Meeting with Cisco Systems
San Jose, CA, 5th Dec 2017

BGPStream and OpenBMP

Alberto Dainotti, Alistair King
alberto@caida.org, alistair@caida.org

caida

Center for Applied Internet Data Analysis
University of California, San Diego

AGENDA

20 min + Q/A

- **BGPStream**
- **Collab w/ Cisco OpenBMP group**
- ***V2* & Applications**
- **Future Work**

BGP STREAM

Game-changing technology to enable BGP analytics

- Open Source Software **APIs** for historical and <u>*live*</u> BGP data analysis
  - *Python, C, Command Line tools, …*
- Accelerates existing and enables fundamentally new analytic capabilities
- Design goals:
  - Efficiently deal with large amounts of distributed BGP data
  - Offer a time-ordered data stream of data from heterogeneous sources
  - Support near-realtime data processing
  - Target a broad range of applications and users
  - Scalable (*e.g., use Apache Spark to crunch billions of updates*)
  - Easily extensible
  - Simple API
  - Facilitates reproducibility and repeatability

BGP STREAM

For/with the operator + research community

- **RIPE 70** and tech report - *May 2015*
- Version 1 and **IETF 94** Tech Plenary - *Nov 2015*
- Version 1.1 and **BGP Hackathon** - *Feb 2016*
- **NANOG 66** - *Feb 2016*
- **ACM IMC** paper - *Nov 2016*
- **IRTF** ANRP award - *Jan 2017*
- **IETF 98** - *Mar 2017*
- Version 2.0 beta1 - *Today!*


BGPStream: A Software Framework for Live and Historical BGP Data Analysis

Chiara Orsini [1], Alistair King [1], Danilo Giordano [2], Vasileios Giotsas [1], Alberto Dainotti [1]
[1] CAIDA, UC San Diego
[2] Politecnico di Torino

Orsini et al.
"BGPStream: a software framework for live and historical BGP data analysis"
ACM SIGCOMM IMC 2016

PEOPLE USE IT

hackathons, papers, net admins, …

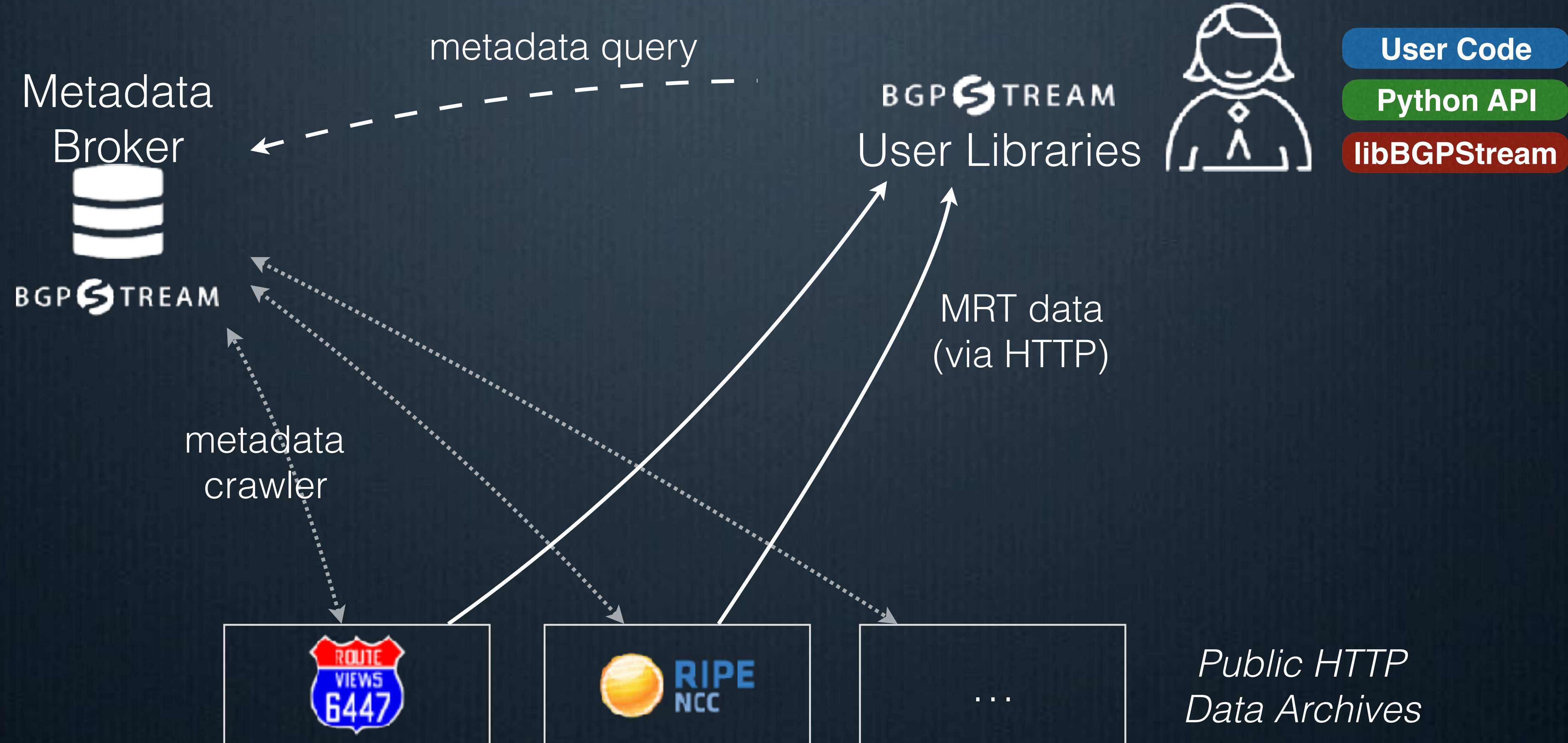
- Various hackathons: *NANOG, RIPE, …*
- *github.com/caida/bgpstream*
    - *some significant pull requests from 3rd parties*

