RPKI Route Server Deployment at JPNAP

APIX#23

Nasato Goto / goto@mfeed.ad.jp

JPNAP

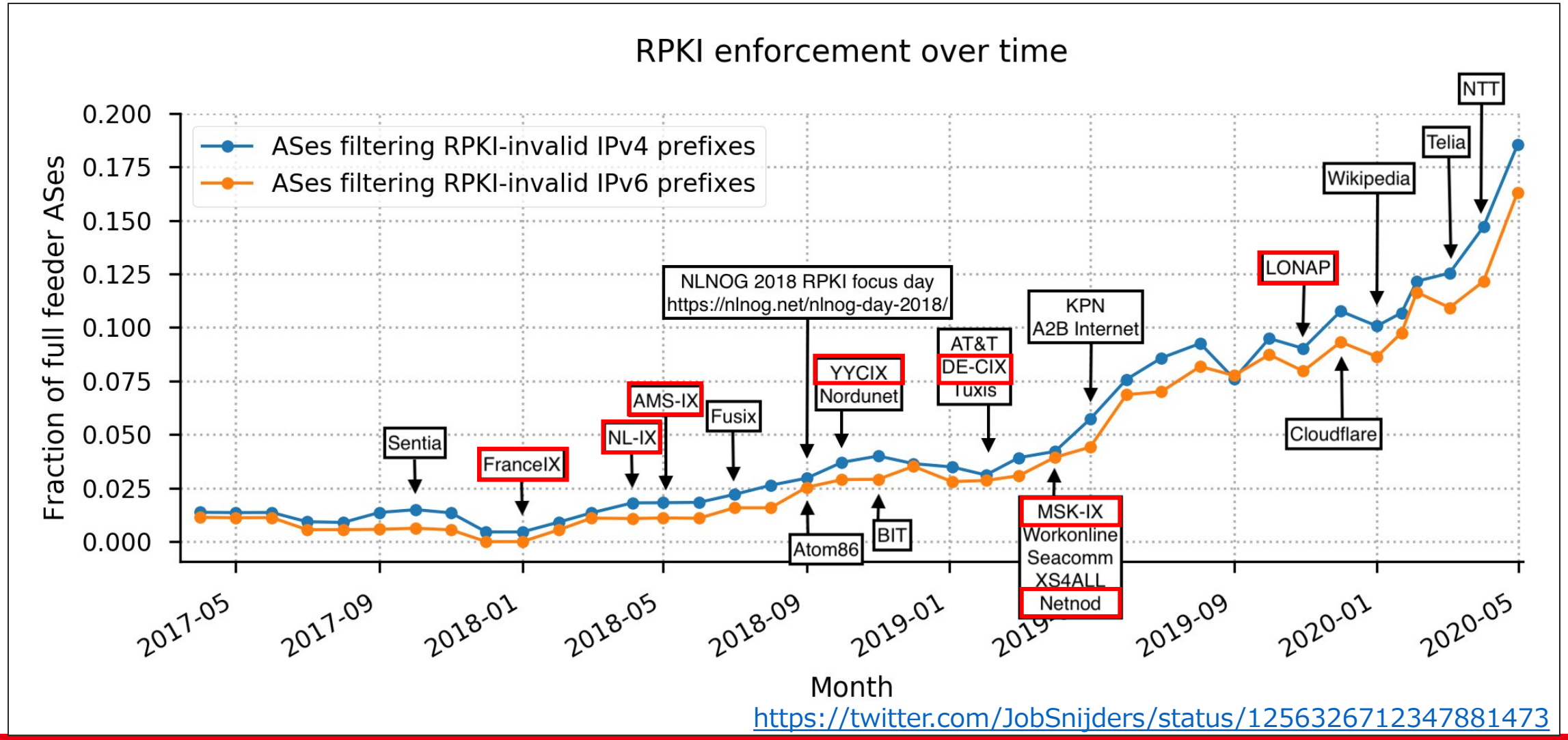
About this presentation

- JPNAP started RPKI ROV on production route servers in December 2020
- Let me share our pathway to deploy it and have a discussion to proceed RPKI on Asia Pacific region

- This is English version of our JANOG47 presentation
  - https://www.janog.gr.jp/meeting/janog47/en/rpkiix-en/

Background: World IXPs RPKI Adoption

- Since 2018, large IXPs mainly in Euro region have started to deploy RPKI



https://twitter.com/JobSnijders/status/1256326712347881473

Background: World IXPs (and others) RPKI Adoption

- APIX members also deploy it gradually
  - 4 of the 32 IXPs (survey in 2021/01)
    - BKNIX : 2019/03~
    - HKIX : 2020/08~
    - IX Australia : 2020/09~
    - TWIX : 2020/??~
- In addition to IXP, many ISPs and CSPs have reported ROA creation/ROV implementation
  - ROUTING SECURITY: RPKI UPDATE Q2/20
    - Telia's report says many Tier1 ISPs rejects Invalids in 2020
  - Expanding our commitment to secure Internet routing
    - Google has registered more than 99% of its routes to ROA, will deploy ROV in 2021
  - How AWS is helping to secure internet routing
    - AWS have over 99% their address space covered under ROA, and they are now dropping Invalids in all their POP

Tell me if you are not in the list :)

Is BGP safe yet? No.

NAME	TYPE	DETAILS	STATUS
Telia	transit	signed + filtering	safe
Cogent	transit	signed + filtering	safe
GTT	transit	signed + filtering	safe
NTT	transit	signed + filtering	safe
Hurricane Electric	transit	signed + filtering	safe
TATA	transit	signed + filtering	safe
PCCW	transit	signed + filtering	safe
RETN	transit	partially signed + filtering	safe
Cloudflare	cloud	signed + filtering	safe
Amazon	cloud	signed + filtering	safe
Netflix	cloud	signed + filtering	safe
Wikimedia Foundation	cloud	signed + filtering	safe
Scaleway	cloud	signed + filtering	safe
Telstra International	transit	signed	partially safe
AT&T	ISP	signed + filtering peers only	partially safe
Google	cloud	signed	partially safe

https://isbgpsafeyet.com/

Now is the best time to start RPKI!

Deployment Steps

1.Design

Policy, Technology Selection, Parameters

2.Test

Test ROV, RTR connections and so on

3.Production migration

Timeline for production environment, customer care and the result of adoption

4.Operation

Monitoring, operation and tools

Deployment Steps

1.Design

Policy, Technology Selection, Parameters

2.Test

Test ROV, RTR connections and so on

3.Production migration

Timeline for production environment, customer care and the result of adoption

4.Operation

Monitoring, operation and tools

Design: Policy

- Decide routes filtering policy for ROV results (Valid/NotFound/Invalid)
  - ≒ Can we just drop Invalids?

