[General Info on RPKI](#)

[Route Origin Validation (ROV)](#)
- [Validators:](#)
- [Known Issues in RPKI Code](#)
- [Implementation Considerations](#)
- [RPI Validation Monitoring](#)
- [Example Configs for RPKI Validation](#)

[Route Origin Authorization (ROA)](#)
- [Creating Hosted ROAs in ARIN](#)
- [Generating ARIN Hosted ROAs with the API](#)
- [Krill Delegated CA 🦐](#)
- [Setting up an ARIN ORG-ID to a Krill delegated CA](#)
- [Testing Out a Delegated CA with ARIN's OT&E](#)
- [Legal aspects of RPKI](#)

General Info on RPKI

Resource Public Key Infrastructure (RPKI) is a solution for the prevention of BGP Origin hijack attacks, both accidental and malicious. It does not fix all problems with BGP security but is a critical step in doing so. It doesn't fix the problem of route leaks or BGP Path attacks. Please see the NANOG presentation https://www.youtube.com/watch?v=0Fi2ghCnXi0 for more info. For information on BGP origin hijacks that RPKI could have helped with, please see these URLs: https://www.manrs.org/category/routing-security-incidents/ https://blogs.oracle.com/internetintelligence/

- General RPKI information:
  - Great source of info on validators, delegated CAs and RPKI in general: https://rpki.readthedocs.io
    - rpki.readthedocs.io also includes a community driven FAQ covering everything from common technical questions to legal aspects of RPKI.
  - Great high level presentation on RPKI by Job (pronounced Yōb) Snijders: https://www.brighttalk.com/webcast/5648/396013/rpki-101-with-job-snijders

    - Job Snijders has a closed email list for large scale RPKI deployers. If you are ACTIVELY working on deploying RPKI in a large service provider, please request to subscribe: https://puck.nether.net/mailman/listinfo/rpki-deployers
        - Tip, there's some good info in the archives!
    - Discord RPKI channel with a lot of great discussion. Good place to get questions answered quickly: https://discord.gg/8dvKB5Ykhy
    - RIPE has created some routing beacons (prefixes advertised intentionally as invalid, valid, unknown) for testing. The prefixes are found at the bottom of this page: https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/current-ris-routing-beacons
    - RPKI Technical Analysis by ICANN https://www.icann.org/en/system/files/files/octo-014-02sep20-en.pdf
    - Hosted CA vs. Delegated CA RIPE presentation: https://www.ripe.net/participate/meetings/open-house/ripe-ncc-open-house-hosted-vs-delegated-rpki
    - Juniper Day One free book on Routing Security https://www.juniper.net/documentation/en_US/day-one-books/DO_BGP_SecureRouting2.0.pdf
- Deployment status tracking:
    - Monitor the deployment of RPKI in each RIR and globally: https://rpki-monitor.antd.nist.gov
    - Cloudflare's site to test if your ISP is doing ROV facing CF: https://isbgpsafeyet.com
    - Cloudflare's RPKI explorer site: https://rpki.cloudflare.com/
    - NLnet Lab's RPKI explorer site: https://jdr.nlnetlabs.nl
    - RIPE/NTT Origin Validation test https://ripe.net/s/rpki-test
    - Looking Glass to show RPKI state: https://stat.ripe.net/widget/routing-status
    - Looking Glass with RPKI state: http://lg.ring.NLNOG.net/
    - Public deployment tracker: http://rpki.exposed/ (anonymous editing, no verification)
    - List of ASNs doing RPKI: https://github.com/cloudflare/isbgpsafeyet.com/blob/master/data/operators.csv
    - Job's RPKI Console site: http://console.rpki-client.org
    - BGP Stuff: https://bgpstuff.net/
        - FYI, you can do "curl https://bgpstuff.net/invalids/<ASN>" to find out if an ASN is announcing any invalids.
    - Info on recent changes in the ROAs: https://rpki.today
    - MANRS ROA Stats https://roa-stats.manrs.org
    -
- Standards docs:
    - RFC 6811 - BGP Prefix Origin Validation: https://tools.ietf.org/html/rfc6811

    - RFC 6480 - An Infrastructure to Support Secure Internet Routing: https://tools.ietf.org/html/rfc6480
    - A list of all RFCs related to RPKI and how they relate to each other: https://rpki-rfc.routingsecurity.net

FYI, implementation of Route Origin Validation (ROV) and Route Origin Authorizations (ROAs) can be done concurrently and are independent of one another. Most Internet Service Providers (ISPs) and other network operators start with ROV as a first step, as it is seen as an easier first deployment step.

Route Origin Validation (ROV)

Validators:

There are two distinct functions to performing ROV: 1) fetching and validating ROAs to build a validated cache of authorized routes, and 2) serving the validated cache to routers over the RPKI-To-Router (RTR) protocol. Some validators only perform fetching and validation, some have all functions integrated into a single binary, and some have split them into separate binaries.

Validators on the ISP's network connect to repositories via RPKI Repository Delta Protocol (RRDP) over HTTPS and RSYNC (RSYNC should be deprecated in a few years).
Routers performing ROV connect to validators over RPKI-To-Router (RTR) protocol[1] (TCP/323 by default). The RTR protocol is unencrypted and can be tunneled through SSH on the routers. The RFC also allows tunneling through TLS but no router vendors support this right now.

Most common validators being used are:

- FORT (RRDP & Rsync fetching and validation + RTR server) from NIC.mx and LACNIC, written in C
- Routinator (RRDP & Rsync fetching and validation + RTR server) from NLnet Labs, written in Rust
    - RTRTR (used for distributing Routinator validation database to multiple PoPs, DCs, etc.) written in Rust
- OpenBSD's rpki-client (Rsync fetching and validation), packages for many operating systems, what NTT uses, written in C
- OctoRPKI (RRDP & Rsync fetching and validation) and GoRTR (RTR server) from CloudFlare, written in Go
- ~~RIPE NCC Validator 3 (RRDP & Rsync fetching and validation in one binary, RTR server in another) written in Java~~
    - **RIPE is stopping development of Validator3! Please switch to another validator if you are running it.**

[1] https://blog.cloudflare.com/rpki-and-the-rtr-protocol/

https://labs.ripe.net/Members/nathalie_nathalie/life-cycle-of-the-ripe-ncc-rpki-validator-1

All are open source and free which means community support. NLnet Labs (https://nlnetlabs.nl/projects/rpki/support/) and Zones (https://www.zones.com/) can provide commercial support on Routinator and Krill if required.

Job Snijders believes that if the JSON located here: http://bgpfilterguide.nlnog.net/guides/slurm_ta/ is placed in the SLURM file on the validators, that it could help in the case that an attacker is able to compromise a RIR's CA and hijack the RIR's IP prefixes. Others believe that the risk of the data in the JSON becoming stale and causing an outage outweighs the risk of the RIR CA compromise/hijack.

Many ISPs are initially deploying three or four validators and making them geographically diverse (East Routinator/FORT validator and West Routinator/FORT validator)

- Reason for this is that if there is an issue with one software, there is still another platform that will continue functioning <— doesn't hold true with various types of bugs… :(
- Some providers are using different OS's for each validator (eg. Centos for one, Ubuntu for another, FreeBSD for a third)

Known Issues in RPKI Code

Known RPKI issues in router code:

- Juniper Junos bug: "The rpd scheduler slips might be seen on an RPKI route validation-enabled BGP peering router in a scaled setup." https://prsearch.juniper.net/InfoCenter/index?page=prcontent&id=PR1461602
- Junos opens up TCP port 2222 on the RE as soon as a validator is configured!
- Cisco IOS-XR bug: CSCvp82287: "After RPKI session re-establishes the RTR protocol does not go into RESET state, ROA not downloaded" https://bst.cloudapps.cisco.com/bugsearch/bug/CSCvp82287
- Cisco IOS-XR bug: "RPKI purge timer is not being used when deleting the ROA table, on RPKI session teardown due to dropped route" https://bst.cloudapps.cisco.com/bugsearch/bug/CSCvw01872
- Cisco IOS-XR prior to 6.6.3 didn't have the ability to specify a source interface for the RTR traffic
    - There is a "hack" that can be done to tunnel the RTR traffic through SSH. With this you can define the source interface for the SSH session and thus get around the limitation.
- Juniper "VRP flip flap" issue: "BGP RPKI ROA withdrawal might lead to unexpected BGP route flapping" https://prsearch.juniper.net/InfoCenter/index?page=prcontent&id=PR1483097
- Cisco IOS-XR bug: "BGP RPKI server password is not saved in the configuration" https://bst.cloudapps.cisco.com/bugsearch/bug/CSCvw38010

