Инструмент планирования, тестирования управления качеством и версионного контроля исходного кода SberTrack

Руководство по установке

С доступом в интернет

Установка инструмента производилась на ВМ с операционной системой Alt Linux 8.2, 8.4.

Для установки необходимо:

1. Исправить репозитории для установки ПО:

   vim /etc/apt/sources.list.d/altsp.list

   #

   rpm http://alt8.mirror.v-serv.ru/ Sisyphus/x86_64 classic

   rpm http://alt8.mirror.v-serv.ru/ Sisyphus/x86_64-i586 classic

   rpm http://alt8.mirror.v-serv.ru/ Sisyphus/noarch classic

   #
2. Обновить пакеты в системе:

   apt-get update

   apt-get dist-upgrade
3. Удостовериться в наличии требуемой версии необходимых пакетов:

   apt-cache policy ruby

   ruby:

    Installed: 2.7.6-alt1:sisyphus+296637.220.56.4@1654656369

    Candidate: 2.7.6-alt1:sisyphus+296637.220.56.4@1654656369

    Version Table:

   *** 2.7.6-alt1:sisyphus+296637.220.56.4@1654656369 0

         500 http://alt8.mirror.v-serv.ru Sisyphus/x86_64/classic pkglist

   100 RPM Database

   и произвести установку необходимых для продолжения пакетов:

   apt-get install -y ruby GraphicsMagick postfix perl-Image-ExifTool golang apf node npm postgresql13-server libruby-devel gcc-c++ libgpgme-devel libicu-devel libre2 libre2-devel libpq5-devel cmake libcurl-devel libpcre2-devel yarn nginx

   Postgresql так же необходим локально при установке БД на выделенный сервер или кластер.
4. Добавить пользователя git

   useradd git

   visudo

   git ALL=(ALL) NOPASSWD: ALL
5. Скопировать или склонировать ветку мастер из репозитория:

   scp -P<ssh_port> ./<local_file> gr_prom@<ip_gitlab>/tmp/

Распаковать, в случае копирования архивом, или просто переместить дистрибутив в папку /home/git

tar -zxvf gitlab-pv.tar.gz

mv gitlab-pv /home/git/gitlab

6. При наличии папки /home/git/gitlab/ee ее необходимо удалить

   rm -rf /home/git/gitlab/ee

7. При отсутствии в системе пакета git версии 2.33.+, его необходимо установить:

   git clone https://gitlab.com/gitlab-org/gitaly.git /tmp/gitaly

   cd /tmp/gitaly

   make git GIT_PREFIX=/usr/local

   так же нужно будет поправить конфигурационный файл гитлаба /home/git/gitlab/config/gitlab.yml и указать путь до скомпилированного пакета git: /usr/local/bin/git

8. Инициализация БД
   В случае локальной установки необходимо выполнять все команды на локальной машине, в случай выделенной ВМ(кластера) под БД, команды необходимо выполнять на удаленной машине.

   /etc/init.d/postgresql initdb

   #(если установлен кластер, он уже запущен и выполнять следующие команды нет необходимости)

   service postgresql start

   service postgresql enable

9. Создание БД и пользователя для работы с базой:

   su - postgres -s /bin/bash

   psql -h<db_ip/vip> -p5433 -d template1 -c "CREATE USER git CREATEDB;"

   #CREATE ROLE

   psql -h<db_ip/vip> -p5433 -d template1 -c "CREATE EXTENSION IF NOT EXISTS pg_trgm;"

   #CREATE EXTENSION

   psql -h<db_ip/vip> -p5433 -d template1 -c "CREATE EXTENSION IF NOT EXISTS btree_gist;"

   #CREATE EXTENSION

   psql -h<db_ip/vip> -p5433 -d template1 -c "CREATE DATABASE gitlabhq_production OWNER git;"

   #CREATE DATABASE

   Если команды по созданию экстеншинов выполняются с ошибкой в системе отсутствует пакет postgresql12-contrib его необходимо доустановить.

