RIPE NCC
RIPE NETWORK COORDINATION CENTER

RIPE RIS Update

February 2025

Ties de Kock | RIPE NCC | 11 Feb 2025

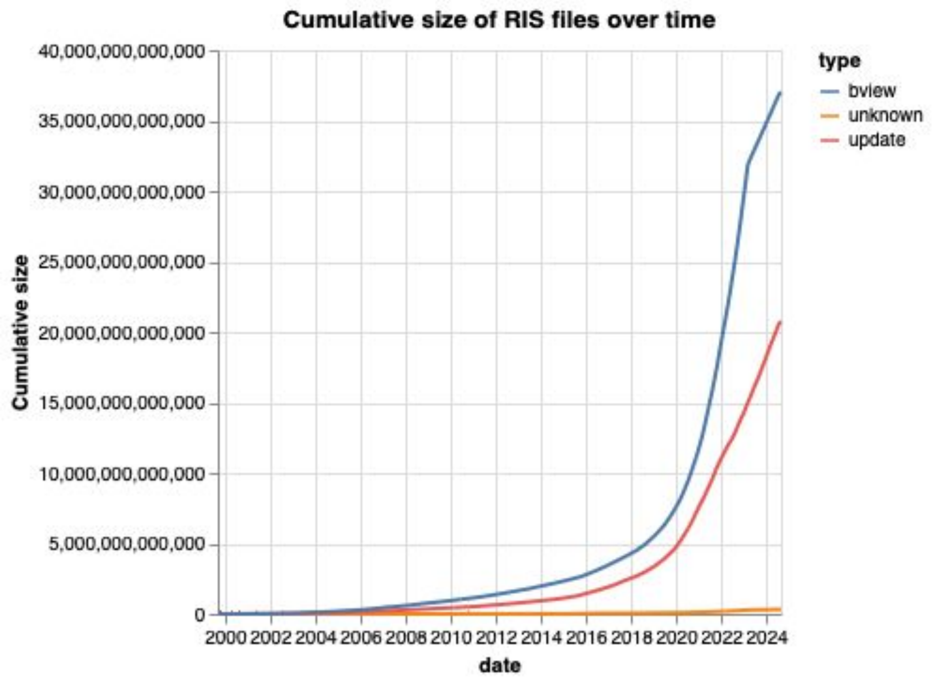
RIPE RIS

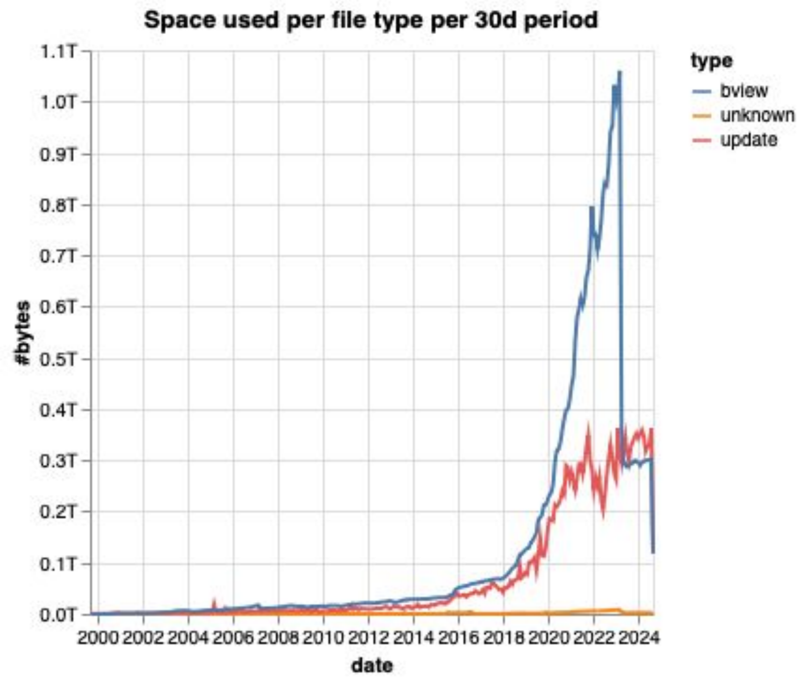
What are we doing

- BGP data collection and peering coordination
- Longitudinal data collection
- RISwhois
  - ~650 requests/s
- RIS beacons
- RIPEstat derived data
- RISlive

RIPE RIS

Data volume





RIPE RIS: 2024

What happened in 2024

- Mostly internal focused due to a move between legacy and more modern hadoop infrastructure.
- Slowly adding peers
  - Handling a peering request is a dialogue with an operator
- We were adding
  - IXP route servers
  - Peers in underrepresented areas