- Cisco IOS-XR bug: “BGP with RPKI are not being established”
  https://bst.cloudapps.cisco.com/bugsearch/bug/CSCvw43883
- Cisco IOS-XR NCS bug: “NCS5508/6.6.3/RPKI "show bgp rpki table" is showing a wrong negative ASN "-80847294"
  https://bst.cloudapps.cisco.com/bugsearch/bug/CSCvw65184
- Juniper bug: “Rpd scheduler slips might be seen on RPKI route validation enabled BGP peering router in a scaled setup.”
  https://prsearch.juniper.net/InfoCenter/index?page=prcontent&id=PR1461602
    - This bug should only affect very large (tier 1) ISPs.
- Juniper bug: “Specific packets can trigger rpd crash when BGP Origin Validation is configured with RPKI (CVE-2021-0281)”
  https://kb.juniper.net/InfoCenter/index?page=content&id=JSA11185&actp=METADATA

Known issues in RPKI Validator code:

- Routinator: CVE-2020-17366 (manifest handling issue, fixed in 0.8.0)
    - Routinator 0.8.0 has been released, which now includes validation according to draft-ietf-sidrops-6486bis
- Routinator 0.9.0 has a memory consumption issue. It gradually increases memory consumption until it gets killed by the kernel
- RIPE NCC: CVE-2020-16164 (missing .roa files can result in marking BGP routes invalid)
- RIPE NCC: CVE-2020-16162 (missing validation checks on X.509 CRLs)
- RIPE NCC: CVE-2020-16163 (now fixed)
- OctoRPKI: Same issue as CVE-2020-17366 defined for Rouinator

Implementation Considerations

Many ISPs are either implementing a policy to tag invalids with a BGP community and then later change to reject after a few weeks of monitoring; or slowly implementing the rejection of invalids on routers over the course of a couple of months; or run a simulation using traffic analysis and deploy in 1 day.

A lot of documentation online shows example RPKI configs that lower the local pref on unknowns. No ISPs that we know of have implemented this! All ISPs we know just reject invalid route announcements in EBGP-IN/EBGP-OUT, and filter “not-found” and “valid” routes using standard IRR, and AS_PATH filters such as Peerlock. RPKI Origin Validation is *in addition* to existing Best Current Practises (see https://www.manrs.org/ for more info).

Job Snijders has created this tool to show what prefixes with what AS Origins are impacted by the RFC 6811 Origin Validation procedure. https://github.com/job/rpki-ov-checker

Other service providers are also using http://www.pmacct.net/ to collect netflow and BMP info to see how much traffic would be impacted by ROV. Kentik also has support for RPKI analysis https://seclists.org/nanog/2019/Mar/204

NTT has developed a tool called BGPalerter (https://github.com/nttgin/BGPalerter) which will generate an alert if you are advertising invalids or if you don't have ROAs for all prefixes advertised from an ASN. It also does BGP hijack detection.

Not very many ISPs have shown interest in implementing the RPKI extended community to signal RPKI state via iBGP as defined in RFC 8097 (https://tools.ietf.org/html/rfc8097)

FYI, if you have a route policy to accept more specifics (you accept /32 or /128 RTBH routes from your customers or you accept longer than a /24 from a customer) you will need to put this part of the policy above the ROV policy **or else it may be dropped by the ROV policy!**

RPKI Validation Monitoring

There are not a lot of good monitoring messages that are generated by the routers themselves. The routers will send a message when a session to a validator goes down but not much else. Many operators are using "screen scraping" scripts that log into the routers performing ROV and execute a series of commands and then save the output of those commands and perform some action if the delta has changed significantly.   For example, get the number of invalids/valids/unknown with "show validation statistics" on Junos. If the number of invalids has increased by 5% from the last collection, then generate an alert.
Much more info can usually be retrieved from the validators themselves. For example, Routinator has a Prometheus endpoint (https://rpki.readthedocs.io/en/latest/routinator/monitoring.html) which can be used to generate Grafana graphs and alerts: https://grafana.com/grafana/dashboards/11922

Example Configs for RPKI Validation

A good resource for implementing RPKI on a Juniper can be found here:
https://www.juniper.net/documentation/en_US/day-one-books/DO_BGP_SecureRouting2.0.pdf

JunOS Example Config:

```
# block TCP port 2222 that gets opened up on the RE when adding a
validator!

set firewall filter <IPv4 RE Filter> term RPKI-VALIDATOR from protocol tcp

set firewall filter <IPv4 RE Filter> term RPKI-VALIDATOR from port 2222

set firewall filter <IPv4 RE Filter> term RPKI-VALIDATOR then discard

set firewall family inet6 filter <IPv6 RE Filter> term RPKI-VALIDATOR from
next-header tcp
```

```
set firewall family inet6 filter <IPv6 RE Filter> term RPKI-VALIDATOR from
port 2222

set firewall family inet6 filter <IPv6 RE Filter> term RPKI-VALIDATOR then
discard


# By default Junos will only maintain two sessions to the validators

set routing-options validation group rpki-validator max-sessions <number
of validators to maintain connections to.  Ex. 4>

# Repeat these command for each validator IP

set routing-options validation group rpki-validator session <IP address of
RPKI Validator> port <323 is the default port>

# Configure a liveliness check interval for a configured resource public
key infrastructure (RPKI) cache server. After every period of time
configured on the refresh-time statement (in seconds), a serial query
protocol data unit (PDU) with the last known serial number is transmitted.

# This should be 600 or more. This is to prevent too many ROUTE-REFRESH
messages being sent to neighbors.

set routing-options validation group rpki-validator session <IP address of
RPKI Validator> refresh-time 600

# The length of time in seconds the session will be considered alive
without any activity.  Must be >= 2x the refresh time!

set routing-options validation group rpki-validator session <IP address of
RPKI Validator> hold-time 1200

# Route Validation (RV) records learned from a cache are valid while the
session to that cache remains up, and for up to "record-lifetime" seconds
thereafter.  RV records are expired when the session to the cache has gone
down, and remained down for "record-lifetime" seconds.

set routing-options validation group rpki-validator session <IP address of
RPKI Validator> record-lifetime 3600

set routing-options validation group rpki-validator session <IP address of
RPKI Validator> local-address <Loopback0 IP or whatever you specify>


# probably need to do this for your IPv4 and IPv6 import policies

set policy-options policy-statement <Import Policy Name> term VALID from
protocol bgp
```

```
set policy-options policy-statement <Import Policy Name> term VALID from
validation-database valid

set policy-options policy-statement <Import Policy Name> term VALID then
validation-state valid

set policy-options policy-statement <Import Policy Name> term VALID then
next term

set policy-options policy-statement <Import Policy Name> term INVALID from
protocol bgp

set policy-options policy-statement <Import Policy Name> term INVALID from
validation-database invalid

# Uncomment to set local pref to 50 and remove reject to accept invalids

set policy-options policy-statement <Import Policy Name> term INVALID then
reject

# set policy-options policy-statement <Import Policy Name> term INVALID
then local-preference 50

set policy-options policy-statement <Import Policy Name> term INVALID then
validation-state invalid

set policy-options policy-statement <Import Policy Name> term INVALID then
next term


# Add these lines below if you want to apply the extended community as
defined in RFC 8097 and send validation state to neighbors via iBGP

set policy-options policy-statement <Import Policy Name> term UNKNOWN then
community add origin-validation-state-unknown

set policy-options policy-statement <Import Policy Name> term INVALID then
community add origin-validation-state-invalid

set policy-options policy-statement <Import Policy Name> term VALID then
community add origin-validation-state-valid

set policy-options community origin-validation-state-invalid members
0x4300:0.0.0.0:2

set policy-options community origin-validation-state-unknown members
0x4300:0.0.0.0:1

set policy-options community origin-validation-state-valid members
0x4300:0.0.0.0:0
```