- Discussion with community and our users
  - Strategy for deploying RPKI ROV to Route Server on IX, 2019/09 APNIC48
    - JPNAP asked “How about if the IXP customers can decide handling of Invalid routes?”
    - Feedback “Stop it! Don’t force customers too complicated!”
  - Questionnaire in JPNAP users meeting, 2019/10
    - More than half customers answered “they don’t need Invalids”

- Due to increase of adoption rates in 2020, “tag and advertise Invalids” phase was finished, then now we are in the world everyone do “drop Invalids”

JPNAP decided to accept Valid, NotFound and reject Invalid

Design: Selecting BGPd

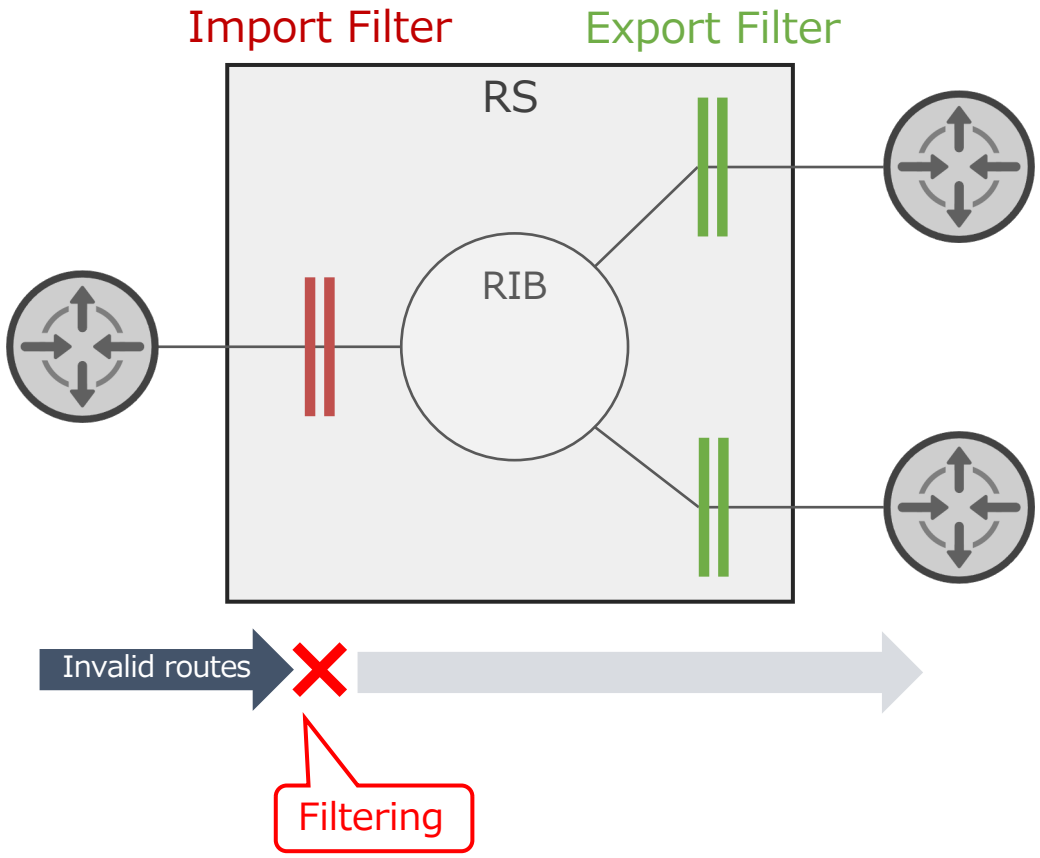
- Background: one of our BGPd was too old, without support of RPKI
  - (There was another project “upgrading BGPd” behind the scenes)
- OSS BGPd’s RPKI features status

	BIRD1.0	BIRD2.0	GoBGP	OpenBGPD	FRR
RTR	No ※static VRP in config	Yes	Yes	No *use rpki-client *will be Yes in May 2021	Yes
ROV Filtering	Yes	Yes	Yes	Yes	Yes

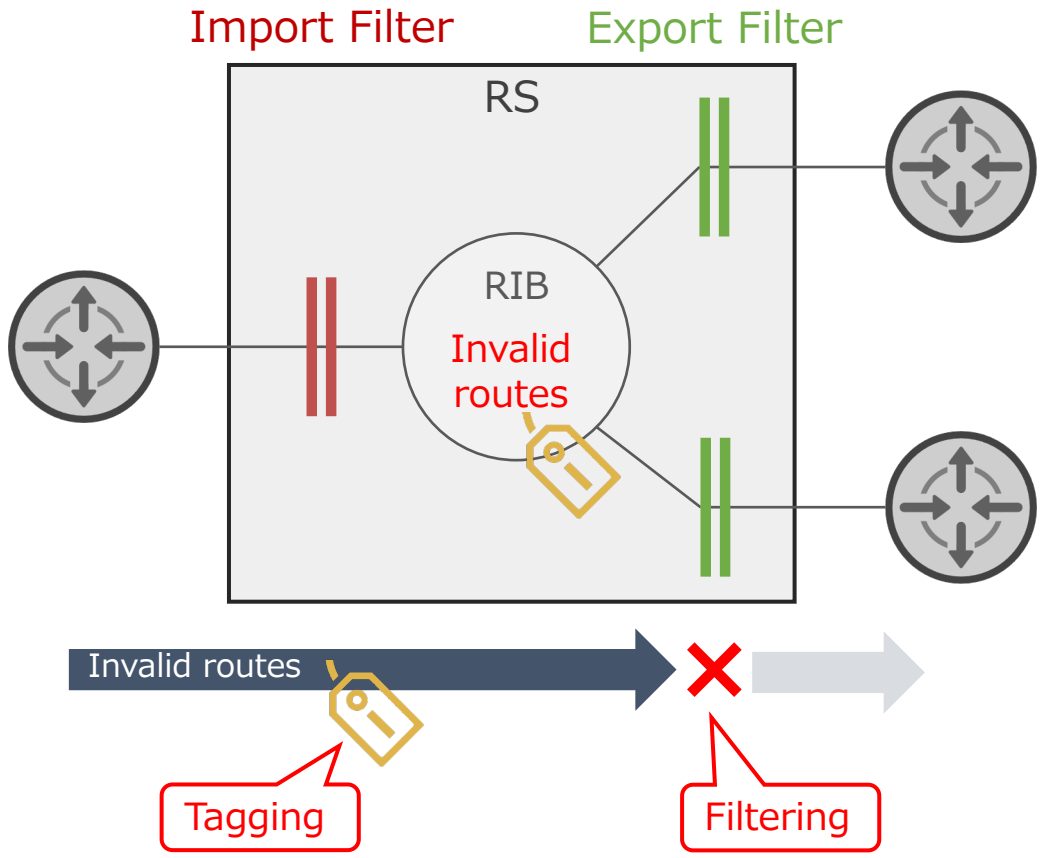
- We selected BIRD2.0
- Reasons:
  - Many IXPs use BIRD, as you know
  - Tool supports
    - API server、exporter、Looking Glass
  - Less IXPs use BIRD2.0 compared with BIRD1.0, but developers recommend to migration

Design: ROV Filtering Strategy

Pattern1: Drop Invalids immediately at Import filter



Pattern2: Tag Invalid routes with Community, and drop them at Export filter



※BIRD2.0 can do both

Design: ROV Filtering Strategy

Pattern1: Drop Invalids immediately at Import filter

- ✓ Good performance (a bit, maybe)
  ∵Don't install Invalid routes
- ✓ Applying unified filtering policy
  "Reject routes that do not meet the condition"
  → Keep the BGPd config simple
- ✗ Unable to achieve "Advertise Invalids with tags"

Pattern2: Tag Invalid routes with Community, and drop them at Export filter

- ✓ Being able to advertise Invalid routes
  → Meet special demands such like experiment
- ✗ RIB contains Invalid routes
  → Invalids are unnecessary in many cases
- ✗ Need to ensure consistency with other route filters
  → How do we treat routes rejected on IRR filter ?
  → This lead the config to be messy

Which pattern do other 10 IXPs select?

