DATE : 2025-02-11

CLOUDFLARE

Cloudflare Radar and BGP Data Pipelines

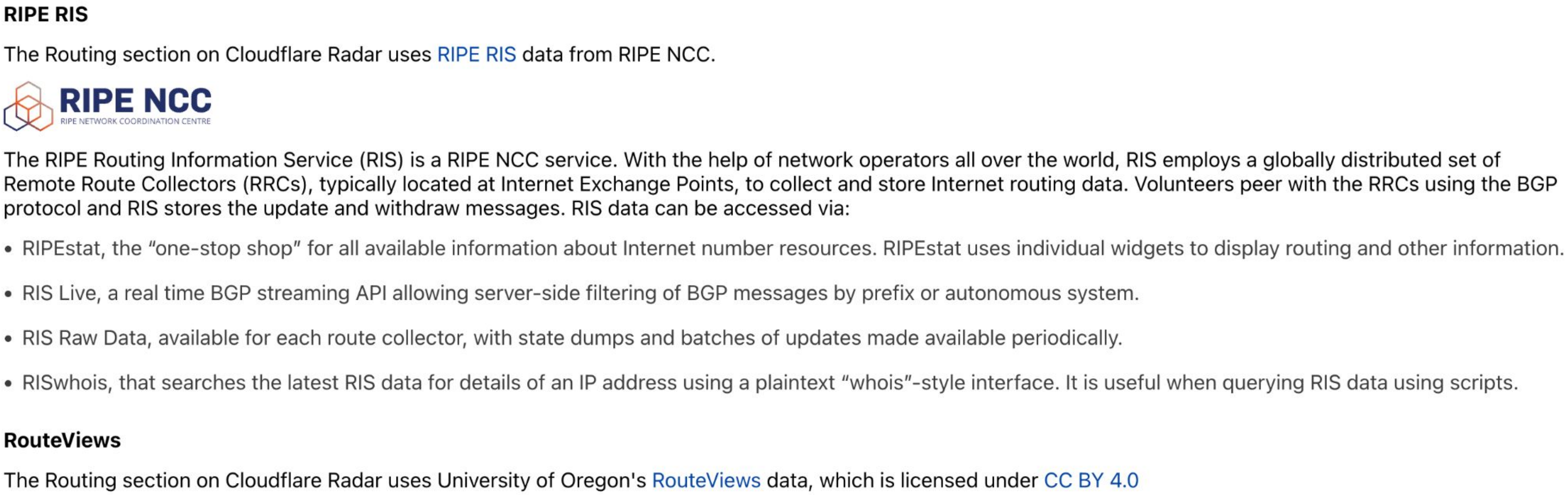
2025–02-11 GMI-AIMS at UCSD/CAIDA
Mingwei Zhang
mingwei@cloudflare.com

Overview

- About me:
  - Ex-CAIDA, PhD in CS from U of Oregon (home of RouteViews)
  - Open-source maintainer for BGPKIT
  - Cloudflare engineer working on public side of Cloudflare Radar
  - Contact
    - mingwei@cloudflare.com (for Cloudflare-related)
    - mingwei@bgpkit.com (for everything else)
- Topics
  - Cloudflare Radar Routing section
  - Building BGP data pipelines with Rust

Cloudflare Radar for Routing Info

Acknowledgement



RIPE RIS

The Routing section on Cloudflare Radar uses RIPE RIS data from RIPE NCC.

RIPE NCC
RIPE NETWORK COORDINATION CENTRE

The RIPE Routing Information Service (RIS) is a RIPE NCC service. With the help of network operators all over the world, RIS employs a globally distributed set of Remote Route Collectors (RRCs), typically located at Internet Exchange Points, to collect and store Internet routing data. Volunteers peer with the RRCs using the BGP protocol and RIS stores the update and withdraw messages. RIS data can be accessed via:

- RIPEstat, the “one-stop shop” for all available information about Internet number resources. RIPEstat uses individual widgets to display routing and other information.
- RIS Live, a real time BGP streaming API allowing server-side filtering of BGP messages by prefix or autonomous system.
- RIS Raw Data, available for each route collector, with state dumps and batches of updates made available periodically.
- RISwhois, that searches the latest RIS data for details of an IP address using a plaintext “whois”-style interface. It is useful when querying RIS data using scripts.

RouteViews

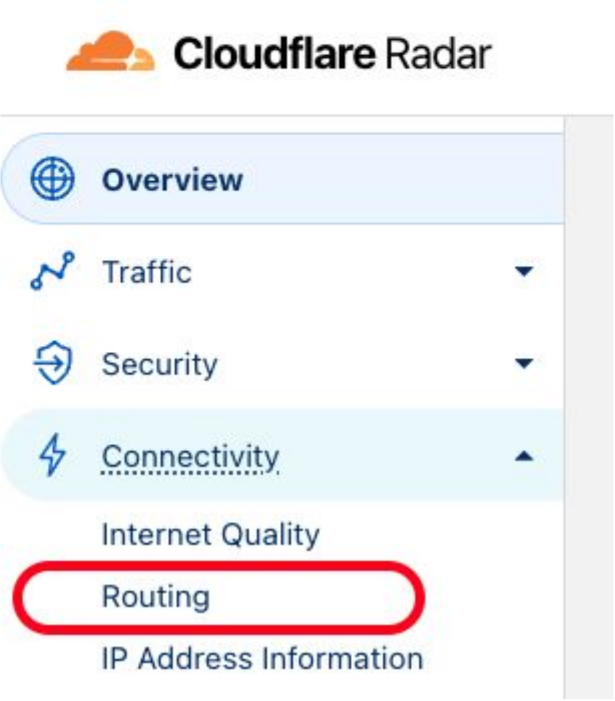
The Routing section on Cloudflare Radar uses University of Oregon's RouteViews data, which is licensed under CC BY 4.0