- Selected papers:
    - ***Counter-RAPTOR: Safeguarding Tor Against Active Routing Attacks*** *[SP'17] - Sun et al.*
    - ***I-Seismograph: Observing, Measuring, and Analyzing Internet Earthquakes*** *[ToN'17] - Zhang et al.*
    - ***Sibyl: A Practical Internet Route Oracle.*** *[NSDI'16] - Cunha et al.*
    - ***PathCache: A Path Prediction Toolkit.*** *[SIGCOMM'16] - Singh et al.*

State of the Art?

```
wget http://archive.org/xyz/abc/file.mrt
bgpdump -m file.mrt | my_parser.py
```

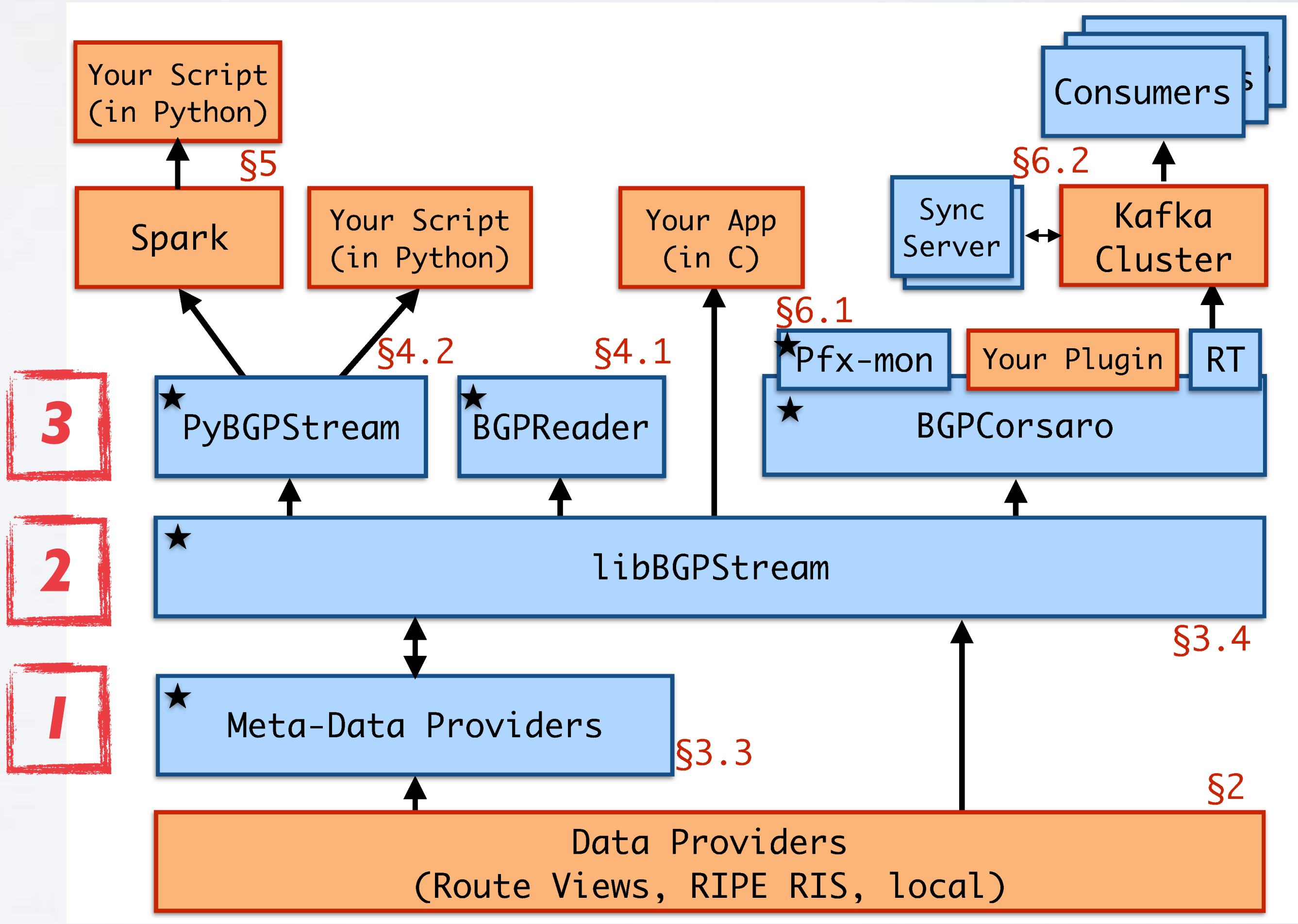
The BGPStream Framework



BGPSTREAM

bgpstream.caida.org

1. *A web service ("BGPStream Broker")*
    - enables SIMPLE **access** to many heterogeneous BGP sources
2. *LibBGPStream:*
    - Acquires the data and provides to upper layers a realtime stream of BGP data
    - makes it SIMPLE to **process** data from many heterogeneous BGP sources
3. Command-line tools and APIs in C and *Python*