RIPE RIS: 2025

Roadmap for 2025

- Another infrastructure migration
- Looking to replace our looking glass, exposed by RIPEstat (now: in-memory SQL)

- We have an experiment with getting peering sessions
  - IXP route servers: How feasible is it to add IXP route servers as remote peers?
  - Hyperscalers: We want “standard peer” sessions from hyperscalers

- Adding a *limited* number of collectors
  - Likely: collector by “topic”
  - Multi-hop scales better

What issues do we see

Working with BGP data is hard

- There is a tooling gap
  - Processing MRT is hard
  - File size only increases

- We are experimenting internally with parquet based data
  - “Good enough for most analysis”
  - “Only guarantee is that I will change the format”

What issues do we see

Open Questions

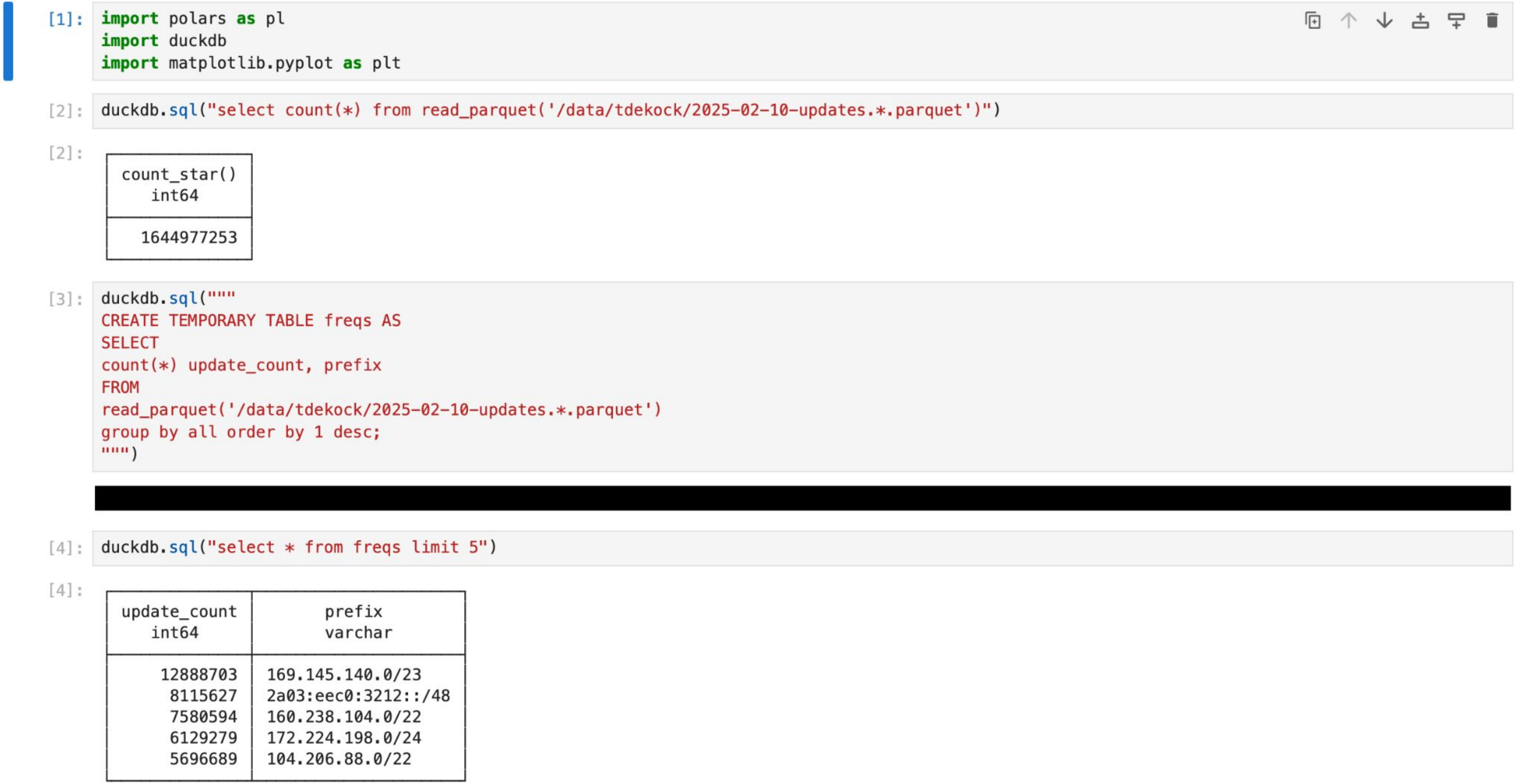
- Data modeling choices
    - String for AS, u32 for AS?

