bgproutes.io

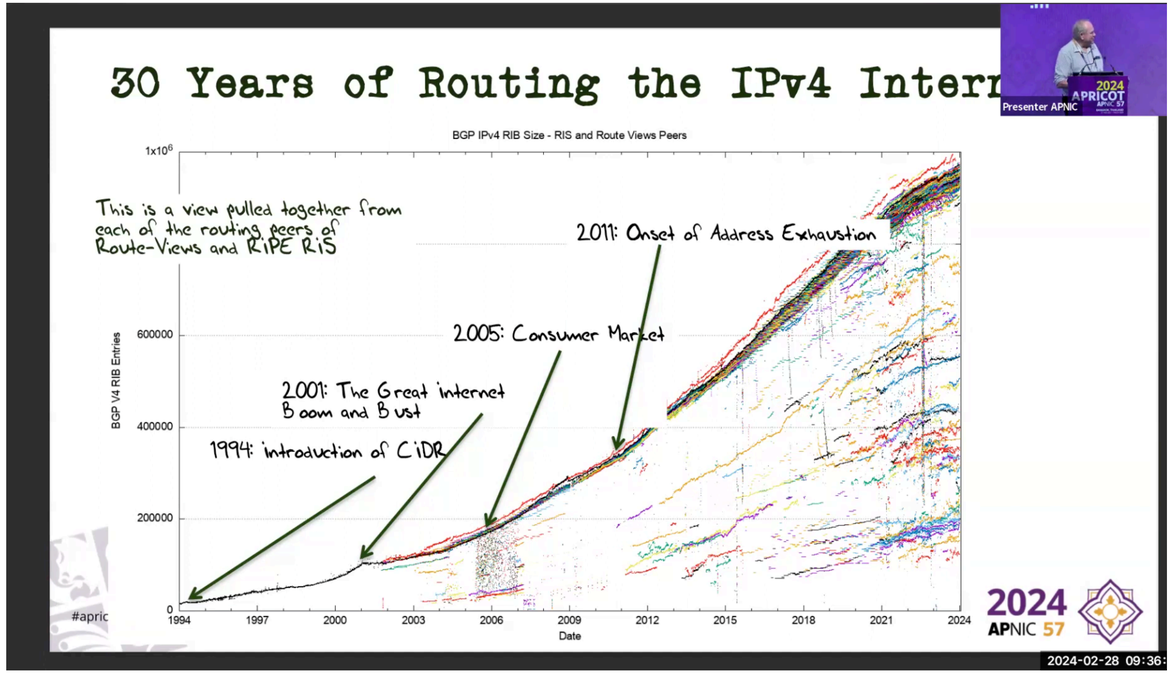
The Next Generation of BGP Data Collection Platforms

Thomas Holterbach
University of Strasbourg

Current sponsors:

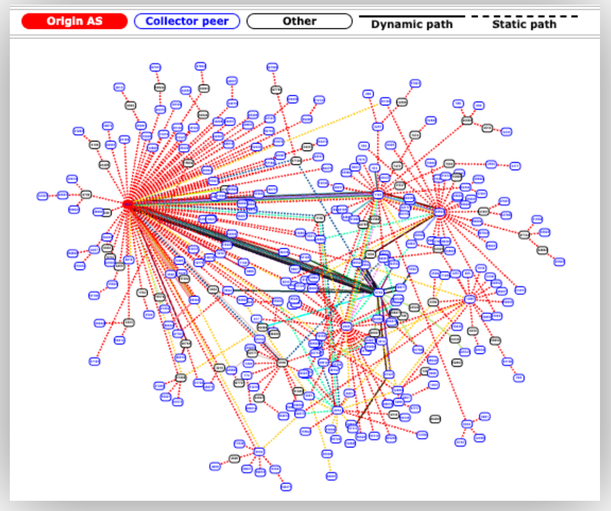
Joint work with:
Thomas Alfroy
Cristel Pelsser

The BGP data collected from actual BGP routers is essential for understanding and monitoring Internet routing



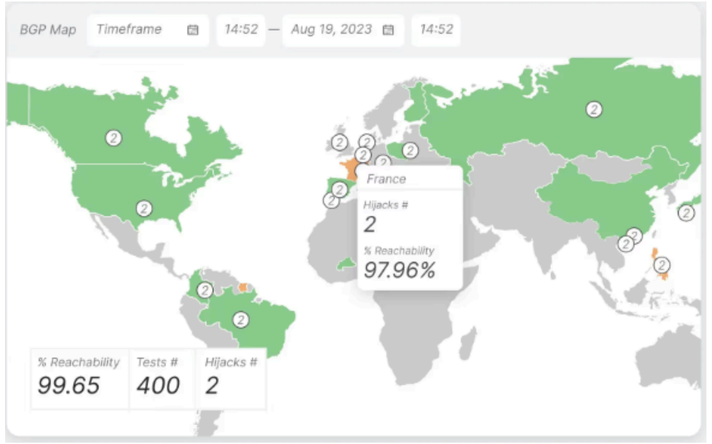
Geoff Huston: BGP in 2023

APRICOT'24



BGPlay

RIPEStat



Catchpoint's BGP map

Routers that share their BGP routes are like satellites observing the Earth's surface

Each router (or "*vantage point*") provides a partial view

But together they allow monitoring at larger scale

The three longstanding platforms that collect, archive, and publicly share BGP data are RIPE RIS, RouteViews and PCH

- RIS and RouteViews collect and store data from ~1500 BGP routers (*a.k.a.* "*vantage points*")

bgproutes.io

Our new next-gen BGP route collection platform

https://bgproutes.io

You can contribute data

bgproutes.io

Our new next-gen BGP route collection platform

We launched it
a year ago

https://bgproutes.io

You can contribute data

BGP data collection platforms typically pursue two main objectives

High coverage	Perpetual data retention

BGP data collection platforms typically pursue two main objectives **that are not easily compatible**

Priority	
High coverage	Perpetual data retention

BGP data collection platforms typically pursue two main objectives that are not easily compatible

	Priority	
	High coverage	Perpetual data retention
RIPE RIS RouteViews	Lower priority	High priority 

BGP data collection platforms typically pursue two main objectives that are not easily compatible

	Priority	
	High coverage	Perpetual data retention
RIPE RIS RouteViews	Lower priority	High priority 
bgproutes.io	High priority 	Lower priority

BGP data collection platforms typically pursue two main objectives that are not easily compatible

	Priority	
	High coverage	Perpetual data retention
RIPE RIS RouteViews	Lower priority	High priority 
bgproutes.io	High priority 	Lower priority

Our goal is to have accurate real-time
and short-term historical monitoring tools

bgproutes.io

Our new next-gen BGP route collection platform

- Collection: High coverage
- Distribution: API + Dashboards

https://bgproutes.io

You can contribute data

bgproutes.io

Our new next-gen BGP route collection platform

- Collection: High coverage
- Distribution: API + Dashboards

https://bgproutes.io

You can contribute data

bgproutes.io relies on three ingredients to increase its coverage

Ingredient #1

Ingredient #2

Ingredient #3

bgproutes.io relies on three ingredients to increase its coverage

Ingredient #1 | Automated peering establishment

Ingredient #2 |

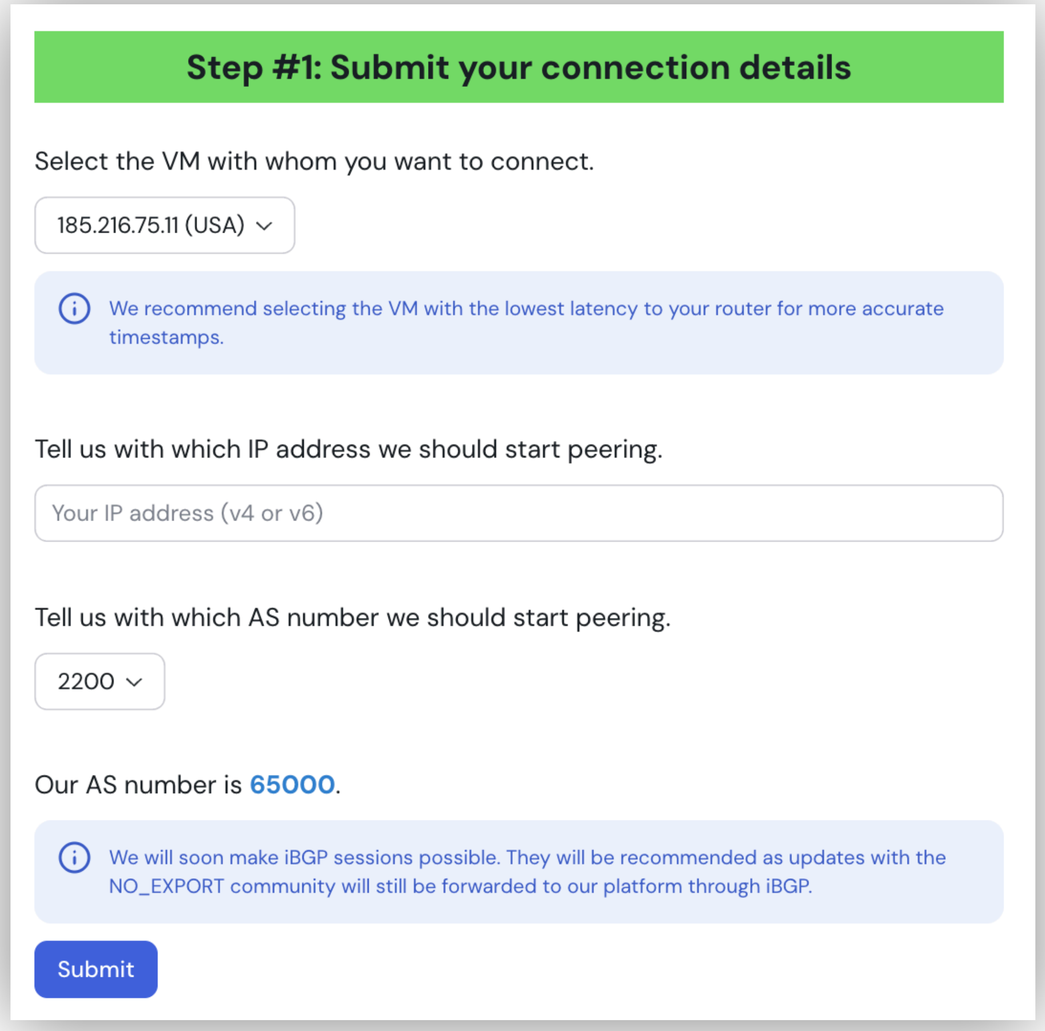
Ingredient #3 |

bgproutes.io simplifies, automates
and opens BGP data contribution to every ASN

bgproutes.io simplifies, automates and opens BGP data contribution to every ASN

Network operators can authenticate using peeringDB

Network operators just have to fill a form to start peering with bgproutes.io



bgproutes.io/contribute

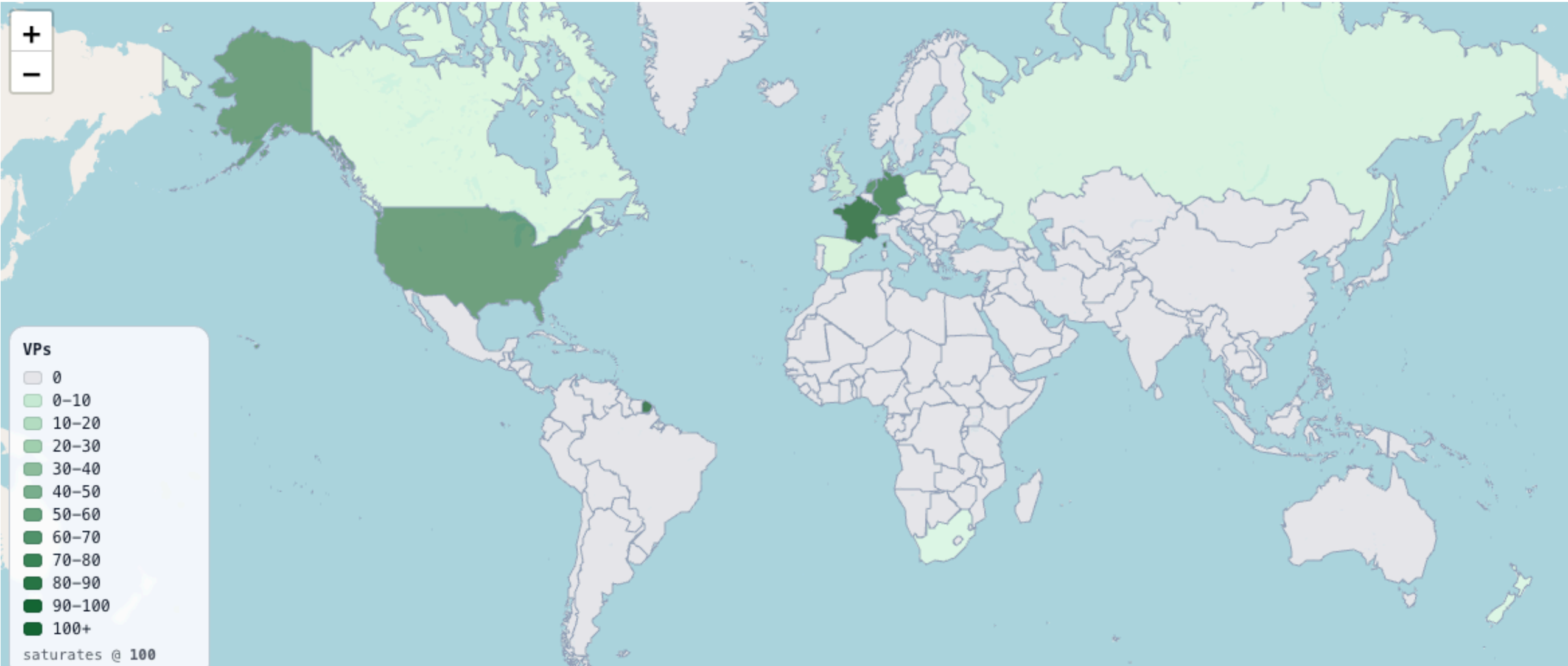
After one year, bgproutes.io receives BGP data from ~70 networks

