YETI installation manual

Contents

1 Management interface installation

we are recommend to use a separate host for web-interface and database

(placing web interface at the same host with database will increase performance by reducing network delay for database requests from web-interface)

Server requirements:

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

1.1 Repositaries

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
deb http://deb.kamailio.org/kamailio wheezy main
```

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

1.2 Packages

Install the following packages:

- yeti-web
- postgresql-9.3
- postgresql-9.3-prefix
- postgresql-contrib-9.3
- skytools3-ticker
- skytools3
- postgresql-9.3-pgq3

```
root@yeti-main:~# aptitude update && aptitude install yeti-web postgresql-9.3 postgresql-9.3-prefix
    postgresql-contrib-9.3 skytools3-ticker skytools3 postgresql-9.3-pgq3
```

2 Databases

2.1 Create Databases

Create routing database:

```
root@yeti-main:~# su - postgres
postgres@yeti-main:~$ psql
psql (9.3.5)
postgres=# create user yeti encrypted password 'somepassword' superuser;
CREATE ROLE
postgres=# create database yeti owner yeti;
CREATE DATABASE
postgres=# \q
```

Create databse to store CDR:

```
root@yeti-main:~# su - postgres
postgres@yeti-main:~$ 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

2.2 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

2.3 Init databases data

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

To initialize empty databases:

```
root@yeti-main:~# yeti-db init
root@yeti-main:~# yeti-db --cdr init
```

To upgrade database to the lastest version:

```
root@yeti-main:~# yeti-db apply_all
root@yeti-main:~# yeti-db --cdr apply_all
```

You can check actual database versions:

```
root@yeti-main:~# yeti-db version
root@yeti-main:~# 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 Launch web-interface and CDR processing daemons

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

```
root@yeti-main:~# /etc/init.d/yeti-web start
root@yeti-main:~# /etc/init.d/yeti-cdr-billing start
root@yeti-main:~# /etc/init.d/yeti-delayed start
```

4 DSS Storage installation (Redis)

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.

Redis server must be installed at the same host with switch. 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://apt.postgresql.org/pub/repos/apt/ wheezy-pgdg main
```

Install package **redis-server**

```
root@yeti-main:~# aptitude install redis-server
```

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 package

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://apt.postgresql.org/pub/repos/apt/ wheezy-pgdg main
```

Install package **yeti-mgmt**

```
# aptitude install yeti-mgmt
```

5.2 Configuration files

5.2.1 /etc/yeti/management.cfg

This file contains configuration 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://127.0.0.1:4444"
  }
  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
      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

Use following command to launch configured management server instance

```
root@yeti-main:~# /etc/init.d/yeti-mgmt start
```

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

6 Traffic switch server installation

6.1 Install package

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://apt.postgresql.org/pub/repos/apt/ wheezy-pgdg main
```

Install package **sems-yeti**

```
# 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:

```
root@yeti-main:~# cd /etc/sems/etc
root@yeti-main:~# mv /etc/sems/etc/sbc.dist.conf /etc/sems/etc/sbc.conf
root@yeti-main:~# mv /etc/sems/etc/oodprofile.yeti.dist.conf /etc/sems/etc/oodprofile.yeti.conf
root@yeti-main:~# mv /etc/sems/etc/xmlrpc2di.dist.conf /etc/sems/etc/xmlrpc2di.conf
```

6.3 Launch traffic switch

Use following command to launch configured traffic switch instance

```
root@yeti-main:~# /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

7 Install load balancer

Instal package **yeti-lb**

```
root@yeti-main:~# yeti-lb 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*

To launch load balancer:

```
root@yeti-main:~# /etc/init.d/kamailio start
```