GMI-1.2.5.3&1.2.5.4: Improve and Document Data and Metadata APIs

1. Periscope: Unified Interface to Looking Glass Internet Measurement Servers

CAIDA's Periscope looking glass API provides a uniform interface to hundreds of Looking Glass and Scamper servers with access to thousands of network probing vantage points (monitors) that can perform traceroute and BGP queries.

Improvements

We enhanced this API with features that include support for access to scamper processes running on BGP collectors (Note: policy to access such capability is determined by collectors and peers), new monitor *type* and *server* fields, and CAIDA's new Keycloak-based authentication/authorization framework.

We recently designed and implemented an authentication/authorization solution was for CAIDA's live Periscope application. Keycloak has a feature called groups, which is a set of users (similar to unix groups). Roles can be assigned to groups, so that every user in that group inherits the roles assigned to the group. In Keycloak, roles are organized by the client (application) they belong to. We implemented a client called *periscope-api*, which has roles *read-LG*, *read-SCRV*, *measure-LG*, *measure-SCRV* (we will be adding additional roles, e.g., *admin* or *view-usage-stats*.) Roles can also be composite, meaning they contain other roles. E.g., *measure-LG* will contain *read-LG*, so that any user with *measure-LG* permission will automatically get *read-LG* so they can read the results of measurements they created.

Documentation

The service consists of three modules:

- A *Database* stores the looking glass queries to be executed
- *Agents* that retrieve the requested queries from the database, execute them, convert the results to standardized formats, and update the database with the results.
- A *Controller* assigns queries to Agents. Looking Glass servers mostly support low-frequency querying; if an Agent exceeds the LG provider's rate limits, Periscope will block the Agent and their queries will fail. The Controller is responsible for staying within the LG rate limits.

The Controller and the Database can run on the same host, but Agents run on different hosts with distinct *public* IP address. To support this requirement Periscope uses a cloud computing infrastructure that allows on-demand creation of computing instances. The current versions runs on Google's Cloud Platform (GC).

Detailed documentation can be found at https://github.com/CAIDA/periscope (public dataset)

2. Hoiho: Hostname-based Geolocation of IP addresses

Holistic Orthography of Internet Hostname Observations (Hoiho) is an open-source tool released as a part of scamper. It uses CAIDA's Macroscopic Internet Topology Data Kit (ITDK) and observed round trip times to infer regular expressions that extract apparent geolocation hints from hostnames. The ITDK contains a large dataset of routers with annotated hostnames, which are used as input to Hoiho for its inference rules (encoded as regular expressions) that extract these annotations.

Improvements

CAIDA created an HTTP API for Hoiho. This public interface provides hostname-location lookups. There are both a GET method (for simple lookups) and a POST method (for doing bulk lookups). It also provides metadata indicating the date of the dataset as well as the number of matches found. Load testing found it to be robust under moderate load, so we have not yet implemented manual throttling and simply request that users keep their requests under 1/sec.

Documentation

The HTTP API landing page (https://api.hoiho.caida.org/) displays SwaggerUI documentation of known endpoints and the data model used for return values, including error formats. The git repository for the code (https://gitlab.caida.org/CAIDA/hoiho/hoiho-api) includes documentation for how to run the API, as well as what the different return values might be and where to get input data. The most recent data was created for CAIDA's ITDK and is stored as a simple JSONL file. It is automatically generated from hostname data, but requires some human intervention to improve its accuracy.

3. Spoofer

Spoofer is a suite of open-source software tools to assess and report on the deployment of source address validation (SAV) best anti-spoofing practices. This client-server system periodically tests a network's ability to both send and receive packets with forged source IP addresses (spoofed packets). The CAIDA Spoofer Data API (https://api.spoofer.caida.org/) provides a public data interface to the publicly shareable data collected by the Spoofer service.

