# Analyzing Data with Zeppelin

**Date of Publish:** 2018-12-21


**https://docs.hortonworks.com/**

# Contents

# Analyzing Enriched Data Using Apache Zeppelin

Apache Zeppelin is a web-based notebook that supports interactive data exploration, visualization, sharing and collaboration. HCP users will use Zeppelin at two levels.

- Senior analysts and data scientists can use Zeppelin to produce workbooks to analyze data and to create recreatable investigations or runbooks for junior analysts.
- Junior analysts can use recreatable investigations or runbooks in Zeppelin to discover cybersecurity issues much like they do with the Metron Dashboard. However, Zeppelin can perform more complex calculations and handle larger groups of data.

## Setting up Zeppelin to Run with HCP

You can import the Zeppelin Notebook using Ambari or manually. To complete your set up you'll need to use Zeppelin interpretors and load the telemetry information.

Setting up Zeppelin is very simple. To access Zeppelin, go to http://$ZEPPELIN_HOST:9995.

In addition to this documentation, there are three other sources for Zeppelin information.

- The Zeppelin installation for HCP provides a couple sample notes including tutorials specific to Metron. These notes are listed on the left side of the **Welcome** screen and in the **Notebook** menu.
- Zeppelin documentation provides information on launching and using Zeppelin, and you can refer to the following links for this information:
  - Launching Zeppelin
  - Working with Notes
- Apache Zeppelin documentation provides information on Zeppelin basic features, supported interpreters, and more.

### Using Zeppelin Interpreters

When you install Zeppelin on HCP the installation includes the interpreter for Spark. Spark is the main backend processing engine for Zeppelin. Spark is also a front end for Python, Scale, and SQL and you can use any of these languages to analyze the telemetry data.

### Loading Telemetry Information into Zeppelin

Before you can analyze telemetry information in Zeppelin, you must first download it from Metron. Metron archives the fully parsed, enriched, and triaged telemetry for each sensor in HDFS. This archived telemetry information is simply raw JSON files which makes it simple to parse and analyze the information with Zeppelin.

The following is an example of some Bro telemetry information.

```
%sh

hdfs dfs -ls -C -R /apps/metron/indexing/indexed/bro
/apps/metron/indexing/indexed/bro/enrichment-null-0-0-1484124296101.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-0-1484128332104.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-0-1484131460758.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-1-1484217861096.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-10-1484995461039.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-11-1485081861043.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-12-1485168261040.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-13-1485254661040.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-14-1485341061047.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-15-1485427461040.json
```

```
/apps/metron/indexing/indexed/bro/enrichment-null-0-16-1485513861039.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-17-1485600261045.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-18-1485686661035.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-19-1485773061037.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-2-1484304261042.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-20-1485859461037.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-21-1485945861039.json
/apps/metron/indexing/indexed/bro/enrichment-null-0-22-1486032261036.json
```

You can use Spark to load the archived information from HDFS into Zeppelin.

For example if you are loading information received from Bro, your command would look like the following:

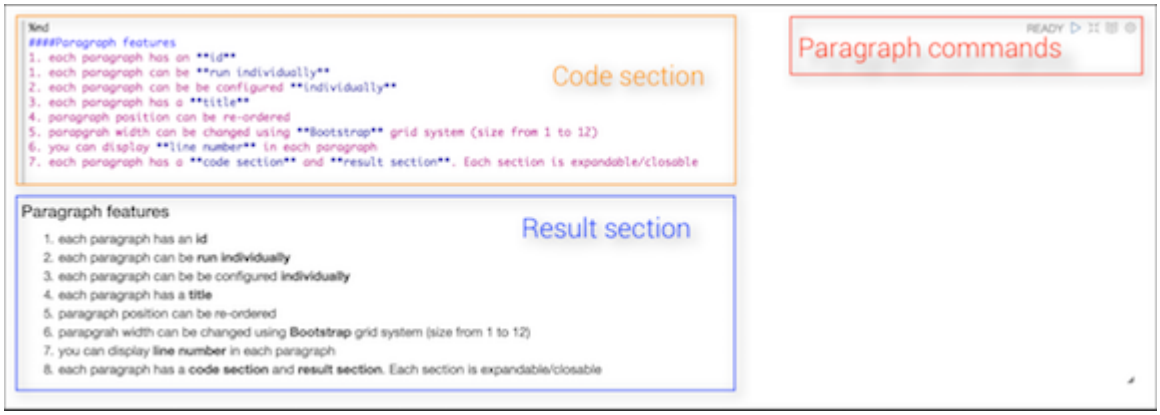
```
%spark
sqlContext.read.json("hdfs:///apps/metron/indexing/indexed/
bro").cache().registerTempTable("bro")
```

