Network Performance Monitoring

Throughout this lab, each section will be broken down into a series of steps. To navigate between sections, click each header to expand or collapse the sections.

Make sure you are logged into Datadog using the Datadog training account credentials provisioned for you. You can find that information by running `creds` in the lab terminal.

Enabling NPM

The Datadog Agent container has already been configured for NPM for this lab. This was done by adding environment variables to Agent container in `docker-compose.yml`:

- Under `environment`, `DD_SYSTEM_PROBE_ENABLED=true`
- Under `volumes`, `/sys/kernel/debug:/sys/kernel/debug`
- `cap_add` section to allow the container to access host resources
- `security_opt` section to avoid AppArmor restrictions

```yaml
version: '3'
services:
  datadog:
    image: 'datadog/agent:7.31.1'
    environment:
      - DD_API_KEY
      - DD_HOSTNAME=dd101-sre-host
      - DD_LOGS_ENABLED=true
      - DD_LOGS_CONFIG_CONTAINER_COLLECT_ALL=true
      - DD_PROCESS_AGENT_ENABLED=true
      - DD_SYSTEM_PROBE_ENABLED=true
      - DD_DOCKER_LABELS_AS_TAGS={"my.custom.label.team":"team"}
      - DD_TAGS='env:dd101-sre'
      - DD_APM_NON_LOCAL_TRAFFIC=true
    ports:
      - 127.0.0.1:8126:8126/tcp
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
      - /proc/:/host/proc/:ro
      - /sys/fs/cgroup/:/host/sys/fs/cgroup:ro
      - /sys/kernel/debug/:/sys/kernel/debug
    cap_add:
      - SYS_ADMIN
      - SYS_RESOURCE
      - SYS_PTRACE
      - NET_ADMIN
      - NET_BROADCAST
      - NET_RAW
      - IPC_LOCK
      - CHOWN
    security_opt:
      - apparmor:unconfined
```

The `cap_add` and `security_opt` sections are required because NPM relies on eBPF, a high-performance, kernel-level interface to a Linux system's data link layer that requires heightened privileges to access.

Configuring NPM on a host, Kubernetes, or ECS is similar, and covered is in the NPM Installation documentation.

In the next step, you'll look at Storedog's network data in Datadog.

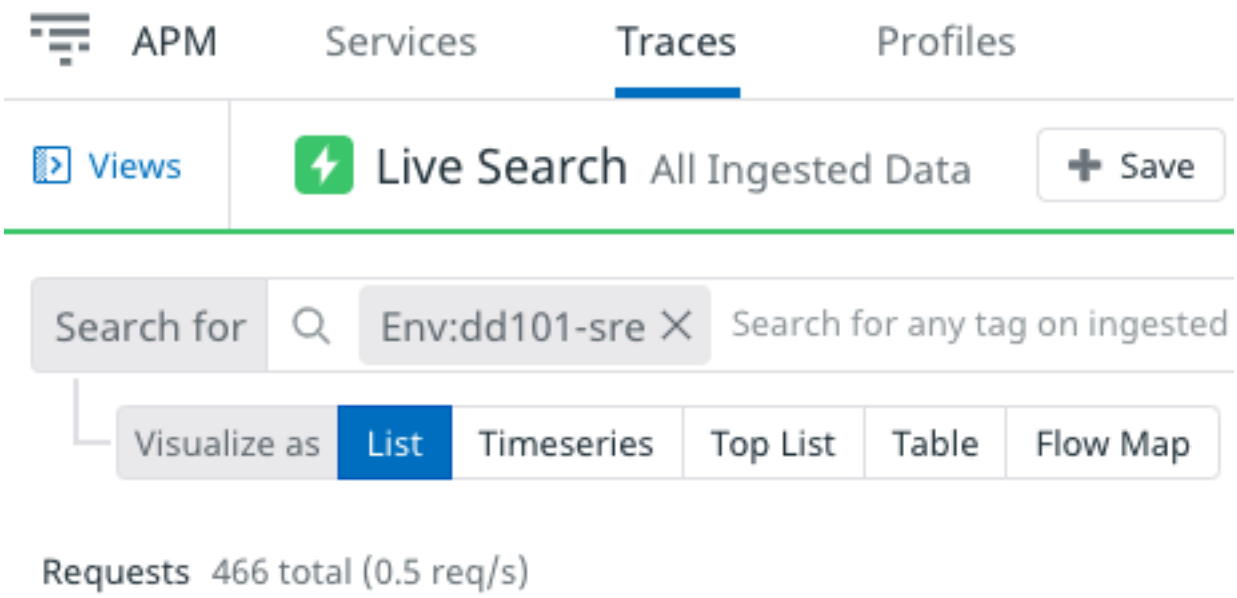
Observe the Network

First, check out the Storedog application. Click the **Storedog** tab to open the Storedog website in a new browser window. The homepage will take a long time to load. It may even cause error messages, such as `Net::OpenTimeout in Spree::HomeController#index`. These are symptoms of a problem that NPM will help you troubleshoot.

Since you started this lab, a background process has been automatically making requests to the Storedog app. You should have a good amount of network traffic to look at in Datadog, even though you can't browse it.

Next, to observe the network metrics in Datadog, you need to first confirm that Storedog is sending metrics to Datadog:

1. Log in to Datadog using the trial credentials the lab created for you. You can run `creds` in the lab terminal whenever you need to retrieve your Datadog training account credentials.
2. Navigate to **APM > Traces** to ensure that the Datadog agent is sending data to Datadog. Make sure that the search field is filtering only `Env:dd101-sre`



   You should see a steady stream of traces resulting from the traffic generated by the lab background process.

If you do not see traces from Storedog services such as `store-frontend`, `advertisements-service`, or `database`, make sure that you're logged in using the trial credentials the lab created for you. You can run `creds` in the lab terminal whenever you need to retrieve your Datadog training account credentials.

Network Performance Page

Next, check out the Network Performance page:

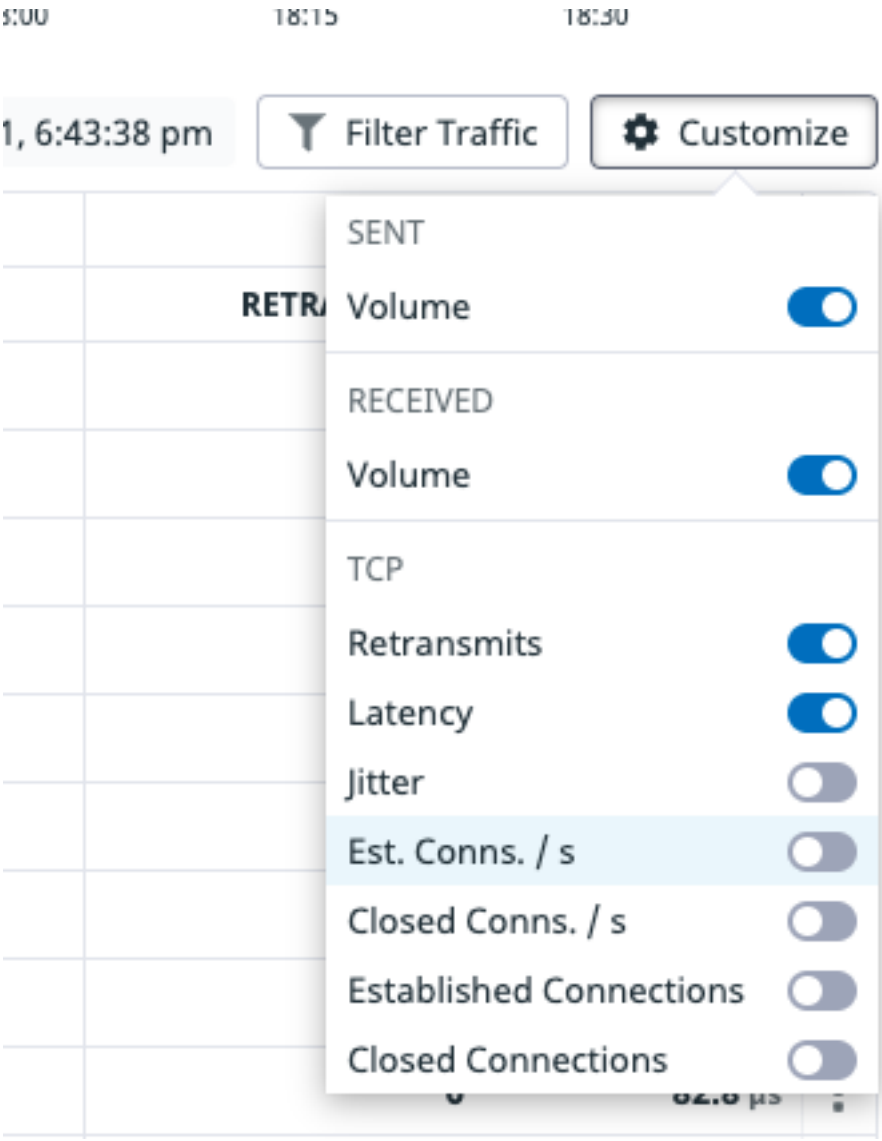
1. Navigate to **Infrastructure > Network Performance**.
2. You might see the Network Performance Overview page.

