

CCP Creating Runbooks with Zeppelin 2.0.0

Creating Zeppelin Runbooks

Date of publish: 2017-11-06

The Cloudera logo, featuring the word "CLOUDERA" in a bold, orange, sans-serif font. The letter "E" is stylized with a horizontal bar through its middle.

<https://docs.cloudera.com/>

Legal Notice

© Cloudera Inc. 2019. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 ("ASLv2"), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER'S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Creating Runbooks Using Apache Zeppelin.....	4
Setting up Zeppelin to Run with CCP.....	4
Using Zeppelin Interpreters.....	4
Loading Telemetry Information into Zeppelin.....	4
Working with Zeppelin.....	5
Using Zeppelin to Create Runbooks.....	5

Creating Runbooks 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 notebooks 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 handle larger groups of data.

Setting up Zeppelin to Run with CCP

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

Procedure

- 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 CCP provides a couple sample notes including tutorials specific to CCP. These notes are listed on the left side of the Welcome screen and in the Notebook menu.
 - Hortonworks Zeppelin documentation provides information on launching and using Zeppelin.
 - Apache Zeppelin documentation provides information on Zeppelin basic features, supported interpreters, and more.

Using Zeppelin Interpreters

When you install Zeppelin on CCP the installation includes the interpreter for Spark. Spark is the main backend processing engine for Zeppelin. Spark is also a front end for Python, Scala, 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 like the following:

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

Working with Zeppelin

The Zeppelin user interface consists of notes that are divided into paragraphs. Each paragraph consists of two sections: the code section where you put your source code and the result section where you can see the result of the code execution. When you use the Spark interpreter, you can enter source code in Python, Scala, or SQL. When you run the code from the browser, Zeppelin sends the code to a backend processor such as Spark. The processor or service then returns results; you can then use Zeppelin to review and visualize results in the browser.

For more information on using notes, see [Working with Notes](#).

Using Zeppelin to Create Runbooks

Zeppelin enables data scientists and senior analysts to create workbooks for junior analysts that can be used as runbooks for recreatable investigations. These runbooks can be static, which require no input from the junior analyst, or dynamic, which require the junior analyst to enter or choose information. You can see an example of a static type of notebook in the Metron - YAF Telemetry note. This section provides instructions for creating both kinds of runbooks.

Procedure

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.

To make your runbook dynamic, use one or more of the dynamic forms that Zeppelin supports:

- Text input form

FINISHED ▶ ⌵ ⌵ ⌵ ⌵

```
%spark.sql
select ip_src_addr,ip_src_port,ip_dst_addr,ip_dst_port
from yaf
where protocol = "TCP"
and ip_src_addr = "${ip_src_addr}"
```

ip_src_addr

192.168.138.158

⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵

ip_src_addr	ip_src_port	ip_dst_addr	ip_dst_port
192.168.138.158	49,196	62.75.195.236	80
192.168.138.158	49,198	72.34.49.86	80
192.168.138.158	49,199	204.152.254.221	80

- Text input form with default value
- Select form

FINISHED ▶ ⌵ ⌵ ⌵ ⌵

```
%spark.sql
select ip_src_addr,ip_src_port,ip_dst_addr,ip_dst_port
from yaf
where protocol = "${protocol=TCP,TCP|UDP}"
```

protocol

TCP ⌵

⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵ ⌵

ip_src_addr	ip_src_port	ip_dst_addr	ip_dst_port
192.168.138.158	49,196	62.75.195.236	80
192.168.138.158	49,198	72.34.49.86	80
192.168.138.158	49,199	204.152.254.221	80
192.168.138.158	49,200	72.34.49.86	80



- Checkbox form


FINISHED ▶ ⌵ ⌵ ⌵ ⌵

```
%spark.sql
select ${checkbox:fields=,ip_src_addr|ip_src_port|ip_dst_addr|ip_dst_port}
from yaf
where protocol = "TCP"
```


fields

☒ ip_src_addr ☐ ip_src_port ☒ ip_dst_addr ☐ ip_dst_port







ip_src_addr	ip_dst_addr
192.168.138.158	62.75.195.236
192.168.138.158	72.34.49.86
192.168.138.158	204.152.254.221

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.

- Run your new code by clicking the triangle button in the cell that contains your code. Zeppelin attempts to run the code and displays the status near the triangle button: PENDING, RUNNING, ERROR, or FINISHED. Zeppelin also displays another empty paragraph so you can add another command.
- Continue adding commands until you've completed the runbook.
- Choose the appropriate type of visualization for your code results from the settings toolbar below the code section of the paragraph.



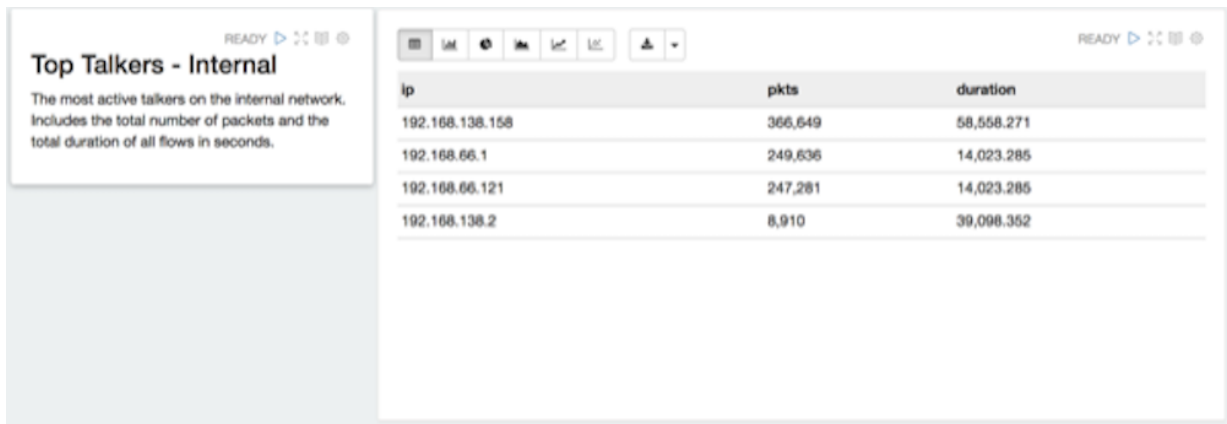
- If appropriate, notify the junior analyst about the runbook that he can clone and use.

What to do next

The following examples provide sample paragraphs you might want to include in a runbook:

- Top Talkers - Internal

This paragraph is static and requires no input from the user.



- Flows by hour

This paragraph is static and requires no input from the user.

