graphing time-series data using python end-to-end:

based on experiences at deviantART

Chase Pettet, 2014
Network And Systems Engineer
https://github.com/chasemp

Why?

At Velocity 2012 I saw OmniTI CEO give a talk called ‘It’s all about Telemetry”

I walked away thinking Cacti was in the stone age.

Lessons learned:

Data presentation *matters* a lot

Cacti and RRD in general are too static for presentation

Trending data (a baseline) has an extremely high ROI.

There are a lot of projects in this space

https://github.com/sensu

http://collectd.org/

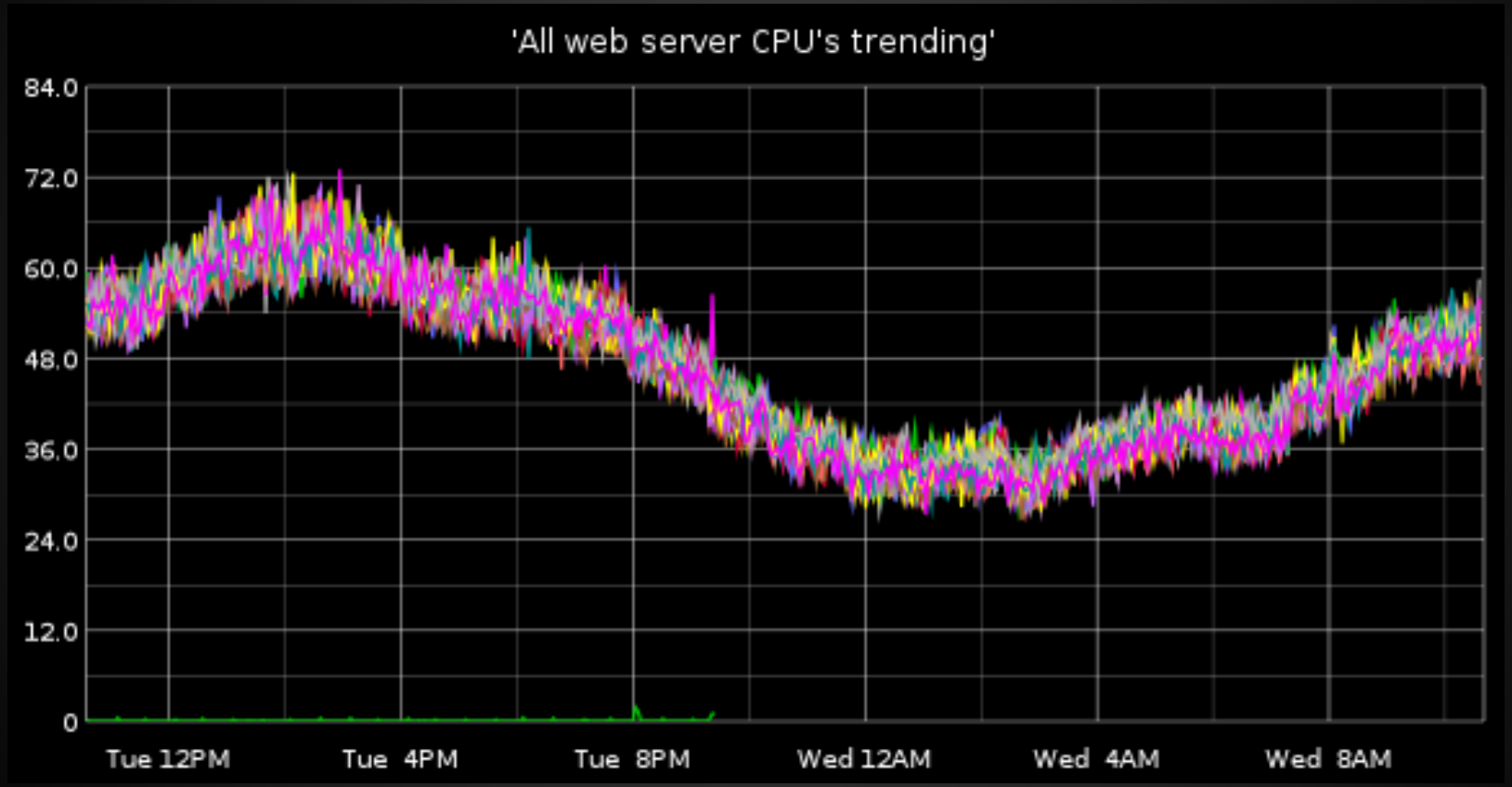
https://github.com/noahhl/batsd

For more see:

http://graphite.readthedocs.org/en/latest/tools.html

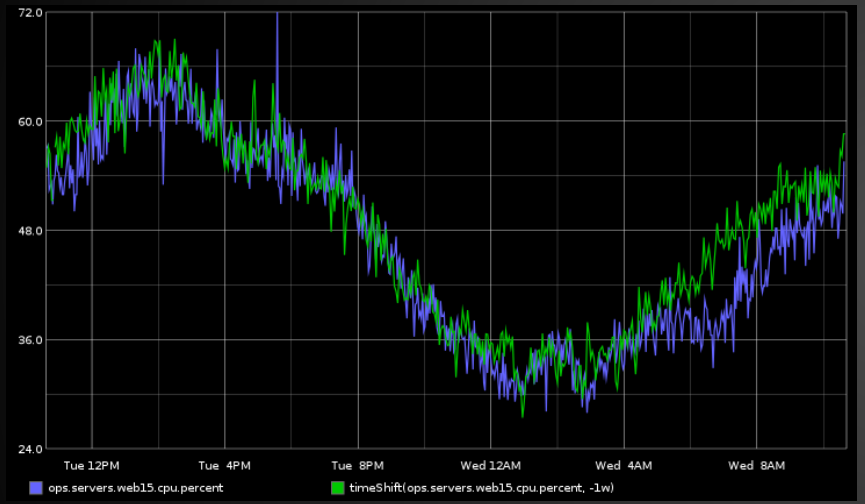
Seeing value in Graphite

You can use globbing across metrics: web*.cpu.percent

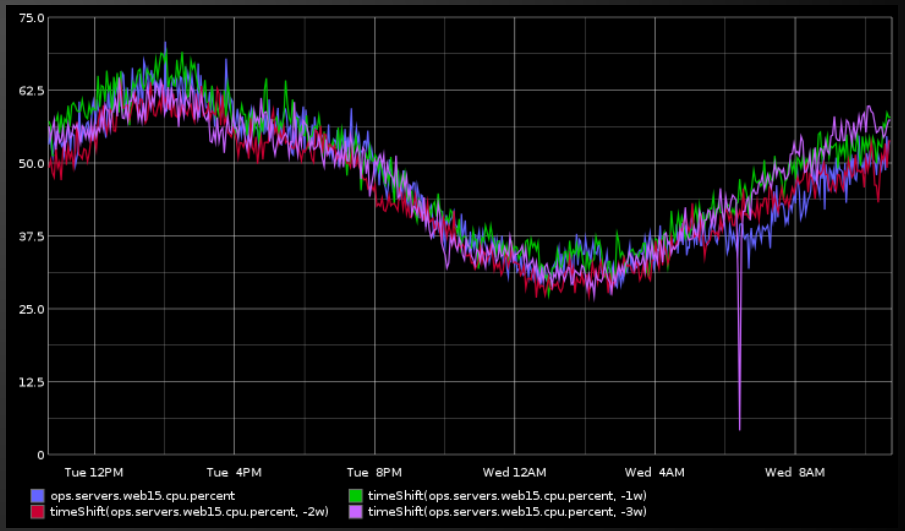


You can use timeshifting to compare to previous:

This week vs. last week



This week vs. last 3 weeks

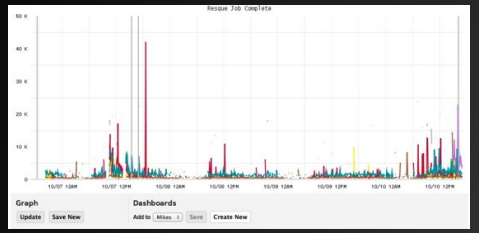


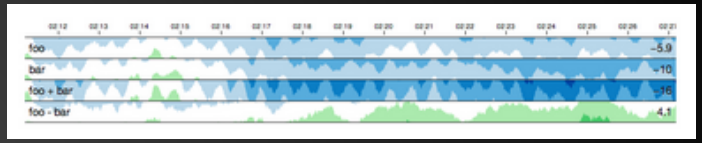
You can get raw data in json format for any purpose.

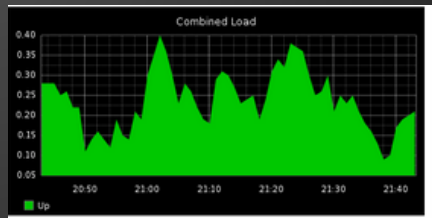
Graphite is a platform for integrating trending data into your work.