Commands to validate that Junos is doing RPKI:

```
# Shows RTR connections to validators
```

```
show validation session

# Shows stats of database copied over from validators
show validation statistics

# Shows what routes are invalid
show route validation-state invalid

# shows the list of prefixes copied over from validators
show validation replication database

# This is Cloudflare's intentionally invalid prefix
show route 103.21.244.0/24

# Another Cloudflare prefix that should always be valid
show route 1.1.1.0/24
```

Cisco Example IOS-XR Config:

```
router bgp <ASN>

rpki server <IP Address of RPKI Validator>

transport tcp port <323 is default>

# Configures the time BGP waits in between sending periodic serial queries
to the cache. Set refresh-time in seconds. Range for the refresh time is
15 to 3600 seconds.

# This should be 600 or more. This is to prevent too many ROUTE-REFRESH
messages being sent to neighbors.

refresh-time 600

# Configures the time BGP waits for a response after sending a serial or
reset query. Set response-time in seconds. Range for the response time is
15 to 3600 seconds.

# If you have a FW between your validators and your routers, setting this
too high can cause the session to timeout on the firewall

response-time 600

# Configures the time BGP waits to keep routes from a cache after the
cache session drops. Range for the purge time is 30 to 360 seconds.

purge-time 360
```

```
!! this command is only available in IOS-XR 6.6.3 or later!!

bind-source interface Loopback 0

bgp origin-as validation time 5

address-family ipv4 unicast

# Add this line if you want to apply the extended community as defined in
RFC 8097 and send validation state to neighbors via iBGP

bgp origin-as validation signal ibgp

# This command is needed in +6.5.1

bgp origin-as validation enable

address-family ipv6 unicast

# Add this line if you want to apply the extended community as defined in
RFC 8097 and send validation state to neighbors via iBGP

bgp origin-as validation signal ibgp

# This command is needed in +6.5.1

bgp origin-as validation enable


route-policy <Peer Import Policy>
if validation-state is invalid then
## if you only want to depref first, then uncomment this next line out and
change the "drop" to a "pass"
# set local-preference 50
  drop
endif

<insert the rest of policy>

end-policy
```

Commands to validate that IOS-XR is doing RPKI:

```
# General overview of RTR sessions and database copied from validators
show bgp rpki summary

# This is Cloudflare’s intentionally invalid prefix
show bgp ipv4 unicast 103.21.244.0/24

# Another Cloudflare prefix that should always be valid
show bgp ipv4 unicast 1.1.1.0/24
```

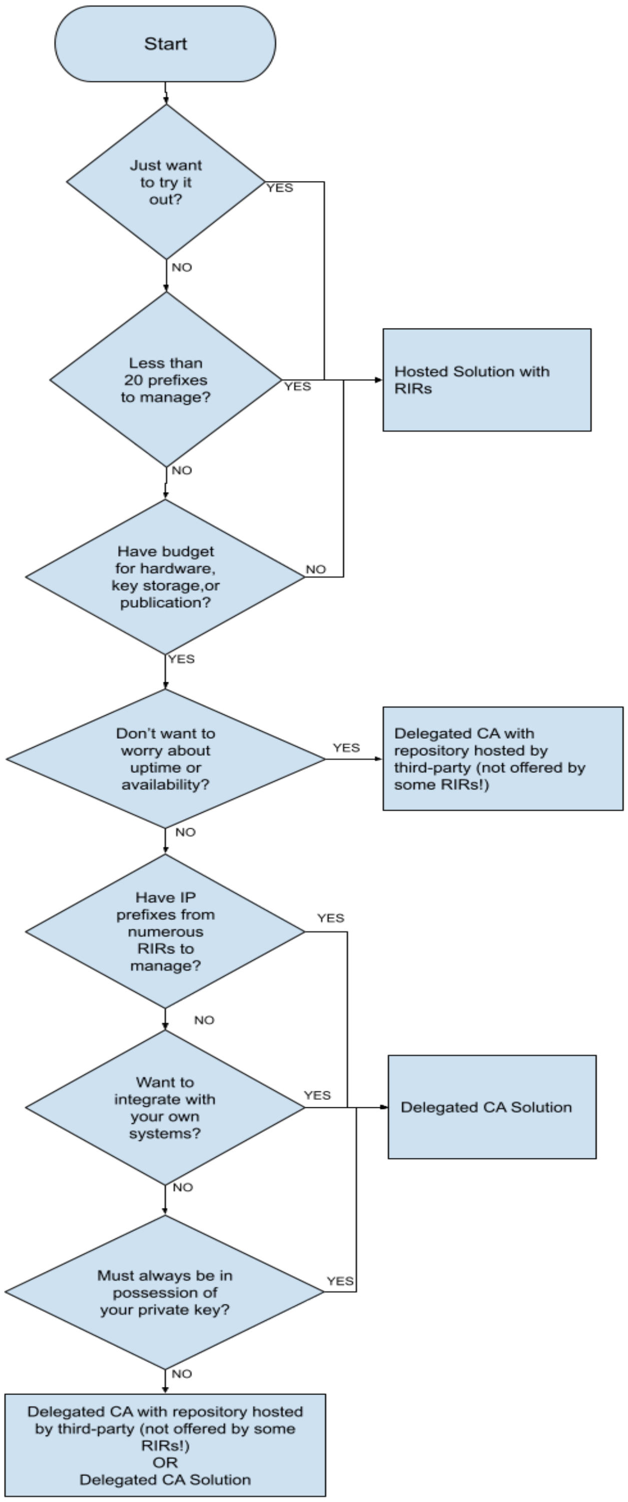
```
# Shows all the RIB entries that have been defined as
valid/invalid/unknown
show bgp origin-as validity [valid|invalid|not-found]
```

Route Origin Authorization (ROA)

(acronym is usually pronounced ROW-ah)

- Very important to first identify what your “source of truth” will be for the creation of the ROAs
    - Could be IPAM, BGP, BMP
- There can be multiple prefixes in one ROA but only one ASN!
    - Most ISPs are only putting one prefix in a ROA. If multiple prefixes are put into a ROA then if one of the prefixes needs to change the origin AS, then the entire ROA needs to be deleted and recreated.
- There is an optional field in the ROA called maxLength which can be used to reduce the number of ROAs that are needed. Most ISPs are either setting the maxLength value to the same as the prefix mask or not using that field. If you set the maxLength too large (ex. /24 or /48) you open yourself up to a potential forged-origin subprefix hijack (see this IETF doc: https://tools.ietf.org/html/draft-ietf-sidrops-rpkimaxlen-06)
- If you have a DDoS service provider that uses BGP to divert your traffic for scrubbing, you will need to create ROAs to allow them to originate your prefixes from their ASN!
- 3 options for ROA creation/repository:
    - Hosted CA with your RIR (easiest)
        - PROS:
            - Very easy for a few prefixes
            - It’s FREE!
            - ARIN and others have an API for generating ROAs
            - Does transfer the burden of maintaining CA/repository to RIR
        - CONS:
            - API has limited functionality
            - Certificates do not auto-renew
            - User interface is cumbersome
            - Doesn’t provide for suggestions for easy ROA creation
            - Requires ORG ID match with the prefix
            - Easily susceptible to fat fingering or incorrectly pasting data that can cause RPKI INVALID route announcements
            - RIRs have had some issues keeping the repositories available
    - Delegated CA (ex. Krill, see section below)
        - PROS:
            - Fully functional API
            - Better integration with your Infrastructure (eg. integrate with IPAM and monitoring)
            - Better security (private key stays in your control)