Cloudflare Radar

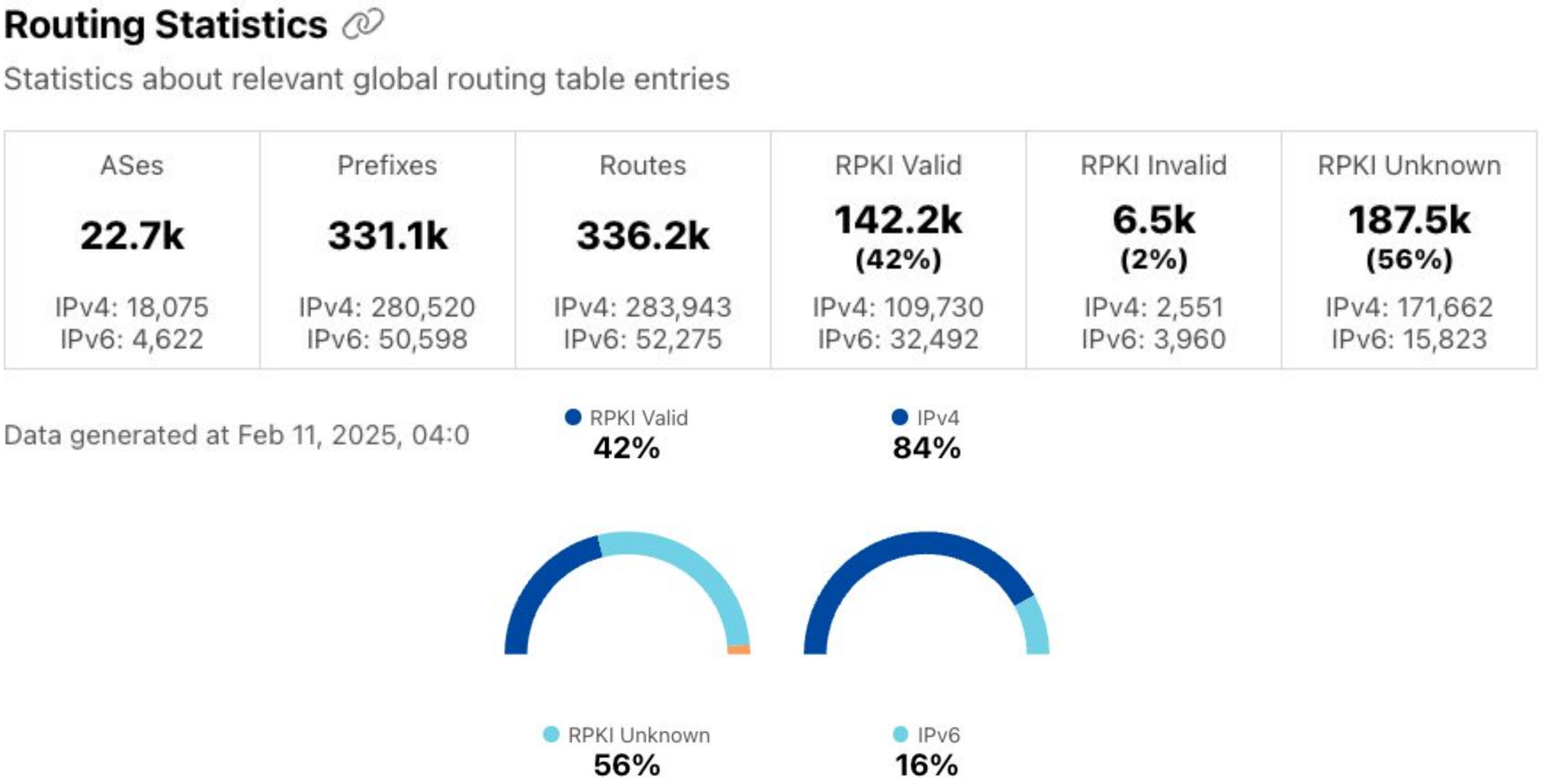
- Hub for Internet traffic, attack, and technology trends and insights
- Freely available, intended to be as open as possible
- All data and charts shown on the website can be consumed from also freely available public APIs

https://radar.cloudflare.com/

Routing Info on Cloudflare Radar



- Routing statistics
- RPKI deployment
- Hijacks and leaks
- Announced address space
  - both v4 and v6 overtime
- AS-level connectivity
- All extracted from public BGP data and a few other open datasets