Improvements

The latest improvements include

- Created docker image for spoofer client using docker host networking driver
  https://hub.docker.com/r/kenkeys/spoofer

- Created a process for statically-linking the spoofer binary for the Linux command line.
- Incorporation of spoofer results into the AS Rank API. AS Rank now includes Spoofer information and links to the spoofer page for each ASN (e.g. https://asrank.caida.org/asns?asn=174 links to https://spoofer.caida.org/as.php?asn=174).
- Matching the contents of tables and graphs on both of the AS Rank and Spoofer websites with Spoofer information for each ASN displayed on as.php (only last year of data is considered).
- Adding a date search parameter
- Sorting by date in descending order
- Added functionality to return JSON-LD as well as a parameter (e.g., *itemsPerPage*) that users can use to get more results for a large request
- Transitioned spoofer to new server. Removed hard-coded configuration parameters, switched to systemd scripts

Documentation

The Spoofer API 1.0.0 can be found at https://api.spoofer.caida.org. To expand the documentation for each endpoint, click the GET buttons. Once expanded, each endpoint displays the example parameters with buttons to *Try it out*. The *Try it out* button provides a graphical interface that allows the user to build and execute queries to the Spoofer API and see the example responses.

We offer an example script, implemented in Perl, that programmatically queries the Spoofer Data API. The script takes command line arguments to query by date or *sessionid* with an optional Autonomous System Number (ASN). The script assumes the local system has the curl data transfer tool installed as well as the JSON::XS and Getopt::Long; PERL packages. At runtime, the script collects its command-line arguments and builds a query for the Spoofer Data API. The API returns a list of sessions one page (30 lines) at a time. The example script decodes each line of JSON and prints the page numbers (to STDERR so as not to send redirected results output to file) preceding each page of results received. The curl command makes use of the '-s' (silent) mode as well as '-X' switch to specify a custom GET command-line.

4. DZDB

CAIDA's DNS Zone Database (DZDB) is a platform providing access to time-series data derived from current and historical zone files provided by generic Top-Level Domains (gTLDs) participating in the Central Zone Data Service (CZDS) or directly by Registries Operators in compliance with corresponding license agreements.

The system was originally designed and developed by Ian Foster as http://dns.coffee. Ian has been working with CAIDA to transition it to CAIDA's cyberinfrastructure and use it as the basis for designing a more comprehensive system of DNS data and metadata and linking such data to other measurements. At this point DZDB and dns.coffee are using the same database that Ian Foster updates daily. We are working on transition DZDB to a separate database and on further development of DZDB branches of UI and API. We are in the process of signing an MOU with Ian documenting objectives and goals of this

collaboration, and conditions that parties agree to abide by while CAIDA continues developing open source DZDB software.

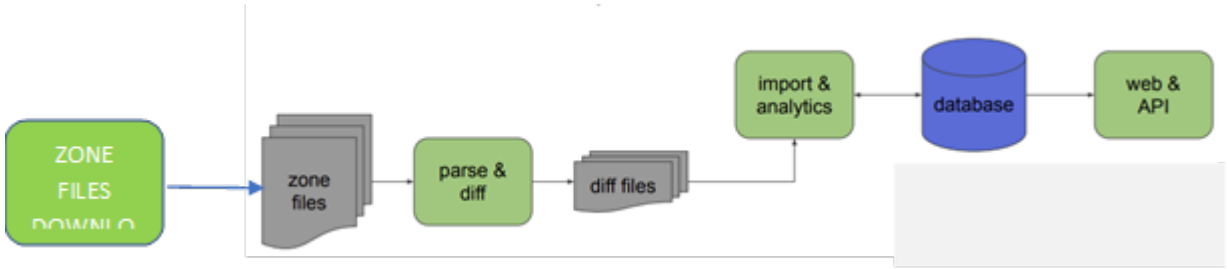
Improvements

We modified the DZDB API to let users type in a prefix and get all nameserver IPs/hostnames and domains served by that prefix. One of the implications of this functionality is that we could take blacklist data and study 'how many domains are served by these blacklisted IPs'. We discussed the possibility of creating a new database of blacklisted domains to ask questions of 'how many blacklisted domains are served by these IPs/prefixes'.

We added a SwaggerUI documentation page for the internal HTTP API (https://dzdb.caida.org/api) used by the web pages.

