A software framework for **historical** and **live** BGP data analysis

- 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
- Easily extensible
**PYBGPSTREAM**

*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 _pybgs_stream import BGPSream, BGPRecord, BGPElem
from collections import defaultdict
from itertools import groupby
import networkx as nx

stream = BGPSream()
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(lambda x: [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
```

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

Example: monitor your own address space on BGP

The “prefix-monitor” plugin (distributed with source) monitors a set of IP ranges as they are seen from BGP monitors distributed worldwide:
- how many prefixes announced
- how many origin ASes
- generates detailed logs

Hijacking of AS137 (GARR) - Jan 2015*