- Pattern1: **7 IX** (DE-CIX, LINX, LONAP, etc)
- Pattern2: **3 IX** (AMS-IX, BKNIX, France-IX)

JPNAP selected Pattern1 to achieve our policy simply

※If we find out the benefits for customers by Pattern2, then we'll reconsider

FYI: ROV Filtering Strategy in Other IXPs

- Pattern1: Drop Invalids immediately at Import filter
  - LINX: https://portal.linx.net/tech-info-help/route-servers
  - LONAP: https://www.lonap.net/tech/route-servers
  - DE-CIX: https://www.de-cix.net/en/locations/germany/frankfurt/routeserver-guide
  - MSK-IX: https://kb.msk-ix.ru/en/ix/services/route-server/
  - NETNOD: https://www.netnod.se/ix/route-servers
  - HKIX: https://www.hkix.net/hkix/route-policy.htm
  - swissix: https://www.swissix.ch/infrastructure/routeserver
-
- Pattern2: Tag Invalid routes with Community, and drop them at Export filter
  - France-IX: https://www.franceix.net/en/technical/france-ix-route-servers/
  - AMS-IX: https://www.ams-ix.net/ams/documentation/ams-ix-route-servers
  - BKNIX: https://bknix.co.th/en/index.php?module=technical&content=9

Design: Selecting Relying Party (Validator)

- Routinator3000
  - 2018~, Github Star 200+
- Reasons:
  - Single binary, easy installation
  - Built-in RTR server
  - Web API、Cooperation with Grafana using exporter
  - Active development
  - Good reputation in APRICOT2020 RPKI Deployathon

- Deployment
  - Only Routinator, no Relying Party implementation diversity for now
    - To reduce costs for management and development Infrastructure as a Code (ansible)
    - (We want to achieve this in the future)
  - Deploy Relying Party on the same management network with RS
    - 2 instances for JPNAP{Tokyo|Osaka|Fukuoka}, RS establish RTR with each 2
- Question for you
  - Do you provide Validator service for your customers via IX network or have plan to do this?
    - Providing Relying Party to IXP customers on IXP L2 network. Increase security (private > IX > public)
  - I asked that “Would you like to use Relying Party service on IXP?” to JANOG Community.
    - Some said “Yes, for Relying Party redundancy”

FYI: Relying Party (Validator) Comparison

		RIPE NCC RPKI Validator	Routinator3000	FORT	OctoRPKI	rpki-client(-portable)
Overview	Maintaner	RIPE NCC	NLnet Labs	FORT Project (LACNIC + NIC.MX)	Cloudflare	RSSF
	Established	2011	2018	2018	2019	2020
	Github Star	50+	200+	20+	100	10 ※portable ver
	Written in	Java	Rust	C	Go	C
Features	Built-in RTR Server	No (separated in same repo)	Yes	Yes	No (GoRTR)	No (GoRTR)
	Secure RTR	No	Yes ※ssh proxy	Yes	- (GoRTR)	- (GoRTR)
	SLURM対応	Yes	Yes	Yes	- (GoRTR)	- (GoRTR)
Additional	WebAPI	Yes	Yes	No	No	No
	Prometheus exporter/metrics	Yes	Yes	No	Yes	No
	Document ※Personal subjectivity	☺	☺	😐	😐	☹
Note	-	2021/07 EOL				

※2021/01

FYI: Relying Party VRP Number Diffs

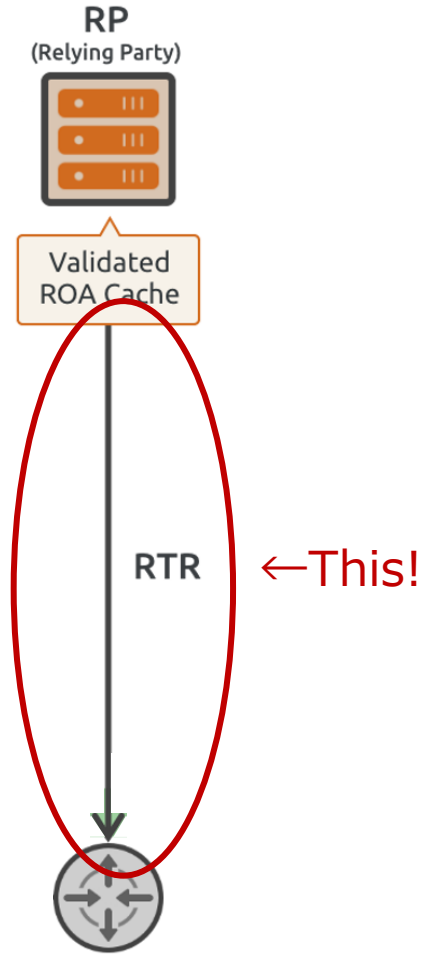
- This post in APNIC Blog says that there are diff in the number of VRP generated by Relying Party
  - Routinator and RIPE NCC Validator generate same and maximum # of VRP, v4: 114,961, v6: 19,307
  - FORT generates few VRPs compared to those (just error?)
  - OctoRPKI generates about 1,200 fewer to those
- We did same test

Date		RIPE NCC RPKI Validator	Routinator3000	FORT	OctoRPKI	rpki-client(-portable)
2021/01/14	v4	181,788	181,788	181,788	180,516	181,788
	v6	30,601	30,600	30,600	29,901	30,600
2021/01/17	v4	182,010	182,010	182,010	180,755	181,939
	v6	30,646	30,646	30,645	29,957	30,634
2021/01/20	v4	182,414	182,414	182,414	181,170	182,415
	v6	30,941	30,941	30,941	30,248	30,942

- Result
  - Same with APNIC Blog, OctoRPKI generated about 2,000 fewer VRPs
  - Other Relying Parties generates almost same number of VRPs
    - just a timing issue?
  - We haven't found the cause of this issue (tell me if you know)