   Click on the **Analytics** tab to access the Network Performance page. Wait for the data to come in.

   Feel free to read the Network Page documentation while you wait.

   Once Datadog has processed NPM data, this page will display network *flows* that Datadog has detected between the application services. A flow is a network connection between any two tagged objects–from services to availability zones, or from Kubernetes pods to security groups.
3. At the top of the page, in the time range selector, select `1 hour`.
4. Change the **Group by** fields for both **Source** and **Destination** to `service`.

   Note: If `service` is not an option in the **Group by** fields, give Datadog some more time to process networking data.
5. In the **Customize** menu above the network flows table, make sure `Volume` under **Sent** and **Received** is toggled on as well as the `Retransmits` and `Latency` under **TCP**:



When you are done customizing the table, the flow table for the Storedog application should look like the following:

		SENT	RECEIVED	TCP		
SOURCE	DESTINATION	↓ VOLUME	VOLUME	RETRANSMITS	LATENCY	
store-frontend	N/A (Untagged traffic)	58.6 MB / 16.3 KB/s	1.35 MB / 375 B/s	0	221 µs	⋮
N/A (Untagged traffic)	N/A (Untagged traffic)	9.15 MB / 2.54 KB/s	12.5 MB / 3.46 KB/s	14	6.60 ms	⋮
discounts-service	N/A (Untagged traffic)	8.30 MB / 2.31 KB/s	41.3 KB / 11.5 B/s	1.89k	250 µs	⋮
database	discounts-service	3.43 MB / 952 B/s	3.17 MB / 880 B/s	99	12.6 ms	⋮
discounts-service	database	3.08 MB / 856 B/s	3.34 MB / 927 B/s	2.30k	362 µs	⋮
advertisements-service	store-frontend	2.73 MB / 759 B/s	41.6 KB / 11.5 B/s	0	27.8 µs	⋮
discounts-service	store-frontend	1.59 MB / 442 B/s	18.1 KB / 5.03 B/s	240	230 µs	⋮
N/A (Untagged traffic)	aws.elb	1.08 MB / 299 B/s	166 KB / 46.0 B/s	22	88.2 ms	⋮
advertisements-service	N/A (Untagged traffic)	645 KB / 179 B/s	39.5 KB / 11.0 B/s	0	72.0 µs	⋮

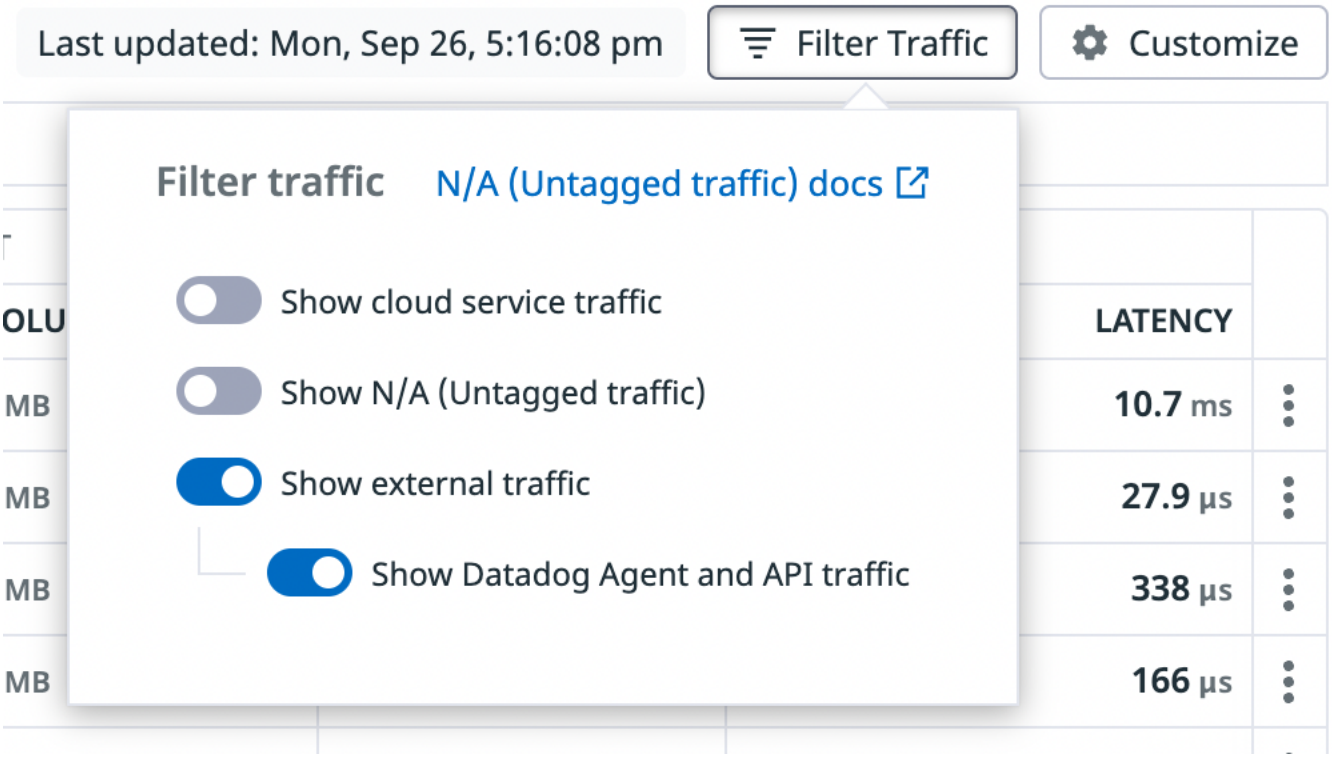
For the Storedog application, you should see many flows between `advertisements-service`, `store-frontend`, `discounts-service`, and `database`. Each flow represents the network communication among these services where one is the source, and another is the destination.