After one year, bgproutes.io receives BGP data from ~70 networks

This is useful, but far from sufficient for accurate monitoring

After one year, bgproutes.io receives BGP data from ~70 networks

This is useful, but far from sufficient for accurate monitoring



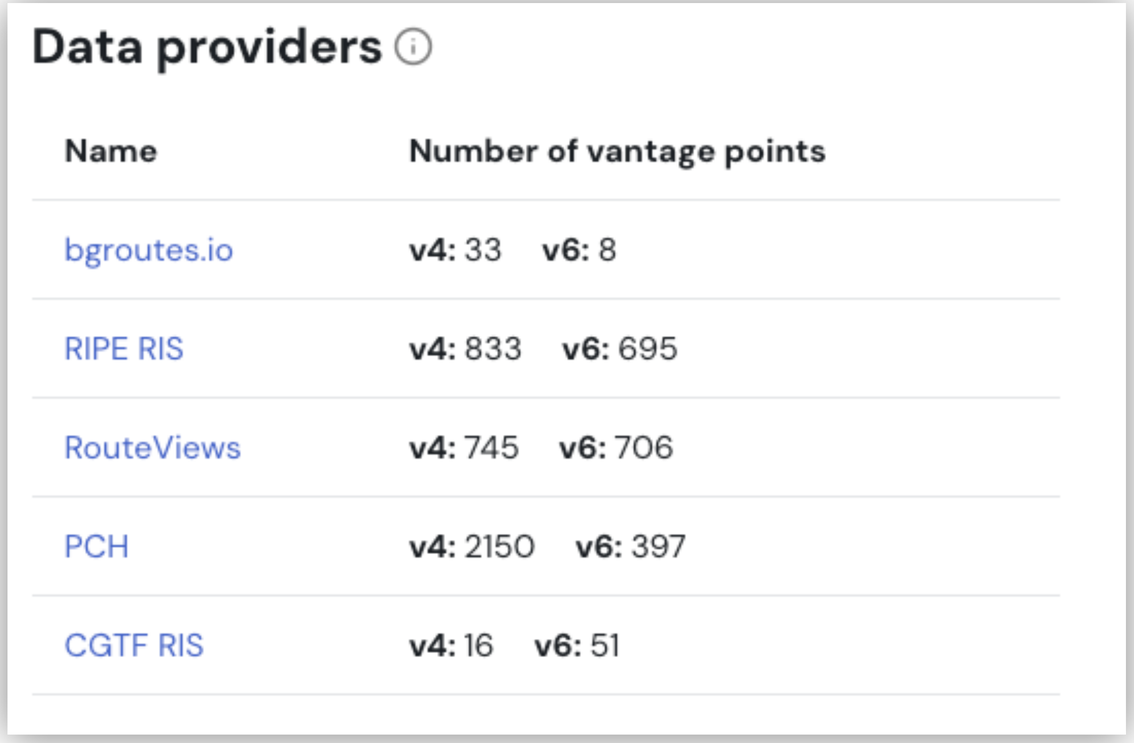
bgproutes.io relies on three ingredients to increase its coverage

Ingredient #1 | Automated peering establishment

Ingredient #2 | Data aggregation

Ingredient #3 |

bgproutes.io increases coverage by centralising all data collected by RIPE RIS, RouteViews, PCH, CGTF into a single database

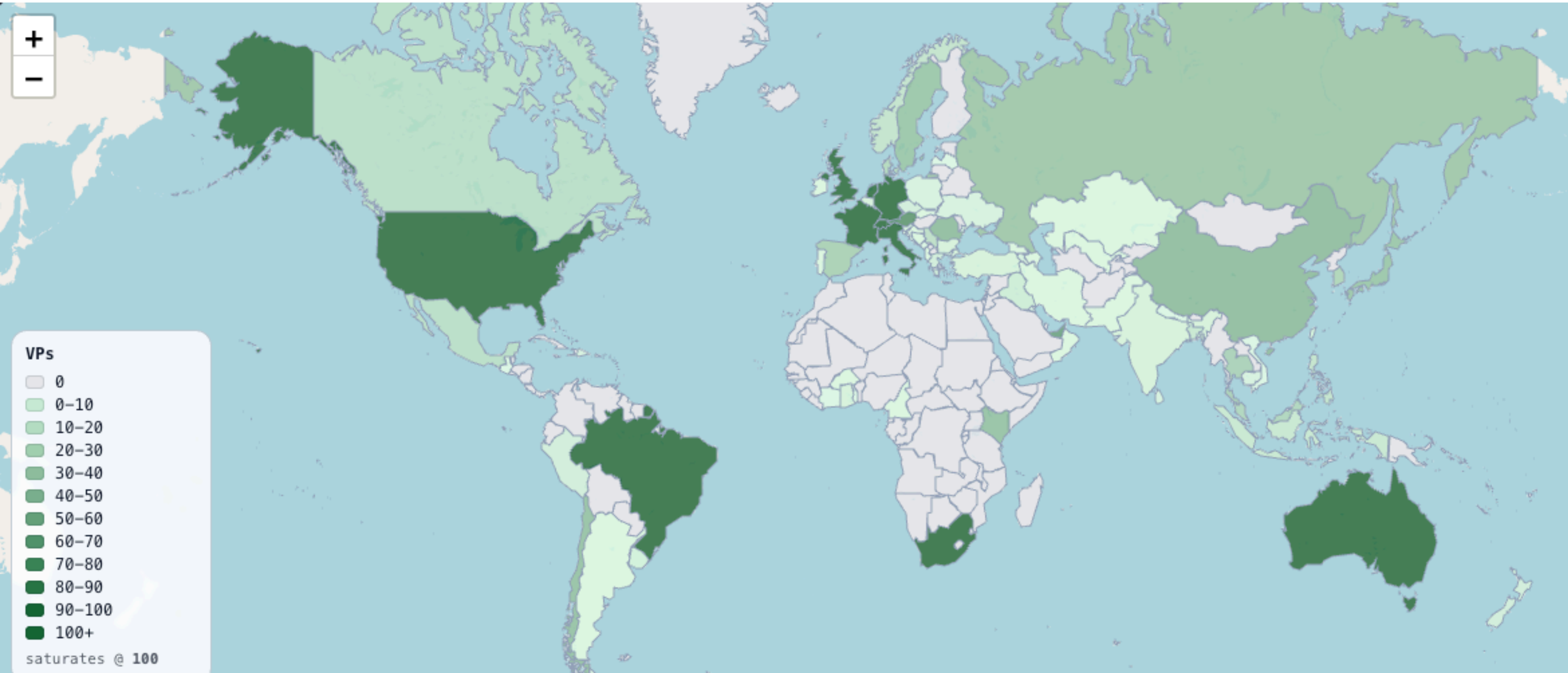


Data providers

Name	Number of vantage points
bgroutes.io	**v4:** 33 **v6:** 8
RIPE RIS	**v4:** 833 **v6:** 695
RouteViews	**v4:** 745 **v6:** 706
PCH	**v4:** 2150 **v6:** 397
CGTF RIS	**v4:** 16 **v6:** 51

https://vp-explorer.bgproutes.io

bgproutes.io already stores data
for more than 5000 vantage points



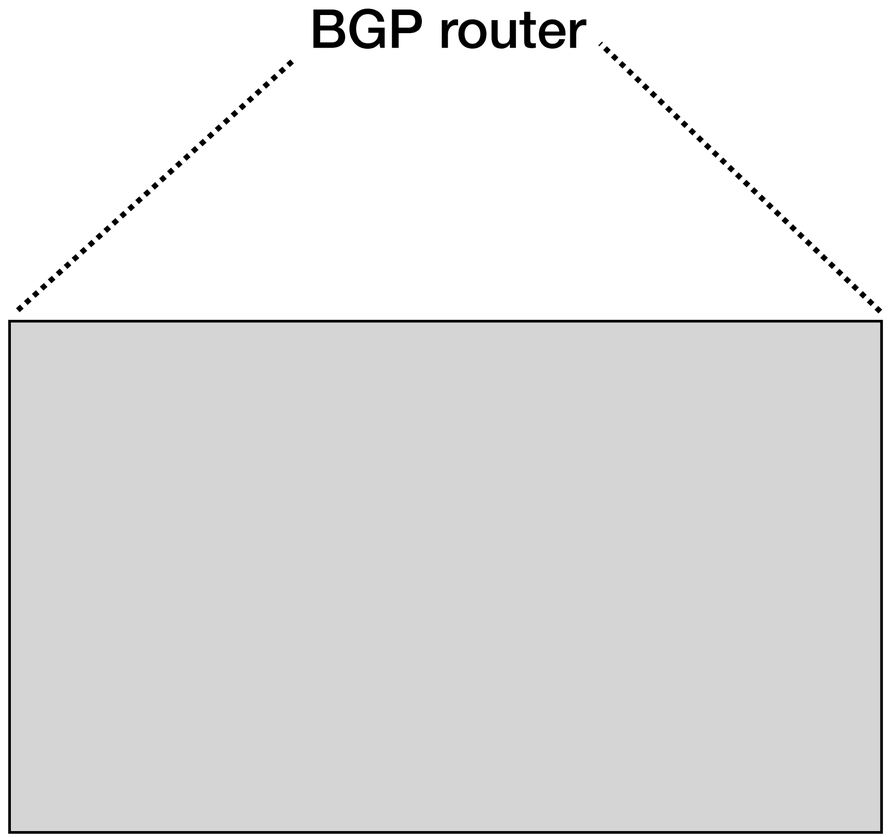
bgproutes.io relies on three ingredients to increase its coverage

