Scaling AMS-IX Route Servers

David Garay

Supervisor: Stavros Konstantaras
Research Project 2, 2019

Motivation: Security

Zwitsers bedrijf routeerde KPN- en ander Europees verkeer via China Telecom

De Zwitserse colocatieaanbieder Safe Host heeft donderdag twee uur lang verkeer van meerdere Europese telecomaanbieders per ongeluk omgeleid via het netwerk van China Telecom. Onder andere KPN-verkeer verliep via de omweg.

Apnic beschrijft hoe Safe Host in het Duitse Frankfurt meer dan zeventigduizend *routes* via China Telecom liet verlopen. Daarbij ging het om dertienhonderd Nederlandse prefixes, waarvan veel van KPN. De omleiding begon voor KPN donderdagochtend rond 10.00, blijkt uit een grafiek van Oracle op basis van BGP Data. Na twee uur had Safe Host het probleem opgelost.

Donderdag bleek al dat KPN-klanten met internetproblemen te maken kregen. Ook pinverkeer ondervond hinder van de storing. Toen al gaven onder andere gebruikers op Gathering of Tweakers aan dat er sprake leek van bgp-hijacking. In een reactie zei KPN toen dat de fout lag bij een configuratiewijziging in het netwerk van een transitpartij in Zwitserland.

June 6, 2019

Large European Routing Leak Sends Traffic Through China Telecom

Doug Madory

Beginning at 09:43 UTC today (6 June 2019), Swiss data center colocation company AS21217 leaked over 70,000 routes to China Telecom (AS4134) in Frankfurt, Germany. China Telecom then announced these routes on to the global internet redirecting large amounts of internet traffic destined for some of the largest European mobile networks through China Telecom’s network. Impacts were seen by some of Europe's largest networks in Switzerland, Holland, and France among other countries.

… 4134 21217 21217 21217 21217 21217 21217 13237 1136

Motivation: Scalability

IXP	Clients	Connected to Route Server *	Update frequency
AMS-IX [1]	845	714	1 hour
DE-CX [2],[5] (Frankfurt)	870	846	6 hours
LINX [3] (London)	819	640	At least 3 hours [4]

** IPv4 only*

Security *requires* ***dynamic*** *configuration capabilities*

Background Information

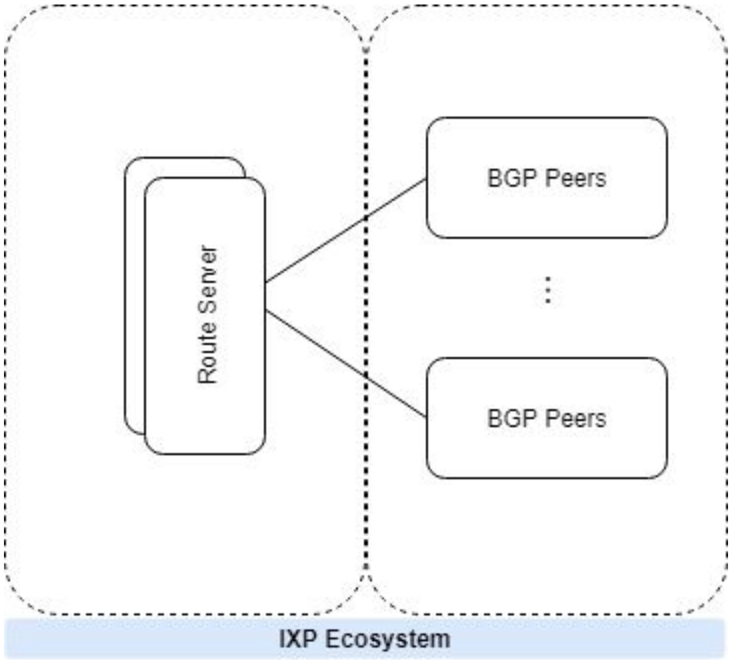


Fig 1: What is a Route Server?

- Central point for exchange of network prefixes, alternative to full-mesh topology.
- It filters prefixes exchanged, following policies configured by network operators.
- A route server is not a route reflector.

