TO UNDERSTAND ANOMALIES..
...YOU MUST FIRST UNDERSTAND YOURSELF

2024-06-28

Wes Hardaker <hardaker@isi.edu>

Network operators are plagued with odd anomalies

Let’s think about this

How can we do *intelligent* analysis of anomalies?

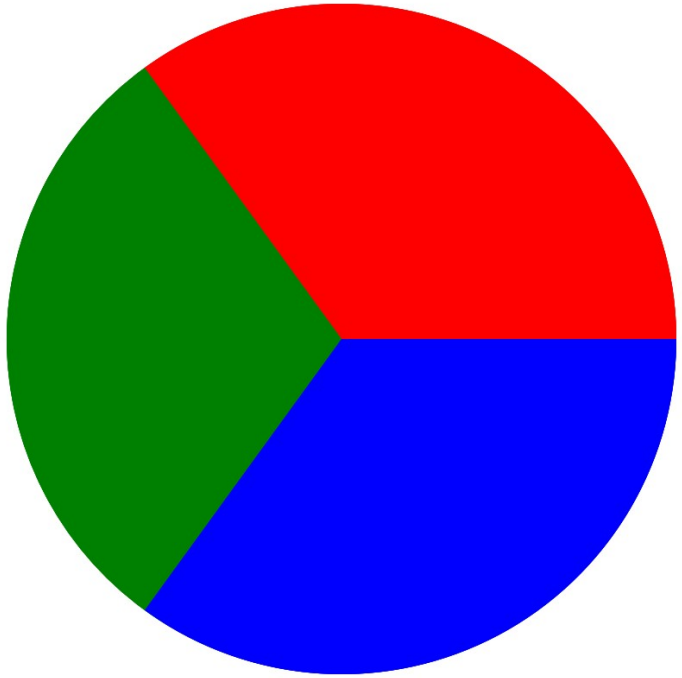
How do we know *what is normal traffic* v.s. *what is new*?

To find what is abnormal, you need to know what is normal

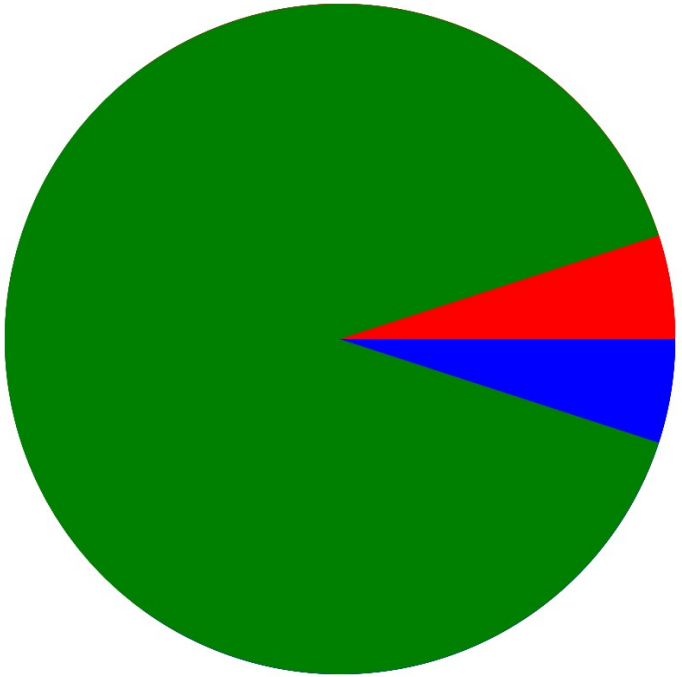
- Doctors use **bilateral comparison**
  - When your foot hurts… compare your left to your right
  - Is one more swollen than the other?
  - Is one larger?
  - Differently colored?
  - Imagine if we didn't have a left and a right!
  - Your doctor would have to memorize what normal was

- Network anomalies should be treated with similar analysis

An example DDoS attack field values
Spot the difference!
What value is the problem?