Design: Parameters

- RTR Parameters
  - Refresh interval: 1h
    - How long to wait before next attempting to poll the cache using a Serial Query or a Reset Query packet
    - Recommended value in RFC8210 is 1h
  - Expire period: 36h
    - Received records are deleted if the client was unable to successfully refresh data for this time period
    - We set it as 36h to tolerate RTR failures to occur for up to 1.5 days
      - We'll be struggle to fix RTR server errors in the period...
    - RFC8210 recommends the value as 2h
      - Actually, we haven't been had such RTR connection errors
      - We'll reconsider this value to get closer to the recommendation value

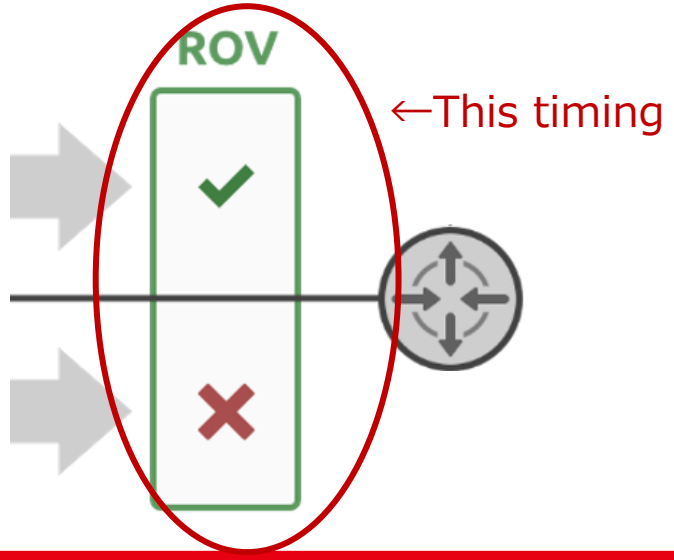


Design: Parameters

Background: JPNAP only update IRR base route filters once a day

※Now we consider to increase this frequency

- ROV re-validation
  - Re evaluate the routes that are already installed in Adj-RIBs-IN by updated VRPs
    - RFC6811 *"When a mapping is added or deleted, the implementation MUST re-validate any affected prefixes and run the BGP decision process if needed. "*
  - BIRD2.0 doesn't support auto re-validation in its current version 2.0.7
  - Then we re-validate explicitly at the timing of route filter update by executing bird command
    - `'birdc reload in all'`
    - Send ROUTE-REFRESH and re-validate received routes



Deployment Steps

1.Design
Policy, Technology Selection, Parameters

2.Test
Test ROV, RTR connections and so on

3.Introduction
Timeline for production environment, customer care and the result of adoption

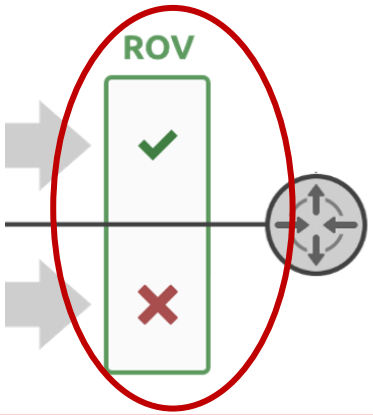
4.Operation
Monitoring, operation and tools

Test: BGPd/ROV

- Point: Can BGPd perform ROV correctly according to the algorithm described in RFC6811?
- Test items
  - Invalid origin
  - Invalid length
  - AS0
- (This table is also useful for our actual operation☺)

No	ROA Prefix/MaxLen	ROA OriginAS	BGP Route Prefix	BGP Route OriginAS	ROV Result
1	10.101.0.0/24-24	65001	10.101.0.0/24	65001	Valid
2	-	-	10.102.0.0/24	65001	NotFound
3	10.103.0.0/24-24	65111	10.103.0.0/24	65001	Invalid
4	10.104.0.0/24-24	65001	10.104.0.0/23	65001	NotFound
5	10.105.0.0/24-24	65001	10.105.0.0/24	65001	Valid
6	10.106.0.0/24-24	65001	10.106.0.0/25	65001	Invalid
7	10.107.0.0/23-25	65001	10.107.0.0/22	65001	NotFound
8	10.108.0.0/23-25	65001	10.108.0.0/23	65001	Valid
9	10.109.0.0/23-25	65001	10.109.0.0/24	65001	Valid
10	10.110.0.0/23-25	65001	10.110.0.0/25	65001	Valid
11	10.111.0.0/23-25	65001	10.111.0.0/26	65001	Invalid
12	10.112.0.0/23-25	65111	10.112.0.0/22	65001	NotFound
13	10.113.0.0/23-25	65111	10.113.0.0/24	65001	Invalid
14	10.114.0.0/23-25	65111	10.114.0.0/26	65001	Invalid
15	10.115.0.0/23-25	0	10.115.0.0/22	65001	NotFound
16	10.116.0.0/23-25	0	10.116.0.0/24	65001	Invalid
17	10.117.0.0/23-25	0	10.117.0.0/26	65001	Invalid

Correctness of this ↓

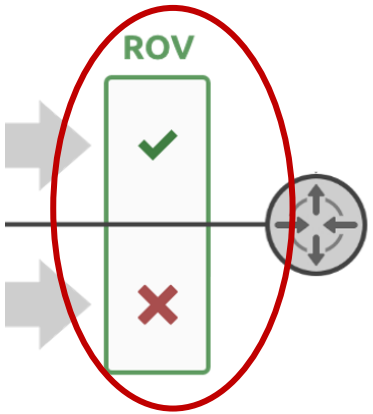


Test: BGPd/ROV

- Point: Can BGPd perform ROV correctly according to the algorithm described in RFC6811?
- Test items
  - Multiple ROAs for same prefix

No	ROA Prefix/MaxLen	ROA OriginAS	BGP Route Prefix	BGP Route OriginAS	ROV Result
1	10.103.0.0/24-24	65001	10.103.0.0/24	65001	Valid
	10.103.0.0/24-24	65001			
2	10.104.0.0/24-24	65001	10.104.0.0/24	65001	Valid
	10.104.0.0/24-24	65111			
3	10.105.0.0/23-25	65001	10.105.0.0/24	65001	Valid
	10.105.0.0/24-24	65111			
4	10.106.0.0/23-23	65001	10.106.0.0/24	65001	Invalid
	10.106.0.0/23-25	65111			

