BGP Measurements Project

Table of Contents

Introduction

This project is to be done in the class VM where we have the BGP Stream libraries installed. Per the instructions below, you will need to install additional libraries to complete this project. Your code will need to run without modification in the course VM.

In this project, we will use the *BGPStream* tool and its Python interface *PyBGPStream* to understand the BGP protocol and interact with BGP data. The goal is gaining a better understanding of BGP itself, and to experience how researchers, practitioners and engineers have been using BGPStream to get insights.

More specifically, we will be using a newly developed tool that gives us the option to both browse BGP data in real-time and go back in time and browse through historic BGP data.

Background/Resources and Getting Started

The tool we will be using is called BGPStream. It is an open-source tool that provides access to historical and real-time BGP data.

Paper. We highly recommend that you read the paper that describes this tool to get a high-level overview about how the tool was developed to give access to BGP data. The paper can be found here: https://www.caida.org/publications/papers/2016/bgpstream/bgpstream.pdf

Example Code Snippets. For the following tasks, we require that you use the Python interface of the PyBGPStream module to interact with the tool. You are strongly encouraged to browse the following resources to get familiar with the tool and see ***example code snippets***:

- PyBGPStream API: https://bgpstream.caida.org/docs/api/pybgpstream
- PyBGPStream API Tutorial: https://bgpstream.caida.org/docs/tutorials/pybgpstream
- GitHub repository: https://github.com/CAIDA/pybgpstream
- Official Examples: https://github.com/CAIDA/pybgpstream/tree/master/examples

BGPStream Record Format. The BGPReader command line tool provides you access to BGP records. Here we are show samples raw command line output for illustration.

Updates

We have highlighted the type (A stands for Advertisement) advertised prefix (210.180.224.0/19), the path (11666 3356 3786), and the origin AS (3786):

```
update|A|1499385779.000000|routeviews|route-
views.eqix|None|None|11666|206.126.236.24|210.180.224.0/19|206.1
26.236.24|11666 3356 3786|11666:1000 3356:3 3356:2003 3356:575
3786:0 3356:22 11666:1002 3356:666 3356:86|None|None
```

RIBs

As another example we show the RIB record below. As with the update example the | character separates different fields. RIB records have the following format:

```
<dump-type>|<dump-pos>|<record-
ts>|<project>|<collector>|<router-name>|<router-ip>|<record-
status>|<dump-time>
```

Example with actual data (consecutive | characters indicating fields without data):

```
R|R|1445306400.000000|routeviews|route-
views.sfmix|||32354|206.197.187.5|1.0.0.0/24|206.197.187.5|32354
15169|15169|||
```

For this project you will not have to interact with records in this format. PyBGPStream presents these records in a way that is more accessible from code.

Required Python Packages

Plotting packages

For the plotting portion of this assignment you are free to use any library you want but we recommend using the `matplotlib` library. To use `matplotlib` in the class virtual machine you will need to install the `gi-cairo` backend package. This can be done with the following command:

```
sudo apt install python3-gi-cairo
```

Stats Packages

In this project, you will create several Empirical Cumulative Distribution Function (ECDF) charts from the statsmodels package:

https://www.statsmodels.org/stable/generated/statsmodels.distributions.empirical_distribution.ECDF.html

More information on what an ECDF is can be found at this link:

https://towardsdatascience.com/what-why-and-how-to-read-empirical-cdf-123e2b922480

To use statsmodels, you will need to install the Python package in the VM:

```
pip3 install statsmodels
```

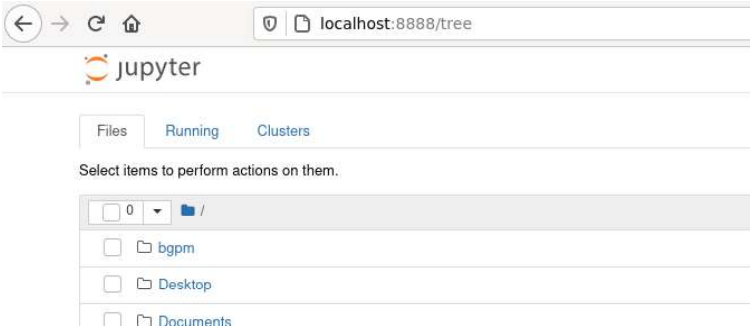
Jupyter Notebook for Development and Delivering

This project uses a Jupyter Notebook for developing and delivering your solution. Setting up and configuring Jupyter notebook may take some time. You are advised to make sure your environment is configured and working early.