10. Локальная установка БД Redis версии 5+:

    apt-get install redis

    sudo cp /etc/redis/redis.conf /etc/redis/redis.conf.orig

```
sudo sed 's/^port .*/port 0/' /etc/redis/redis.conf.orig | sudo tee /etc/redis/redis.conf
echo 'unixsocket /var/run/redis/redis.sock' | sudo tee -a /etc/redis/redis.conf
echo 'unixsocketperm 770' | sudo tee -a /etc/redis/redis.conf

useradd redis
usermod -aG redis git

/usr/sbin/redis-server /etc/redis/redis.conf

chmod 777 /var/run/redis/redis.sock
```

11. Установка БД Redis на выделенную ВМ версии 5+:

```
apt-get install redis
vim /etc/redis/redis.conf
#
bind <redis_ip>
requirepass <password>
#
```

12. Для установки Redis в кластере рекомендую использовать ансибл роль:
13. Копирование конфигурационных файлов и настройка git:

```
cd /home/git/gitlab
sudo -u git -H cp config/secrets.yml.example config/secrets.yml
sudo -u git -H chmod 0600 config/secrets.yml
sudo chown -R git log/
sudo chown -R git tmp/
sudo chmod -R u+rwX,go-w log/
sudo chmod -R u+rwX tmp/
sudo chmod -R u+rwX tmp/pids/
sudo chmod -R u+rwX tmp/sockets/
sudo -u git -H mkdir -p public/uploads/
sudo chmod -R u+rwX builds/
sudo chmod -R u+rwX shared/artifacts/
sudo chmod -R ug+rwX shared/pages/
sudo -u git -H cp config/puma.rb.example config/puma.rb
sudo -u git -H git config --global gc.auto 0
sudo -u git -H git config --global repack.writeBitmaps true
```

sudo -u git -H git config --global receive.advertisePushOptions true
sudo -u git -H git config --global core.fsyncObjectFiles true
sudo -u git -H cp config/resque.yml.example config/resque.yml
sudo -u git -H cp config/cable.yml.example config/cable.yml
sudo -u git -H cp config/database.yml.postgresql config/database.yml
sudo -u git -H chmod o-rwx config/database.yml
sudo -u git -H bundle config set --local deployment 'true'
sudo -u git -H bundle config set --local without 'development test mysql aws kerberos'

14. Установка гем-зависимостей:
    cd /home/git/gitlab
    sudo -u git -H vim Gemfile

    # GPG
    #gem 'gpgme', '~> 2.0.19'   #закомментировать эту строчку

    sudo -u git -H bundle config set --local deployment 'false'
    gem install gpgme -- --use-system-libraries
    Проверка конфигурации:
    cd /home/git/gitlab
    sudo -u git -H vim .bundle/config

    BUNDLE_DEPLOYMENT: "false"
    BUNDLE_WITHOUT: "development:test:mysql:aws:kerberos"
    Установка гем:
    bundle install
    sudo -u git -H bundle install
15. Перенос папки репозиториев на отдельный диск:
    mkdir /u01/repositories
    chown -R git: /u01/repositories
16. Правка конфигурационных файлов:
    sudo -u git -H vim /home/git/gitlab/config/gitlab.yml
    production: &base
      gitlab:
        host: <ip_gitlab>

```
    port: 80
    external_url: https://<fqdn_haproxy>/stage-ci
    relative_url_root: /stage-ci
    ssh_host: <ip_gitlab>
    ssh_user: git
    ssh_port: 9022
    trusted_proxies:
      - <ip_haproxy>/24
    email_enabled: true
    email_from: gostech@sbermail-cloud.sber.ru
    email_display_name: GitLab
    email_reply_to: noreply@sbermail-cloud.sber.ru
    email_subject_suffix: ''
    email_smime:
  repositories:
    storages:
      default:
path: /u01/repositories/ ###
sudo -u git -H vim /home/git/gitlab/config/database.yml

production:
  main:
    adapter: postgresql
    encoding: unicode
    database: gitlabhq_production
    username: git
    password: "secure password"
    #host: localhost # при локальной установке БД
    host: <ip_vip_db>
    port: <port_db>
sudo -u git -H vim /home/git/gitlab/config/resque.yml

production:
  #url: unix:/var/run/redis/redis.sock   #при локальной установке
```

url: redis://:requepass@<ip_redis>:6379 #rediss с двумя s для ssl соединения, требует дополнительной настройки

sudo -u git -H vim /home/git/gitlab/config/cable.yml

production:
adapter: redis
#url: unix:/var/run/redis/redis.sock
url: redis://:requepass@<ip_redis>:6379
channel_prefix: gitlab_production

