carbon, graphite and grafana

carbon

install carbon-cache service

```
apt-get install graphite-carbon
```

enable boot startup in */etc/default/graphite-carbon*

enable ENABLE_LOGROTATION in */etc/carbon/carbon.conf*

extend retentions

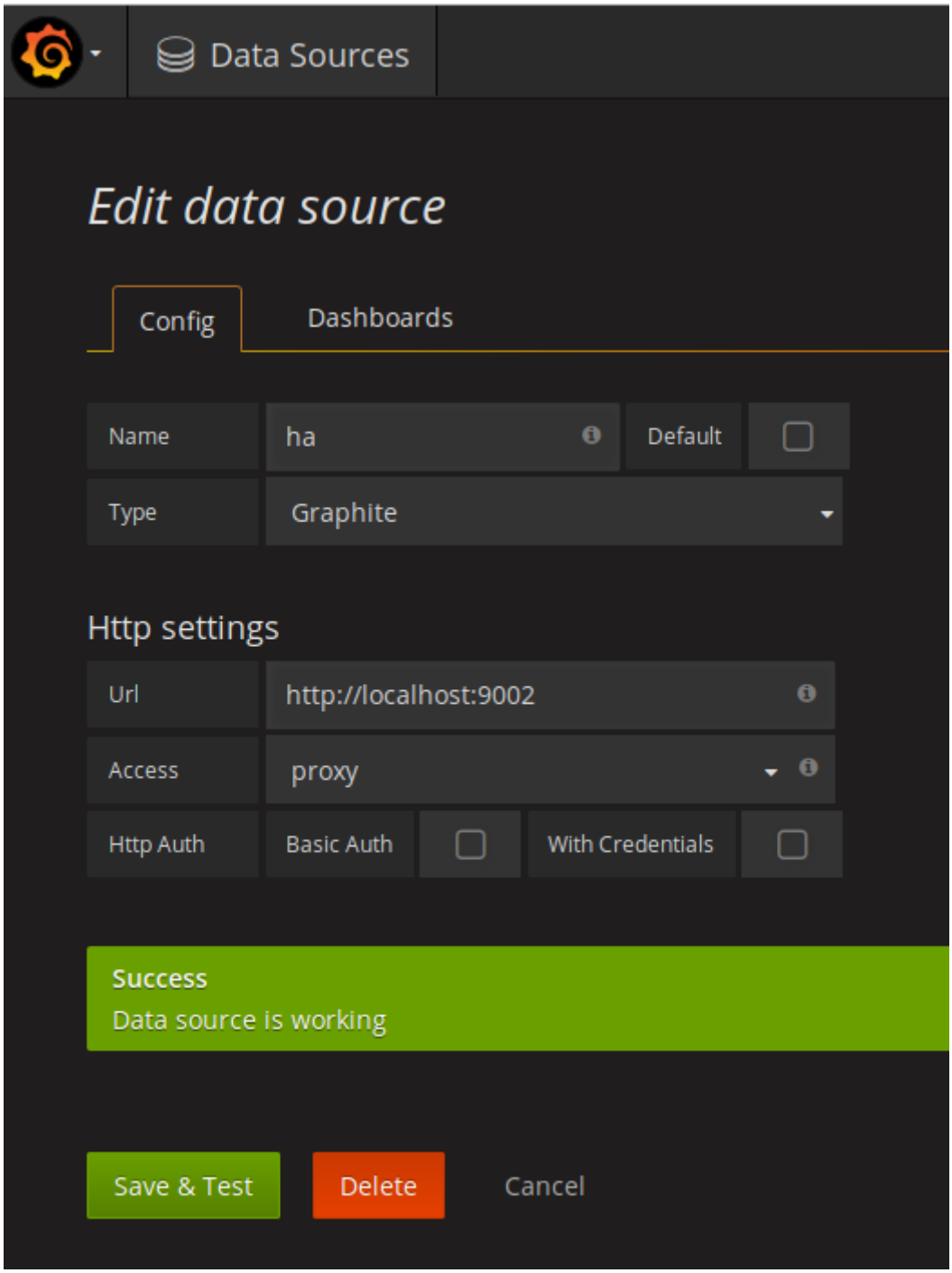
/etc/carbon/storage-schemas.conf

```
[carbon]
pattern = ^carbon\.
retentions = 60:90d

[default]
pattern = .*
retentions = 30s:7d,5m:30d,10m:1y,1d:10y
```