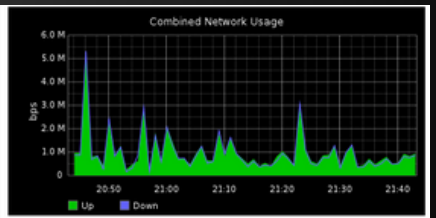

```
[{"target": "statsd.ops.numStats", "datapoints": [[29787.0, 1393353840], [29152.0, 1393353850], [28975.0, 1393353860], [29267.0, 1393353870], [29
1393353880], [29249.0, 1393353890], [28314.0, 1393353900], [27788.0, 1393353910], [29838.0, 1393353920], [29416.0, 1393353930], [28916.0,
1393353940], [29318.0, 1393353950], [28752.0, 1393353960], [27426.0, 1393353970], [29127.0, 1393353980], [28992.0, 1393353990], [28877.0,
1393354000], [28744.0, 1393354010], [28947.0, 1393354020], [27371.0, 1393354030], [29256.0, 1393354040], [29111.0, 1393354050], [29728.0,
1393354060], [29574.0, 1393354070], [28383.0, 1393354080], [26901.0, 1393354090], [28989.0, 1393354100], [29371.0, 1393354110], [29567.0,
1393354120], [29507.0, 1393354130], [29757.0, 1393354140], [27494.0, 1393354150], [27885.0, 1393354160], [29615.0, 1393354170], [29770.0,
1393354180], [29037.0, 1393354190], [29168.0, 1393354200], [28823.0, 1393354210], [30229.0, 1393354220], [29584.0, 1393354230], [29848.0,
1393354240], [29720.0, 1393354250], [29176.0, 1393354260], [28106.0, 1393354270], [29577.0, 1393354280], [29402.0, 1393354290], [29204.0,
1393354300], [29425.0, 1393354310], [29566.0, 1393354320], [28161.0, 1393354330], [29626.0, 1393354340], [29439.0, 1393354350], [30220.0,
1393354360], [29645.0, 1393354370], [28917.0, 1393354380], [27694.0, 1393354390], [29501.0, 1393354400], [29030.0, 1393354410], [29622.0,
1393354420], [29264.0, 1393354430], [29036.0, 1393354440], [26903.0, 1393354450], [29204.0, 1393354460], [29001.0, 1393354470], [29275.0,
1393354480], [29458.0, 1393354490], [28897.0, 1393354500], [29346.0, 1393354510], [29478.0, 1393354520], [29735.0, 1393354530], [29316.0,
1393354540], [29027.0, 1393354550], [29382.0, 1393354560], [27408.0, 1393354570], [29281.0, 1393354580], [28241.0, 1393354590], [29081.0,
1393354600], [29333.0, 1393354610], [28958.0, 1393354620], [28892.0, 1393354630], [29607.0, 1393354640], [29217.0, 1393354650], [29843.0,
1393354660], [29324.0, 1393354670], [29164.0, 1393354680], [28137.0, 1393354690], [29480.0, 1393354700], [29389.0, 1393354710], [29451.0,
1393354720], [28581.0, 1393354730], [29338.0, 1393354740], [27941.0, 1393354750], [29273.0, 1393354760], [28993.0, 1393354770], [28227.0,
1393354780], [28891.0, 1393354790], [28941.0, 1393354800], [27908.0, 1393354810], [28625.0, 1393354820], [29377.0, 1393354830], [29269.0,
1393354840], [28314.0, 1393354850], [29162.0, 1393354860], [27658.0, 1393354870], [28646.0, 1393354880], [29218.0, 1393354890], [29087.0,
1393354900], [29111.0, 1393354910], [28568.0, 1393354920], [27226.0, 1393354930], [28236.0, 1393354940], [28899.0, 1393354950], [28721.0,
1393354960], [29355.0, 1393354970], [28347.0, 1393354980], [27945.0, 1393354990], [27165.0, 1393355000], [28731.0, 1393355010], [29624.0,
1393355020], [29203.0, 1393355030], [29854.0, 1393355040], [26553.0, 1393355050], [26930.0, 1393355060], [29118.0, 1393355070], [28144.0,
```

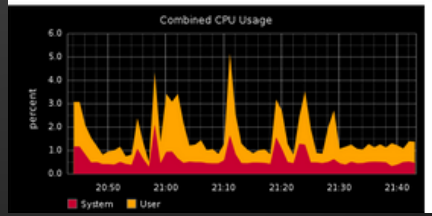
Since the API is so nice people have built lots of dashboards.