Ingredient #1 | Automated peering establishment

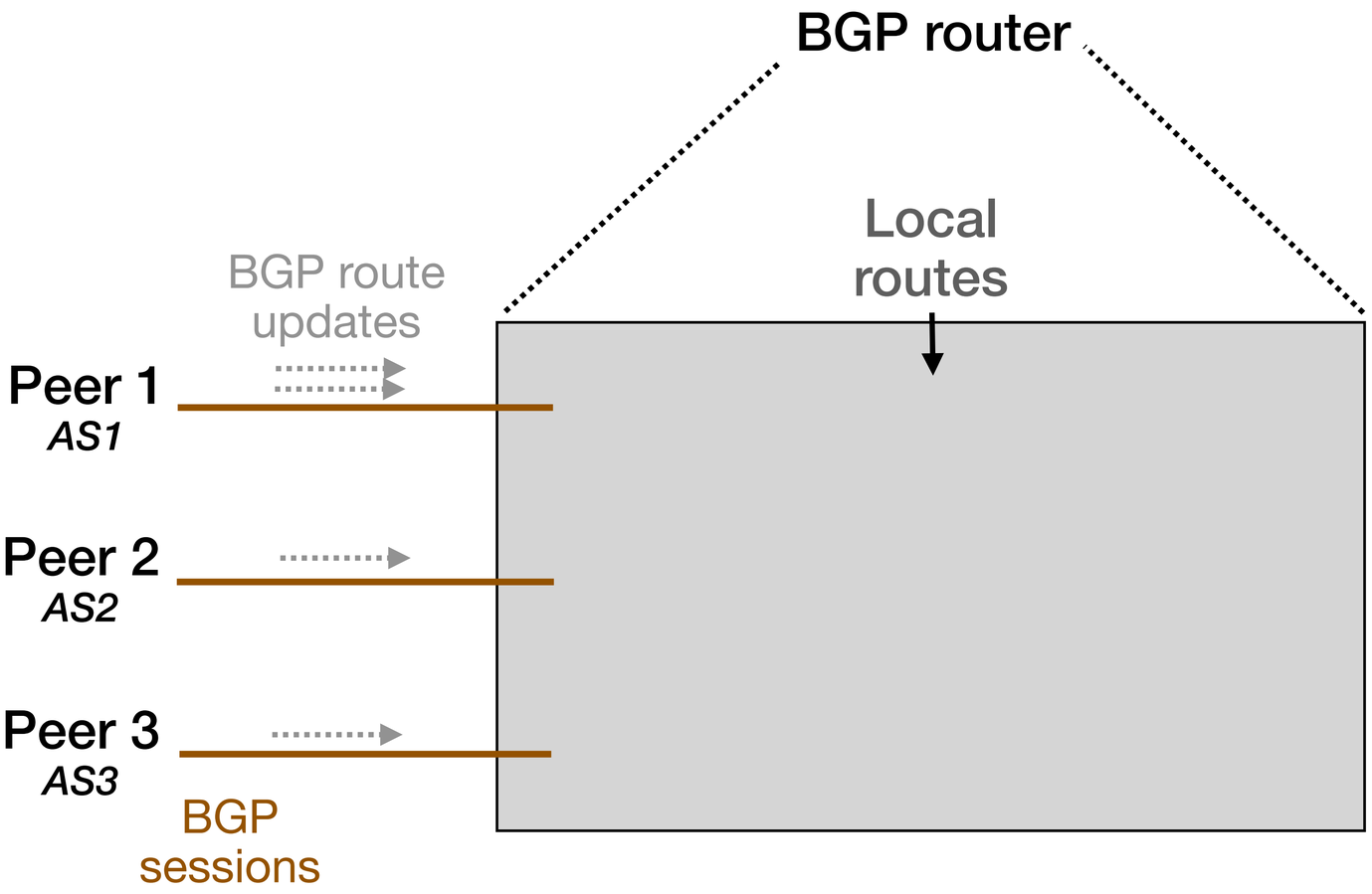
Ingredient #2 | Data aggregation

Ingredient #3 | **BMP**

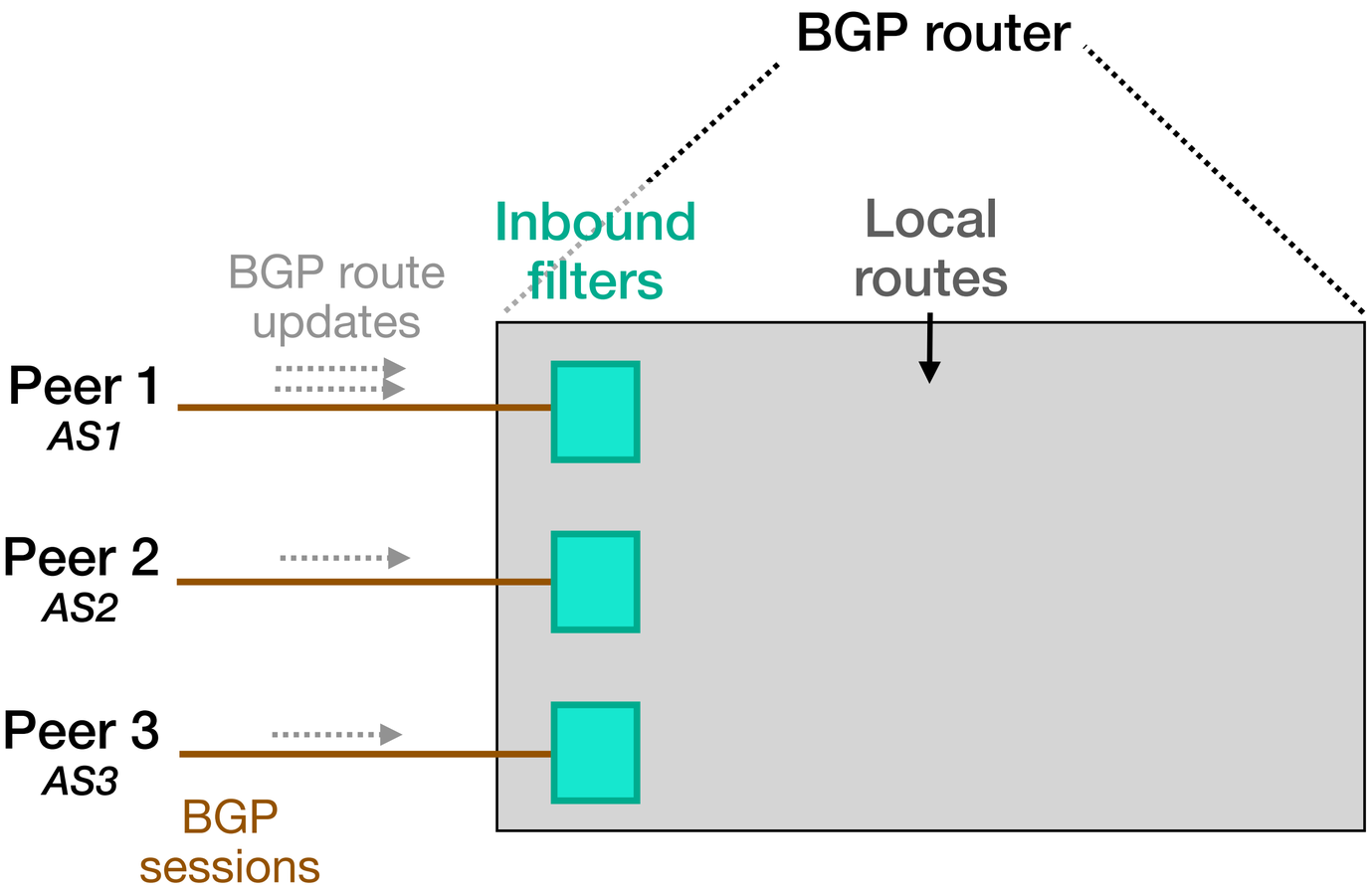
Problem: When an operator shares data via a BGP session,
only a small fraction of its routes are visible due to BGP route hiding



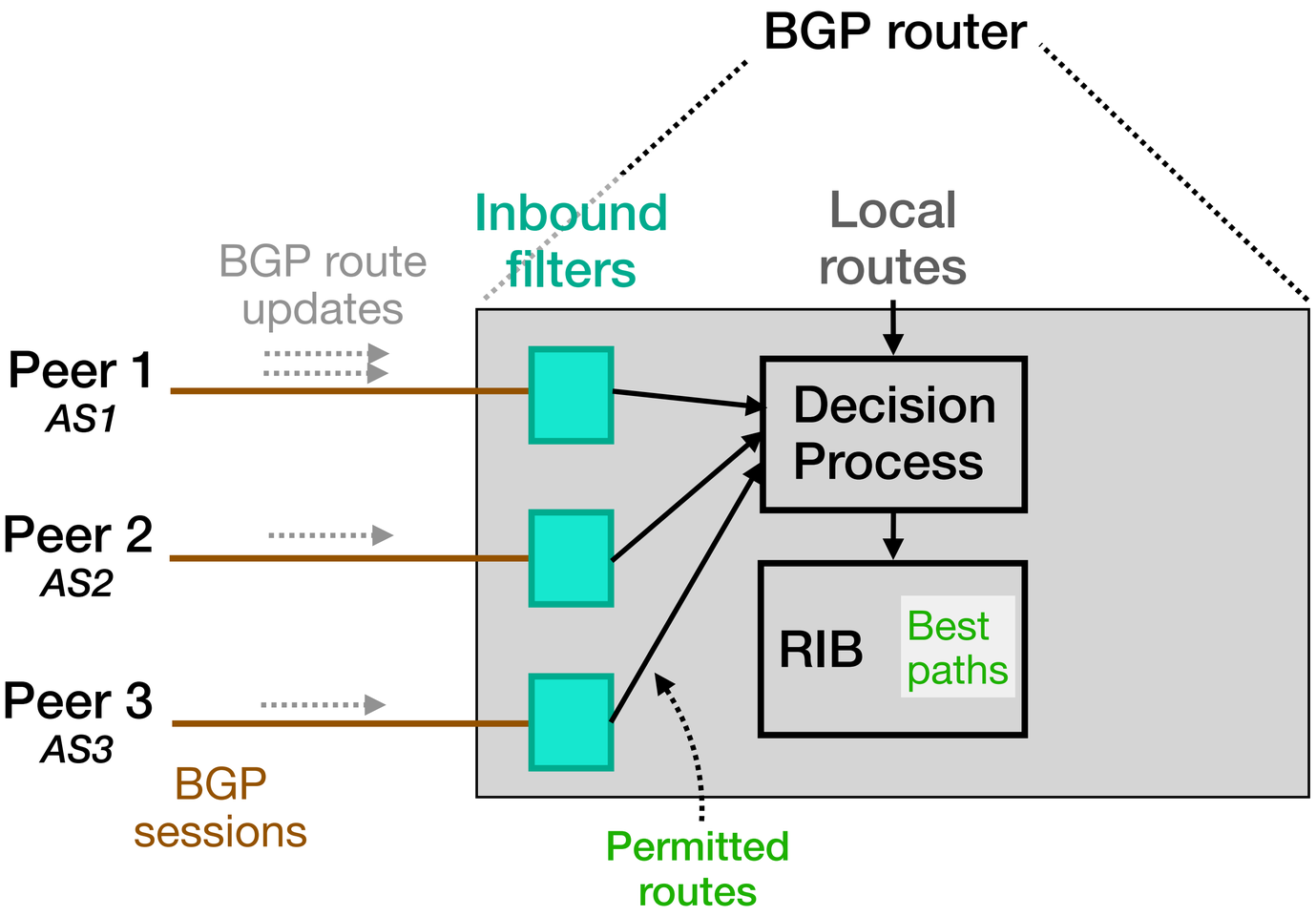
Problem: When an operator shares data via a BGP session, only a small fraction of its routes are visible due to BGP route hiding



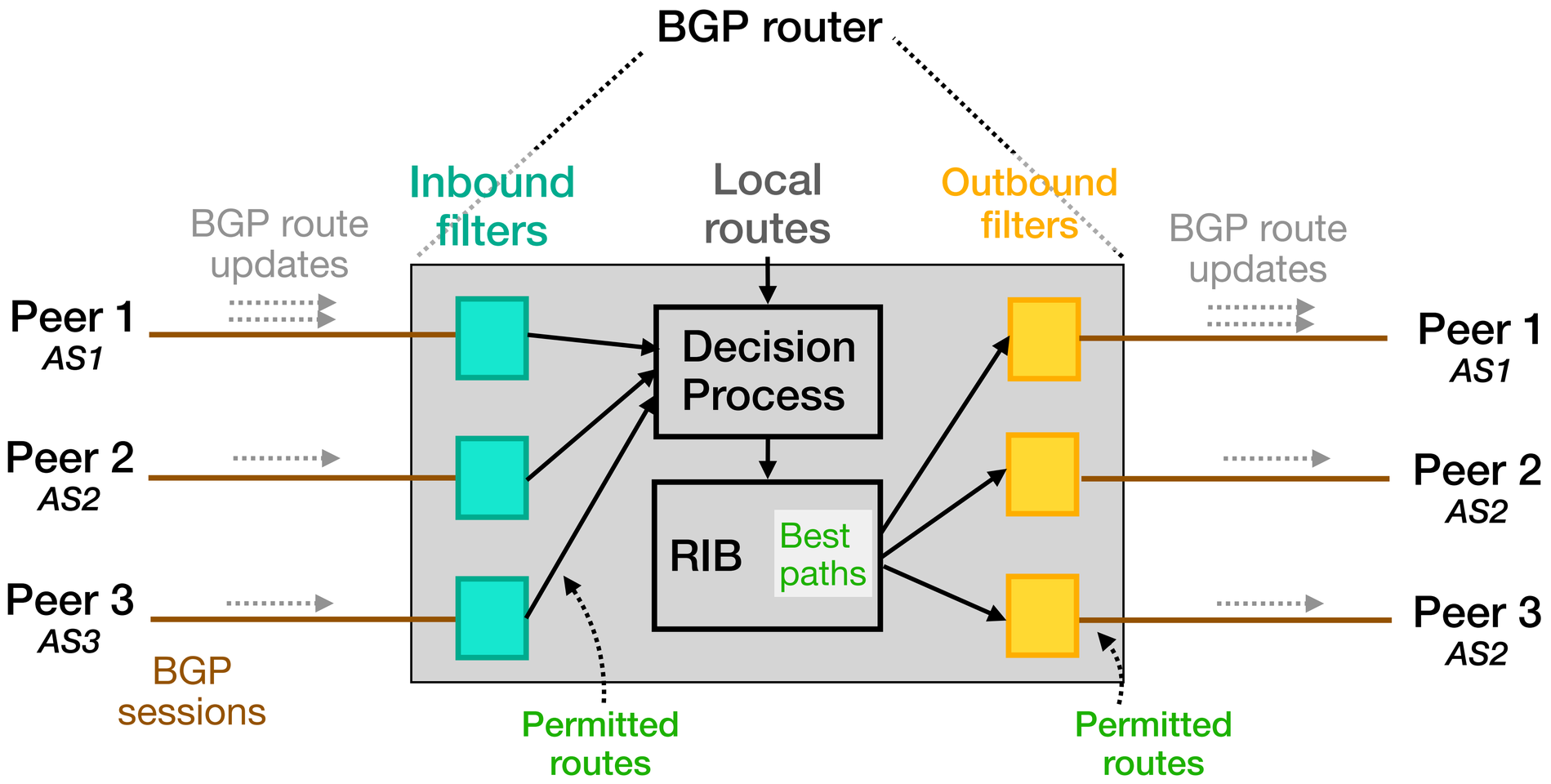
Problem: When an operator shares data via a BGP session, only a small fraction of its routes are visible due to BGP route hiding



