DNS-STATS Visualizer Overview and Basic Install

Table of Contents

1. Overview

This document provides an overview of the DNS-STATS Visualizer system and instructions for a basic, default install that users can perform to familiarize themselves with the system and do basic testing. A separate document, the Advanced User Guide provides more in-depth information on day to day operation of Visualizer, details on advanced configuration options and how to customise the installation.

1.1. About

A DNS-STATS Visualizer is a system which can

- consume DNS traffic data files recorded in Compacted-DNS (C-DNS) format from nameservers (such as those generated by DNS-STATS Compactor.)
- populate a ClickHouse database with per query/response level data (and additionally aggregate data at a chosen time interval)
- produce DSC-like statistics graphs of the recorded traffic in Grafana.

As a result, users can either perform ad-hoc queries directly against the database or customize Grafana to create graphs specific to their needs. By default, the Grafana graphs are based on data aggregated to 5 minute intervals so that the resulting graphs are performant for high traffic installations.

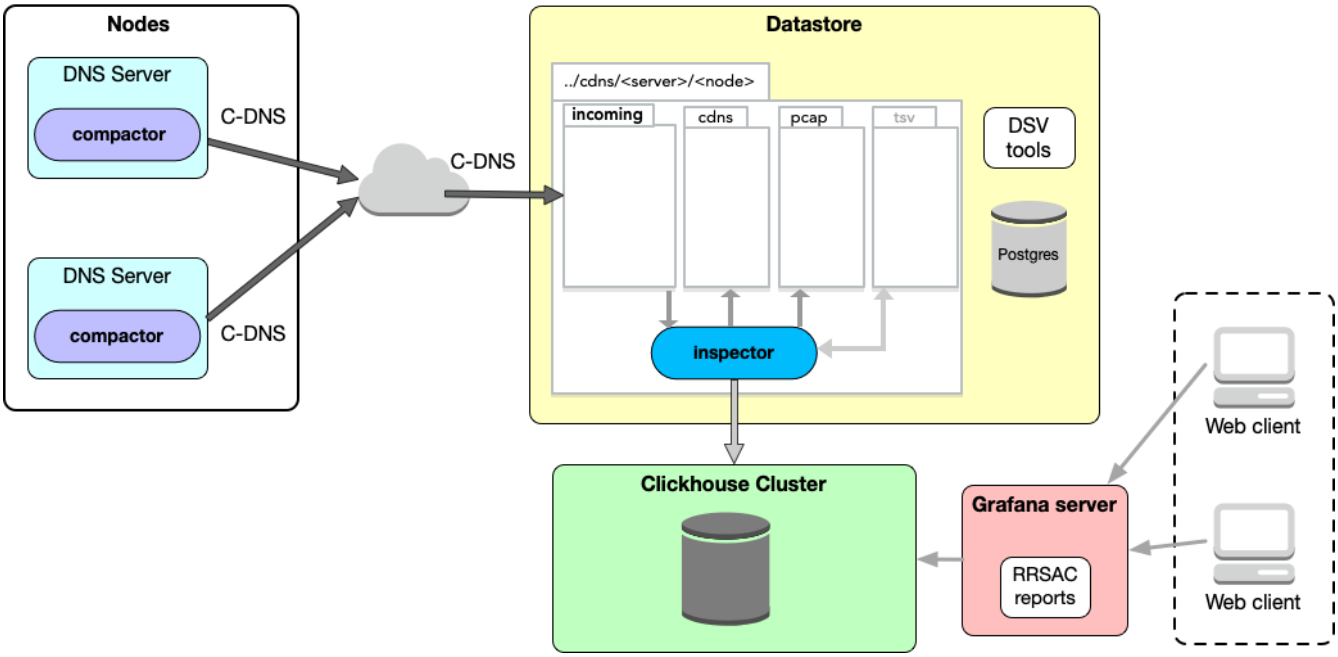

Figure 1. Visualizer system overview

While the Visualizer project can be used to install a complete system with a basic set of graphs and charts, the intention is that the project serves to provide a framework which users can adapt to their own particular requirements. For example:

- Add (and remove) data aggregations and Grafana dashboards to tailor displays to the requirements of a particular organisation.
- Use the raw query/response data in the database for ad-hoc analysis, and omit Grafana and related data aggregations altogether.
- Depending on the volume of data being handled and user requirements, you may choose a different aggregation interval or to dispense with aggregations altogether and have Grafana deal directly with raw data.