You may see flows where the source and destination are **N/A**. This represents traffic where the source or destination endpoint cannot be resolved. This happens when:

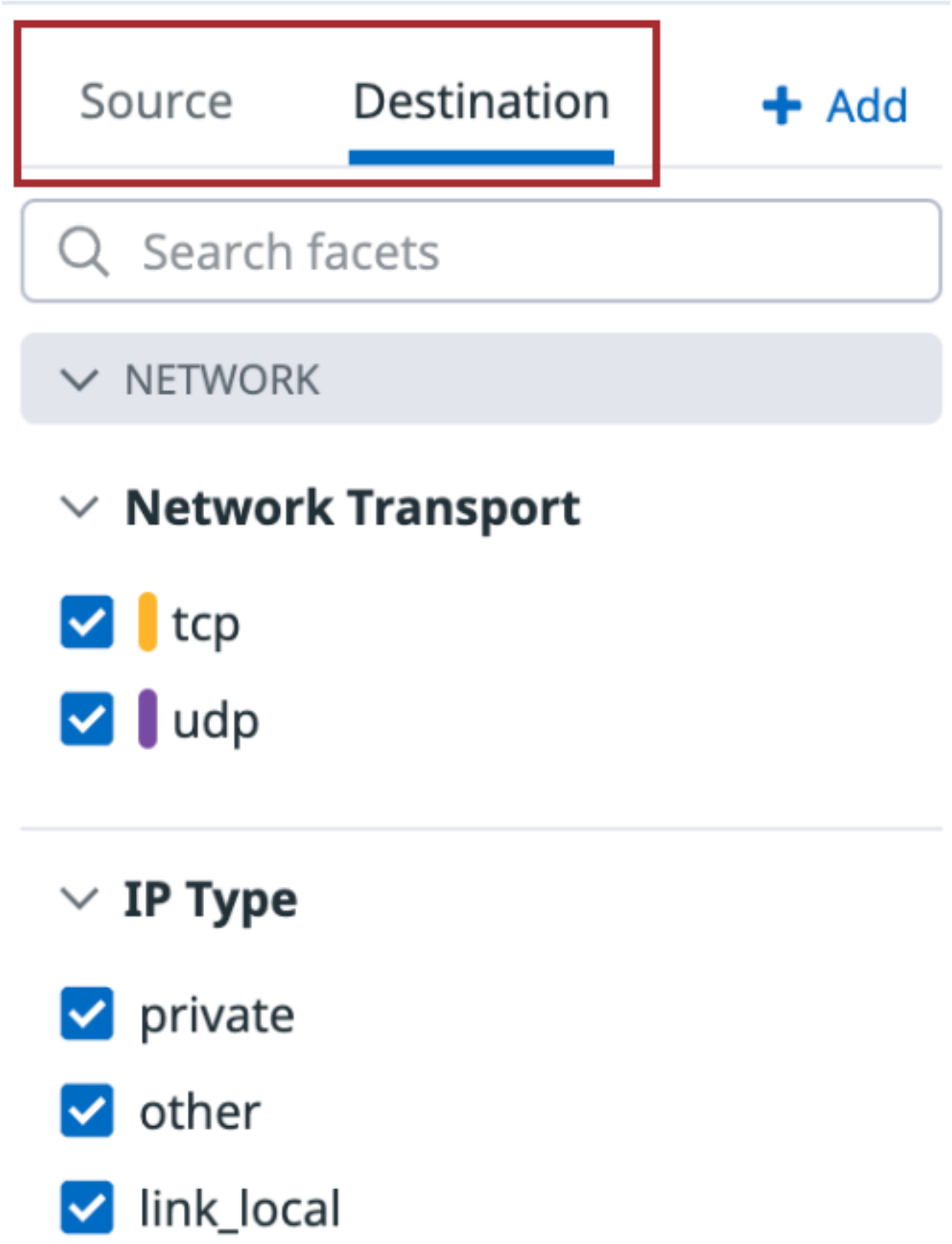
- The host or container source or destination IPs are not tagged with the source or destination tags used for traffic aggregation.
- The endpoint is outside of your private network, and accordingly is not tagged by the Datadog Agent.
- The endpoint is a firewall, service mesh or other entity where a Datadog Agent cannot be installed.

In your current view, the flows marked as `N/A` do not have `service` tags. This includes the container generating traffic to Storedog, and flows to destinations such as package repositories, network time protocol servers, public APIs, etc.

You can hide or display untagged traffic using the **Show N/A (Untagged traffic)** toggle in the **Filter Traffic** settings menu.



NPM automatically indexes a broad range of network traffic details, as exhibited by the facet panel to the left of the network flows table. Notice that facets are grouped by **Source** and **Destination**, which are selectable by the tabs at the top of the facets panel:



Spend some time familiarizing yourself with these facets. If you select facets that display no results in the network flows table, either change the **Group by** fields to ∗, or enable the **Show N/A (Untagged traffic)** toggle in the **Filter Traffic** settings

menu.

> **Note**: If you don't see all of the Storedog services in the table yet, Datadog is still processing NPM data. All the services will appear eventually.

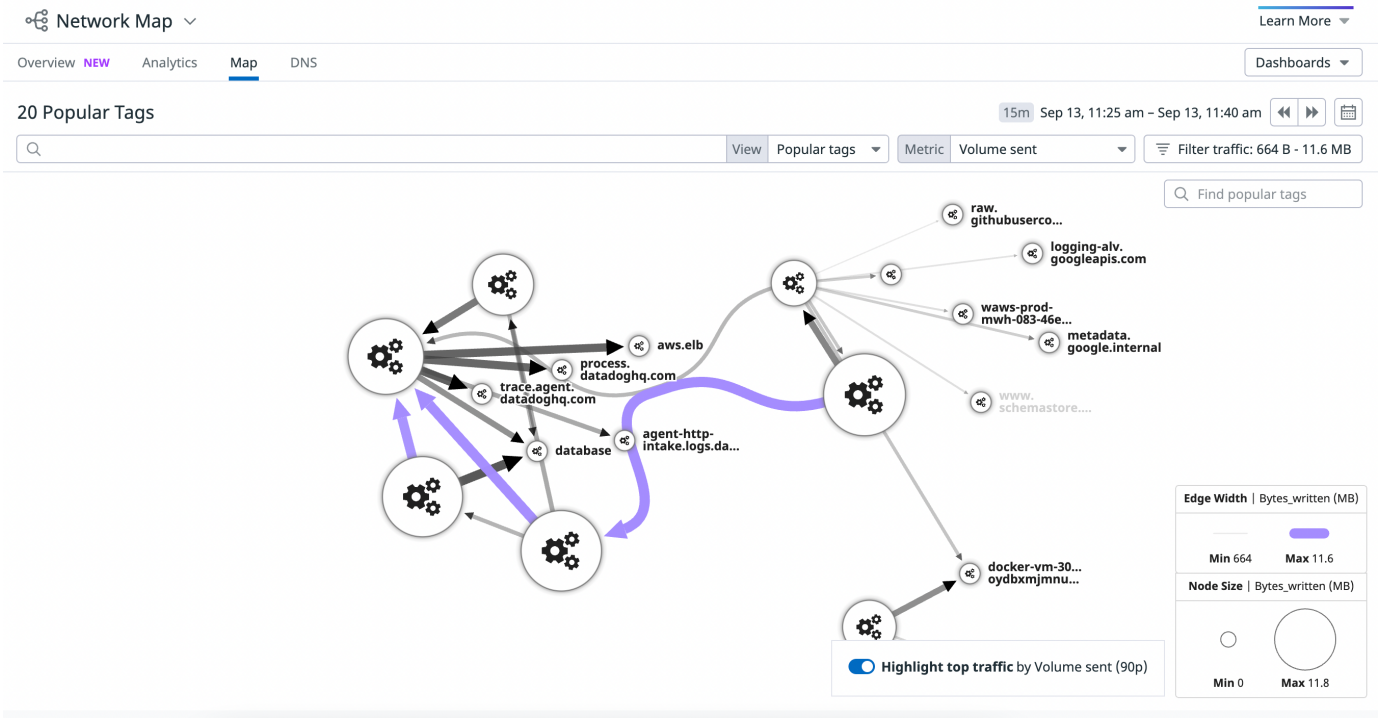
Network Map

Next, check out the Network Map. Like the APM Services Map, it can take a very long time for Datadog to fully render this map after it collects new network data. If you don't see a map yet, you can come back later.