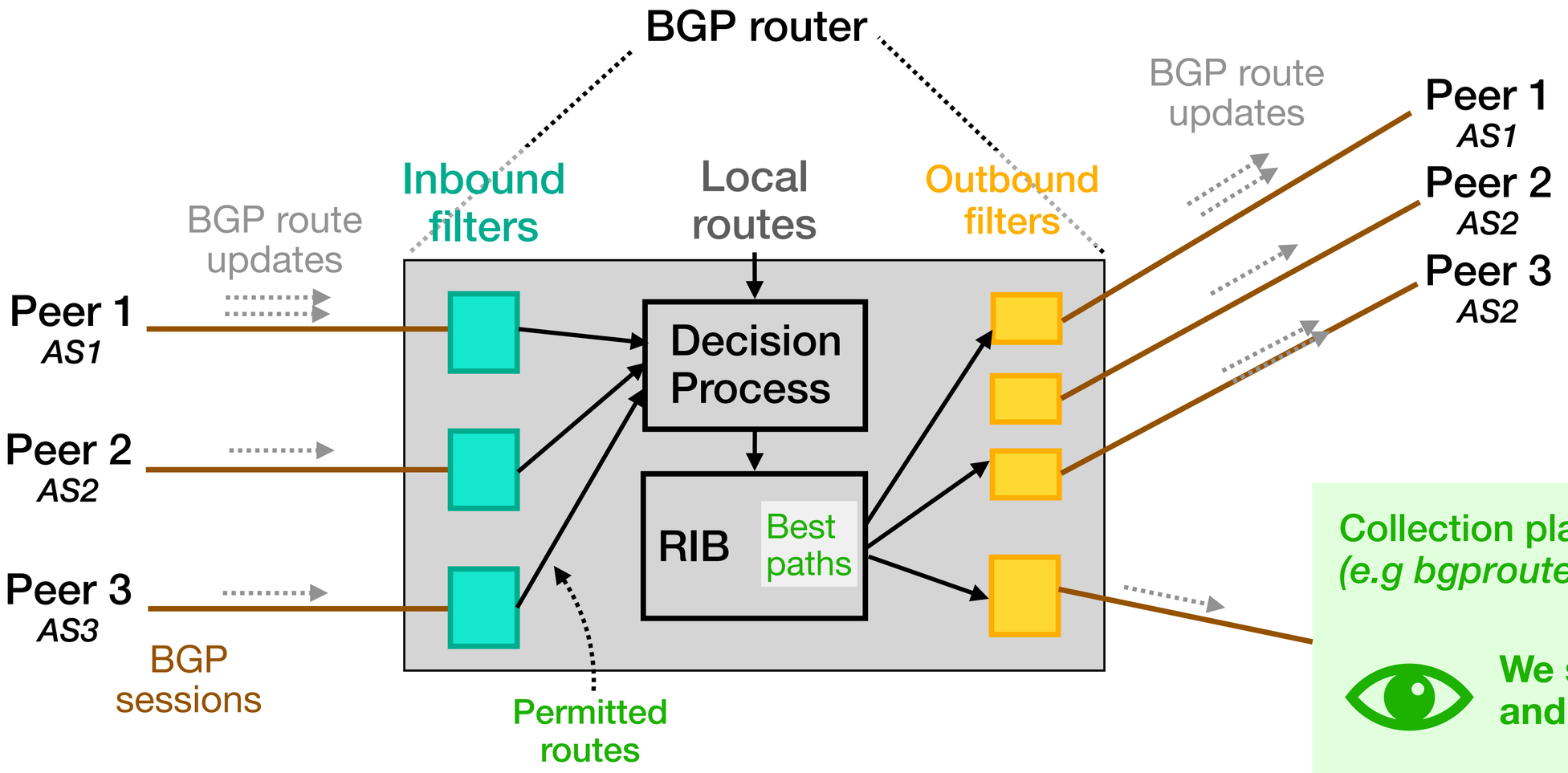
Problem: When an operator shares data via a BGP session, only a small fraction of its routes are visible due to BGP route hiding



Problem: When an operator shares data via a BGP session, only a small fraction of its routes are visible due to BGP route hiding



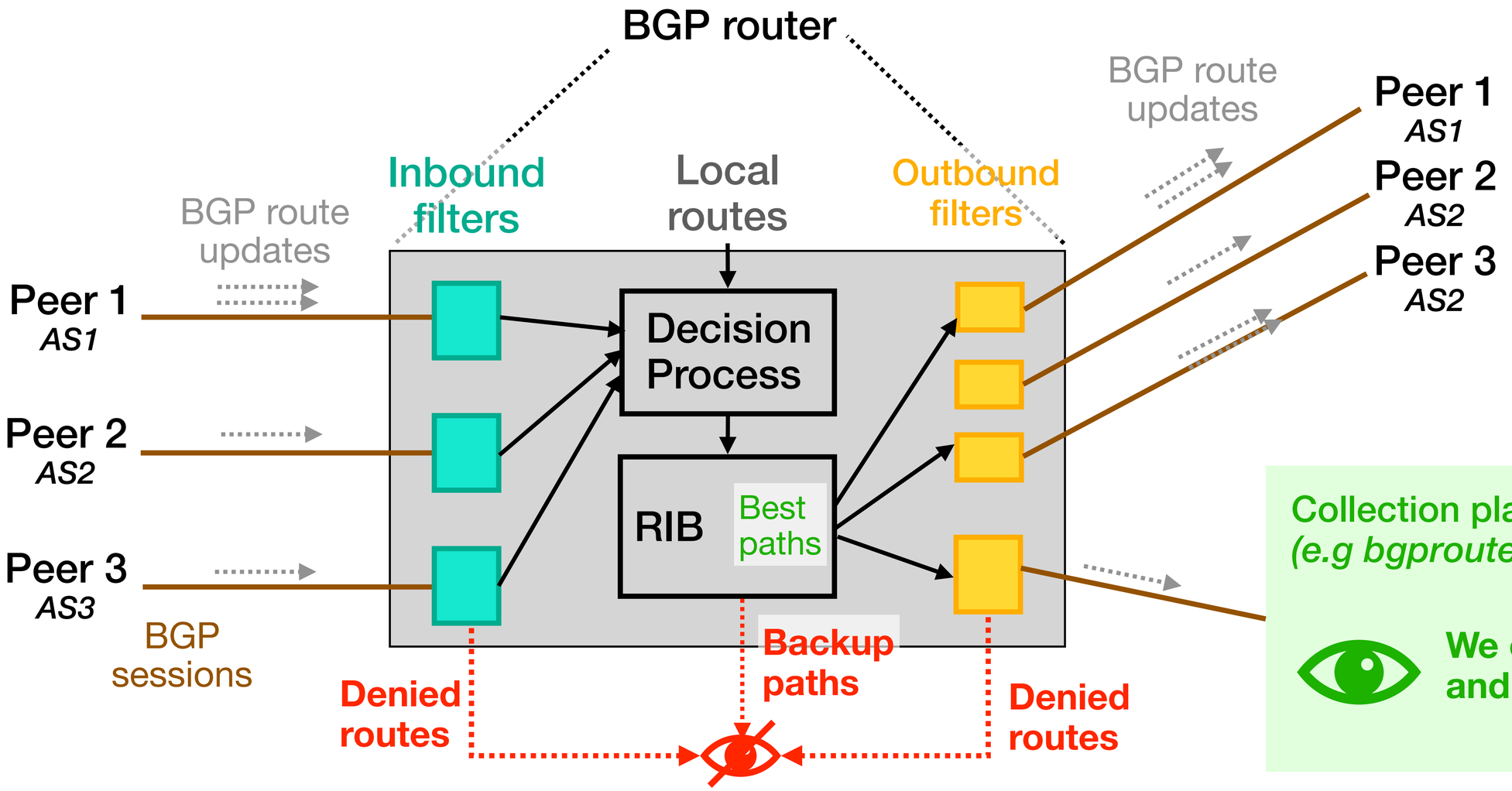
Problem: When an operator shares data via a BGP session, only a small fraction of its routes are visible due to BGP route hiding



Collection platform
(e.g bgproutes.io, RIS, …)

We see the best and non-filtered routes

Problem: When an operator shares data via a BGP session, only a small fraction of its routes are visible due to BGP route hiding



Collection platform
(e.g bgproutes.io, RIS, …)

We only see the best and non-filtered routes

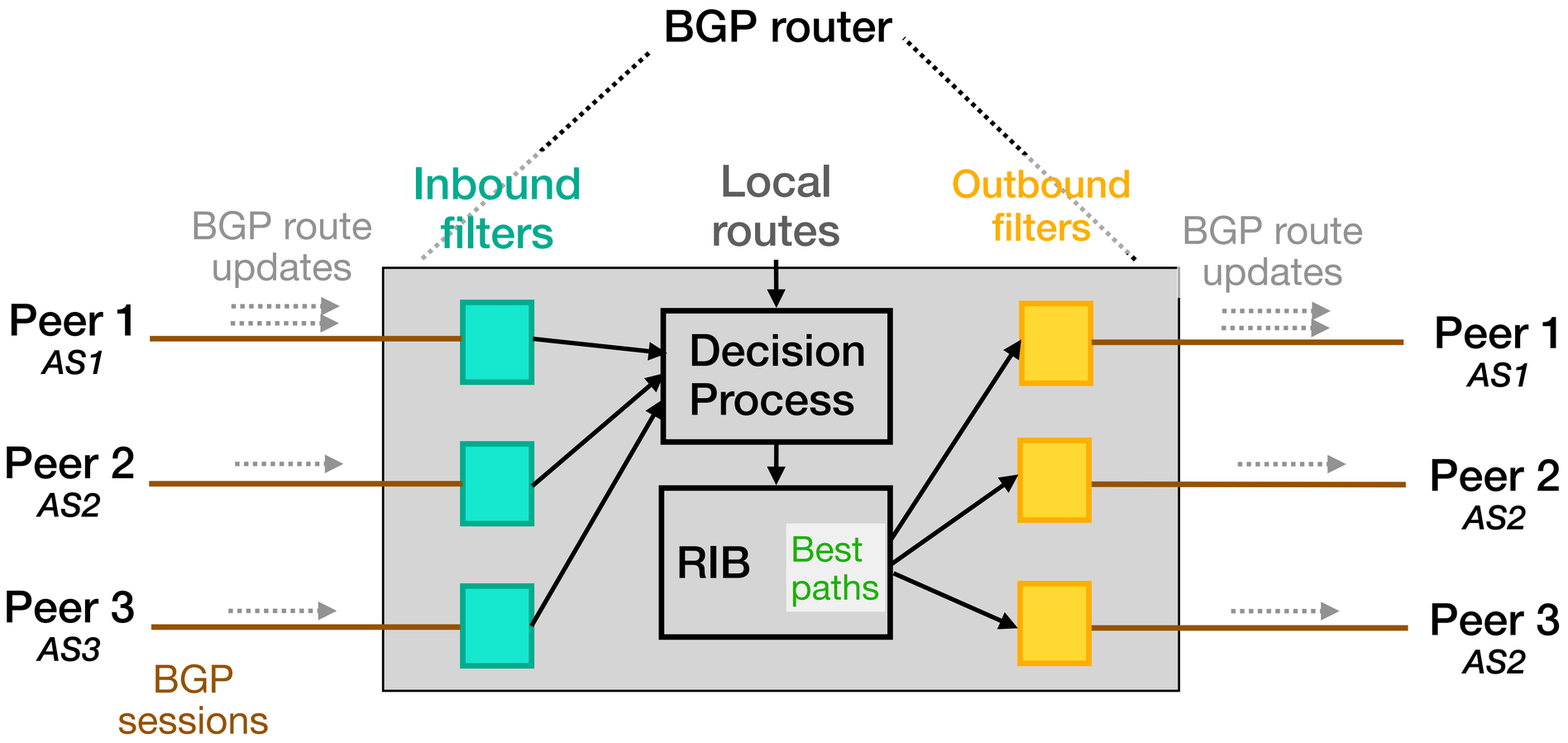
bgproutes.io can gather BGP route updates via the BGP Monitoring Protocol (BMP) — *and it's much more efficient!*