Background Information

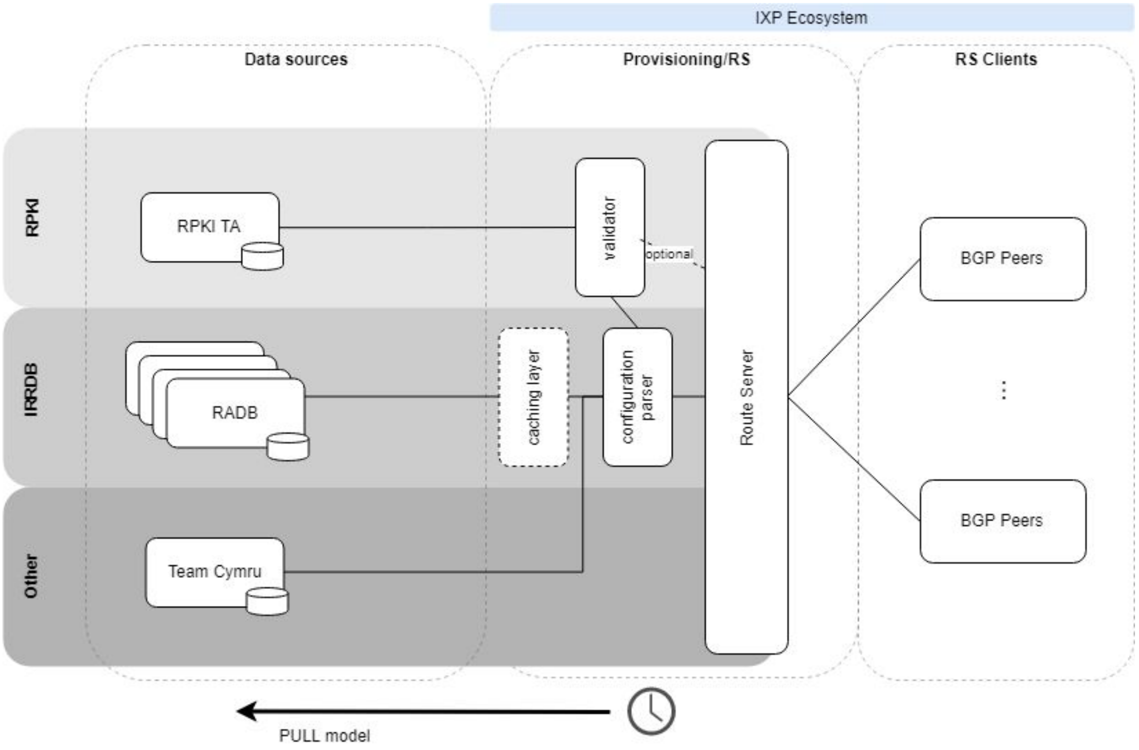


Fig 2: Data sources for a Route Server

Policies are periodically updated with dynamic data:

- **Internet Routing Registry DB:** source for whois information. Stores data using the *Routing Policy Specification Language* (RPSL).
- **Resource Public Key Infrastructure:** establishes the legitimacy of a prefix/autonomous system number ASN) pairing.
- **Team Cymru:** maintains the bogon reference.

Research Questions

- With regards to the route server's policy update process, what are the performance and scalability **performance indicators**? And what are the **bottlenecks** of the process, and what is their **impact**?
    - How can we improve these indicators in a new, feasible design?

Related Research

Problem Characterisation:

Jenda Brands and Patrick de Niet looked at BGP Parallelization, as a way to overcome the CPU bottlenecks which cause long converge times, present in Route Servers BGP implementations.

Solution Design:

Gregor Hohpe present patterns in Enterprise Integration Patterns that help designing messaging systems.

Methodology

- Current utilization
- Current setup evaluation and experiment design.
    - What are the bottlenecks and their impact?
- Solution design

Utilization in the last 6 months


```
$ curl https://stat.ripe.net/data/historical-whois/data.json?resource=1103
&sourceapp=os3_uva-sne-research
 {
    "status": "ok",
...
    "data": {
        "num_versions": 164,
        "resource": "1103",
        "version": "2019-07-03T19:59:00",
        "database": "RIPE",
        "suggestions": [],
        "versions": [
            {
                "version": 164,
                "from_time": "2019-01-04T10:36:23",
                "to_time": "2019-07-03T19:59:00"
            },
            {
                "version": 163,
                "from_time": "2018-12-12T09:52:20",
                "to_time": "2019-01-04T10:36:23"
            },
            {
                "version": 162,
                "from_time": "2018-06-12T11:42:55",
                "to_time": "2018-12-12T09:52:20"
            },
```


Fig 3: Number of changes per hour of relevant objects

- With the help of **RIPE's STATs**, we count every time a object *aut-num* and *route* change, and aggregate them per hour.
- Note: not every policy change and route/prefixes is **relevant** to our IXP.
- Only AMS-IX clients, and prefixes in the route servers where used.

Utilization in the last 6 months

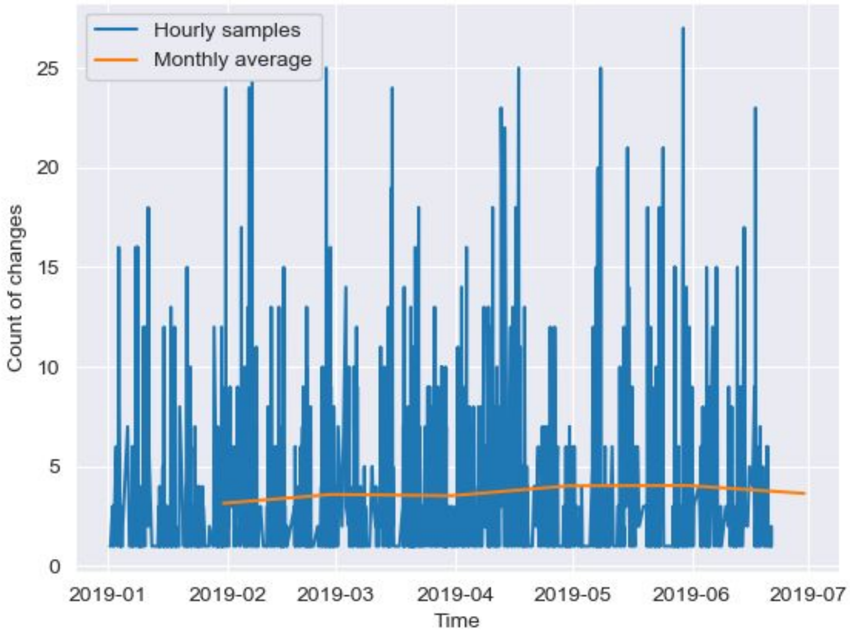


Fig 4: Number of changes per hour of relevant objects

How often are relevant changes happening?

- Dimensioning decision based on monthly averages or peaks?

