-point links, a secure channel (SC) would have one recipient. An SC has a succession of secure associations (SAs).

SCI Secure Channel Identifier

A Secure Channel Identifier (SCI) is a concatenation of the Secure Channel sender's MAC address and a port ID. This stays the same throughout rollovers of secure associations within a secure channel. The EX4300 includes an SCI so any peers of this switch should not disable the SCI.

SecTAG MAC Security Tag

A SecTAG is the MACsec header, beginning with the MACsec ethertype of 0x88e5. This follows the standard destination and source MAC fields normally present on any Ethernet packet.

XPN Packet Number and eXtended Packet Number (PN, XPN)

A sequence number increasing by one for each packet sent within a SA. The initial MACsec specification used a 32-bit identifier, which meant a new SA had to be negotiated relatively frequently on high-speed links. eXtended Packet Numbering (XPN) increases the packet sequence number space to 64-bits.