NO MANUAL DOWNLOADS

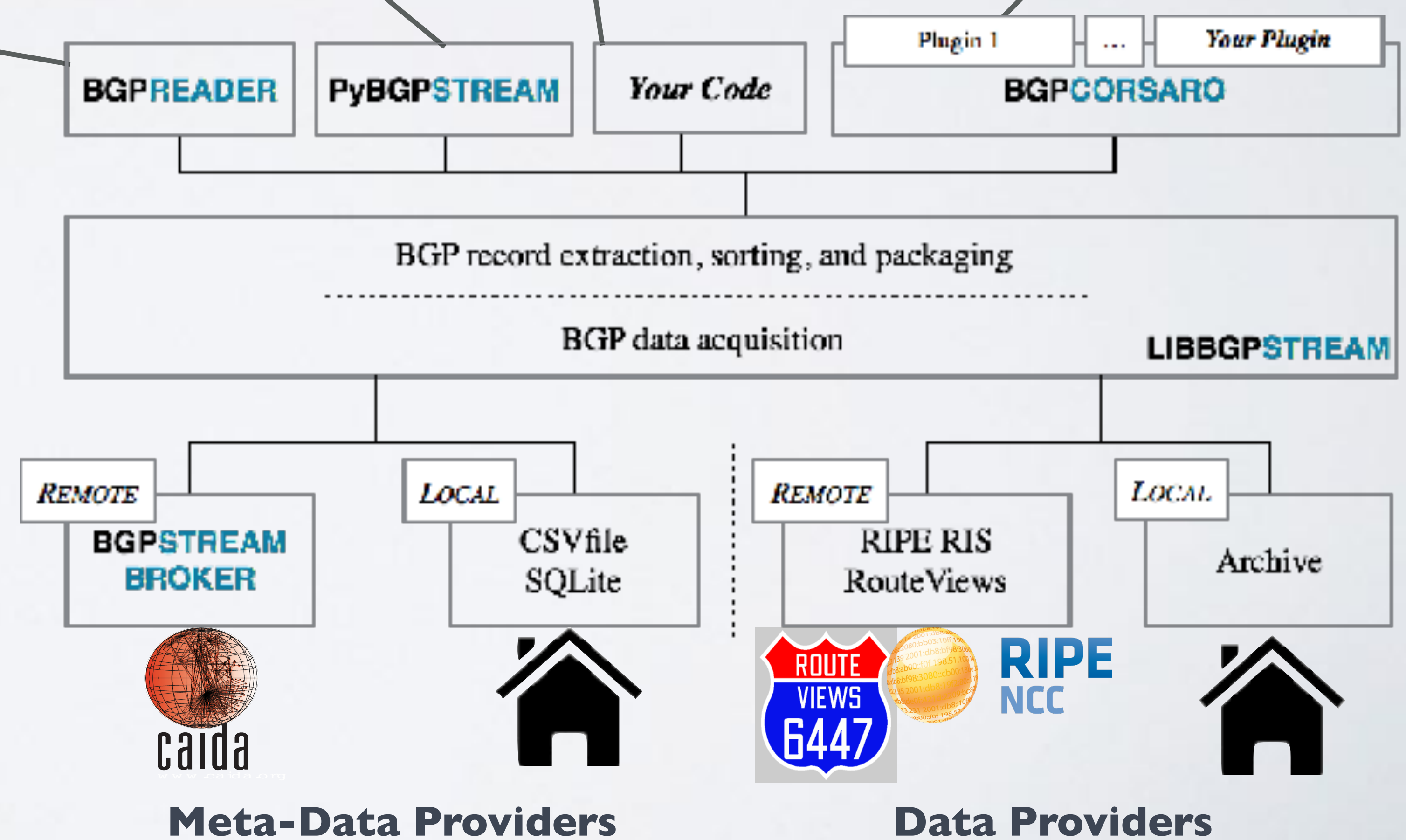
libBGPStream talks to the broker and gets the data

```
bgpstream_add_filter(bs, BGPSTREAM_FILTER_TYPE_COLLECTOR, "rrc06");
bgpstream_add_filter(bs, BGPSTREAM_FILTER_TYPE_COLLECTOR, "route-views.jinx");
bgpstream_add_filter(bs, BGPSTREAM_FILTER_TYPE_RECORD_TYPE, "updates");
bgpstream_add_interval_filter(bs, 1286705410, 1286709071);
```

```
stream.add_filter('record-type', 'ribs')
stream.add_filter('collector', 'route-views.sfmix')
stream.add_interval_filter(1445306400,1445306402)
```

```
$ bgpcorsaro -w 1445306400,1445306402 -p ris
```

```
$ bgpreader -w 1445306400,1445306402 -c route-views.sfmix -t updates
```



Experiments can be easily reproduced: a script defines the (public) data used

Center for Applied Internet Data Analysis
University of California San Diego

BGPREADER

command-line tool for ASCII output w/ filters

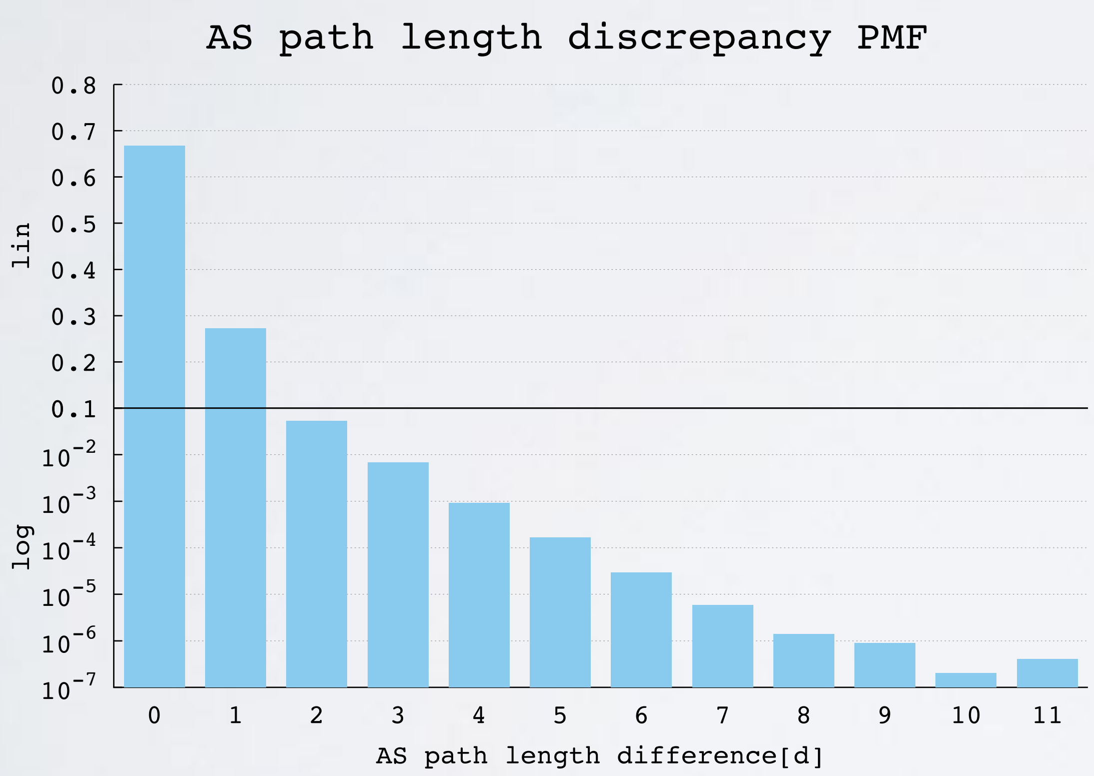
```
$ bgpreader -w 1445306400,1445306402 -c route-views.sfmix
R|B|1445306400|routeviews|route-views.sfmix
R|R|1445306400|routeviews|route-views.sfmix|32354|206.197.187.5|1.0.0.0/24|206.197.187.5|32354 15169|15169|||
...
R|R|1445306401|routeviews|route-views.sfmix|14061|2001:504:30::ba01:4061:1|2c0f:ffd8::/32|
2001:504:30::ba01:4061:1|14061 1299 33762|33762|1299:30000||
R|R|1445306401|routeviews|route-views.sfmix|32354|2001:504:30::ba03:2354:1|2c0f:ffd8::/32|
2001:504:30::ba00:6939:1|32354 6939 37105 33762|33762|||
R|R|1445306401|routeviews|route-views.sfmix|14061|2001:504:30::ba01:4061:1|3803:b600::/32|
2001:504:30::ba01:4061:1|14061 2914 3549 27751|27751|2914:420 2914:1008 2914:2000 2914:3000||
R|E|1445306401|routeviews|route-views.sfmix
U|A|1445306401|routeviews|route-views.sfmix|32354|2001:504:30::ba03:2354:1|2402:ef35::/32|
2001:504:30::ba03:2354:1|32354 6939 6453 4755 7633|7633|||
U|A|1445306401|routeviews|route-views.sfmix|14061|2001:504:30::ba01:4061:1|2a02:158:200::/39|
2001:504:30::ba01:4061:1|14061 2914 44946|44946|2914:410 2914:1201 2914:2202 2914:3200||
...
```