The project was initially developed for for ICANN by Sinodun IT, and is now released via DNS-STATS as an open source project licenced under the Mozilla Public License v2.0.

Visualizer is currently tested on Ubuntu 18.04 Bionic Beaver only. Install packages are available from Launchpad: ppa:dns-stats/visualizer

For more information about DNS-STATS and Visualizer see the DNS-STATS website.

Also see the presentation at OARC 33 on the use of a customised version of this system by ICANN IMRS.

1.2. System outline

Visualizer is designed as a multi-host system. Hosts in a Visualizer system fill one of the following basic roles:

- Datastore server. The hosts to which nameservers send their C-DNS files are known as *datastores*. They are responsible for importing the C-DNS data into the main ClickHouse table (via an intermediate tab-separated values (TSV) file). C-DNS files are archived after import.
  - C-DNS can also, optionally, be converted to PCAP files.
- ClickHouse server. A host or a cluster of hosts running a ClickHouse server.
- Grafana server. A host running a Grafana instance providing Visualizer displays (*dashboards*).

Depending on the volume of traffic received, processing can also be split between multiple datastores and across a ClickHouse cluster. Our example install is for 3 separate hosts, with pointers to additional configuration options.

1.2.1. File structure on the Datastore

Nameservers sending files in C-DNS format are known as `nodes` in Visualizer, and their files are grouped logically on disk in subdirectories under `<default_path>/<Server>/<Node>`. Under each node directory are these subdirectories:

- `incoming`. C-DNS files received from the node and awaiting processing. Visualizer does not specify how files are received from nodes and placed in `incoming` or provide any mechanism to do this.
- `cbor`. C-DNS files that have been processed.
- `pcap`. PCAP files that can be optionally re-generated from C-DNS files.
- `error-<queue name>`. Files stored for re-processing where processing has failed.

1.2.2. Queue processing

Each datastore runs an instance of the GearMan server. GearMan is a simple and lightweight queue manager.

The following queues are maintained by Gearman:

- `cdns-to-tsv`. Convert C-DNS file to TSV.
- `cdns-to-pcap`. Convert C-DNS file to PCAP.
- `import-tsv`. Import the TSV file data into the ClickHouse database and then delete the TSV file.

[DNS-STATS Visualizer queue overview] | *dsv-queues.png*

Figure 2. Visualizer queue overview

More details on file and queue handling in Gearman Queues.

1.3. ClickHouse schema

The ClickHouse schema is separated into two main sets of tables

- **Raw tables** held in the `dsv` database. The largest raw table is `QueryResponse` which holds a record for each individual query/response logged. Some additional tables and underlying aggregation data is also stored here.
- **Aggregated data tables** held in the `dsv_five_minute` database. Data here is aggregated on import at 5 minute intervals (by default). This data is used for the Grafana plots to ensure they are performant enough, although it is possible to create plots using the raw data.

1.4. Pre-requisites

Visualizer relies on the following components:

Table 1. Visualizer pre-requisite components

DNS-STATS Inspector	Convert C-DNS files into tab-separated value (TSV) files suitable for import into ClickHouse.
ClickHouse	ClickHouse is a fast, open source, online analytical processing (OLAP) database. It is column-oriented, scalable, and allows SQL querying of large volumes of data in real time.
Grafana	Grafana is an open source analytics platform, allowing you to query, visualize, alert on and understand metrics.
PostgreSQL	Visualizer uses a small PostgreSQL installation in which to keep configuration data that is subject to change over time, such as details of the nodes, classification of top level domains, etc. Both the volume of data and the volume of reads and writes is negligible. Typically a datastore runs the PostgreSQL server; installations with more than one datastore may consider installing PostgreSQL on each datastore, configuring one as a replication master and the rest as replication slaves.
Gearman	GearMan is an open source framework for farming out work to other machines or processes. It is used in Visualizer to manage queues of C-DNS files for conversion and for queueing data for import to ClickHouse.
MaxMind GeoLite2 geolocation data	When converting C-DNS for input to ClickHouse, Visualizer uses GeoLite2 data to add a client location ID to each query/response record. Visualizer also provides a table of location ID and latitude/longitude for use with geographic plots. **NOTE:** *You will need to obtain a free license for GeoLite, as described at https://dev.maxmind.com/geoip/geoip2/geolite2/, and then enter the license key details during installation.*