- ✅ BMP provides all route updates on a per-peer basis

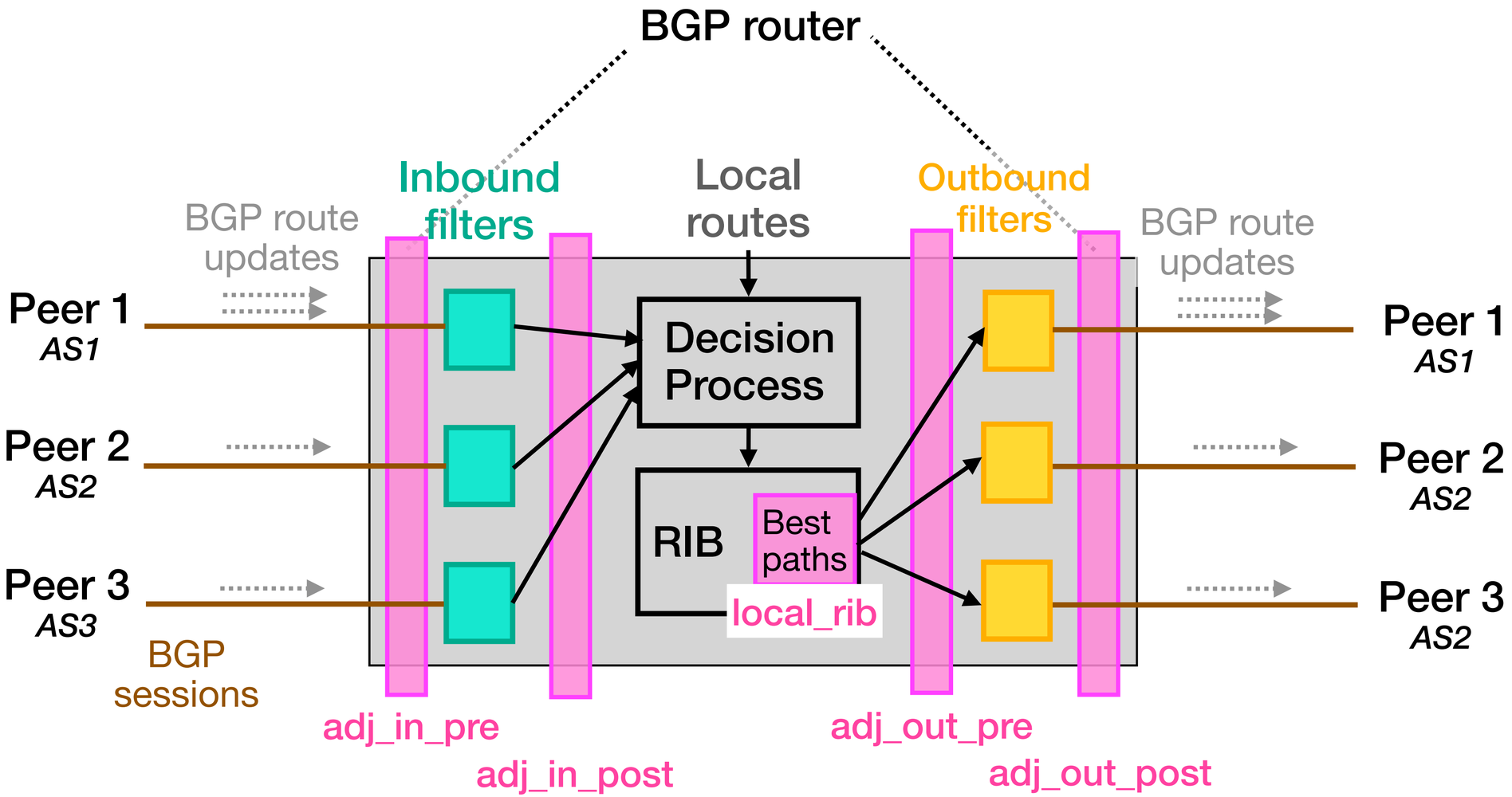
- ✅ BMP provides route updates at all stages of the BGP processing pipeline

➡ BMP is also simple to configure

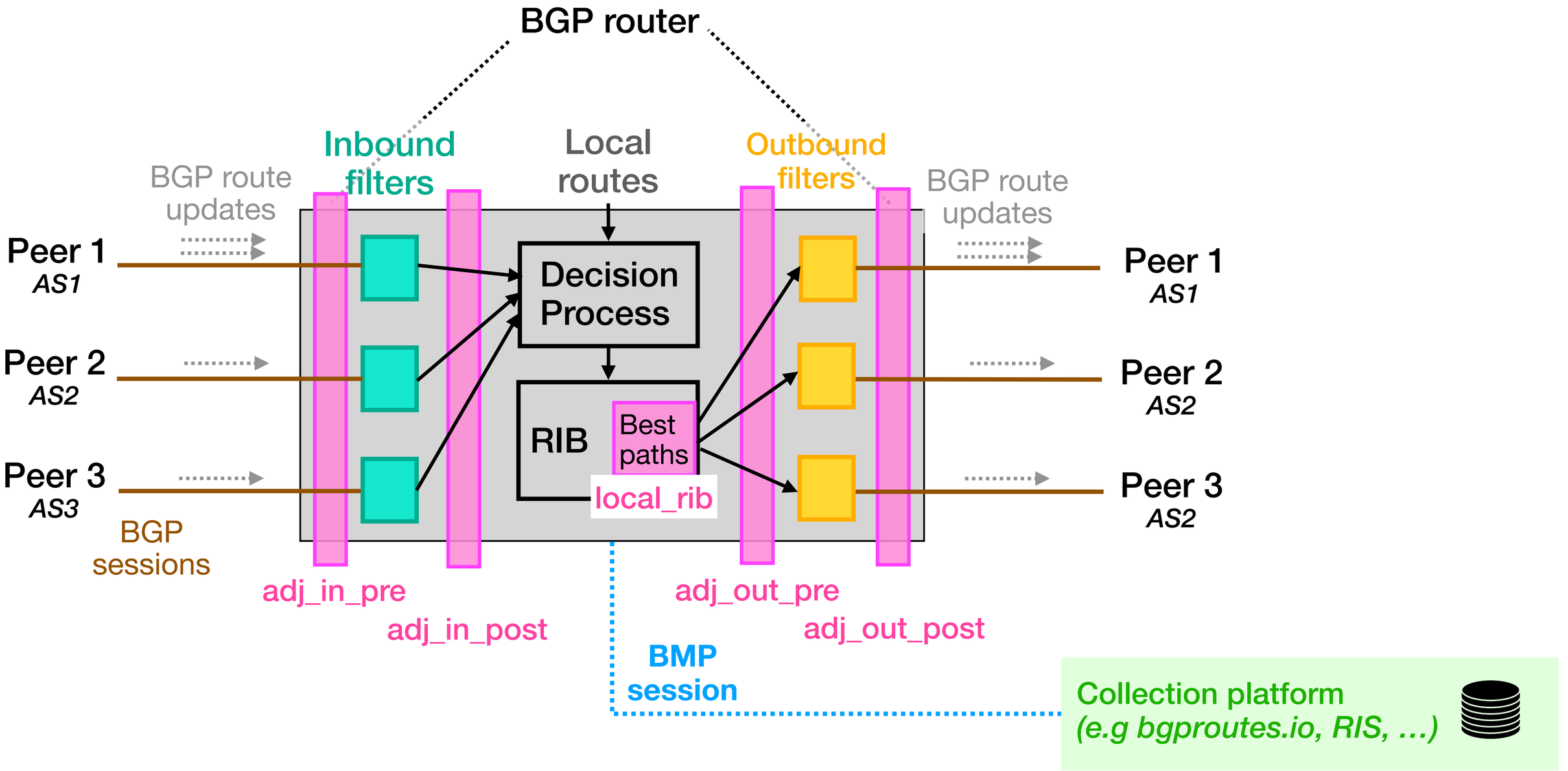
BMP allows to collect routes at every stage of the BGP router