PYBGPSTREAM

BGP STREAM

Example: studying AS path inflation

How many AS paths are longer than the shortest path between two ASes due to routing policies? (directly correlates to the increase in BGP convergence time)



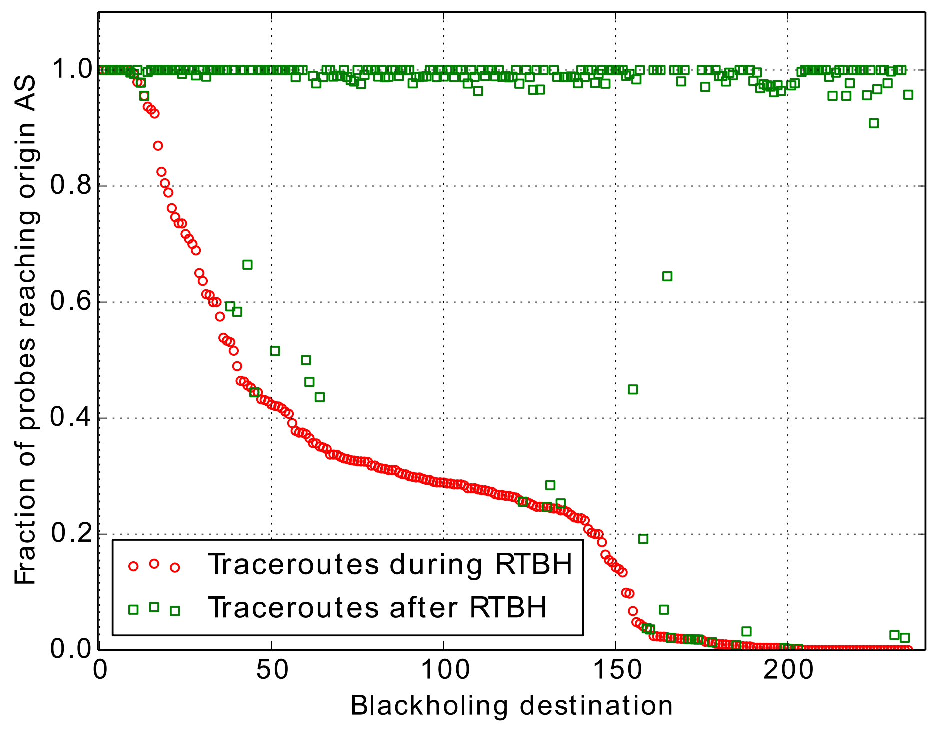

```python
from _pybgpstream import BGPStream, BGPRecord, BGPElem
from collections import defaultdict
from itertools import groupby
import networkx as nx

stream = BGPStream()
as_graph = nx.Graph()
rec = BGPRecord()
bgp_lens = defaultdict(lambda: defaultdict(lambda: None))
stream.add_filter('record-type','ribs')
stream.add_interval_filter(1438415400,1438416600)
stream.start()

while(stream.get_next_record(rec)):
    elem = rec.get_next_elem()
    while(elem):
        monitor = str(elem.peer_asn)
        hops = [k for k, g in groupby(elem.fields['as-path'].split(" "))]
        if len(hops) > 1 and hops[0] == monitor:
            origin = hops[-1]
            for i in range(0,len(hops)-1):
                as_graph.add_edge(hops[i],hops[i+1])
            bgp_lens[monitor][origin] = \
                min(filter(bool,[bgp_lens[monitor][origin],len(hops)]))
        elem = rec.get_next_elem()
for monitor in bgp_lens:
    for origin in bgp_lens[monitor]:
        nxlen = len(nx.shortest_path(as_graph, monitor, origin))
        print monitor, origin, bgp_lens[monitor][origin], nxlen
```