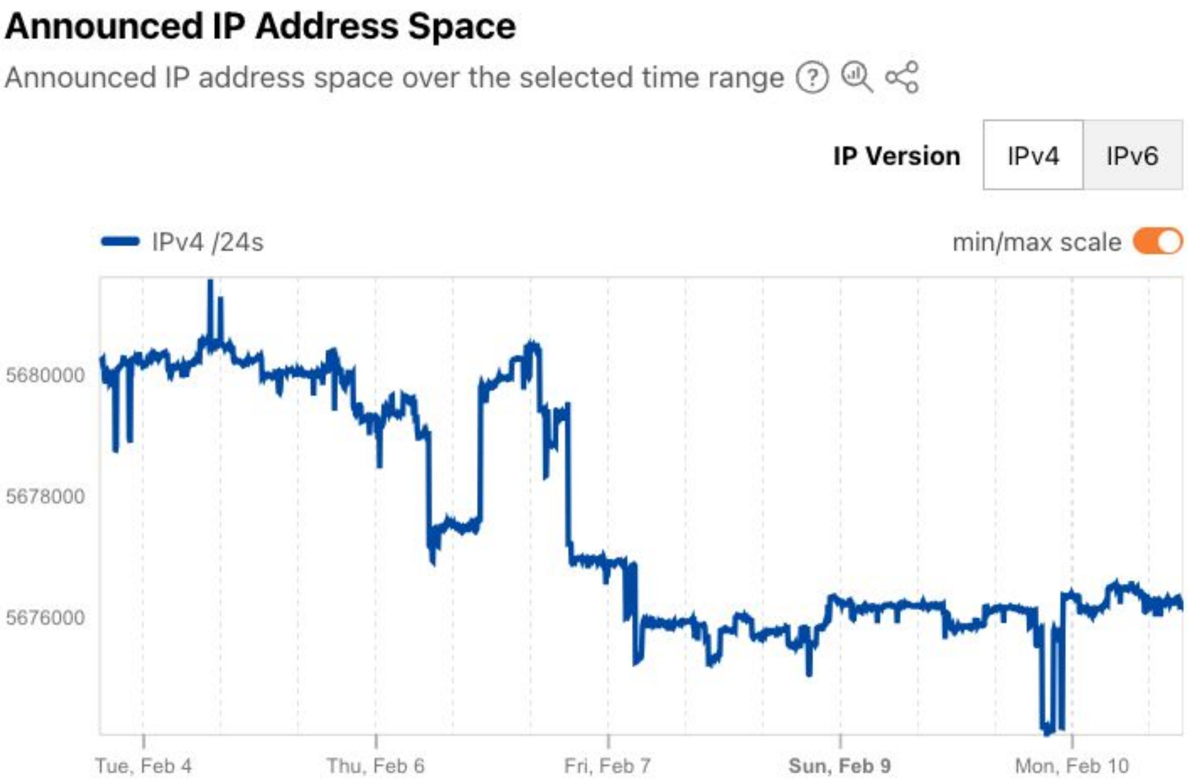


Routing Statistics

Statistics about relevant global routing table entries

ASes	Prefixes	Routes	RPKI Valid	RPKI Invalid	RPKI Unknown
22.7k	**331.1k**	**336.2k**	**142.2k** **(42%)**	**6.5k** **(2%)**	**187.5k** **(56%)**
IPv4: 18,075 IPv6: 4,622	IPv4: 280,520 IPv6: 50,598	IPv4: 283,943 IPv6: 52,275	IPv4: 109,730 IPv6: 32,492	IPv4: 2,551 IPv6: 3,960	IPv4: 171,662 IPv6: 15,823