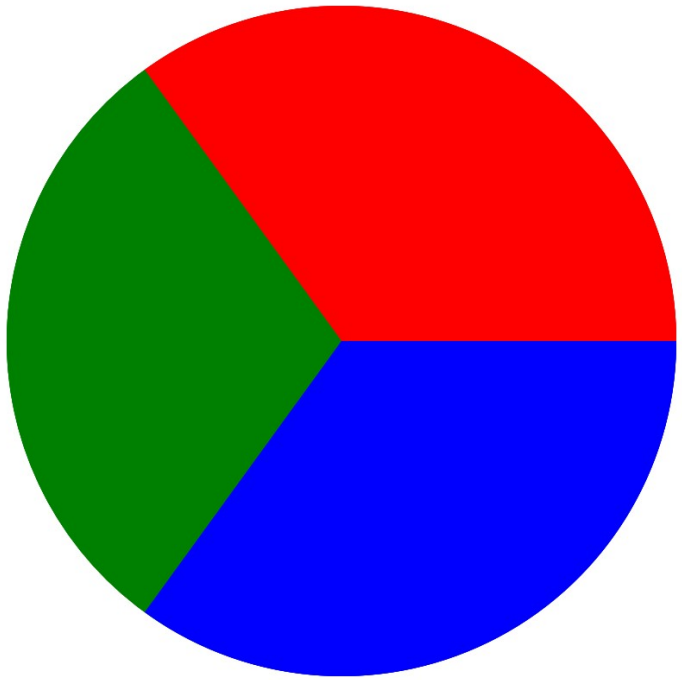


Normal Traffic

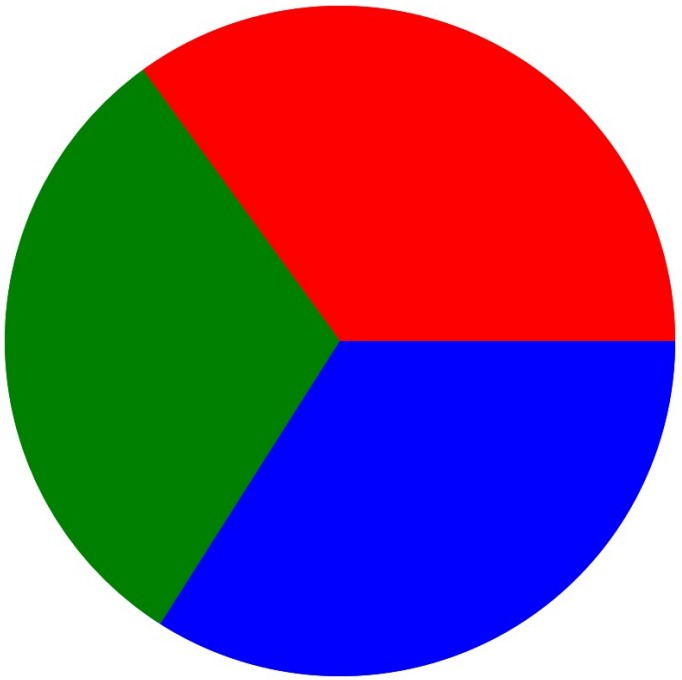


Attack Traffic Sample

A smaller anomaly – spot the difference!

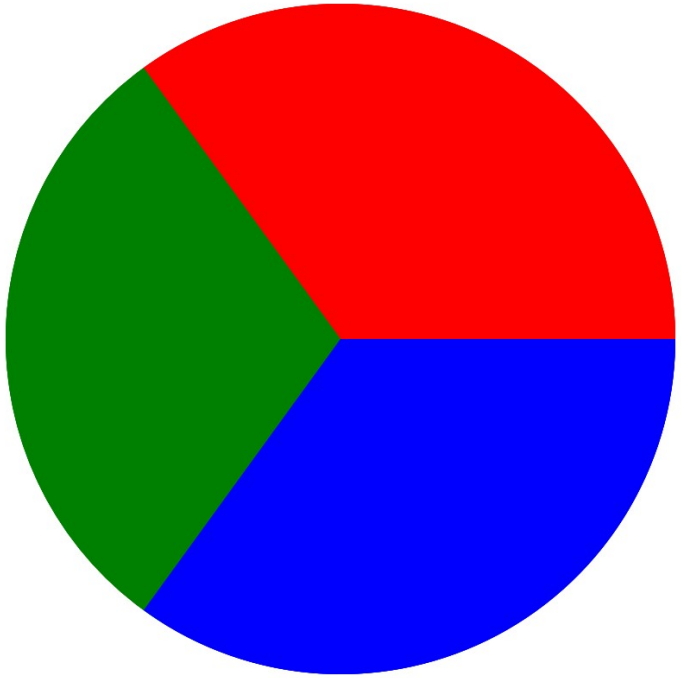


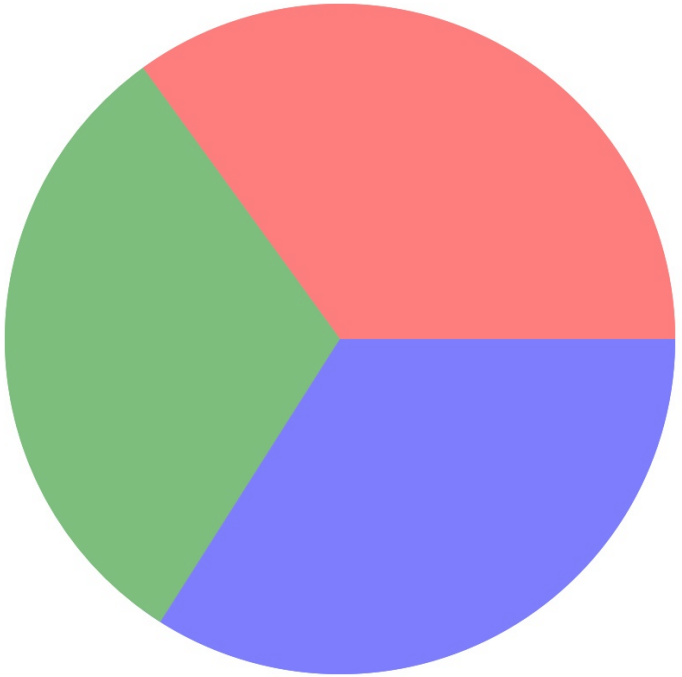
Normal Traffic



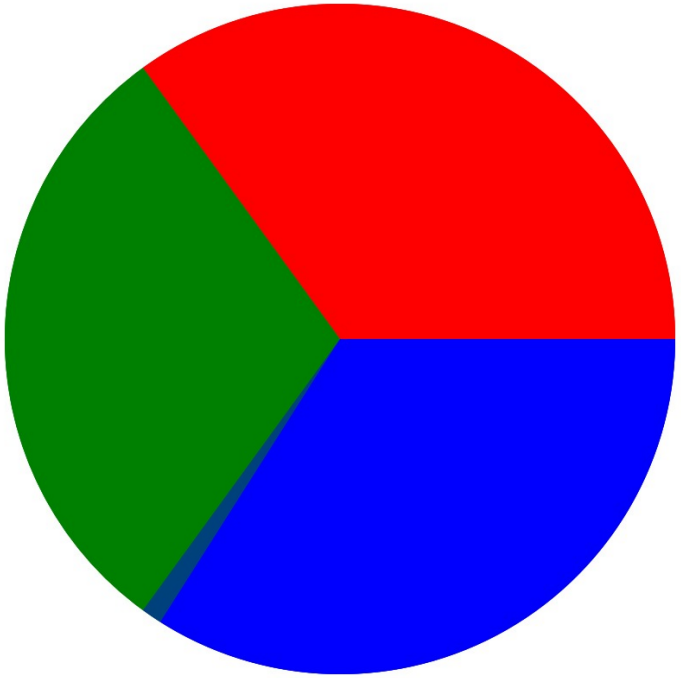
Anomaly Traffic Sample

Let’s subtract – what is: *right* – *left*?