Correctness of this ↓



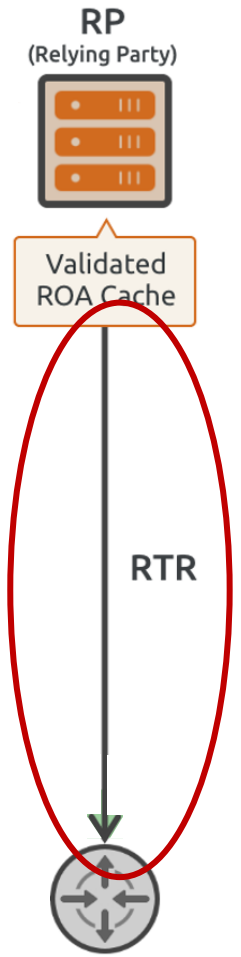
FYI: BGPd Test Automation

- 50% of BGPd test are done automatically on our laptop
  - Extending GoBGP Scenario Test framework (Thank you GoBGP ☺)
- This is an image of ROV tests as mentioned above
  - Check ROV results by asserting Large Communities attached to the routes (sorry for Japanese)

```
# 検証対象のルートサーバー（BIRD2.0）とクライアント（GoBGP）コンテナを起動
rs = BirdContainer(name='rs', asn=45686, ip_addr='210.173.176.213')
g1 = GoBGPContainer(name='g1', asn=65001, router_id='210.173.176.1', ip_addr='210.173.176.1')

# クライアントから経路を広報
g1.add_route('10.101.0.0/24', aspath=[65001])
g1.add_route('10.102.0.0/24', aspath=[65001])
g1.add_route('10.103.0.0/24', aspath=[65001])

# 以降、ROV判定結果をテスト

# No: 1
# Expected RPKI ROV Result: Valid
lc = [p['bgp']['large_communities'] for p in rs.get_adj_rib_in(g1) if p['prefix'] == '10.101.0.0/24']
assertTrue([45686, 1000, 1] in lc[0])

# No: 2
# Expected RPKI ROV Result: NotFound
lc = [p['bgp']['large_communities'] for p in rs.get_adj_rib_in(g1) if p['prefix'] == '10.102.0.0/24']
assertTrue([45686, 1000, 2] in lc[0])

# No: 3
# Expected RPKI ROV Result: Invalid
lc = [p['bgp']['large_communities'] for p in rs.get_adj_rib_in(g1) if p['prefix'] == '10.103.0.0/24']
assertTrue([45686, 1000, 4] in lc[0])
assertTrue([45686, 1101, 13] in lc[0])
```

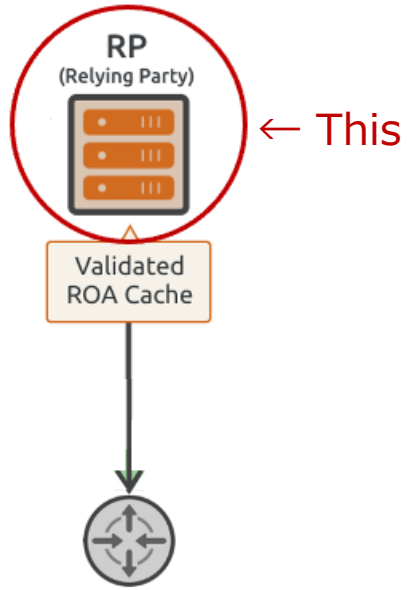
Test: BGPd/RTR Connection

- Point: Can BGPd talk with Relying Party?
  - We tested it based on actual behavior rather than protocol specification
- Test items
  - Connect/Disconnect
    - Establish RTR sessions for multiple cache servers?
    - Perform reset and re-connect correctly by command?
    - Logging
  - Parameters
    - Update VRPs according interval value?
    - Retry updates when get connection errors?
  - Expire
    - Keep VRPs and ROV results after RTR session disconnected, before expiration?
    - Erase local VRPs after expiration?
    - ROV result in NotFound after expiration?
  - and so on



Test: Relying Party

- Point: Can Relying Party verify ROA cryptographically correctly and generate VRPs?
- However, it's hard to test Relying Party itself without cryptographical knowledge
  - To do this, we need to setup CA and generate ROAs by ourselves
    - Now we have OSS CA Krill, so it looks easier compared to few years ago
    - I couldn't even install Dragon Research Labs CA 2,3 years ago...
- Minimum test items
  - ROA download, periodic update, output logs
  - Generate same number of VRPs with other Relying Party
  - Feed VRPs to routers by RTR
- It's important for us to select trustworthy Relying Party software
  - Trustworthy ≒ Many users, Active development, Smooth bugfix
  - + use alternative Relying Party implementation for redundancy...



Deployment Steps

1.Design
Policy, Technology Selection, Parameters

2.Test
Test ROV, RTR connections and so on

3.Production migration
Timeline for production environment, customer care and the result of adoption

