Purpose

This document describes how to use the full flexibility of Nagios Network Analyzer to get the most out of your network flow data.

Target Audience

Network admins performing forensic analysis on a network's flow data to drill directly to the information they need.

Terminology

The following terms will be used throughout this document:

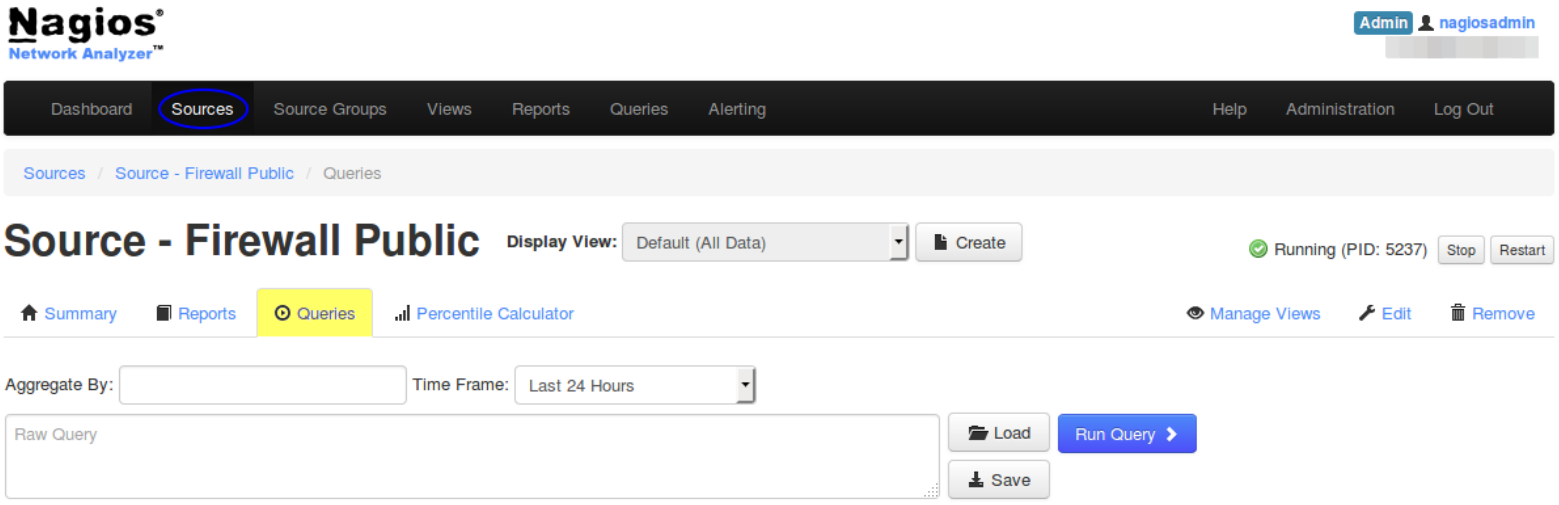
- `src` - Source
- `dst` - Destination
- `srcip` - Source IP is the IP Address the traffic originated from
- `dstip` - Destination IP is the IP Address the traffic is going to
- `srcport` - Source Port is the network port the traffic is transmitted on
- `dstport` - Destination Port is the network port the traffic is received on

Introduction

This documentation will show you how to use Nagios Network Analyzer to turn existing flow data into meaningful information. This manipulation will not destroy your data at all, so feel free to experiment, as there is no chance at all that you will break anything. You will need to have an existing source with flow data to be able to follow the examples in this documentation.

Performing A Query

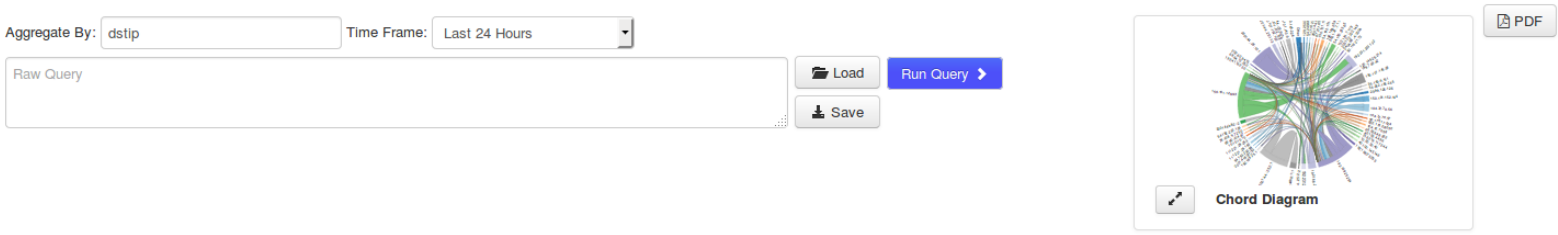
Click **Sources** on the top menu and then click one of your sources. Click the **Queries** tab to bring up the query options.



