Manual Installation Instructions

This documentation is for Release 6.3 and above. For documentation on previous releases, please select from the options below.

- Release 1.x version of this page: v.81.
- Release 2.x version of this page: v.87.
- Release 3.0-3.1 version of this page: v.111.
- Release 3.2 version of this page: v.116.
- Release 4.0 version of this page: v.143.
- Release 5.x version of this page: v.163.
- Release 6.0 version of this page: v.177

These instructions provide a recipe for building your own all-in-one Avalon system from scratch on CentOS or Red Hat Enterprise Linux, version 6.x is supported, 7.x will be supported soon. Please note that while an all-in-one installation as outlined here is certainly suitable for testing and demos, a single, all-in-one, server may not be suitable for production environments.

- Ready the Installation Environment
- Main Components
  - Fedora Commons Repository
  - Solr
  - MySQL
  - Media Streaming Server
  - FFmpeg
  - HTTPD
  - Matterhorn
  - Apache Passenger and Ruby
  - Avalon
  - Redis & Resque
- Additional Configurations
  - Dropbox
  - Batch ingest
  - Using the System
  - Restarting the Server

Ready the Installation Environment

Make sure a valid hostname is resolvable

The default hostname is “avalon.dev”, so name the machine this and enter it into `/etc/hosts`

```
# hostname
avalon.dev
# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4 avalon.dev
```

Configure iptables

The Avalon Media System requires several ports to be open to client browsers.

Here are the port settings that will need to be configured:

Port	Purpose	External?
80	HTTP (Avalon)	Yes
8983	HTTP (Solr)	No
8984	HTTP (Fedora)	No
8080	HTTP (Matterhorn)	Yes

The preferred method is to create a shell script that will do the work for you. Here is an example script that you should look through and customize as needed: avalon-iptables-config.sh

> If you're connected over ssh, it might kick you off.

Save your script to /etc/sysconfig/avalon-iptables-config.sh, make it executable and run it.

```
chmod +x /etc/sysconfig/avalon-iptables-config.sh
/etc/sysconfig/avalon-iptables-config.sh
```

If you run into connection issues you can disable the iptables, by running "service iptables stop". This will completely drop your firewall. When finished troubleshooting run "service iptables start".

Disable SELinux

```
echo 0 > /selinux/enforce
vim /etc/selinux/config #change the value of `SELINUX` from `enforcing` to `permissive`
```

> **You may have to disable SELinux completely if there's Passenger installation problem**
>
> `vim /etc/selinux/config #change the value of `SELINUX` to `disabled``

Install EPEL

This package has libyaml-devel which is required by ruby and not provided by Redhat.

```
rpm -ivh http://linux.mirrors.es.net/fedora-epel/6/i386/epel-release-6-8.noarch.rpm
```

Add the Avalon repository

Create the Avalon repository config file:

```
vim /etc/yum.repos.d/avalon-public.repo
```

Append the following code:

```
[avalon_public]
name=Avalon Public RHEL repository
baseurl=http://repo.avalonmediasystem.org/x86_64
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-avalon
cost=150
```

Install and place the Avalon GPG key in the proper location:

```
curl http://repo.avalonmediasystem.org/RPM-GPG-KEY-avalon -o /etc/pki/rpm-gpg/RPM-GPG-KEY-avalon
```

Install development libraries and packages for building Ruby

```
yum groupinstall "Development Tools"
yum install readline-devel zlib-devel libyaml-devel libffi-devel openssl-devel libxml2-devel libxslt-devel cmake
```

Install Java 8

```
yum install java-1.8.0-openjdk
```

Main Components

Fedora Commons Repository

Tomcat

Fedora runs as a webapp in Tomcat

Install Apache Tomcat

RHEL 7

```
yum install tomcat
vim /etc/tomcat/server.xml #line 71, change the Tomcat connector port from 8080 to 8984
```

