RPKI Deployathon

Philip Smith
NSRC

Summary & Findings
18th February 2020

RPKI Deployathon: Agenda

- Session 1:
  - Why Routing Security – Tashi Phuntsho
  - A look at ROAs – Tashi Phuntsho
- Session 2:
  - Deploying Validators
- Session 3:
  - Deploying RPKI on routers
- Session 4:
  - Deploying ROV and exploring interoperability

Helpers:
Tashi Phuntsho – APNIC
Aftab Siddiqui – ISOC
Mark Tinka – SEACOM
Warren Finch – APNIC
Taiji Kimura – JPNIC
Md Abdul Awal – NSRC

Deploying Validators

- 2 containers per group (of 4) for validator install
- Four well known validators chosen:
  - NLnetLabs Routinator
  - RIPE NCC Validator
  - Cloudflare OktoRPKI
  - FORT
- Unfortunately Dragon Labs validator no longer maintained ☹

Deploying Validators: Routinator

- Easy to install, even though participants had never worked with Rust before
- Routinator documentation was easy to follow
- APNIC Training guide was useful supplement to the Routinator documentation
- Routinator worked well, quick, and small memory footprint
- Easy to enable for Prometheus UI for monitoring
- Conclusion: the clear favourite, it just worked

Deploying Validators: RIPE NCC

- Initially participants followed the development version
  - Failed to set up as Linux version (Ubuntu 16.04) wasn't correct
  - No mention of what was actually required to set it up
- Following standard version (v3) was easy to install
  - Couldn't connect to router
  - No documentation explaining that dependencies required
  - Huge memory foot print (6Gbytes!)
- Once the RTR support was installed, connection to routers was easy
- Conclusion: set up and installation of validator was not straightforward – hard to convince people to use this one

Deploying Validators: OctoRPKI

- Installation was not straightforward at all
  - Even following the documentation
  - Instructions not clear
- Installing Docker version would have been easier
  - Again not clearly defined how and what to do
- Eventually participants used 3 different sets of instructions available online to install the validator
  - No mention that goRTR had to be installed as well
- OctoRPKI install was straightforward only by following the APNIC Training guide
- Conclusion: hard work

Deploying Validators: FORT

- Big problem with dependencies
  - Ubuntu 16.04 has too old version of OpenSSL
  - Participants had to compile up the version supported by FORT, which then broke other dependencies, etc.
  - No mention of the problem or solution in the install instructions
- No instructions about the ARIN TAL and how to install
  - Luckily APNIC Training guide covered how to do this
- Conclusion: dependency problem and ARIN TAL problem

Deploying RPKI on Routers

- Mix of real hardware and virtual environment
  - Cisco, Juniper, Nokia
    - Couldn’t get Cisco IOS XR virtual environment running; only Cisco IOS-XE available
  - Didn’t manage to get an Arista router
  - No one tried BIRD or FRR even though it was suggested to the participants
- Observations
  - Router talking to validator set up was easy, no issues noted

Router Implementation Observations

- Cisco IOS-XE seriously broken
  - Drops invalids automatically: workaround

```
bgp bestpath prefix-validate allow-invalid
```

  - Prefixes distributed by iBGP automatically marked Valid
    - No workaround until more recent IOS releases
  - If validator becomes unreachable, the RPKI table was flushed within 5 minutes, despite ROA lifetime
    - Not configurable
    - Only RPKI table refresh time is configurable

Router Implementation Observations

- Juniper:
  - Setting up to talk to validator well documented online
  - Keeps RPKI table for 3600sec (in case of losing connection to validator)
    - Can be configured
    - Life time is 6 hours in ROAs so the implementations should flush before then
  - Maintains state of the validation table across multiple routing engines
- Nokia
  - Easy to set up, good instructions
  - RPKI table kept for max 3600sec (in case of losing connection to validator)
    - How to set longer??
  - Maintains state of the validation table across multiple routing engines

Other Observations

- Propagating validation state:
  - Many say don't do this – keep it simple
  - But if we do want to, RFC8097 has this:

Extended Community	Meaning
0x4300:0:0	Valid
0x4300:0:1	NotFound
0x4300:0:2	Invalid

  - JunOS from 17.4R3, 18.2R3, 18.4R2 supports this
    - The MX204s we had came with 17.4R2.4 code, so didn't work, needing upgrade