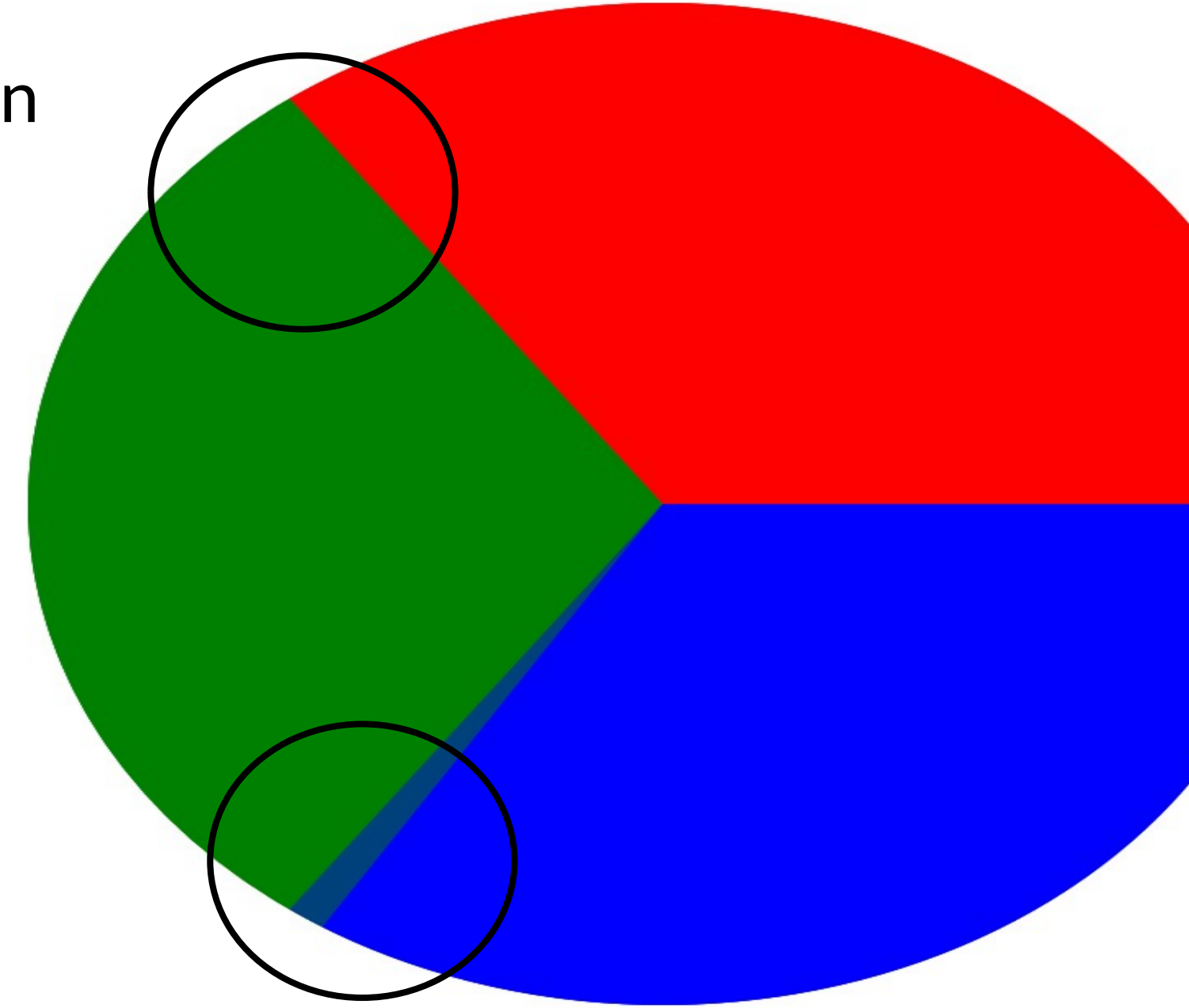




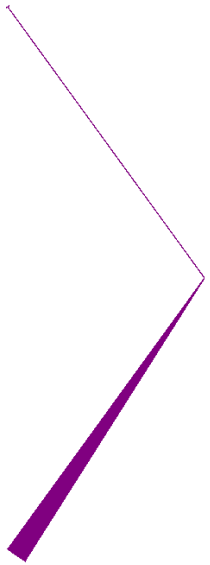
This is right overlapped with left – spot the difference!