Data generated at Feb 11, 2025, 04:0

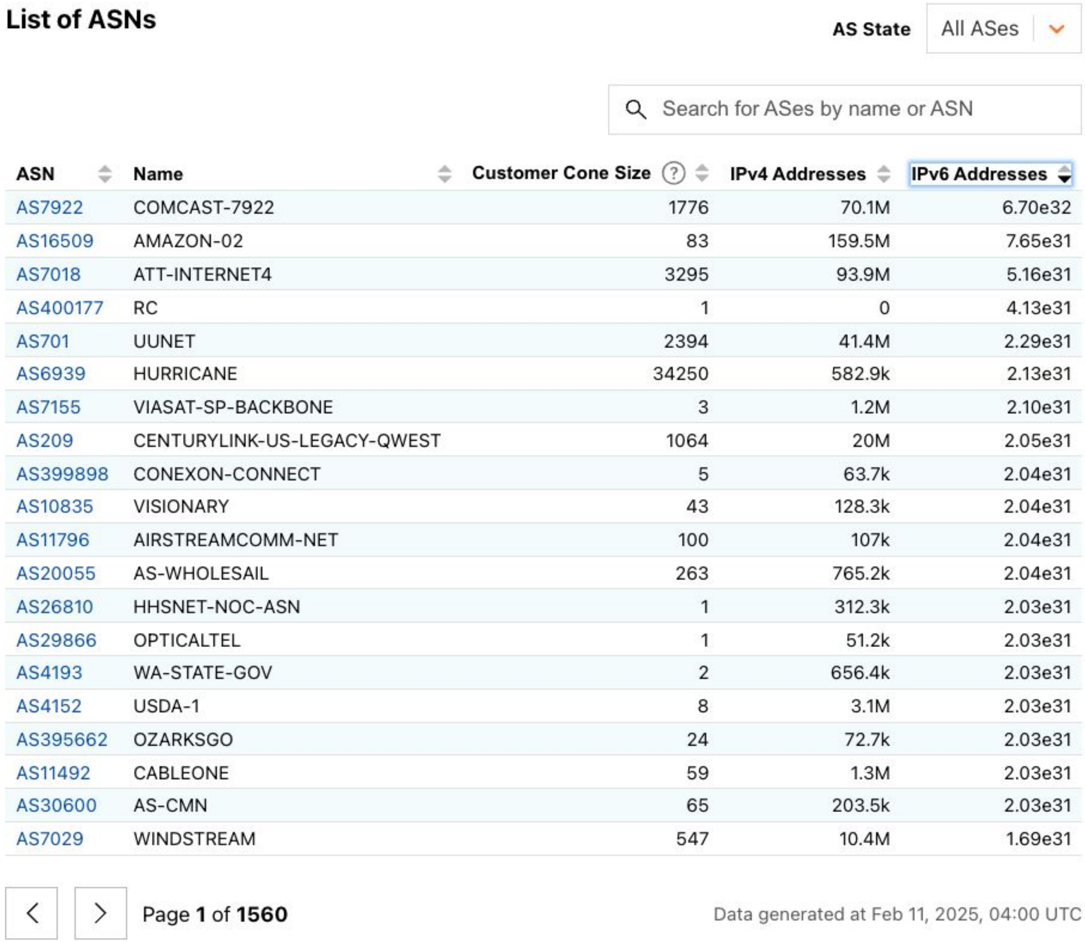


Announced IP Address Space

Announced IP address space over the selected time range

IP Version: IPv4 | IPv6

min/max scale



List of ASNs

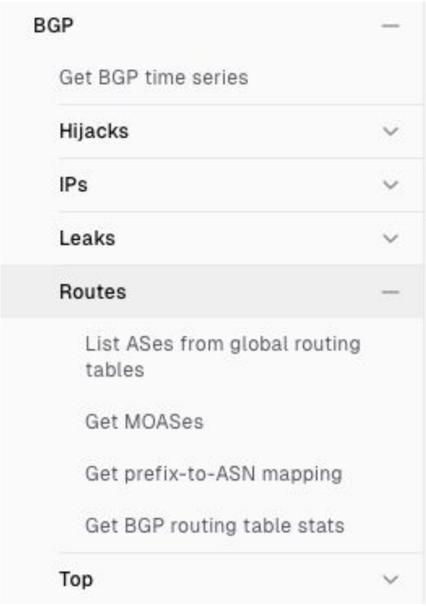
ASN	Name	Customer Cone Size	IPv4 Addresses	IPv6 Addresses
AS7922	COMCAST-7922	1776	70.1M	6.70e32
AS16509	AMAZON-02	83	159.5M	7.65e31
AS7018	ATT-INTERNET4	3295	93.9M	5.16e31
AS400177	RC	1	0	4.13e31
AS701	UUNET	2394	41.4M	2.29e31
AS6939	HURRICANE	34250	582.9k	2.13e31
AS7155	VIASAT-SP-BACKBONE	3	1.2M	2.10e31
AS209	CENTURYLINK-US-LEGACY-QWEST	1064	20M	2.05e31
AS399898	CONEXON-CONNECT	5	63.7k	2.04e31
AS10835	VISIONARY	43	128.3k	2.04e31
AS11796	AIRSTREAMCOMM-NET	100	107k	2.04e31
AS20055	AS-WHOLESAIL	263	765.2k	2.04e31
AS26810	HHSNET-NOC-ASN	1	312.3k	2.03e31
AS29866	OPTICALTEL	1	51.2k	2.03e31
AS4193	WA-STATE-GOV	2	656.4k	2.03e31
AS4152	USDA-1	8	3.1M	2.03e31
AS395662	OZARKSGO	24	72.7k	2.03e31
AS11492	CABLEONE	59	1.3M	2.03e31
AS30600	AS-CMN	65	203.5k	2.03e31
AS7029	WINDSTREAM	547	10.4M	1.69e31

Page **1** of **1560**

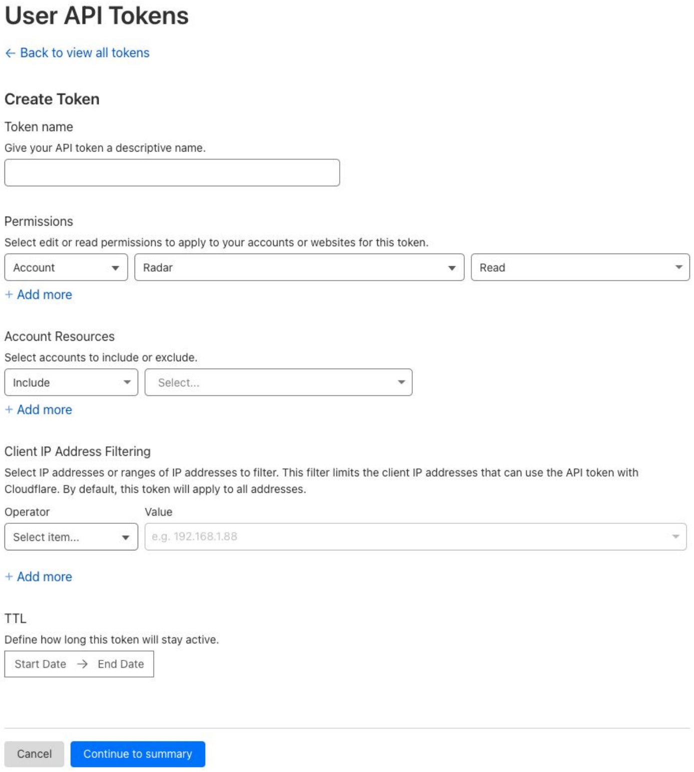
Data generated at Feb 11, 2025, 04:00 UTC

Cloudflare Radar API

- https://developers.cloudflare.com/api/resources/radar/
- /BGP is where I live
  - Hijacks
  - Leaks
  - IP space
  - Prefix-to-asn mapping
  - MOASes
  - ...


BGP
Get BGP time series
Hijacks
IPs
Leaks
Routes
List ASes from global routing tables
Get MOASes
Get prefix-to-ASN mapping
Get BGP routing table stats
Top

Accessing Cloudflare Radar API



User API Tokens

← Back to view all tokens

Create Token

Token name

Give your API token a descriptive name.

Permissions

Select edit or read permissions to apply to your accounts or websites for this token.

Account	Radar	Read

+ Add more

Account Resources

Select accounts to include or exclude.

Include	Select...

+ Add more

Client IP Address Filtering

Select IP addresses or ranges of IP addresses to filter. This filter limits the client IP addresses that can use the API token with Cloudflare. By default, this token will apply to all addresses.

