Установка

1. Устанавливаем redmine под отдельного пользователя, назовём его redmine
2. Ruby ставим через rvm
3. Сам redmine будет работать через unicorn, устанавливается вместе с gem'ами
4. СУБД будем использовать MySQL

Настройка окружения

Создаём пользователя redmine, входим в него:

```
 su redmine
```

Устанавливаем rvm:

```
curl -sSL https://get.rvm.io | bash
```

rvm сам скачивает и собирает ruby запрошенной версии и раскладывает у себя в ${HOME}/.rvm. Он умеет держать в себе разные версии ruby, какая из них будет использоваться зависит от переменных в ${HOME}/.rvm

Важно При использовании ruby установленного через rvm - очень важны переменные окружения, устанавливаемые rvm в "~/.profile"
Для корректного подключения всех переменных можно делать так:

```
[root@localhost ~]# su redmine
[redmine@localhost root]$ bash --login
```

Для сборки ruby он предложит установить следующие пакеты:

```
yum install -y patch libyaml-devel glibc-headers autoconf gcc-c++ glibc-devel patch readline-devel
 zlib-devel libffi-devel openssl-devel automake libtool bison sqlite-devel
```

+ нам потребуются:

1. ImageMagick-devel (для gem rmagick)
2. mysql-devel (для gem mysql2)

После установки пакетов устанавливаем ruby-2.4.5-railsexpress, поскольку он является наиболее производительным:

```
rvm install ruby-2.4.5-railsexpress
```

После выполнения rvm list должен иметь вид:

```
~]$ rvm list

rvm rubies

=* ruby-2.4.1-railsexpress [ x86_64 ]

# => - current
# =* - current && default
```

```
#   * - default
```

В случае установки не на чистый сервер - если установлено несколько разных версий ruby - выбираем нужный так:

```
rvm --default ruby-2.4.5-railsexpress
```

Установка redmine

Подготовка базы

Для предотвращения проблем с кодировкой - база должна быть в кодировке UTF8. При создании базы скриптом admin.sh - эта кодировка устанавливается, однако при создании базы руками - стоит указать кодировку:

```
CREATE DATABASE redminedb CHARACTER SET utf8 COLLATE utf8_general_ci;
```

Созданём пользователя базы данных redminedb и выдаём ему права на неё:

```
CREATE USER redmineuser@localhost IDENTIFIED BY 'some-strong-pass';
GRANT ALL ON redminedb.* to redmineuser@localhost;
FLUSH PRIVILEGES;
```

Установка файлов redmine

Устанавливаем под юзером redmine в /opt/redmine

```
 git clone -b 3.4-stable  https://github.com/redmine/redmine.git /opt/redmine/
```

Задаём параметры подключения к базе в файле /opt/redmine/config/database.yml:

```
production:
  adapter: mysql2
  database: <название бд>
  host: localhost
  username: <имя пользователя бд>
  password: "<пароль>"
  encoding: utf8
```

Без настроек подключения к базе некоторые gem'ы, в данном случае mysql2, собраны не будут

+ Заранее создадим директории для pid-файлов и сокетов:

```
mkdir /opt/redmine/tmp/pids
mkdir /opt/redmine/tmp/sockets
```

Установка gem'ов

Для начала нам потребуется особый gem, служащий для установки других gem'ов - bundler

```
gem install bundler
```

Список нужных gem'ов для работы redmine находится в файле /opt/redmine/Gemfile. Для добавления нестандартных gem'ов - их нужно записать в файл /opt/redmine/Gemfile.local
Мы добавим:

```
gem 'unicorn'
gem 'dalli'
```

unicorn - сервер, который будет обрабатывать запросы к redmine
dalli - позволяет зранить сессии и кеш у memcache

Далее - выполняем установку всех нужных gem'ов через bundle:

```
bundle install --without development test sqlite postgresql --path vendor/bundle
```

Опция --path vendor/bundle - указывает расположить все установленные gem'ы в директорию /opt/redmine/vendor/bundle, что весьма полезно при обновлении

Если в системе или в rvm стоит несколько ruby - имеет смысл убедиться, что вызывается нужный bundle:

```
 which bundle
```

должен выдать:

```
 ~/.rvm/gems/ruby-2.4.5-railsexpress/bin/bundle
```

Настройка redmine

Генерируем ключ для шифрования cookie:

```
bundle exec rake generate_secret_token
```

Эта команда создаст файл: /opt/redmine/config/initializers/secret_token.rb , важно что бы этот файл был

Далее инициализируем базу:

```
RAILS_ENV=production bundle exec rake db:migrate
```

Установка названий по умолчанию:

```
RAILS_ENV=production bundle exec rake redmine:load_default_data
```

будет запрошено для какого языка загрузить данные, выбираем ru
Если на этом шаге возникли ошибки - скорее всего проблема с кодировкой базы/таблиц.