Along with this project description, you will find a `bgpm.ipynb` file on the Canvas assignment page for BGP Measurements. You will need to install Jupyter Notebook on the VM as follows:

1. Open the VM and launch a terminal window
2. Enter the following command to make sure you are in your home directory:
   `cd`
3. Install Jupyter Notebook by executing the following command:
   `sudo pip3 install notebook`
4. Make a directory for your BGP Measurements work in your home directory:
   `mkdir bgpm`
5. Open and log into Canvas in the VM browser
6. Download the `bpgm.ipynb` file from Canvas and move it from the download location to the bgpm directory in your home directory
7. Next, launch Jupyter Notebook in your browser. Run this command:
   `jupyter notebook &`

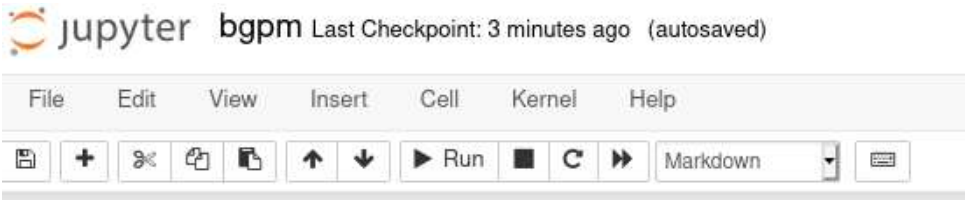
8. You will see this interface at localhost:8888/tree:



9. In the Jupyter Notebook browser interface, click on the `bgpm` directory link
10. You will see the bgpm.ipynb file in the Jupyter Notebook browser, click on it
11. You will see the bgpm.ipynb file open in the Jupyter Notebook in your browser
12. You can also edit .ipynb Notebook files in Pycharm or VS Code, we don't provide support for those scenarios, you'll need to research how your IDE handles Notebook files

A few notes on running Jupyter Notebook files in the browser:

1. At the top of the notebook file display, you will see the following menus and buttons:



2. You can look at the menus, especially the Insert and Cell menus, to see what they do. In addition, you can hover over a button on the button row to see its function.
3. When the cursor is in a cell, you can use Ctrl-Enter (Ctrl held down while pressing the return key) to run the contents of the current cell.
4. If the current cell depends on cells above, you can use the Cell menu to run all cells above
5. To run the entire Notebook, use the Run button
6. **NOTE: To edit the markdown-based non-code fields, double-click on the field you want to edit. When done editing a text field, use Ctrl-Enter to make it a text display field again.**

7. To edit a code field, just click on the field to insert your cursor.
8. The Notebook will auto-save every few minutes
9. Pressing Ctrl-S or using the Save and Checkpoint item on the File menu is a useful way to mark milestones in your code development, you can always revert to a previous checkpoint.
10. For more information about using Jupyter Notebooks, please see the tutorial at https://realpython.com/jupyter-notebook-introduction/
11. Note that once you import a library or define a variable, it is available in the cells below.

Tasks

Task 1. Historic measurements. Prefixes and ASes Growth Over Time.

In this task, we will look into how the number of advertised prefixes and ASes grow over time. The growth of unique prefixes contributes to ever growing routing tables handled by routers in the Internet core.

Cached Data. Locate the directory ***rib_files*** included with this assignment. The directory contains the 8am Routing Information Base (RIB) dump for the first day of January for each year from 2013 to 2021 (inclusive). For example, the file named ris.rrc06.ribs.1357027200.120.cache includes the collector (RIS RRC06) the type of data (ribs) and a timestamp for the historical data (1357027200 which is January 1, 2013 8:00:00 AM).