Add Tomcat manager user

By default, no user has access to the Tomcat Manager App. Define a user in `/etc/tomcat/tomcat-users.xml` with access to the manager-gui role. Below is a very basic example.

```
<tomcat-users>
  <role rolename="manager-gui"/>
  <user username="admin" password="<insert strong password here>" roles="manager-gui"/>
</tomcat-users>
```

Configure Tomcat for Fedora

Append the following to `/etc/sysconfig/tomcat`

```
JAVA_OPTS="${JAVA_OPTS} -Dfcrepo.modeshape.configuration=classpath:/config/file-simple/repository.json -Dfcrepo.
home=/var/avalon/fedora/"
```

Restart Tomcat

```
service tomcat restart
```

Download and run the fcrepo installer

```
mkdir -p /var/avalon/fedora
chown tomcat:tomcat /var/avalon/fedora
wget https://github.com/fcrepo4/fcrepo4/releases/download/fcrepo-4.7.3/fcrepo-webapp-4.7.3.war -O /usr/share
/tomcat/webapps/fedora4.war
```

See if you can access Fedora's REST interface at http://<server host name>:8984/fedora4/rest

Try it out on your local machine and on another machine. If you can't reach the app from another machine, your iptables might need to be changed to allow access. If Fedora is not up, check the tomcat logs in */var/log/tomcat/*. Catalina.out and localhost.<date>.log usually provide the best information.

Solr

Avalon makes use of Solr through the Blacklight gem for faceting and relevance-based searching.

Install prerequisites

```
yum install lsof
```

Download the solr tarball and run the installation script

Download Solr from http://archive.apache.org/dist/lucene/solr/

```
wget http://archive.apache.org/dist/lucene/solr/6.4.2/solr-6.4.2.tgz
tar xzf solr-6.4.2.tgz solr-6.4.2/bin/install_solr_service.sh --strip-components=2
bash ./install_solr_service.sh solr-6.4.2.tgz
```

By default, the script extracts the distribution archive into `/opt`, configures Solr to write files into `/var/solr`, and runs Solr as the `solr` user. Follow the linked guide if you wish to change these defaults.

Create Avalon core for Solr

```
mkdir -p /tmp/avalon_solr/
wget https://raw.githubusercontent.com/avalonmediasystem/avalon/master/solr/config/solrconfig.xml -O /tmp
/avalon_solr/solrconfig.xml
wget https://raw.githubusercontent.com/avalonmediasystem/avalon/master/solr/config/schema.xml -O /tmp
/avalon_solr/schema.xml
su solr # Needs to run as solr user
/opt/solr/bin/solr create_core -c avalon -d /tmp/avalon_solr
exit
```

If you have successfully installed Solr you should be able to access the dashboard page at http://<server host name>:8983/solr

Instructions on how to manually start/stop Solr: https://cwiki.apache.org/confluence/display/solr/Running+Solr

MySQL

> **MariaDB**
>
> MariaDB is now the default database system for CentOS/RHEL7. Feel free to change Mysql below to MariaDB

Avalon uses MySQL for storing search queries, user data and roles, and as a back end for asynchronously sending requests to Matterhorn.

Install MySQL server

```
yum install mysql-server
service mysqld start
```

Create databases and users

Enter the mysql monitor

```
#mysql
Welcome to the MySQL monitor. Commands end with ; or \g.
...etc...
mysql>
```

Create a database for the Avalon web application and add a user to it

```
create database rails;
create user 'rails'@'localhost' identified by 'rails';
grant all privileges on rails.* to 'rails'@'localhost';
flush privileges;
```

