BGP – Secure Routing Extension (SRx) Quick installation Guide

For SRx Software Suite 4.2.0

Introduction

BGP-SRx consists of a set of software modules that enable the usage of the Resource Public Key Infrastructure (RPKI) Route Origin Validation as well as the usage of BGPSEC path validation. The design of BGP-SRx is modular and can be used on a single machine or distributed.

The following software modules are provided:

SRxCryptoAPI

This is the Crypto module for BGPSEC path validation It is a standalone API that can be integrated into router software such as Quagga, BIRD or others and provides the functionality of validating and signing BGPSEC path attributes. Within the SRx Software Suite 4.2 this API is one of the core APIs needed.

SRx-Proxy / SRx-Server

The SRx-Server allows to outsource RPKI origin validation and BGPSEC path validation. It can run on the BGP router or any other Linux based network reachable server. It provides the functionality of gathering ROAS for RPKI validation / and Keys for BGP validation. It keeps track of changes within the validation state and reports back to the router in case of changes. The SRx-Server is accessed via the SRx-Proxy which is a light-weight API that does the communication with the server.

The SRx-Server comes with a set of test tools that allow to generate public and private keys, create a file based key volt as well as test harnesses that simulate a validation cache etc. These tools are mainly for test purpose but have the capability to be extended for more.

QuaggaSRx

This is the SRx routing platform. Based on Quagga 0.99.22 this implementation is extended to allow policy configuration for RPKI origin validation as well as includes the BGPSEC Path protocol implementation. With QuaggaSRx 0.4.2.0 the software includes the latest draft specification of BGPSEC.

Even though we tested this implementation in different scenarios, QuaggaSRx 0.4.2.0 is still in its BETA state and we appreciate bug reports to be send to bgpsec-dev@antd.nist.gov

The current implementation does not fully include confederations at this point but this will be added in the near future.

BGPSEC-IO

BGPSEC-IO is the latest in the line of SRx software tools. It allows to test crypto modules as well as BGPSEC router implementations. BGPSEC-IO allows to send BGPSEC traffic via eBGP as well as iBGP as well as BGP4 traffic. It allows to have different modes for signing failures – Fake key and signature information as well as BGP4 whenever it can't sign. It allows to generate multi hop fully signed bgpssec updates using either a configuration script, pre-generated binary traffic, via command line or via pipe input. The configuration script as well as the command line help -? are pretty much self-explanatory.

Quick Installation

This Guide is meant for a quick installation. The software is developed on CentOS6 as well as CentOS7 whereas the rpm script generation is not ported to the new CentOS7 system-d infrastructure yet.

Install and test the software using the following steps. This makes it easier to troubleshoot in case something went wrong. Most of the errors occur due to configuration mistakes.

To make it easy we specify two folder structure, SRX_FOLDER and SRX_INSTALL. The structure SRX_FOLDER will contain all source files and SRX_INSTALL will contain the compiled and installed binaries / configurations. In the following paragraphs we simply refer to the m as STRX_FOLDER and SRX_INSTALL and it is understood that they have to be replaced with the chosen folder names.

Now extract all software modules srx-crypto-api, bgpsec-io, srx-server, and quagga-srx inside the folder SRC_FOLDER. This will provide the following folder structure:

/SRX_FOLDER/bgpsec-io
/SRX_FOLDER/quagga-srx
/SRX_FOLDER/srx-crypto-api
/SRX_FOLDER/srx-server

Install SRxCryptoAPI

This first step is to install and test the SRxCryptoAPI.

1. Copy the code to a folder of your choice. We call it the SRX_FOLDER. Specify an installation folder. Let us call it SRX_INSTALL. You should have the following structure:

   /SRX_FOLDER/srx-crypto-api

   Enter this folder

   cd SRX_FOLDER/srx-crypto-api

2. Call the configuration script and specify the installation folder.

   ./configure --prefix=/SRX_INSTALL

   The configuration script checks if the system is ready for the software and reports possible errors. Most likely header files are missing. For example, the library *readline* also has a *readline-devel* which is not installed automatically with *readline*. In this case use yum install *readline-devel*. An easy way to find which package contains the header file is by querying yum.
   For instance, *yum provides */abc.h* returns a list of packages that include the specified file.

3. Compile and install the libraries

   make
   make install

   Normally no errors should be reported.