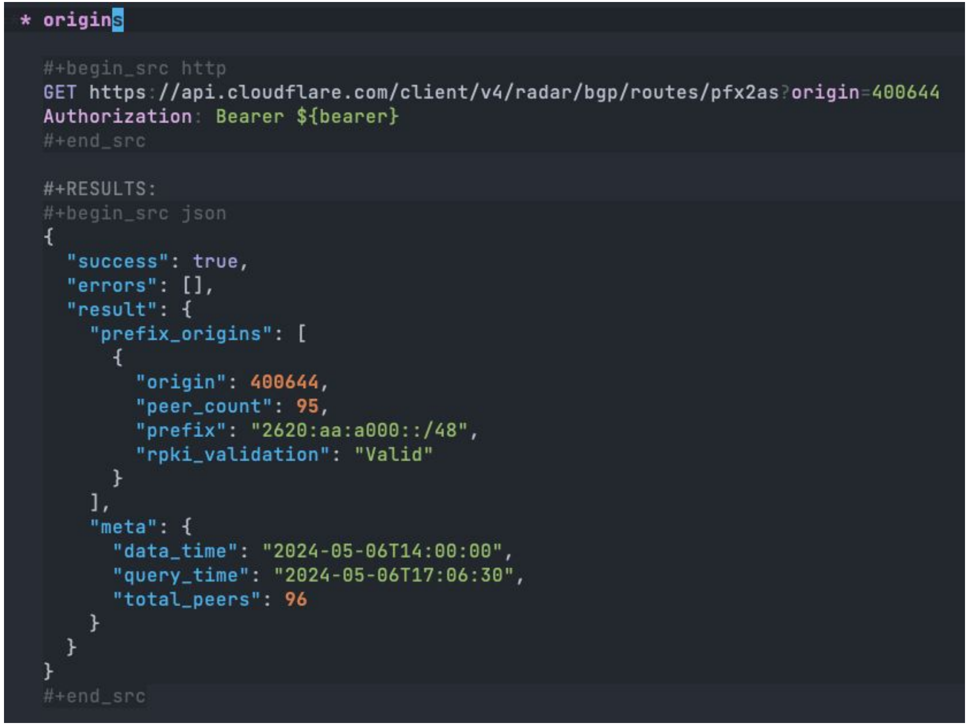
Operator	Value
Select item...	e.g. 192.168.1.88

+ Add more

TTL

Define how long this token will stay active.

Start Date → End Date

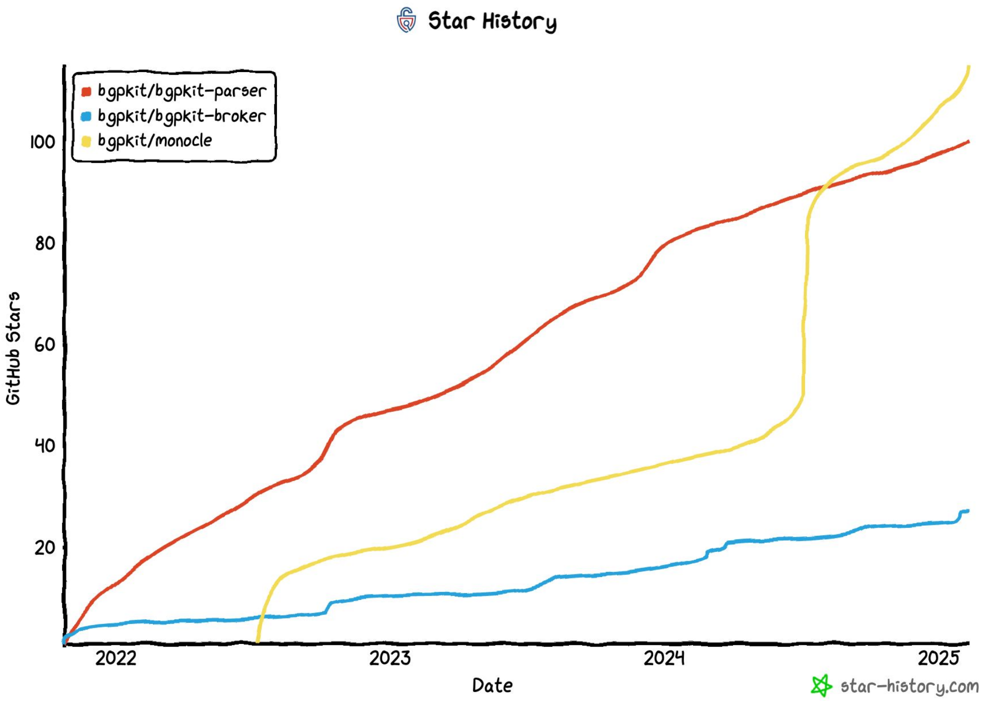


```
* origins

  #+begin_src http
  GET https://api.cloudflare.com/client/v4/radar/bgp/routes/pfx2as?origin=400644
  Authorization: Bearer ${bearer}
  #+end_src

  #+RESULTS:
  #+begin_src json
  {
    "success": true,
    "errors": [],
    "result": {
      "prefix_origins": [
        {
          "origin": 400644,
          "peer_count": 95,
          "prefix": "2620:aa:a000::/48",
          "rpki_validation": "Valid"
        }
      ],
      "meta": {
        "data_time": "2024-05-06T14:00:00",
        "query_time": "2024-05-06T17:06:30",
        "total_peers": 96
      }
    }
  }
  #+end_src
```

BGPKIT for BGP Data Pipelines

github.com/bgpkit

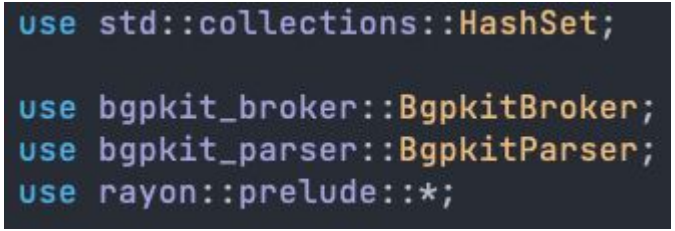
- Rust-based BGP data tools and SDKs
- Build pipelines by writing "simpler" Rust code
  - Hijack detection
  - Leak detection
  - Routing data extraction
- MIT-license for all code and data



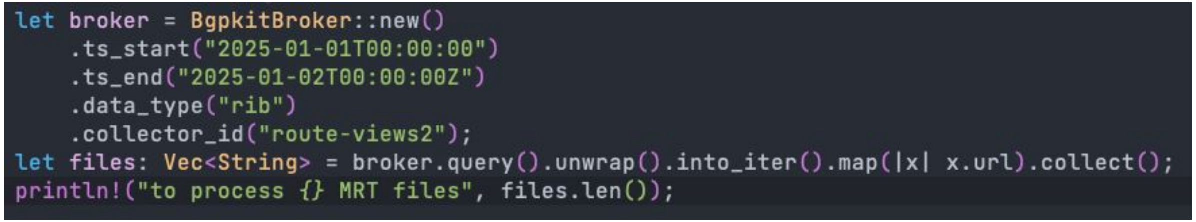
Live coding Example: find all prefixes originated by AS400644