If you look carefully…
you can see discoloration

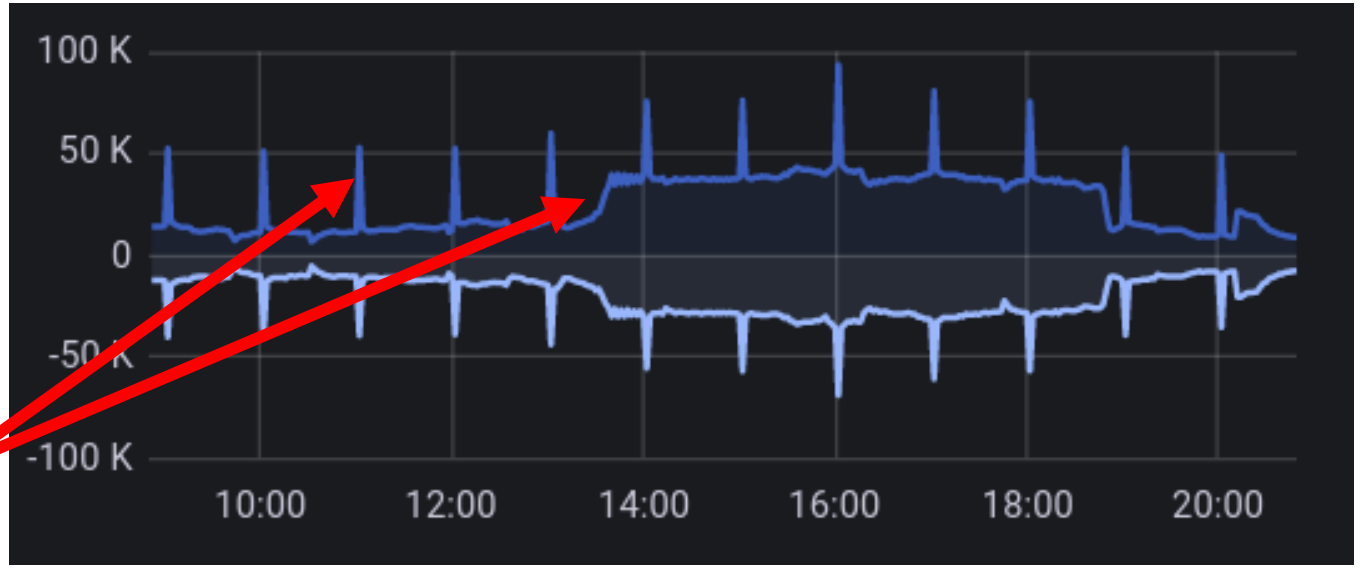


This is the real *right - left*



When we remove “normal”
we are **left with just the abnormal**

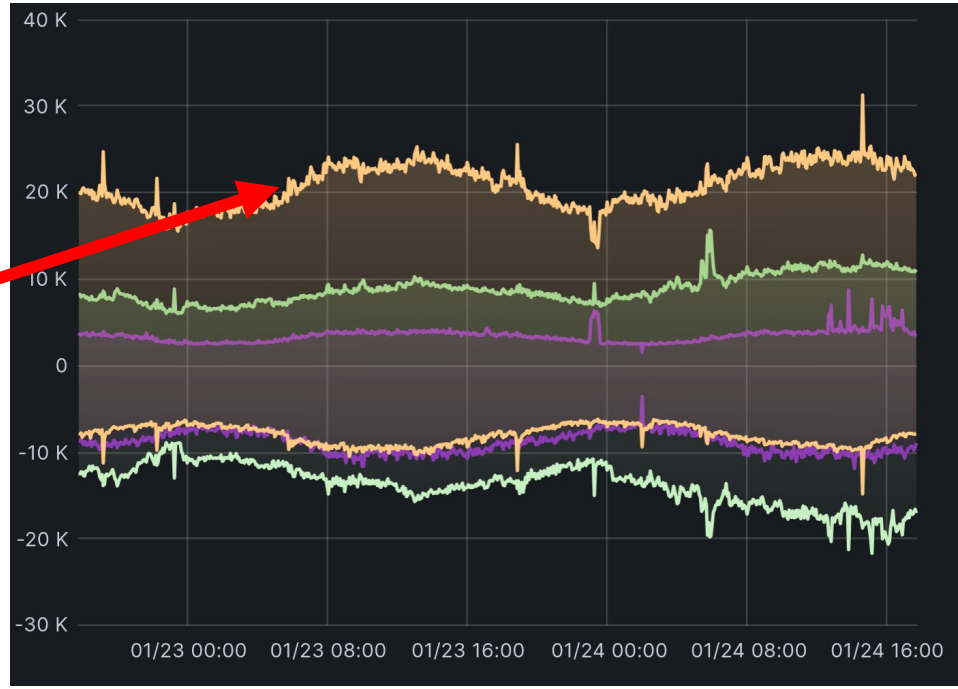
Bilateral Comparisons
A **specific problem space**