4.Operation
Monitoring, operation and tools

Migration: Timeline

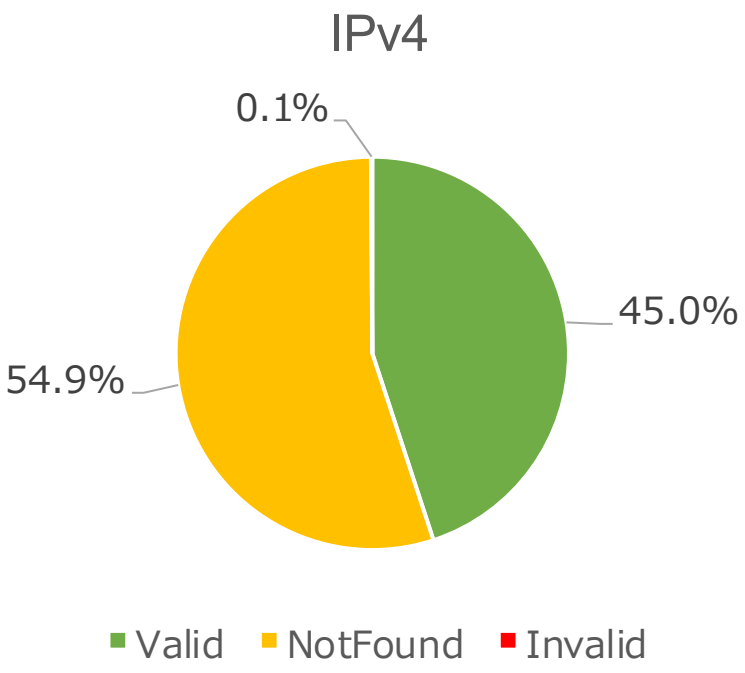
JPNAP Tokyo Long Way

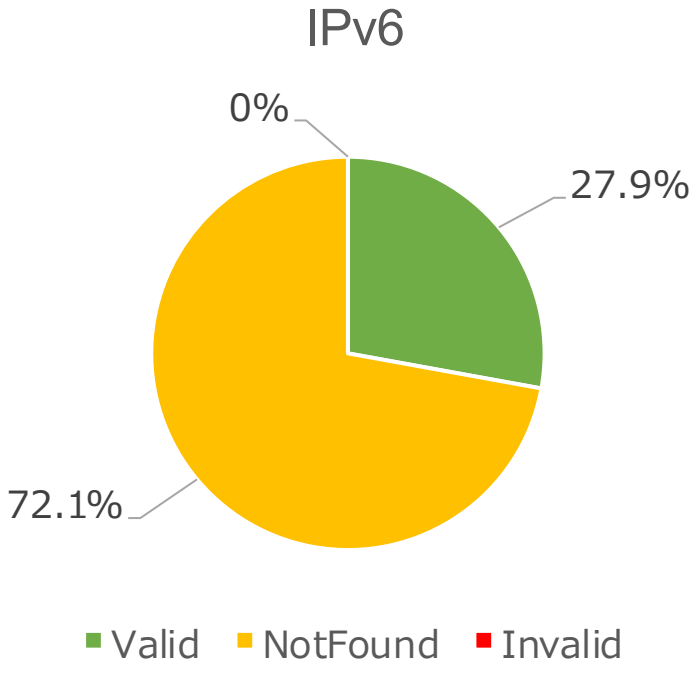
- 2020/01 (10mos. before) Start design and test BIRD2.0/RPKI
- 2020/05 (6mos. before) Had RPKI technical session at JPNAP users meeting
  - Review RPKI itself, share user AS case study, discuss world trends
- 2020/06 (5mos. before) Deploy RPKI-enabled trial route server at JPNAP Tokyo
  - Volunteer users connected to 3rd route server which enabled ROV
- 2020/10 (3weeks before) Announce ROV deployment in users meeting and mailing list
  - JPNIC gave us feedback (see later)
  - No objection and negative comments
    - We re-recognized that there were no feelings of refusal for dropping Invalids
- 2020/11 (2weeks before) Send e-mail to some users who advertised Invalid routes at that time
  - This was because we changed our policy, and it led the rejection of customer routes
  - We sent e-mail to 13 customers
  - 4 customers respond to us, and some of them made treatment
- 2020/11 Deploy BIRD2.0 and start ROV

Migration: Routes Status

- JPNAP Tokyo, 2021/01/17
- ROV result
  - Almost half of IPv4 routes are Valid
    - Compare to IPv4, the Valid rate of IPv6 are still low
    - Despite high register rate of IPv6 prefix in JPNIC region, but...
  - Zero Pv6 Invalids (!)

FYI:
JPNIC Stats (Japanese only)
ROA cover rate for allocated address space
- IPV4: 44.4%
- IPv6: 57.2%





Deployment Steps

1.Design
Policy, Technology Selection, Parameters

2.Test
Test ROV, RTR connections and so on

3. Production migration
Timeline for production environment, customer care and the result of adoption

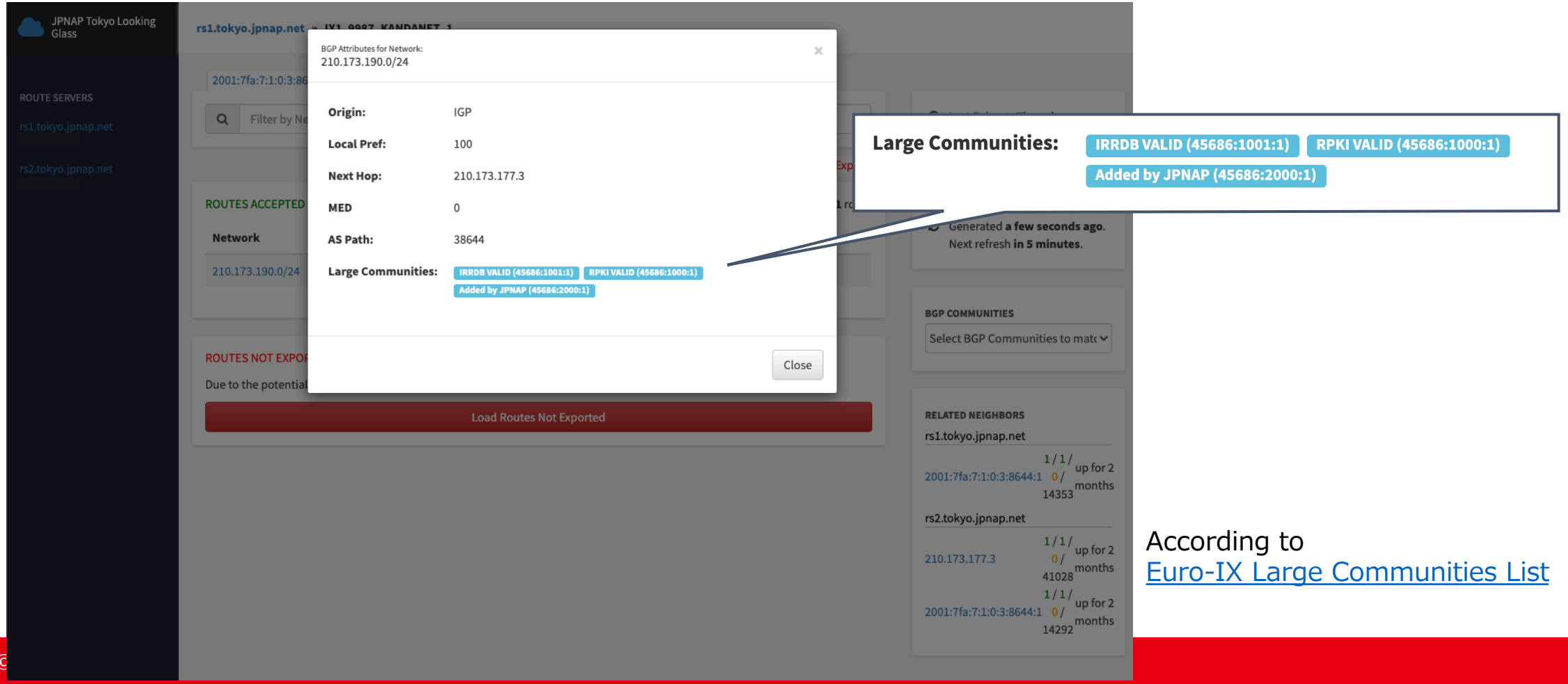
4.Operation
Monitoring, operation and tools

Operation: Monitoring/Manual Operation

- Monitoring
  - Feature monitoring
    - (Adding to usual server monitoring...)
    - RTR session between BGPd and cache server
    - Exit status of rsync when Relying Party correct ROAs fromrepository (not yet RRDP)
    - Next step: Track ROV validity change (Valid, NotFound → Invalid) for user routes
      - Need to develop tools such like exporter, because BIRD2.0 doesn't support such metrics
      - Extra: Provide this monitoring result to IXP customers???
  - Blackbox monitoring (not yet)
    - Using fake customers, monitor its route acceptance and ROV result?
- Manual operation
  - Update VRPs, ROV re-validation
  - Bypass ROV filtering for specific AS temporary for emergency
    - Not yet prepared SLURM (Is this needed?)

Operation: Tool for Customers

- We provide Looking Glass for our customers by using OSS [Alice-LG]
- It shows the validity of ROV, associated to (Large) BGP Community
  - = Customers can check the validation result by themselves



According to
[Euro-IX Large Communities List]

Deployment Steps

1.Design
Policy, Technology Selection, Parameters

2.Test
Test ROV, RTR connections and so on

3.Production migration
Timeline for production environment, customer care and the result of adoption

4.Operation
Monitoring, operation and tools

2020.12.24

JPNAP RouteFEED service implemented RPKI ROV on all route server platform. This improves the security of route exchange between RouteFEED customers. JPNAP will continue our efforts to improve internet routing security.