This is where we will be doing the majority of work and explanation in this document, and will most likely be the entry point for any deep-diving you do into the flow data. On this page, you'll see many fields. This section will give a description what each one is for, and how to use it.

Aggregate By - This is how the flows will be associated with each other. This field should be a comma delimited list of aggregate values such as `dstip`, `srcip`, `dstport` and `srcport`. When the flows get aggregated, it groups all like values for that aggregate value together. For instance, if we simply specify `dstip` for our value, all unique values of `dstip` will be grouped together.

Try it out, type `dstip` in the **Aggregate By** field, leave the "Raw Query" field blank and click the **Run Query** button. The screenshot on the following page is an example of what you might get.



Query: Custom Query

External API | Use Via HTTP

Showing **last 24 hours** for query "" aggregated by **dstip**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 138

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 16:38:46.159	2017-06-07 17:47:40.497	4134.338	*	54.230.245.75	*	*	12	1.75 K	2	3	0	149
2017-06-07 11:12:06.321	2017-06-08 09:07:21.500	78915.179	*	91.189.95.83	*	*	148	20.36 K	16	2	0	140
2017-06-08 02:42:56.295	2017-06-08 03:52:50.953	4194.658	*	180.163.19.11	*	*	3	210	3	0	0	70
2017-06-07 20:18:18.068	2017-06-07 21:28:12.282	4194.214	*	168.63.152.107	*	*	40	1.64 K	2	3	0	42

You can see that a Chord Diagram is generated, you can click the icon on the diagram to enlarge it. Hovering the mouse on an address in the diagram will highlight the relationship with the other addresses.

Underneath is the detailed table of the results from the query. Notice that all the IPs listed are unique. This is because as our query looked through the flow data, it grouped all of the `dstip` that were the same data, and treated them as one entity, and simply computed a running sum for all unique destination IP's metrics.

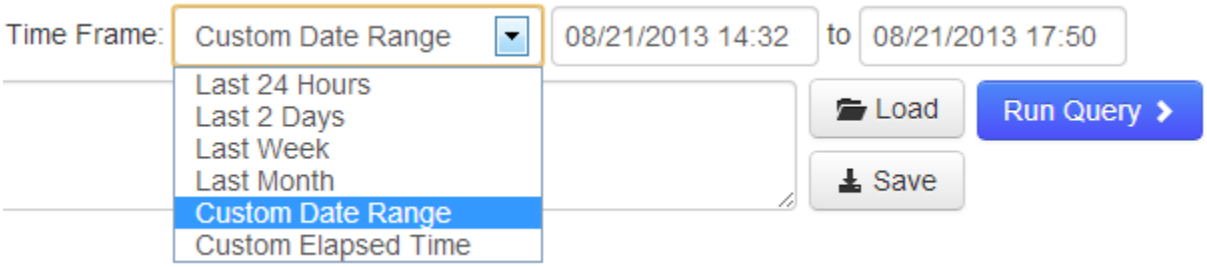
You can increase the granularity here by adding `srcip` to the "Aggregate By" field. Try this by changing the **Aggregate By** field to `dstip,srcip` and click the **Run Query** button. This will now treat `dstip` and `srcip` as unique entries. Two connections from IP A to IP B will be summed and represented as one, however IP B to IP A will be not be in that same category.

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 16:36:51.196	2017-06-07 17:46:45.318	4194.122	192.168.25.254	150.100.6.8	*	*	1	69	1	0	0	69
2017-06-07 16:26:26.252	2017-06-07 20:54:03.795	16057.543	150.101.126.93	192.0.76.3	*	*	68	7.33 K	5	3	0	110
2017-06-08 06:10:09.523	2017-06-08 07:20:01.049	4191.526	74.108.117.43	150.101.126.93	*	*	5	270	1	0	0	54
2017-06-07 15:31:09.100	2017-06-07 16:41:03.164	4194.064	150.101.126.93	66.45.125.172	*	*	2	165	2	0	0	82

The more aggregate values you have, the more unique values show up, so the queries will generally take longer to run the more aggregate values you have. You are not limited to only aggregating by similar values, for example you could have a query like `dstip,dstport` and get results like the following screenshot.