17. Настройка сервера NFS:

apt-get install nfs-utils

mkdir /u01/nfs
chown nobody: /u01/nfs
echo "/u01/nfs *(rw,sync,all_squash,insecure,fsid=0)" >> /etc/exports

18. Подключение к NFS:

mount.nfs4 <ip_nfs>:/ /mnt
sudo -u git -H mkdir /mnt/stageci
sudo -u git -H mv /home/git/.ssh /mnt/stageci/.ssh
sudo -u git -H mv /home/git/gitlab/public/uploads /mnt/stageci/gitlab/uploads
sudo -u git -H mv /home/git/gitlab/shared /mnt/stageci/gitlab/shared
sudo -u git -H mv /home/git/gitlab/builds /mnt/stageci/gitlab/builds
umount /mnt

vim /etc/fstab
<ip_nfs>:/stageci/.ssh /home/git/.ssh nfs4 defaults 0 0
<ip_nfs>:/stageci/gitlab/uploads /home/git/gitlab/public/uploads nfs4 defaults 0 0
<ip_nfs>:/stageci/gitlab/shared /home/git/gitlab/shared nfs4 defaults 0 0
<ip_nfs>:/stageci/gitlab/builds /home/git/gitlab/builds nfs4 defaults 0 0

mount -a

19. Установка gitlab-shell

sudo -u git -H bundle exec rake gitlab:shell:install RAILS_ENV=production

20. Установка gitlab-workhorse

sudo -u git -H bundle exec rake "gitlab:workhorse:install[/home/git/gitlab-workhorse]" RAILS_ENV=production

21. Установка gitlab-pages

    cd /home/git

    sudo -u git -H git clone https://gitlab.com/gitlab-org/gitlab-pages.git

    cd gitlab-pages

    sudo -u git -H git checkout v$(</home/git/gitlab/GITLAB_PAGES_VERSION)

    sudo -u git -H make

22. Установка gitaly

    cd /home/git/gitlab

    sudo -u git -H bundle exec rake "gitlab:gitaly:install[/home/git/gitaly,/home/git/repositories]" RAILS_ENV=production

    sudo chmod 0700 /home/git/gitlab/tmp/sockets/private

    sudo chown git /home/git/gitlab/tmp/sockets/private

23. Запуск gitaly

    gitlab_path=/home/git/gitlab

    gitaly_path=/home/git/gitaly

    sudo -u git -H sh -c "$gitlab_path/bin/daemon_with_pidfile $gitlab_path/tmp/pids/gitaly.pid $gitaly_path/_build/bin/gitaly $gitaly_path/config.toml >> $gitlab_path/log/gitaly.log 2>&1 &"

24. Копирование скриптов запуска gitlab

    cd /home/git/gitlab

    sudo cp lib/support/init.d/gitlab /etc/init.d/gitlab

    sudo cp lib/support/init.d/gitlab.default.example /etc/default/gitlab

    sudo cp lib/support/logrotate/gitlab /etc/logrotate.d/gitlab

25. Миграция БД и формирование файла db/structure.sql

    sudo -u git -H bundle exec bin/rails db:migrate RAILS_ENV=production

    sudo -u git -H bundle exec bin/rails db:structure:dump RAILS_ENV=production

26. Установка gitlab

    sudo -u git -H bundle exec rake gitlab:setup RAILS_ENV=production GITLAB_ROOT_PASSWORD="password_for_root" DISABLE_DATABASE_ENVIRONMENT_CHECK=1

27. Проверка переменных

    sudo -u git -H bundle exec rake gitlab:env:info RAILS_ENV=production

    Если не отображается версия redis при удаленном исполнении, все равно работает.

28. Установка gettext

    sudo -u git -H bundle exec rake gettext:compile RAILS_ENV=production

29. Компиляция ресурсов

    sudo -u git -H yarn install --production —pure-lockfile

    sudo -u git -H bundle exec rake gitlab:assets:compile RAILS_ENV=production NODE_ENV=production