## Working with Zeppelin Notes

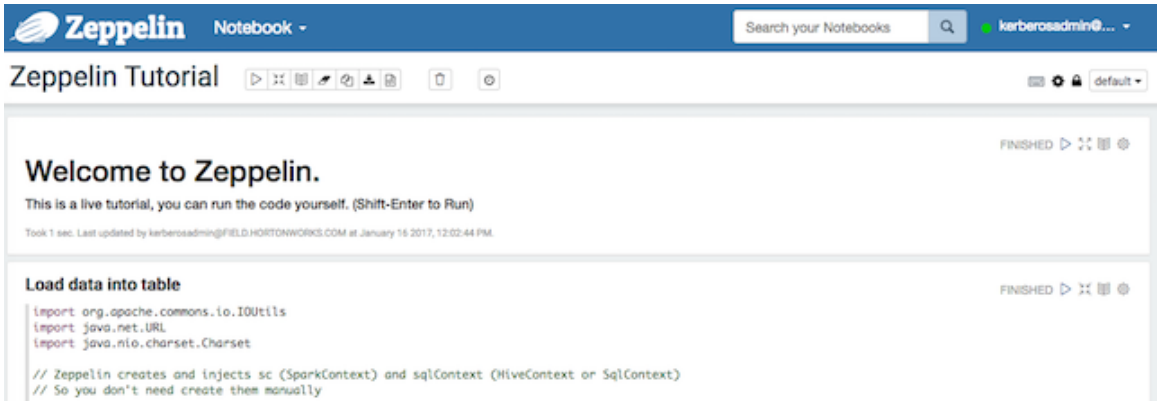
This section provides an introduction to Apache Zeppelin notes.

An Apache Zeppelin note consists of one or more paragraphs of code, which you can use to define and run snippets of code in a flexible manner.

A paragraph contains code to access services, run jobs, and display results. A paragraph consists of two main sections: an interactive box for code, and a box that displays results. To the right is a set of paragraph commands. The following graphic shows paragraph layout.




Zeppelin ships with several sample notes, including tutorials that demonstrate how to run Spark scala code, Spark SQL code, and create visualizations.




To run a tutorial:

1. Navigate to the tutorial: click one of the Zeppelin tutorial links on the left side of the welcome page, or use the Notebook pull-down menu.
2. Zeppelin presents the tutorial, a sequence of paragraphs prepopulated with code and text.
3. Starting with the first paragraph, click the triangle button at the upper right of the paragraph. The status changes to PENDING, RUNNING, and then FINISHED when done.
4. When the first cell finishes execution, results appear in the box underneath your code. Review the results.
5. Step through each cell, running the code and reviewing results.

### Create and Run a Note

Use the following steps to create and run an Apache Zeppelin note.

To create a note:

1. Click "Create new note" on the welcome page, or click the "Notebook" menu and choose "+ Create new note."
2. Type your commands into the blank paragraph in the new note.

When you create a note, it appears in the list of notes on the left side of the home page and in the Notebook menu. By default, Zeppelin stores notes in the $ZEPPELIN_HOME/notebook folder.

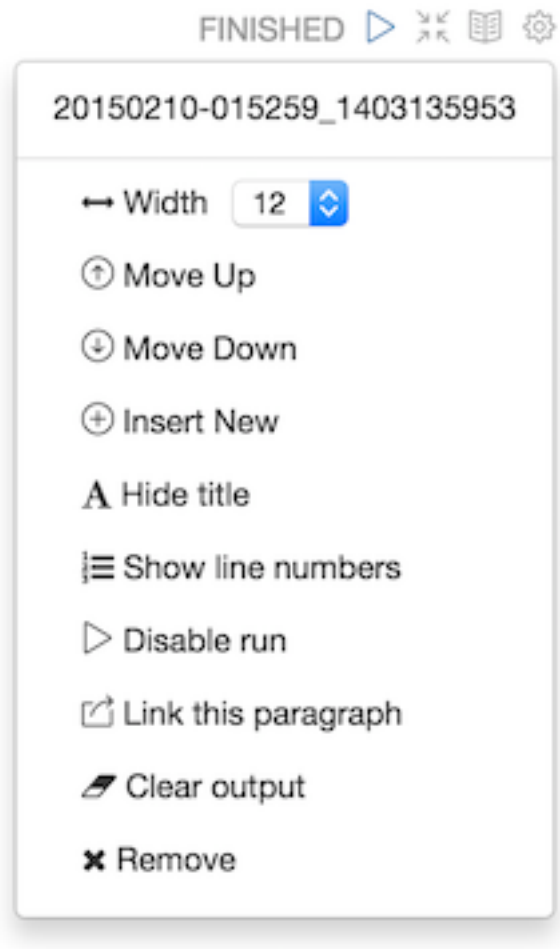
To run your code:

1. Click the triangle button in the cell that contains your code:

2. Zeppelin displays status near the triangle button: PENDING, RUNNING, ERROR, or FINISHED.
3. When finished, results appear in the result section below your code.

