Route Origin Validation Lab

Part-1: Installing RPKI Validator (RIPEv3)

VM Details

```
[192.168.30.13]
......
[192.168.30.20]
```

Login Details

- Username `apnic` and password `training`.

Preinstalled packages

To save time, the following essential packages have been preinstalled on the containers:

- `GCC (GNU C toolchain)`
- `rsync`

Lab Setup

For this lab, we will use RPKI Validator 3 from the RIPE NCC as the RPKI validator.

1. Login to your server (SSH from the jumphost to your container using the `username` and `password` given above), where `X` is your VM number:

   ```
   ssh apnic@192.168.30.X
   ```

2. Update the repository

   ```
   sudo apt update && sudo apt upgrade
   ```

3. Install dependencies (RIPE's Validator requires `OpenJDK 8` or higher, and `rsync`):

   - For Java we will go with headless OpenJDK-11:

```
sudo apt install -y openjdk-11-jre-headless
```

- Verify the installed Java version

```
java -version
```

```
apnic@group13:~$ java -version
openjdk version "11.0.6" 2020-01-14
OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1)
OpenJDK 64-Bit Server VM (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1, mixed mode, sharing)
```

4. Download the latest version of the validator and extract where you want it:

```
wget https://ftp.ripe.net/tools/rpki/validator3/prod/generic/rpki-validator-3.1-2020.08.06.14.39-dist.tar.gz

mkdir rpki-validator-3.1

tar xf rpki-validator-3.1-2020.08.06.14.39-dist.tar.gz -C rpki-validator-3.1 --strip-components 1

cd rpki-validator-3.1
```

5. The validator comes with the TALs for each RIR (found in the `preconfigured-tals` directory) except ARIN. To add ARIN's TAL, you can download it from here and move it to the `preconfigured-tals` directory as shown below:

- **NOTE:** By downloading ARIN's TAL, you agree to be bound by ARIN's Relying Party Agreement (RPA):

```
wget https://www.arin.net/resources/manage/rpki/arin-ripevalidator.tal

mv arin-ripevalidator.tal preconfigured-tals/
```

- Now you have TALs from each RIR:

```
ls  preconfigured-tals/
```

```
afrinic.tal  apnic.tal  arin-ripevalidator.tal  lacnic.tal  ripe-ncc.tal
```

6. Run the validator:

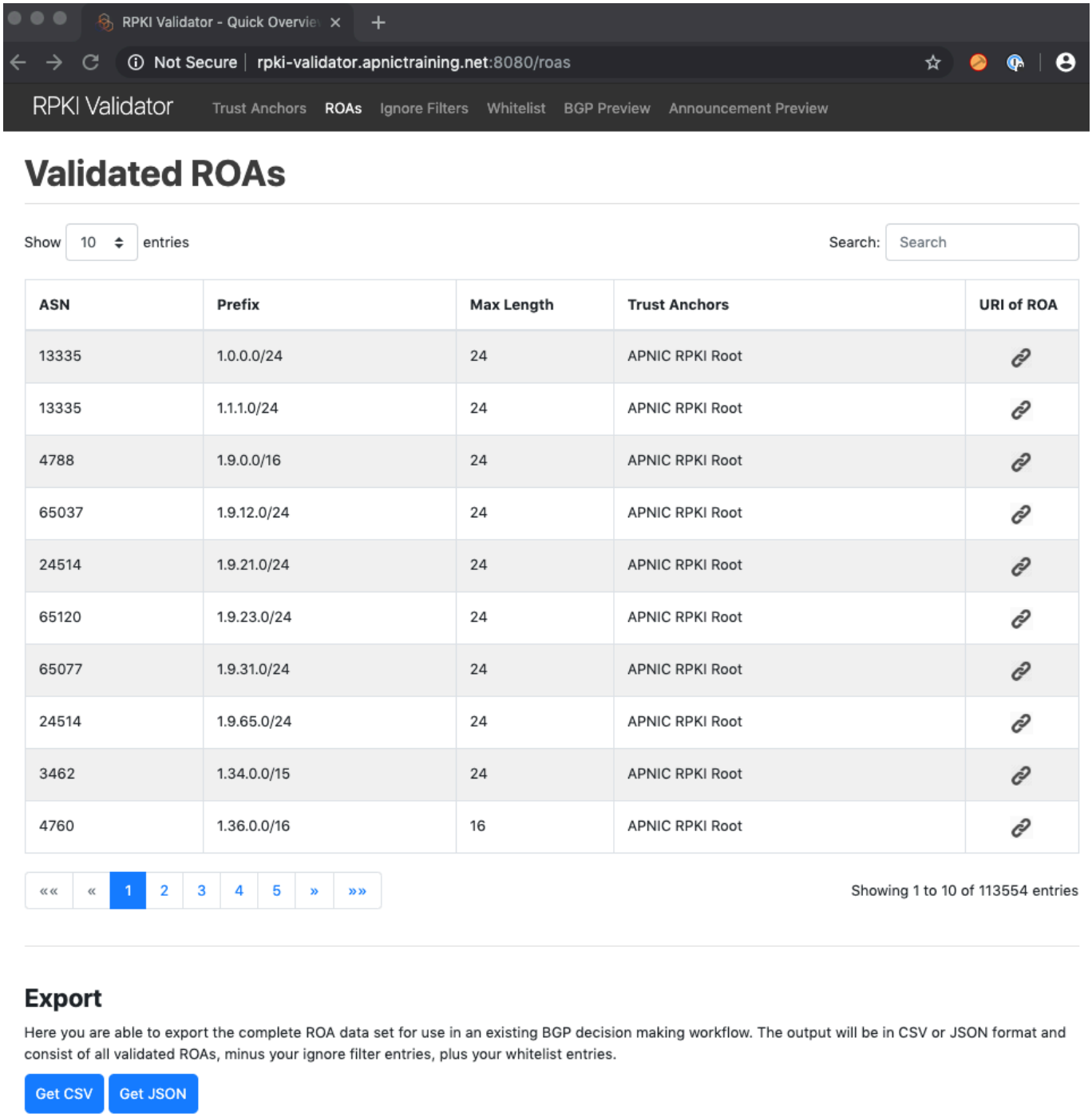
```
sudo nohup ./rpki-validator-3.sh > out 2> err &
```