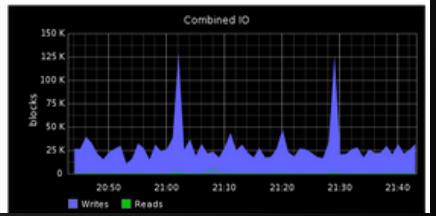






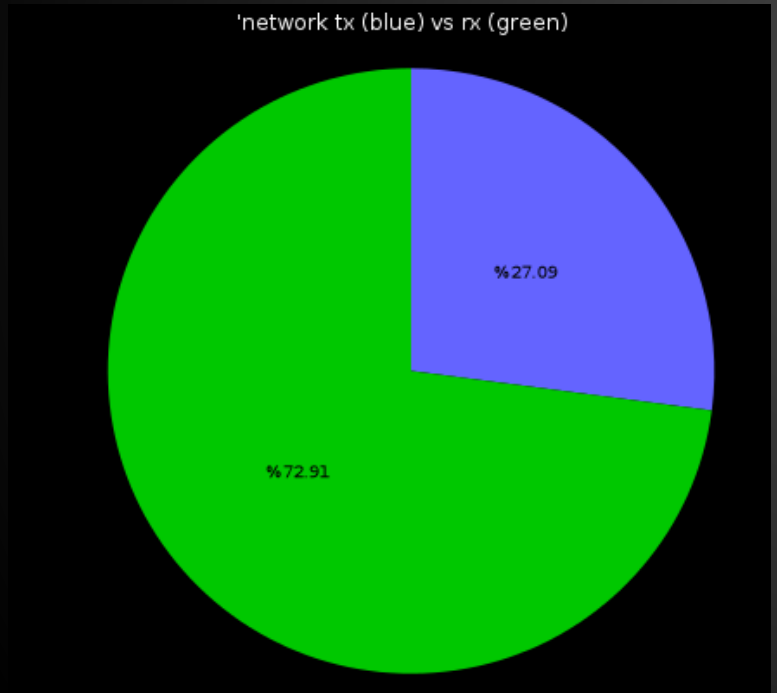


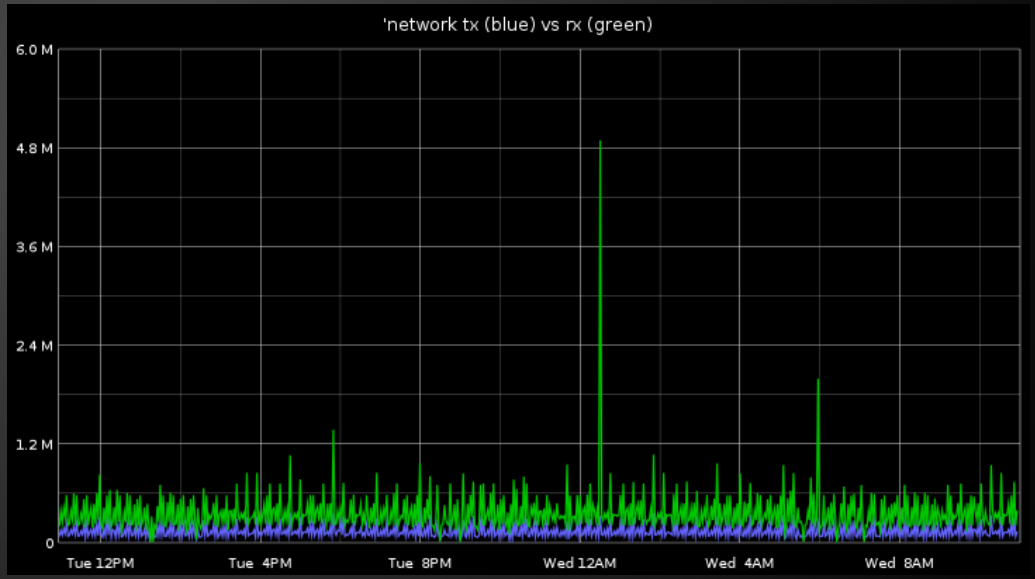




But you don’t need to use an external dashboard.

Graphite has a rendering engine built in. That means you can embed graphs anywhere.





Graphite has a bunch of built in post-processing functions for data.

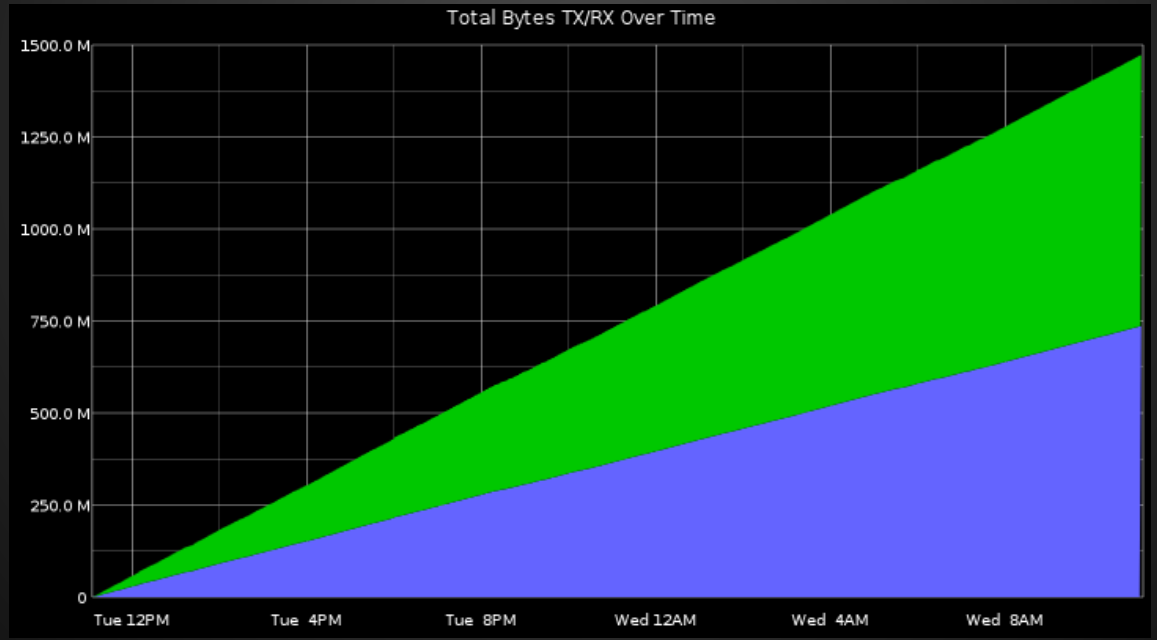
integral(*requestContext*, *seriesList*)

This will show the sum over time, sort of like a continuous addition function. Useful for finding totals or trends in metrics that are collected per minute.