30 LINES OF PYTHON CODE

PYBGPSTREAM

Example: synchronizing with active measurements

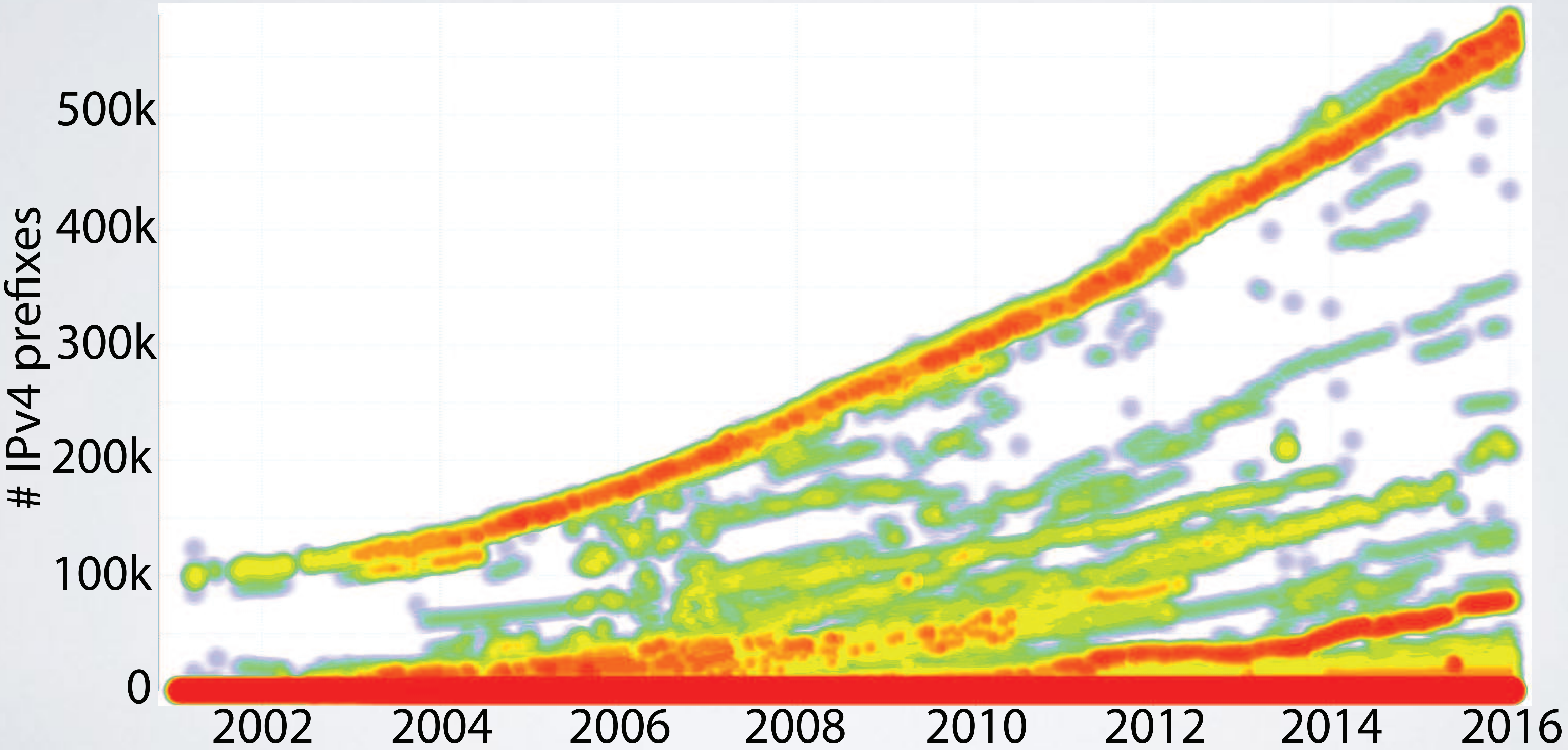
- We monitor **community-based black-holing**
  - Victim of DoS attack announces prefix with special community attribute to request that neighbors drop traffic
- We trigger traceroutes to characterize the black-holing event (using 50-100 probes per event)
  - probed 253 victims (90-95% of black-holing events) while black-holing in effect
- ***Combined passive control-plane and active data-plane measurements to capture and investigate transient routing policies***