Setup and experiment design

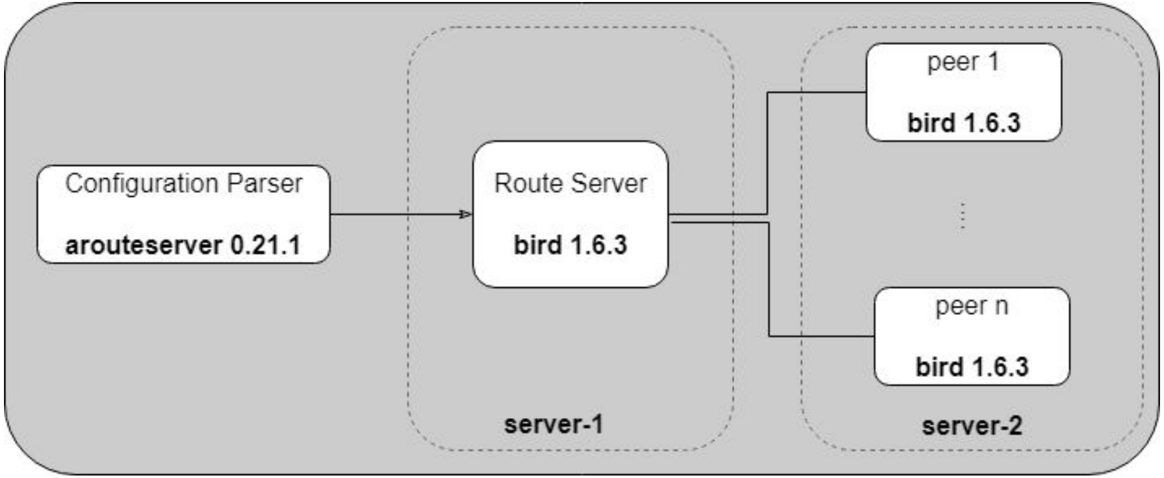


Fig 5: Experiments setup

We monitored the effects of policy updates on CPU, memory and traffic. We designed three experiments:

- Route server reconfigurations with different **file sizes**;
- Route server reconfigurations, where **BGP updates** were triggered;
- Route server peering with a large number of peers (>1100).

Results

Experiments	Result	Tooling / Remarks
Reconfiguration time as result of file size	~0,3s per 10MB file size increase	ars issue #48
Reconfiguration time as result of BGP update traffic	~ 0,5s per additional peer	
CPU utilization as result of the number of peers	Crash at 1013 peers in our setup	Ulimit configuration - insufficient system resources.

Reconfiguration time vs Number of Peers

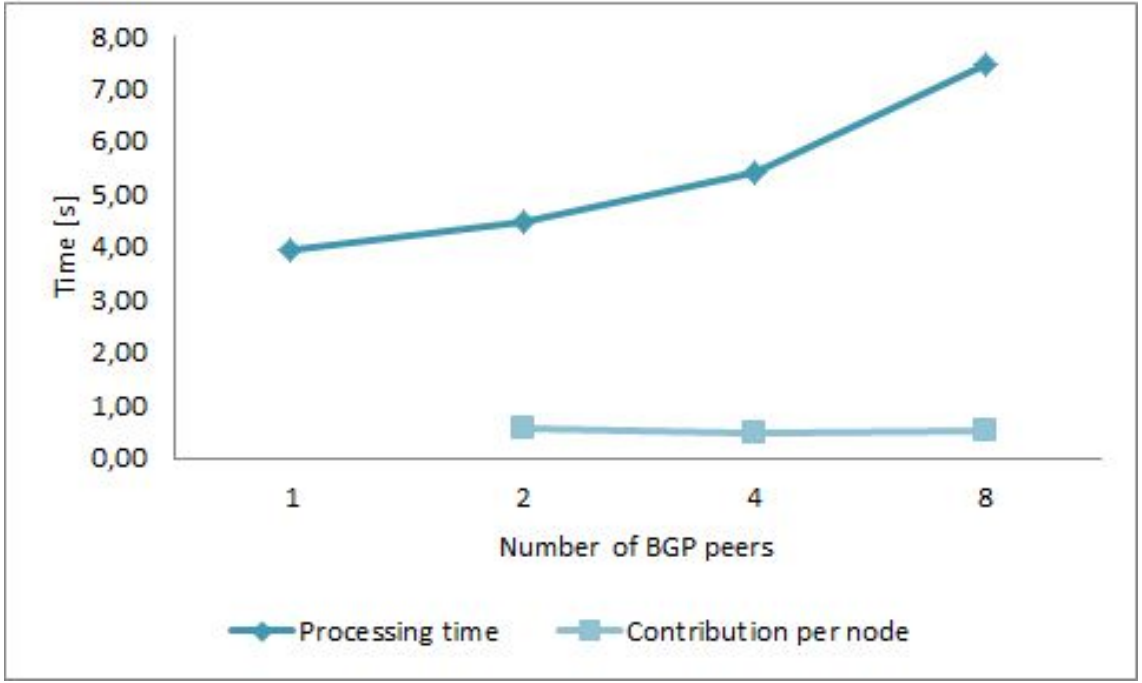


Fig 7: Reconfiguration time vs number of peers sending BPG updates as result of policy change, contribution per peer

Summary of challenges

- Policy updates are **not applied in real-time**.
- Updates cause high CPU utilization, **blocking** the Route Server to new tasks.
    - If moving to a information Push model, route server might be busy.
- Network load increase as result of updates

Application Integration Alternatives

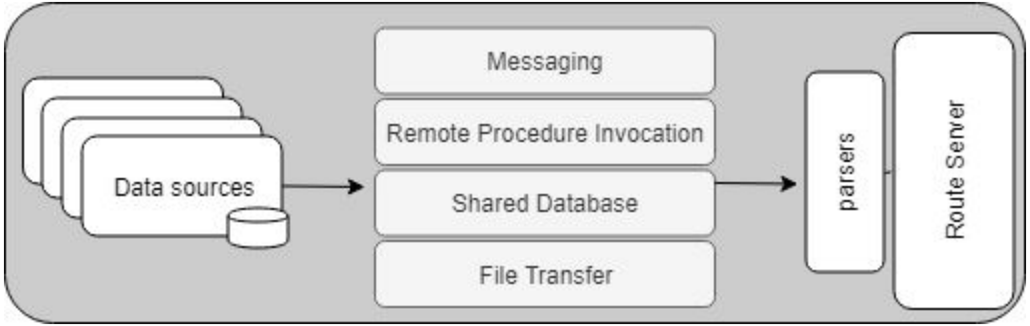


Fig 8: Integration alternatives

Data Transfer:

File Transfer and Shared Database.

Disadvantages: stale data, or if polling in use, inefficient use of resources.

Invoke remote functionality:

Remote Procedure Invocation(RPI) and Messaging.