30. Конфигурация nginx

    cp lib/support/nginx/gitlab /etc/nginx/sites-available.d/gitlab_nossl.conf

    ln -s /etc/nginx/sites-available.d/gitlab_nossl.conf /etc/nginx/sites-enabled.d/gitlab_nossl.conf

    cd /etc/nginx/sites-available.d/

    vim gitlab_nossl.conf

    #необходимо поправить следующие параметры

    listen <ip_gitlab>:80;

    server_name <fqdn_haproxy>;

    location /stage-ci { ###location

    proxy_pass http://gitlab-workhorse; #проверить

    root /home/git/gitlab/public; #проверить

    sudo nginx -t

    usermod -aG git _nginx

    chmod 755 /home/git

    systemctl start nginx && systemctl enable nginx.service

31. Конфигурация haproxy

    vim /etc/haproxy/haproxy.cfg

    #

    acl gitlab-acl path_beg -i /stage-ci #location

    #

    use_backend gitlab if gitlab-acl

    #

    backend gitlab

    server static <ip_of_gitlab>:80

32. Проверка конфигурации

    sudo -u git -H vim /home/git/gitaly/config.toml

    url = "http://<ip_gitlab_fqdn_haproxy>/stage-ci"

    sudo -u git -H vim /home/git/gitlab/config/gitlab.yml

    production: &base

    gitlab:

    host: <ip_gitlab>

    port: 80

    external_url: https://<fqdn_haproxy>/stage-ci

    relative_url_root: /stage-ci

    ssh_port: 9022

trusted_proxies:

- <ip_haproxy>/24

sudo vim /etc/default/gitlab

gitlab_workhorse_options="-listenUmask 0 -listenNetwork unix -listenAddr $socket_path/gitlab-workhorse.socket -authBackend http://<ip_gitlab>:8080/stage-ci -authSocket $socket_path/gitlab.socket -documentRoot $app_root/public"

sudo -u git -H cp /home/git/gitlab/config/initializers/relative_url.rb.sample /home/git/gitlab/config/initializers/relative_url.rb

sudo -u git -H vim /home/git/gitlab/config/initializers/relative_url.rb

#

Rails.application.configure do

config.relative_url_root = "/stage-ci"

end

#

vim /home/git/gitlab-shell/config.yml

---

user: git

gitlab_url: https://<fqdn_haproxy>/stage-ci/

http_settings:

self_signed_cert: false

auth_file: "/home/git/.ssh/authorized_keys"

log_level: INFO

audit_usernames: false

sudo -u git -H vim /home/git/gitlab/config/puma.rb

workers 3

ENV['RAILS_RELATIVE_URL_ROOT'] = "/stage-ci"

require_relative "/home/git/gitlab/lib/gitlab/cluster/lifecycle_events"

33. Финальная проверка

sudo -u git -H bundle exec rake gitlab:check RAILS_ENV=production

в выводе могут быть ошибки и команды для их устранения

sudo -u git -H RAILS_ENV=production bin/background_jobs start

#

sudo chmod 700 /home/git/gitlab/public/uploads

#

mkdir ~/gitlab-check-backup-1648903642

sudo mv /home/git/.ssh/config ~/gitlab-check-backup-1648903642

#

gitlab_path=/home/git/gitlab

gitaly_path=/home/git/gitaly

sudo -u git -H sh -c "$gitlab_path/bin/daemon_with_pidfile $gitlab_path/tmp/pids/gitaly.pid $gitaly_path/_build/bin/gitaly $gitaly_path/config.toml >> $gitlab_path/log/gitaly.log 2>&1 &"