Details - THIS AND FOLLOWING SLIDES ADDED ON 19-2

Prototype state

- Process MRT files in rust using bgpkit
- Create various artefacts
- Analysis is easy: you can use your existing SQL skills

In the meeting presentation there was a live demo. Some screenshots of the code on the next slides.



```
[1]: import polars as pl
     import duckdb
     import matplotlib.pyplot as plt
```

```
[2]: duckdb.sql("select count(*) from read_parquet('/data/tdekock/2025-02-10-updates.*.parquet')")
```

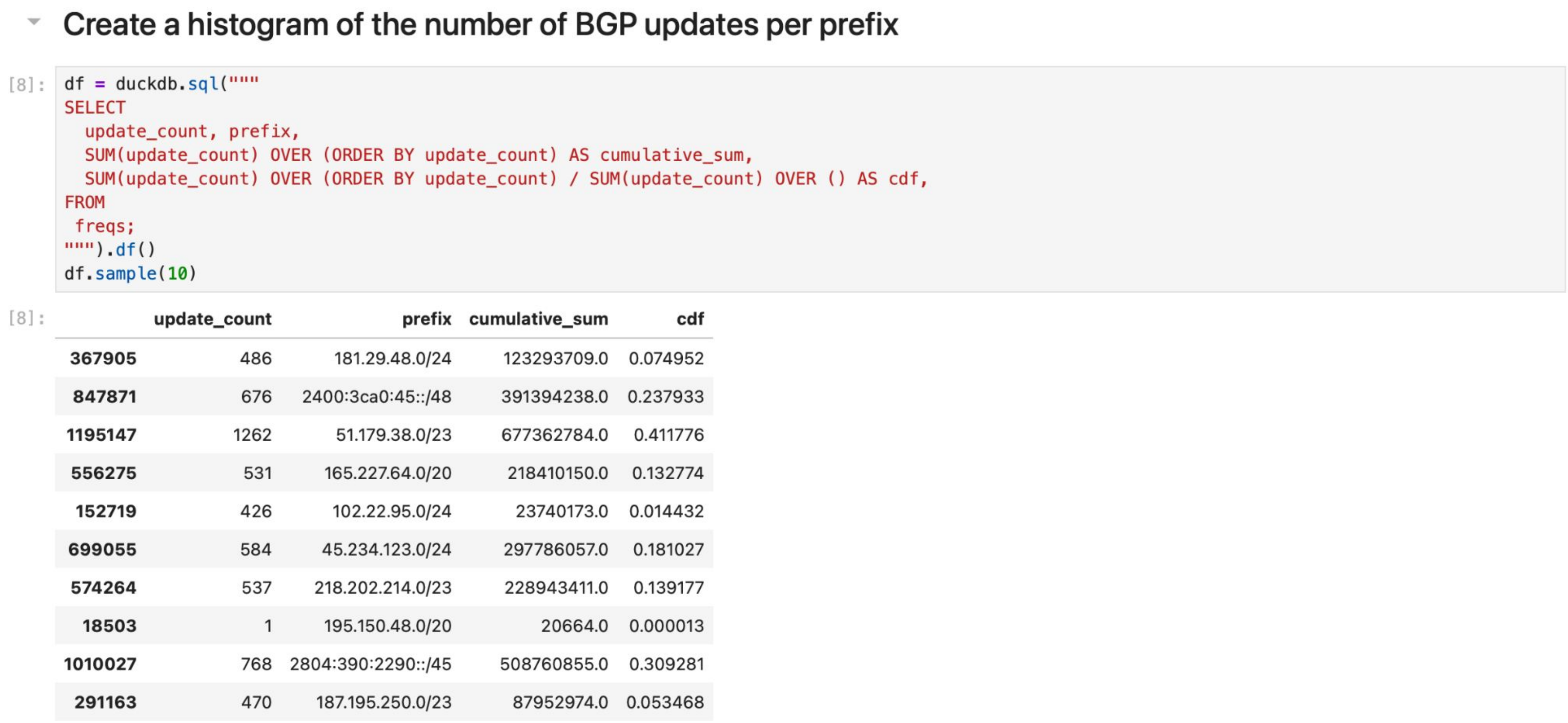
```
[2]:
┌──────────────┐
│ count_star() │
│    int64     │
├──────────────┤
│   1644977253 │
└──────────────┘
```

```
[3]: duckdb.sql("""
     CREATE TEMPORARY TABLE freqs AS
     SELECT
     count(*) update_count, prefix
     FROM
     read_parquet('/data/tdekock/2025-02-10-updates.*.parquet')
     group by all order by 1 desc;
     """)
```