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 15:37:34.545	2017-06-07 16:47:29.035	4194.490	::	2001:44...:25:2:1	*	23878	2	290	2	0	0	145
2017-06-08 08:58:22.272	2017-06-08 10:08:16.916	4194.644	*	188.138.94.119	*	53	4	296	4	0	0	74
2017-06-07 12:15:47.844	2017-06-07 13:25:41.567	4193.723	*	150.101.126.93	*	14622	2	84	1	0	0	42
2017-06-07 16:34:05.279	2017-06-07 17:43:59.544	4194.265	*	192.168.25.254	*	63104	1	791	1	1	0	791

Time Frame - This is where you set the time frame for the query. This section is largely self-explanatory, but you can set either hard date times to search between, or you can set soft date times. Hard date times would be exact times, like from 1:00PM on January 1st until 2:00PM on January 1st. You can also set elapsed time frames, to specify something like 3 hours ago until now.



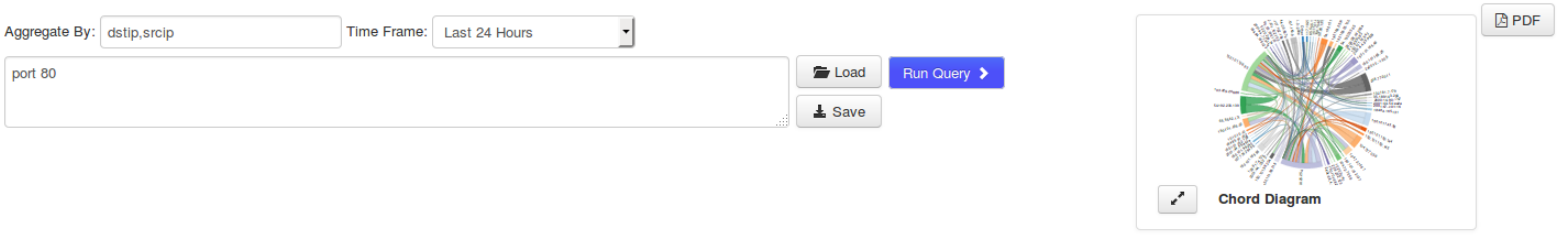
Raw Query - This field is the most powerful tool when querying data. In this field you will enter a query string to sort through the data, and if you have ever used tcpdump before, this section will be familiar.

In this query string you can specify quite a few parameters to limit what is shown to you and chain parameters together to isolate exactly what you'd like to see.

Let us assume that we would only like to see traffic on port 80. It doesn't matter if its coming from port 80, or if its going to port 80. In our query box you would type:

```
port 80
```

Here is an example of what that looks like:



Query: Custom Query

External API | Use Via HTTP

Showing **last 24 hours** for query "**port 80**" aggregated by **dstip,srcip**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 62

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 16:26:30.281	2017-06-07 17:35:24.529	4134.248	150.101.126.93	192.0.76.3	*	*	8	1.53 K	1	3	0	196
2017-06-07 16:37:30.050	2017-06-07 17:46:23.447	4133.397	150.101.126.93	35.187.205.99	*	*	9	1.63 K	1	3	0	185
2017-06-07 20:17:06.630	2017-06-07 21:25:38.487	4111.857	150.101.126.93	111.221.29.193	*	*	16	2.04 K	2	4	0	130
2017-06-07 16:37:58.566	2017-06-07 17:47:53.206	4194.640	208.81.233.32	150.101.126.93	*	*	13	1.75 K	2	3	0	138

This shows all `dstip,srcip` aggregates that are talking on port 80. Now change the **Aggregate By** field to `dstip,srcip,dstport,srcport` and click the Run Query button, you will get results like the following.