Heads-up: Traffic Issue due to ROV

- In the fully ROV deployed world, it might occur traffic trouble caused by ROV

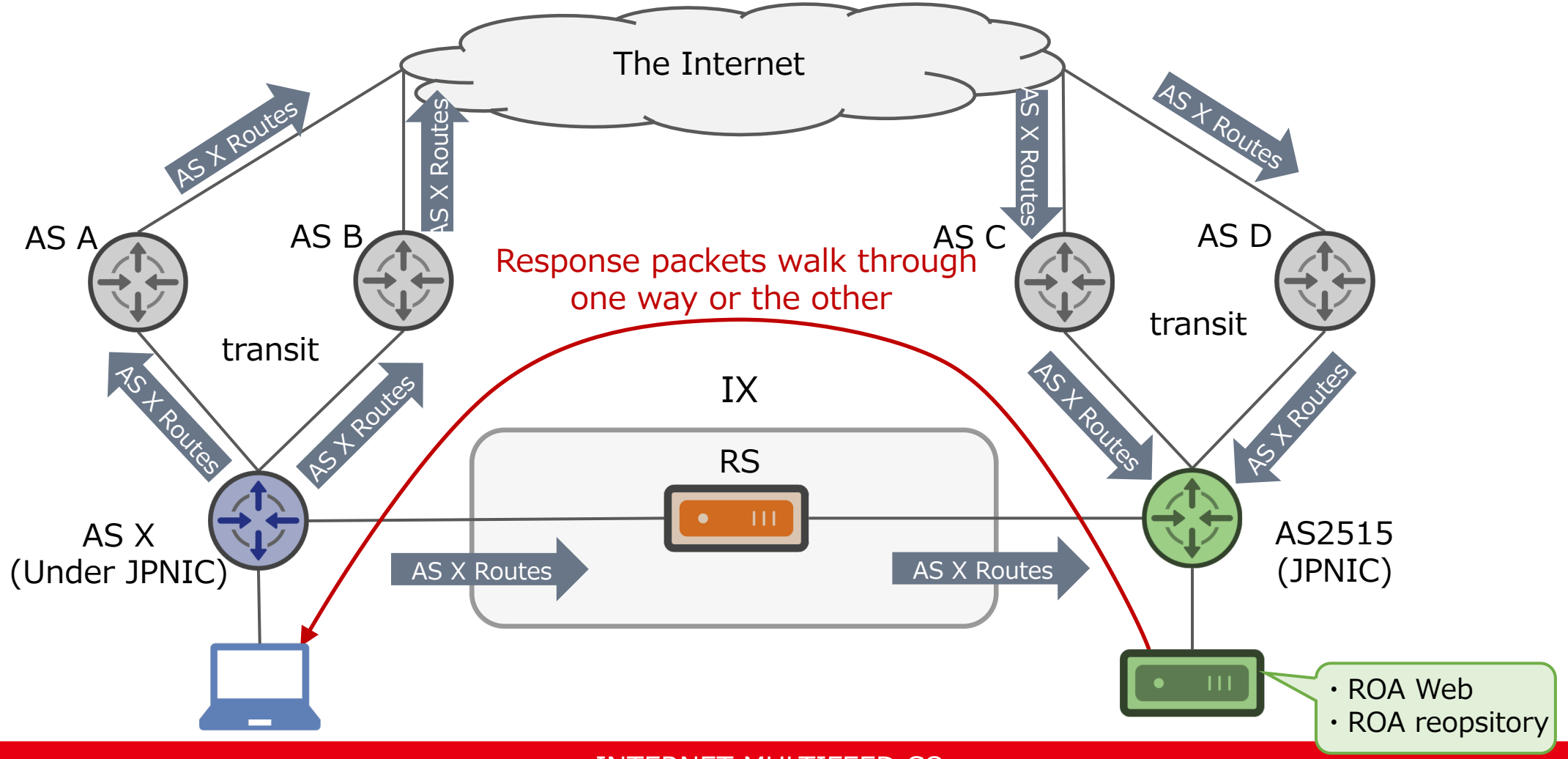
- Background
  - JPNIC gave us a feedback about this issue
  - It's a bit extreme trouble for now, but we should be careful about these kind of issues in the future

- Issue overview
  - When an AS register wrong ROAs under the situation that its neighbor ASs drop RPKI Invalid routes, this will lead the AS to be isolated from Internet
  - An operators in the AS can not access to JPNIC ROA Web anymore, then they can't fix wrong ROAs

- I wonder this happens only in Japan???

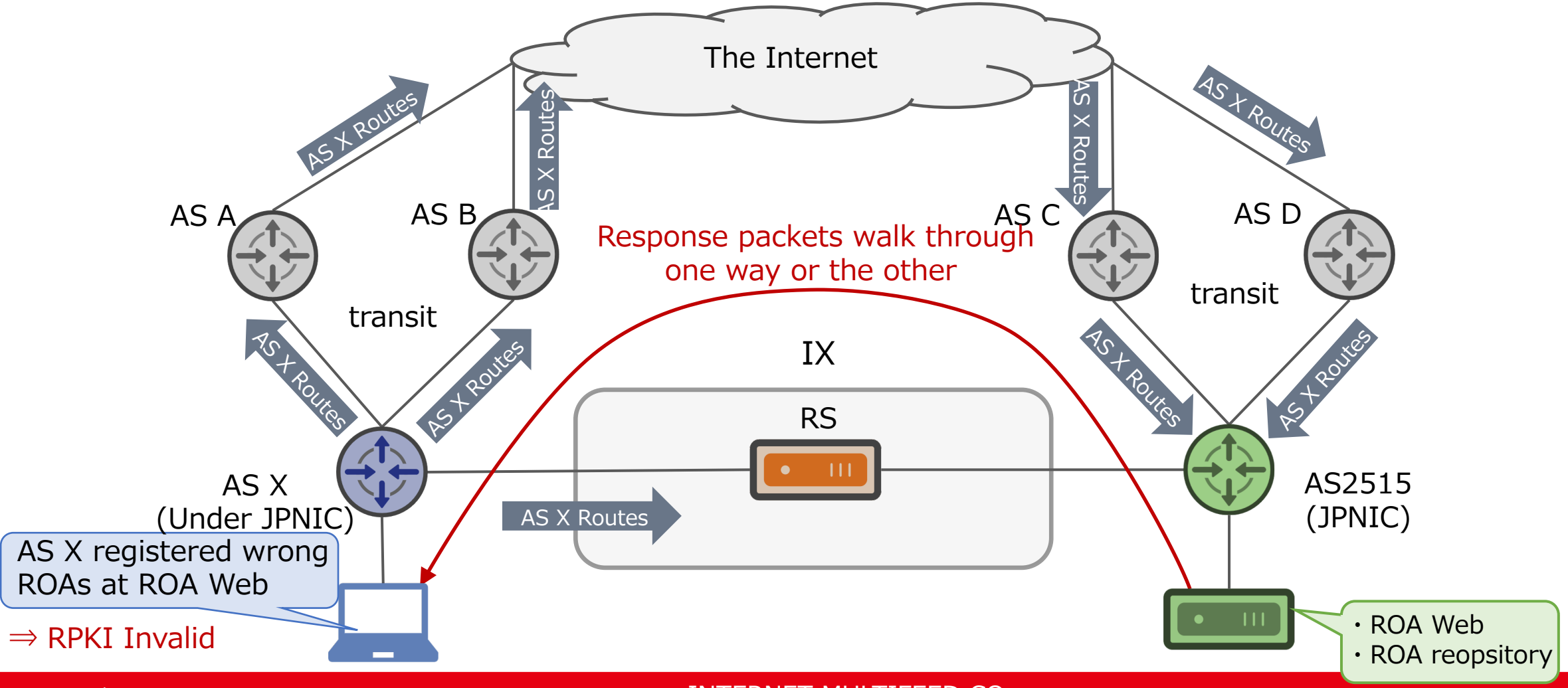
Heads-up: Scenario (1/3)