Application Integration Alternatives

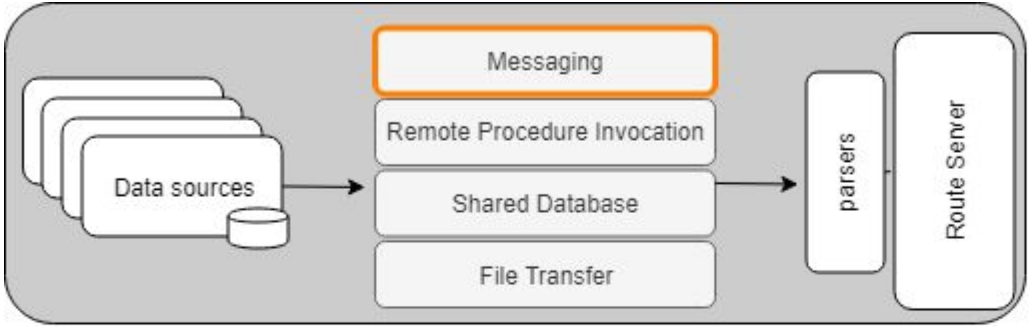


Fig 8: Integration alternatives

- With RPI, we have up to **NxM IXPs and ASNs**, simultaneous processes at the data source.
    - Addressing, failures and performance are not transparent.
- **Messaging** offers loose-coupling asynchronous communications.

Application Integration Alternatives

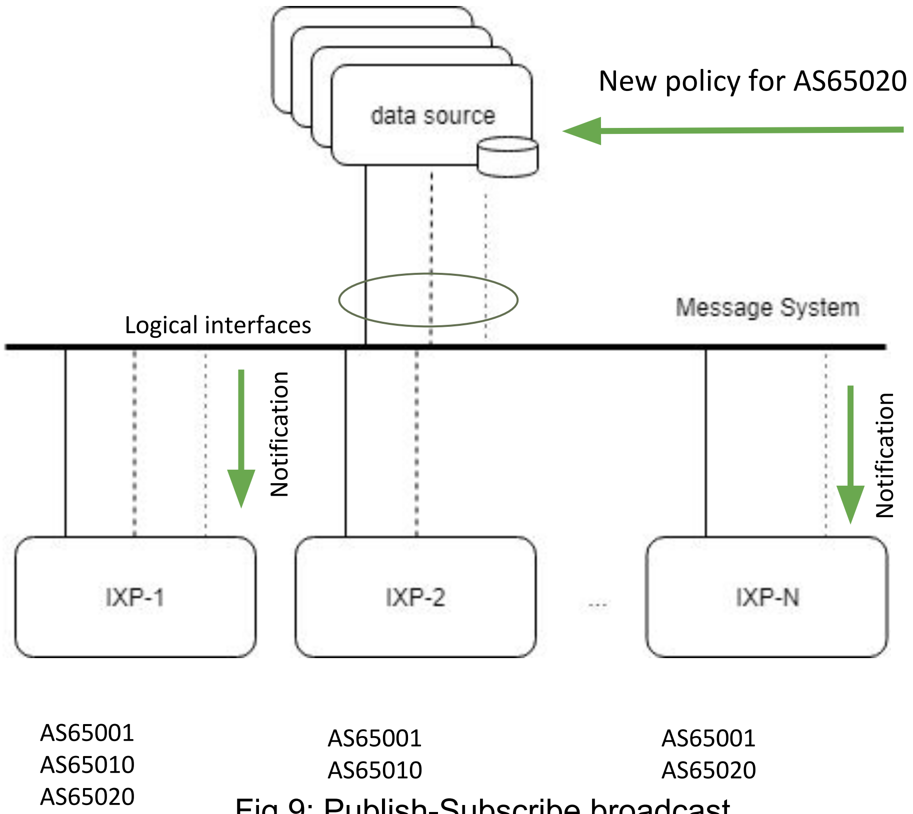


Fig 9: Publish-Subscribe broadcast

With a Messaging system, broadcast of messages is more efficiently.

- In a **Publish-Subscribe**r channel, clients receive real-time notifications about topics they have subscribed to.
- In our example, when AS65020 changes its policy, interested IXPs can receive it immediately.
- Messages remain in the system until consumed, or expire.