The settings icon (outlined in red) offers several additional commands:


These commands allow you to perform several note operations, such as showing and hiding line numbers, clearing the results section, and deleting the paragraph.




### Import a Note

Use the following steps to import an Apache Zeppelin note.

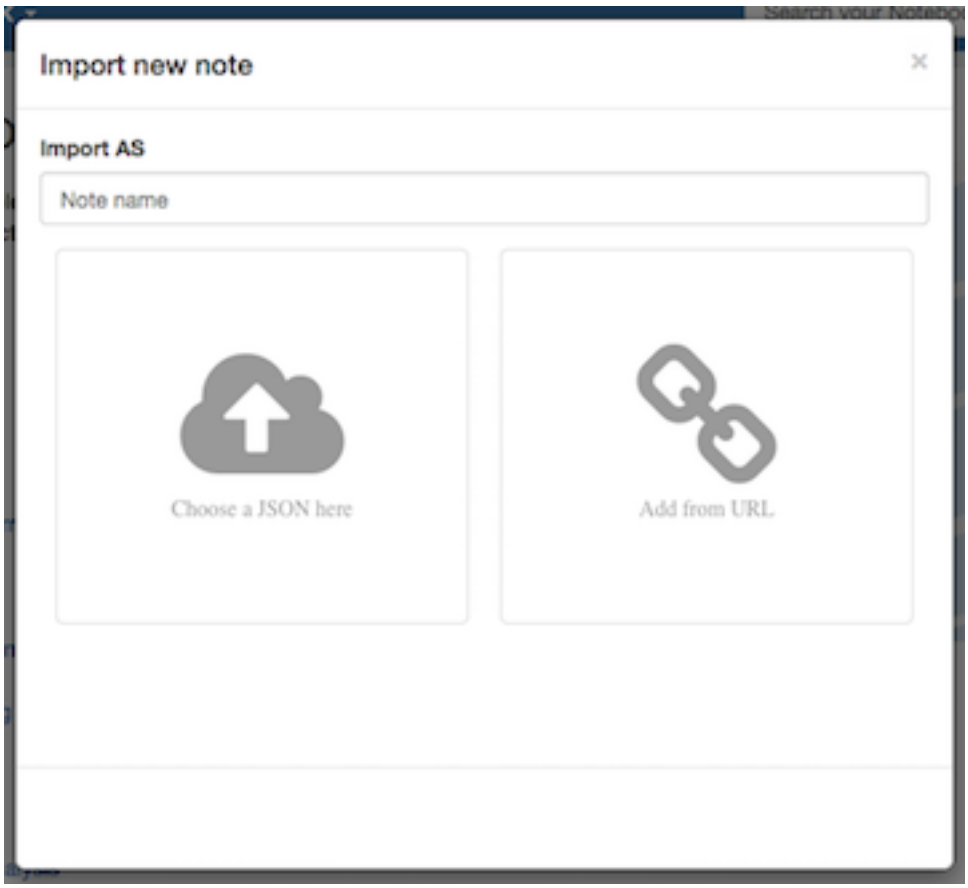
### About this task

To import a note from a URL or from a JSON file in your local file system:

### Procedure

1. Click "Import note" on the Zeppelin home page:


2. Zeppelin displays an import dialog box:



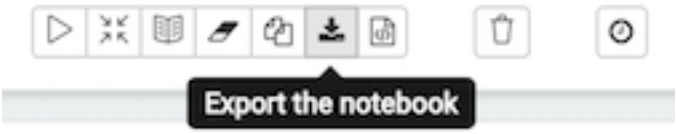

3. To upload the file or specify the URL, click the associated box.

   By default, the name of the imported note is the same as the original note. You can rename it by providing a new name in the "Import AS" field.

### Export a Note

Use the following steps to export an Apache Zeppelin note.

To export a note to a local JSON file, use the export note icon in the note toolbar:




Zeppelin downloads the note to the local file system.

Note: Zeppelin exports code and results sections in all paragraphs. If you have a lot of data in your results sections, consider trimming results before exporting them.

### Using the Note Toolbar

This section describes how to use the Apache Zeppelin Note toolbar.

