An open source implementation of SNBI & ACP with ODL Beryllium

Vijay Anand R vanandr@cisco.com
Toerless Eckert, Cisco Systems, eckert@cisco.com
IETF 95, Apr 4. 2016, Buenos Aires
From Inception to current status (1)

Ideally NOC equipment should be autonomic
  Use case: “stable connectivity” management from NOC

Open Daylight: Open Source NOC (controller) Open Source reference
  Inception: Lets make it autonomic
    ODL project: **Secure Network Bootstrap Infrastructure**
    Started with building AN Registrar code in Java (standard ODL dev. env.)

Challenge: ACP inside controller
  ODL uses OS-level transport (TCP/UDP). Building Java-level ACP (eg: Ipsec secure channels) and plug them underneath OS transport is challenging.
From Inception to current status (2)

Solution:

Linux Open Source Autonomic Router code – “SNBI-FE”

Packaged with Docker

- Makes experimentation/fast-deployment easy.
- For embedded platforms one would rather install only the SNBI software packages needed natively. Eg: OpenWrt (TBD)

SW Architecture

- “HOST package” part includes those components that are Linux specific
- Porting SNBI to other Oss possible by replacing HOST package
- Higher layers are OS independent
Current functionality: ODL Beryllium release

Docker:

• RA: Registrar - Karaf ODL package (stripped to only include what is needed for Registrar)
  • White List configuration.
  • Internal CA – Bouncy Castle.

• FE: Forwarding Element
  • SNBI Daemon
    • Neighbor Discovery.
    • Device Bootstrap with device domain Certificates.
    • Proxy bootstrap new devices.
    • Protocols: Not GRASP (yet), but those used in Cisco Autonomic Implementation

• HOST Package
  • Secure channels via IPSEC/GRE leveraging linux Kernel functionalities (no kernel changes).
  • Unstrung ipsec.
  • IPv6 Routing across the secure channels via RPL – unstrung (Michael Richardson).

• Linux Kernel Version - 4.4.3-040403-generic #201602251634 SMP
  Thu Feb 25 21:36:25 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux
  required for some channel details – IPv6 secure association via link-local address etc..
What can it do?

Full Autonomic Network

But very rough on the edges.

1. your “greenfield” devices have SNBI code

2. Configure Registrar (FirstFE)
   (eg: whitelist, domain-name)
   Registrar enrolls itself into autonomic

3. Plug together FirstFE, Dev1, Dev2

3.1 Dev1 enrolls via registrar,
    ACP FirstFE/Dev1 forms

3.2 Dev1 acts as enrollment proxy for Dev 2,
    Dev2 enrolls. ACP Dev 1 – Dev 2 forms.
Docker
Step-by-Step

1. Provision SNBI-RA interface

2. Registrar binds to SNBI-RA

3. Start daemon with registrar IP

4. First FE attempts to boot itself

5. Creates SNBI-FE interface and with a Domain Specific ULA address.

6. Hello Packets for Neighbor discovery

7. IPSEC/GRE tunnels to validated neighbors

8. Both the SNBI-FE ULA address and SNBI-RA address routes are advertised over the tunnels
References

• Contact:
  • snbi-dev@lists.opendaylight.org
• Tutorial:
  • https://wiki.opendaylight.org/view/SNBI_Beryllium:Tutorial

• SNBI Project Main Wiki
  https://wiki.opendaylight.org/view/SecureNetworkBootstrapping:Main
• Beryllium Release Plan
• Beryllium Release Review
• Documentation
  • Adoc Gerrit - https://git.opendaylight.org/gerrit/#/c/34063/