Proposed design: New functionalities

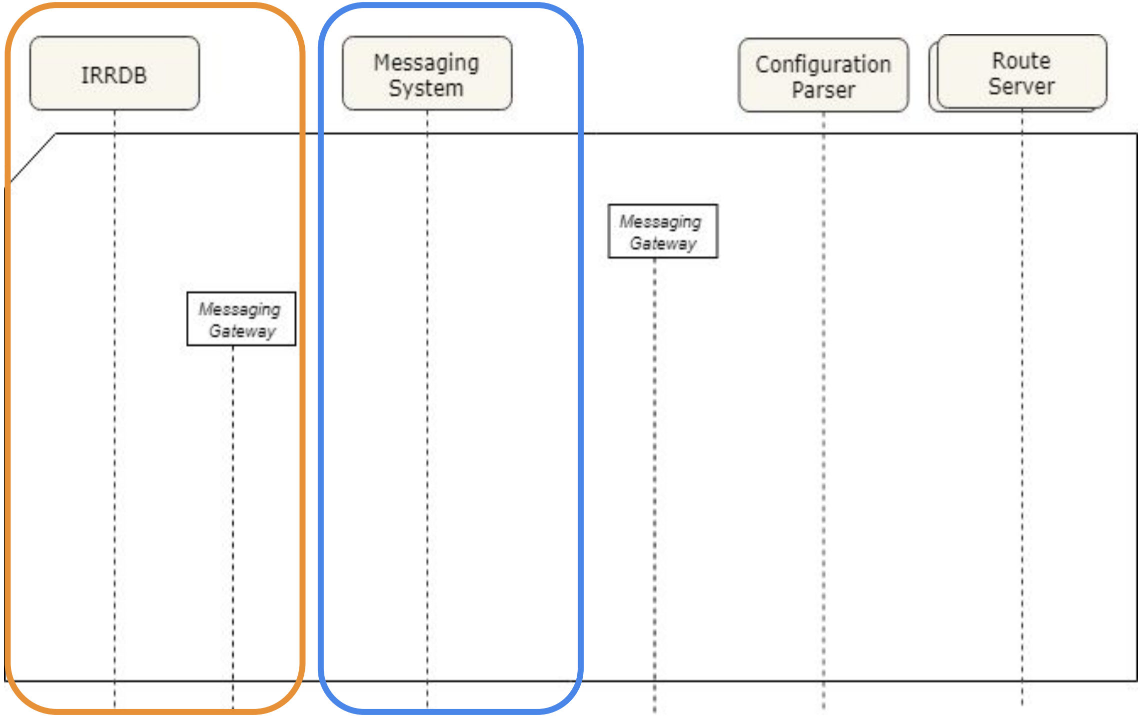
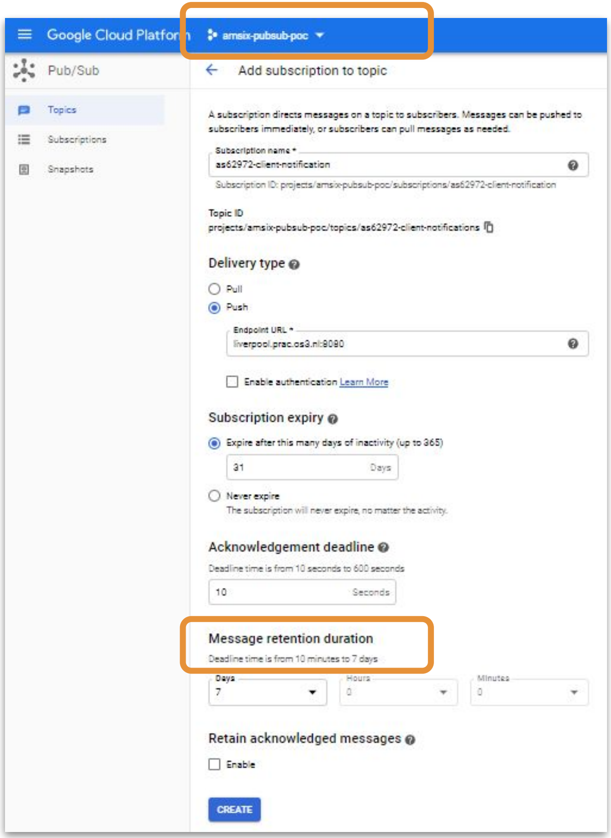


Fig 10: Sequence diagram - Policy updates push model

Modifications required:

- Message Gateway.
- Messaging system.

Example: Google PubSub



C# GO JAVA NODE.JS PHP PYTHON RUBY

VIEW ON GITHUB FEEDBACK

```python
from google.cloud import pubsub_v1

# TODO project_id = "Your Google Cloud Project ID"
# TODO topic_name = "Your Pub/Sub topic name"

publisher = pubsub_v1.PublisherClient()
# The `topic_path` method creates a fully qualified identifier
# in the form `projects/{project_id}/topics/{topic_name}`
topic_path = publisher.topic_path(project_id, topic_name)

for n in range(1, 10):
    data = u'Message number {}'.format(n)
    # Data must be a bytestring
    data = data.encode('utf-8')
    # When you publish a message, the client returns a future.
    future = publisher.publish(topic_path, data=data)
    print('Published {} of message ID {}.'.format(data, future.result()))

print('Published messages.')
```

Fig 11: Messaging system example (left) and client (right)

Proposed design: Policy updates procedure

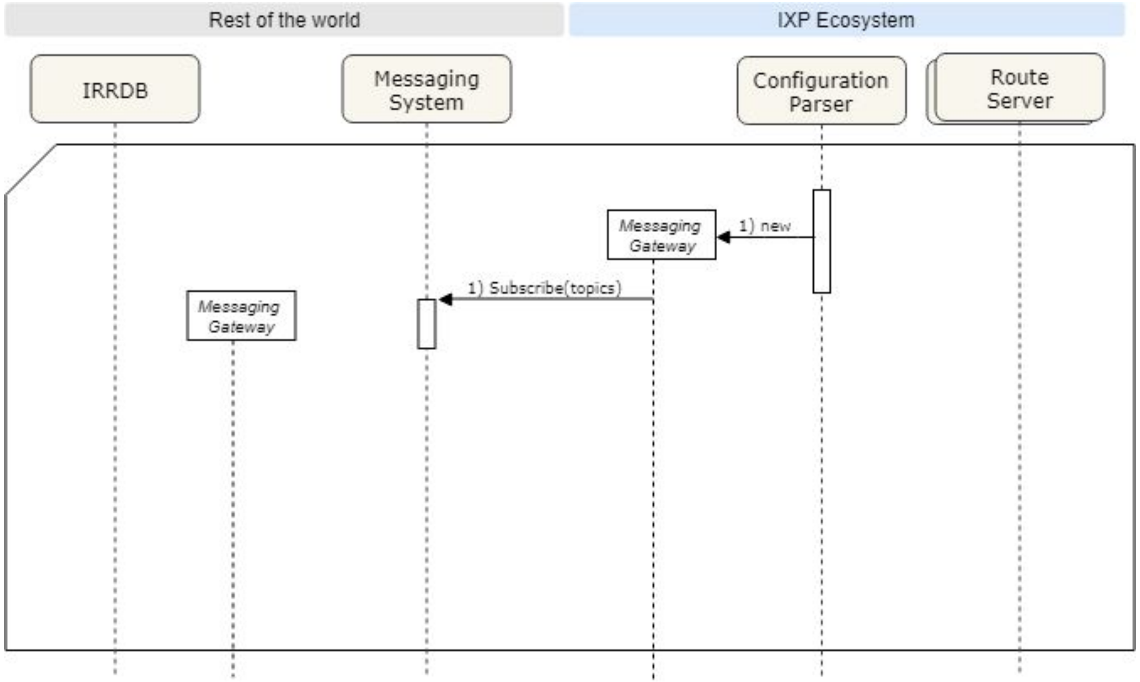


Fig 12: Sequence diagram - Policy updates push model

To receive policy change notifications, a client subscribes to the topic of the respective ASN.