BMP allows to collect routes at every stage of the BGP router

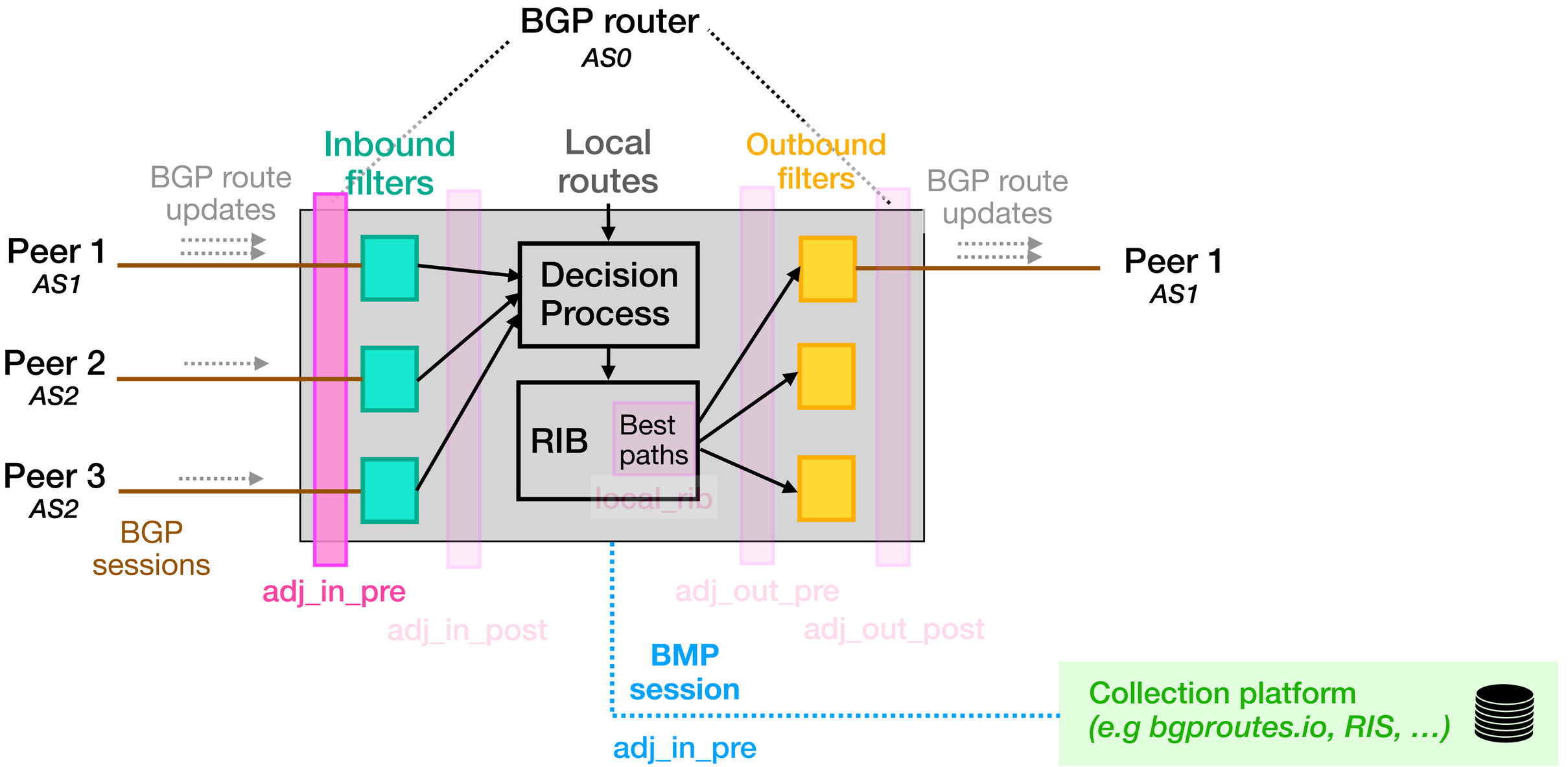


BMP allows to collect routes at every stage of the BGP router

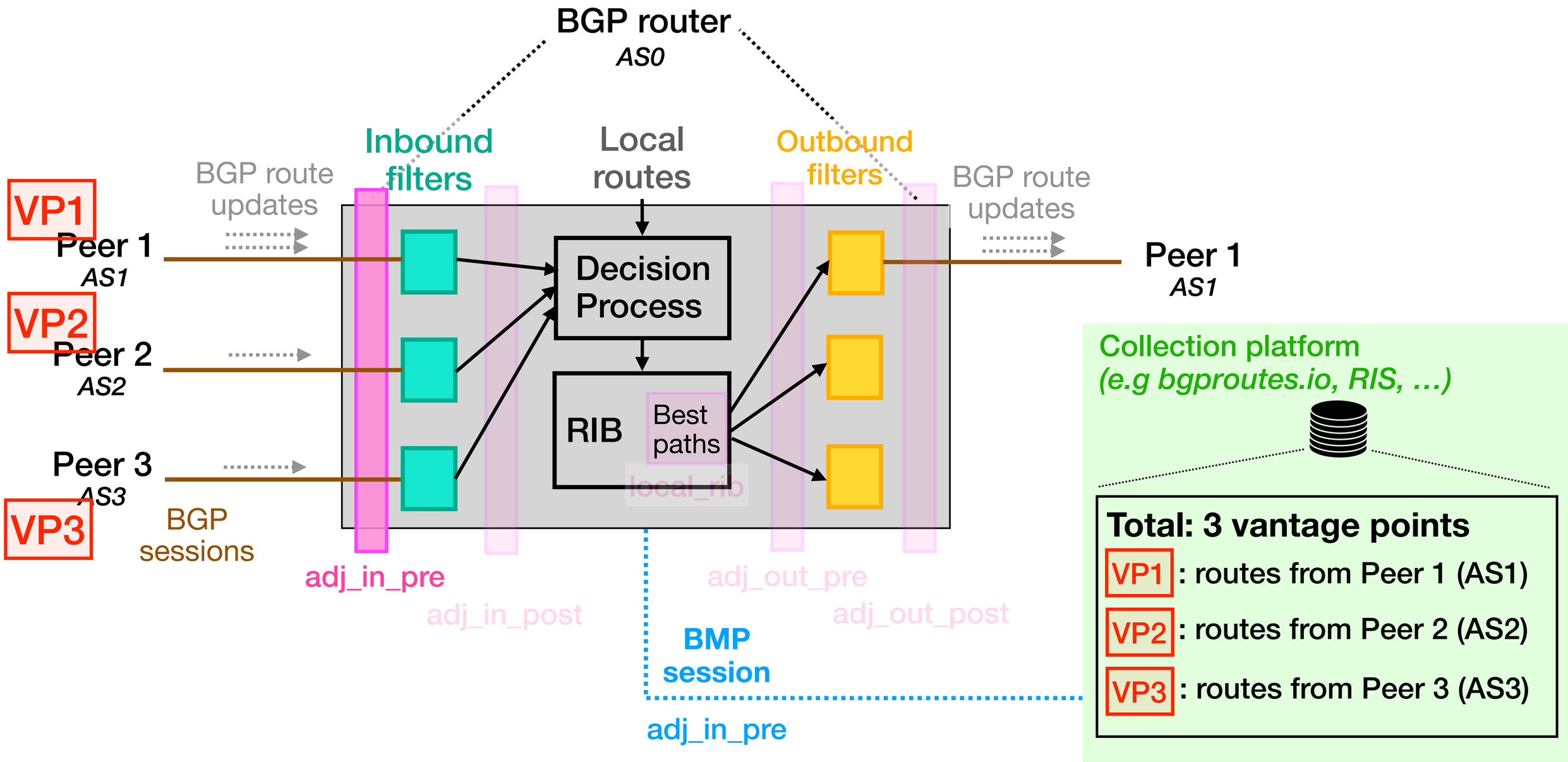


We recommend exporting data for the **adj_in_pre** stage only

We recommend exporting data for the **adj_in_pre** stage only

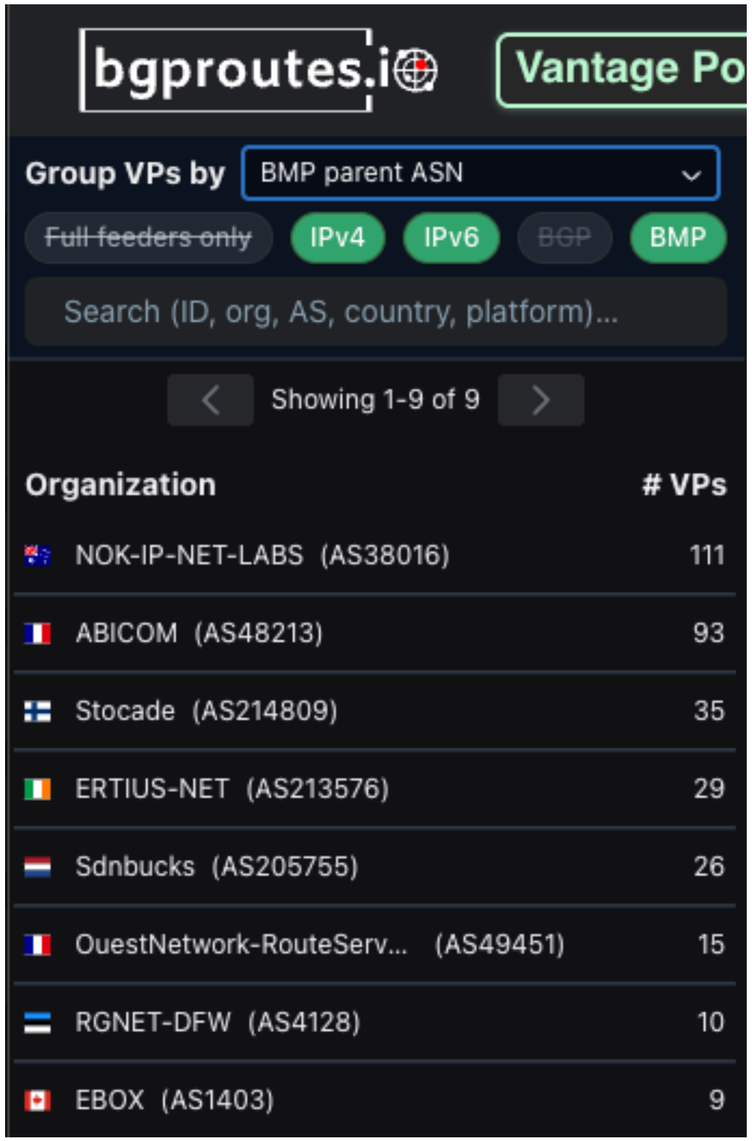


We recommend exporting data for the **adj_in_pre** stage only

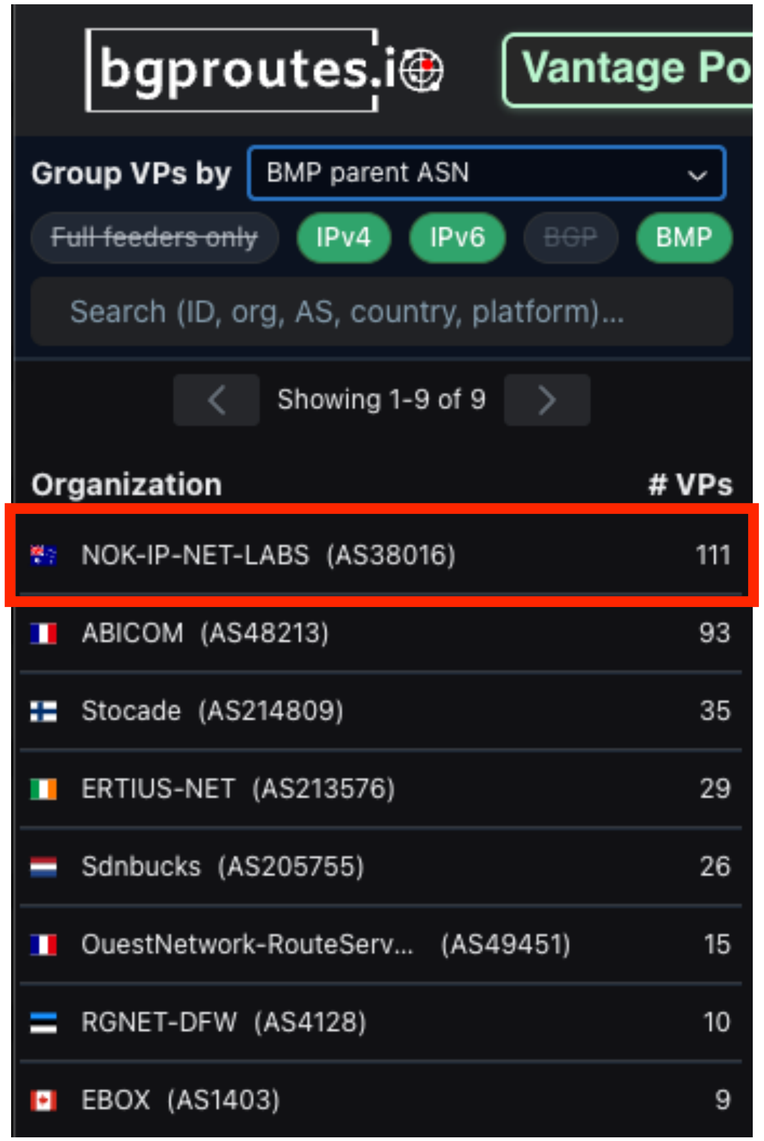


bgproutes.io has ~15 BMP sessions with 9 networks,
which translate into **~200 VPs!**

bgproutes.io has ~15 BMP sessions with 9 networks, which translate into **~200 VPs!**



bgproutes.io has ~15 BMP sessions with 9 networks, which translate into **~200 VPs!**



One BMP session exports data received from **111** BGP sessions!

bgproutes.io

Our new next-gen BGP route collection platform

- Collection: High coverage
- Distribution: API + Dashboards

https://bgproutes.io

You can contribute data

Collecting more data is useful

But only if users can effectively process it

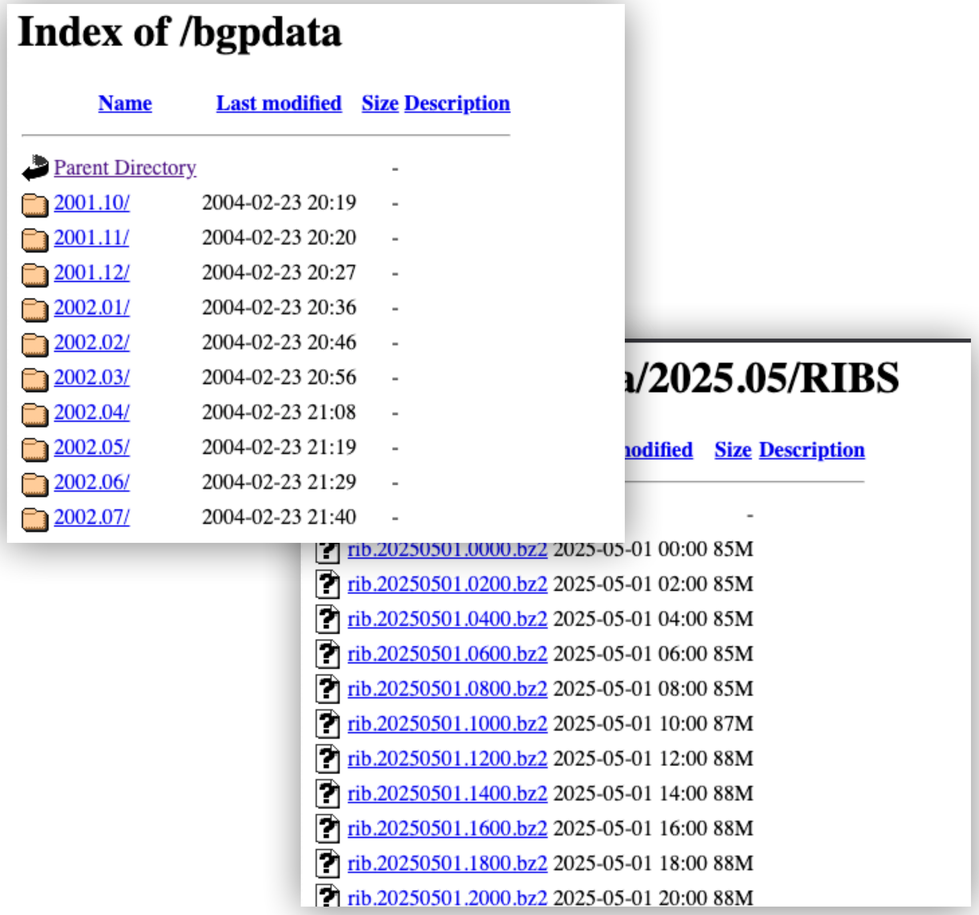
Collecting more data is useful

But only if users can effectively process it

Current MRT archives consist of 1000+ compressed files for updates and RIB dumps

The BGP data is intermingled across all these files

➡ MRT is hard to process



Typical MRT archive

bgproutes.io comes with a simple and fast API that provides high-granularity access to BGP data

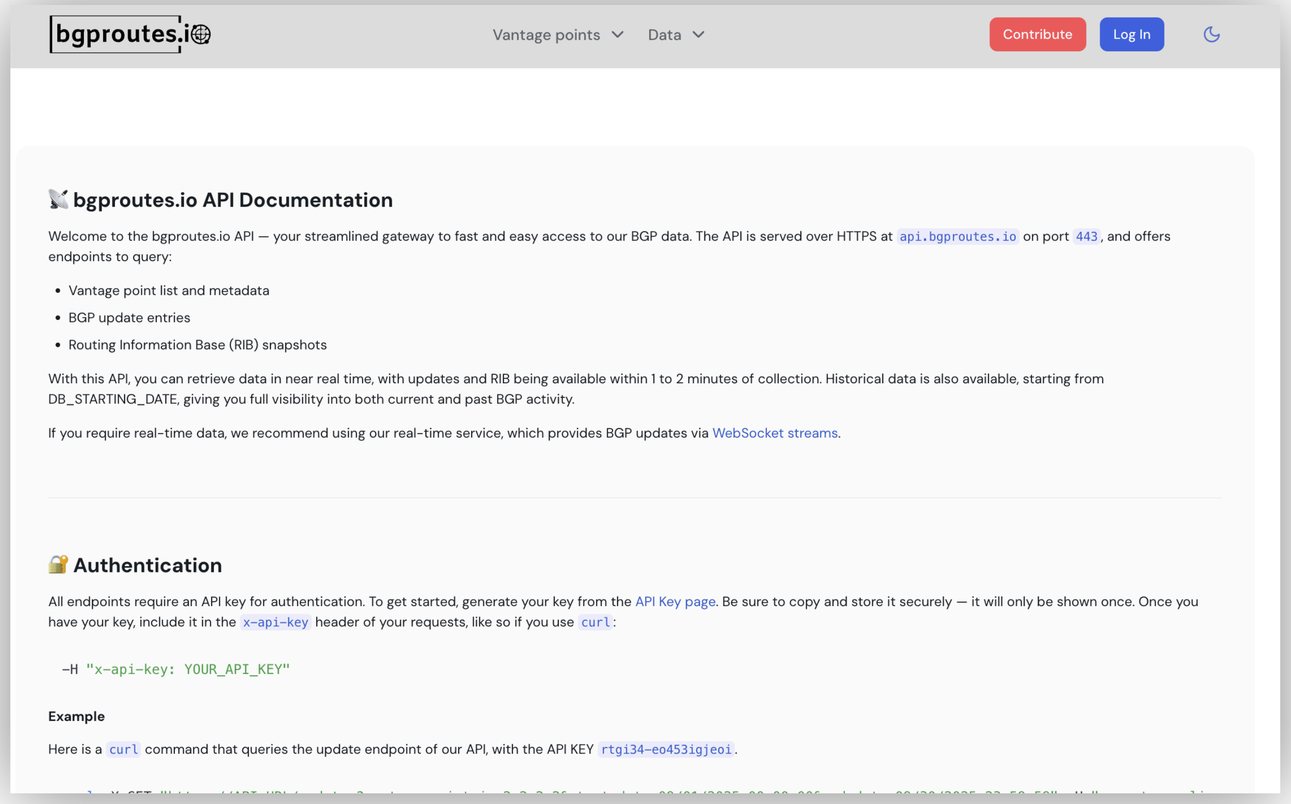
The API offers four endpoints:

vantage_points

updates

rib

topology



bgproutes.io/data_api

Imagine a network operator trying to retrieve all BGP updates for which their ASN appear in the AS path on February 10, 2026

```
# Let’s use our Python client to retrieve the data
from pybgproutesapi import vantage_points, updates
```