Check your work and exit

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| rails              |
| test               |
+--------------------+
5 rows in set (0.00 sec)
mysql> select user, host from mysql.user;
+--------+--------------+
| user   | host         |
+--------+--------------+
| root   | 127.0.0.1    |
|        | 129.79.32.87 |
| root   | 129.79.32.87 |
|        | localhost    |
| rails  | localhost    |
| root   | localhost    |
+--------+--------------+
7 rows in set (0.00 sec)

mysql> exit;
Bye
```

See documentation for your version of MySQL Server for detailed syntax (http://dev.mysql.com/doc/refman/5.1/en/create-database.html)

Media Streaming Server

Nginx replaces Red5 as the default streaming server since Avalon 6.3. With the upgrade to MediaElementjs 4, we now rely completely on HLS.

An HLS-enabled server like Wowza, Adobe Media Server (commercial) or Nginx + the HLS module (open-source) can take an mp4 created from distribute-streaming and stream it on the fly.

Nginx instructions

Install Nginx with vod module

```
rpm -ihv http://installrepo.kaltura.org/releases/kaltura-release.noarch.rpm
yum install kaltura-nginx
```

Add `/etc/nginx/nginx.conf`

```
user nginx;
worker_processes 4;

events {
  worker_connections 1024;
}

http {
  server {
    listen 8980;

    vod_mode local;
    vod_last_modified 'Sun, 19 Nov 2000 08:52:00 GMT';
    vod_last_modified_types *;
    vod_metadata_cache metadata_cache 512m;
    vod_response_cache response_cache 128m;
    gzip on;
    gzip_types application/vnd.apple.mpegurl;
    open_file_cache          max=1000 inactive=5m;
    open_file_cache_valid    2m;
    open_file_cache_min_uses 1;
    open_file_cache_errors   on;

    location ~ ^/avalon/(?<stream>.+)/(?<resource>.+\.(?:m3u8|ts)) {
      alias /var/avalon/derivatives/$stream;
      vod hls;

      set $token "$arg_token";
      add_header X-Stream-Auth-Token "$token";

      sub_filter_types application/vnd.apple.mpegurl;
      sub_filter_once off;
      sub_filter '.ts' ".ts?token=$token";

      auth_request /auth;
      add_header Access-Control-Allow-Headers '*';
      add_header Access-Control-Expose-Headers 'Server,range,Content-Length,Content-Range';
      add_header Access-Control-Allow-Methods 'GET, HEAD, OPTIONS';
      add_header Access-Control-Allow-Origin '*';
      expires 100d;
    }

    location = /auth {
      # resolver 127.0.0.1;
      proxy_pass http://127.0.0.1/authorize.txt?token=$token&name=$stream;
      proxy_pass_request_body off;
      proxy_set_header Content-Length "";
      proxy_set_header X-Original-URI $request_uri;
    }
  }
}
```

`listen` should use a public open port.

`alias` should point to where the actual stream files are.

`proxy_pass` needs changing if installing Nginx on a different server.

Add `/etc/init.d/nginx`. For RHEL7 use this script.

```
#!/bin/sh
#
# nginx - this script starts and stops the nginx daemon
#
# chkconfig:   - 85 15
# description:  NGINX is an HTTP(S) server, HTTP(S) reverse \
#               proxy and IMAP/POP3 proxy server
```

```
# processname: nginx
# config:      /etc/nginx/nginx.conf
# config:      /etc/sysconfig/nginx
# pidfile:     /var/run/nginx.pid

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

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ "$NETWORKING" = "no" ] && exit 0

nginx="/usr/sbin/nginx"
prog=$(basename $nginx)

NGINX_CONF_FILE="/etc/nginx/nginx.conf"

[ -f /etc/sysconfig/nginx ] && . /etc/sysconfig/nginx

lockfile=/var/lock/subsys/nginx

make_dirs() {
   # make required directories
   user=`$nginx -V 2>&1 | grep "configure arguments:.*--user=" | sed 's/[^*]*--user=\([^ ]*\).*/\1/g' -`
   if [ -n "$user" ]; then
      if [ -z "`grep $user /etc/passwd`" ]; then
         useradd -M -s /bin/nologin $user
      fi
      options=`$nginx -V 2>&1 | grep 'configure arguments:'`
      for opt in $options; do
          if [ `echo $opt | grep '.*-temp-path'` ]; then
              value=`echo $opt | cut -d "=" -f 2`
              if [ ! -d "$value" ]; then
                  # echo "creating" $value
                  mkdir -p $value && chown -R $user $value
              fi
          fi
       done
    fi
}