Example:

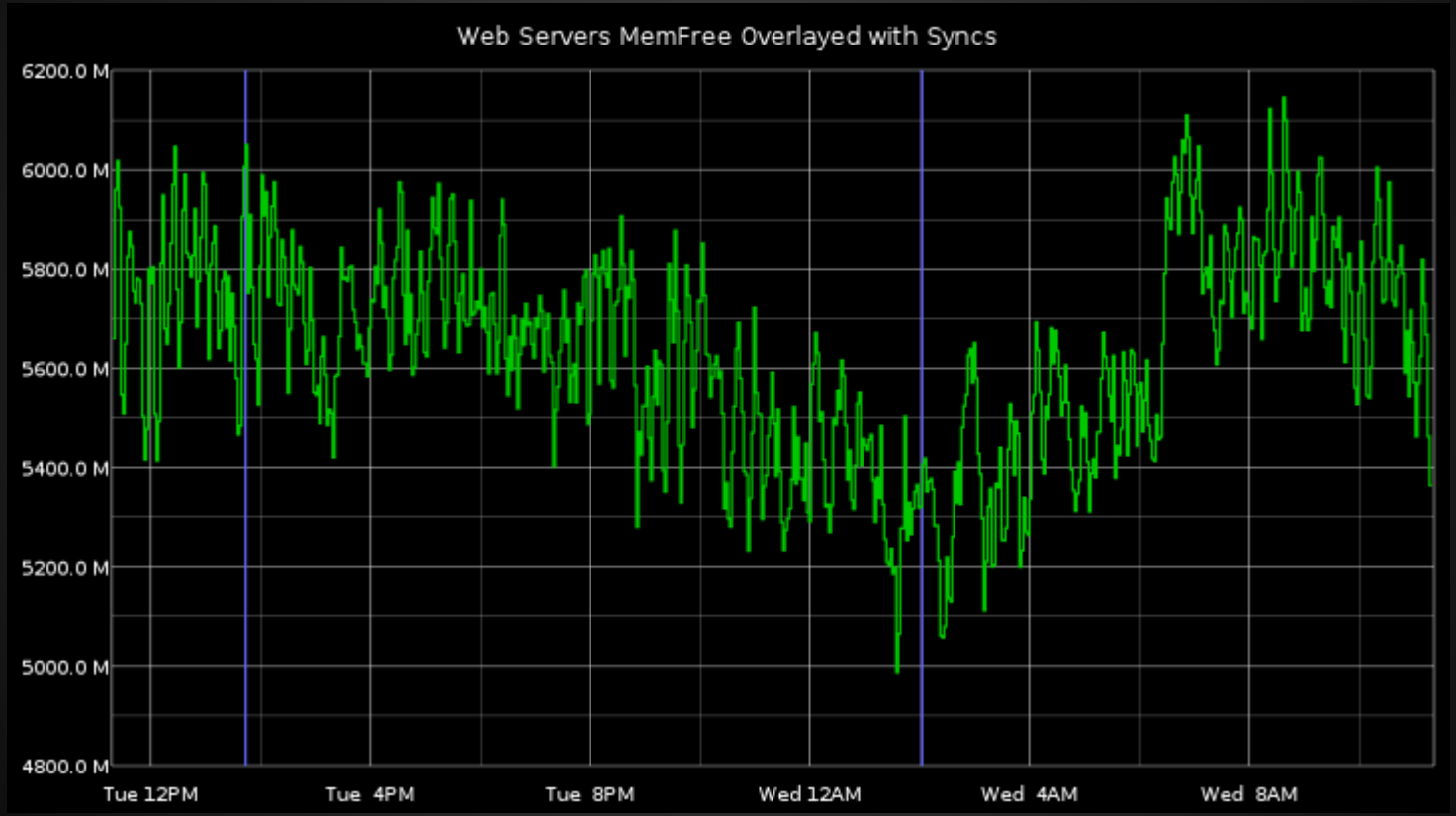
```
&target=integral(company.sales.perMinute)
```

This would start at zero on the left side of the graph, adding the sales each minute, and show the total sales for the time period selected at the right side, (time now, or the time specified by '&until=').