1. To visualize network flows, navigate to **Infrastructure > Network Map**.

   > **Note**: If you are taken to the "Discover Network Performance Monitoring" page, manually change the URL to https://app.datadoghq.com/network/map.

   You should see an intricate network map like the following:

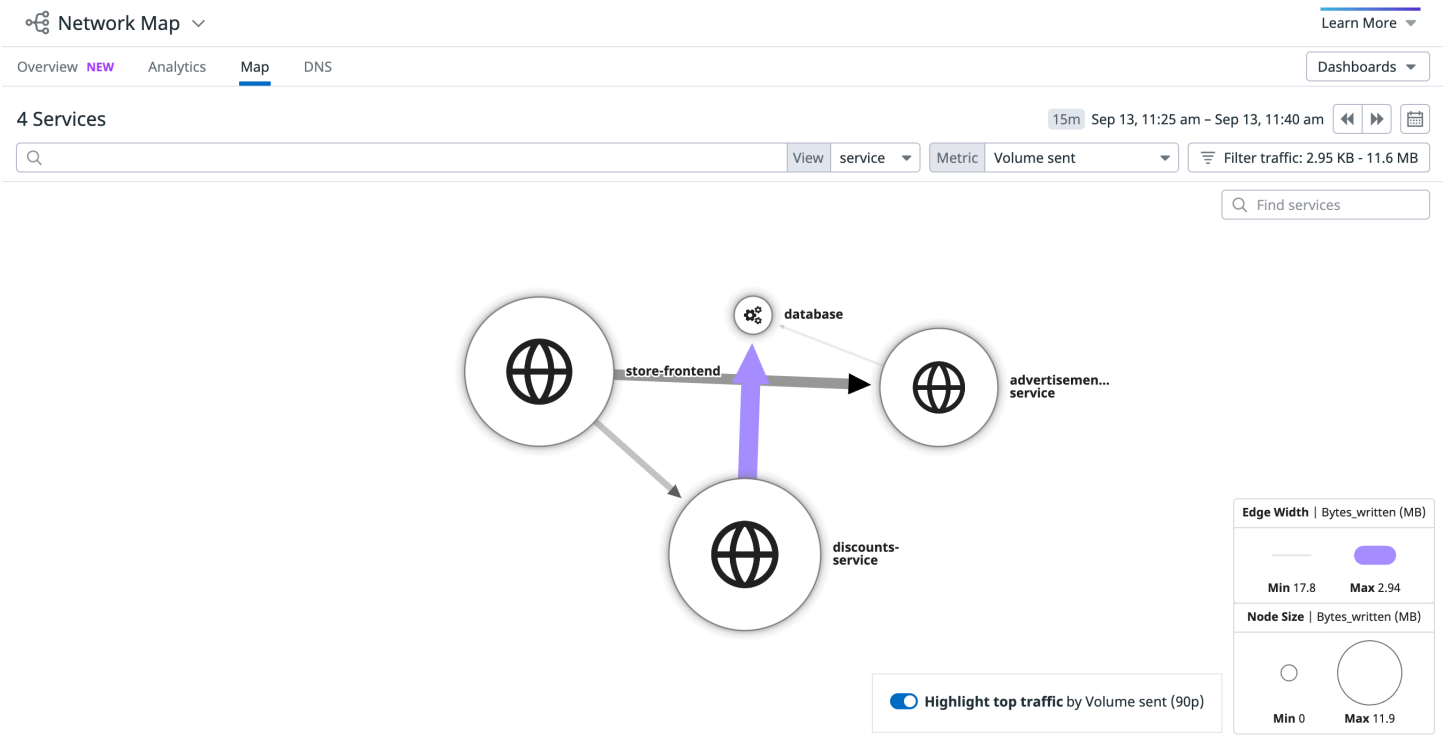


   You can filter out some traffic so it is easier to view.

2. Next to the search bar, change the **View** to `service`.
3. Click **Filter traffic** and toggle the `Show N/A (Untagged traffic)` off to hide untagged traffic.

   Note that you can also filter the range of the chosen metric, `Volume` in this case.

Your network map should now look similar to the following:



The network map is a diagram of the same flows on the network page, where each service's size is scaled relative to the volume of network traffic it has sent. The connections between each service are also scaled to indicate what proportion of that volume was sent to each destination.

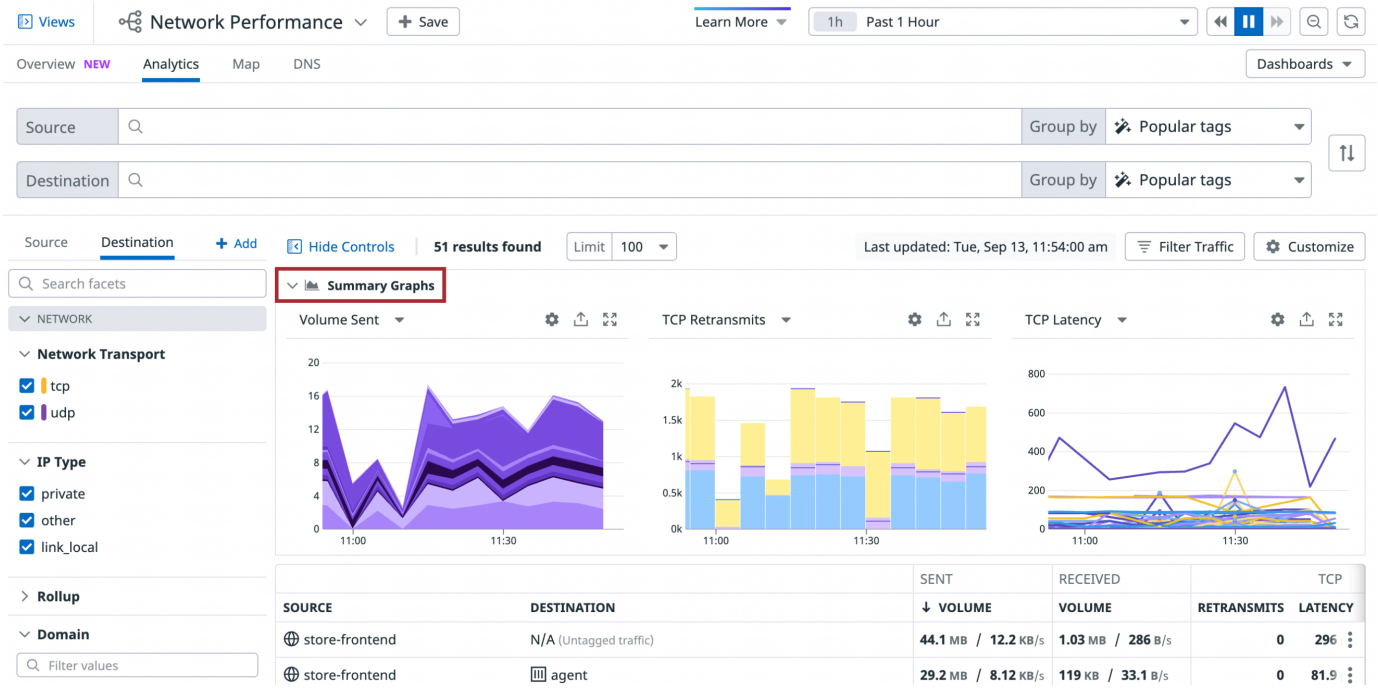
You can hover over each service to see more detailed network information.

You can also choose other metrics by which to scale the diagram using the **Metric** field in the upper-right corner. Select some other metrics to see how they affect the diagram, and what that diagram tells you about the application.