start() {
    [ -x $nginx ] || exit 5
    [ -f $NGINX_CONF_FILE ] || exit 6
    make_dirs
    echo -n $"Starting $prog: "
    daemon $nginx -c $NGINX_CONF_FILE
    retval=$?
    echo
    [ $retval -eq 0 ] && touch $lockfile
    return $retval
}

stop() {
    echo -n $"Stopping $prog: "
    killproc $prog -QUIT
    retval=$?
    echo
    [ $retval -eq 0 ] && rm -f $lockfile
    return $retval
}

restart() {
    configtest || return $?
    stop
    sleep 1
    start
}
```

```
reload() {
    configtest || return $?
    echo -n $"Reloading $prog: "
    killproc $nginx -HUP
    RETVAL=$?
    echo
}

force_reload() {
    restart
}

configtest() {
  $nginx -t -c $NGINX_CONF_FILE
}

rh_status() {
    status $prog
}

rh_status_q() {
    rh_status >/dev/null 2>&1
}

case "$1" in
    start)
        rh_status_q && exit 0
        $1
        ;;
    stop)
        rh_status_q || exit 0
        $1
        ;;
    restart|configtest)
        $1
        ;;
    reload)
        rh_status_q || exit 7
        $1
        ;;
    force-reload)
        force_reload
        ;;
    status)
        rh_status
        ;;
    condrestart|try-restart)
        rh_status_q || exit 0
            ;;
    *)
        echo $"Usage: $0 {start|stop|status|restart|condrestart|try-restart|reload|force-reload|configtest}"
        exit 2
esac
```

Add `nginx` user and let it own nginx stuff

```
useradd -M -s /bin/nologin nginx
chown -R nginx:nginx /etc/nginx /var/log/nginx
```

Start nginx

```
chmod +x /etc/init.d/nginx
service nginx start
```

Avalon config should be updated to be compatible with Nginx:

```
streaming:
  server: :nginx
  http_base: 'http://localhost:8980/avalon'
  content_path: '/var/avalon/derivatives'
```

If you enable SSL on Avalon server, you should also enable SSL on the streaming server to avoid Mixed content warning.

FFmpeg

Installation prerequisites

Install prerequisite packages using yum and the Avalon repository (note: mediainfo is pinned to 0.7.61-1 because of a bug with time fragment formatting in most recent version 0.7.87-1 in epel):

```
yum install SDL-devel a52dec-devel bzip2-devel faad2-devel freetype-devel frei0r-plugins-devel \
        gsm-devel imlib2-devel lame-devel libdc1394-devel libraw1394-devel librtmp-devel libtheora-devel \
        libva-devel libfaac-devel libvdpau-devel libstdc++-devel libvorbis-devel libvpx-devel \
        mediainfo-0.7.61-1 opencore-amr-devel opencv-devel openjpeg-devel openssl-devel schroedinger-devel \
        speex-devel texi2html vo-aacenc-devel x264-devel xvidcore-devel yasm zlib-devel
```

Install rpmdev-setuptree

```
yum install rpmdevtools
```

Install ffmpeg srpm

The following commands need to run under a user other than root. Change to another user to continue.

```
su - someuser
```

Run the ffmpeg installer

```
rpmdev-setuptree
rpm -ivh https://github.com/avalonmediasystem/avalon-installer/blob/master/files/ffmpeg/ffmpeg-2.4.2-1.el6.src.
rpm?raw=true

# Retrieving https://github.com/avalonmediasystem/avalon-installer/blob/master/files/ffmpeg/ffmpeg-2.4.2-1.el6.
src.rpm
#   1:ffmpeg                 warning: user makerpm does not exist - using root
# warning: group makerpm does not exist - using root
########################################### [100%]
# warning: user makerpm does not exist - using root
# warning: group makerpm does not exist - using root
# Build ffmpeg binary as non-root and install as root

rpmbuild -bb rpmbuild/SPECS/ffmpeg24.spec
```

Log back in as root and finish the install.