Создаём файл /opt/redmine/config/additional_environment.rb со следующим {{collapse(содержимым)

```
config.action_controller.perform_caching = true
config.action_dispatch.x_sendfile_header = 'X-Accel-Redirect'

config.redmine_search_cache_store = :dalli_store
config.cache_store = :dalli_store
config.session_store = :dalli_store

Rails.application.config.session_store ActionDispatch::Session::CacheStore, :expire_after => 31.da
ys, :memcache_server => ['127.0.0.1']
```

}}
Устанавливаем **memcache** переводим его на localhost и запускаем
Подробнее об этом написано в разделе **Тюнинг**

Настройка Unicorn, Nginx

Для работы unicorn требуется создать конфиг unicorn с {{collapse(типовым содержимым)
/opt/redmine/config/unicorn.rb

```
worker_processes 4
working_directory "/opt/redmine/"
timeout 30
listen "/opt/redmine/tmp/sockets/redmine.sock", :backlog => 2048

pid "/opt/redmine/tmp/pids/unicorn.pid"

stderr_path "/opt/redmine/log/unicorn.stderr.log"
stdout_path "/opt/redmine/log/unicorn.stdout.log"

preload_app true
GC.respond_to?(:copy_on_write_friendly=) and
  GC.copy_on_write_friendly = true

before_fork do |server, worker|
defined?(ActiveRecord::Base) and
ActiveRecord::Base.connection.disconnect!
end

after_fork do |server, worker|
defined?(ActiveRecord::Base) and
ActiveRecord::Base.establish_connection
end
```

}}

Шаблон init.d-скрипта для redmine с unicorn выглядит {{collapse(так)
/etc/init.d/redmine

```
#!/bin/sh
#
# redmine        Starts redmine
#
# chkconfig:   - 85 15
# description: redmine daemon
```

```
# Source function library.
. /etc/init.d/functions

set -e
# Example init script, this can be used with nginx, too,
# since nginx and unicorn accept the same signals

# Feel free to change any of the following variables for your app:
TIMEOUT=${TIMEOUT-60}
USERNAME=redmine
APP_ROOT=/opt/redmine
PID=$APP_ROOT/tmp/pids/unicorn.pid
ENV=production
source "/home/redmine/.rvm/scripts/rvm"
CMD="bundle exec unicorn -D -E $ENV -c config/unicorn.rb"
action="$1"
set -u

old_pid="$PID.oldbin"

cd $APP_ROOT || exit 1

sig () {
        test -s "$PID" && kill -$1 `cat $PID`
}

oldsig () {
        test -s $old_pid && kill -$1 `cat $old_pid`
}

case $action in
start)
        sig 0 && echo >&2 "Already running" && exit 0
        su -l $USERNAME -c "cd $APP_ROOT && $CMD"
        ;;
stop)
        sig QUIT && exit 0
        echo >&2 "Not running"
        ;;
force-stop)
        sig TERM && exit 0
        echo >&2 "Not running"
        ;;
restart|reload)
        sig HUP && echo reloaded OK && exit 0
        echo >&2 "Couldn't reload, starting '$CMD' instead"
        su -l $USERNAME -c "cd $APP_ROOT && $CMD"
        ;;
upgrade)
        if sig USR2 && sleep 2 && sig 0 && oldsig QUIT
        then
                n=$TIMEOUT
                while test -s $old_pid && test $n -ge 0
                do
                        printf '.' && sleep 1 && n=$(( $n - 1 ))
                done
                echo

                if test $n -lt 0 && test -s $old_pid
                then
                        echo >&2 "$old_pid still exists after $TIMEOUT seconds"
                        exit 1
                fi
                exit 0
        fi
        echo >&2 "Couldn't upgrade, starting '$CMD' instead"
```

```
        su -l $USERNAME -c "cd $APP_ROOT && $CMD"
        ;;
reopen-logs)
        sig USR1
        ;;
*)
        echo >&2 "Usage: $0 <start|stop|restart|upgrade|force-stop|reopen-logs>"
        exit 1
        ;;
esac
```

}} Unit-файл unicorn для {{collapse(CentOS7)
/usr/lib/systemd/system/redmine.service

```
[Unit]
Description=Redmine Unicorn Server
Wants=mariadb.service
After=mariadb.service nginx.service

[Service]
User=redmine
WorkingDirectory=/opt/redmine
EnvironmentFile=/home/redmine/redmine.env
SyslogIdentifier=redmine
PIDFile=/opt/redmine/tmp/pids/unicorn.pid
ExecStart=/usr/bin/env bundle exec unicorn -c config/unicorn.rb -E $RAILS_ENV -D
[Install]
WantedBy=multi-user.target
```