Diagnose Latency

Currently, the Storedog application is slow to respond, might display errors, or might not respond at all. There could be a problem in the Storedog code, or the system resources that it's using. It could also be a problem with a service that it depends upon, or the network connection between them. Because this lab is about Network Performance Monitoring, you can start troubleshooting by looking at the Datadog network page.

1. Navigate to **Infrastructure > Network Performance**.
2. Click on the **Analytics** tab to view the Network Performance page.
3. Click on the **Summary Graphs** header to expand that section and display three graphs: **Volume Sent**, **TCP Retransmits**, and **TCP Latency**.



A couple of issues are readily apparent in the two graphs on the right: A high number of **Retransmits**, and some comparatively high **Latency** for one of the services.

Retransmits represent detected failed TCP packets that are retransmitted to ensure delivery, measured in the count of retransmits from the source. You can hover over the bars of this graph to see the number of retransmits and the flows in which they occurred.

Latency is the time between a TCP frame being sent and acknowledged. Here too, you can hover over the points of the graph to see the round-trip time values and the flows in which they occurred.

You can deduce from these graphs that the common service in these problematic flows is `discounts-service`. To confirm that, take a look at the numbers of **Retransmits** by service that are displayed in the flow table under the graphs, along with **Volume Sent** and average **Latency**.

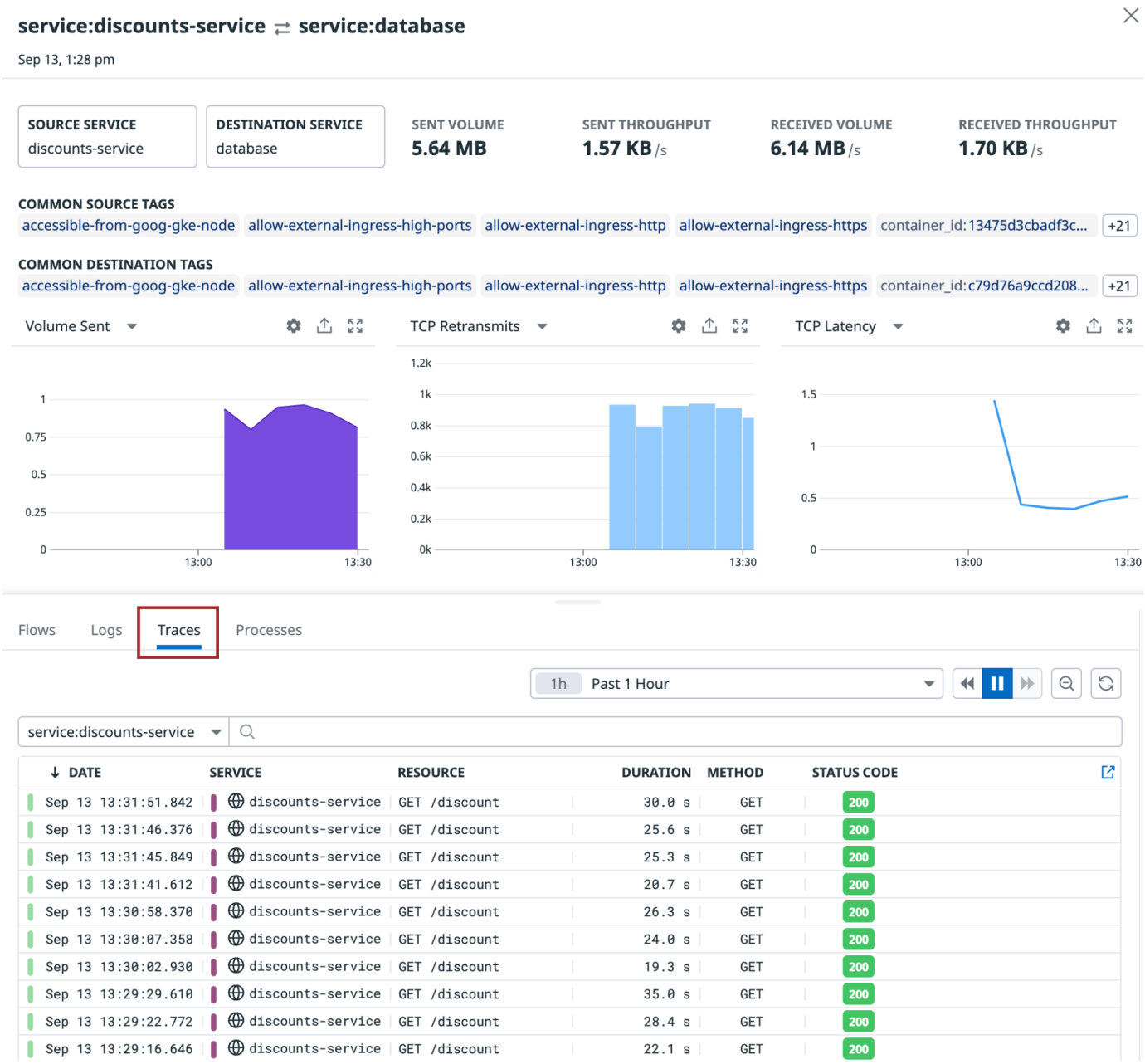
1. Click the **Latency** column heading of the flow table to sort by latency in descending order. You can see that `discounts-service` has the highest latency.
2. Click the **Retransmits** column heading to sort by retransmits. You should see that the largest numbers indeed involve the `discounts-service`.

NPM links to Application Performance Monitoring (APM) so you can look at the code traces associated with network flows. This is useful to dig deeper into applications when you're investigating interesting network metrics.

1. To look at application traces associated with retransmits, click on one of the flows that has a high number of retransmits. This will reveal the flow details side panel.

   Note: If you don't see traces, make sure the time selector is set to `1 hour`.

2. Click on the **Traces** tab near the bottom of the side panel. You can resize the section by dragging the horizontal divider at the top. Here you'll see all the traces related to that flow:

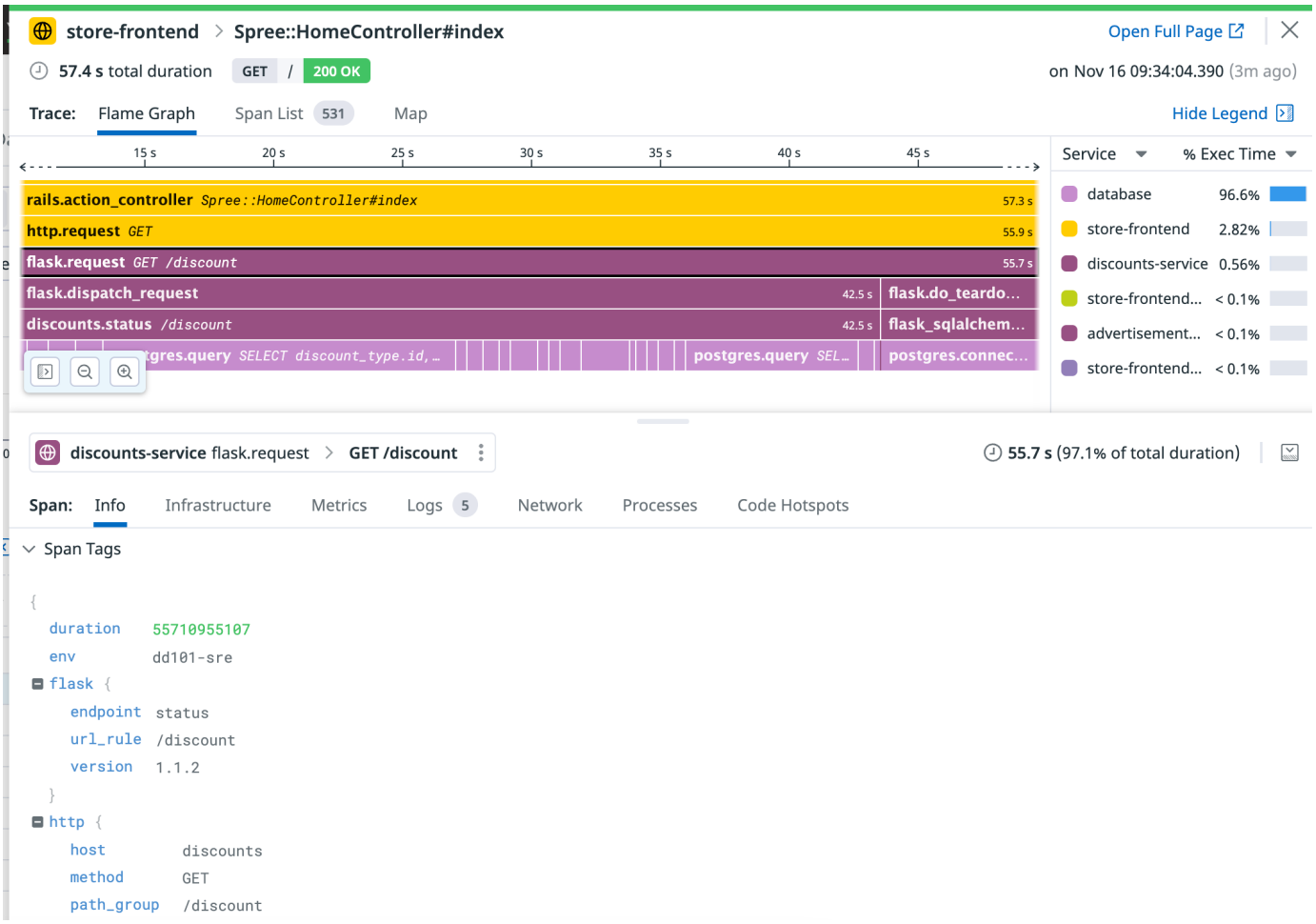


3. Click on one of the traces. This will open the APM trace detail side panel for the time period in which the retransmits occurred.

   Note: If this opens up a blank page, close it and manually navigate to **APM > Traces**.

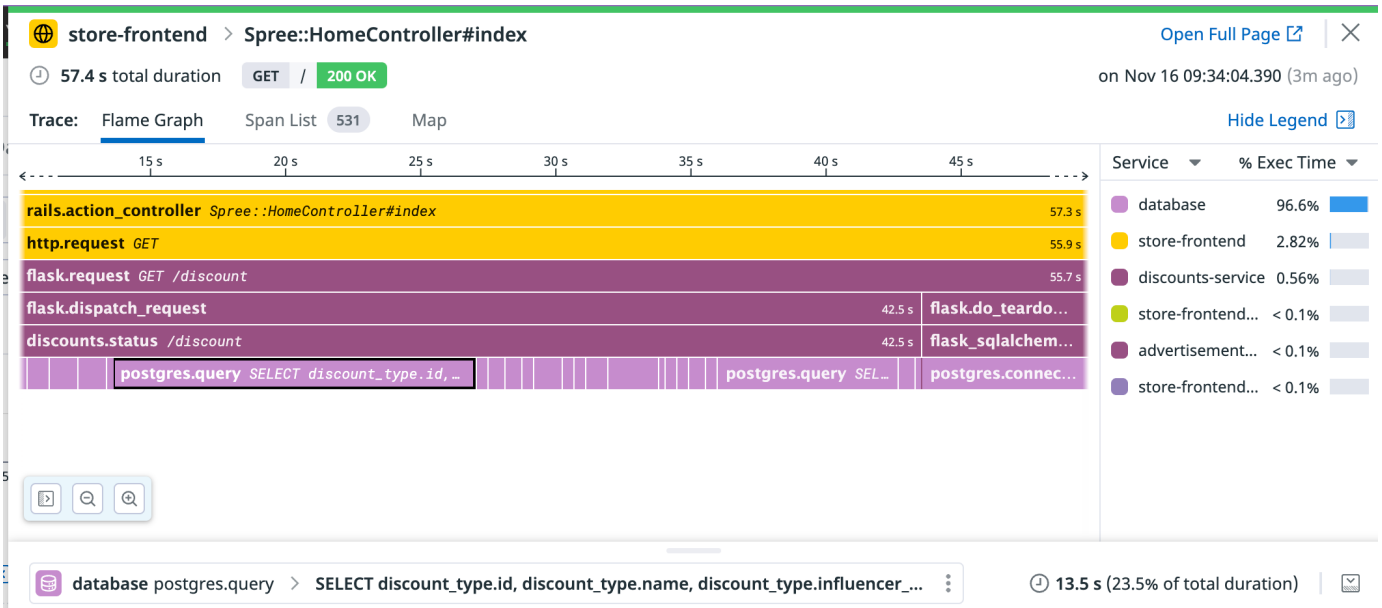
4. Close the trace details side panel. On the Traces pages, scroll through the traces and look for errors or unusually long **Duration** values. You are likely to home in on `Spree::HomeController#index` as a frequent problem.

5. Click on a slow trace to look at the spans:



Looking at these spans, it's clear that the Storedog `Spree:HomeController#index` spent a shocking amount of time making a `GET` request to the `discounts` service. There are empty spans not attributed to application work, and there are no application errors.

You'll also find several traces where `database` seems to be taking up most of the process time, but without evidence of failed queries or application errors:



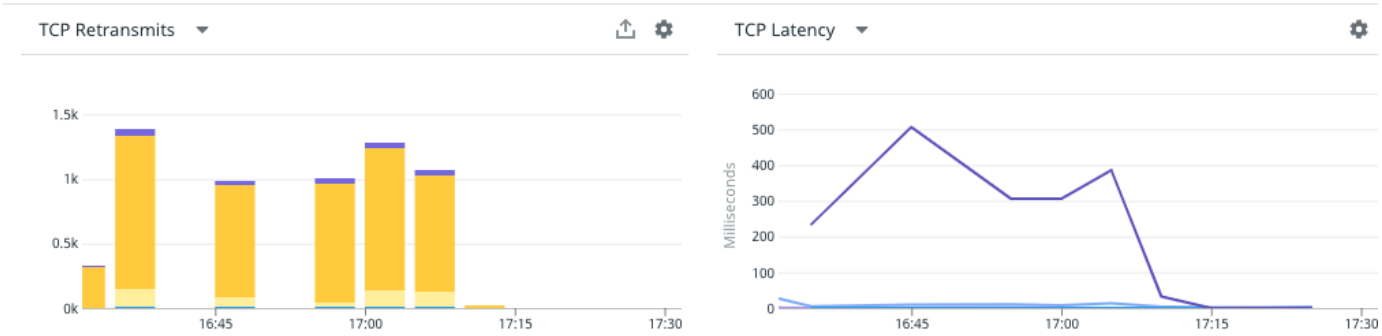
Spend some time inspecting flows on the network page and drilling down into APM traces to find other slow traces. Also take a look at flows involving `advertisements-service`, which are very similar but are not exhibiting poor latency and retransmit measurements.

Following your research, you suspect a faulty network connection on the `discounts-service` container that is causing tremendous packet loss in flows to `database` and `storefront-frontend`. In the real world, you might perform some low-level network diagnostics on the Docker host to explain these symptoms and determine a root cause.

1. In this lab, you can solve the problem by running the following command in the lab terminal:

   ```
   fixnetwork
   ```

2. Navigate back to the Network Performance page and watch the number of retransmits and the average round-trip times taper downward:



   Note: It may take 5 minutes or more to see this change.

Once you're satisfied that latency and retransmits are dropping to zero, move onto the next section.

Domain Names and DNS Monitoring

NPM provides some useful DNS features, such as tagging network flows with resolved domain names, and DNS server monitoring.