At the top of each note there is a toolbar with buttons for running code in paragraphs and for setting configuration, security, and display options:

There are several buttons in the middle of the toolbar:

These buttons perform the following operations:

- Execute all paragraphs in the note sequentially, in the order in which they are displayed in the note.
- Hide or show the code section of all paragraphs.
- Hide or show the result sections in all paragraphs.
- Clear the result section in all paragraphs.
- Clone the current note.
- Export the current note to a JSON file.

  Note that the code and result sections in all paragraphs are exported. If you have extra data in some of your result sections, trim the data before exporting it.
- Commit the current note content.
- Delete the note.
- Schedule the execution of all paragraphs using CRON syntax. This feature is not currently operational. If you need to schedule Spark jobs, consider using Oozie Spark action.

There are additional buttons on the right side of the toolbar:


These buttons perform the following operations (from left to right):

- Display all keyboard shortcuts.
- Configure interpreters that are bound to the current note.
- Configure note permissions.
- Switch display mode:
  - Default: the notebook can be shared with (and edited by) anyone who has access to the notebook.
  - Simple: similar to default, with available options shown only when your cursor is over the cell.
  - Report: only your results are visible, and are read-only (no editing).

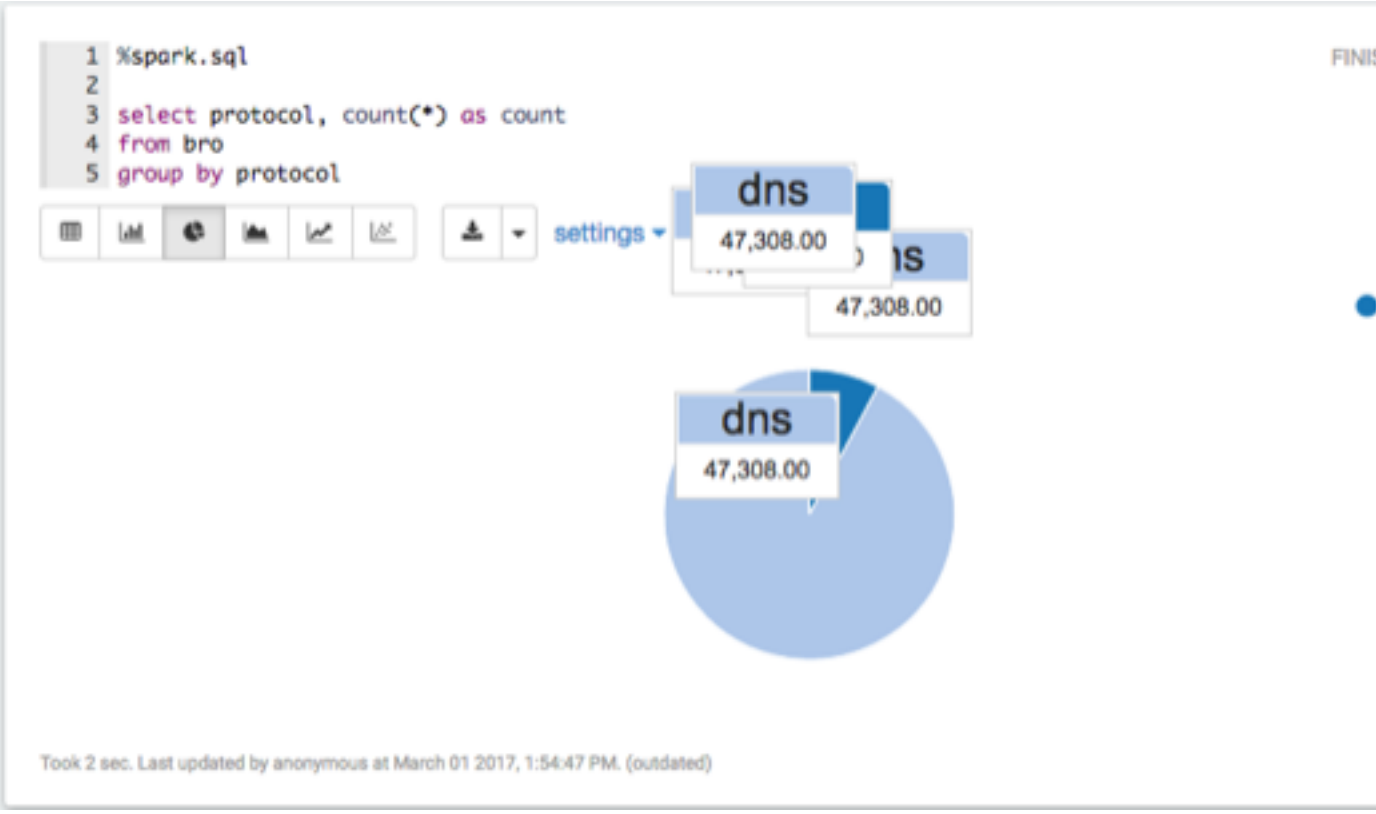
  Note: Zeppelin on HDP does not support sharing a note by sharing its URL, due to lack of proper access control over who and how a note can be shared.