Стоит обратить внимание на строки:

```
Wants=mariadb.service
After=mariadb.service nginx.service
```

По умолчанию в CentOS7 вместо MySQL идёт MariaDB, в случае если используется другая вариация СУБД - следует заменить mariadb.service на имя соответвующего СУБД сервиса

Важно systemd не съест переменные окружения из /home/redmine/.rvm/scripts/rvm, для того, что бы их задать - следует создать файл (/home/redmine/redmine.env) с этими переменными (для наших целей достаточно переменной PATH):

```
su redmine
bash --login
printenv  | grep -iE 'path' > /home/redmine/redmine.env
echo "RAILS_ENV=production" >> /home/redmine/redmine.env #туда же занесём RAILS_ENV
```

Либо из под root:

```
su redmine -l -c "printenv  | grep -iE 'path' > /home/redmine/redmine.env" ; echo "RAILS_ENV=produ
ction" >> /home/redmine/redmine.env
```

Перед запуском потребуется выполнить systemctl daemon-reload, что бы systemd подтянул новый unit-файл
}}

И шаблон конфига nginx для хоста выглядит {{collapse(так)

```
upstream redmine {
```

```
    server unix:/opt/redmine/tmp/sockets/redmine.sock fail_timeout=0;
}

server {
    listen 80;
    server_name redmine.server.name;

    location / {
        rewrite ^(.*) https://redmine.server.name$1 permanent;
    }
}

server {
    listen 443 ssl http2;
    server_name redmine.server.name;

    charset utf-8;
    client_max_body_size 1G;

    ssl_session_timeout 90m;

    access_log /opt/redmine/log/nginx.access.log main;
    error_log /opt/redmine/log/nginx.error.log;

    ssl_certificate /etc/nginx/ssl/redmine.crt;
    ssl_certificate_key /etc/nginx/ssl/redmine.key;

    rewrite ^/articles/show$ /knowledgebase/articles/$arg_article_id;

    location / {
        root /opt/redmine/public;
        try_files $uri @redmine;
        expires 7d;
    }

    location ^~ /attachments/download {
        proxy_set_header X-Sendfile-Type X-Accel-Redirect;
        proxy_set_header X-Accel-Mapping /opt/redmine/files=/files;

        proxy_pass http://redmine;
        proxy_set_header Proxy "";
        proxy_redirect off;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;

        proxy_read_timeout 1800;
        proxy_buffer_size 16k;
        proxy_buffers 32 16k;
    }

    location ^~ /files {
        root /opt/redmine;
        internal;
    }

    location @redmine {
        proxy_pass http://redmine;
        proxy_set_header Proxy "";
        proxy_redirect off;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;

        proxy_read_timeout 1800;
        proxy_buffer_size 16k;
        proxy_buffers 32 16k;
```

```
    }

    error_page 500 502 503 504 /500.html;
}
```

}}

На этом установка самого redmine завершена.

Установка плагинов

Установка плагинов к redmine тривиальна. Для каждого плагина есть набор инструкций, для его установки и перед работами следует с ними ознакомиться.
Но как правило установка плагинов сводится к следующим пунктам:

- загрузить плагин в директорию plugins (в нашем случае /opt/redmine/plugins), если плагин в архиве - распаковать.
- под пользователем redmine выполнить (не забываем про bash --login):
  1. RAILS_ENV=production bundle exec rake redmine:plugins:migrate
  2. RAILS_ENV=production bundle exec rake redmine:plugins:assets

Если для плагина требуется установка дополнительного gem'a - слеудет этот gem добавить в Gemfile.local, и удалить **Gemfile.lock** перед выполнением bundle install --without ...
Всё, плагин установлен.