- Goal:
  - Examine all BGP announcements originated by AS400644
  - Print all unique prefixes during 2025-01-01
  - Restrict to a single collector
  - As fast as possible

Example: find all prefixes originated by AS400644 in 2025-01-01

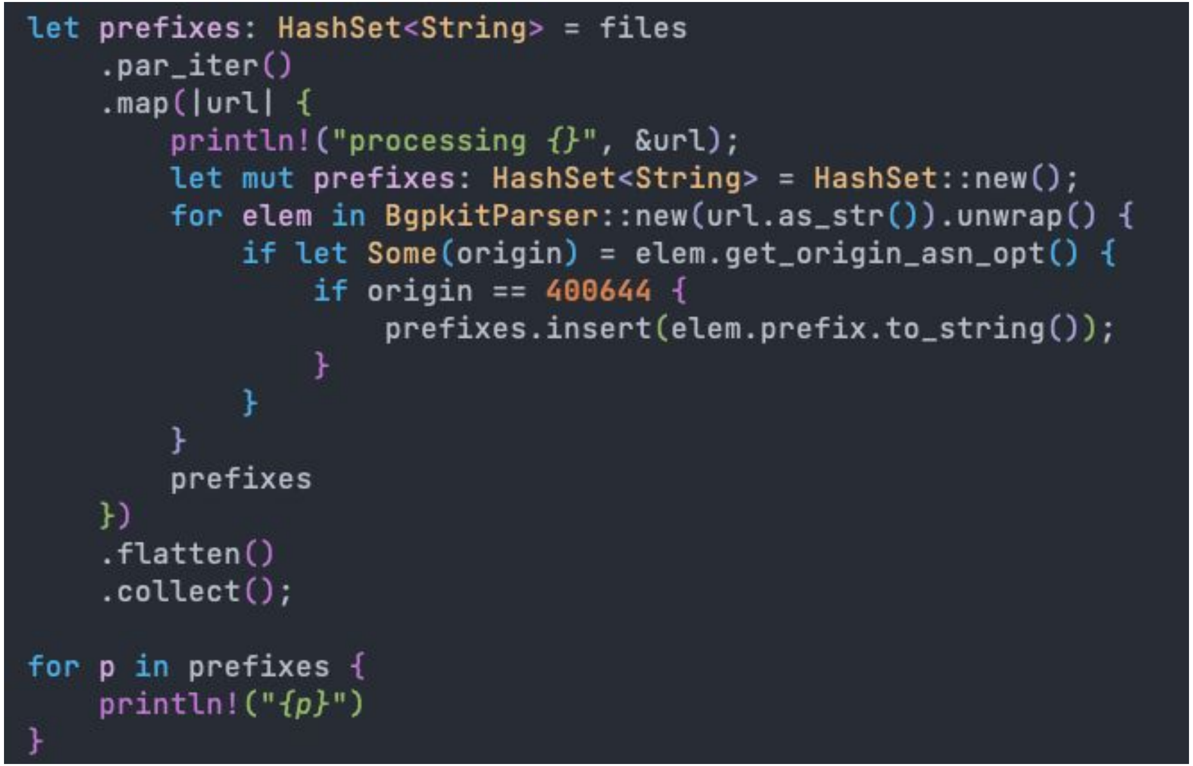


```rust
use std::collections::HashSet;

use bgpkit_broker::BgpkitBroker;
use bgpkit_parser::BgpkitParser;
use rayon::prelude::*;
```



```rust
let broker = BgpkitBroker::new()
    .ts_start("2025-01-01T00:00:00")
    .ts_end("2025-01-02T00:00:00Z")
    .data_type("rib")
    .collector_id("route-views2");
let files: Vec<String> = broker.query().unwrap().into_iter().map(|x| x.url).collect();
println!("to process {} MRT files", files.len());
```

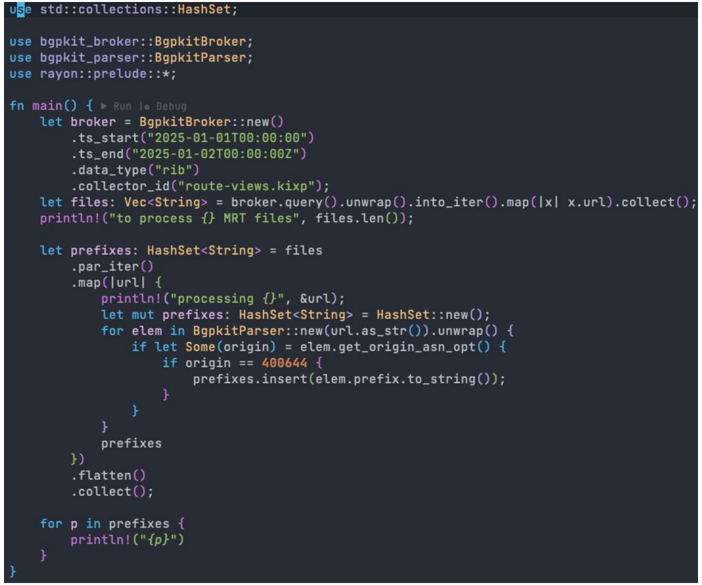
Example: find all prefixes originated by AS400644 in 2025-01-01



```rust
let prefixes: HashSet<String> = files
    .par_iter()
    .map(|url| {
        println!("processing {}", &url);
        let mut prefixes: HashSet<String> = HashSet::new();
        for elem in BgpkitParser::new(url.as_str()).unwrap() {
            if let Some(origin) = elem.get_origin_asn_opt() {
                if origin == 400644 {
                    prefixes.insert(elem.prefix.to_string());
                }
            }
        }
        prefixes
    })
    .flatten()
    .collect();

for p in prefixes {
    println!("{p}")
}
```