4. Now configure and test the installation.
   a. Prepare the configuration

      cd /SRX_INSTALL/etc
      cp srxcryptoapi.conf.sample srxcryptoapi.conf

   b. Edit the configuration file using vi or any other text editor. We use vi

      vi srxcryptoapi.conf

      Edit the following entries:

      Line 9: *key-volt = "/var/lib/bgpsec-keys"*
      into
      key-volt = "/SRX_FOLDER/bgpsec-io/data"

Line 42: *init_value = "PUB:/var/lib/bgpsec-keys/ski-list.txt;PRIV:var/lib/bgpsec-keys/ski-priv-list.txt"*
into
init_value = "PUB:/SRX_FOLDER/bgpsec-io/data/ski-list.txt;PRIV:/SRX_FOLDER/bgpsec-io/data /ski-priv-list.txt"

c. Perform the test
Switch into the install folder and call the tester

cd /SRX_INSTALL/sbin
./srx_crypto_tester

If all went well the printout should end in

API initialized!
done

So far errors we encountered were due to errors in the configuration.

Install BGPSEC-IO

1. Copy the code to a folder of your choice. We call it the SRX_FOLDER. Specify an installation folder. Let us call it SRX_INSTALL. You should have the following structure:

   /SRX_FOLDER/bgpsec-io

   Enter this folder

   cd SRX_FOLDER/bgpsec-io

2. Call the configuration script and specify the installation folder.

   ./configure --prefix=/SRX_INSTALL sca_dir=/SRX_INSTALL

   The configuration script checks if the system is ready for the software and reports possible errors. Most likely header files are missing. For example, the library *readline* also has a *readline-devel* which is not installed automatically with *readline*. In this case use yum install *readline-devel*. An easy way to find which package contains the header file is by querying yum.
   For instance, *yum provides */abc.h* returns a list of packages that include the specified file.

3. Compile and install the libraries

   make
   make install

   Normally no errors should be reported.

4. Now configure the installation.
   a. Prepare the generate the configuration file

      cd /SRX_INSTALL/bin
      ./bgpsecio -C bgpsecio.conf

      This will generate a default fully documented configuration script.

   b. Edit the bgpsecio.conf file
      Open the file bgpsecio.conf with your preferred editor and adjust the following entries:

      Line3: *ski_file = "/var/lib/key-volt/ski-list.txt";*
      Change to:
      ski_file = "/SRX_FOLDER/bgpsec-io/data/ski-list.txt";

Line4: *ski_key_loc = "/var/lib/key-volt/";*
Change to:
ski_file = "/SRX_FOLDER/bgpsec-io/data/";

This change allows us to test the installed keys and that the srx-crypto-api functions nicely
line9: *mode = "BGP";*
Change to:
mode = "CAPI";

This change allows to specify a different srx-crypto-api configuration file than the default one. Here we specify the default one to make sure it uses the config file we want it to use.
Line 20: *#capi_cfg = <configuration file>*
Change to:
capi_cfg = "/SRX_INSTALL/etc/srx/srxcryptoapi.conf"

Now if you want to run in BGP mode you need to configure the session starting with line 23. This version of bgpsec-io only supports one session but it is planned to support multiple sessions in the future.

Furthermore, the configuration file allows to script bgpsec traffic – see the update section. An update consists of the <prefix> < comma> (<as>[p<pCount>]?)*
In case no as list is provided, the prefix will be an origin announcement by bgpsec-io.

The disconnect value specifies the minimum up time after the last update is send. A zero "0" value disables this function and the session stays up forever or until it times out.

There are other interesting settings that are pretty much self-explanatory.

One important issue to the AS path, the ski_list contains the AS numbers bgpsec-io has keys for. In case an AS number is scripted where bgpsec-io does not have a private key for, it will either DROP this update, generate a FAKE update or generate a BGP4 update instead.

Also BGPSEC-IO can print the traffic it receives. The printout is in a human readable "pretty" format that follows the Wireshark GUI printout.

Also updates can be scripted either in the session or at the end globally. This makes more a difference once bgpsec-io supports multi session and at this time it does not really matter where updates are scripted.