Установка нашего плагина sidekiq

Для нашего плагина sidekiq, требуется:

1. склонировать в /opt/redmine/plugins/ сам плагин:

   ```
   cd /opt/redmine/plugins
   git clone https://github.com/ogom/redmine_sidekiq
   ```

2. устанавливаем gem 'sidekiq-cron' (занести в Gemfile.local redmine'a, и повторить bundle install предварительно удалив файл **Gemfile.lock**)
3. установливаем redis (не забыть проверить что бы redis висел на localhost)
4. заполнить {{collapse(конфиг)
   /opt/redmine/config/sidekiq.yml

   ```
   ---
   :pidfile: tmp/pids/sidekiq.pid
   :logfile: log/sidekiq.log
   :timeout: 30
   :concurrency: 25
   :queues:
     - default
     - telegram
     - index_queue
   ```

   }}
5. Добавить ещё {{collapse(один файл)
   /opt/redmine/config/initializers/zz-cron.rb

   ```
   Sidekiq::Logging.logger = Logger.new(Rails.root.join('log', 'sidekiq.log'))

   class HelpdeskEmailReceiveWorker
     include Sidekiq::Worker

     def perform
       Project.active.has_module(:contacts_helpdesk).each do |project|
         begin
           HelpdeskMailer.check_project(project.id)
   ```

```
      rescue Exception => e
        puts "Helpdesk MailHandler: can't get mail for project #{project.name} with error: #{e
.message}"
      end
    end
  end
end

array = [
  {
    'name' => 'helpdesk',
    'class' => 'HelpdeskEmailReceiveWorker',
    'cron' => '1-59/4 * * * *'
  }
]

Sidekiq::Cron::Job.load_from_array array


}}
```

{{collapse(init.d скрипт для sidekiq)
В общих чертах повторяет скрипт unicorn

```
#!/bin/sh
#
# sidekiq        Starts sidekiq for redmine
#
# chkconfig:   - 85 15
# description: sidekiq daemon

# Source function library.
. /etc/init.d/functions

set -e
# Example init script, this can be used with nginx, too,
# since nginx and unicorn accept the same signals

# Feel free to change any of the following variables for your app:
TIMEOUT=${TIMEOUT-60}
USERNAME=redmine
APP_ROOT=/opt/redmine
PID=$APP_ROOT/tmp/pids/sidekiq.pid
ENV=production
source "/home/redmine/.rvm/scripts/rvm"
CMD="bundle exec sidekiq -d -C config/sidekiq.yml"
action="$1"
set -u

old_pid="$PID.oldbin"

cd $APP_ROOT || exit 1

sig () {
        test -s "$PID" && kill -$1 `cat $PID`
}

oldsig () {
        test -s $old_pid && kill -$1 `cat $old_pid`
}

case $action in
start)
        sig 0 && echo >&2 "Already running" && exit 0
        su -l $USERNAME -c "cd $APP_ROOT && $CMD"
        ;;
```

```
stop)
        sig QUIT && exit 0
        echo >&2 "Not running"
        ;;
force-stop)
        sig TERM && exit 0
        echo >&2 "Not running"
        ;;
restart|reload)
        sig HUP && echo reloaded OK && exit 0
        echo >&2 "Couldn't reload, starting '$CMD' instead"
        su -l $USERNAME -c "cd $APP_ROOT && $CMD"
        ;;
upgrade)
        if sig USR2 && sleep 2 && sig 0 && oldsig QUIT
        then
                n=$TIMEOUT
                while test -s $old_pid && test $n -ge 0
                do
                        printf '.' && sleep 1 && n=$(( $n - 1 ))
                done
                echo

                if test $n -lt 0 && test -s $old_pid
                then
                        echo >&2 "$old_pid still exists after $TIMEOUT seconds"
                        exit 1
                fi
                exit 0
        fi
        echo >&2 "Couldn't upgrade, starting '$CMD' instead"
        su -l $USERNAME -c "cd $APP_ROOT && $CMD"
        ;;
reopen-logs)
        sig USR1
        ;;
*)
        echo >&2 "Usage: $0 <start|stop|restart|upgrade|force-stop|reopen-logs>"
        exit 1
        ;;
esac
```

}}

{{collapse(CentOS7 unit-file)
/usr/lib/systemd/system/sidekiq.service

```
[Unit]
Description=Redmine Sidekiq
Wants=redis.service
After=redis.service

[Service]
User=redmine
WorkingDirectory=/opt/redmine
EnvironmentFile=/home/redmine/redmine.env
SyslogIdentifier=redmine-sidekiq
PIDFile=/opt/redmine/tmp/pids/sidekiq.pid
ExecStart=/usr/bin/env bundle exec sidekiq -C config/sidekiq.yml
[Install]
WantedBy=multi-user.target
```

В целом - аналогично unit-файлу unicorn
}}

Тюнинг

Производительный ruby

Проверяем через rvm list установленную версию ruby, если она не ruby-2.4.5-railsexpress - ставим ruby-2.4.5-railsexpress и включаем её по умолчанию

```
rvm install ruby-2.4.5-railsexpress
rvm --default ruby-2.4.5-railsexpress
```