Example: find all prefixes originated by AS400644 in 2025-01-01

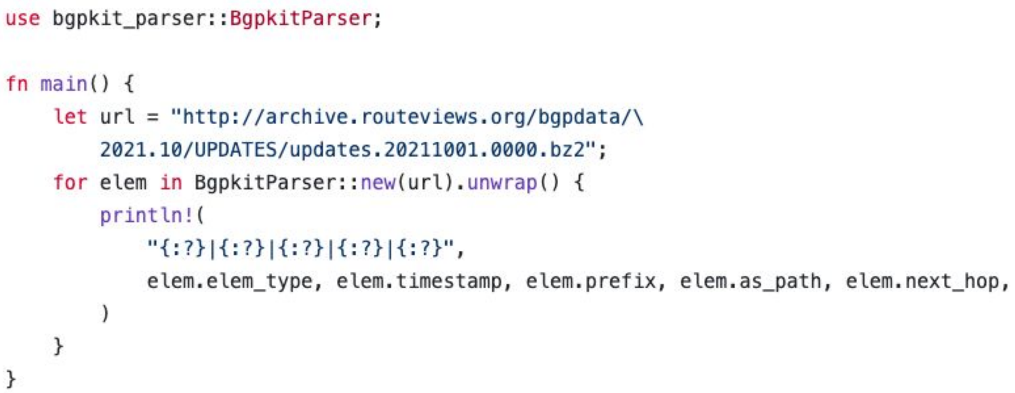
- 35 lines of code
- Concurrent parsing using all available cores



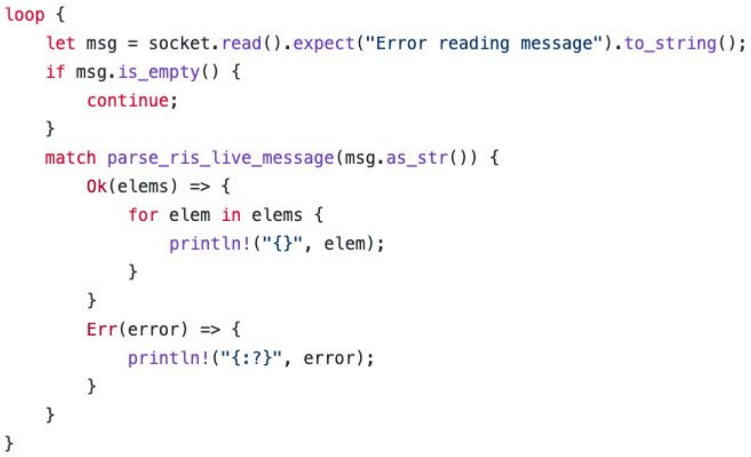
```rust
use std::collections::HashSet;

use bgpkit_broker::BgpkitBroker;
use bgpkit_parser::BgpkitParser;
use rayon::prelude::*;

fn main() {
    let broker = BgpkitBroker::new()
        .ts_start("2025-01-01T00:00:00")
        .ts_end("2025-01-02T00:00:00Z")
        .data_type("rib")
        .collector_id("route-views.kixp");
    let files: Vec<String> = broker.query().unwrap().into_iter().map(|x| x.url).collect();
    println!("to process {} MRT files", files.len());

    let prefixes: HashSet<String> = files
        .par_iter()
        .map(|url| {
            println!("processing {}", &url);
            let mut prefixes: HashSet<String> = HashSet::new();
            for elem in BgpkitParser::new(url.as_str()).unwrap() {
                if let Some(origin) = elem.get_origin_asn_opt() {
                    if origin == 400644 {
                        prefixes.insert(elem.prefix.to_string());
                    }
                }
            }
            prefixes
        })
        .flatten()
        .collect();

    for p in prefixes {
        println!("{p}")
    }
}
```

bgpkit-parser

- BGP/MRT/BMP message parser
  - Parsing BGP raw bytes
  - Decoding and encoding MRT files for RIB dumps and updates
  - Handles remote/local raw/compressed files seamlessly



```rust
use bgpkit_parser::BgpkitParser;

fn main() {
    let url = "http://archive.routeviews.org/bgpdata/\
        2021.10/UPDATES/updates.20211001.0000.bz2";
    for elem in BgpkitParser::new(url).unwrap() {
        println!(
            "{:?}|{:?}|{:?}|{:?}|{:?}",
            elem.elem_type, elem.timestamp, elem.prefix, elem.as_path, elem.next_hop,
        )
    }
}
```



```rust
loop {
    let msg = socket.read().expect("Error reading message").to_string();
    if msg.is_empty() {
        continue;
    }
    match parse_ris_live_message(msg.as_str()) {
        Ok(elems) => {
            for elem in elems {
                println!("{}", elem);
            }
        }
        Err(error) => {
            println!("{:?}", error);
        }
    }
}
```

bgpkit-parser

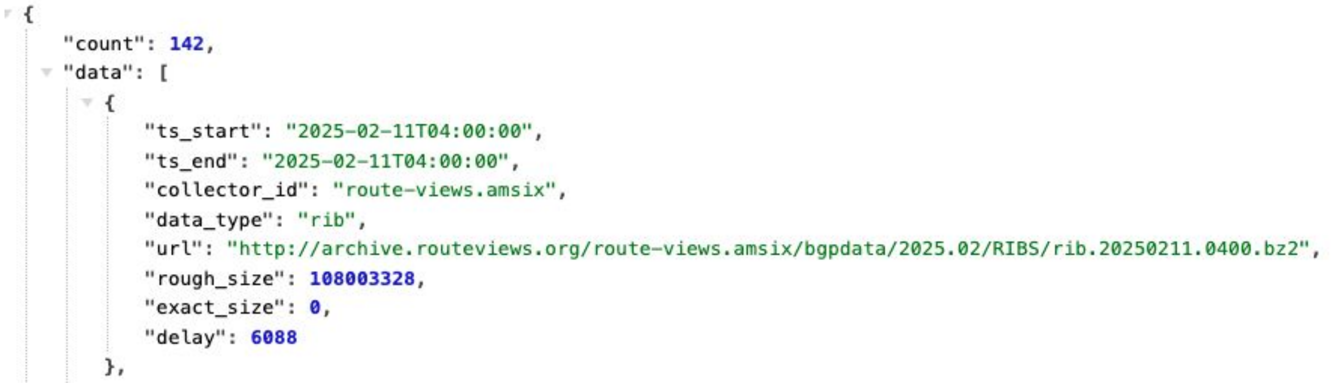
- Continuously adding supports for relevant RFCs
  - https://github.com/bgpkit/bgpkit-parser/tree/main?tab=readme-ov-file#rfcs-support
- Recent ones includes
  - ASPA
  - RFC9234: only-to-customer
  - BMP Adj-RIB-Out and Local RIB
  - BGP extended and large communities