BIG DATA IN A FEW LINES

44Billion BGPElems processed w/ Spark + PyBGPStream

Heatmap of routing table size *(color reflects # peers)*

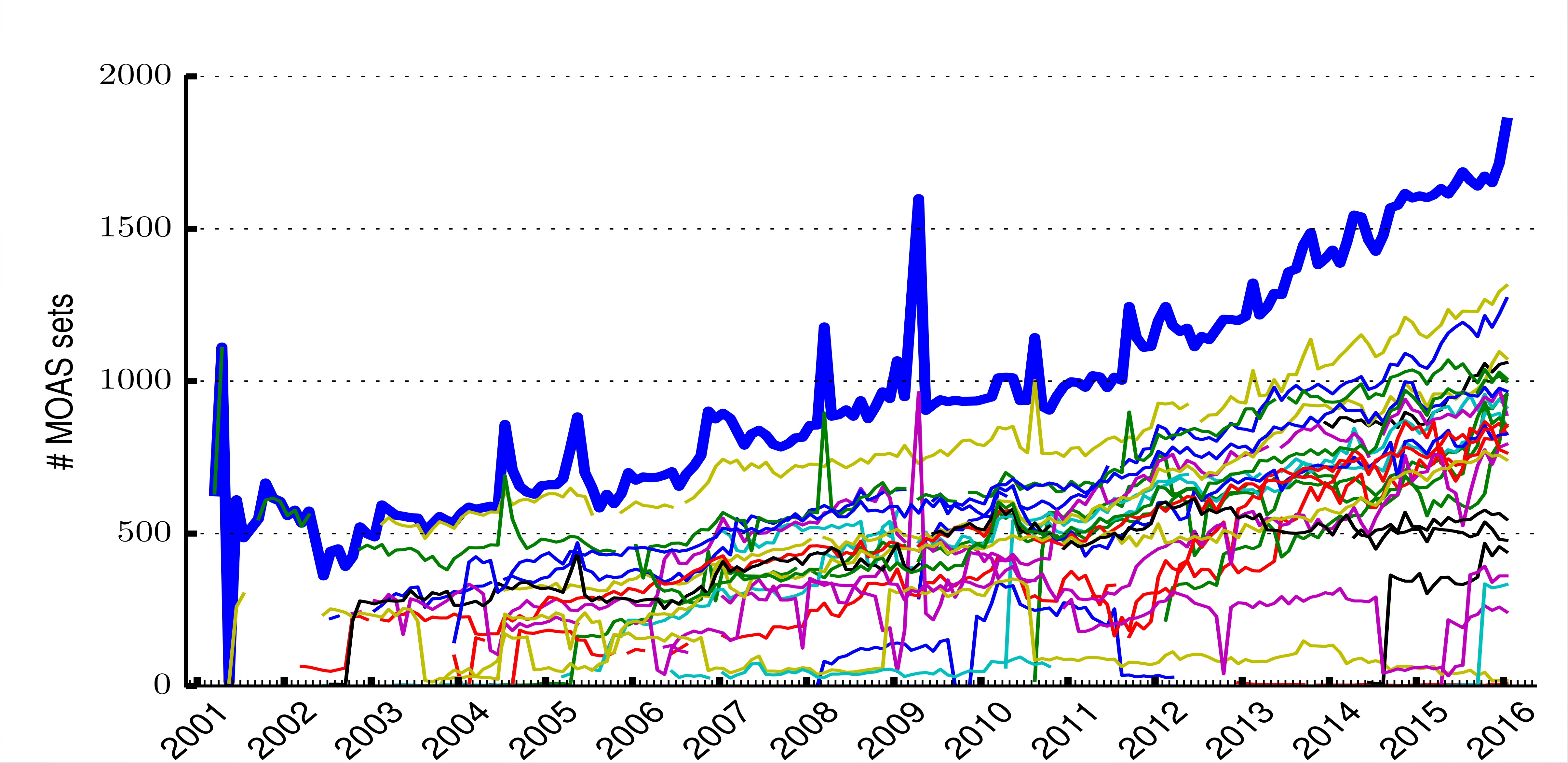


Code at www.caida.org/publications/papers/2016/bgpstream/supplemental

BIG DATA IN A FEW LINES

44Billion BGPElems processed w/ Spark + PyBGPStream

MOAS per collector and aggregate



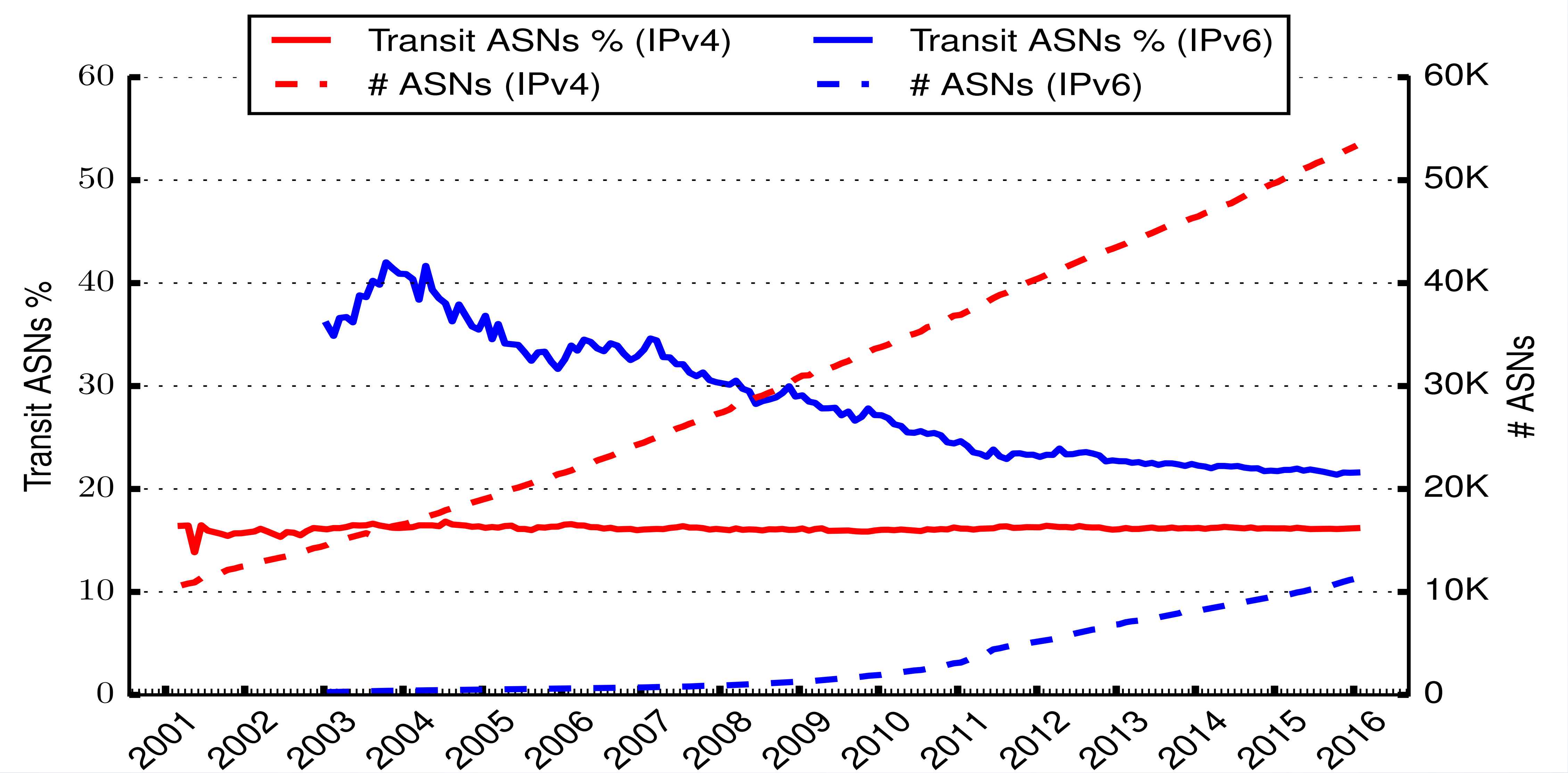
Center for Applied Internet Data Analysis
University of California San Diego

Code at www.caida.org/publications/papers/2016/bgpstream/supplemental

BIG DATA IN A FEW LINES

44Billion BGPElems processed w/ Spark + PyBGPStream

Transit ASes



Center for Applied Internet Data Analysis
University of California San Diego

Code at www.caida.org/publications/papers/2016/bgpstream/supplemental

COLLAB W/ CISCO

COLLAB W/ CISCO

Tasks

- **1: Native OpenBMP Support in BGPStream**
  - Rearchitect BGPStream
  - Add support for Kafka Encapsulation
  - Deserialize BMP data into BGPStream