The order in which updates are generated and played are:

i. Session
ii. Global
iii. Binary input
iv. Command line / piped input

5. Test the installation using the CAPI mode as configured above.
   Switch into the install folder and start bgpsec-io

   cd /SRX_INSTALL/bin
   ./bgpsecio -f bgpsecio.conf

   At this point it should have successfully validated 4 updates – only if the default data was used.
   Now to run it against a BGP router, modify the entry *mode="CAPI"* into *mode="BGP"* and restart bgpsecio.

Install SRx-Server

1. Copy the code to a folder of your choice. We call it the SRX_FOLDER. Specify an installation folder. Let us call it SRX_INSTALL. You should have the following structure:

   /SRX_FOLDER/srx-server

   Enter this folder

   cd SRX_FOLDER/srx-server

2. Call the configuration script and specify the installation folder.

   ./configure --prefix=/SRX_INSTALL sca_dir=/SRX_INSTALL

   The configuration script checks if the system is ready for the software and reports possible errors. Most likely header files are missing. For example, the library *readline* also has a *readline-devel* which is not installed automatically with *readline*. In this case use yum install *readline-devel*. An easy way to find which package contains the header file is by querying yum.
   For instance, *yum provides */abc.h* returns a list of packages that include the specified file.

3. Compile and install the libraries

   make
   make install

   Normally no errors should be reported.

4. Configure the server
   the configuration file of the server is located in /SRX_INSTALL/etc – But for now there is not much to configure. Have a look – it's pretty simple.

5. Start the server
   Switch into the install folder and start bgpsec-io

   cd /SRX_INSTALL/bin
   ./srx_server -f /SRX_INSTALL/etc/srx_server.conf

   To send it into the background add an "&" after the command
   ./srx_server -f /SRX_INSTALL/etc/srx_server.conf &

   To shut down the server at a later time, open a telnet session to port 17901 and type the command
   "shutdown x"

Install QuaggaSRx

6. Copy the code to a folder of your choice. We call it the SRX_FOLDER. Specify an installation folder. Let us call it SRX_INSTALL. You should have the following structure:

   /SRX_FOLDER/quagga-srx

   Enter this folder

   cd SRX_FOLDER/quagga-srx

7. Call the configuration script and specify the installation folder.

   ./configure --prefix=/SRX_INSTALL sca_dir=/SRX_INSTALL srx_dir=/SRX_INSTALL --disable-doc

   Now on a stock install of CentOS 6 and CentOS 7 we needed to install the **net-snmp-devel** as well as the **dia** package. The later one is problematic on CentOS 7 at this point because no rpm is yet published. At least not that I know of.
   To prevent issues with **dia** I recommend the configuration switch **--disable-doc** as shown in the configuration example above.

   The configuration script checks if the system is ready for the software and reports possible errors. Most likely header files are missing. For example, the library *readline* also has a *readline-devel* which is not installed automatically with *readline*. In this case use yum install *readline-devel*. An easy way to find which package contains the header file is by querying yum.
   For instance, *yum provides */abc.h* returns a list of packages that include the specified file.

8. Compile and install the libraries

   make
   make install

   Normally no errors should be reported.

9. Configure the router
   For configuration use the provided sample file located in /SRX_INSTALL/etx/bgpd.conf.samleSRx

   You need to modify at least the following lines:
   Line 13: router bgp <My ASN>
   Line 14: bgp router-id <Best is the ip of the router itself>

   Line 96: srx evaluation origin_only
   enable bgpsec path validation
   srx evaluation bgpsec

Line 292: network <prefix>
here repeat this line for each prefix this router will originate

Line 304: srx bgpsec 0 1 DEADBEEF…
remove the comment character '!'
replace DEADBEEF…. With the correct SKI. Look into the file ski-list.txt
remove the comment character '!'
the first number 0 specifies the key index, the second number the algorithm id

Line 309: srx bgpsec active 0
remove the comment character '!'

Line 314: neighbor 10.0.50.2 remote-as 32
Line 321: neighbor 10.0.50.2 bgpsec both
Here modify the IP addresses accordingly as well as the ASN of the peer.

10. Start the server
    The router needs to be started as root
    cd /SRX_INSTALL/sbin
    sudo ./bggpd -f /SRX_INSTALL/etc/bgpd.conf.sampleSRx -i /tmp/bgpd.pid

if the bgpsec-io is configured to print out received traffic one can create a topology where a bgpsec-io generator sits on both ends, one sends traffic the other doesn't – just listens.

Questions

For questions and bug reports, please send an email to bgpsrx-dev@antd.nist.gov