```
su - root
rpm -ivh /home/someuser/rpmbuild/RPMS/x86_64/ffmpeg-*.rpm
```

You can also build a more modern ffmpeg from source, but not all versions work. Version 3.1 is known to work - check out the release/3.1 branch and build from there. If your modern ffmpeg uses fdk_aac instead of libfaac, adjust the Matterhorn etc/encoding/avalon.properties accordingly.

HTTPD

Install and start the httpd service.

```
yum install httpd
service httpd start
```

With newer httpd you may need to in */etc/httpd/conf.d/10-mod_rewrite.conf,* replace *RewriteLock* line with *Mutex sem*

Matterhorn

Install Matterhorn

Create a user for Matterhorn and then install Matterhorn

```
useradd matterhorn

wget https://github.com/avalonmediasystem/avalon-felix/archive/1.4.x.tar.gz
tar xvf 1.4.x.tar.gz
mv avalon-felix-1.4.x /usr/local/matterhorn

wget --no-check-certificate https://raw.github.com/avalonmediasystem/config-files/master/matterhorn
/matterhorn_init.sh
mv matterhorn_init.sh /etc/init.d/matterhorn

chmod +x /etc/init.d/matterhorn
chown -R matterhorn:matterhorn /usr/local/matterhorn
```

Add avalon user and create avalon directory.

```
useradd avalon
mkdir /var/www/avalon
chown -R avalon:avalon /var/www/avalon
```

Create and configure the media_path (upload) and streaming directories.

```
mkdir -p /usr/local/masterfiles
chown avalon:avalon /usr/local/masterfiles

mkdir -p /var/avalon/derivatives
chown matterhorn:matterhorn /var/avalon/derivatives

chmod 0775 /var/avalon/derivatives
```

Configure Matterhorn

Download Matterhorn config and verify property values.

```
wget --no-check-certificate https://raw.github.com/avalonmediasystem/config-files/master/matterhorn/config.
properties
vim config.properties
```

And verify the configuration of the streaming directories

```
org.opencastproject.streaming.directory=/var/avalon/derivatives
```

Also check in `/usr/local/matterhorn/etc/load/org.opencastproject.organization-mh_default_org.cfg`

```
prop.avalon.stream_base=file:///var/avalon/derivatives
```

Move the config to the appropriate spot

```
mv config.properties /usr/local/matterhorn/etc/
```

Add matterhorn user to the avalon group.

```
usermod -G avalon matterhorn
```

Optional, but recommended to avoid problems with batch ingest: Change number of processes available to matterhorn user, cleaning up work dir and using external DB.

Apache Passenger and Ruby

Change current user to avalon then install RVM and ruby 2.4.1

```
su - root
yum install sqlite-devel
su - avalon
curl -L https://get.rvm.io | bash -s stable --ruby=2.4.1
```

Source the RVM shell (as avalon user) or close the terminal and open it back up.

```
source /home/avalon/.rvm/scripts/rvm
rvm use 2.4.1
```

Install Passenger via Gem (as avalon user)

```
gem install passenger
```

Check to make sure passenger installed in the expected location (as avalon user)

```
$ passenger-config --root
/home/avalon/.rvm/gems/ruby-2.4.1/gems/passenger-5.0.17
```

Install Passenger apache module requirements (as root)

```
su - root
yum install curl-devel httpd httpd-devel apr-devel apr-util-devel
```

Build passenger for your version of Apache and Ruby (as avalon user)

```
su - avalon
passenger-install-apache2-module #copy the suggested Apache configuration file settings for later
```

Create an apache configuration file (as root)

```
su - root
vim /etc/httpd/conf.d/passenger.conf
```

Example contents of /etc/httpd/conf.d/passenger.conf, which may need to be changed based off of the current version of ruby and passenger:

```
LoadModule passenger_module /home/avalon/.rvm/gems/ruby-2.4.1/gems/passenger-5.0.17/buildout/apache2
/mod_passenger.so
<IfModule passenger_module>
  PassengerRoot /home/avalon/.rvm/gems/ruby-2.4.1/gems/passenger-5.0.17
  PassengerDefaultRuby /home/avalon/.rvm/wrappers/ruby-2.4.1/ruby
  PassengerMaxPoolSize 30
  PassengerPoolIdleTime 300
  PassengerMaxInstancesPerApp 0
  PassengerMinInstances 3
  PassengerSpawnMethod smart-lv2
</IfModule>
```

Apache security configuration

```
wget --no-check-certificate https://raw.github.com/avalonmediasystem/config-files/master/sbin/avalon_auth -O
/usr/local/sbin/avalon_auth
chmod +x /usr/local/sbin/avalon_auth
wget --no-check-certificate https://raw.github.com/avalonmediasystem/config-files/master/apache/10-mod_rewrite.
conf -P /etc/httpd/conf.d/
```

Create a virtual host for avalon in */etc/httpd/conf.d/avalon.conf*

```
wget --no-check-certificate https://raw.github.com/avalonmediasystem/config-files/master/apache/20-avalon.conf -
P /etc/httpd/conf.d/
vim /etc/httpd/conf.d/20-avalon.conf
```

In 20-avalon.conf add this line inside the VirtualHost tag:

```
RailsEnv development
```

If using SSL, the following fix should be added to address BEAST, POODLE, RC4 issues (after the `SSLEngine on`)

```
SSLProtocol all -SSLv2 -SSLv3
SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:!RC4:+HIGH:+MEDIUM:-LOW
```

Restart apache. With apache running, check passenger-status

```
service httpd restart
su - avalon
which passenger-status
#> ~/.rvm/gems/ruby-2.4.1/bin/passenger-status
```

Avalon

Grab Avalon code from github

```
cd ~
git clone git://github.com/avalonmediasystem/avalon.git
cd avalon
git checkout master #make sure you are in the master branch (should be by default)
su - root
chown avalon:avalon /var/www/avalon/public/
su - avalon
mv public/* /var/www/avalon/public/
rmdir public
mv  * /var/www/avalon/
```

Configure Avalon

As of 6.3, Avalon is using the flexible and increasingly popular Config gem. Default settings for Avalon now live at `config/settings.yml`, which should not be altered. Any custom config should be placed in `config/settings/<environment>.local.yml` which will selectively override the default values.

Properly formatted environment variables can also override Avalon settings. For example, `SETTINGS__REDIS__HOST` will override

```
redis:
  host:
```

A list of config values can be found at Configuration Files#config/settings.yml

If using vim with default settings and pasting the the code below, it will automatically comment out the last line. To prevent that, enable paste using the command :set paste and then use just ctrl+shift+v instead of going into insert mode.

Create */var/www/avalon/config/setup_load_paths.rb* and add:

```
if ENV['MY_RUBY_HOME'] && ENV['MY_RUBY_HOME'].include?('rvm')
   begin
     gems_path = ENV['MY_RUBY_HOME'].split(/@/)[0].sub(/rubies/,'gems')
     ENV['GEM_PATH'] = "#{gems_path}:#{gems_path}@global"
     require 'rvm'
     RVM.use_from_path! File.dirname(File.dirname(__FILE__))
   rescue LoadError
     raise "RVM gem is currently unavailable."
   end
 end
 # If you're not using Bundler at all, remove lines bellow
 ENV['BUNDLE_GEMFILE'] = File.expand_path('../Gemfile', File.dirname(__FILE__))
 require 'bundler/setup'
```

Configure database settings

```
cd /var/www/avalon/config
vim database.yml
```

Replace database.yml with the correct values for your production environment