Grafanalib	Grafana dashboards are designed interactively and saved as JSON files. Grafanalib allows JSON to be produced using Python scripts. Common items across dashboards can be generated in shared code, and the source Python scripts versioned and managed.
Grafana Plotly plugin	A modified version of the standard Grafana Plotly plugin, this Grafana plugin adds Plotly bar charts to the available plots and enables different plot traces to be built from query data.
Python ClickHouse client	A modified version of the Python ClickHouse driver, changed to be pure Python for ease of distribution.

Visualizer commands are implemented in Python, and require Python 3.6 or later. There are also some scripts requiring `bash`.

1.5. Optional modules

Visualizer also includes some optional modules:

- Generate RSSAC reports, as outlined in ICANN document RSSAC002. This includes obtaining and reporting the `load-time` metric but not the `zone-size` metric.
- Mirror incoming files on a datastore to another host.

2. Installing DNS-STATS Visualizer

This section describes how to perform a fresh install of Visualizer onto three new servers using the distributed Visualizer packages.

Table 2. Servers

Type	Hostname	Role
Datastore	`dsv-datastore`	Receive and convert C-DNS files, and import into ClickHouse.
ClickHouse	`dsv-clickhouse`	ClickHouse database host.
Grafana	`dsv-grafana`	Grafana site serving Visualizer dashboards.

The install instructions are intended to take you through to a complete working installation of Visualizer as briefly as possible. The sample configuration files assume the above hostnames are used. The main user manual includes notes on installing and configuring Visualizer to your requirements.

It is assumed that each host has the base OS installed - but no existing ClickHouse or Grafana install. it is also assumed that the names and IP addresses of each host are known to the other hosts in the

install; in other words, that you get a response from the other hosts when attempting a network connection.

```
dsv-datastore $ ping dsv-clickhouse
<response from dsv-clickhouse>
```

Visualizer is currently tested on Ubuntu 18.04 Bionic Beaver only, so all instructions assume that is the target system.

Visualizer requires that all servers are configured to use the UTC timezone (due to dependancies in some of the drivers used).

2.1. Installing the Datastore host

Before starting the datastore host install, you will need to obtain a free licence for MaxMind GeoLite2. You will need the account ID and the licence key. Details of how to obtain the licence are at https://dev.maxmind.com/geoip/geoip2/geolite2/,

1. Add the ClickHouse package repository as described in the ClickHouse installation manual at https://clickhouse.tech/docs/en/getting-started/install/
2. Install PostgreSQL. This can be selected as part of the Ubuntu install process, or:

   ```
   $ sudo apt install postgresql
   ```

3. Install DNS STATS inspector `dns-stats-inspector` and the DNS-STATS Visualizer `dns-stats-visualizer-import` packages.

   ```
   $ sudo add-apt-repository ppa:dns-stats/compactor-bionic
   $ sudo add-apt-repository ppa:dns-stats/visualizer
   $ sudo apt install dns-stats-inspector dns-stats-visualizer-import
   ```

4. Create an OS user `dsv` to own the Visualizer files, and also create the PostgreSQL `dsv` database. A convenient script `dsv-datastore-setup` will do both.