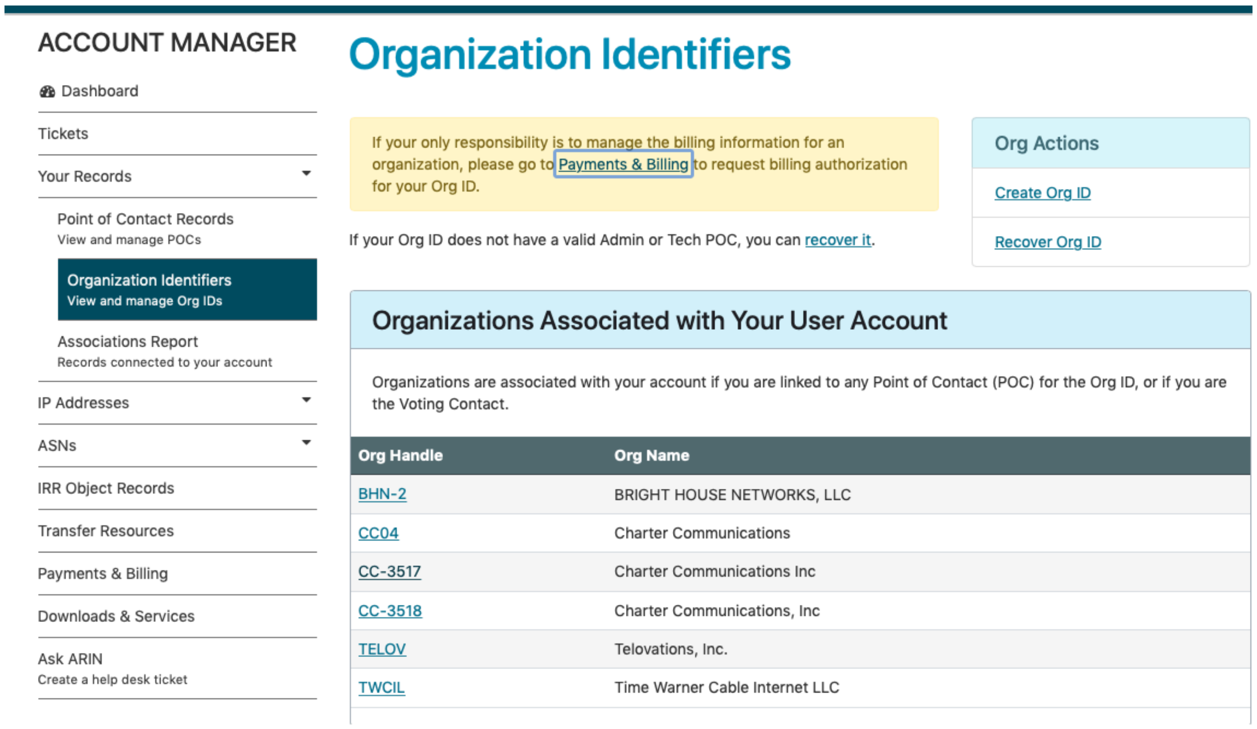
                - Auto-renew of certificates
                - Shows how a potential ROA will affect your advertisement before creation
                - Lets you to manage resources from multiple RIRs in a single instance
                - Lets you to delegate a subset of your resources to a different business unit or a customer, so that they can manage ROAs themselves
            - CONS:
                - You are responsible for making sure the system is up and available
                - Costs money, time, manpower
                - There is only ONE delegated CA implementation, there are no alternatives to Krill that are actively maintained. (Think BIND in early years)
                    - There is the RPKI Toolkit from Dragon Research Labs which is used by TWNIC, CNNIC and JPNIC to run their CAs. As far as is known it's currently not actively maintained.
                -
        - Hybrid Model (Delegated CA with the repository run by your RIR)
            - Offered by APNIC, will be offered in 2022 by ARIN and RIPE. LACNIC will offer this option in 2022.
            - PROS:
                - Same as Delegated CA above
                - The RIR is responsible for making sure the repository is up and available
            - CONS:
                - RIRs have had some issues keeping the repositories up
- Flow chart to help you decide which option to select:
  Flow Chart Should I run my own RPKI CA?


Start
Just want to try it out?
YES
NO
Less than 20 prefixes to manage?
YES
Hosted Solution with RIRs
NO
Have budget for hardware, key storage,or publication?
NO
YES
Don't want to worry about uptime or availability?
YES
Delegated CA with repository hosted by third-party (not offered by some RIRs!)
NO
Have IP prefixes from numerous RIRs to manage?
YES
NO
Want to integrate with your own systems?
YES
Delegated CA Solution
NO
Must always be in possession of your private key?
YES
NO
Delegated CA with repository hosted by third-party (not offered by some RIRs!) OR Delegated CA Solution

Creating Hosted ROAs in ARIN

NOTE, you should probably do the ROA creation process during a maintenance window. The reason for this is that if you create an invalid ROA, the prefixes specified in the ROA will be dropped by those providers doing ROV! It can take up to an hour after you remove the INVALID ROA from the repository for it to be propagated across the Internet.
This procedure is documented at https://www.arin.net/resources/manage/rpki/roa_request/
Here's the procedure to quickly create a ROA in ARIN using your browser:

- Create your public/private key pair using OpenSSL:
    - `openssl genrsa -out org keypair.pem 2048`
        - This command generates a ROA Request Generation Key Pair and saves it as a file named orgkeypair.pem.
    - `openssl rsa -in orgkeypair.pem -pubout -outform PEM -out org_pubkey.pem`
        - This command extracts the public key from the ROA Request Generation key pair and writes it to a file named org_pubkey.pem.
    - Keep the orgkeypair.pem file private, perhaps in an HSM. If the security of the private key is compromised, you should delete all of the ROAs created with that key and generate a new key pair and new ROAs.
- Log in to ARIN Online and select Your Records > Organization Identifiers from the navigation menu.



Organization Identifiers

If your only responsibility is to manage the billing information for an organization, please go to Payments & Billing to request billing authorization for your Org ID.

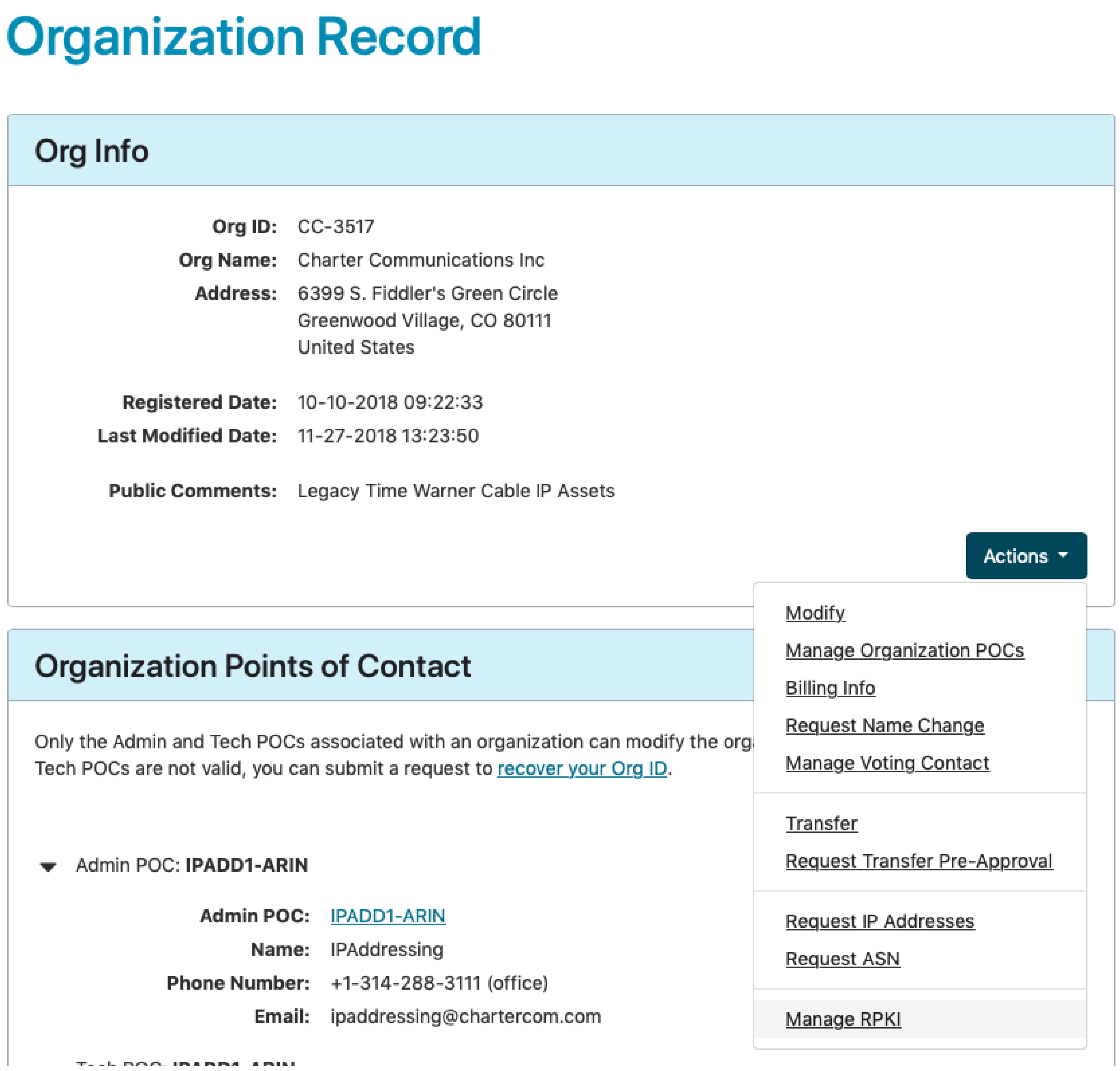
If your Org ID does not have a valid Admin or Tech POC, you can recover it.