## Using Zeppelin to Analyze Data

Zeppelin enables you to analyze the enriched telemetry information Metron archives in HDFS.

### Procedure

1. Bro produces data about a number of different network protocols. View which types of protocols exist in the data.

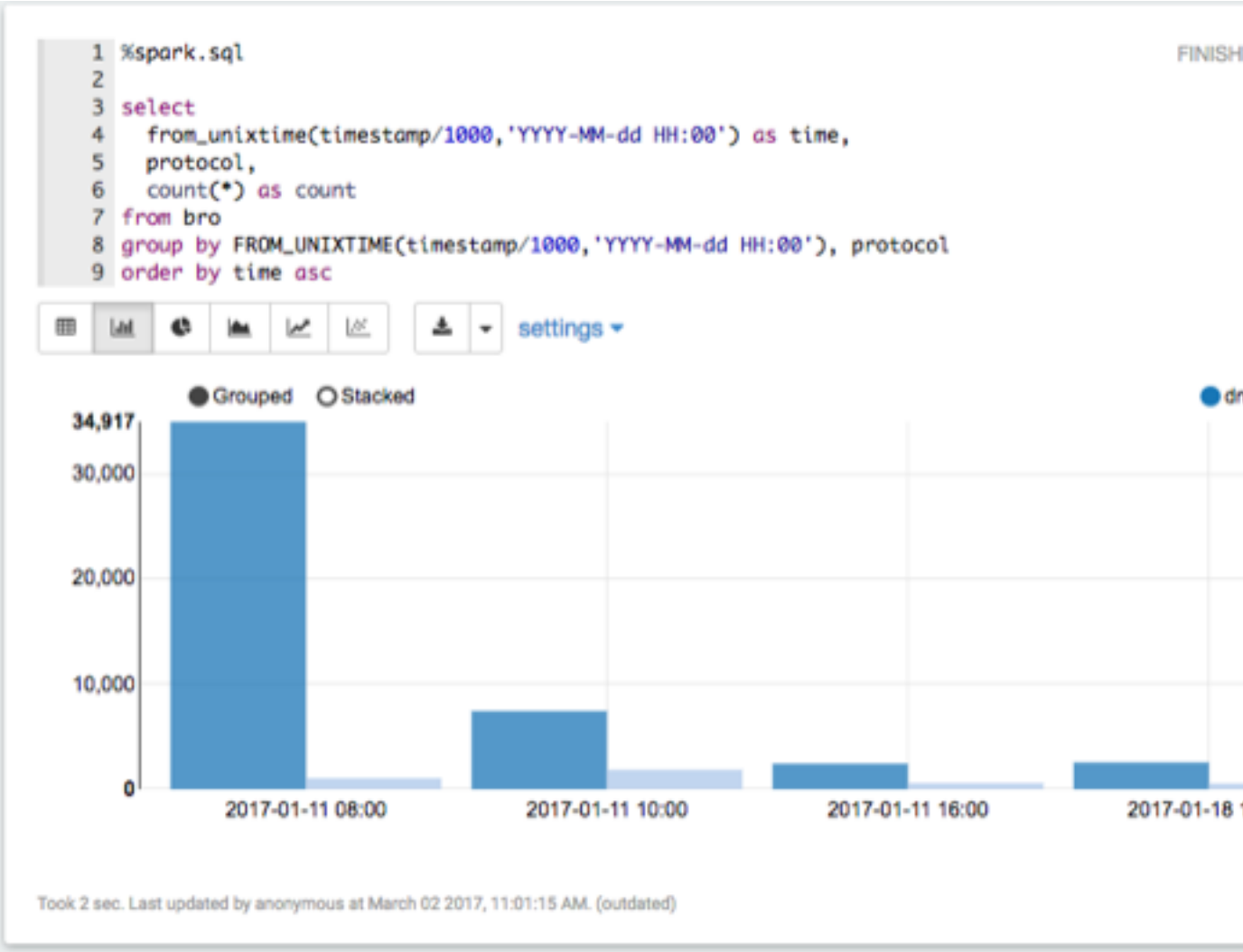



2. View when Bro telemetry information was received.

   You can check for any odd gaps and fluctuating periods of high and low activity.