Read in/Process/Filter. Write code to process the data for all of the .cache files in the *rib_files directory* with PyBGPstream.

You will need to set the stream data_interface to "single file" with the following command: `pybgpstream.BGPStream(data_interface="singlefile")`

After the data_interface has been set, make sure its interface options are set with:

`stream.set_data_interface_option("singlefile", "rib-file", filepath)` *where filepath is a placeholder for a specific cache file.*

A. **Plots.** Using the data from part A, plot a **line graph** that shows the number of total unique prefixes over time.

1. The X-axis should represent time (e.g. January 2013, January 2014 and so on)
2. The Y-axis represents the number of unique prefixes within the specific time period (e.g. January 2013, January 2014 and so on).
3. Repeat steps 1 and 2 for the unique number of ASes over time.
4. Make sure each graph and its axes are clearly labeled.

B. **Questions.** In the text/markdown field after your code cell in the Notebook, answer the following:

1. What do you notice about the overall trend of the two graphs (e.g. how would you describe the overall growth of prefixes, overall growth of ASes)?
2. For each origin AS examine the total unique prefixes they advertise over time. What is the trend that you notice for most origin ASes?
3. Locate the origin ASes that experience the largest growth of advertised unique prefixes over time. To calculate the growth for an AS, identify the first and the last snapshot where the AS appeared in the dataset. Calculate the percentage increase of the advertised prefixes, using the first and the last snapshot. Report the top 10 origin ASes.

Task 2. Historic Measurements. AS-path Length Evolution Over Time

In this task we will look into the AS path lengths and how are they evolving over time. For example, given a collector (vantage point), are ASes reachable over longer or shorter path lengths as time progresses?

Data. For this task you should use the same data as in Task 1.

AS path length. For each origin AS, for each month, compute the shortest AS path length. First, for each origin AS X collect all the paths for which the AS X appears as origin. Compute the length of each path by considering each AS only once (by removing duplicate entries of the same AS) and count the AS hops in the path. Among all the paths compute the shortest path length.

A. Plots. For each month (eg Jan 2013), compute the shortest AS path length for each origin AS, and plot an ECDF of those lengths. Repeat with a separate ECDF for Jan 2014 and so on. Plot all ECDFs on one figure, and label each ECDF.

B. Questions. In the text/markdown fields after your code cell in the Notebook, answer these questions:

1. What is the trend of the AS path lengths progressing over time? Justify your answer with the data you collected – be specific.
2. From the shortest paths you have already computed, which are the top 5 origin ASes with the longest paths per month, and do these ASes stay the same as time progresses?

Task 3. Blackholing events

In this task we will look into Blackhole events. To proceed you will need the following resources:

What is a Blackhole event? https://tools.ietf.org/html/rfc7999#section-2

Example Blackhole event: https://blog.apnic.net/2019/03/22/bgp-communities-a-weapon-for-the-internet-part-2/

A. **Data.** Locate the directory **update_files_blackholing** included with this assignment. This is a collection of BGP Updates from January 2, 2021 5:00 to January 2, 2021 9:00. As in Task 1B, write code to process all of the update cache files.

This time you will need to use "update-file" instead of "rib-file" when setting the data_interface_option

B. **Identifying Blackholing events.** Identify the IPV4 prefixes that are tagged with at least one Remote Triggered Blackholing (RTBH) community. Measure the time duration of the RBTH events. The duration of an RTBH event is the time elapsed between the Announcement and Withdrawal of the IPV4 prefix tagged with the RTBH community - for a given peerIP. Consider only non-zero duration events.

For example, for a single prefix, and for a single peerIP, consider the stream A1 A2 A3_RTBH_tagged W1 W2 W3 W4. Compute the duration as the time difference between A3_RTBH_tagged and W1.

Consider all peerIPs in your given dataset to look for events. For each prefix consider all RTBH events it is associated with.