[redacted]

```
[4]: duckdb.sql("select * from freqs limit 5")
```

```
[4]:
┌──────────────┬─────────────────────┐
│ update_count │       prefix        │
│    int64     │       varchar       │
├──────────────┼─────────────────────┤
│     12888703 │ 169.145.140.0/23    │
│      8115627 │ 2a03:eec0:3212::/48 │
│      7580594 │ 160.238.104.0/22    │
│      6129279 │ 172.224.198.0/24    │
│      5696689 │ 104.206.88.0/22     │
└──────────────┴─────────────────────┘
```



Create a histogram of the number of BGP updates per prefix

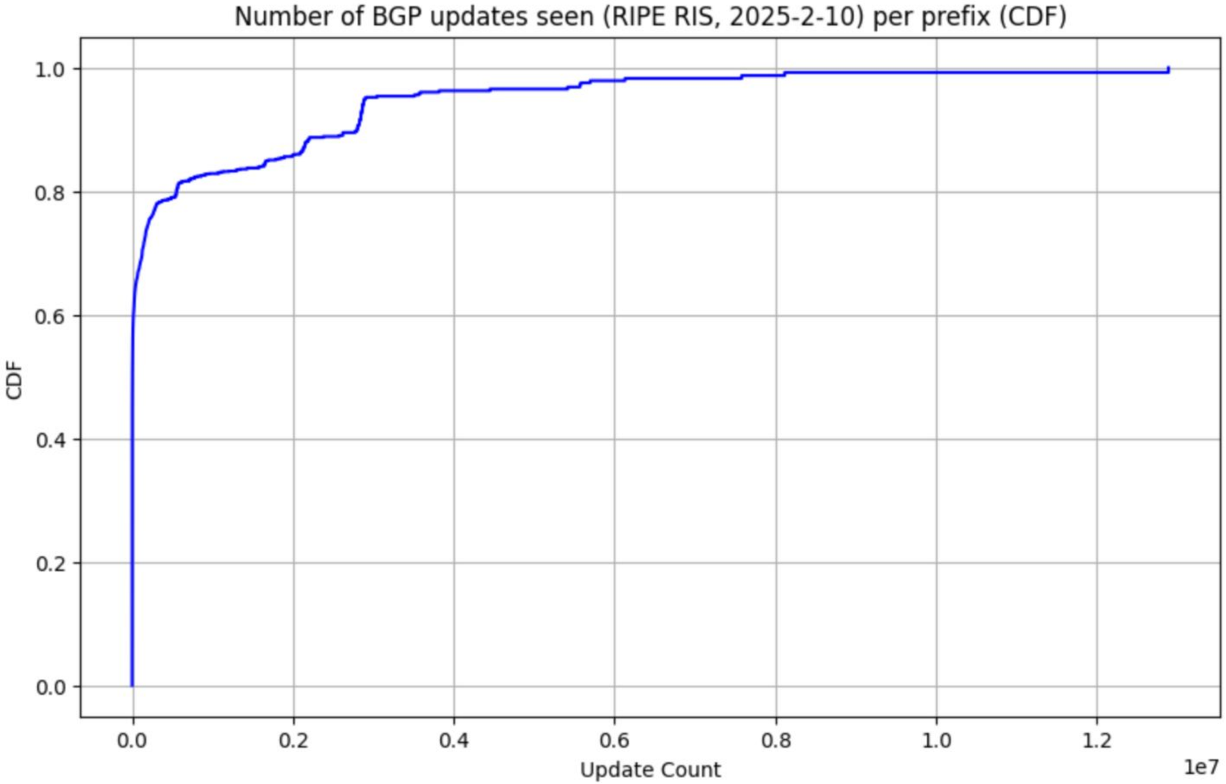
[8]:

```
df = duckdb.sql("""
SELECT
  update_count, prefix,
  SUM(update_count) OVER (ORDER BY update_count) AS cumulative_sum,
  SUM(update_count) OVER (ORDER BY update_count) / SUM(update_count) OVER () AS cdf,
FROM
 freqs;
""").df()
df.sample(10)
```

[8]:

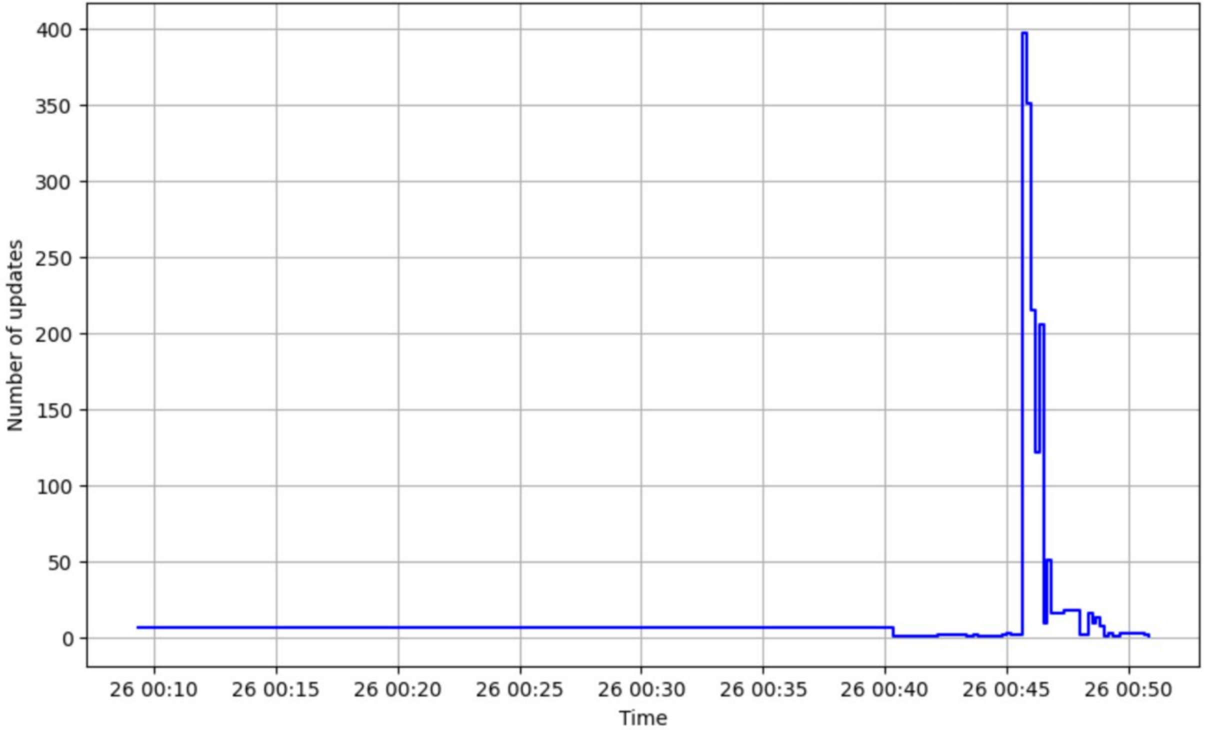
	update_count	prefix	cumulative_sum	cdf
367905	486	181.29.48.0/24	123293709.0	0.074952
847871	676	2400:3ca0:45::/48	391394238.0	0.237933
1195147	1262	51.179.38.0/23	677362784.0	0.411776
556275	531	165.227.64.0/20	218410150.0	0.132774
152719	426	102.22.95.0/24	23740173.0	0.014432
699055	584	45.234.123.0/24	297786057.0	0.181027
574264	537	218.202.214.0/23	228943411.0	0.139177
18503	1	195.150.48.0/20	20664.0	0.000013
1010027	768	2804:390:2290::/45	508760855.0	0.309281
291163	470	187.195.250.0/23	87952974.0	0.053468

```
[9]: plt.figure(figsize=(10, 6))
     plt.step(df["update_count"], df["cdf"], where='post', linestyle='-', color='b')
     plt.xlabel('Update Count')
     plt.ylabel('CDF')
     plt.title('Number of BGP updates seen (RIPE RIS, 2025-2-10) per prefix (CDF)')
     plt.grid(True)
     plt.show()
```



```
[9]: # Plot the CDF
     plt.figure(figsize=(10, 6))
     plt.step(df["time_window_start"], df["update_count"], where='post', linestyle='-', color='b')
     plt.xlabel('Time')
     plt.ylabel('Number of updates')
     plt.title('Number of BGP announcements seen in RIS with 1299 8194 in AS path (RIPE RIS, 2025-1-26, 0:00-01:30 UTC)')
     plt.grid(True)
     plt.show()
```

Number of BGP announcements seen in RIS with 1299 8194 in AS path (RIPE RIS, 2025-1-26, 0:00-01:30 UTC)


Number of updates
26 00:10
26 00:15
26 00:20
26 00:25
26 00:30
26 00:35
26 00:40
26 00:45
26 00:50
Time