   ```
   $ dsv-datastore-setup
   ```

5. Edit the file `/etc/dns-stats-visualizer/dsv.cfg`.
   a. Add the MaxMind license key in the `[geo]` clause.
   b. If you wish, sensitive settings such as passwords can be specified instead in `/etc/dns-stats-visualizer/private.cfg` and that file made readable only by user `dsv`.
6. PostgreSQL must be configured to permit access for the `dsv` user from the install (Database) host

and any ClickHouse host(s).

a. Enable access for user `dsv` to the PostgreSQL database `dsv`. Edit `/etc/postgresql/10/main/pg_hba.conf` and add, for example, the following (you will probably want to use more restrictive permitted network subnets).

```
# Allow password access by user dsv to database dsv from
# local Unix sockets and the Visualizer cluster.
local   dsv         dsv                             md5
host    dsv         dsv          ::0/0              md5
host    dsv         dsv          0.0.0.0/0          md5
```

The above will need to appear in the file **before** this line:

```
local   all             all                     peer
```

b. Allow PostgreSQL to listen for connections from other hosts (if using separate hosts for PostgreSQL and ClickHouse). Append the following to `/etc/postgresql/10/main/postgresql.conf`:

```
listen_addresses = '*'
```

c. After editing, ensure PostgreSQL restarts with the new configuration.

```
$ sudo systemctl restart postgresql
```

7. Load the initial PostgreSQL tables.

```
$ dsv-postgres-update -v
Applied 1.
```

8. Incoming and processed C-DNS files are stored in a directory structure with a root at a location given in the configuration item `datastore.path`. This directory structure must be owned by user `dsv`. If using the default directory, `/var/lib/dns-stats-visualizer/cdns/`, that directory must have its owner changed to user `dsv`.

```
$ sudo chown -R dsv:dsv /var/lib/dns-stats-visualizer/cdns/
```

9. Complete the configuration for MaxMind GeoLite.

a. Edit the file `/etc/GeoIP.conf` installed by the `geoipupdate` package.

i. Enter your account ID and license key details. (The text in this configuration file may be out of date and indicate that these fields can be left as 0s - this is no longer the case.)

ii. Also replace the last line starting with `EditionIDs` with the following line

```
EditionIDs GeoLite2-City GeoLite2-Country GeoLite2-ASN
```

iii. Entries `UserId` and `ProductId` are obsolete and should be commented out or removed.

b. After that, install the databases by running:

```
$ sudo geoipupdate
```

You may wish to schedule a regular update via `cron` as described in Automatically Scheduled Jobs.

c. Geographic Grafana plots need to convert client location IDs to latitude/longitude. Visualizer uses a PostgreSQL table for this. This table is updated using:

```
$ dsv-geo-update
```

This command can take a short while to run. Again, you may wish to schedule a regular update via `cron` as described in Automatically Scheduled Jobs.

10. Some Visualizer Grafana displays use information on current Top Level Domains (TLDs). This is also held in PostgreSQL tables, The data is downloaded from IANA and the tables updated by `dsv-tld-update`.

```
$ dsv-tld-update
```

This command can take a short while to run. Again, you may wish to schedule a regular update via `cron` as described in Automatically Scheduled Jobs.

11. Set up default `supervisord` controlled instances of the `dsv-worker` process to process incoming files. The sample configuration runs 5 instances of `dsv-worker`. Adjust the number of instances to suit your anticipated workload and system resources.

```
$ sudo apt install supervisor
$ sudo cp /etc/supervisor/conf.d/dsv.conf.sample /etc/supervisor/conf.d/dsv.conf
$ sudo supervisorctl reload
```

12. Once a ClickHouse host is also installed, the import of C-DNS files can be tested. See Section 2.3, “Testing the import process”.

2.2. Installing the ClickHouse host

1. Add the ClickHouse package repository as described in the ClickHouse installation manual at

https://clickhouse.tech/docs/en/getting-started/install/.