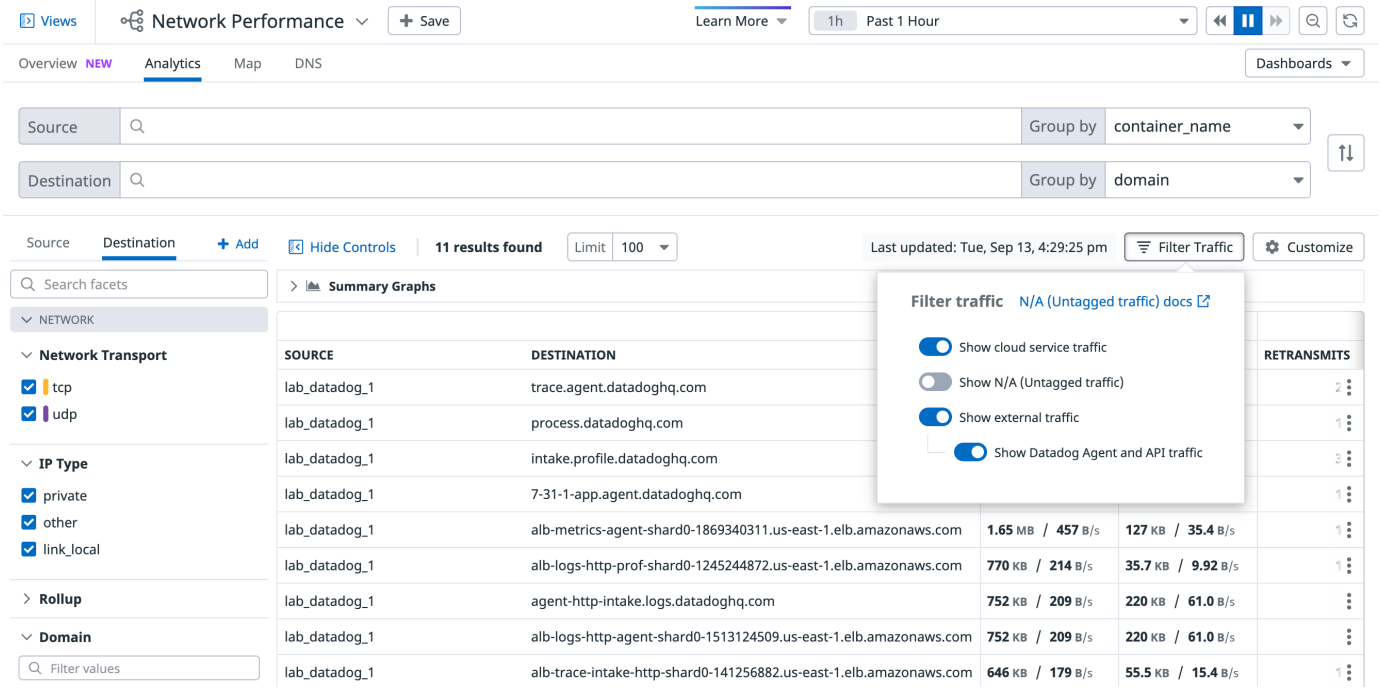
Domain Tags

Datadog will attempt to resolve the domain names of network flow destination IP addresses. This can surface some interesting information.

1. Navigate to **Infrastructure > Network Performance**.
2. Click on the **Analytics** tab to view the Network Performance page.
3. Clear the **Source** and **Destination** search fields at the top of the page.
4. For the **Group by** field next to **Source**, select `container_name`.
5. For the **Group by** field next to **Destination**, select `domain`.

   Note: If `domain` is not an option, you may need to refresh the page.

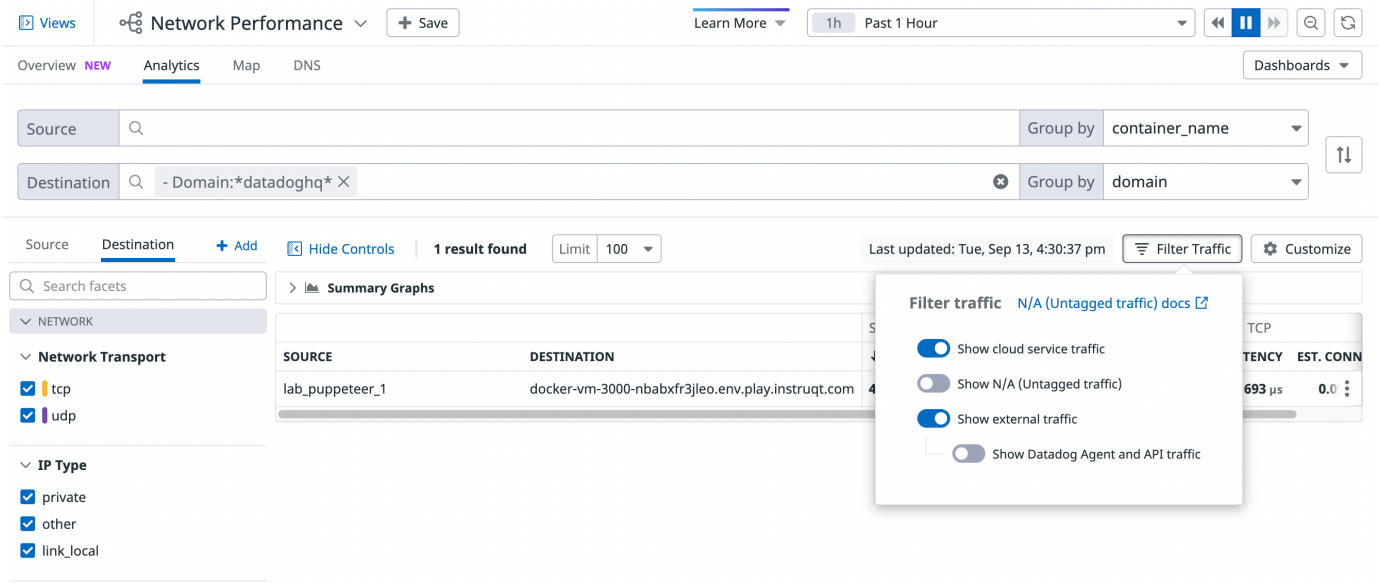
6. Above the flow table, in the **Filter Traffic** settings, disable **Show N/A (Untagged traffic)**.



You'll see all the domain names that NPM was able to resolve and tag in flows originating from Storedog containers during this time period. Note that there are a lot of flows from `lab_datadog_1`, which is the Datadog Agent container.

You can see that the Agent sends different types of data, such as traces, logs, and processes, to different `datadoghq.com` subdomains.

7. In the **Filter Traffic** settings, disable **Show Datadog Agent and API traffic**.



You have isolated the flows to those from tagged containers to tagged external destinations. In this case, destinations tagged with a resolved domain name.

You can see that you're still capturing some Agent flows, but only those going to a Network Time Protocol (NTP) service, which is not a `datadoghq.com` subdomain.

The only other flow listed is from `lab_puppeteer_1`, which is the container automatically generating traffic by loading the Storedog app.