Organizations Associated with Your User Account

Organizations are associated with your account if you are linked to any Point of Contact (POC) for the Org ID, or if you are the Voting Contact.

Org Handle	Org Name
BHN-2	BRIGHT HOUSE NETWORKS, LLC
CC04	Charter Communications
CC-3517	Charter Communications Inc
CC-3518	Charter Communications, Inc
TELOV	Telovations, Inc.
TWCIL	Time Warner Cable Internet LLC

- Choose the organization for which you want to configure RPKI.
- Choose Actions and select Manage RPKI.



Organization Record

Org Info

Org ID:	CC-3517
Org Name:	Charter Communications Inc
Address:	6399 S. Fiddler's Green Circle Greenwood Village, CO 80111 United States
Registered Date:	10-10-2018 09:22:33
Last Modified Date:	11-27-2018 13:23:50
Public Comments:	Legacy Time Warner Cable IP Assets

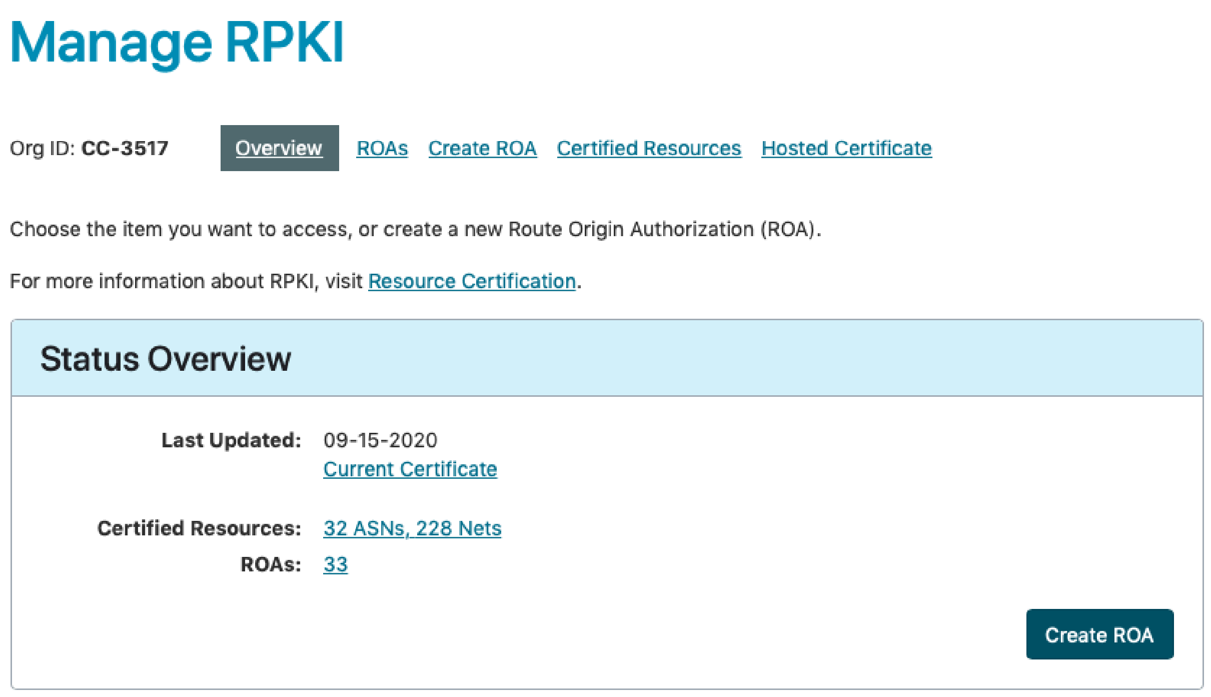
Organization Points of Contact

Only the Admin and Tech POCs associated with an organization can modify the org
Tech POCs are not valid, you can submit a request to recover your Org ID.

Admin POC: **IPADD1-ARIN**

Admin POC:	IPADD1-ARIN
Name:	IPAddressing
Phone Number:	+1-314-288-3111 (office)
Email:	ipaddressing@chartercom.com

-
- Choose Configure Hosted and accept the Terms of Service, if required.
- Copy and paste the public key you created in Step 1 into your certificate request.
- After you submit your request, ARIN will create a certificate request ticket. When your request is approved, ARIN will issue an RPKI certificate that covers the resources assigned to your Org.
- It may take a few days for ARIN to work your ticket and create a RPKI certificate based off of your public key!
- Once the ticket has been worked and you have a certificate, navigate back to “Your Records > Organization Identifiers” and choose the organization for which you want to configure RPKI.
- Choose Actions and select Manage RPKI.
- Choose Create ROA.


Manage RPKI

Org ID: **CC-3517**

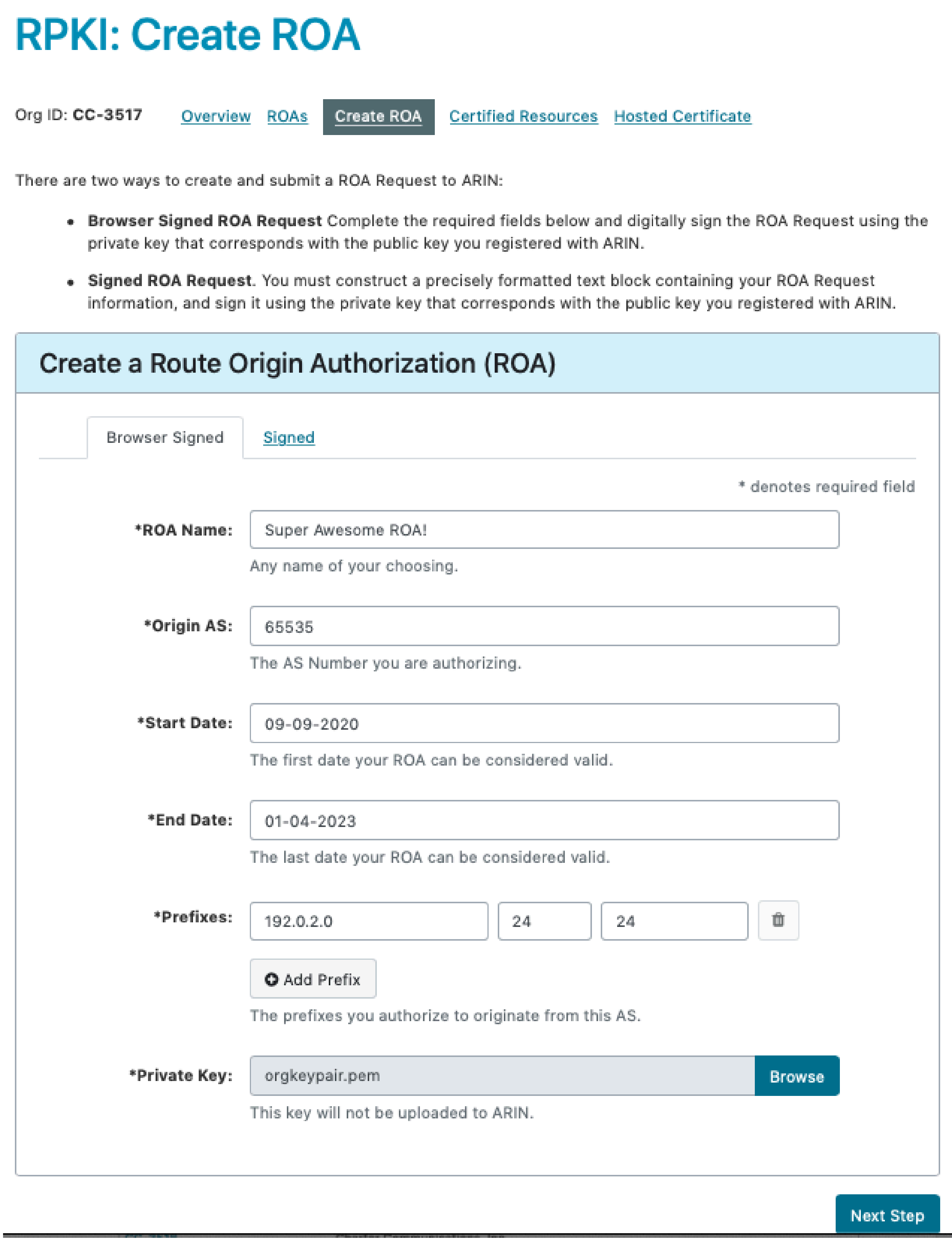
Choose the item you want to access, or create a new Route Origin Authorization (ROA).