Showing **last 24 hours** for query "**port 80**" aggregated by **dstip,srcip,dstport,srcport**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 918

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-08 07:10:28.569	2017-06-08 07:10:28.569	0.000	185.110.132.239	150.101.126.93	46713	80	1	40	1	0	0	40
2017-06-07 19:02:12.505	2017-06-07 20:11:36.240	4163.735	54.192.233.139	150.101.126.93	80	65376	4	664	1	1	0	166
2017-06-08 07:30:47.156	2017-06-08 08:40:11.045	4163.889	192.168.25.254	150.101.165.33	31993	80	6	649	1	1	0	108
2017-06-07 14:30:58.110	2017-06-07 15:40:20.848	4162.738	150.101.126.93	54.192.233.139	17825	80	31	1.89 K	1	3	0	62

Notice how many pages of entries this returns, 918 pages times 20 entries per page gives us 18,360 entries! If you scroll through them you'll notice they all have port 80 as one of their ports, this is because you are seeing traffic in both directions. If you only want to see when the source port is 80 amend the query to:

```
src port 80
```

Now the query will be limited to the source port:

Showing **last 24 hours** for query "**src port 80**" aggregated by **dstip,srcip,dstport,srcport**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 349

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 19:02:12.505	2017-06-07 20:11:36.240	4163.735	54.192.233.139	150.101.126.93	80	65376	4	664	1	1	0	166
2017-06-07 11:53:00.619	2017-06-07 13:02:23.933	4163.314	150.101.165.48	192.168.25.254	80	65088	5	1.70 K	1	3	0	347
2017-06-07 13:35:40.762	2017-06-07 14:45:04.434	4163.672	150.101.165.33	150.101.126.93	80	20888	5	1.81 K	1	3	0	370
2017-06-08 06:21:46.539	2017-06-08 07:29:59.870	4093.331	150.101.152.152	150.101.126.93	80	50005	6	1.59 K	1	3	0	271

You can also click any of the hyperlinks in the table of data to drill down further into the query. This will populate the Raw Query field with a new query based on what you clicked on. In the screenshot above this is the **Source IP**, **Destination IP**, **Source Port** and **Destination Port** columns.

Save Query

You can save a query to use again at a later stage. Click the **Save** button to the right of the Raw Query window.

You will need to provide a **Name** and **Description**.

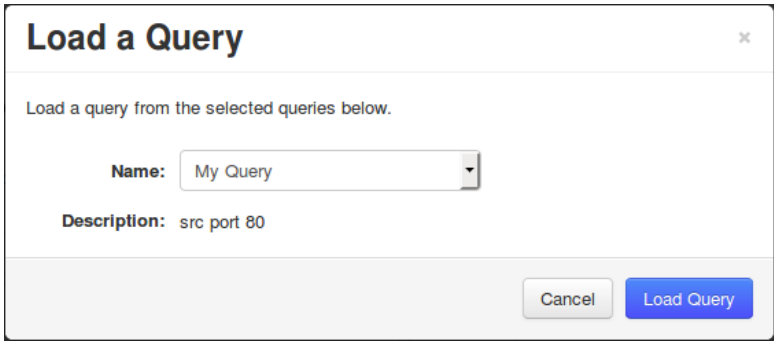
Click the **Save Query** button to save.

Load Query

To load a query that you previously saved click the **Load** button to the right of the Raw Query window.

Select a query from the Name list.

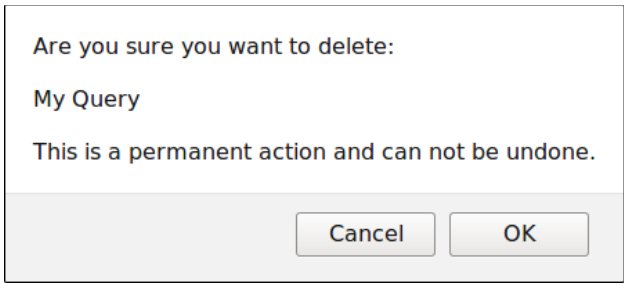
Click the **Load Query** button to load it.



Delete Query

If you want to delete an existing query, click the **Delete** icon to the right of the **Loaded Query** field.

You will need to click **OK** on the window that appears to delete the query.



Advanced Queries

So far you've seen some basic queries. The following sections explain how you can make a query more specific depending on what information you are after.

IP / Network

A raw query can use an IP address or a network scope. Here is an example of using an IP address:

```
ip 10.25.2.1
```

Showing **last 24 hours** for query "**ip 10.25.2.1**" aggregated by **srcip,dstip**.