- Transport options depend on Messaging System implementation, and message format remain RPSL to leverage existing tools

Proposed design: Policy updates procedure

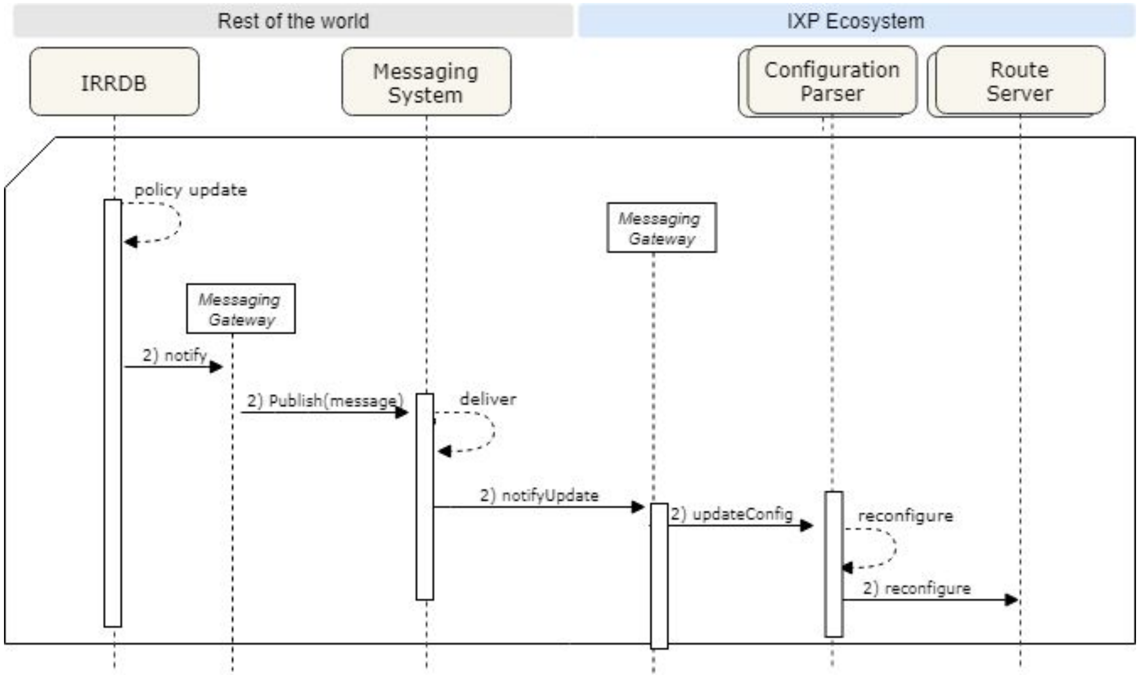


Fig 13: Sequence diagram - Policy updates push model

Notifications are received in real-time.

- Duplicated messages policy, throttling and parallelization are handled at the client’s Messaging Gateway.

Architecture Vision

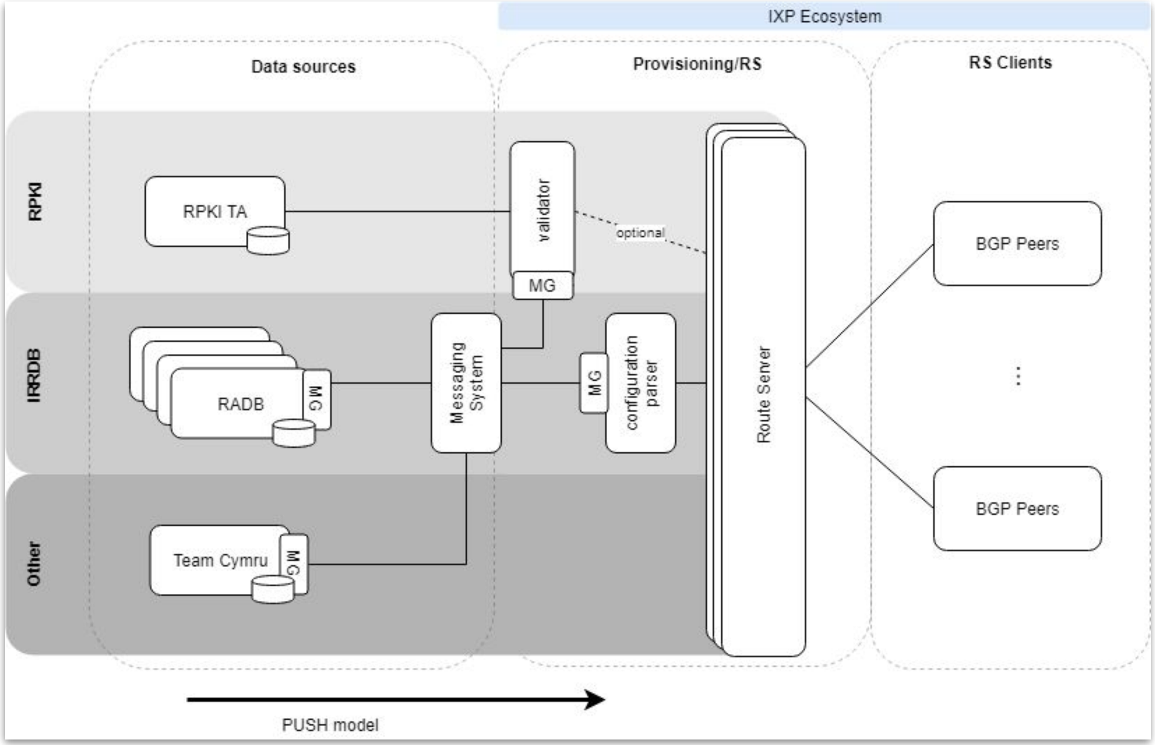


Fig 14: Architecture vision

Discussion

- **Design**
  - Does it address the real-time and throttling requirements?
  - Is the design future proof?
  - Is there justification for a Message System?
- **Limitations in our methodology**
  - Limited usa cases evaluated
  - Validation against production statistics, simulation in scale.