For more information about RPKI, visit Resource Certification.

Status Overview

Last Updated:	09-15-2020 Current Certificate
Certified Resources:	32 ASNs, 228 Nets
ROAs:	33

- Choose the tab corresponding to how you want to create and submit the ROA:
  - Browser Signed: (described in the next section) This is the easiest method, where the browser uses JavaScript to parse your private key (it is not uploaded to ARIN) and sign the ROA.



RPKI: Create ROA

Org ID: **CC-3517** Overview ROAs Create ROA Certified Resources Hosted Certificate

There are two ways to create and submit a ROA Request to ARIN:

- **Browser Signed ROA Request** Complete the required fields below and digitally sign the ROA Request using the private key that corresponds with the public key you registered with ARIN.
- **Signed ROA Request**. You must construct a precisely formatted text block containing your ROA Request information, and sign it using the private key that corresponds with the public key you registered with ARIN.

Create a Route Origin Authorization (ROA)

Browser Signed | Signed

* denotes required field

***ROA Name:** Super Awesome ROA!
Any name of your choosing.

***Origin AS:** 65535
The AS Number you are authorizing.

***Start Date:** 09-09-2020
The first date your ROA can be considered valid.

***End Date:** 01-04-2023
The last date your ROA can be considered valid.

***Prefixes:** 192.0.2.0 24 24

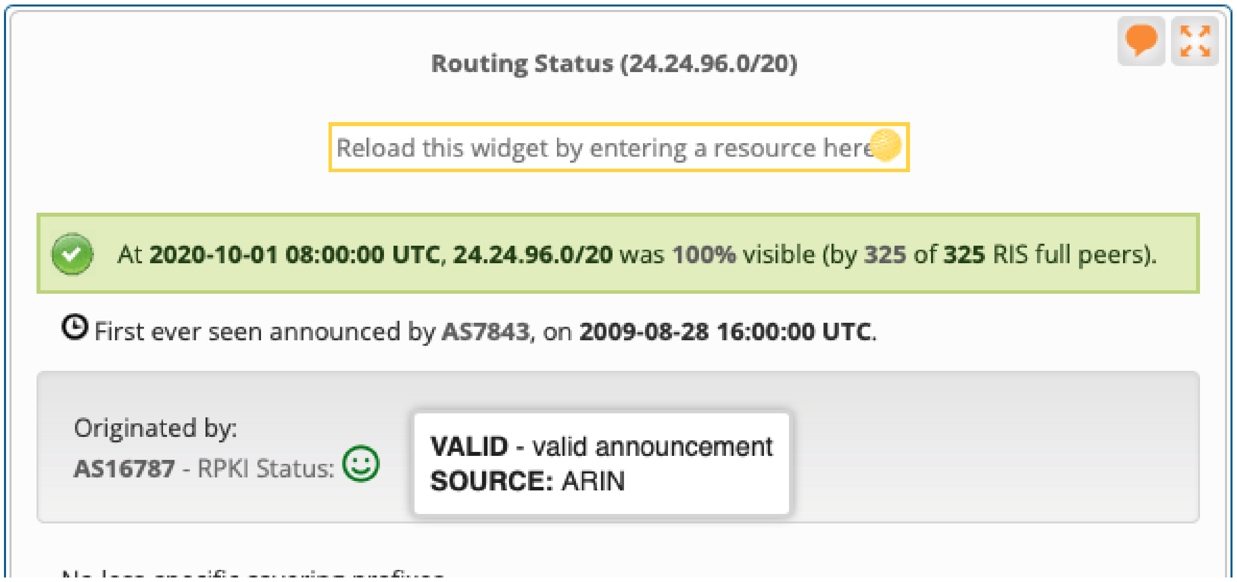
Add Prefix

The prefixes you authorize to originate from this AS.

***Private Key:** orgkeypair.pem Browse
This key will not be uploaded to ARIN.

Next Step

- In a few minutes to an hour you can validate that the ROAs are being seen on the Internet by navigating to a looking glass site like https://stat.ripe.net/widget/routing-status and putting in the prefix. You should see something like this:



Routing Status (24.24.96.0/20)

Reload this widget by entering a resource here

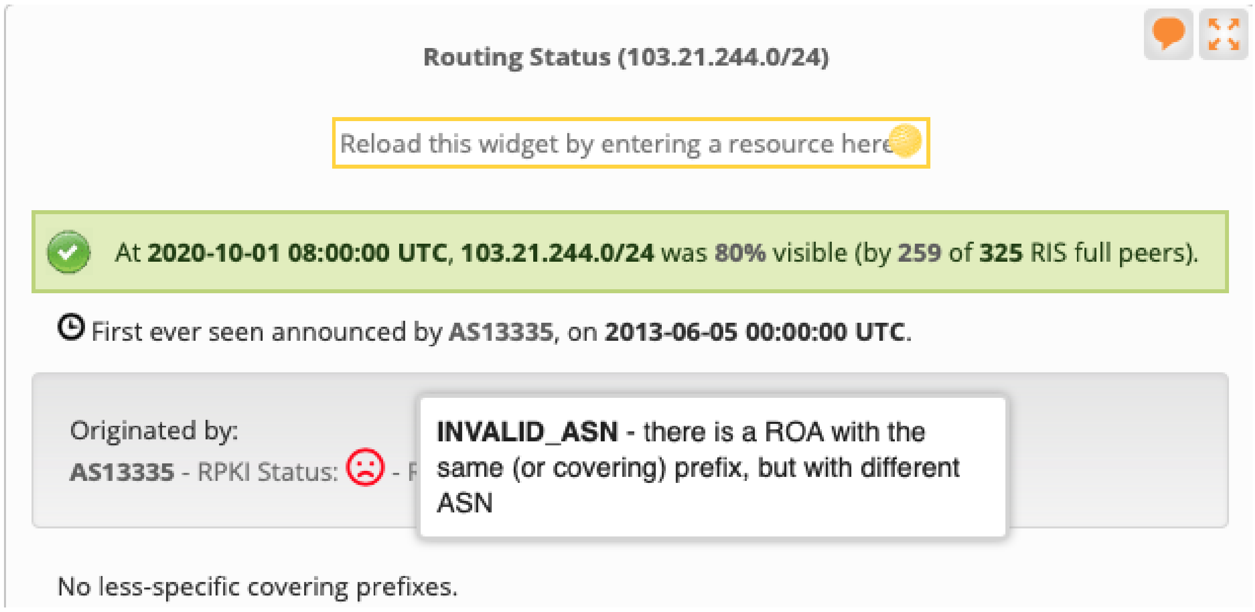
At **2020-10-01 08:00:00 UTC**, **24.24.96.0/20** was **100%** visible (by **325** of **325** RIS full peers).