First « 1 2 3 4 5 6 7 8 9 » Last Of 9

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 13:29:02.297	2017-06-08 13:04:46.763	84944.466	10.25.2.1	255.255.255.255	*	*	535	175.55 K	433	16	0	336
2017-06-07 16:02:14.225	2017-06-07 20:09:04.464	14810.239	64.208.140.18	10.25.2.1	*	*	4	424	4	0	0	106
2017-06-08 08:45:04.953	2017-06-08 09:54:51.990	4187.037	211.140.14.36	10.25.2.1	*	*	8	880	2	1	0	110
2017-06-08 08:50:34.747	2017-06-08 10:18:37.092	5282.345	10.25.2.1	106.75.128.65	*	*	10	830	10	1	0	83

Here is an example of using a network scope by using the slash notation:

```
net 10.25.0.0/16
```

Showing **last 24 hours** for query "**net 10.25.0.0/16**" aggregated by **srcip,dstip**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 113

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 19:49:46.667	2017-06-08 07:31:33.781	42107.114	10.25.2.2	13.107.4.1	*	*	32	2.49 K	32	0	0	79
2017-06-08 08:58:43.749	2017-06-08 10:08:38.598	4194.849	212.47.0.10	10.25.2.2	*	*	4	618	4	1	0	154
2017-06-08 08:58:22.463	2017-06-08 10:08:16.400	4193.937	10.25.2.2	188.138.94.119	*	*	4	296	4	0	0	74
2017-06-08 00:13:07.293	2017-06-08 01:22:54.374	4187.081	10.25.254.4	191.239.50.77	*	*	14	1.36 K	2	2	0	99

In the screenshots above you can see that the IP address or the network scope being queried appears in either the **Source IP** or **Destination IP** columns.

Defining Source Or Destination

Queries can be prepended by using `src` or `dst` to target a specific traffic direction. Here is a net example:

```
src net 10.25.0.0/16
```

Showing **last 24 hours** for query "**src net 10.25.0.0/16**" aggregated by **srcip,dstip**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 64

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 19:49:46.667	2017-06-08 07:31:33.781	42107.114	10.25.2.2	13.107.4.1	*	*	32	2.49 K	32	0	0	79
2017-06-08 08:58:22.463	2017-06-08 10:08:16.400	4193.937	10.25.2.2	188.138.94.119	*	*	4	296	4	0	0	74
2017-06-08 00:13:07.293	2017-06-08 01:22:54.374	4187.081	10.25.254.4	191.239.50.77	*	*	14	1.36 K	2	2	0	99
2017-06-08 06:34:25.517	2017-06-08 07:44:19.773	4194.256	10.25.2.2	13.107.5.1	*	*	4	304	4	0	0	76

You can see in the screenshot above that all the `10.25.0.0/16` addresses are in the Source IP column.

Here is a port example:

```
dst port 80
```

Showing **last 24 hours** for query "**dst port 80**" aggregated by **srcip,dstip,srcport,dstport**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 455

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-08 07:30:46.823	2017-06-08 08:40:10.712	4163.889	192.168.25.254	150.101.165.33	31993	80	12	1.27 K	2	2	0	108
2017-06-08 01:49:03.264	2017-06-08 02:58:55.784	4192.520	192.168.25.254	17.253.67.205	40995	80	12	1.21 K	2	2	0	103
2017-06-07 15:21:06.662	2017-06-07 16:30:30.284	4163.622	192.168.25.254	150.101.152.153	16360	80	12	1.36 K	2	2	0	116
2017-06-07 19:22:26.587	2017-06-07 20:31:48.981	4162.394	192.168.25.254	150.101.143.38	17377	80	12	1.26 K	2	2	0	107

You can see in the screenshot above that `port 80` is only in the Destination Port column.

Logic Operator: AND

Using the AND operator can allow you to have more granular queries, for example:

```
src ip 10.25.254.50 AND dst port 80
```

Showing **last 24 hours** for query "**src ip 10.25.254.50 AND dst port 80**" aggregated by **srcip,dstip**.