Conclusion

- In our experiments, we found that the route server ***blocks*** as result of policy updates. The blocking time depends on the file size and on the amount of peers undergoing BGP Update procedures.
- We propose a **messaging** based design which addresses the lack of real-time policy updates, we discuss the component required and discuss how ***throttling and queueing*** can help alleviate the impact of the BGP policy updates.
- Our statistics regarding rate of policy updates are limited in the amount of objects monitored, and we recommend IXPs to *perform measurements in production* on policy changes to assess their impact on the network.

Future Work

- Improve Bird’s reconfiguration efficiency by evaluating Binary configuration formats
- Study other use cases (e.g. Policy implementation feedback)
- Extend statistical investigation to include IPv6 objects, and other objects.

Backup

Reconfiguration time vs Number of Peers

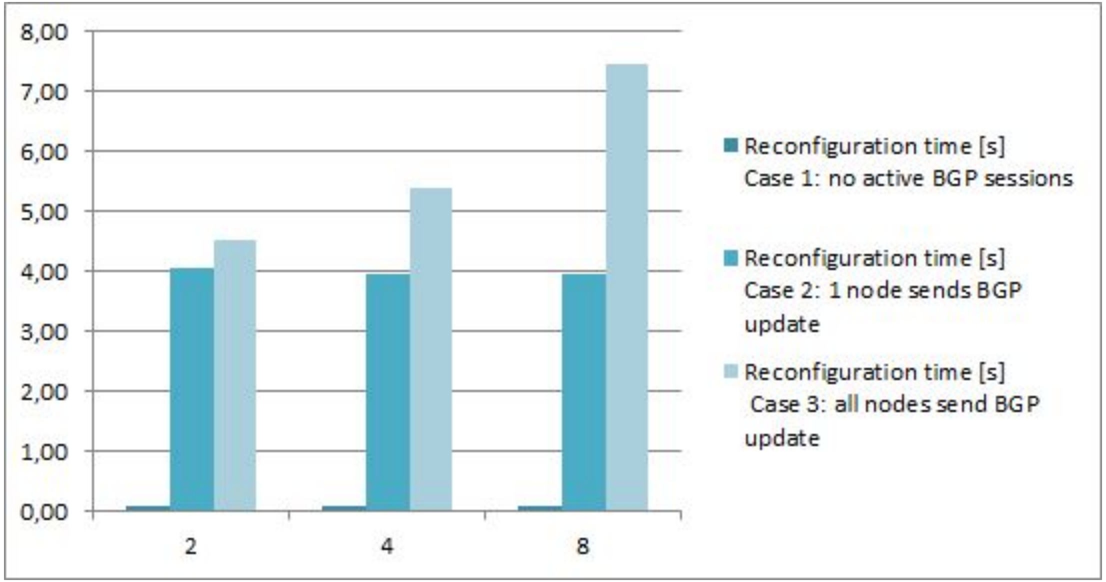
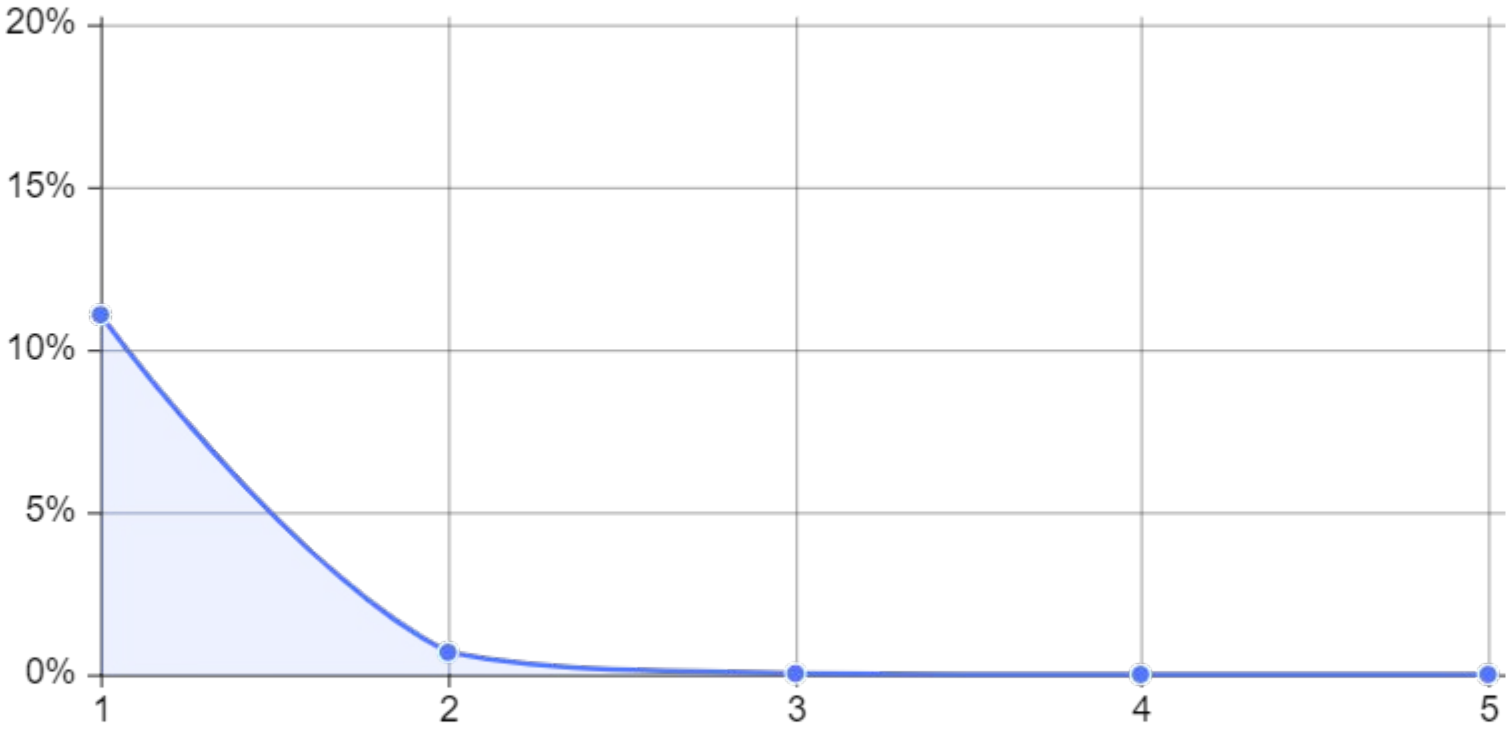


