Modernizing BGP Data Access with BGPFiend

Thomas Krenc, Justin Loye and Dimitrios Giakatos
IIJ Research Laboratory
{tkrenc,jloye,dimitrios}@iij.ad.jp

Hans Kuhn and Owen Conway
RouteViews, University of Oregon
{hans,owen}@routeviews.org

Ties de Kock
RIPE NCC
tdekock@ripe.net

kc claffy
CAIDA, UC San Diego
kc@caida.org

***Abstract*—BGP data is essential for understanding Internet routing and security, but the volume of data required for analysis has grown substantially over time. To preserve compatibility with long-standing consumer workflows, public route collector archives such as RIPE RIS and RouteViews continue to rely on a more than 20-year-old storage and dissemination model. As a result, BGP data consumers must download and process large volumes of often irrelevant data, making analysis bandwidth-intensive and inefficient.**

In this poster paper, we present BGPFiend, a web service that modernizes BGP data access without replacing existing archives or disrupting established workflows. BGPFiend enables selective, collector-side preprocessing of BGP data and returns only explicitly requested results to consumers using generic HTTP tools. Through an evaluation of prefix, community, and peer AS queries using an initial prototype, we show that BGPFiend considerably reduces consumer-side data transfer and query duration compared to conventional consumer-side parsing. By preserving existing formats and trust assumptions while enabling incremental adoption of modern archival techniques, BGPFiend provides a practical path toward scalable BGP data analysis.

I. Introduction

The Border Gateway Protocol (BGP) [1] is the Internet's interdomain routing protocol and a primary data source for Internet measurement and operational research. Large-scale BGP data is collected by long-running route collector projects such as RIPE RIS [2] and RouteViews [3].

Despite sustained growth in BGP data volume, the underlying archival model has remained largely unchanged for more than two decades. Route collector archives disseminate data primarily by time range and collector, while fine-grained filtering by prefix, community, or peer AS is typically performed at the consumer-side after entire MRT [4], [5] files have been downloaded (e.g., using tools such as BGPStream [6] or BGPKIT [7]). As a result, consumers must transfer and parse large volumes of data, often hundreds of megabytes per query, even when only a small subset is relevant.

We introduce BGPFiend, a web service that enables collector-side preprocessing and fine-grained filtering while preserving existing archives and long-standing consumer workflows. Returning only explicitly requested results, BGPFiend substantially reduces data transfer and improves the efficiency of consumer-side analysis. It targets interactive, query-driven access and complements bulk-download and streaming pipelines. The feasibility of this approach is demonstrated through an initial prototype implementation, evaluated on representative prefix, community, and peer AS queries.

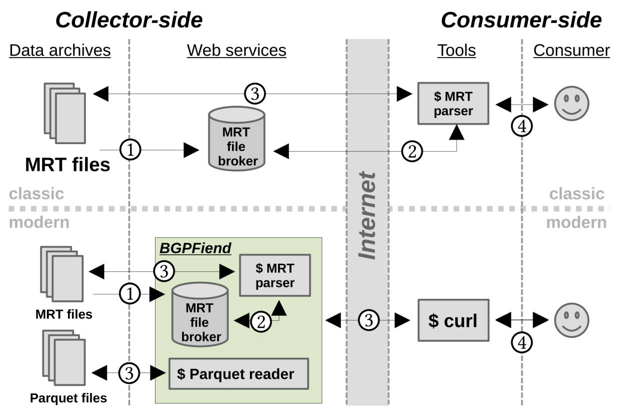

Fig. 1. **Comparison of the classic (top) and modern (bottom) BGP Data Access Architectures.** BGPFiend (green box) shifts parsing, filtering, and preprocessing to the collector side, improving information flow through ① MRT file indexing, ② file selection, ③ BGP data delivery, and ④ consumer-side analysis.

This poster highlights the following additional advantages of BGPFiend:

- Backward-compatible deployment that preserves existing collection infrastructure and consumer workflows.
- HTTP-native BGP data access for consumers via generic tools, removing dependence on specialized MRT parsers.
- Filtering and preprocessing at the collector-side, reducing consumer-side data transfer and processing requirements.
- Support for incremental adoption of improved indexing and alternative archival formats such as Parquet [8].

II. Classic and modern Architecture

Figure 1 contrasts two BGP data access architectures: a classic consumer-side workflow (top) and a modern collector-side workflow enabled by BGPFiend (bottom).

In the classic workflow, a consumer retrieves compressed BGP data from route collector archives such as RIPE RIS or RouteViews using MRT parsers such as BGPStream or BGPKIT ④. These tools rely on an MRT file broker to identify relevant MRT files based on coarse parameters such as time range or collector ②. The broker periodically scans the archives for newly available files ①, which are then downloaded in full from the archives over the Internet ③. Fine-grained filtering, e.g., by prefix, community, or peer AS, is applied only after the data has been transferred to the consumer.