First ever seen announced by **AS7843**, on **2009-08-28 16:00:00 UTC**.

Originated by:
AS16787 - RPKI Status:

VALID - valid announcement
SOURCE: ARIN

If you see that the ROA is showing as INVALID as in the following screenshot, you will need to **delete the invalid ROA ASAP** (the advertisement for this prefix is being dropped by those providers doing ROV), figure out what the issue was to make it invalid, and try again to create a valid ROA.



Routing Status (103.21.244.0/24)

Reload this widget by entering a resource here

At **2020-10-01 08:00:00 UTC**, **103.21.244.0/24** was **80%** visible (by **259** of **325** RIS full peers).

First ever seen announced by **AS13335**, on **2013-06-05 00:00:00 UTC**.

Originated by:
AS13335 - RPKI Status:

INVALID_ASN - there is a ROA with the same (or covering) prefix, but with different ASN

No less-specific covering prefixes.

- Other sites that can show you if you are advertising invalids:
  - https://rpki-monitor.antd.nist.gov/?p=0&s=3&asn=AS1678
  - https://rpki-validator.ripe.net/bgp-preview
  - https://bgpstuff.net/invalids
    - FYI, you can do "curl https://bgpstuff.net/invalids/<ASN>" in a script to do monitoring for invalids

- Many ISPs are setting an expiration date of 5-10 years from the start date.
- The ROA name can be anything you like. Might be best to use something that is relevant for your organization. Like defining the class of service for the prefix, region where the prefix is used, etc.
- Numerous prefixes can be added to a ROA but it is not recommended. If you have numerous prefixes in one ROA and you have to change one, the entire ROA needs to be deleted and re-created.

Generating ARIN Hosted ROAs with the API

- If you have more than a handful of IPs but you want to stick with the hosted ROA solution, you may want to use the RIR's APIs to generate and manage ROAs.
- Here is a script to generate ROAs using ARIN's API: https://github.com/racompton/arin-roa-request/

Krill Delegated CA 🦐

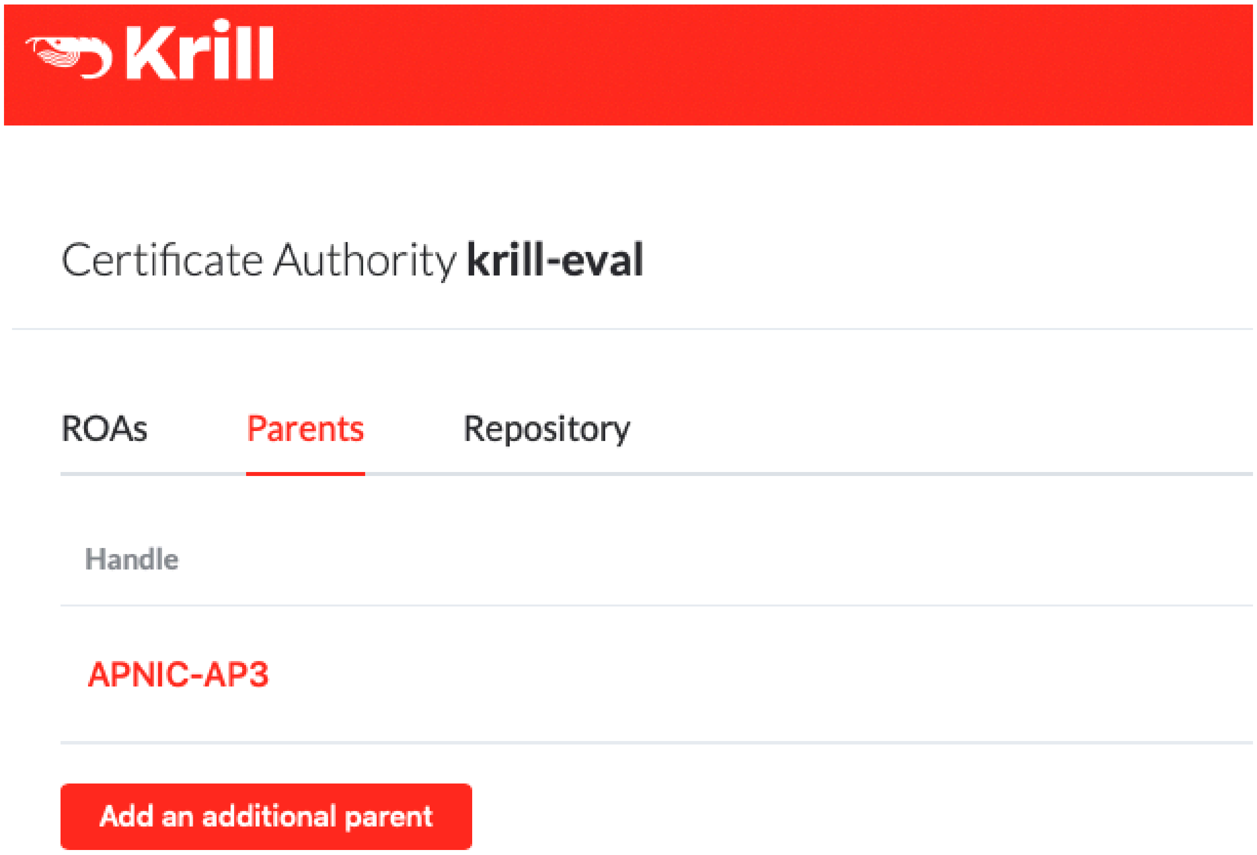
- Most common solution used for delegated CA
- Krill is open source (https://github.com/NLnetLabs/krill). NLnet Labs develops Krill and can provide commercial support if required.
- Has a fully functional RESTful API
- Shows info about BGP advertisements of your resources from RIPE's RIS
    - It can warn you if a ROA that you are going to create will make one of the BGP advertisements invalid
- Most delegated CAs also have a public repository for RRDP and RSYNC. You may want to do load balancing (and maybe GLSB) to multiple servers running rsyncd and apache/nginx. The RRDP repository can also be hosted on a CDN.
    - **Optimize for availability, not for load!**
        - It is expected that the number of connections per second should be very low. A back of the envelope calculation is that if all the ASNs in the world (around 70k) have 4 validator servers connecting every 10 mins then the number of connections will be about 500 connections per second.
    - Since the repository is public and needs to be highly available, **don't forget DDoS protection!**

Setting up an ARIN ORG-ID to a Krill delegated CA

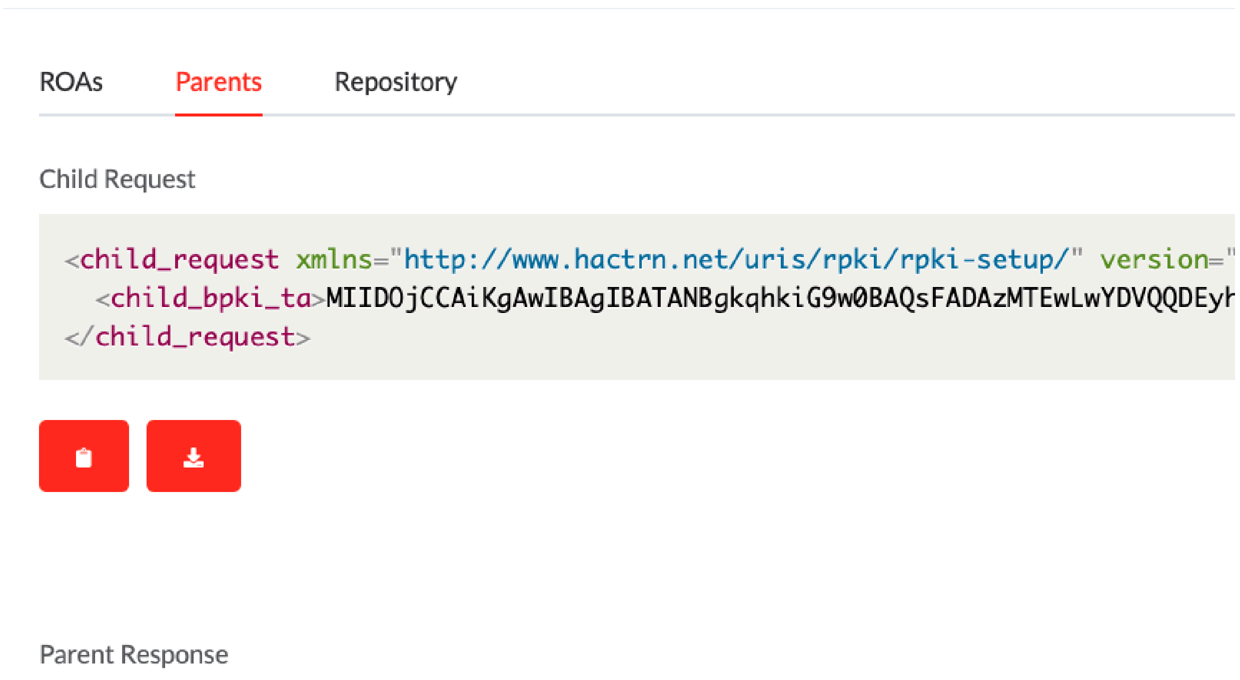
The ARIN portion of this procedure is documented on ARIN's site:
https://www.arin.net/resources/manage/rpki/delegated/#configuring-delegated-rpki
NOTE: before doing this procedure you must have an instance of the Krill CA that can reach ARIN's servers and a repository that is reachable via HTTPS (with a valid cert) and RSYNC from any IP on the Internet!