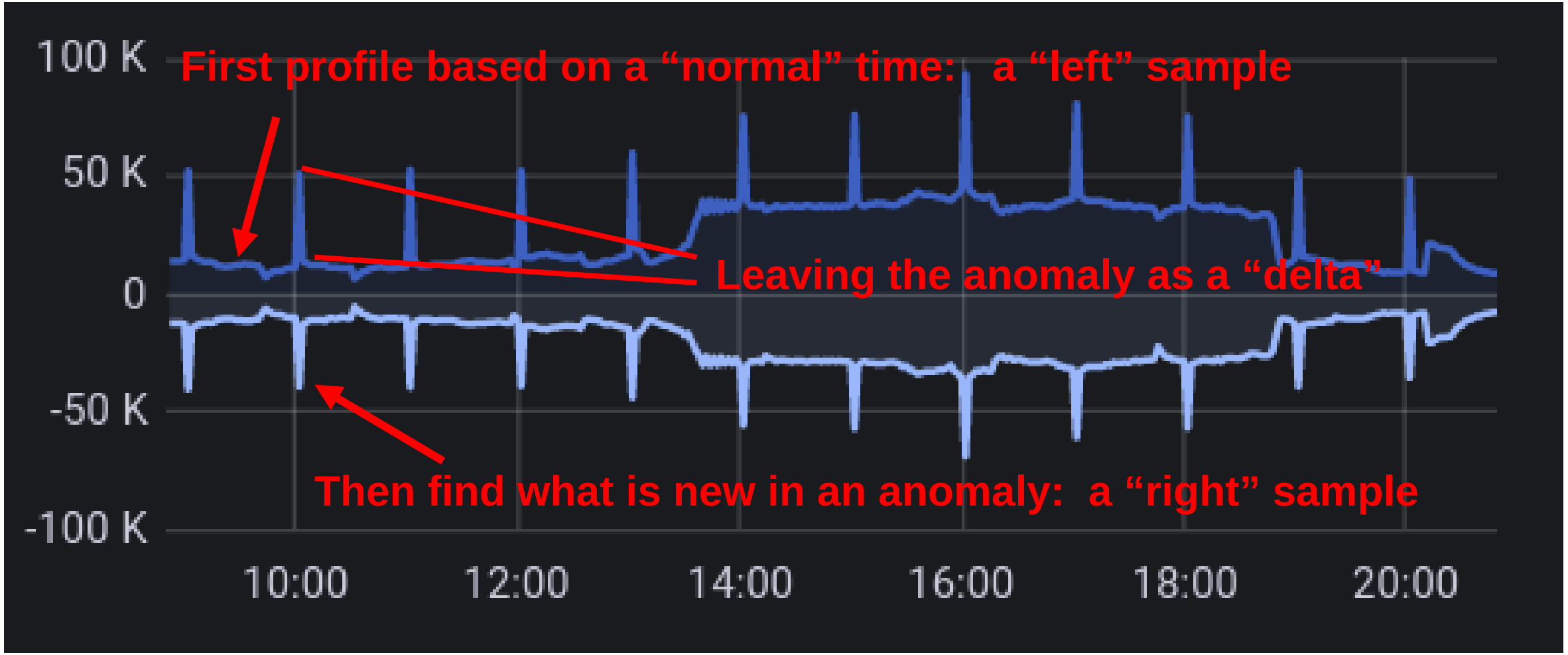
<u>Important considerations</u>

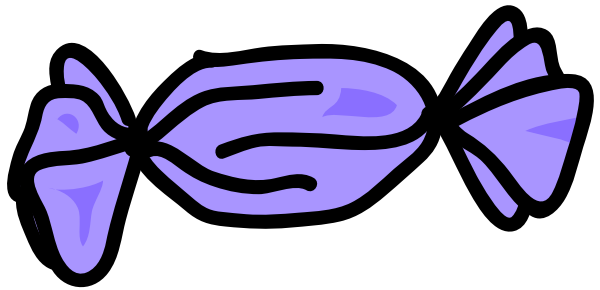
- Major shifts in traffic are from a **single cause**
  - Something has clearly **changed**. *What?*

- *"More of the same" r*amp ups are not an "anomaly"
  - AKA diurnal patterns are "normal"



Terminology: left = “normal” right = “anomaly”

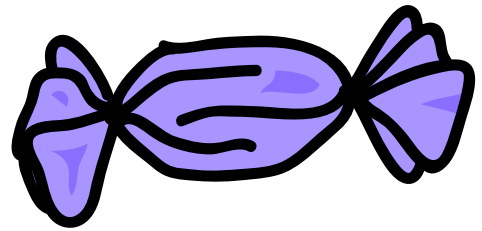




INTRODUCING “TRAFFIC-TAFFY”