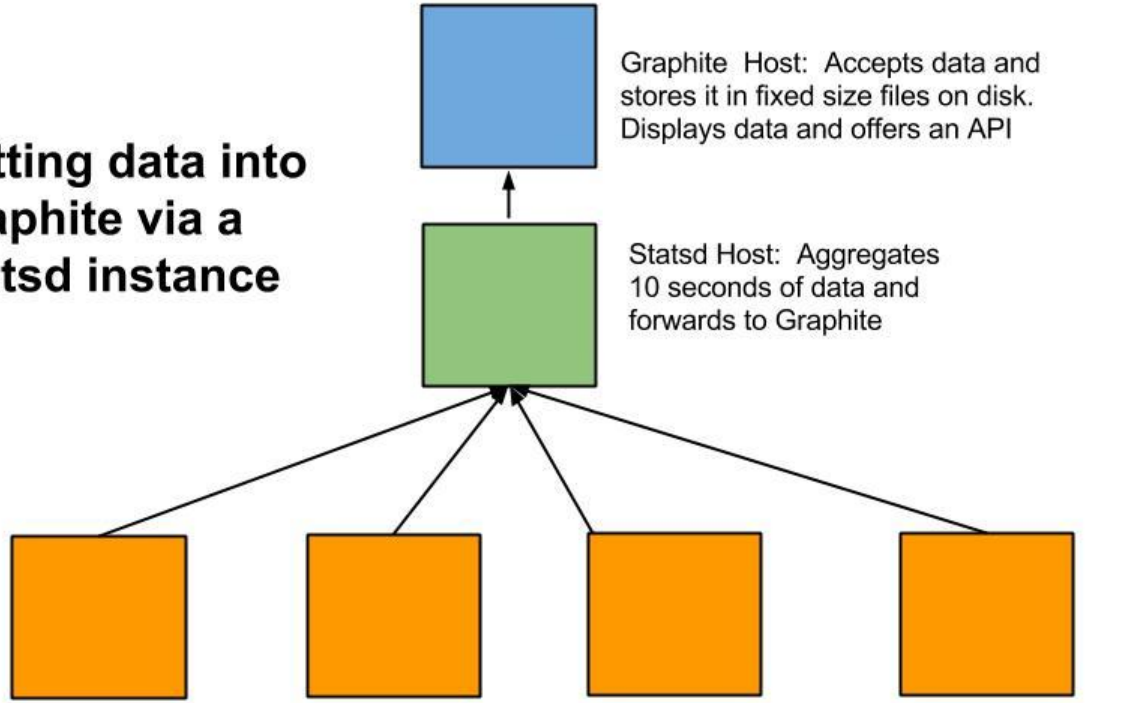
Graphite can overlay any events relevant to your data:

```
curl -X POST http://localhost:8000/events/ -d '{"what": "Something Interesting", "tags" : "tag1 "}'
```



tl;dr: Graphite is flexible.

Getting data into Graphite via a Statsd instance



How is this devopsy?


Ops and Dev…
shared tools man...

What?

Diamond

Host based collection and submission service for providing statistics to an upstream receiver.

Existing output handlers:

Graphite, RabbitMQ, Riemann, Sentry, ZeroMQ, Syslog, Statsd, HTTP, etc

Statsd

Service that receives, aggregates, and flushes statistics at a set interval to Graphite.

I am using a deviantART specific fork that is mostly compatible with the canonical version from etsy.

https://github.com/deviantART/pystatsd

Aggregation methods:

Counter, Set, Gauge, Timer

Logster

Aggregates statistics from log files and submits them to an upstream receiver.

Existing format handlers:

Postfix, Squid, Log4j, etc

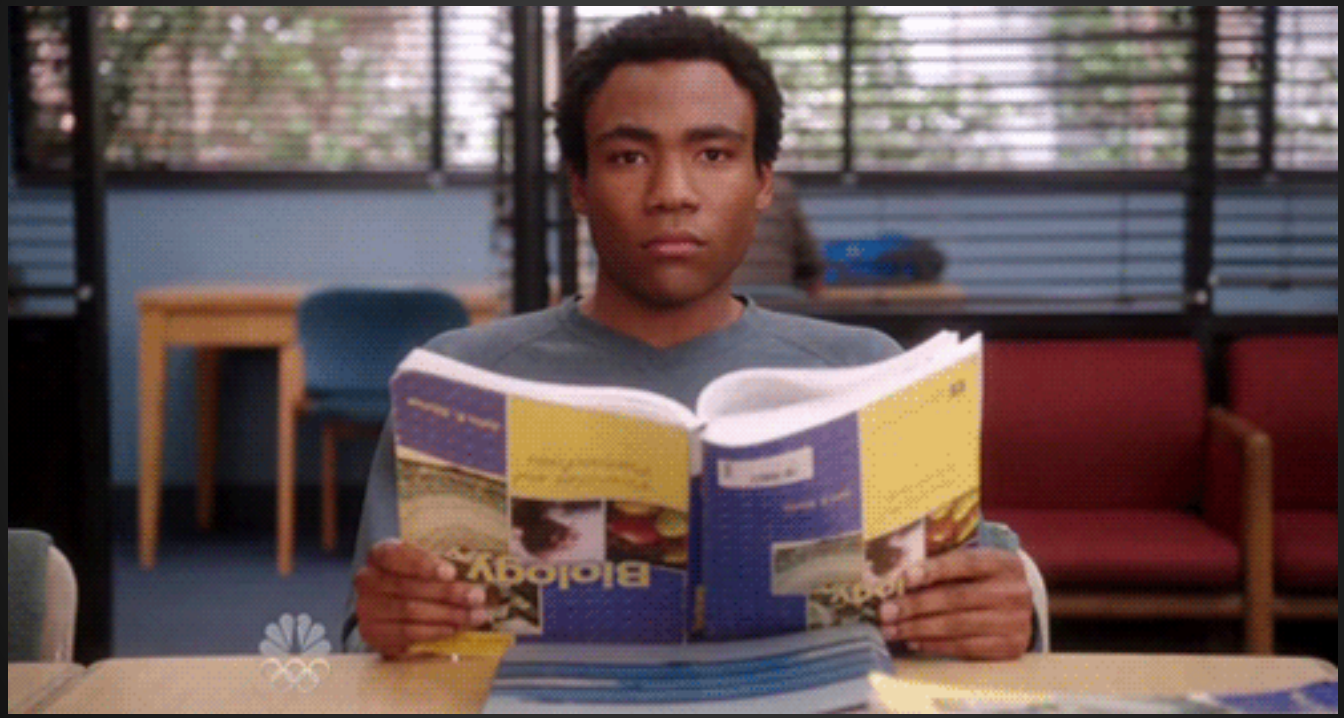
Graphite

A highly scalable graphing system.

Internals:

- Carbon receives and persists statistics
- Whisper is the fixed size time series file format
- Graphite-Web is a Django based front end

Lots of stuff, right?



How?

Running Diamond Agent

Installation:

aptitude install python-configobj

git clone https://github.com/BrightcoveOS/Diamond.git

cd Diamond/ && make builddeb

dpkg -i build/diamond_3.4.266_all.deb

Default Statistics Collectors:

- CPU
- Disk Space / Usage
- Load Average
- Memory
- MountStats
- SocketStats
- VMStat

tail /var/log/diamond/diamond.log

```
[2014-02-25 09:40:17,344] [Thread-29735] servers.idle23.memory.Buffers 1245184 1393350017
[2014-02-25 09:40:17,346] [Thread-29735] servers.idle23.memory.Active 457539584 1393350017
[2014-02-25 09:40:17,346] [Thread-29735] servers.idle23.memory.Inactive 100413440 1393350017
[2014-02-25 09:40:17,346] [Thread-29735] servers.idle23.memory.SwapTotal 0 1393350017
[2014-02-25 09:40:17,346] [Thread-29735] servers.idle23.memory.SwapFree 0 1393350017
```

Running Logster: Log Culling

Installation:

```
git clone https://github.com/etsy/logster.git
python bin/logster \
        -p servers.myhost.postfix \
        --output=statsd \
        --statsd-host=statsd:8125 \
        PostfixLogster /var/log/mail.log
```

```
statsd:8125 servers.proxy01b.postfix.numSent:1000|g
statsd:8125 servers.proxy01b.postfix.pctSent:100.0|g
statsd:8125 servers.proxy01b.postfix.numDeferred:2|g
statsd:8125 servers.proxy01b.postfix.pctDeferred:3.0|g
statsd:8125 servers.proxy01b.postfix.numBounced:0|g
statsd:8125 servers.proxy01b.postfix.pctBounced:0.0|g
```

Running Statsd

Running statsd (not persistent):

aptitude install python-twisted

git clone https://github.com/deviantART/pystatsd.git

cd pystatsd/ && python statsd.py

Sending data to Statsd:

```
import socket                                                    #get socket module
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM, 0)          #setup socket handler
s.connect((10.10.10.10, 8125))                                   #open 'connection'
s.send("counter_at_50_perc_sample:1|c|@0.5")                     #send a counter sampled at half your rate
s.send("counter_at_one_to_one:1|c|@1")                           #send a counter sampled at 1:1 rate
s.send("mytimer:100|ms")                                         #send the operation time of mytime
s.send("mygauge:1|g")                                            #send a gauge for 1
s.send("myset:1")                                                #send a set of 1
s.close()
```

dA Statsd Niceties

- Uses Twisted to handle the network portion
- Can withstand Qualys and Nessus Vulnerability Scanning
- Provides an XMLRPC interface for monitoring
- Publishes a lot more meta stats about internals
- Cleanly handles 'bad' metric types w/ discard
- Survives and reports failed Graphite flushes
- Allows multiple metrics in a single message using newline character
- Failures go to stderror
- Has a '-d' debug option for dumping matching incoming stats to terminal
- Can notify NSCA on failures if an notify_nsca function is provided
- Allows incoming stats over TCP as well as UDP

Can handle 50,000+ metric output on a 1 core VM using >1Gig RAM

statsdmonitor.py output

```
flush_duration: 0.410356044769
flush_errors: 0
last_publish.duration: 0.352137088776
last_publish.time: 1393294923.76
metrics.batch.count: 1454
metrics.batch.discarded: 0
metrics.discarded: 120
pending_stats: 10334
publish_error.socket: 0
publish_error.unknown: 0
render.counters.count: 0
render.counters.duration: 0
render.gauge.count: 28977
render.gauge.duration: 0.049978017807
render.set.count: 0
render.set.duration: 0
render.time: 0.0500249862671
render.timers.count: 0
render.timers.duration: 0
render.total: 28990
start_time: 1392235332.92
```

Understanding Statsd

Question:

If Diamond can send statistics directly to Graphite then why do I need Statsd?