- You monitor the status/progress: `tail -f out` (use `Ctrl + C` to exit)

7. Use the following command to retrieve the validated ROA payloads (produces a list of ASNs and prefixes). If this command produces no output, then the validator is still working through the initial synchronisation process, which generally takes a few minutes. By default, the server will resynchronise its state every 10 minutes.

```
curl -H "Accept: text/csv" localhost:8080/api/export.csv
```

   - You can also access it through the web interface (`http://<validator-name/validator-address>:8080/roas`)



   - You can also write the VRP output to a file (for you to compare the outputs with other validators):

```
curl -H "Accept: text/csv" localhost:8080/api/export.csv > ripe.csv
```

8. **[Optional]** For you to compare the VRPs, you would need to sort and normalise the RIPEv3.1 output.
    - As you can see, first get ride of the Trust Anchor column:

      ```
      cut -d, -f4 --complement ripe.csv > ripe_trimmed.csv
      ```

      - Have a look at the trimmped output: `more ripe_trimmed.csv`
    - Now we need to get rid of the double quotes ("") as well as adding `AS` to the ASN list (to make it similar to other validator outputs):

      ```
      sed "s/\"//g" ripe_trimmed.csv | sed 's/^/AS/' > ripe_normalised.csv
      ```

      - Have a look at the `ripe_normalised.csv` file
    - Let us now sort the normalised output with a basic `sort` command:

      ```
      sort ripe_normalised.csv > ripe_sorted.csv
      ```

      - Have a look at the sorted output: `more ripe_sorted.csv`
    - Now your routinator VRP file is ready to be compared with other validator outputs. Example below to compare RIPEv3.1 and Routinator outputs:

      ```
      diff -u ripe_sorted.csv rout_sorted.csv
      ```

      OR

      ```
      diff -y ripe_sorted.csv rout_sorted.csv
      ```

Now your validator is ready to feed the validated cache to BGP speaking routers through the RTR (RPKI-to-Router) protocol.

Part-2: RTR session

Validator side

The rpki-rtr server component of the RIPE validator allows RPKI-enabled routers to connect to it and fetch the validated cache (ROA cache). By default, the server listens for rpki-rtr requests on port `8323`.

1. Download and extract the rpki-rtr server:

```
wget https://ftp.ripe.net/tools/rpki/validator3/prod/generic/rpki-rtr-server-latest-dist.tar.gz

mkdir rpki-rtr-server

tar xf rpki-rtr-server-latest-dist.tar.gz -C rpki-rtr-server --strip-components 1

cd rpki-rtr-server
```

2. Set the address for the rpki-rtr server by editing the `conf/application.properties` file:
    - For example, `rtr.server.address=::` to listen on all interfaces
3. Run the rpki-rtr server:

```
nohup ./rpki-rtr-server.sh > out 2> err &
```

NOTE: Now you can configure your BGP speaking routers to talk to the rpki-rtr server to fetch the validated cache (ROA cache).