YETI installation manual

Contents

1 Overview

Minimal system installation requires following compoments:

- Routing database (postgresql-9.3) see section §2
- CDRs database (postgresql-9.3) see section §2
- Management interface (web interace written on RoR) see section §3
- DSS storage (Redis) see section §4
- Management server (configurations server) see section §5
- Traffic switch (SBC based on SEMS) see section §6
- Load balancer (kamailio) see section §7

2 Databases installation

System requires two databases: one for routing and one for CDRs.

We recommend to place routing database at the same host with management interface to reach best interface responsiveness by reducing network delay for database requests from web-interface.

2.1 Install packages

both routing and CDRs databases requires similar sets of connected repositories and installed packages.

Make sure that following repositories added on your system:

```
deb http://ftp.us.debian.org/debian/ wheezy main
deb http://security.debian.org/ wheezy/updates main
deb http://ftp.us.debian.org/debian/ wheezy-updates main
deb http://pkg.yeti-switch.org/debian wheezy/
deb http://apt.postgresql.org/pub/repos/apt/ wheezy-pgdg main
```

System repositories can be changed by editing of file: /etc/apt/sources.list

import public keys for repositories:

```
# apt-key adv --keyserver keys.gnupg.net --recv-key 9CEBFFC569A832B6
# apt-key adv --keyserver keys.gnupg.net --recv-key 7FCC7D46ACCC4CF8
```

Install packages:

```
# aptitude update && aptitude install postgresql-9.3 postgresql-contrib-9.3 postgresql
   -9.3-prefix postgresql-9.3-pgq3 postgresql-9.3-yeti skytools3 skytools3-ticker
```

2.2 Create databases

Create routing database:

```
# su - postgres
$ psql
postgres=# create user yeti encrypted password 'somepassword' superuser;
CREATE ROLE
postgres=# create database yeti owner yeti;
CREATE DATABASE
postgres=# \q
```

Create database to store CDR:

```
# su - postgres
$ psql
postgres=# create database cdr owner yeti;
CREATE DATABASE
postgres=# \q
```

Note: It's recommended to specify values for databases names, usernames, passwords differ from specified in this manual for security reasons.

For large installations is reasonable to place CDR database on dedicated server

2.3 Check

Check databases created and accessible:

```
root@evial:/# psql -h 127.0.0.1 -U yeti -d yeti
Password for user yeti:  psql (9.3.9) SSL connection (cipher: DHE-RSA-AES256-GCM-SHA384,
    bits: 256)
Type "help" for help.

yeti=# \q
root@evial:/#

root@evial:/# psql -h 127.0.0.1 -U yeti -d cdr
Password for user yeti:
psql (9.3.9)
SSL connection (cipher: DHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.

cdr=# \q
root@evial:/#q
```

Don't forget to make changes in */etc/postgresql/9.3/main/pg_hba.conf* and apply them if you plan to access this databases from another hosts

2.4 Init schema and data

Look at section 3.2 and section 3.3 for further databases initialization instructions.

3 Management interface installation

Server requirements:

- OS Debian 7 Wheezy with architecture amd64
- at least 1GB of RAM

3.1 Install packages

Make sure that following repositories added on your system:

```
deb http://ftp.us.debian.org/debian/ wheezy main
deb http://security.debian.org/ wheezy/updates main
deb http://ftp.us.debian.org/debian/ wheezy-updates main
deb http://pkg.yeti-switch.org/debian wheezy/
deb http://packages.dotdeb.org wheezy all
deb http://apt.postgresql.org/pub/repos/apt/ wheezy-pgdg main
```

System repositories can be changed by editing of file: /etc/apt/sources.list

import public keys for repositories:

```
# apt-key adv --keyserver keys.gnupg.net --recv-key 9CEBFFC569A832B6
# apt-key adv --keyserver keys.gnupg.net --recv-key E9C74FEEA2098A6E
# apt-key adv --keyserver keys.gnupg.net --recv-key 7FCC7D46ACCC4CF8
```