2. Install the DNS-STATS Visualizer `dns-stats-visualizer-clickhouse-server` package.

```
$ sudo add-apt-repository ppa:dns-stats/visualizer
$ sudo apt install dns-stats-visualizer-clickhouse-server
```

3. The package installs a total of 5 sample ClickHouse configurations in `*.xml.dsv` files in `/etc/clickhouse-server/config.d` and `/etc/clickhouse-server/users.d`. For a default install, only one file needs updating:
   a. `/etc/clickhouse-server/users.d/users.xml.dsv`. Update the settings for the default and `dsv` users if required.
4. Then rename all five files to `*.xml` so they will override the default ClickHouse configuration, for example

```
$ sudo apt install rename
$ sudo su root
$ rename 's/.xml.dsv/.xml/' /etc/clickhouse-server/config.d/*.xml.dsv
$ rename 's/.xml.dsv/.xml/' /etc/clickhouse-server/users.d/*.xml.dsv
```

5. Edit the settings in `/etc/dns-stats-visualizer/dsv.cfg` to match those used in the same file on your Datastore host.
6. ClickHouse uses ODBC to communicate with PostgreSQL on the datastore. Copy `/etc/odbc.ini.dsv` to `/etc/odbc.ini`.

```
$ sudo cp /etc/odbc.ini.dsv /etc/odbc.ini
```

   Test the setup:

```
$ isql dsv
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------------------------------+
SQL>
```

7. Then restart ClickHouse with the new configuration.

```
$ sudo systemctl restart clickhouse-server
```

8. Load the initial ClickHouse tables.

```
$ dsv-clickhouse-update -v
Applied 1.
Applied 2.
Applied 10.
Applied 11.
```

2.3. Testing the import process

2.3.1. On the Datastore host

1. To test the import process, add a real or test node to the database. Make a copy of the file `/etc/dns-stats-visualizer/nodes.csv.sample` called `nodes.csv`

   ```
   $ sudo cp /etc/dns-stats-visualizer/nodes.csv.sample /etc/dns-stats-
   visualizer/nodes.csv
   ```

   and edit it to add a server e.g. uncomment the last line

   ```
   TestServer,TestNode,TestRegion,TestCountry,TestCity,TestInstance
   ```

2. Import nodes into the database

   ```
   $ dsv-nodes-update -c /etc/dns-stats-visualizer/dsv.cfg /etc/dns-stats-
   visualizer/nodes.csv
   ```

3. A test C-DNS file is installed by the package in `/usr/share/dns-stats-visualizer/sampledata/testnode.cdns.xz`. Copy this as user `dsv` into a node incoming directory. For example:

   ```
   $ sudo -u dsv mkdir -p /var/lib/dns-stats-
   visualizer/cdns/TestServer/TestNode/incoming
   $ sudo -u dsv cp /usr/share/dns-stats-visualizer/sampledata/testnode.cdns.xz
   /var/lib/dns-stats-visualizer/cdns/TestServer/TestNode/incoming
   ```

   You can check the queue status and should see one file in `CDNS incom`

   ```
   $ dsv-queue-details -p
   ```

4. Run the `dsv-import` command directly to process the waiting file.

```
$ sudo -u dsv dsv-import -s incoming -v
```

which will report adding the file to the cdns-to-tsv processing queue.

5. Re-check the queue status and after a few seconds the files should be gone and no errors reported (you may transiently see a file in the `TSV pend` queue)

```
$ dsv-queue-details -p
```

You can also see the status of the processing queues using

```
$ dsv-status
```

You can also check the logs to see the result of the import process:

```
$ sudo tail /var/log/syslog
```

2.3.2. On the Clickhouse host

1. Run a query against the database (note that the default ClickHouse prompt is `dsv-clickhouse :)`)

```
$ clickhouse-client -d dsv

dsv-clickhouse :) show tables;
SHOW TABLES

Query id: 8176d410-0fb9-4a4e-b0da-4aaae10e2f47

 ┌─name──────────────────────────────────────────────────────────────────────────────────────────┐
 │ AAATopUndelegatedTldPerFiveMins         │
 │ AAATopUndelegatedTldPerFiveMinsShard    │
 │ AAATopUndelegatedTldPerFiveMinsShardMV  │
 │ ImportQueueSizes                        │
 │ ImportQueueSizesShard                   │
 │ PacketCounts                            │
 │ PacketCountsShard                       │
 │ QueryResponse                           │
 │ QueryResponseShard                      │
 │ ZoneLatency                             │
 │ ZoneLatencyShard                        │
 │ ddl_history                             │
 │ geolocation                             │
 │ iana_text                               │
 │ node_text                               │
 │ server_address                          │
 │ tld_text                                │
 └─────────────────────────────────────────────────────────────────────────────────────────────────
┘

17 rows in set. Elapsed: 0.002 sec.

dsv-clickhouse :) SELECT count() FROM QueryResponse;

SELECT count()
FROM QueryResponse

Query id: 4790c017-8c1e-40e8-864f-c35c88bd7c55

 ┌─count()─┐
 │     999 │
 └─────────┘

1 rows in set. Elapsed: 0.004 sec.
```