Imagine a network operator trying to retrieve all BGP updates for which their ASN appear in the AS path on February 10, 2026

```
# Let’s use our Python client to retrieve the data
from pybgproutesapi import vantage_points, updates

# Let’s retrieve the vantage points to use
vps = vantage_points(source=[‘bgproutes.io', ‘ris’])
```

Imagine a network operator trying to retrieve all BGP updates for which their ASN appear in the AS path on February 10, 2026

```
# Let’s use our Python client to retrieve the data
from pybgproutesapi import vantage_points, updates

# Let’s retrieve the vantage points to use
vps = vantage_points(source=[‘bgproutes.io', ‘ris’])

# We retrieve updates and print them for every VP
for vp in vps:
    vp_upd = updates(
        vp_ip=vp,
        start_date=“2026-02-10T00:00:00”,
        end_date=“2026-02-11T00:00:00”,
        aspath_regexp=“(^|  )3(  |$)“)

    print (vp_upd)
```

Imagine a network operator trying to retrieve all BGP updates for which their ASN appear in the AS path on February 10, 2026

```
# Let’s use our Python client to retrieve the data
from pybgproutesapi import vantage_points, updat

# Let’s retrieve the vantage points to use
vps = vantage_points(source=[‘bgproutes.io', ‘ris’])

# We retrieve updates and print them for every VP
for vp in vps:
    vp_upd = updates(
        vp_ip=vp,
        start_date=“2026-02-10T00:00:00”,
        end_date=“2026-02-11T00:00:00”,
        aspath_regexp=“(^|  )3(  |$)“)

    print (vp_upd)
```

Example of an output

Time	Prefix	ASPath
T1	1/24	1-3-7
T2	2/24	3-9-4
T3	3/24	2-4-3
T4	4/24	3-4-5

Other attributes omitted for space

Each returned update includes the RPKI ROV and ASPA status

Each returned update includes the RPKI ROV and ASPA status

Route Origin Validation: Verify that the origin of an AS path is legitimate
Possible outcome: Valid, Invalid, Unknown

Each returned update includes the RPKI ROV and ASPA status

Route Origin Validation: Verify that the origin of an AS path is legitimate
Possible outcome: Valid, Invalid, Unknown

ASPA-based Validation: Verify that there is no valley-free violation in the AS path
Possible outcome:

Upstream Valid or Upstream Invalid or Upstream Unknown	And	Downstream Valid or Downstream Invalid or Downstream Unknown

Each returned update includes the RPKI ROV and ASPA status

```
from pybgproutesapi import vantage_points, updates
vps = vantage_points(source=['bgproutes.io', 'ris'])

# We retrieve updates and print them for every VP

for vp in vps:
    vp_upd = updates(
        vp_ip=vp,
        start_date="2026-02-10T00:00:00",
        end_date="2026-02-11T00:00:00",
        aspath_regexp="(^|  )3(  |$)")


    print (vp_upd)
```

Example of an output

Time	Prefix	ASPath	ROV status	ASPA status
T1	1/24	1-3-7	Valid	Up-valid Down-invalid
T2	2/24	3-9-4	Unknown	Up-valid Down-valid
T3	3/24	2-4-3	Invalid	Up-invalid Down-invalid
T4	4/24	3-4-5	Valid	Up-valid Down-Unknown

Other attributes omitted for space

Users can filter BGP route updates based on their ROV or ASPA status

Users can filter BGP route updates based on their ROV or ASPA status

```
from pybgproutesapi import vantage_points, updates
vps = vantage_points(source=['bgproutes.io', 'ris'])

# We retrieve updates and print them for every VP
for vp in vps:
    vp_upd = updates(
        vp_ip=vp,
        start_date="2026-02-10T00:00:00",
        end_date="2026-02-11T00:00:00",
        aspath_regexp="^3 | 3 | 3$"
        rov_status=["valid", "unknown"],
        aspa_status=['down-invalid','down-unknown'])

    print (vp_upd)
```

Example of an output

Users can filter BGP route updates based on their ROV or ASPA status

```
from pybgproutesapi import vantage_points, updates
vps = vantage_points(source=['bgproutes.io', 'ris'])

# We retrieve updates and print them for every VP
for vp in vps:
    vp_upd = updates(
        vp_ip=vp,
        start_date="2026-02-10T00:00:00",
        end_date="2026-02-11T00:00:00",
        aspath_regexp="^3 | 3 | 3$"
        rov_status=["valid", "unknown"],
        aspa_status=['down-invalid','down-unknown'])

    print (vp_upd)
```

Example of an output

Time	Prefix	ASPath	ROV status	ASPA status
T1	**1/24**	**1-3-7**	**Valid**	**Up-valid Down-invalid**
T2	2/24	3-9-4	Unknown	Up-valid Down-valid
T3	3/24	2-4-3	Invalid	Up-invalid Down-invalid
T4	**4/24**	**3-4-5**	**Valid**	**Up-valid Down-Unknown**

Other attributes omitted for space

Users can filter BGP route updates based on their ROV or ASPA status

… ***and easily compute statistics!***

Users can filter BGP route updates based on their ROV or ASPA status

… ***and easily compute statistics!***

Example: Let’s analyse ASPA for downstream AS paths
using all collected updates within a 2-hour timeframe

Users can filter BGP route updates based on their ROV or ASPA status

… ***and easily compute statistics!***

Example: Let's analyse ASPA for downstream AS paths
using all collected updates within a 2-hour timeframe

99.3% are Unknown

There are only ~500 ASPA objects so far

Users can filter BGP route updates based on their ROV or ASPA status

… ***and easily compute statistics!***

Example: Let's analyse ASPA for downstream AS paths
using all collected updates within a 2-hour timeframe

99.3% are Unknown

There are only ~500 ASPA objects so far

0.04% are Invalid

Potential route leaks!

Users can filter BGP route updates based on their ROV or ASPA status

… ***and easily compute statistics!***

Example: Let's analyse ASPA for downstream AS paths
using all collected updates within a 2-hour timeframe

99.3% are Unknown

There are only ~500 ASPA objects so far

0.04% are Invalid

Potential route leaks!

0.7% are Valid

Most of them are short AS paths (avg length: 2.6)

bgproutes.io

Our new next-gen BGP route collection platform

- Collection: High coverage
- Distribution: API + Dashboards

https://bgproutes.io

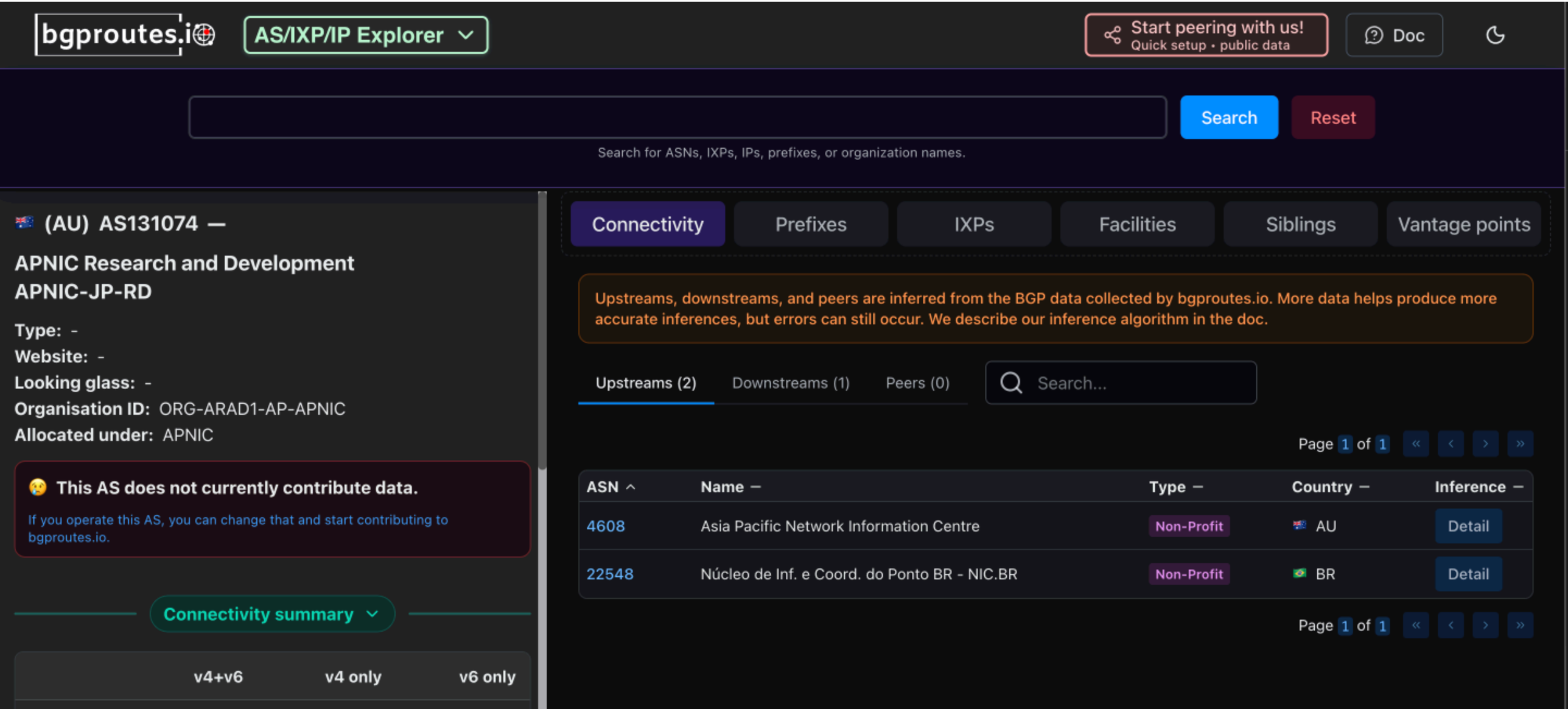
You can peer with us!

bgproutes.io provides four dashboards, and more will come soon

AS Explorer	Prefix Explorer	Vantage Points Explorer	AS Rank
General info about ASes, prefixes, IXPs	“Giant” looking glass	List all VPs and provide statistics	Rank ASes based on various metrics

Our goal is that these dashboards bridge the gap between users and the raw data

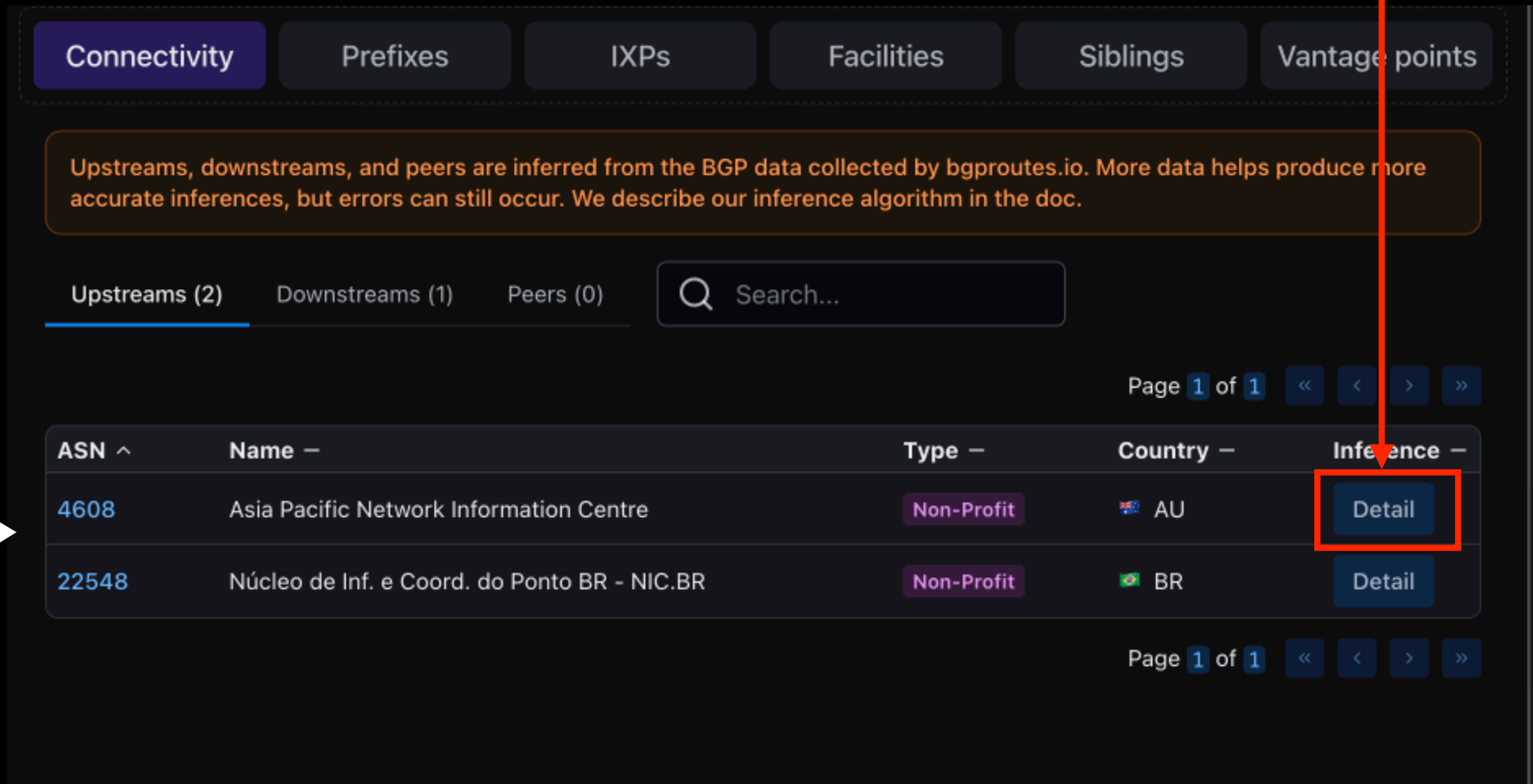
Our goal is that these dashboards bridge the gap between users and the raw data