C. **Plots.** Plot an ECDF of the event durations.

D. ***Questions.*** *In the text/markdown fields after your code cell in the Notebook, answer these questions:*

1. How many prefixes did you identify?
2. How many RTBH events did you identify for all your identified prefixes?

Task 4. Durations of Announcements Withdrawal (AW) Events

In this task, we will measure how long do prefix Announcements last before being withdrawn? In this task, we will only consider explicit AW withdraws - not implicit. An explicit AW withdrawal occurs when a prefix is advertised with an Announcement, and then it is withdrawn with a W. (In

contrast, an implicit withdrawal occurs with a prefix is announced with an Announcement and then another Announcement follows with different BGP attributes.)

A. **Data.** Locate the directory **update_files_pt4** included with this assignment. This is a collection of BGP Updates for a two-hour window starting at 21-01-02 05:10. As with previous tasks, you will need to write code to process all of the cache files.

   To compute the duration of an Explicit AW event, for a given prefix, you will need to monitor the stream of Announcements (As) and Withdrawals (Ws) separately per peerIP. For example, for a prefix, and for a peerIP you have a stream A1 A2 A3 W1 W2 W3 W4. The duration of the explicit AW event for this prefix is the time difference between A3 and W1.

   Consider only non-zero AW durations.

B. ***Questions.*** *In the text/markdown fields after your code cell in the Notebook, answer these questions. Please note that we are only interested in non-zero durations.*
   1. Which are the tuples <advertised prefix IPV4 – peerIP> with the top 10 shortest AW durations?
   2. What are the corresponding durations?

Submission

To submit this project, submit your Notebook .ipynb file to the Canvas assignment page.
Per the Piazza post on "How to Submit", you will submit your Jupyter Notebook .ipynb file directly to Canvas as GTLogin_bgpm.ipynb -- do not zip the file.

NOTE: GTLogin should be replaced with your ID you use to log into Canvas (e.g., smith7 as in smith7_bgpm.ipynb

1. Your submission must have all the graphs generated in the notebook before saving. In other words, when the graders open your notebook file, the graphs should already exist on the page. Please make your answers clear and do not include extraneous output. Your answers will be graded by people.
2. Make sure your Jupyter Notebook file works in the virtual machine. That is the environment we will use to grade it.
3. When done, submit to Canvas. Then, in the VM, log into Canvas and download your submission and test it out to make sure it works there. Make sure your submission uploaded correctly. Late submissions will result in reduced or zero points.

Grading Rubric

Points	Task to be completed
25	Task 1: - Line graph showing unique prefixes over time (5 pts) - Line graph showing ASes over time (5 pts) - Observations (5 pts each)
25	Task 2: - Line graph (2 pts / ECDF – 20 pts total) - Observation 1 (3 pts) - Observation 2 (2 pts)
25	Task 3: - Question 1 (5 pts) - Question 2 (5 pts) - ECDF showing event duration (15 pts)
25	Task 4: - Question 1 (10 pts) - Question 2 (15 pts)

Honor Code / Academic Integrity / Plagiarism

Please refer to the Georgia Tech Honor Code located here:
https://policylibrary.gatech.edu/student-affairs/academic-honor-code

We strictly enforce Section 3. Student Responsibilities including these prohibited actions:

- Unauthorized Access: Possessing, using, or exchanging improperly acquired written or verbal information in the preparation of a problem set, laboratory report, essay, examination, or other academic assignment.
- Unauthorized Collaboration: Unauthorized interaction with another Student or Students in the fulfillment of academic requirements.
- Plagiarism: Submission of material that is wholly or substantially identical to that created or published by another person or persons, without adequate credit notations indicating the authorship.
- False Claims of Performance: False claims for work that has been submitted by a Student.

What You Are Allowed to Share

1. For this project you can share ***ONLY*** graphs generated by your experiments.
2. You are not permitted to share data you collected/generated, stats you collected/generated, any code from which these graphs are generated, etc.
3. You cannot share your Jupyter Notebook files