Answer:

You need Statsd if you want to submit data at intervals smaller than the smallest one stored by Graphite.

Statsd accepts data at all times, summarizes it and submits to Graphite.

Graphite only needs to store one datapoint per X seconds saving on disk space, and resources.

If you want to submit a value very ten seconds and you store 10 second intervals of data in Graphite you do not need Statsd.

Running Graphite

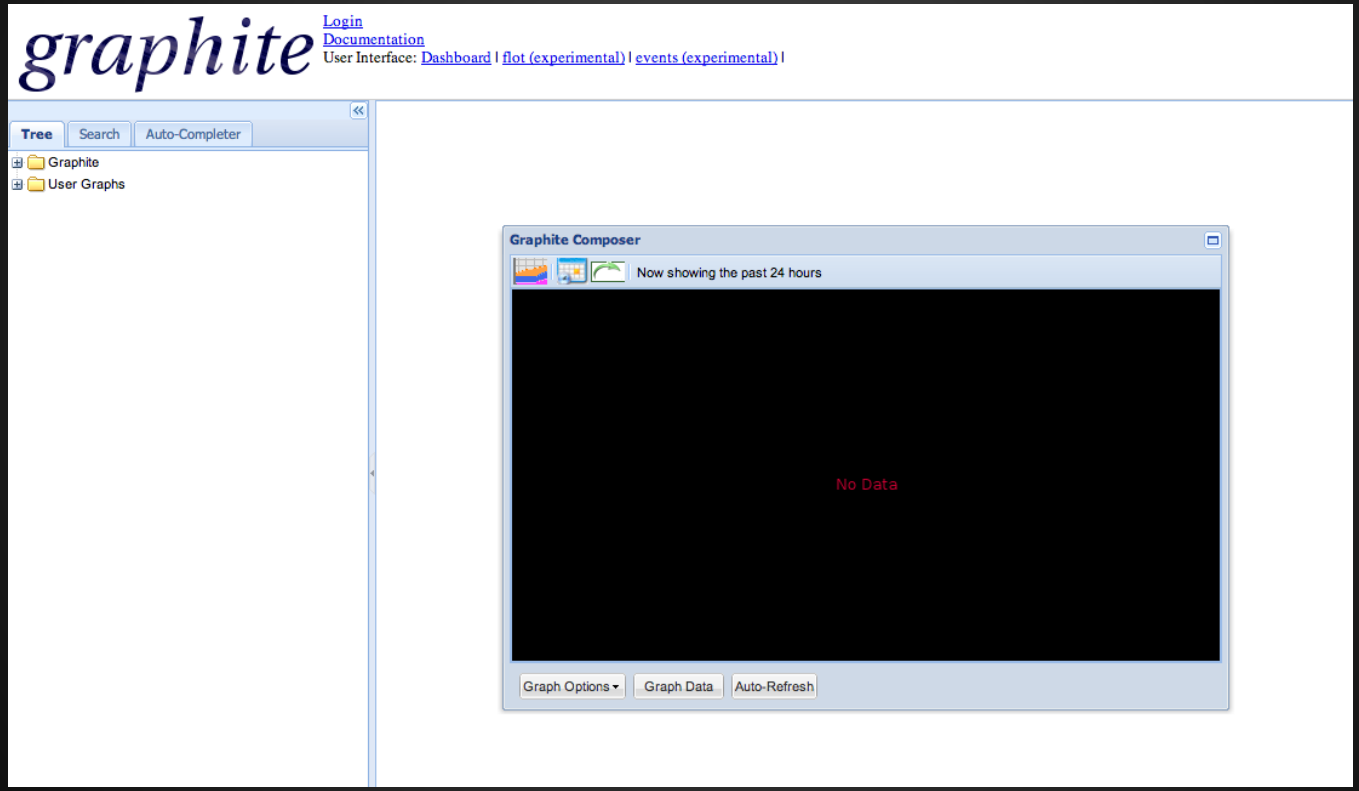
Installation:

```
pip install whisper
pip install carbon
pip install graphite-web
```

Further instructions:

http://chasemp.github.io/2013/06/15/debian-graphite-install/
http://chasemp.github.io/2013/09/12/graphite-on-centos6-lessons-learned/

Graphite Web Interface



Resources:

http://velocityconf.com/velocity2012/public/schedule/detail/23354

https://github.com/BrightcoveOS/Diamond

https://github.com/chasemp/pystatsd

https://github.com/graphite-project

https://github.com/etsy/logster

http://chasemp.github.io/2013/05/17/using-graphite-events/

http://chasemp.github.io/2013/08/15/graphite-test-clients/

http://chasemp.github.io/2013/06/15/debian-graphite-install/

http://chasemp.github.io/2013/03/01/graphite-all-metrics/

http://chasemp.github.io/2013/09/12/graphite-on-centos6-lessons-learned/