First « 1 » Last Of 1

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 13:49:22.786	2017-06-08 13:07:21.731	83878.945	10.25.254.50	91.189.95.83	*	*	168	23.20 K	18	2	0	141
2017-06-07 13:49:24.877	2017-06-08 12:45:10.006	82545.129	10.25.254.50	150.101.98.240	*	*	272	18.88 K	20	1	0	71
2017-06-08 09:46:58.379	2017-06-08 10:56:48.065	4189.686	10.25.254.50	150.101.98.201	*	*	12	1.06 K	2	2	0	90
2017-06-07 19:08:56.370	2017-06-07 20:18:46.257	4189.887	10.25.254.50	150.101.143.24	*	*	12	1.06 K	2	2	0	90

Logic Operator: OR

Using the OR operator can allow you to have more flexible queries, for example:

```
src ip 10.25.254.50 OR dst ip 10.25.2.1
```

Showing **last 24 hours** for query "**src ip 10.25.254.50 OR dst ip 10.25.2.1**" aggregated by **srcip,dstip**.

First « 1 2 3 4 5 6 7 8 9 » Last Of 9

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 16:51:27.295	2017-06-08 06:56:18.164	50690.869	10.25.254.50	224.0.0.22	*	*	16	736	8	0	0	46
2017-06-08 10:27:41.841	2017-06-08 11:36:34.000	4132.159	10.25.254.50	34.198.58.82	*	*	72	7.33 K	4	14	0	104
2017-06-07 16:02:14.225	2017-06-07 20:09:04.464	14810.239	64.208.140.18	10.25.2.1	*	*	4	424	4	0	0	106
2017-06-08 08:45:04.953	2017-06-08 09:54:51.990	4187.037	211.140.14.36	10.25.2.1	*	*	8	880	2	1	0	110

Logic Operator: NOT

Using the NOT operator can allow you to have queries that exclude data, for example:

```
NOT dst port 53
```

Showing **last 24 hours** for query "**NOT dst port 53**" aggregated by **srcip,dstip,srcport,dstport**.

First « 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 » Last Of 5332

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-07 14:55:48.460	2017-06-07 16:05:42.263	4193.803	192.168.25.254	54.208.199.64	14466	8200	8	344	2	0	0	43
2017-06-08 00:49:09.399	2017-06-08 01:59:03.607	4194.208	fe80::6..2d:1800	ff02::1:3	55051	5355	4	284	2	0	0	71
2017-06-07 18:59:29.596	2017-06-07 20:09:23.708	4194.112	134.170.108.48	192.168.25.254	53	12864	2	372	2	0	0	186
2017-06-07 16:17:27.846	2017-06-07 17:27:22.044	4194.198	10.25.5.86	203.213.88.59	39663	123	2	152	2	0	0	76

Metrics

You can create queries on the amount of traffic that went through for each flow.

```
dst port 80 AND bytes > 1m
```

Showing **last month** for query "**dst port 80 AND bytes > 1m**" aggregated by **srcip,dstip,dstport**.

First « 1 2 3 » Last Of 3

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-05-22 02:53:34.294	2017-05-29 19:59:32.161	666357.867	2001:44..1a:45d5	2001:44..65:8f21	*	80	265.86 K	17.04 M	10	214	0	65
2017-05-19 11:53:13.488	2017-05-30 10:28:09.858	945296.370	2001:44..1a:45d5	2001:44..65:c3d3	*	80	312.43 K	18.58 M	12	164	0	60
2017-06-06 07:17:53.300	2017-06-06 08:30:13.710	4340.410	2001:44..5:14:91	2a01:11..003::50	*	80	250.40 K	15.54 M	6	29.33 K	59	63
2017-05-26 07:54:35.650	2017-05-26 08:02:15.409	459.759	2001:44..5:5:169	2405:50..f0a::20	*	80	391.98 K	29.51 M	14	525.87 K	873	77

You are not limited to < and > operators either, you can use = as well. The query above was for `bytes`, but you can also use `packets` and `flows`.

Using Parenthesis To Group Expressions

You can add parenthesis to your expression to make it clear how the query will be executed, this allows for more complex queries. Here is a simple example:

```
src ip 2001:44b8:3132:25:10:25:254:50 AND (dst port 80 OR dst port 443)
```

Showing **last 24 hours** for query "**src ip 2001:44b8:3132:25:10:25:254:50 AND (dst port 80 OR dst port 443)**" aggregated by **srcip,dstip,dstport**.