Other Observations

- Difference between two validators
  - FORT and Cloudflare validators had different total VRPs
    - FORT missing around 1200
  - RIPE NCC and Routinator had the almost exact same total VRPs
    - That's a relief
- What does this mean in real life?
  - What does the router best path selection do?
  - (Cisco inserts validation before local-preference)
  - Untested, but we need to answer this

Diff FORT & OctoRPKI

```
root@group53:/tmp# diff -u octo.csv fort.csv
--- octo.csv    2020-02-17 06:14:50.303636011 +0000
+++ fort.csv    2020-02-17 06:13:48.901343682 +0000
@@ -11674,7 +11674,6 @@
 AS135134,2403:cfc0:100e::/48,48
 AS135134,2403:cfc0:100f::/48,48
 AS135134,2403:cfc0:1100::/44,48
-AS135134,2a0d:1a40:babe::/48,48
 AS135134,45.129.228.0/24,24
 AS135139,103.114.208.0/22,22
 AS135139,103.114.208.0/23,23
@@ -33377,12 +33376,10 @@
 AS202306,45.138.74.0/24,24
 AS202306,91.103.252.0/24,24
 AS202307,2a0b:b87:ffe9::/48,48
-AS202313,2a0d:1a40:fa0::/44,48
 AS202314,2a06:1e86::/32,48
 AS202314,2a0a:b707:1004::/48,48
 AS202314,2a0a:b707:1010::/44,48
 AS202314,2a0a:b707:1012::/48,48
-AS202314,2a0d:1a40:5550::/48,48
 AS202317,92.255.52.0/24,24
 AS202319,185.166.104.0/24,24
 AS202319,185.166.105.0/24,24
@@ -35119,7 +35116,6 @@
 AS204512,2a0e:9000::/32,32
 AS204521,185.168.216.0/24,24
 AS204526,2001:678:a10::/48,48
-AS204526,2a0d:1a44::/32,48
 AS204526,2a0e:fd44::/32,48
 AS204528,178.175.235.0/24,24
 AS204529,185.114.218.0/24,24
@@ -38901,7 +38897,6 @@
 AS207948,2001:7f8:e3::/48,48
 AS20795,193.109.96.0/22,22
 AS207955,2a0e:46c6:300::/40,48
-AS207960,2a0d:1a40:7900::/40,48
 AS207963,2a0f:5707:ad00::/44,48
 AS207963,2a0f:5707:ad01::/48,48
 AS207967,45.87.244.0/22,22
@@ -39406,7 +39401,6 @@
 AS208481,45.176.188.0/22,22
 AS208481,45.8.172.0/22,24
 AS208483,2a09:be40:3000::/40,48
-AS208483,2a0d:1a40:666::/48,48
 AS208485,160.19.94.0/24,24
 AS208485,160.19.95.0/24,24
 etc
```

Other Observations

- Cisco IOS/IOS-XE behaviour – example:
  - Prefix learned via two paths via two separate EBGP speaking routers
  - Prefix and validation state distributed by IBGP to core router (route reflector):

```
     Network          Next Hop      Metric LocPrf Weight Path
V*>i 61.45.249.0/24   100.68.1.1         0     50      0 121 20 135534 i
N* i                  100.68.1.3         0    200      0 20 135534 i
V*>i 61.45.250.0/24   100.68.1.1         0     50      0 121 30 135535 i
N* i                  100.68.1.3         0    150      0 30 135535 i
V*>i 61.45.251.0/24   100.68.1.1         0     50      0 121 122 40 135536 i
N* i                  100.68.1.3         0    150      0 40 135536 i
```

  - One EBGP speaking router talks with validator
  - The other EBGP speaking router does not (due to error or design)
  - Core router best path selection prefers *valid* path over *not found* even if the latter has higher local preference

Conclusion

- Situation with validators better than September 2019
  - Thanks RIPE NCC for improving docs – but install process still not simple and needs work
  - Dragon Labs validator, anyone?
  - Differences in VRPs is worrying
- Cisco IOS-XE default behaviour remains a serious worry
  - Advice: turn off the defaults if possible, and lobby Cisco to fix this serious problem
- Untested
  - Issues with path selection?
  - Validator deployment best practices?