bgpkit-parser

- Support processing live-streams
  - Support BMP streams, raw BMP or openBMP (RouteViews Kafka)
  - Support RIS-Live websocket stream
  - Asynchronous or synchronous
- Exporting back to MRT files
  - Parse, filter, or construct BGP messages
  - Exporting selected messages back to MRT files

bgpkit-broker

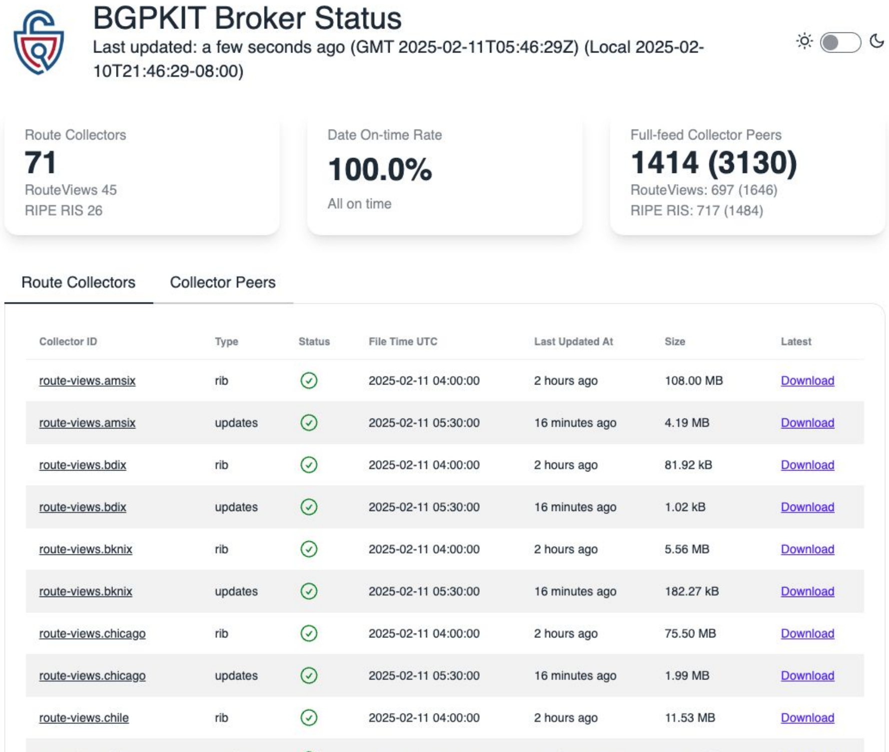
- Fetch and index meta data for public MRT archives
  - RouteViews and RIPE RIS
  - Timestamps, collector, type, URL, rough size, data delay
- Rust, Python SDK
- RESTful API
- “What RIB dump/updates files are available between time A and B?”



```
{
    "count": 142,
    "data": [
        {
            "ts_start": "2025-02-11T04:00:00",
            "ts_end": "2025-02-11T04:00:00",
            "collector_id": "route-views.amsix",
            "data_type": "rib",
            "url": "http://archive.routeviews.org/route-views.amsix/bgpdata/2025.02/RIBS/rib.20250211.0400.bz2",
            "rough_size": 108003328,
            "exact_size": 0,
            "delay": 6088
        },
```

bgpkit-broker

- “What’s the latest available files from all collectors?”
- “Who are the peers of these collectors and how many prefixes do they see?”
- https://status.broker.bgpkit.com/
- https://api.bgpkit.com/



BGPKIT Broker Status

Last updated: a few seconds ago (GMT 2025-02-11T05:46:29Z) (Local 2025-02-10T21:46:29-08:00)

Route Collectors
71
RouteViews 45
RIPE RIS 26

Date On-time Rate
100.0%
All on time

Full-feed Collector Peers
1414 (3130)
RouteViews: 697 (1646)
RIPE RIS: 717 (1484)

Collector ID	Type	Status	File Time UTC	Last Updated At	Size	Latest
route-views.amsix	rib	✓	2025-02-11 04:00:00	2 hours ago	108.00 MB	Download
route-views.amsix	updates	✓	2025-02-11 05:30:00	16 minutes ago	4.19 MB	Download
route-views.bdix	rib	✓	2025-02-11 04:00:00	2 hours ago	81.92 kB	Download
route-views.bdix	updates	✓	2025-02-11 05:30:00	16 minutes ago	1.02 kB	Download
route-views.bknix	rib	✓	2025-02-11 04:00:00	2 hours ago	5.56 MB	Download
route-views.bknix	updates	✓	2025-02-11 05:30:00	16 minutes ago	182.27 kB	Download
route-views.chicago	rib	✓	2025-02-11 04:00:00	2 hours ago	75.50 MB	Download
route-views.chicago	updates	✓	2025-02-11 05:30:00	16 minutes ago	1.99 MB	Download
route-views.chile	rib	✓	2025-02-11 04:00:00	2 hours ago	11.53 MB	Download

BGP Event Analysis with monocle

https://github.com/bgpkit/monocle