Our goal is that these dashboards bridge the gap between users and the raw data

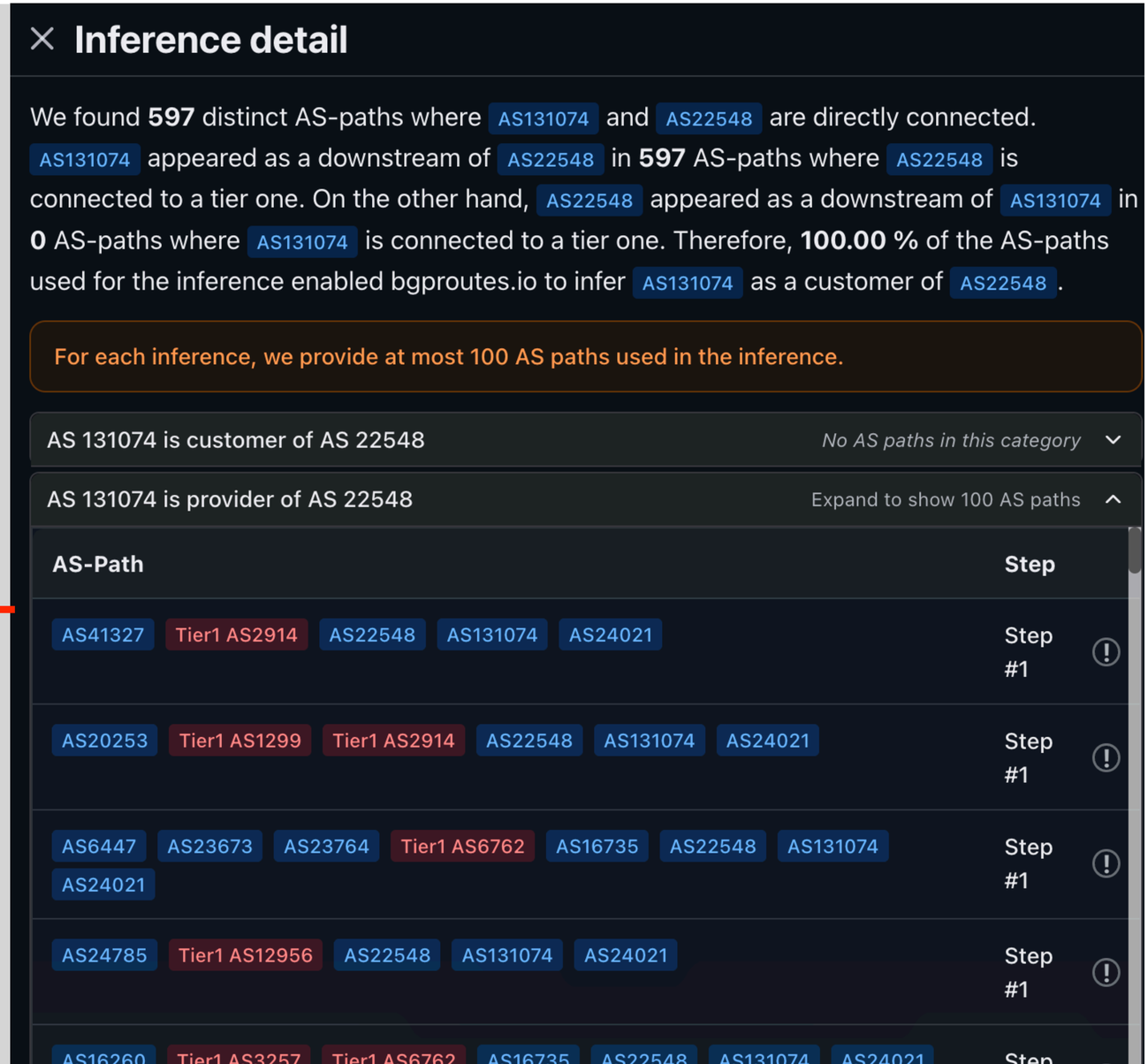
Click to see the raw data used to infer the AS relatonship

AS relationships



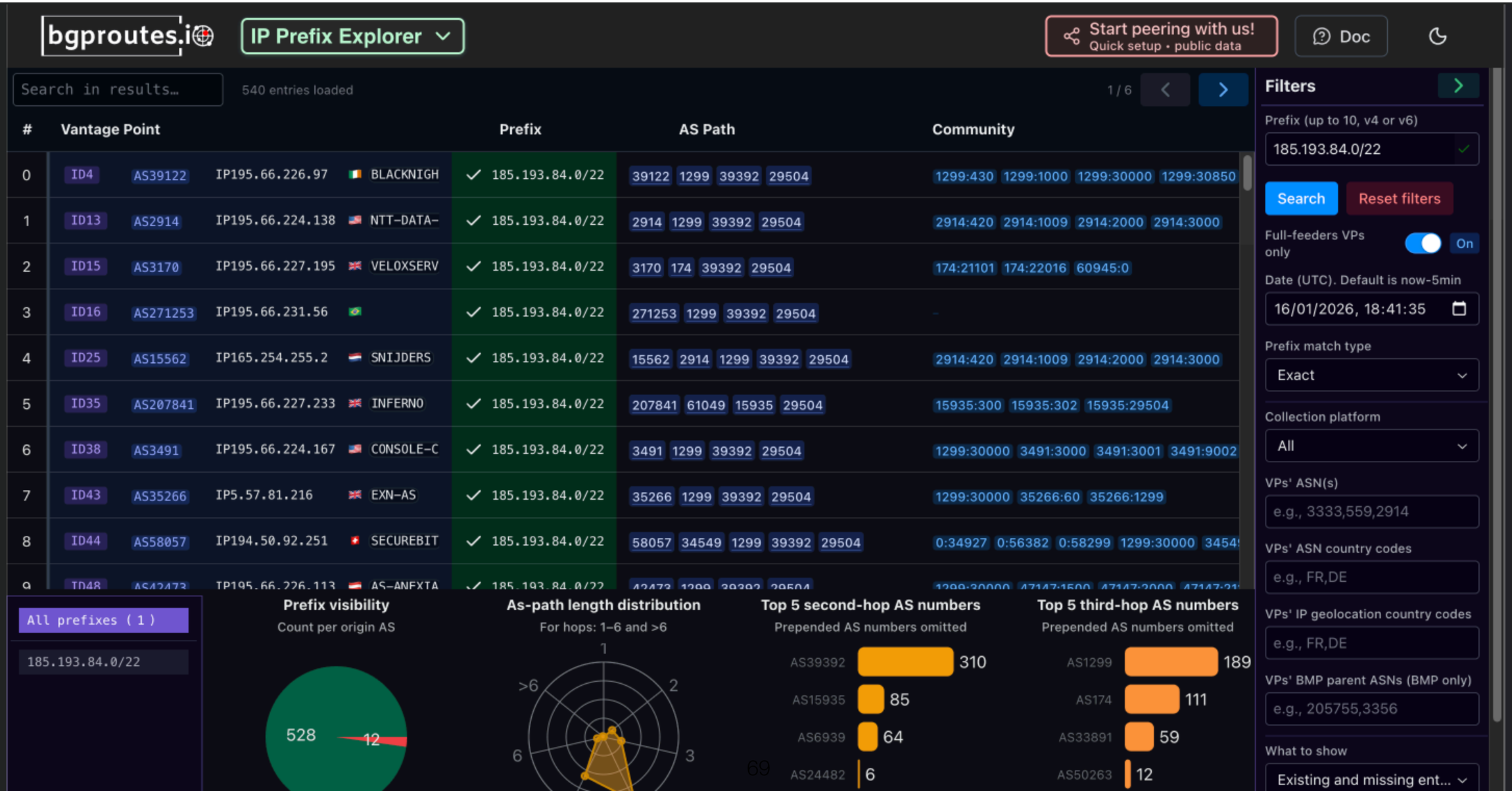
Our goal is that these dashboards bridge the gap between users and the raw data

AS paths used to infer relationships



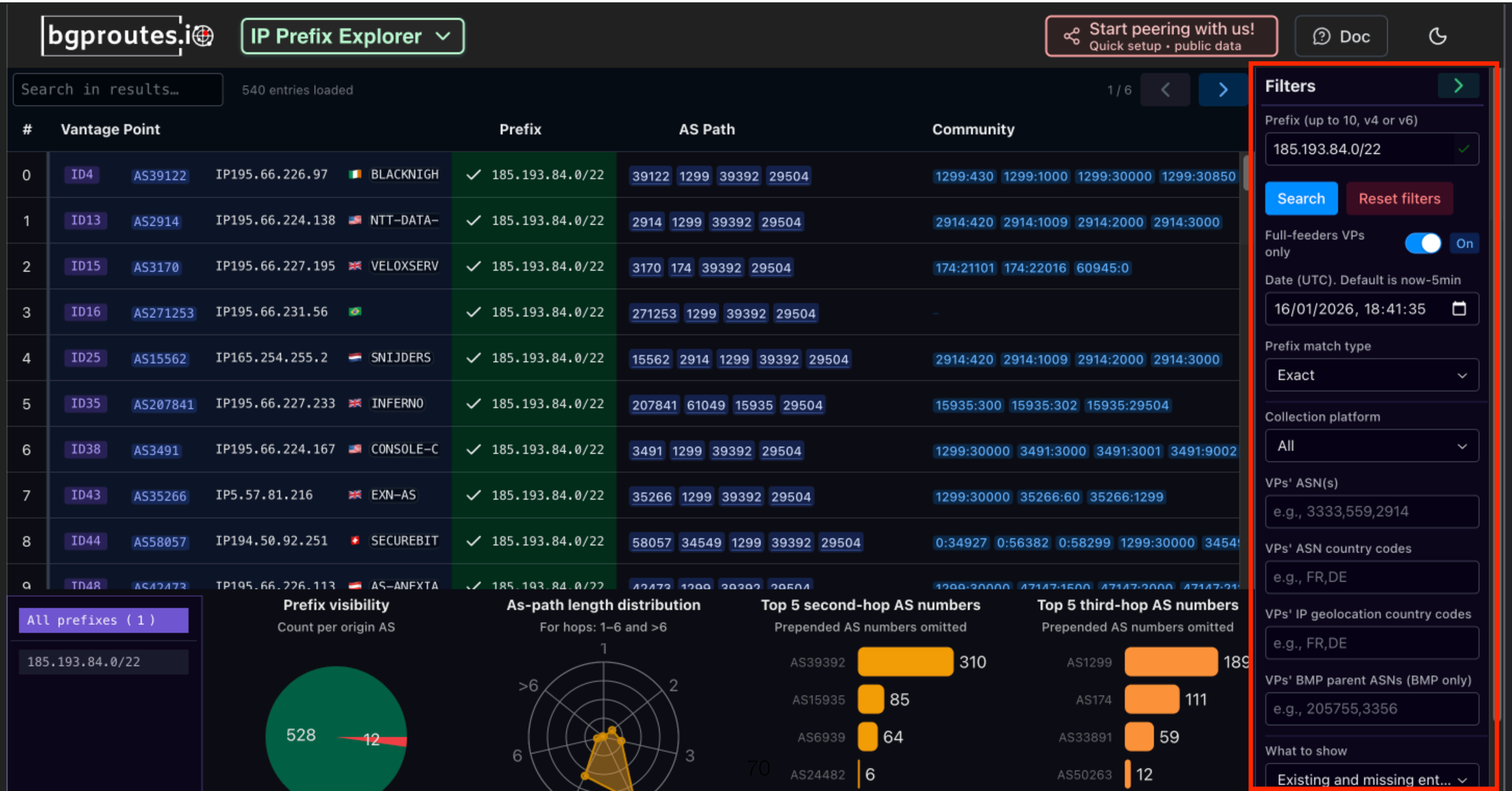
Our goal is that these dashboards bridge the gap between users and the raw data

Our goal is that these dashboards bridge the gap between users and the raw data



Our goal is that these dashboards bridge the gap between users and the raw data

Advanced filtering capabilities



bgproutes.io

https://bgproutes.io

You can contribute data

Current sponsors:

Follow us on Linkedin, Discord,
or reach out at contact@bgproutes.io