Documentation

The system consists of seven parts depicted at the figure below.



Zone Files Downloads

CAIDA uses the scripts documented in the following github repos to perform daily zone files downloads

- https://github.com/lanrat/czds
- https://github.com/lanrat/czds-balance
- https://github.com/lanrat/allxfr
- https://github.com/lanrat/czds-complain
- https://github.com/lanrat/zonewalk

Parse& diff AND Imports & Analytics

https://github.com/lanrat/zonetools

Web Interface and API

https://github.com/CAIDA/dzdb-web

https://dzdb.caida.org/api

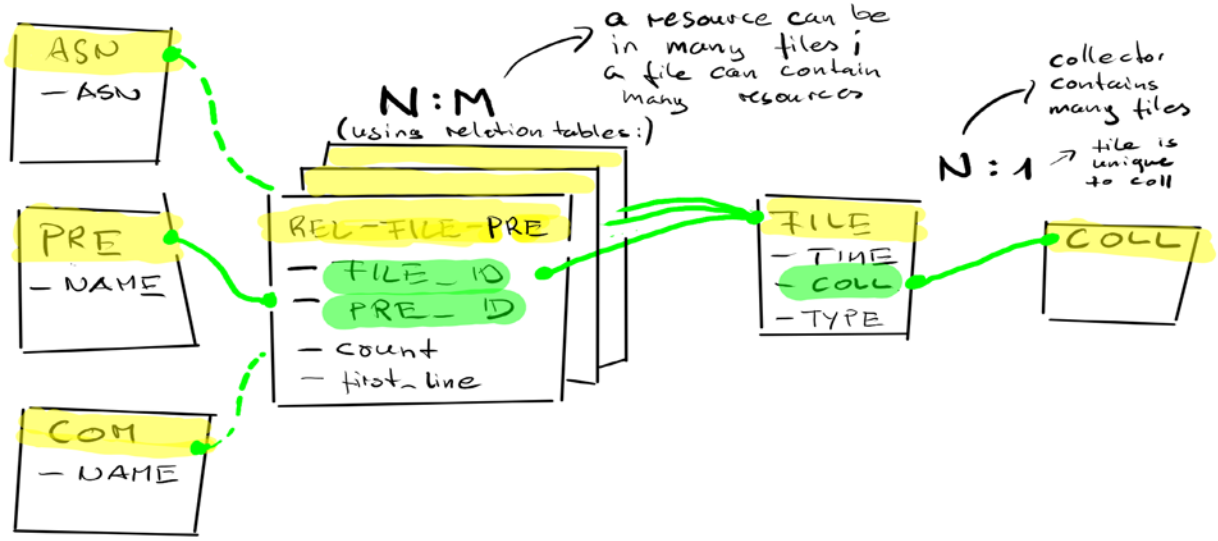
5. BGP2GO

To facilitate security research and analysis, we study the feasibility of indexing numeric identifiers over time: We index BGP prefixes, ASNs, communities, and IP addresses to data sets in which they occur. Currently, we process all BGP update files from RouteViews' route collectors. We prototyped BGP2Go, a web application that assists in selecting and obtaining relevant MRT data sets for further analysis. We hope to extend it to include other types of data, e.g., RIR allocation files, DNS (OpenIntel), DNS data. Indexing more data will facilitate correlation of activities of an identifier across data sets.

Improvements

We have finished indexing BGP update data beginning from 2012, and we index new data as it arrives into the RouteViews archive. We have added a caching mechanism that improves lookup speed on average by 40%.

Documentation



The figure above illustrates BGP2GO's data model. Our database indexes MRT files by ASNs, prefixes, and communities. We hope to add indexing by prefix-to-AS mappings, or larger communities. BGP2Go counts and reports their occurrences within those files, and finds each ID on a 15-minute granularity by default. MRT update files can be identified by time range (year, month, day) and collector.

The BGP2GO web application provides a graphical interface to explore and compile MRT file information (using the BGP metadata database) to facilitate access and sharing of relevant MRT files. With each lookup, a summary is presented to the user: the number of MRT files, approx. download size, the earliest and latest MRT files, and the involved collectors.