Запуск gitlab

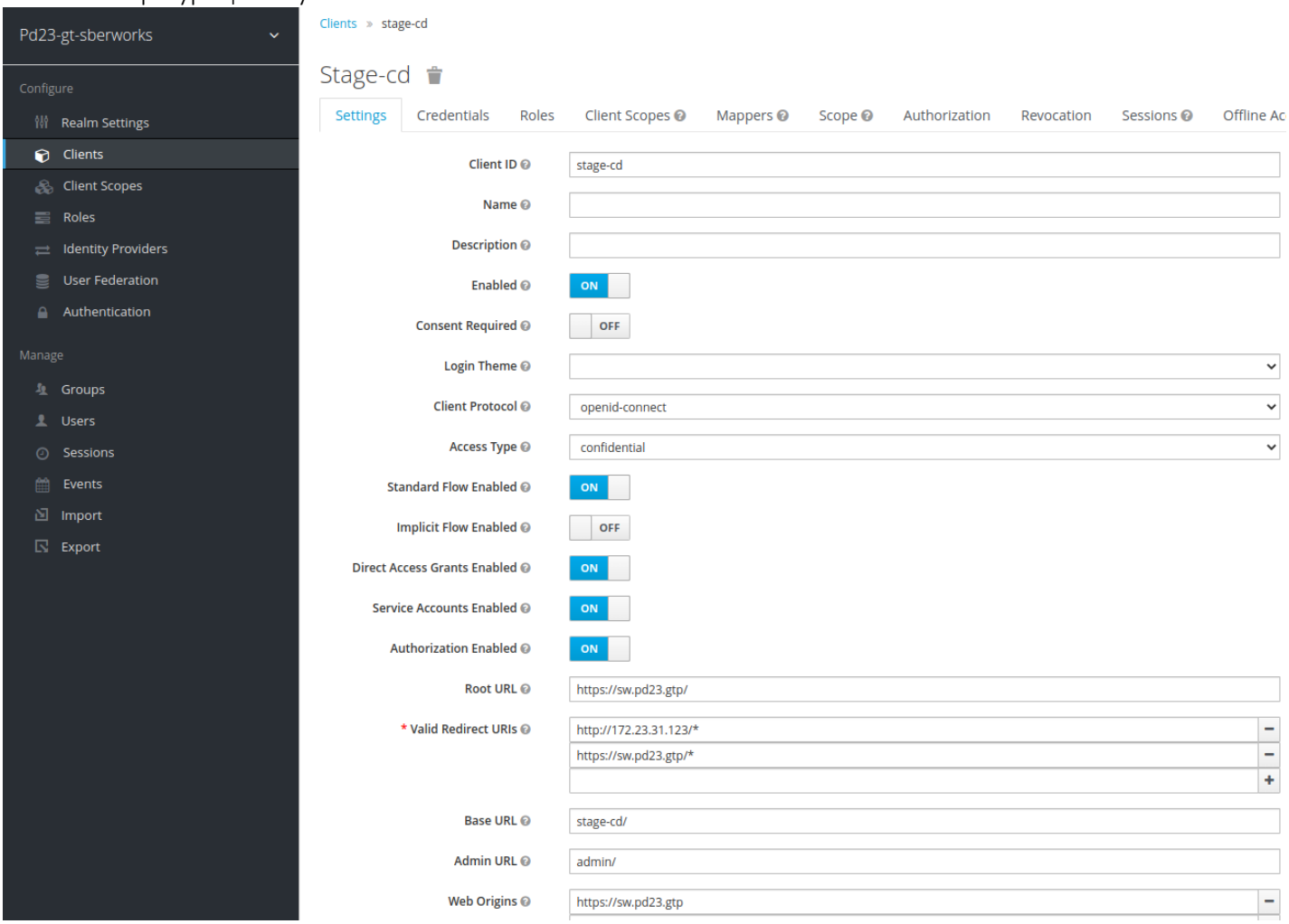
/etc/init.d/gitlab start

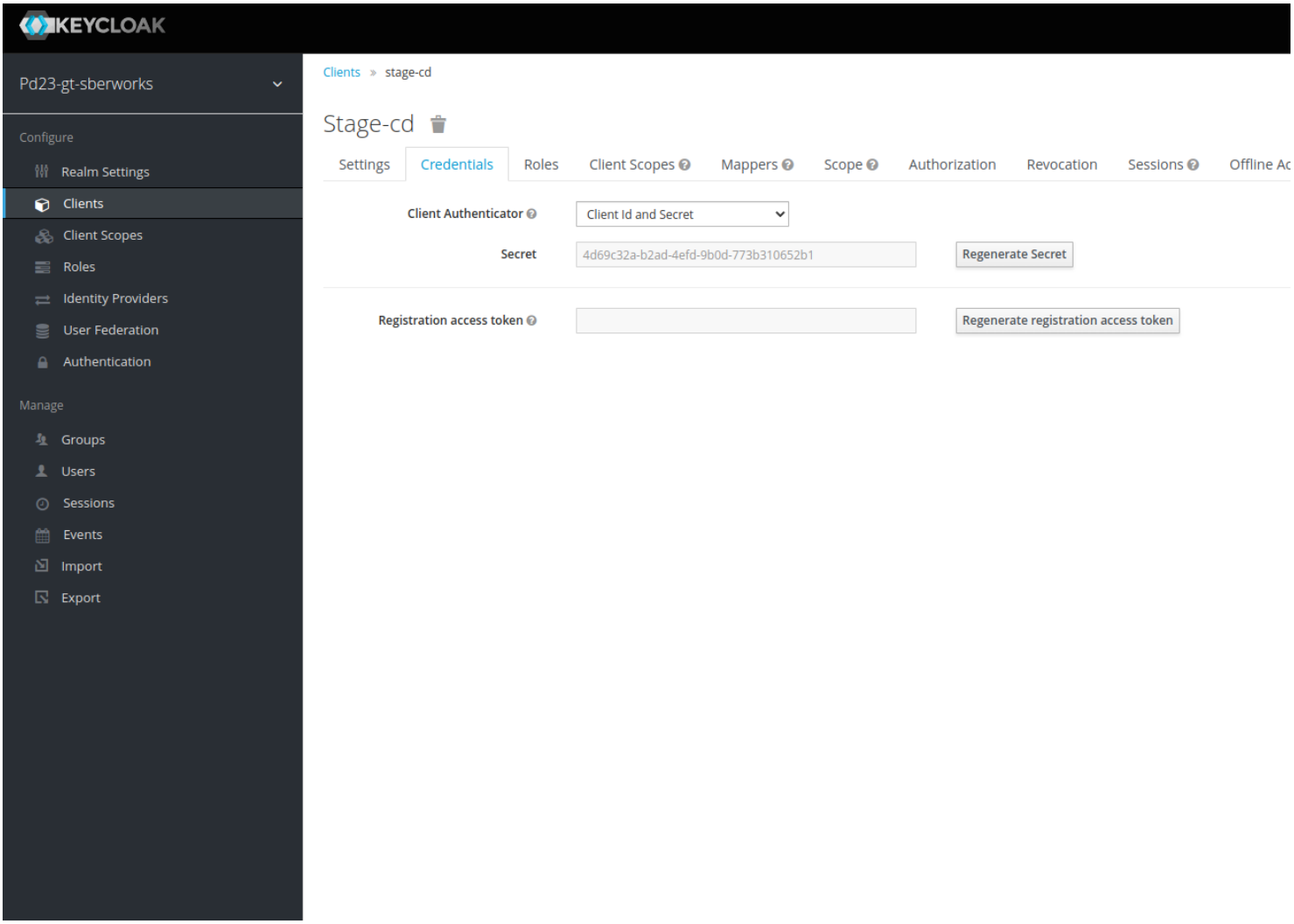
export EDITOR=vim crontab -e

*/2 * * * * (ps -ef|grep -v grep|grep -iq gitlab-workhorse) || (date; /usr/sbin/redis-server /etc/redis/redis.conf; sleep 10; chmod 777 /var/run/redis/redis.sock ; sleep 10; /etc/init.d/gitlab status; /etc/init.d/gitlab stop; sleep 10; /etc/init.d/gitlab start;echo 'gitlab restarted!' | systemd-cat) >> /tmp/gitlab_reboots.log

34. Настройка аутентификации через keycloak

Конфигурация keycloak





35. Конфигурация gitlab

```
omniauth:
  enabled: true
  allow_single_sign_on: ["openid_connect"]
  block_auto_created_users: false
  auto_link_ldap_user: false
  auto_link_saml_user: false
  saml_message_max_byte_size: 250000
  external_providers: []
  providers:
   - {
       name: 'openid_connect',
       label: 'Keycloak',
       args: {
```

```
        name: 'openid_connect',
        scope: ["openid", "profile", "email"],
        response_type: "code",
        issuer:  "https://<fqdn_haproxy>/auth/realms/<realm>",
        discovery: true,
        client_auth_method: "query",
        uid_field: "preferred_username",
        send_scope_to_token_endpoint: "false",
        client_options: {
          identifier: "<client_from_keycloak>",
          secret: "<secret_key_from_keycloak>",
          redirect_uri: "https://<fqdn_haproxy>/<location_gitlab>/users/auth/openid_connect/callback"
        }
      }
}
```

36. Если используется самоподписанный сертификат, необходимо внести изменения в gitlab

```
vim /home/git/gitlab/config/initializers/omniauth.rb
# Добавить. Отключение проверки сертификатов, при валидном сертификате необходимо убрать!
OpenIDConnect.http_config do |config|
  config.ssl_config.verify_mode = OpenSSL::SSL::VERIFY_NONE
end
#
```