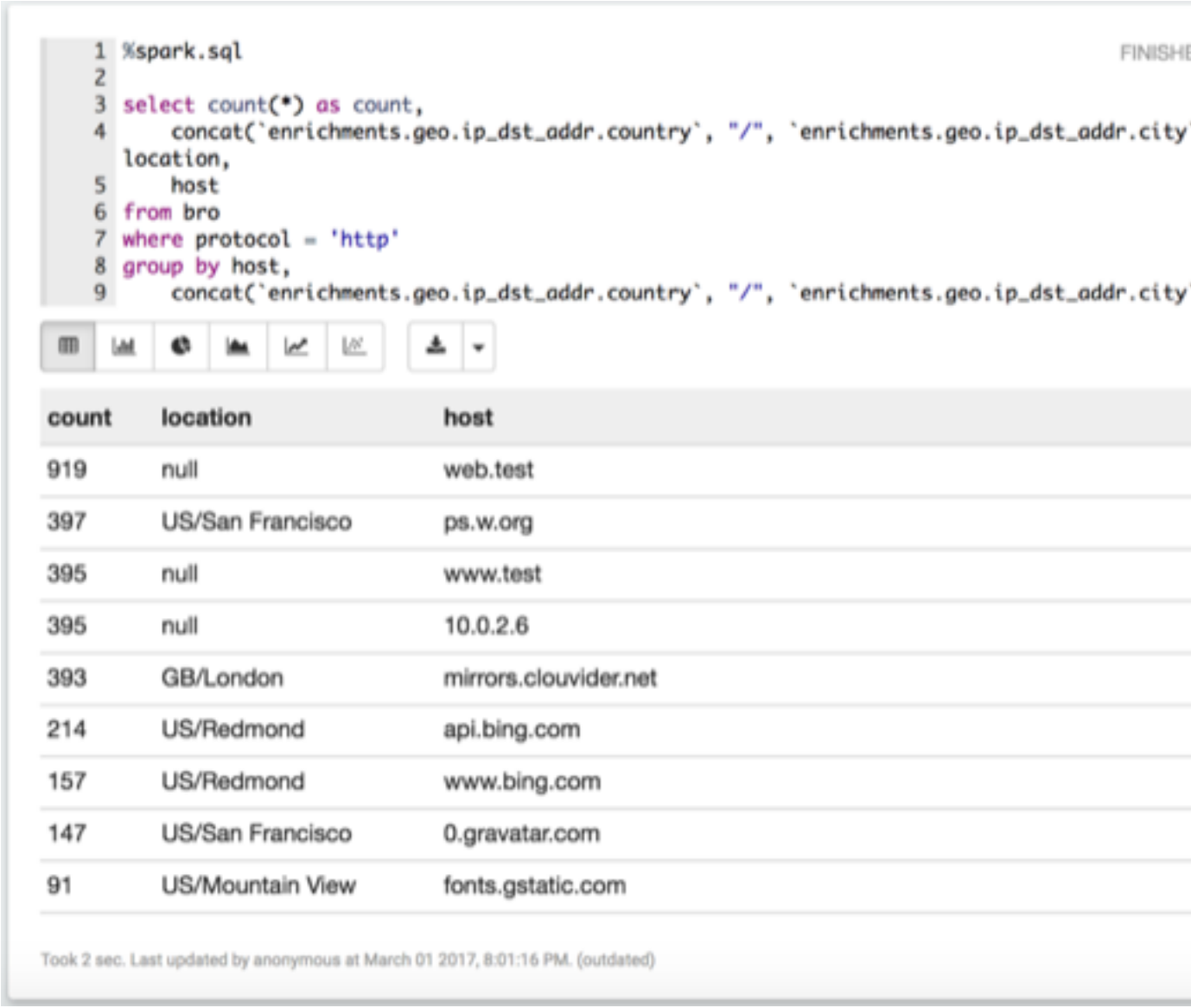
   **Note:** If there is not enough data for this visualization to be interesting, let Metron consume more data before continuing.

   Bro Telemetry Received




3. List the most active Bro hosts.

   Most Active Hosts



```
%spark.sql

select count(*) as count,
    concat(`enrichments.geo.ip_dst_addr.country`, "/", `enrichments.geo.ip_dst_addr.city`) as
location,
    host
from bro
where protocol = 'http'
group by host,
    concat(`enrichments.geo.ip_dst_addr.country`, "/", `enrichments.geo.ip_dst_addr.city`)
```

| count | location | host |
|---|---|---|
| 919 | null | web.test |
| 397 | US/San Francisco | ps.w.org |
| 395 | null | www.test |
| 395 | null | 10.0.2.6 |
| 393 | GB/London | mirrors.clouvider.net |
| 214 | US/Redmond | api.bing.com |
| 157 | US/Redmond | www.bing.com |
| 147 | US/San Francisco | 0.gravatar.com |
| 91 | US/Mountain View | fonts.gstatic.com |