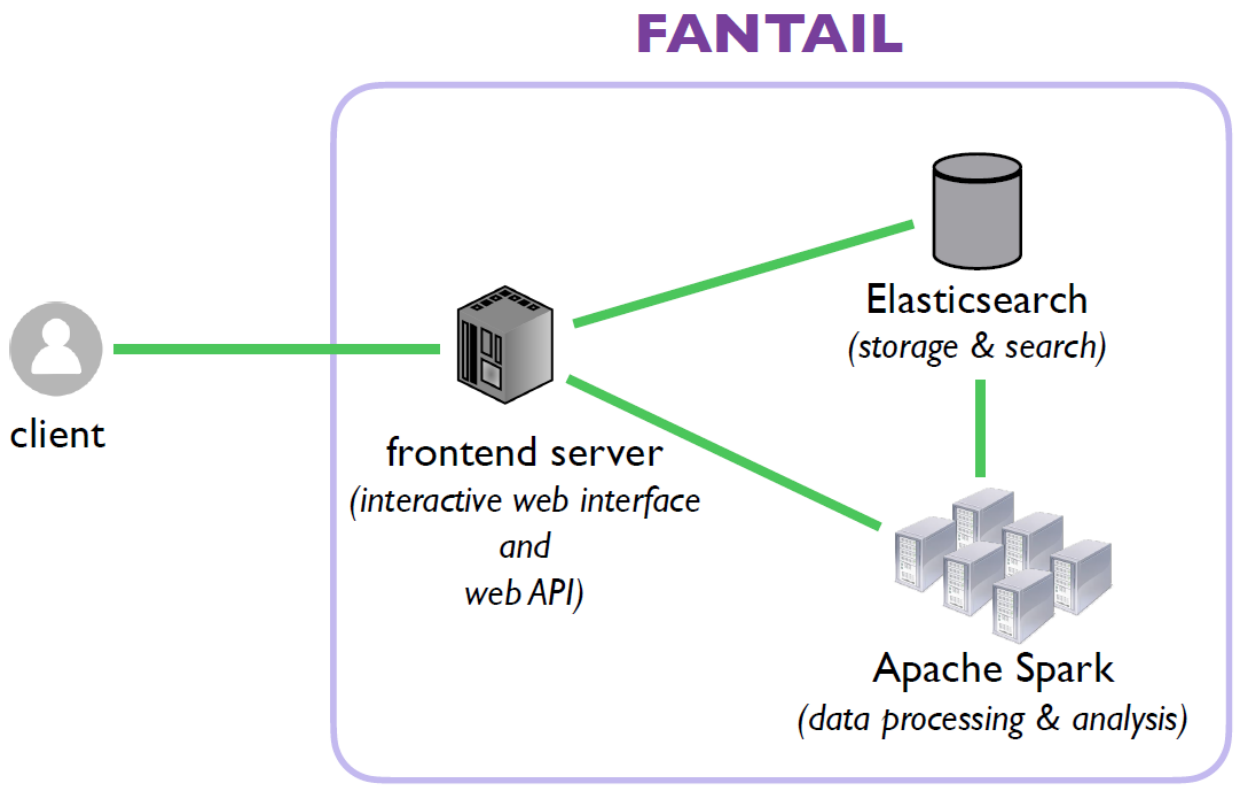
Documentation: https://gitlab.caida.org/CAIDA/caida-science-gateway/metadata-database/-/wikis/home
Slides: here

6. FANTAIL

Facilitating Advances in Network Topology Analysis (FANTAIL) system was developed to enable discovery of the full potential value of massive raw Internet end-to-end path measurement data sets, allowing researchers to use high-level queries to perform data processing and analysis tasks on matching traces without owning/operating a cluster, and without learning big data programming. FANTAIL is a four-component system:

- an interactive web interface (temporarily provided by Vela);
- an API built on web standards;
- a full-text search system based on Elasticsearch; and
- a big data processing system based on Apache Spark.

The FANTAIL system performs the queries, runs the analysis modules, and provides output for download for further analysis or processing by researchers on their own systems..