Fig 7: Reconfiguration time vs number of peers sending BPG updates

Erlang B: 28 arrivals, ~16s processing, 1 server



source

Utilization in the last 6 months

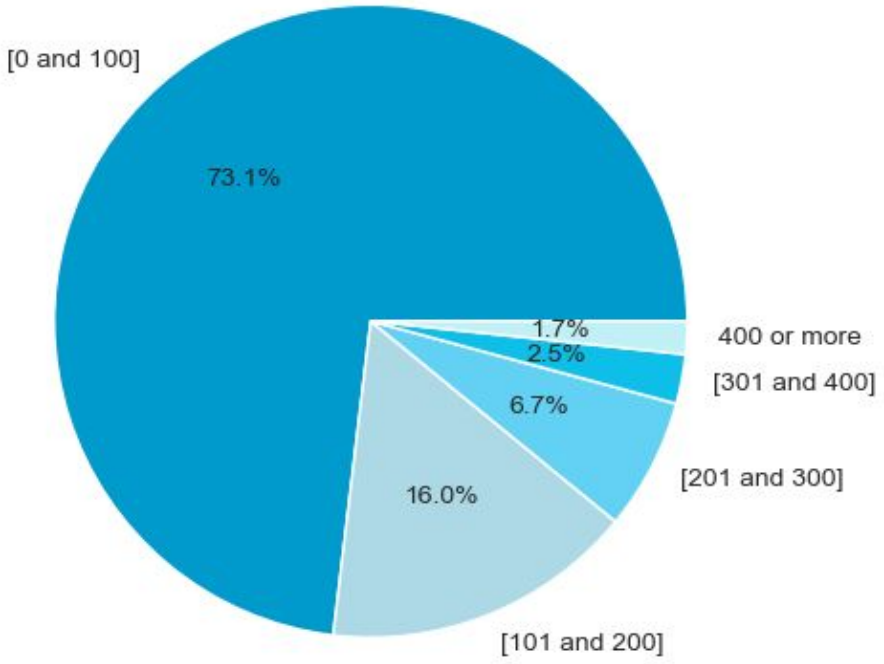


Fig 4: Frequency of changes, in ranges of 100, in the last 6 months

Where are the events coming from?

These are the percentage of networks doing 0-100 changes, 101-200... ; in the last 6 months.

- Most relevant events come from few network operators.

Who is using arouteserver?

- BharatIX, BIRD.
- CATNIX, BIRD.
- IX-Denver, BIRD.
- MBIX, BIRD.
- PIT-IX, BIRD.
- SwissIX, OpenBGPD.
- Unmetered.Exchange, BIRD.
- VANIX.
- YXEIX, BIRD.
- YYCIX, OpenBGPD.

Are you using it? Do you want to be listed here? Drop me a message!

Fig : Frequency of changes, in ranges of 100, in the last 6 months

Reconfiguration time vs File size

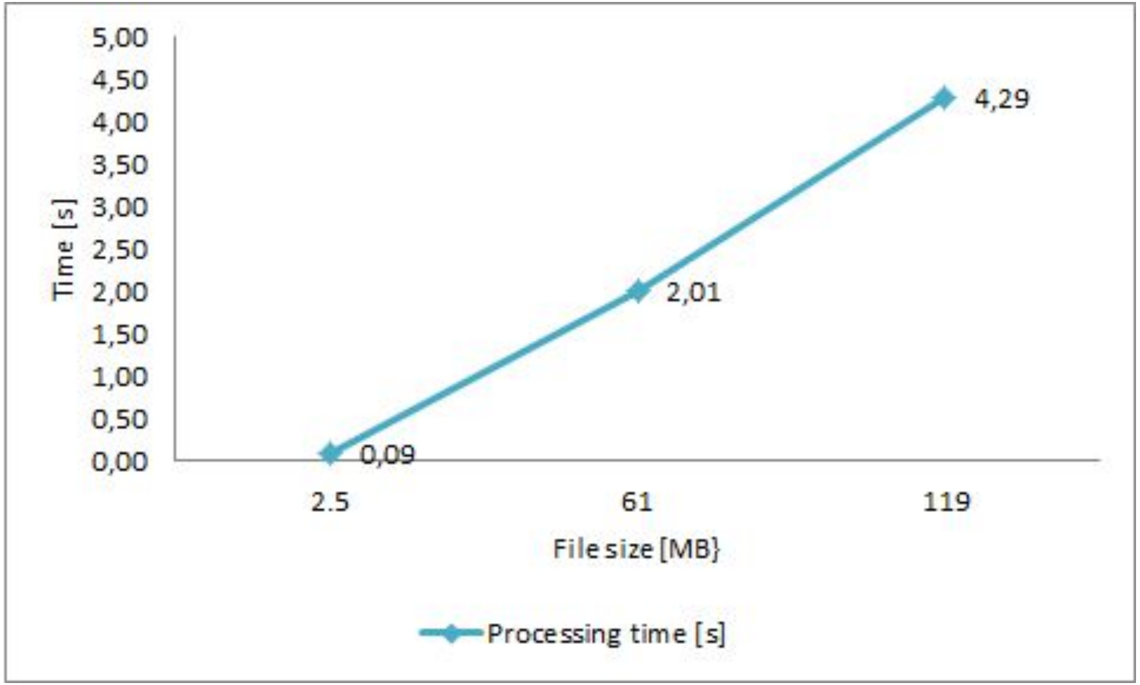


Fig 6: Reconfiguration time vs file size