First « 1 2 3 » Last Of 3

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-06-08 09:10:55.788	2017-06-08 10:20:44.970	4189.182	2001:44..:254:50	2404:68..3::200e	*	443	24	2.32 K	2	4	0	99
2017-06-08 10:33:56.403	2017-06-08 13:27:57.383	10440.980	2001:44..:254:50	2404:68..c04::bd	*	443	32	4.62 K	4	3	0	147
2017-06-07 15:49:25.657	2017-06-08 10:56:47.777	68842.120	2001:44..:254:50	2404:68..4::200e	*	80	36	4.02 K	4	0	0	114
2017-06-08 08:53:38.010	2017-06-08 13:44:12.710	17434.700	2001:44..:254:50	2404:68..2::2003	*	443	340	44.11 K	34	20	0	132

You can see that the example provided results for port `80` OR `443`. This was also an example to demonstrate that IPv6 addresses can also be queried. Here is a more complex example:

```
(src ip 10.25.254.50 OR src ip 10.25.14.10 OR src net
2001:44b8:3132:25:0:0:0:0/64) AND (dst port 80 OR dst port 443) AND NOT src ip
2001:44b8:3132:25:10:25:14:52 AND bytes > 10m
```

Showing **last month** for query "**(src ip 10.25.254.50 OR src ip 10.25.14.10 OR src net 2001:44b8:3132:25:0:0:0:0/64) AND (dst port 80 OR dst port 443) AND NOT src ip 2001:44b8:3132:25:10:25:14:52 AND bytes > 10m**" aggregated by **srcip,dstip,dstport**.

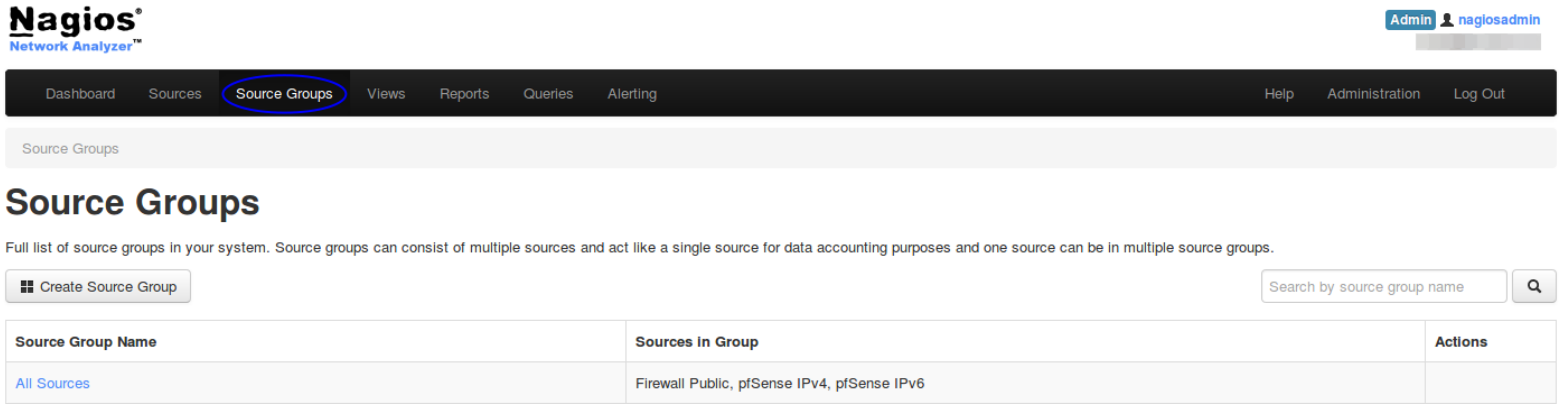
First « 1 » Last Of 1

Flow Start	Flow End	Duration	Source IP	Destination IP	Source Port	Destination Port	Packets	Bytes	Flows	Bytes/Sec	Packets/Sec	Bytes/Packet
2017-05-27 19:31:26.682	2017-05-27 19:37:48.798	382.116	2001:44..25:11:3	2404:68..7::200a	*	443	44.69 K	61.14 M	4	1.28 M	119	1.37 K

While a lot more complicated, you can see only one result was returned which can be very useful when interrogating flow data. The first parenthesis targeted two IP addresses or an entire IPv6 subnet (using multiple ORs). The second parenthesis allowed port `80` OR `443`. Then two more conditions were defined.