Took 2 sec. Last updated by anonymous at March 01 2017, 8:01:16 PM. (outdated)

4. List any DNS servers running on non-standard ports.

   DNS Servers

```
%spark.sql

select protocol,
    concat(ip_src_addr, ":", ip_src_port) as client,
    concat(ip_dst_addr, ":", ip_dst_port) as server,
    count(*) as requests
from bro
where protocol = "dns"
    and ip_dst_port <> "53"
group by protocol,
    concat(ip_src_addr, ":", ip_src_port),
    concat(ip_dst_addr, ":", ip_dst_port)
```

| protocol | client | server | request |
|---|---|---|---|
| dns | 192.168.66.1:5353 | 224.0.0.251:5353 | 651 |

5. List any mime types that could be concerning.

   Mime Types

```
%spark.sql

select protocol,
    concat(ip_src_addr, ":", ip_src_port) as client,
    concat(ip_dst_addr, ":", ip_dst_port) as server,
    count(*) as requests
from bro
where protocol = "dns"
    and ip_dst_port <> "53"
group by protocol,
    concat(ip_src_addr, ":", ip_src_port),
    concat(ip_dst_addr, ":", ip_dst_port)
```

| protocol | client | server | req |
|---|---|---|---|
| dns | fe80::c0d3:531b:f0cf:f567:57703 | ff02::1:3:5355 | 2 |
| dns | 10.0.2.15:54604 | 224.0.0.252:5355 | 12 |

Took 2 sec. Last updated by anonymous at March 02 2017, 12:20:08 PM. (outdated)

6. Explode the HTTP records.

   Each HTTP record can contain multiple mime types. These need to be 'exploded' to work with them properly.

   Exploded HTTP Records


7. Determine where application/x-dosexec originated.

   Suspicious xdosexc

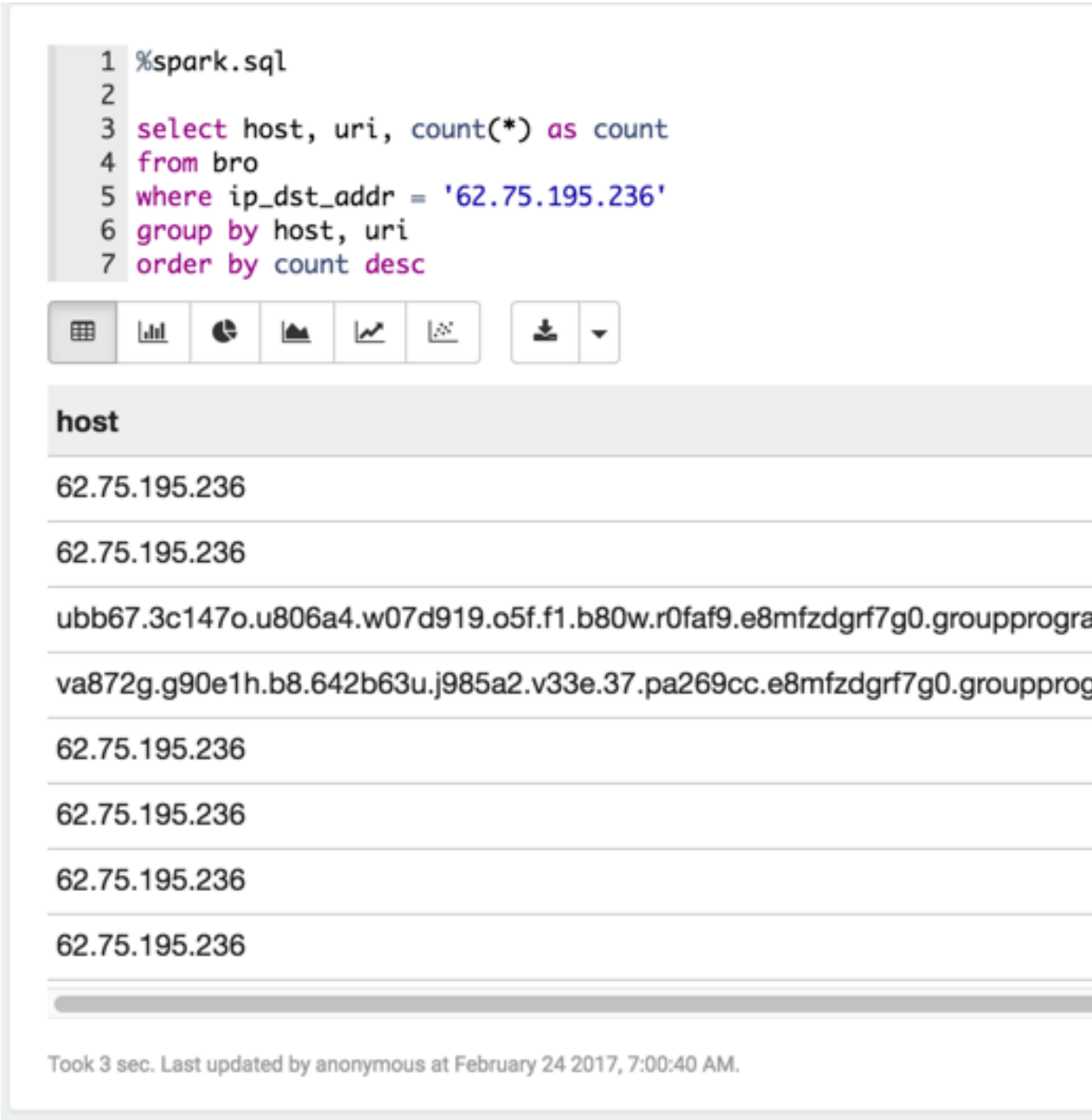
```
%spark.sql

select protocol,
    from_unixtime(timestamp/1000,'YYYY-MM-dd HH:mm') as time,
    ip_src_addr,
    ip_dst_addr,
    `enrichments.geo.ip_dst_addr.city` as city,
    `enrichments.geo.ip_dst_addr.country` as country
from mimes
where mime = 'application/x-dosexec'
```