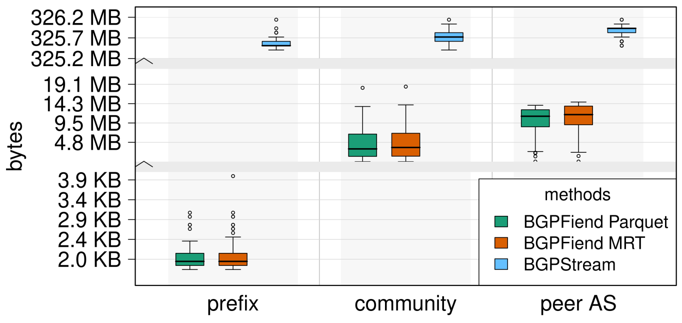

Fig. 2. **Download Volume.** Boxplots showing consumer-side bytes downloaded for prefix, community, and peer AS queries using BGPStream (classic) and BGPFiend with MRT and Parquet (modern). Using BGPFiend, data is reduced by approximately 96% from 325 MB to $<$ 20 MB for community and peer AS queries, and drops to $<$ 3 KB for prefix queries.

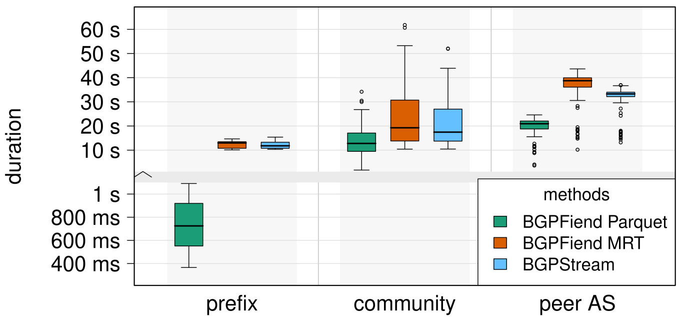

Fig. 3. **End-to-End Duration.** Boxplots showing end-to-end query duration for prefix, community, and peer AS queries using BGPStream, MRT, and Parquet. The modern approach consistently reduces duration, with Parquet achieving sub-second response times for prefix queries and lower median duration and variability for community and peer AS queries.

In the modern workflow, a consumer accesses BGP data using generic HTTP tools such as `curl` or a web browser ④, without requiring consumer-side MRT parsing. HTTP requests are handled on the collector side by BGPFiend (green box), a web service that provides a uniform query interface over the underlying archives. BGPFiend uses MRT parsers to perform file selection, filtering, and preprocessing, returning only explicitly requested data to the consumer ③. This shifts the bulk of data transfer and processing away from the consumer, reducing bandwidth and local processing requirements.

By acting as an intermediary layer, BGPFiend preserves compatibility with existing archives while enabling the integration of improved indexing mechanisms and alternative archival formats, such as Parquet, a columnar storage format, without disrupting established workflows.

The simplified access model allows functionally equivalent queries to be expressed using either conventional BGP tools or HTTP-based queries. Example command-line queries for the prefix case are:

```
$ curl bgpfiend/parquet?k=prefix
$ curl bgpfiend/mrt?k=prefix
$ bgpreader -k prefix
```

III. Analysis

We evaluate the modern approach by retrieving preprocessed BGP data via BGPFiend (Parquet and MRT) and compare it against the classic consumer-side workflow using BGPStream. We measure the download volume and the end-to-end duration of prefix, community, and peer AS queries using 100 randomly selected values per query type.

Figure 2 shows the download volume across the three methods. For prefix queries, transferred data drops from approximately 325 MB with BGPStream to less than 3 KB. For community and peer AS queries, download volume is reduced by approximately 96%, from hundreds of megabytes to a few and tens of megabytes, respectively. This demonstrates that collector-side filtering removes most unnecessary data transfer.

Figure 3 shows end-to-end query duration. The modern approach consistently reduces duration, with the largest gains observed for prefix queries. Using Parquet, prefix queries complete in sub-second time, while BGPStream and MRT require more than 10 s. For community and peer AS queries, duration reductions are smaller but systematic, with Parquet outperforming both BGPStream and MRT. For large result sets, performance converges as transfer and parsing dominate; BGPStream benefits from compressed MRT files, while BGPFiend incurs overhead from text-based streaming and server-side formatting.

Overall, relocating MRT file access and parsing at the collector side considerably reduces consumer-side data transfer and improves time-to-insight, while remaining compatible with existing workflows.

IV. Discussion and Outlook

Our results show that meaningful efficiency gains in BGP data analysis can be achieved without replacing existing archives or consumer tooling. By preserving established formats and access semantics, BGPFiend enables incremental evolution of BGP data access rather than a disruptive redesign. While this work focuses on download volume and duration, the same abstraction provides a foundation for additional capabilities, such as richer indexing, alternative archival formats, and caching. Future work includes broader deployment across route collector projects and extending BGPFiend with additional query primitives and data sources.

References

[1] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, IETF. [Online]. Available: https://www.rfc-editor.org/rfc/rfc4271
[2] RIPE NCC, "RIPE Routing Information Service (RIS)," https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/.
[3] University of Oregon, "RouteViews Project," https://www.routeviews.org.
[4] L. Blunk, M. Karir, and C. Labovitz, "Multi-Threaded Routing Toolkit (MRT) Routing Information Export Format," RFC 6396. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6396
[5] C. Petrie and T. King, "MRT Extensions for BGP Path Attributes," RFC 8050. [Online]. Available: https://www.rfc-editor.org/rfc/rfc8050
[6] CAIDA, "BGPStream," https://bgpstream.caida.org.
[7] BGPKIT Contributors, "BGPKIT," https://bgpkit.com.
[8] Apache Software Foundation, "Apache Parquet," https://parquet.apache.org/.