Подключение memcache

Шаг первый - установить memcache и настроить его на localhost
Шаг второй - проверить gem 'dalli' в /opt/redmine/Gemfile.local. Если нет - добавить и выполнить под пользователем redmine:

```
bundle install --without development test sqlite postgresql --path vendor/bundle
```

Шаг третий - добавить в файл /opt/redmine/config/additional_environment.rb {{collapse(строки)

```
config.action_controller.perform_caching = true
config.redmine_search_cache_store = :dalli_store
config.cache_store = :dalli_store
config.session_store = :dalli_store

Rails.application.config.session_store ActionDispatch::Session::CacheStore, :expire_after => 31.da
ys, :memcache_server => ['127.0.0.1']
```

}}
Шаг четвёртый - перезапустить redmine

Отдача файлов nginx'ом

добавить в файл /opt/redmine/config/additional_environment.rb {{collapse(строку)

```
config.action_dispatch.x_sendfile_header = 'X-Accel-Redirect'
```

}}
Проверить наличие в конфиге nginx для хоста этих {{collapse(строк)

```
    location ^~ /attachments/download {
        proxy_set_header X-Sendfile-Type X-Accel-Redirect;
        proxy_set_header X-Accel-Mapping /opt/redmine/files=/files;

        proxy_pass http://redmine;
        proxy_set_header Proxy "";
        proxy_redirect off;
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;

        proxy_read_timeout 1800;
        proxy_buffer_size 16k;
        proxy_buffers 32 16k;
    }
```

```
    location ^~ /files {
        root /opt/redmine;
        internal;
    }
```

}}

Чистка временных файлов

Записать в /etc/cron.d/redmine {{collapse(эти задания)

```
00 05 * * * root su -l redmine -c "cd /opt/redmine && bundle exec rake -f /opt/redmine/Rakefile re
dmine:tokens:prune RAILS_ENV='production' > /dev/null 2>&1"
05 05 * * * root su -l redmine -c "cd /opt/redmine && bundle exec rake -f /opt/redmine/Rakefile re
dmine:attachments:prune RAILS_ENV='production' > /dev/null 2>&1"
10 05 * * * root su -l redmine -c "cd /opt/redmine && bundle exec rake -f /opt/redmine/Rakefile re
dmine:watchers:prune RAILS_ENV='production' > /dev/null 2>&1"
```

}}

Количество воркеров Unicorn

Unicorn имеет возможность настройки количества своих воркеров, достигается это редактированием файла config/unicorn.rb, строки:

```
worker_processes 4
```

Обновление

1. Сделать бэкап
Данные которые следует забэкапить для возможности полного отката: директория redmine, /home/redmine (из-за .rvm), база данных
2. Обновить файлы redmine - если redmine установлен с git - сделать pull или checkout на бранч нужной версии
в противном случае - скачать новый redmine в отдельную директорию и скопировать в неё {{collapse(следующее)

1. /opt/redmine/config/database.yml
2. /opt/redmine/config/configuration.yml
3. /opt/redmine/config/unicorn.rb
4. /opt/redmine/files
5. /opt/redmine/plugins (если директория не пуста)
6. /opt/redmine/public/themes (если в директории есть что-то кроме "classic","alternate" и "README")
7. Проверить наличие и тоже перенести если эти файлы есть:

```
/opt/redmine/config/additional_environment.rb
/opt/redmine/config/initializers/zz-cron.rb # наш sidekiq
/opt/redmine/config/sidekiq.yml # наш sidekiq
```

8. Так же следует проверить /opt/redmine/config/ на наличие других не стандартных конфигов - возможно нужных для работы плагинов
   }}
   И заменить её старую директорию redmine.

3. После чего нужно удалить **vendor**/**bundle** и **Gemfile.lock** в директории redmine, И под пользователем redmine (всё ещё не забываем про rvm и bash --login) выполнить:

```
bundle install --without development test sqlite postgresql --path vendor/bundle
```

4. Затем обновляем базу, опять же под пользователем redmine:

```
RAILS_ENV=production bundle exec rake db:migrate
RAILS_ENV=production bundle exec rake redmine:plugins:migrate
RAILS_ENV=production bundle exec rake redmine:plugins:assets
```

5. В веб-панель мониторинга sidekiq можно зайти по ссылке http://REDMINE_URL/sidekiq под админом. Так же в главном меню redmine для админа появиться новая ссылка "Sidekiq"

5. Проверяем наличие config/initializers/secret_token.rb в директории с новым redmine, если его нет - генерируем командой:

```
bundle exec rake generate_secret_token
```

Возможные проблемы

Если установленный плагин на старой версии redmine не поддерживает новую - при redmine:plugins:migrate возможны ошибки, в этом случае следует обновить плагин до версии поддерживающей новый redmine.