The toolset that helps automate bilateral traffic analysis

Introducing traffic-taffy

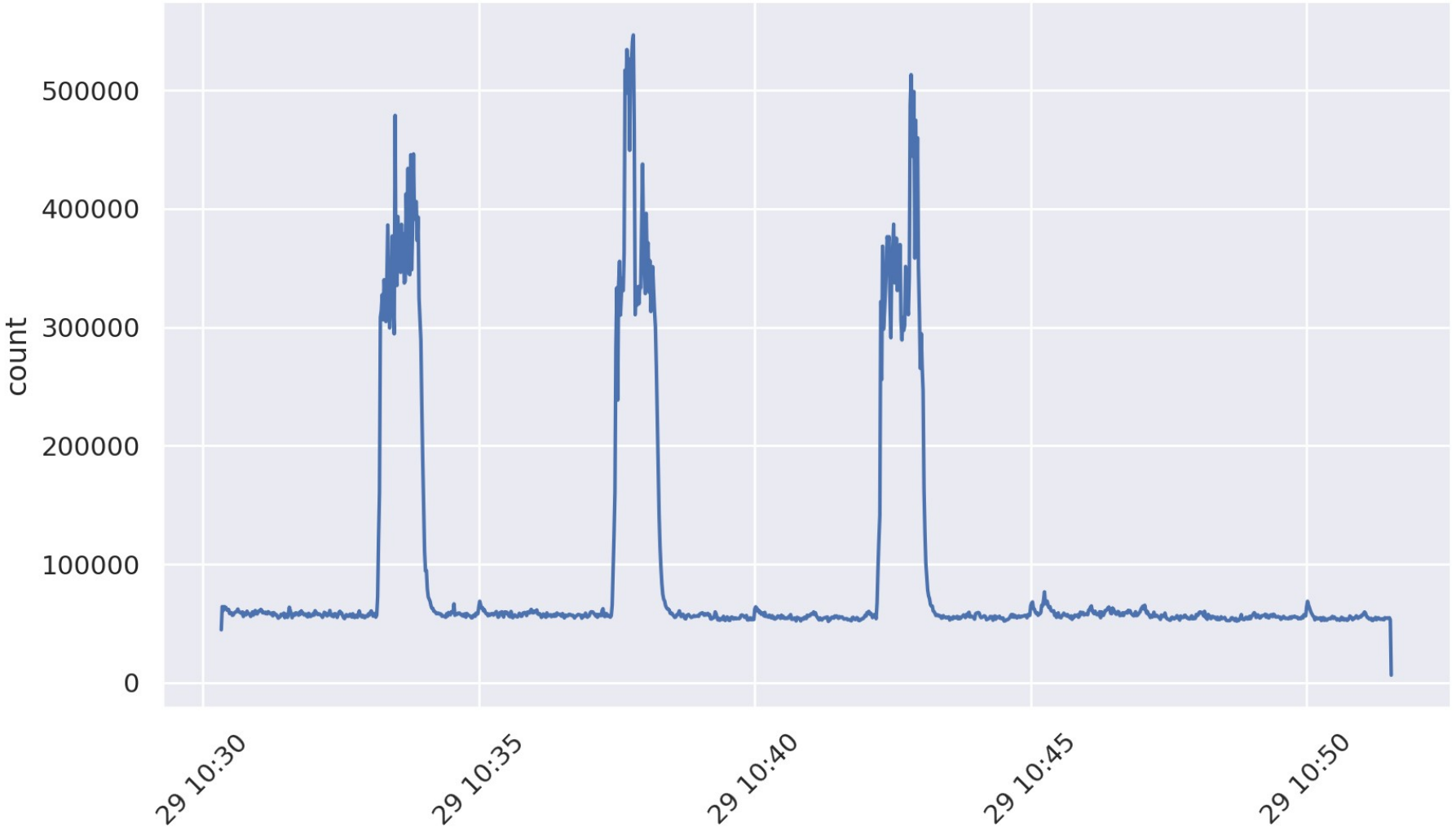


- Approach:
  - Perform deep packet inspection of a **base-line**
  - Perform deep packet inspection of an **anomaly**
  - Compare levels for ***each value*** of ***each protocol field***
  - **Sort, Filter and Report** based on findings
- Some of the tools:
  - taffy-dissect: enumerates field counts in a pcap file
  - taffy-compare: compares one file/time-range against another
  - taffy-graph: graphs enumerated fields in pcap files
- Easy to install: ***pip install traffic-taffy***

Let’s go back to studying traffic anomalies

Using bilateral comparisons!

Case Study 1: three large bumps seen at b.root-servers.net



Dataset:

- Three 5x spikes
- At 1 anycast site
- What are they?

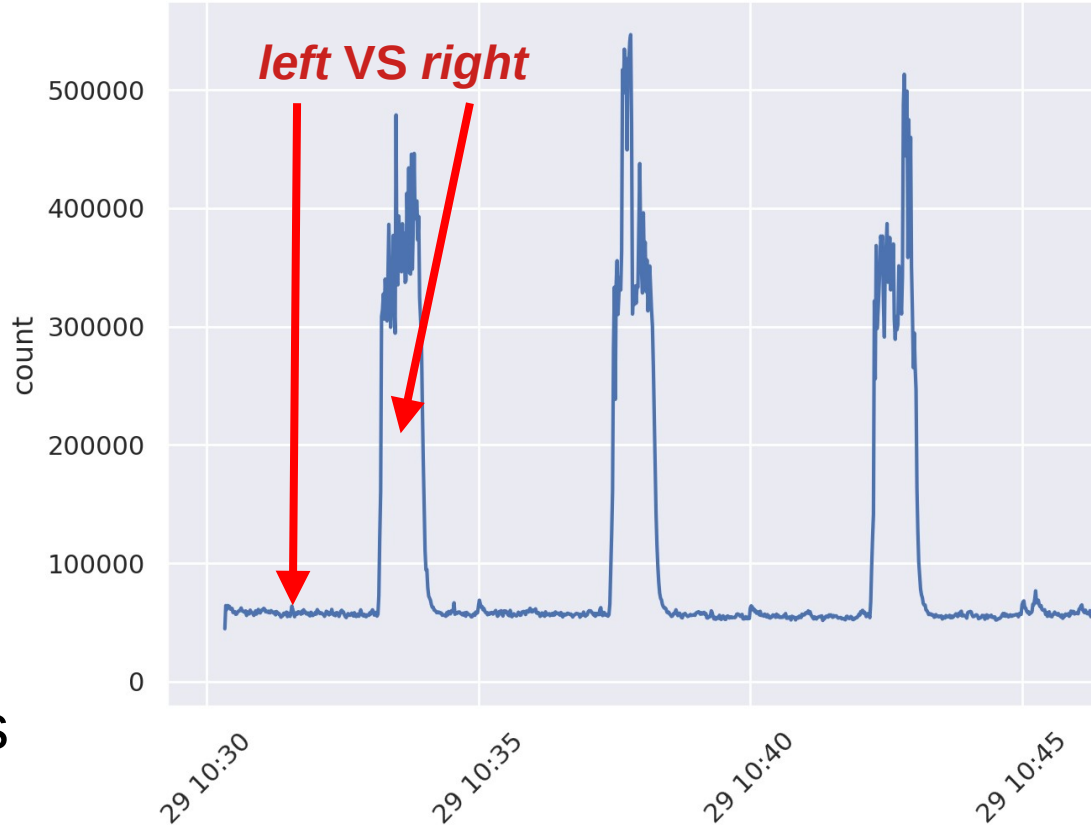
- Can we find the root cause?