| protocol | time | ip_src_addr | ip_dst_ad |
|---|---|---|---|
| http | 2017-02-24 14:50 | 192.168.138.158 | 62.75.195 |
| http | 2017-02-24 14:50 | 192.168.138.158 | 62.75.195 |
| http | 2017-02-24 14:52 | 192.168.138.158 | 62.75.195 |
| http | 2017-02-24 14:54 | 192.168.138.158 | 62.75.195 |
| http | 2017-02-24 14:56 | 192.168.138.158 | 62.75.195 |
| http | 2017-02-24 14:58 | 192.168.138.158 | 62.75.195 |

Took 2 sec. Last updated by anonymous at February 24 2017, 6:59:57 AM.

8. Take a look at the requests for x-dosexc.

   x-dosexc Requests




9. Determine when the interactions with the suspicious host are occurring.

   When Interactions Occur


**10.** Create an IP report in Zeppelin using the Metron IP Report notebook.

For a given IP address, this notebook produces a report of:

- Most frequent connections (YAF defaults to 24 hours)
- Recent connections (Yaf, defaults to 1 hours)
- Top DNS queries (Bro, defaults to 24 hours)
- All ports used (Yaf, defaults to 24 hours)
- HTTP user agents (Bro, defaults to 24 hours)

**11.** Create traffic connection request report using the Connection Volume Report notebook.

This notebook lets the user get connection counts filtered by a CIDR block. This notebook can be used for Bro, Yaf, and Snort.

### What to do next

Through this brief analysis we uncovered something that looks suspicious. So far we have leveraged only the geo-enriched Bro telemetry. From here, we can start to explore other sources of telemetry to better understand the scope and overall exposure. Continue to investigate our suspicions with the other sources of telemetry available in Metron.

- Try loading the Snort data and see if any alerts were triggered.
- Load the flow telemetry and see what other internal assets have been exposed to this suspicious actor.
- If an internal asset has been compromised, investigate the compromised asset's activity to uncover signs of internal reconnaissance or lateral movement.

## Zeppelin Notebooks

HCP provides several notebooks that you can use to analyze data and produce reports:

Zeppelin includes tutorials that can help you learn how to use Zeppelin and start analyzing data.

| | |
|---|---|
| **Metron - Connection Report** | This notebook enables you to determine the number of connections made between IPs. This notebook can be set up for Yaf, Bro, or Spark. |
| **Metron - Connection Volume Report** | This notebook enables you to determine the number of connections filtered by a CIDR block. This notebook is set up for YAF. |
| **Metron - YAF Telemetry** | This notebook enables you to obtain flow telemetry information for YAF, including:<br>• Top talkers - internal and external<br>• Flows by hour - internal and external<br>• Top locations<br>• Flow duration internal and external |
| **Metron IP report** | This notebook enables you to produce a report for a given address that includes the following:<br>• Most frequent connections (YAF, defaults to 24 hours)<br>• Recent connections (YAF, defaults to 1 hour)<br>• Top DNS queries (Bro, defaults to 24 hours)<br>• All ports used (YAF, defaults to 24 hours)<br>• HTTP user agents (Bro, defaults to 24 hours) |