In order to use a delegated CA, you have to create a parent/child relationship between the Delegated CA and the RIR as defined by https://tools.ietf.org/html/rfc8183

- In Krill, navigate to the “Parents” tab in the web UI



- Click the “Add Additional Parent” button
- Click the “Download” ↓ button to download the child request file

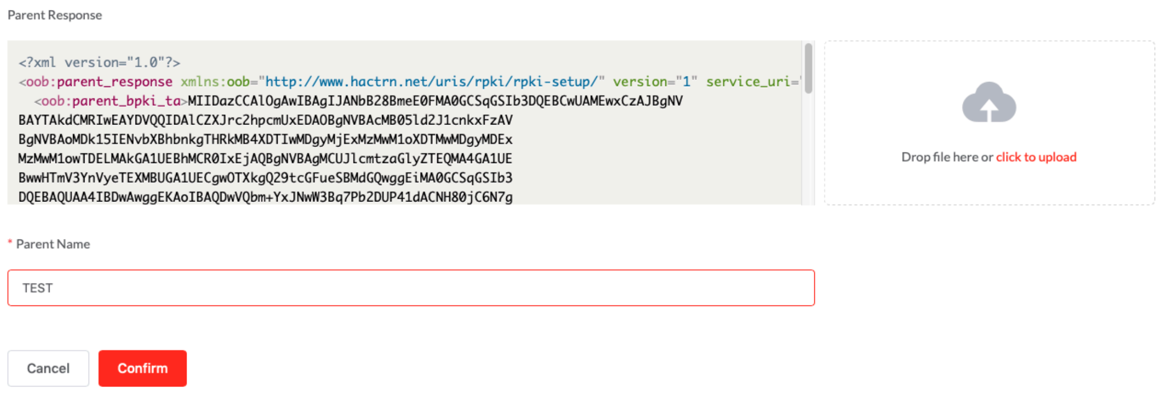


Child Request

```
<child_request xmlns="http://www.hactrn.net/uris/rpki/rpki-setup/" version="
  <child_bpki_ta>MIIDOjCCAiKgAwIBAgIBATANBgkqhkiG9w0BAQsFADAzMTEwLwYDVQQDEyh
</child_request>
```

Parent Response

- Select Your Records > Organization Identifiers from the navigation menu.
- Choose the organization for which you want to configure RPKI.
- Choose Actions and select Manage RPKI. (Note: If you do not see this option, ensure that you meet the requirements for participation).
- Choose Configure Delegated.
- Read and agree to the RPKI Terms of Service. (Note: Not required for resources covered by an RSA version 12 or greater.)
- Browse to select your Child Request XML file (as described in RFC 8183) and choose Submit. (This file obsoletes the `identity.xml` file, but references may still exist to the `identity.xml` file in software and documentation.) A ticketed request is generated for ARIN to sign up your organization for Managed RPKI. Upon approval of your request, your ticket will be updated and include the Parent Response XML File as an attachment. You will need to use this Parent Response XML file in your RPKI software (for example, Krill) when you are configuring your Delegated RPKI.
- ARIN will take a few days to generate a “Parent Response”. Copy the XML and paste in in the “Parent Response” field in Krill or upload a file with the XML in it into Krill


Parent Response

```
<?xml version="1.0"?>
<oob:parent_response xmlns:oob="http://www.hactrn.net/uris/rpki/rpki-setup/" version="1" service_uri='
  <oob:parent_bpki_ta>MIIDazCCAlOgAwIBAgIJANbB28BmeE0FMA0GCSqGSIb3DQEBCwUAMEwxCzAJBgNV
BAYTAkdCMRIwEAYDVQQIDAlCZXJrc2hpcmUxEDAOBgNVBAcMB05ld2J1cnkxFzAV
BgNVBAoMDk15IENvbXBhbnkgTHRkMB4XDTIwMDgyMjExMzMwM1oXDTMwMDgyMDEx
MzMwM1owTDELMAkGA1UEBhMCR0IxEjAQBgNVBAgMCUJlcmtzaGlyZTEQMA4GA1UE
BwwHTmV3YnVyeTEXMBUGA1UECgwOTXkgQ29tcGFueSBMdGQwggEiMA0GCSqGSIb3
DQEBAQUAA4IBDwAwggEKAoIBAQDwVQbm+YxJNwW3Bq7Pb2DUP41dACNH80jC6N7g
```

* Parent Name

TEST

- Krill will then contact the parent (eg. ARIN) via the Up/Down protocol to submit a Certificate Signing Request (CSR) with your repository location information included. ARIN will then publish a signed certificate with your ARIN-issued resources. On the "ROA" tab in Krill you will be able to see the IP prefixes that are associated with that ARIN ORG-ID. You can now start generating ROAs!
    - Note, that the Krill CA will reach out every 10 mins to the RIR via the Up/Down protocol to get an updated list of IP prefixes and ASNs associated with your ORG-ID. As a result, new resources will automatically appear on your certificate so you can create ROAs for it. Resources that you no longer have will be removed from the certificate, along with any ROAs that cover the resources.

Testing Out a Delegated CA with ARIN's OT&E

- A pre-installed Krill instance that includes Nginx and rsyncd, automated TLS config and various monitoring end points can be set up quickly using the 1-click AWS Marketplace or Digital Ocean Marketplace instances.
- After setting up the instance using Krill Manager and installing a valid TLS cert on the instance you can follow the above "Setting up an ARIN ORG-ID to a Krill delegated CA" procedure to set up a delegated CA in ARIN's OT&E test environment: https://account.ote.arin.net
    - The procedure is the same as the above (child request sent, parent response added to Krill) but the OT&E environment is just for testing and **all changes are erased at the end of the month!** So if you set up a test delegated CA in OT&E in September, in October you would need to set it up again.
- If you want the ROAs that you create in your test delegated CA to be downloaded to your validators you will need to get the OT&E TAL (Trust Anchor Link) from here: https://www.arin.net/reference/tools/testing/#trust-anchor-locator-tal
- The TAL needs to be saved as a .tal file on the validators and put in the same directory as your other TALs and the validator process needs to be restarted.
- The ROAs created in the test Krill instance should show up on the routers that have an RTR session to those validators in a few minutes

Legal aspects of RPKI

ARIN TAL unlike other public crypto key files does not come pre-installed in the validators. It must be separately downloaded here: https://www.arin.net/resources/manage/rpki/tal/. Note that this policy is not without controversy as discussed here and here on the NANOG list.

Job Snijders made a video explaining his perspective on the ARIN TAL. Christopher Yoo and David Wishnick authored a paper titled Lowering Legal Barriers to RPKI Adoption.

Ben Cox performed various RPKI measurements and concluded that the ARIN TAL is used far less than TALs from their RIR counterparts. This has led to a situation where ROAs created under the ARIN TAL offer less protection against BGP incidents than other RIRs. State of RPKI: Q4 2018.