after changing *schemas.conf older *wsp has to be changed with whisper-resize <code bash> for i in `find /var/lib/graphite/whisper/ha/ -name "*wsp"`; do whisper-resize $i 30s:7d 5m:30d 10m:1y 1d:10y done </code> ===== graphite-web ===== web interface <code bash> apt-get install graphite-web </code> adjust SECRETKEY (with random string) and TIMEZONE in /etc/graphite/local_settings.py <code bash> # choose user and password graphite-manage syncdb chown graphite:graphite /var/lib/graphite/graphite.db #python /usr/lib/python2.7/dist-packages/graphite/manage.py syncdb </code> ==== apache ==== <code bash> sudo apt-get install apache2 libapache2-mod-wsgi </code> <code bash> sudo cp /usr/share/graphite-web/apache2-graphite.conf /etc/apache2/sites-available sudo a2ensite apache2-graphite </code> ==== nginx ==== <code bash> apt install nginx uwsgi uwsgi-plugin-python </code> <file ini /etc/uwsgi/apps-enabled/graphite.ini> [uwsgi] vacuum = true master = true processes = 2 pidfile = /tmp/uwsgi.pid socket = /tmp/uwsgi.sock chmod-socket = 666 gid = graphite uid = _graphite chdir = /usr/share/graphite-web wsgi-file = graphite.wsgi pymodule-alias = graphite.localsettings=/etc/graphite/local_settings.py buffer-size = 65536 plugin = python </file> <code bash> systemctl restart uwsgi </code> <file txt /etc/nginx/sites-enabled/graphite> upstream graphite { server unix:/tmp/uwsgi.sock; } server { listen 9002; server_name localhost; accesslog /var/log/nginx/graphite-access.log; errorlog /var/log/nginx/graphite-error.log; root /usr/share/graphite-web/static; location / { addheader Access-Control-Allow-Origin “*”; addheader Access-Control-Allow-Methods “GET, OPTIONS”; add_header Access-Control-Allow-Headers “origin, authorization, accept”; uwsgipass graphite; include /etc/nginx/uwsgiparams; } location /media { # This makes static media available at the /media/ url. The # media will continue to be available during site downtime, # allowing you to use styles and images in your maintenance page. alias /usr/lib/python2.7/dist-packages/django/contrib/admin/media; } } </file> ===== grafana ===== ==== install ====

<code bash> echo “deb https://packagecloud.io/grafana/stable/debian/ jessie main” > /etc/apt/sources.list.d/grafana.list curl https://packagecloud.io/gpg.key | sudo apt-key add - apt update apt install -y grafana systemctl enable grafana-server systemctl start grafana-server </code> Login to http://localhost:3000 (admin/admin) add datasource



==== plugins ==== Add plugin in /var/lib/grafana/plugins/ directory piechart <code bash> cd /var/lib/grafana/plugins/ git clone https://github.com/grafana/piechart-panel.git systemctl restart grafana-server.service </code> ===== influxdb ===== <code> curl -sL https://repos.influxdata.com/influxdb.key | apt-key add - source /etc/lsb-release echo “deb https://repos.influxdata.com/${DISTRIB*ID,,} ${DISTRIB*CODENAME} stable” | tee -a /etc/apt/sources.list apt update apt install influxdb </code> enable admin service in /etc/influxdb/influxdb.conf and <code> systemctl restart influxdb </code> create database <code> root@graphite:~# influx Connected to http://localhost:8086 version 1.2.1 InfluxDB shell version: 1.2.1 > create database captive > CREATE USER “captive” WITH PASSWORD 'captive' WITH ALL PRIVILEGES > show databases name: databases name —- _internal captive > </code> test <code>

curl -G http://carbon:8086/query –data-urlencode “q=SHOW DATABASES” </code> python test <code python> from influxdb import InfluxDBClient client = InfluxDBClient('carbon.csgalileo.org', 8086, username='captive', password='captive', database='test') json*body = [{“measurement”: “browser”, “tags”: {“server”:1, “server-name”:“galileo”}, “time”: datetime.utcnow().strftime('%Y-%m-%dT%H:%M:%SZ'), “fields”: {“value”:“ios”}}] client.write*points(json_body) client.query('select value from browser;') </code>