2.4. Setting up a running import system

1. After manual testing that the process is working correctly, you should:
    a. add your specific nodes to the database. See the configuring section of the user guide.

b. upload your data files from your nodes into the datastore directory structure.

c. run the `dsv-import` process periodically from `cron`. See the configuring section of the user guide.

d. and similarly, periodically log details on the status of the work queues to ClickHouse for monitoring with `dsv-queue-details`.

More details on the general operation and administration of the system can be found in the first two sections of the Advanced User Guide.

2.5. Installing Grafana hosts

The standard means of displaying the data collected by Visualizer is to use Grafana with a set of dashboards showing Visualizer plots and other information. Visualizer includes a basic set of Grafana dashboards.

The Visualizer framework chooses to use the file-based provisioning system for Grafana (as opposed to the GUI or the HTTP API which are much less easily automated for our use case). However, this does require some manual set up to be done, specifically for user access.

Testing has only been done on recent versions of Grafana installed using the process below.

1. Use the Grafana Debian/Ubuntu package repository as described at https://grafana.com/docs/grafana/latest/installation/debian/.
2. Start Grafana and configure it to start at boot following the instructions on https://grafana.com/docs/grafana/latest/installation/debian/.
3. Install the `dns-stats-visualizer-grafana-main` package. This will install all the dashboards, data sources and the required plugins.

```
$ sudo add-apt-repository ppa:dns-stats/visualizer
$ sudo apt install dns-stats-visualizer-grafana-main
```

4. Verify the correct plugins have been installed. The versions are unimportant.

```
$ sudo grafana-cli plugins ls
installed plugins:
grafana-image-renderer @ 2.0.0
grafana-worldmap-panel @ 0.3.2
natel-plotly-panel @ 0.0.7-dev
vertamedia-clickhouse-datasource @ 2.2.0
```

5. The package installs a sample Grafana datasource provisioning file at `/etc/grafana/provisioning/datasources/dsv-main.yml.sample`. Copy this to `dsv-main.yml` in the same directory:

```
$ sudo cp /etc/grafana/provisioning/datasources/dsv-main.yml.sample
/etc/grafana/provisioning/datasources/dsv-main.yml
```

You do not need to edit this file for a default install but you may want to review the security settings.

6. After editing, restart Grafana with the new configuration.

```
$ sudo systemctl restart grafana-server
```

7. Log into Grafana via the web interface as administrator by pointing a web browser at http://dsv-grafana:3000 and logging in as user `admin` password `admin`. You may want to change the admin password and otherwise configure authentication at this time.
8. Click on the `Dashboards\Manage` option in the left hand sidebar, then click on the `General→ DNS-STATS Visualizer main menu` item in the list of dashboards.
   a. Mark this dashboard as a favourite by clicking on the star icon to the right of the dashboard title. The icon should turn orange.
9. Click on the `Configuration\Preferences` icon menu in the left hand sidebar. Under the `Home Dashboard` heading select `DNS-STATS Visualizer main menu` from the drop down.

   Save the preferences.

10. Check you can see the test data that was imported by choosing the `Query Statistics` graph from the main dashboard and using the time picker in the top right to set the window to start at 2016-06-29:15:45 and end at 2016-06-29:16:00. You will see a single data point at 15:50.
11. If you want users to be able to view Grafana dashboards without logging in to Grafana, you will need to allow anonymous access to Grafana . You will also need to allow anonymous access if you want to produce RSSAC reports.

    To allow anonymous access, edit `/etc/grafana/grafana.ini` and in section `[auth.anonymous]`, set `enabled = true`. Then restart Grafana:

```
$ sudo systemctl restart grafana-server
```