Source Groups

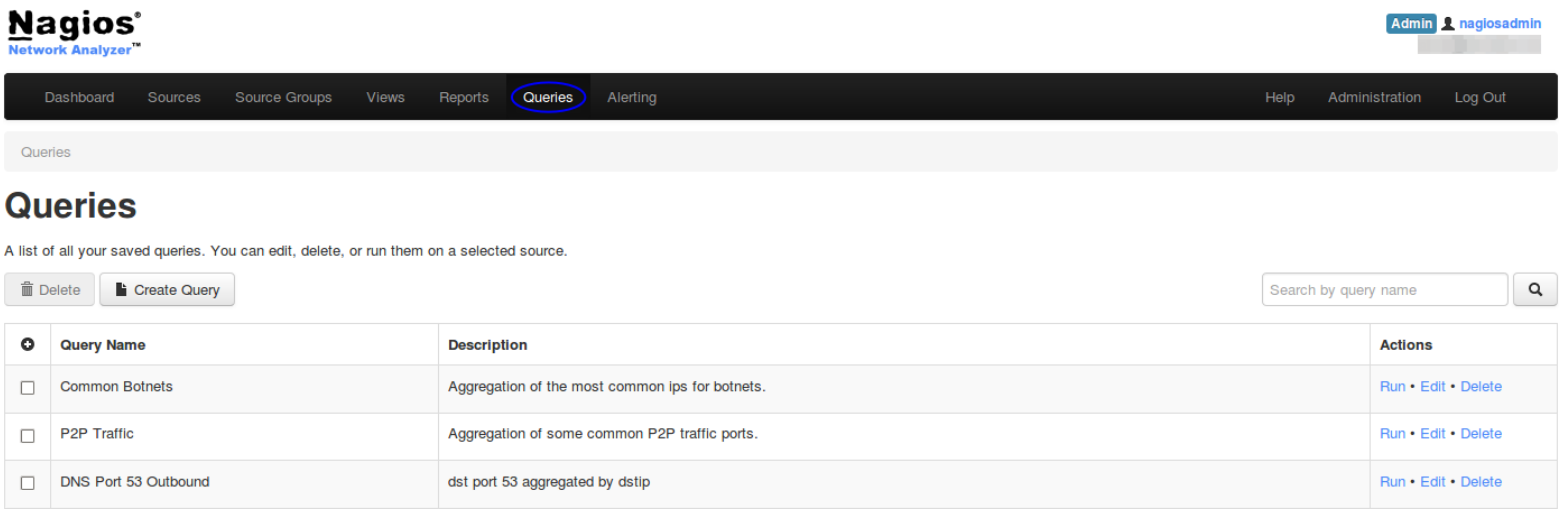
Queries can also be performed on Source Groups via the **Source Groups** menu on the navigation bar.



Click the desired Source Group and then click the Queries tab. The functionality is the same as for Sources.

Managing Queries

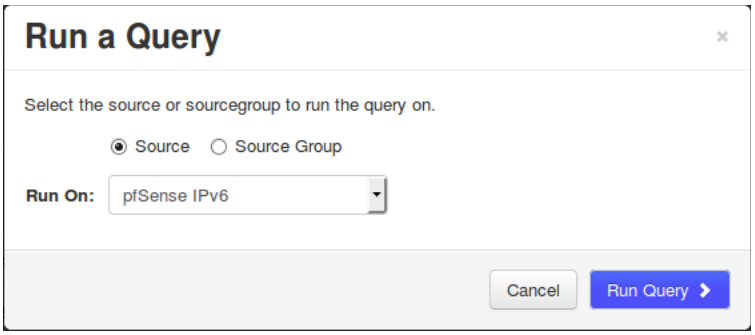
Queries can be managed via the **Queries** menu on the navigation bar.



You can delete multiple queries by checking the boxes in the left column and then clicking the **Delete** button.

In the Actions column you can **Run**, **Edit** and **Delete** a query.

When clicking **Run** you are prompted to select a Source or Source group that you want to execute the query against. Once you click the **Run Query** button you will be taken to the Source or Source Group page with the results of the query just executed.



Further Reading

This documentation covered many of the features available in queries however it did not comprehensively cover all abilities of the query syntax. If you would like to read more please refer to the following link:
http://nfdump.sourceforge.net/

Finishing Up

This completes the documentation on understanding and using custom queries in Nagios Network Analyzer. If you have additional questions or other support related questions, please visit us at our Nagios Support Forums:

https://support.nagios.com/forum

The Nagios Support Knowledgebase is also a great support resource:

https://support.nagios.com/kb