Install packages:

```
# aptitude update && aptitude install yeti-web
```

3.2 Configure database connection

To configure database connection edit file /home/yeti-web/config/database.yml

Create database.yml file with the following content:

```
production:
adapter: postgresql
encoding: unicode
database: yeti
pool: 5
username: yeti
password: somepassword
host: 127.0.0.1
schema_search_path: 'gui,public,switch,billing,class4,runtime_stats,sys,logs,data_import
  '
port: 5432
#min_messages: warning

production_cdr:
adapter: postgresql
encoding: unicode
database: cdr
pool: 5
```

```
username: yeti
password: somepassword
host: 127.0.0.1
schema_search_path: 'cdr,reports,billing'
port: 5432
#min_messages: warning
```

Warning: you should specify correct adresses and credentials using those which you chose in previous section

3.3 Init databases data

To simplify work with databases use utility **yeti-db**

To initialize empty databases:

```
# yeti-db init
# yeti-db --cdr init
```

To upgrade database to the lastest version:

```
# yeti-db apply_all
# yeti-db --cdr apply_all
```

You can check actual database versions:

```
# yeti-db version
# yeti-db --cdr version
```

Attention: During upgrade of the system which working in production command **apply_all** should not be used because this command intended to upgrade to the last version only for system without live traffic. Systems in production must be upgraded using command **apply** which applies just one update in a single run. After each upgrade it is important to amend appropriate configuration files and restart all traffic switch instances. This approach provides zero-downtime upgrade procedure (without loss of traffic and CDRs)

3.4 Launch

After successfull configuration of databases you finally can run software using following commands:

```
# /etc/init.d/yeti-web start
# /etc/init.d/yeti-cdr-billing start
# /etc/init.d/yeti-delayed-job start
```

This will run web-interface and CDR processing daemons

3.5 Check

check if uncorn listens socket:

```
# netstat -lpn | grep unicorn
unix 2 [ ACC ] STREAM LISTENING 2535145 24728/unicorn.rb -E /tmp/yeti-unicorn.sock
```

check if nginx listens for appropriate sockets:

```
# netstat -lpn | grep nginx
tcp 0 0 0.0.0.0:80 0.0.0.0:* LISTEN 23627/nginx
tcp 0 0 127.0.0.1:6666 0.0.0.0:* LISTEN 23627/nginx
```

Log files to check for possible warnings/errors:

- /var/log/yeti-admin.log
- /var/log/yeti-cdr-billing.log
- /home/yeti-web/log/unicorn.stdout.log
- /home/yeti-web/log/unicorn.stderr.log

Try to open management interface in your favorite browser and login with default credentials:
user: admin
password: 111111

4 DSS Storage installation

Redis is used to synchronize data between traffic switch instances. It stores information about used resources (e.g gateways capacity limits) to provide correct limitation among all nodes for distributed installations.

4.1 Install packages

For installation make sure that your system have following repositories:

```
deb http://ftp.us.debian.org/debian/ wheezy main
deb http://security.debian.org/ wheezy/updates main
deb http://ftp.us.debian.org/debian/ wheezy-updates main
```

Install package:

```
# aptitude install redis-server
```

4.2 Check

Try to enter redis console from traffic switch host (redis installed at the same host with traffic switch in this example)

```
# redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> quit
```

5 Management server installation

Since version 1.6.3-175 we started to use central configuration server to store yeti module configuration for all nodes in cluster.

5.1 Install packages

For installation make sure that your system have following repositories:

```
deb http://ftp.us.debian.org/debian/ wheezy main
deb http://security.debian.org/ wheezy/updates main
deb http://ftp.us.debian.org/debian/ wheezy-updates main
deb http://pkg.yeti-switch.org/debian wheezy/
deb http://packages.dotdeb.org wheezy all
```

import public keys for repositories:

```
# apt-key adv --keyserver keys.gnupg.net --recv-key 9CEBFFC569A832B6
# apt-key adv --keyserver keys.gnupg.net --recv-key E9C74FEEA2098A6E
```

Install package:

```
# aptitude install yeti-management
```

5.2 Configuration files

5.2.1 /etc/yeti/management.cfg

This file contains configuation for management daemon.
Set desired logging level and address to listen.
You can set multiple addresses separated by comma to listen multiple addresses.
Possible log_level values are: (1 - error, 2 - info, 3 - debug)

```
daemon {
  listen = {
    "tcp://0.0.0.0:4444"
  }
  log_level = 2
}
```