- AS X is assigned IP addresses by JPNIC
- The operators in AS X use JPNIC RPKI service “ROA Web” to register their ROAs



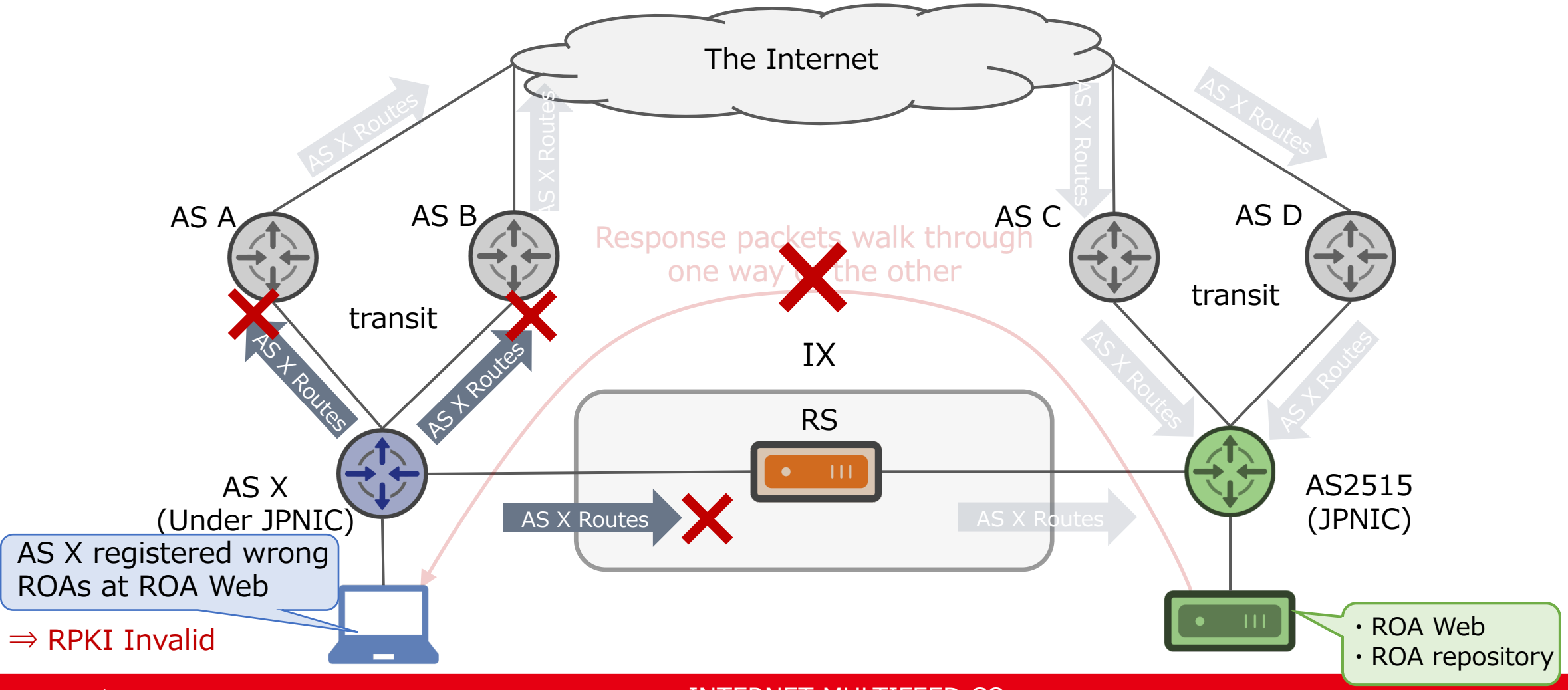
Heads-up: Scenario (2/3)

- AS X registered wrong ROAs for their own network to reach Internet
- The routes are result in Invalid



Heads-up: Scenario (3/3)

- If AS A and B, X's transit, and IX RS enable drops Invalids,
  then AS X loose reachability for Internet and even can not fix wrong ROAs at ROA Web



Heads-up: Consideration of this Issue

- Conditions
  - The issue might occur on an AS...
  - 1) who register inaccurate ROAs for their routes (This is root cause)
  - 2) whose all transit providers and connecting RSs implement ROV and drop Invalids
  - 3) who have no Private/Bilateral peer with JPNIC
    - Based on the assumption that JPNIC doesn't use APNIC TAL for their ROV
- Essence
  - Controls that directly affect the traffic exchange **over the Internet** are performed **through the Internet**
    - Just applying AS number or IP addresses do not affect to Internet routing
    - You might think same issue happens with IRR, but only few ISPs deploy IRR based filtering to their customers (especially in Japan), so the situation is different, I think.
- Note
  - This happens only in Japan? No!!
    - MyAPNIC is provided from AS4608
  - This doesn't matter to RS essentially
    - Without RS, all transit providers drop Invalids, same thing

Heads-up: Countermeasure

- Point: Keep connectivity to JPNIC ROA Web (or your RPKI service provider)

- AS operators can do...
  - Catch up the ROV adoption of their neighbor ASs (transit, IX)
  - Create or modify ROAs very carefully
  - Keep communication path, which not depended on their own network, for ROA Web
    - For example, tethering, public wi-fi, etc
    - Also keep credentials for login to RPKI system closely

- AS operators, cooperating with JPNIC, can do...
  - Establish Private/Bilateral peers between their AS and JPNIC AS
    - JPNIC is Open Policy

As an IXP operator, it's important for us to warn our customers of these kind of risks.

Summary

- Share our knowledge of ROV deploying
  - Design (Policy, Technology Selection, Parameters)
  - Test (ROV algorithm, RTR connections and Relying Party)
  - Introduction (Timeline, customer care, current routes status)
  - Operation (Monitoring, operation and tools)

- Introduce a (extreme) issue which might happen in near future
  - Do not make mistake in registering ROAs!
  - It’s important to warn your customers of these kind of issue

Thank you!!