Graph produced with taffy-graph

taffy-compare: find differences between points in time

Taffy-compare:
- Takes PCAP data from two points in time
- Uses the *left* side as a “normal” profile
- Identifies major shifts in the *right* side

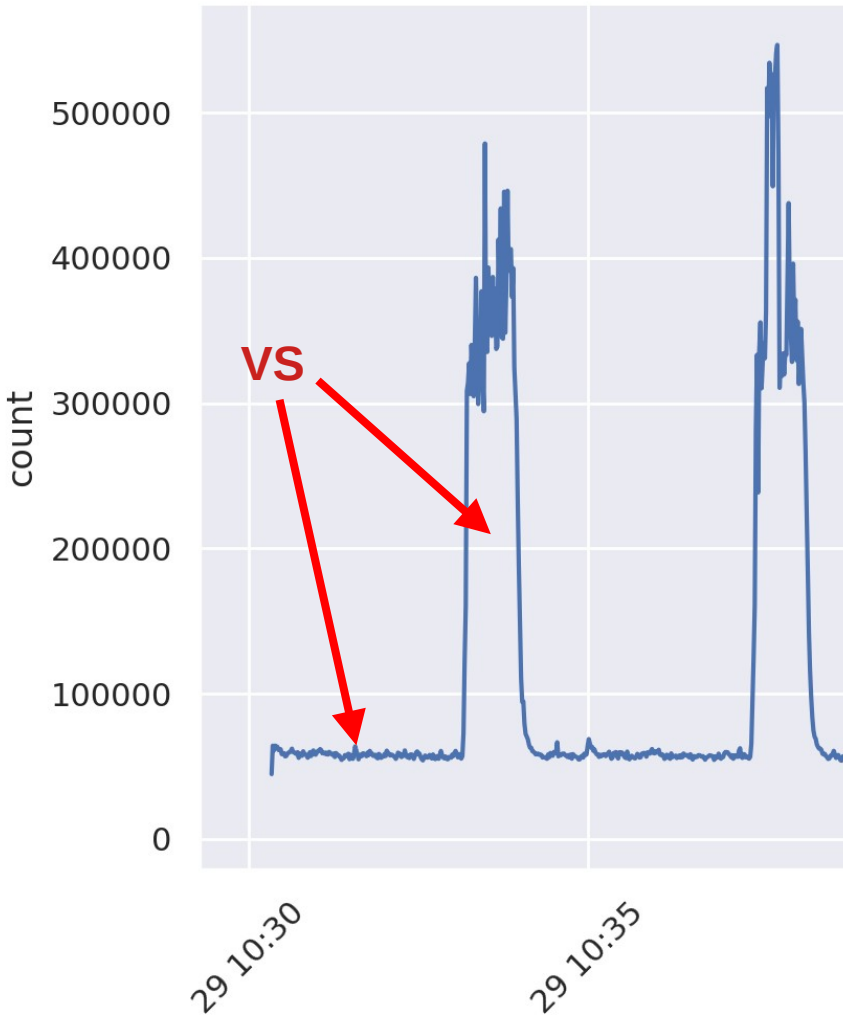
Output: colorized results to the console
- Total counts per protocol field
  - Left and right
- Percentage of traffic for each field value
- Deltas between both values and percentages
- All filterable by threshold values



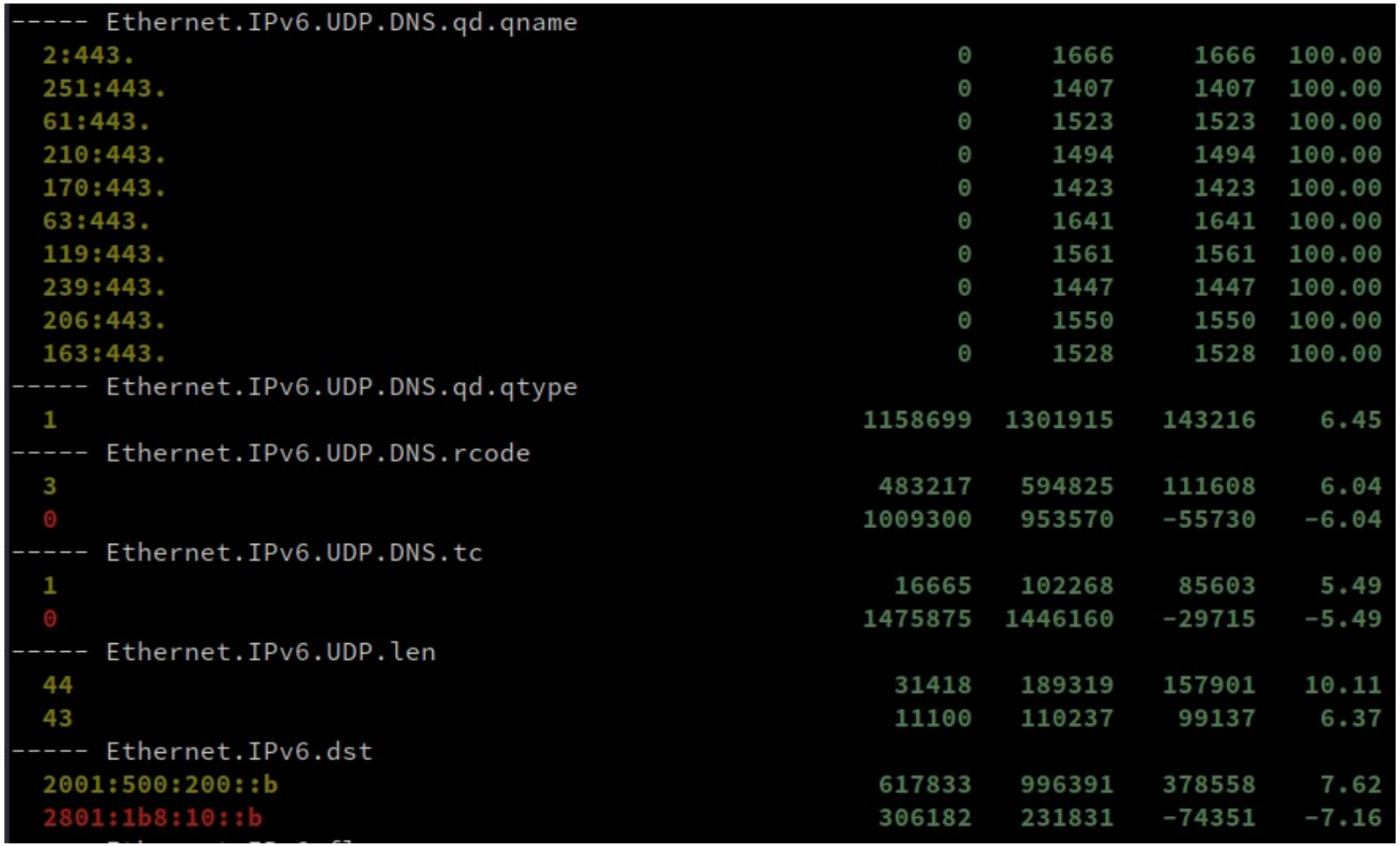
Value	Left	Right	Delta	Left %	Right %	Delta-%
----- Ethernet.IP.UDP.DNS.cd						
0	1961290	2668655	707365	39.43	54.29	14.86
1	3012629	2246678	-765951	60.57	45.71	-14.86