5.2.2 /etc/yeti/system.cfg

This file contains configuration for all nodes.
Each top-level section defines configuration for node of certain type (*signalling* is for traffic switch nodes).
All top-level sections contains mandatory section *globals* which must have all possible values common for all nodes.
Then there is named sections for each node_id which can contains overrides of global parameters.
Note: even if your node does not have any specific values you have to define empty section for this node anyway, otherwise management node will not return configuration for node with such id. Example of minimal configuration file for node with id 0:

```
signalling {
  globals {
    yeti {
      pop_id = 2
      msg_logger_dir = /var/spool/sems/dump
      log_dir = /var/spool/sems/logdump
      routing {
        schema = switch8
        function = route_release
        init = init
        master_pool {
          host = 127.0.0.1
          port = 5432
          name = yeti
          user = yeti
          pass = yeti
          size = 4
          check_interval = 10
          max_exceptions = 0
          statement_timeout=3000
        }
        failover_to_slave = false
        slave_pool {
          host = 127.0.0.1
          port = 5432
          name = yeti
          user = yeti
          pass = yeti
          size = 4
          check_interval = 10
          max_exceptions = 0
          statement_timeout=3000
        }
        cache {
          enabled = false
          check_interval = 60
          buckets = 100000
        }
      }
      cdr {
       dir = /var/spool/sems/cdrs
       completed_dir = /var/spool/sems/cdrs/completed
       pool_size = 2
       schema = switch
       function = writecdr
       master {
         host = 127.0.0.1
         port = 5433
         name = cdr
         user = cdr
         pass = cdr
       }
       failover_to_slave = false
       slave {
         host = 127.0.0.1
         port = 5433
```

```
        name = cdr
        user = cdr
        pass = cdr
      }
      failover_requeue = true
      failover_to_file = false
    }
    resources {
      reject_on_error = false
      write {
        host = 127.0.0.1
        port = 6379
        size = 2
        timeout = 500
      }
      read {
        host = 127.0.0.1
        port = 6379
        size = 2
        timeout = 1000
      }
    }
    registrations {
      check_interval = 5000
    }
    rpc {
      calls_show_limit = 1000
    }
  }
 }
 node 0 { }
}
```

5.3 Launch management server

Launch configured management server instance

```
# /etc/init.d/yeti-management start
```

5.4 Check

Check file */var/log/yeti/yeti-management.log* for daemon logs:

```
# tail -2 /var/log/yeti/yeti-management.log
Sep 12 12:54:47 evial yeti-management[25376]: [25376] info: server/src/yeti_mgmt_server.
   cpp:148: starting version 1.0.5
Sep 12 12:54:47 evial yeti-management[25376]: [25376] info: server/src/mgmt_server.cpp
   :123: listen on tcp://0.0.0.0:4444
```

Check listening port:

```
# netstat -lpn | grep
4444 tcp 0 0 0.0.0.0:4444 0.0.0.0:* LISTEN 25376/yeti_manageme
```

6 Traffic switch server installation

6.1 Install packages

For installation make sure that your system have following repositories:

```
deb http://ftp.us.debian.org/debian/ wheezy main
deb http://security.debian.org/ wheezy/updates main
deb http://ftp.us.debian.org/debian/ wheezy-updates main
deb http://pkg.yeti-switch.org/debian wheezy/
deb http://packages.dotdeb.org wheezy all
```

import public keys for repositories:

```
# apt-key adv --keyserver keys.gnupg.net --recv-key 9CEBFFC569A832B6
# apt-key adv --keyserver keys.gnupg.net --recv-key E9C74FEEA2098A6E
```

Install package:

```
# aptitude install sems-yeti
```

6.2 Configuration files

6.2.1 /etc/sems/sems.conf

Replace <SIGNALLING_IP>, <MEDIA_IP> with correct values for your server

```
interfaces=intern
sip_ip_intern=<SIGNALLING_IP>
sip_port_intern=5061
media_ip_intern=<MEDIA_IP>
rtp_low_port_intern=20000
rtp_high_port_intern=50000
plugin_path=/usr/lib/sems/plug-in/
load_plugins=wav;ilbc;speex;gsm;adpcm;l16;g722;sbc;session_timer;xmlrpc2di;uac_auth;
   di_log;registrar_client
application = sbc
plugin_config_path=/etc/sems/etc/
fork=yes
stderr=no
loglevel=2
max_shutdown_time = 10

session_processor_threads=20
media_processor_threads=2
session_limit="4000;509;Node overloaded"
shutdown_mode_reply="508 Node in shutdown mode"
options_session_limit="900;503;Warning, server soon overloaded"
# cps_limit="100;503;Server overload"
use_default_signature=no
signature="YETI SBC node"
use_raw_sockets=yes
sip_timer_B = 8000
default_bl_ttl=0
registrations_enabled=no
```