- **2: Distribute BMP data**
  - Run a public collector
  - Coordinate w/ RouteViews
  - Tutorial on how to use BGPStream with your BMP router/collector

- **3: Cooperate w/ OpenBMP group**
  - Creatd LibParseBGP
  - New OpenBMP features: NAT/PAT, Router connection rate limiting
  - Bi-dir feedback and testing
  - Dissemination of results and work in progress

TASK 1

Native OpenBMP Support in BGPStream

- Pre-existing conditions: *BGPStream v1.1*
  - No BMP support
  - entirely MRT-based
  - RouteViews, RIPE RIS, … → ~20 min delay! We want real live streaming

TASK 1

Native OpenBMP Support in BGPStream

- **Task 1.1 - Rearchitect BGPStream**
  - LibBGPStream had a monolithic architecture. We turned it into modular *(object-oriented C implementation)*

BEFORE

- Data Interfaces:
  - Broker; Single file; CSV
- Single format
  - MRT
- Single transport
  - Dump files (http/FS)

NOW

- Data Interfaces:
  - Broker; Single file; CSV; **Kafka**
- Formats:
  - MRT; **BMP**; **...**
- Transports:
  - Dump files; **Kafka**

TASK 1

Native OpenBMP Support in BGPStream

- **Task 1.2 - Add support for Kafka encapsulation**
  - Based on Librdkafka *(https://github.com/edenhill/librdkafka)*

BEFORE	***NOW***
• Data Interfaces: - Broker; Single file; CSV • Single format - MRT • Single transport - Dump files (http/FS)	• Data Interfaces: - Broker; Single file; CSV; **Kafka** • Formats: - MRT; **BMP**; **...** • Transports: - Dump files; **Kafka**

TASK 1

Native OpenBMP Support in BGPStream

- **Task 1.3 - Deserialize BMP data into BGPStream**
  - *Deserialization*
    - -Wrote C code inspired by OpenBMP's C++
    - -Created a standalone library ***libParseBGP (Task 3.1)*** *https://github.com/CAIDA/libparsebgp*
      - Good engineering practice (cleanness, modularity, …)
      - Provide the community with a BMP/MRT/BGP parsing library
  - *New format for encapsulation of BMP data*
    - *OpenBMP currently uses an ascii encapsulation*
    - Needed timestamps, info about router and collector, …
    - binary/ascii

LIBPARSEBGP

https://github.com/CAIDA/libparsebgp

- Parses *BGP, BMP, MRT* from a buffer into a *C* structure
- Parsed data is *"close to the RFC"*.
  - *library doesn't assume anything about how you will use it -- e.g., addresses are left as 4, 16 byte network-byte ordered values*
- Data is parsed into a reusable structure
  - *dynamic memory inside the structure is reused between parses -- avoids free/mallocs and drastically improves performance*
- Many path attributes supported
  - *https://github.com/CAIDA/libparsebgp/blob/master/lib/bgp/parsebgp_bgp_update.h#L225*
- Support for selectively parsing features
  - *clients who only need specific features, e.g., AS Path, Community attributes, don't need to parse the entire message*

TASK 2

Distribute BMP data

- **Task 2.1 - Run a public collector**
- Nobody had experience in operating a <u>public</u> BMP collector
- CAIDA's Public BGPStream OpenBMP Collector
  - bmp.bgpstream.caida.org:9092
- Already providing feeds
  - 1 Cisco router, 1 Cisco peer (*AS11017 - CSN*) LIVE

- **Task 2.2 - Coordinate with RouteViews**
- Operated in collaboration with RouteViews
  - 1 RV router, 3 RV peers (operational routers from *Level3, HE, AT&T*) LIVE
  - Work in progress, slowly will add more and more
  - *Lesson learned:* non negligible load on router due to BMP

TASK 2

Distribute BMP data

- **Task 2.3 - Tutorial to use BGPStream w/ your router/ collector**
  - Leverages the OpenBMP docker container
    - *https://bgpstream.caida.org/v2-beta*
    - *live demo now*

- Shows how to analyze your private router's BMP feed from pyBGPStream

TASK 3

Cooperate w/ OpenBMP group

- **Great teamwork — *Thank you Serpil+Tim!***
- libParseBGP *(see previous slides)*
- Contributions to OpenBMP
  - NAT/PAT support
  - Router connection rate limiting
  - New (optional?) OpenBMP encapsulation format
  - Minor bug fixes
- Trained 2 UC San Diego master students