taffy-compare: find differences between points in time

```
----- Ethernet.IPv6.UDP.DNS.qd.qname
  2:443.                                    0     1666     1666  100.00
  251:443.                                  0     1407     1407  100.00
  61:443.                                   0     1523     1523  100.00
  210:443.                                  0     1494     1494  100.00
  170:443.                                  0     1423     1423  100.00
  63:443.                                   0     1641     1641  100.00
  119:443.                                  0     1561     1561  100.00
  239:443.                                  0     1447     1447  100.00
  206:443.                                  0     1550     1550  100.00
  163:443.                                  0     1528     1528  100.00
----- Ethernet.IPv6.UDP.DNS.qd.qtype
  1                                   1158699  1301915   143216    6.45
----- Ethernet.IPv6.UDP.DNS.rcode
  3                                    483217   594825   111608    6.04
  0                                   1009300   953570   -55730   -6.04
----- Ethernet.IPv6.UDP.DNS.tc
  1                                     16665   102268    85603    5.49
  0                                   1475875  1446160   -29715   -5.49
----- Ethernet.IPv6.UDP.len
  44                                    31418   189319   157901   10.11
  43                                    11100   110237    99137    6.37
----- Ethernet.IPv6.dst
  2001:500:200::b                      617833   996391   378558    7.62
  2801:1b8:10::b                       306182   231831   -74351   -7.16
```



taffy-compare example: colored console differences



```
----- Ethernet.IPv6.UDP.DNS.qd.qname
  2:443.                                    0     1666     1666  100.00
  251:443.                                  0     1407     1407  100.00
  61:443.                                   0     1523     1523  100.00
  210:443.                                  0     1494     1494  100.00
  170:443.                                  0     1423     1423  100.00
  63:443.                                   0     1641     1641  100.00
  119:443.                                  0     1561     1561  100.00
  239:443.                                  0     1447     1447  100.00
  206:443.                                  0     1550     1550  100.00
  163:443.                                  0     1528     1528  100.00
----- Ethernet.IPv6.UDP.DNS.qd.qtype
  1                                   1158699  1301915   143216    6.45
----- Ethernet.IPv6.UDP.DNS.rcode
  3                                    483217   594825   111608    6.04
  0                                   1009300   953570   -55730   -6.04
----- Ethernet.IPv6.UDP.DNS.tc
  1                                     16665   102268    85603    5.49
  0                                   1475875  1446160   -29715   -5.49
----- Ethernet.IPv6.UDP.len
  44                                    31418   189319   157901   10.11
  43                                    11100   110237    99137    6.37
----- Ethernet.IPv6.dst
  2001:500:200::b                      617833   996391   378558    7.62
  2801:1b8:10::b                       306182   231831   -74351   -7.16
```

Leaked docker port mappings

For A records

We return more NXdomains

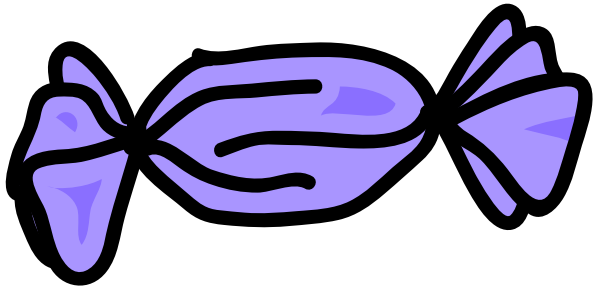
RRL limits hit (increasing TC)

Heading to our older IPv6 address

Important note: I did not pick these fields to study – the tool did!

Warning… this is easy to recreate

Try it!



- Please test it!
  - https://traffic-taffy.readthedocs.io/
    - Multiple (longer) video presentations
    - The DNS-OARC video has greater usage detail
  - **pip install traffic-taffy**

- Thank you to the **Comcast innovation fund** for sponsoring this work!

Wes Hardaker <hardaker@isi.edu>