6.2.2 /etc/sems/etc/yeti.conf

Set address of management server Replace <DB IP>, <DB username>, <DB name>, <DB password>, <CDR DB IP>, <CDR DB name>, <CDR DB username>, <CDR DB password> with addresses and credentials for configured databases. Also if it needed for your configuration specify parameters for slave databases

node_id: unique signalling node id

cfg_timeout: timeout of waiting response from management node

cfg_urls: list of comma separated names for management node addresses

cfg_url_<name>: management node address to fetch configuration (5.2.1)

```
node_id = 0

cfg_timeout = 1000

cfg_urls = main
cfg_url_main = tcp://127.0.0.1:4444
```

6.2.3 Other configuration files

Copy defaults for the rest of needed configuration files:

```
# cd /etc/sems/etc
# mv /etc/sems/etc/sbc.dist.conf /etc/sems/etc/sbc.conf
# mv /etc/sems/etc/oodprofile.yeti.dist.conf /etc/sems/etc/oodprofile.yeti.conf
# mv /etc/sems/etc/xmlrpc2di.dist.conf /etc/sems/etc/xmlrpc2di.conf
```

6.3 Launch traffic switch

Launch configured traffic switch instance:

```
# /etc/init.d/sems start
```

In case of errors it's useful to use command **sems -E -D3** which will launch daemon in foreground with debug logging level

6.4 Check

Check if **sems** process exists and signalling/media/rpc sockets are opened:

```
# pgrep sems
29749
# netstat -lpn | grep sems
tcp 0    0 127.0.0.1:8090 0.0.0.0:*  LISTEN 29749/sems
udp 0    0 127.0.0.1:5061 0.0.0.0:*         29749/sems
raw 2688 0 0.0.0.0:17     0.0.0.0:*  7       29749/sems
```

Check logfile */var/log/sems/sems-main.log* for possible error

7 Load balancer installation

7.1 Install packages

For installation make sure that your system have following repositories:

```
deb http://ftp.us.debian.org/debian/ wheezy main
deb http://security.debian.org/ wheezy/updates main
deb http://ftp.us.debian.org/debian/ wheezy-updates main
deb http://pkg.yeti-switch.org/debian wheezy/
deb http://packages.dotdeb.org wheezy all
deb http://deb.kamailio.org/kamailio wheezy main
```

import public keys for repositories:

```
# apt-key adv --keyserver keys.gnupg.net --recv-key 9CEBFFC569A832B6
# apt-key adv --keyserver keys.gnupg.net --recv-key E9C74FEEA2098A6E
# apt-key adv --keyserver keys.gnupg.net --recv-key FB40D3E6508EA4C8
```

Instal package:

```
# aptitude install yeti-lb
```

Note: On package configuration stage you will be asked specify address of previously installed signalling node and address for load balancer to listen.

After installation you can change any parameters by editing files: */etc/kamailio/dispatcher.list* and */etc/kamailio/lb.conf*

7.2 Launch

Launch load balancer:

```
# /etc/init.d/kamailio start
```

7.3 Check

Check kamailio running and listening desired sockets:

```
# pgrep kamailio
30853
30854
30855
30856
30857
# netstat -lpn | grep kamailio
tcp 0 0 127.0.0.1:5060 0.0.0.0:* LISTEN 30857/kamailio
udp 0 0 127.0.0.1:5060 0.0.0.0:* 30853/kamailio
raw 0 0 0.0.0.0:255 0.0.0.0:* 7 30853/kamailio
unix 2 [ ACC ] STREAM LISTENING 2673337 30856/kamailio /var/run/kamailio//kamailio_ctl
```

Check for /var/log/syslog on possible errors.

Also you can run daemon in foreground with logging to stderr for debugging purposes:

```
# kamailio -eED /etc/kamailio/kamailio.cfg
```