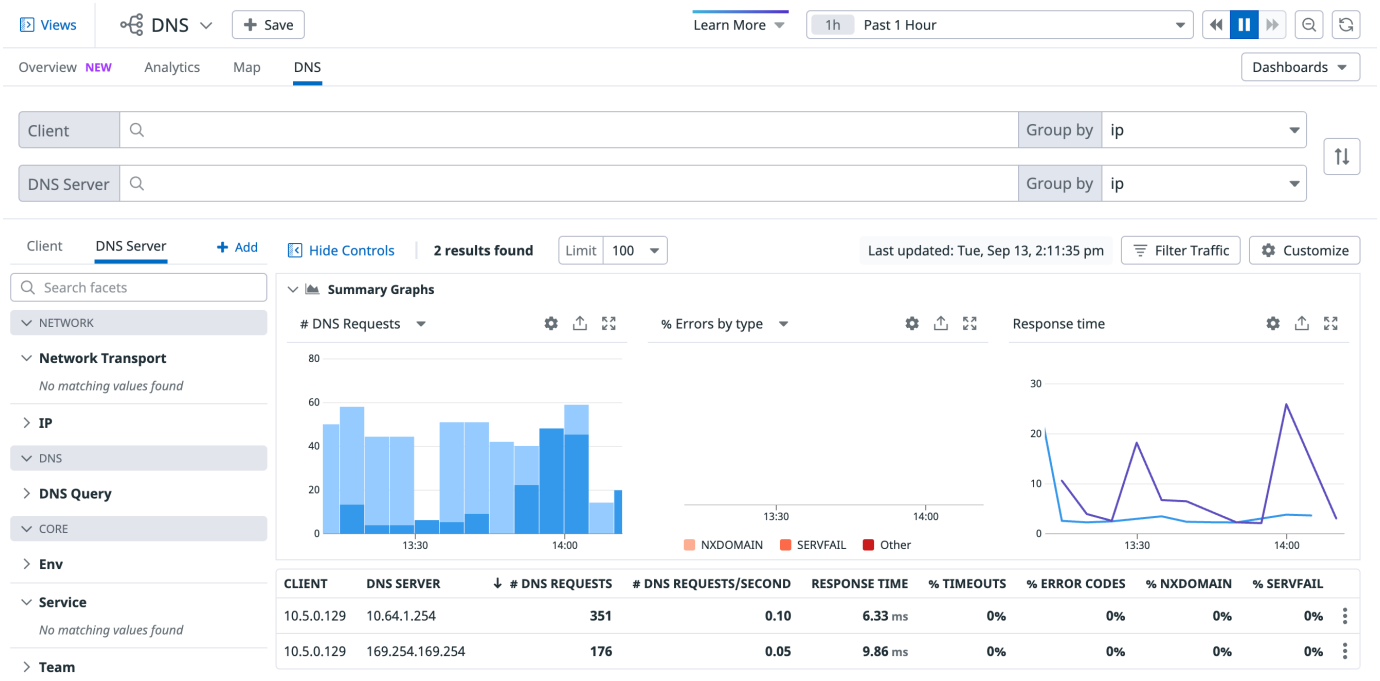
8. In the facets panel to the left, under **Domain**, click on one of the domains near the top of the list. The flow table will filter the related flows to that domain name.
9. Click on the flow to see details about that flow.

This is a useful tool to ensure that your services are talking to the domains you expect them to, and monitor flows involving specific domains.

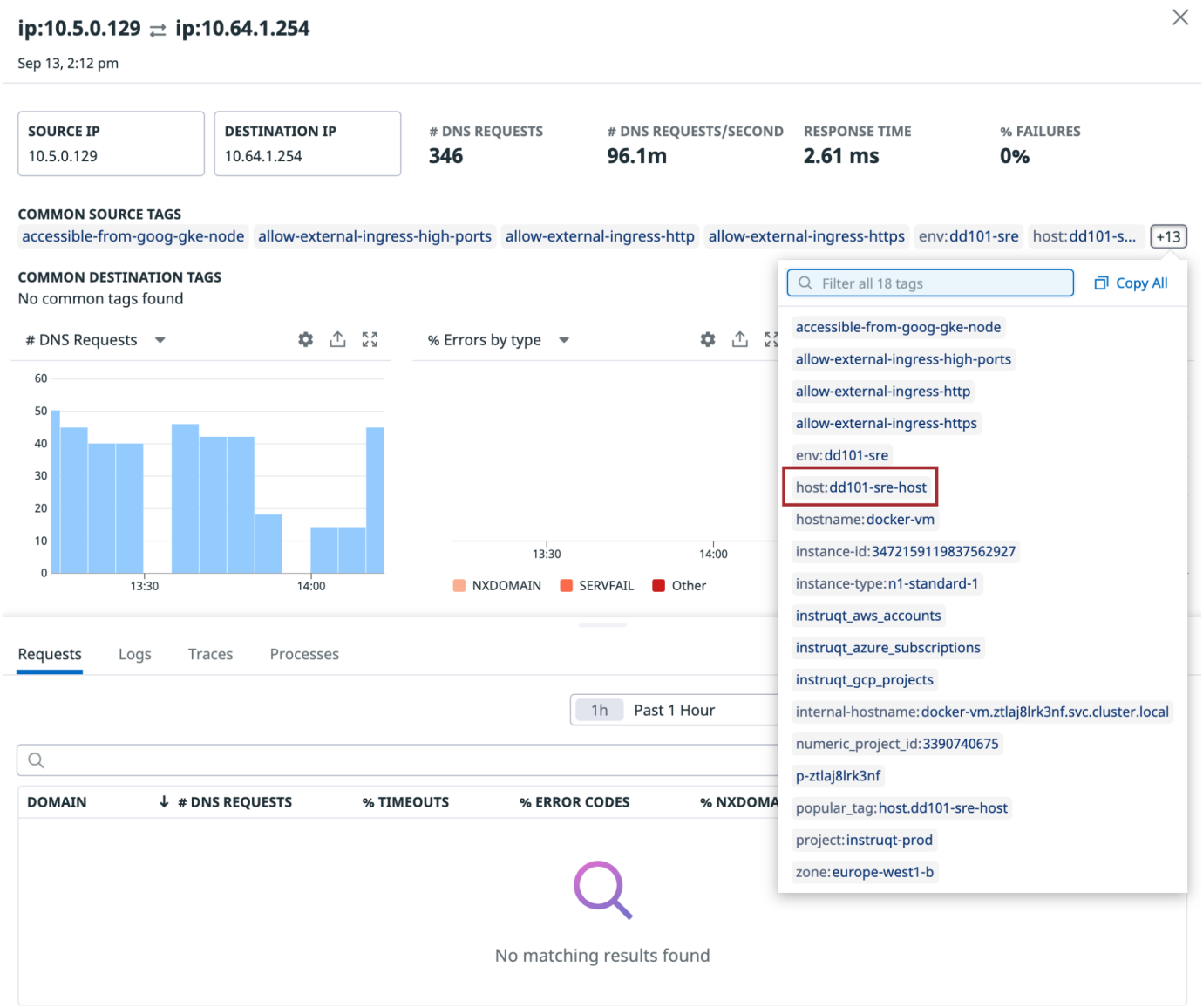
DNS Monitoring

NPM monitors DNS traffic and DNS servers. This lab does not contain its own DNS server; all outbound DNS requests are ultimately made by the Docker host. However, you can monitor those requests.

1. Navigate to **Infrastructure > Network Performance**.
2. Click on the **DNS** tab to view the DNS page.
3. Clear the **Client** and **DNS Server** search fields.
4. For the **Group by** field next to both **Client** and **DNS Server**, select `ip`.
5. Click on the **Summary Graphs** header to expand that section and display three graphs: **# DNS Requests**, **% Errors by type**, and **Response time**.



6. Click on a flow in the results table to see details of the DNS requests in a side panel.
7. Under **COMMON SOURCE TAGS**, you can see `dd101-sre-host`, which is the lab VM running the Docker daemon. Notice the other tags the Agent automatically added to the DNS requests.



If you run your own DNS servers, this is an excellent tool to monitor the health of those servers and the flows to them in your infrastructure.

Lab Conclusion

Congratulations! You learned how to filter and interpret Network Performance Monitoring data. You also used NPM to diagnose performance issues with Storedog.

When you're done, enter the following command in the terminal:

```
finish
```

Click the **Check** button in the lower right corner of the lab and wait for the lab to close down before moving on to the next lesson.