File formats documented at https://github.com/CAIDA/fantail/blob/master/web-app/docs/file-formats.md

The fantail-api script allows you to execute queries and analysis recipes via the FANTAIL API There are two major ways of executing queries with fantail-api. Use the low-level addr, dest, and neigh subcommands to execute standalone queries with maximum configurability. Use the high-level recipe subcommand to execute analysis recipes, which are predefined multi-step querying and processing pipelines with limited configurability, because most choices are purposely hard-coded by the recipe to provide a simple turn-key solution.

The addr, dest, and neigh subcommands accept the full set of available options, whereas the recipe subcommand accepts a subset of these options, as detailed in further sections.

FANTAIL executes queries in three major steps:

1. **Fetch** matching traceroute paths,
2. **Annotate** traceroute paths with one or more metadata, and
3. **Transform** traceroute paths.

The fetch step is always a part of a query, but the annotation and transformation steps are optional and may be configured with command-line options, except in the case of the recipe subcommand, since recipes configure the annotation and transformation steps themselves. Fantail API Client is documented at https://github.com/CAIDA/fantail/blob/master/web-app/bin/fantail-api

7. ASRANK

AS-RANK is CAIDA's ranking of Autonomous Systems (AS) (which approximately map to Internet Service Providers) and organizations (Orgs) (a collection of one or more ASes). This ranking is derived from topological data collected by Border Gateway Protocol (BGP) routing data collected by the Route Views Project and RIPE NCC.
ASes and Orgs are ranked by their customer cone size, which is the number of their direct and indirect customers. Note: We do not have data to rank ASes (ISPs) by traffic, revenue, users, or any other non-topological metric.

Improvements

We have added a default relationship sort to the individual AS details page. This places an AS's providers before its peers and peers before its customers. For an individual AS, its providers are its most important neighbors even if they have lower rank.

We continue to update AS Rank's inferences with operator ground truth, increasing the accuracy of the presented data.

Documentation

As-rank software consists of the following parts

1. Scripts that integrate heterogeneous data (AS-relationships, inferred relationships and customer cones, AS to organization mapping, Netacuity geolocation, etc.) to create a unified view of the Internet's Autonomous Systems.
2. A PostgreSQL database that stores indexes of the unified dataset.
3. A public GraphQL API and RESTFUL interface to that database (https://api.asrank.caida.org/v2/graphiql)
4. A webportal (https://asrank.caida.org/)
   All these parts are documented in private CAIDA git repositories.

8. BGPStream

BGPStream is an open-source software for live and historical BGP data analysis, supporting scientific research, operational monitoring, and post-event analysis. It provides access to real-time and historical Routviews and RIPE RIS BGP data. BGPStream consists of the following components:

1. BGPReader – the simplest interface to BGPStream: a command-line tool for extracting BGP measurement data in ASCII format.

2. LibBGPStream – the central library of the BGPStream framework. It is written in C and presents a simple API for configuring and reading a stream of BGP measurement data.

3. PyBGPStream – Python package that provides bindings to the libBGPStream library, allowing Python scripts to configure and read a stream of BGP measurement data.

4. Metadata Broker – web application that provides a unified HTTP query interface to retrieve metadata about data available from different data providers.

All documentation can be found at https://bgpstream.caida.org/.

9. Vela

Vela is an on-demand topology measurement service that provides an easy way for researchers to conduct on-demand topology measurements on Ark. Users can conduct ping and traceroute measurements in IPv4 and IPv6 using ICMP, UDP, or TCP from any Ark monitor. There are two interfaces to Vela, a command-line interface and a web-based interface. The command-line interface is useful for conducting large-scale feedback-driven, dynamic measurements under the full control of the user's own program (written in any language of the user's choosing, such as Perl, Python, Ruby, or C). The web interface is useful for interactively performing ad-hoc exploratory measurements.

The web API for conducting Vela traceroute/ping measurements on the Ark infrastructure is documented at https://www.caida.org/projects/ark/vela/web-api/ We currently limit access to the Vela web interface to academic researchers. We hope to provide access to network operators in the future.