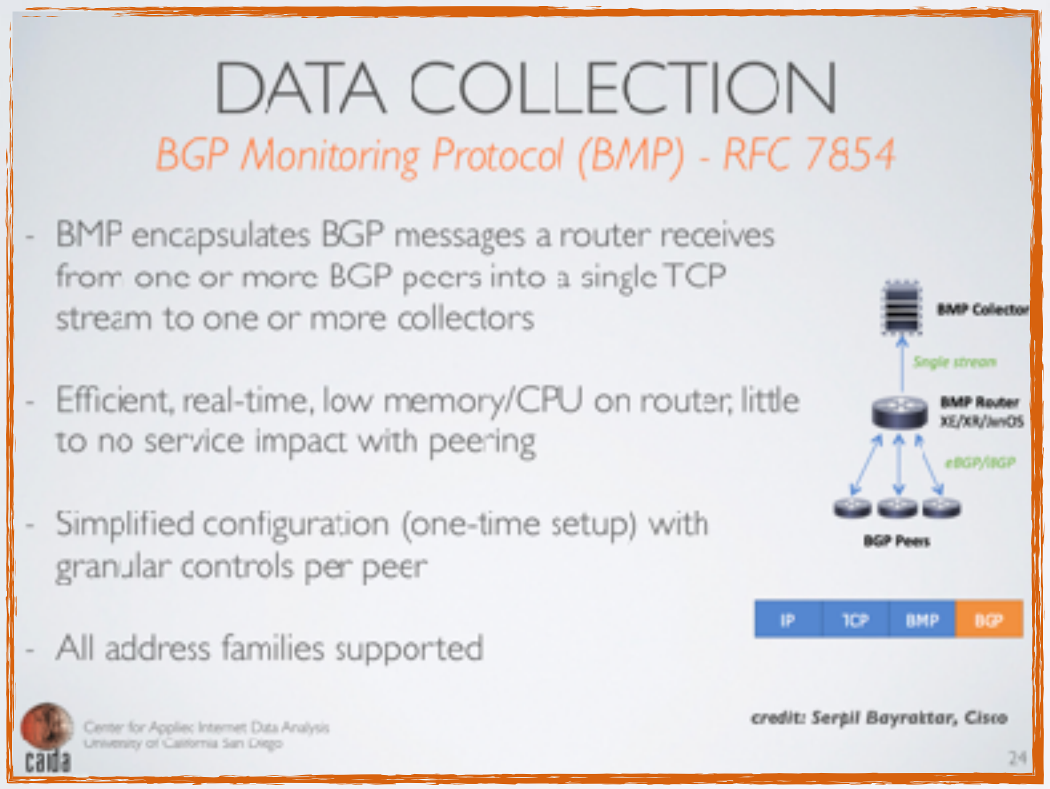
Ojas Gupta

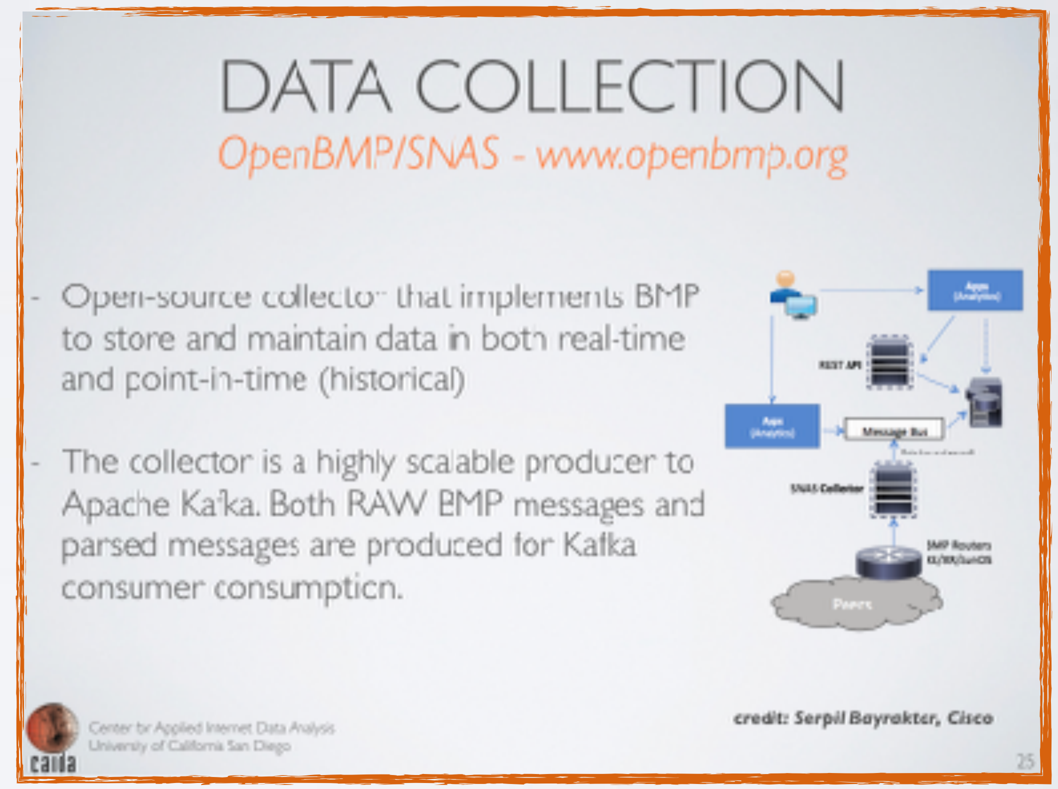
Induja Sreekanthan

TASK 3

Cooperate w/ OpenBMP group

- **Dissemination of results & work in progress**
- IRTF and IETF '98 - (*irtf-open and rtgwg*)
- TMA PhD School, Dublin, Ireland (Lecture + Lab ~40 PhD students)





- Tim's Keynote at SIGCOMM BigData Workshop *(BIG-DAMA)*
- *v2* ml-announcements and presentations *(NANOG, IETF, ACM conferences, …)* **todo**

V2-BETA1 AND APPS

V2.0 BETA 1

https://bgpstream.caida.org/v2-beta

- Public BMP feed *bmp.bgpstream.caida.org:9092*
- BGPStream apps can read BMP
- Projects ready to use it
  - ***IODA*** *[IMC'16 and others] - 24/7 Internet Outage detection*
    - *speeding up BGP detection — dashboards at* ***ioda.caida.org***
  - ***ARTEMIS*** *[wip ToN] - Self-operated BGP prefix hijacking detection and mitigation*
    - *open source framework*
  - ***SWIFT*** *[SIGCOMM'17] - Fast rerouting upon remote outages* ***swift.ethz.ch***
    - *downloadable demo*
  - ***SuperSWIFT*** *[wip SIGCOMM] - P4 version of Swift. Collab with Internet2*
    - *analysis of data vs control plane*

FUTURE WORK

V2 RELEASE

New features in addition to BMP

- BGPStream v2.0 expected to be released in **Feb 2018**
- New **license: BSD**
- v2 features:
  - *RIPE RIS streaming support* **todo**
  - *RPKI validation (RTRlib)* **todo**
  - *Broker support for public BGPStream BMP feed* **todo**
  - *Local (optional) caching of dump files*
  - *New high-level Python API*
  - *New filter interface with a "BPF-like" syntax (hackathon contribution)*
  - *Performance improvements (new MRT parser, better resource management, …)*
  - *Bugfixes*

BGP ANALYTICS

Prefix hijacking detection and more

- Leverages BGPStream
- Combines *control-plane* and *data-plane* measurements
  - detects interesting BGP events *(e.g., MOAS, new edges in the topology, …)*
  - and triggers traceroute measurements from Ark/RIPE probes
- Classifies events and generates alerts
- Visualization dashboard to analyze the events
- Based on NSF funding ending soon

THANKS

bgpstream.caida.org

bgpstream.caida.org/v2-beta

alberto@caida.org