```
development:
  adapter: mysql2
  host: localhost
  database: rails
  username: rails
  password: rails
  pool: 5
  timeout: 5000
```

Install the mysql2 adapter

```
yum install cmake #<--will be required for rugged gem
yum install mariadb-devel
su - avalon
gem install activerecord-mysql2-adapter
gem install mysql2
vim /var/www/avalon/Gemfile
```

Run the bundle install

```
# as root
yum install nodejs # Javascript runtime

# as avalon
cd /var/www/avalon
gem update debugger-ruby_core_source
gem install bundler
bundle install --with mysql
```

Finish configuring Avalon

Edit */var/www/avalon/config/solr.yml* and */var/www/avalon/config/blacklight.yml*

```
  development:
    url: http://localhost:8983/solr/avalon
```

Edit */var/www/avalon/config/fedora.yml*

```
  development:
    user: fedoraAdmin
    password: fedoraPassword
    url: http://127.0.0.1:8984/fedora4/rest
    base_path: ""
```

Create */var/www/avalon/config/matterhorn.yml*

```
development:
  url: http://matterhorn_system_account:CHANGE_ME@localhost:8080/
```

Avalon config file

Avalon settings now live in */var/www/avalon/config/settings.yml.* The default values should be sufficient to start with.

They can be selectively overwritten by creating a *settings/<environment>.yml*, or by using environment variables. Consult the config gem doc to understand how it works, or Avalon's documentation to customize this file for your installation.

Let Avalon know where your HLS streams are

config/settings/development.local.yml

```
streaming:
  content_path: '/var/avalon/derivatives'
```

Change the secrets.yml file:

```
cd /var/www/avalon
rake secret
```

grab the output of rake secret and add it to secrets.yml where instructed.

More information: Configuration Files#config/secrets.yml

Create controlled_vocabulary.yml

```
cp controlled_vocabulary.yml.example controlled_vocabulary.yml
```

Create the database using rake

```
# as avalon user
cd /var/www/avalon
rake db:create
```

If you get an error message saying that you can't connect to the database, take a look at this post and follow some of the troubleshooting steps.

http://stackoverflow.com/questions/5376427/cant-connect-to-local-mysql-server-through-socket-var-mysql-mysql-sock-38

Run the database migrations

```
rake db:migrate
```

Set rails environment to development, if it has not defaulted to this. On the first line of */var/www/avalon/config/environment.rb* make sure it says 'development'

```
ENV['RAILS_ENV'] ||= 'development'
```

Redis & Resque

Avalon uses Resque for background processing, which relies Redis as its key-value store.

Install Redis

```
yum install redis
```

Start Resque

```
# as avalon (replace production with development if necessary)
cd /var/www/avalon/
RAILS_ENV=production BACKGROUND=yes bundle exec rake resque:scheduler
RAILS_ENV=production BACKGROUND=yes QUEUE=* COUNT=2 bundle exec rake resque:workers
```

Resque logs to log/resque.log in the avalon directory.

To restart rescue, simple kill its two processes (`ps aux | grep resque`) and run the above commands again.

Additional Configurations

Dropbox

```
groupadd -r dropbox
useradd -r avalondrop
usermod -G dropbox avalon
mkdir -p /srv/avalon/dropbox
chown avalondrop:dropbox /srv/avalon/dropbox
chmod 2775 /srv/avalon/dropbox
```

Edit */etc/ssh/sshd_config*

```
# override default of no subsystems
Subsystem sftp internal-sftp

# Example of overriding settings on a per-user basis
#Match User anoncvs
# X11Forwarding no
# AllowTcpForwarding no
# ForceCommand cvs server
Match Group dropbox
ChrootDirectory /srv/avalon
X11Forwarding no
AllowTcpForwarding no
ForceCommand internal-sftp
```

Restart SSH

```
service sshd restart
```

Batch ingest

To manually start a batch ingest job, run as avalon user

```
rake avalon:batch:ingest
```

To make batch ingest run automatically whenever a manifest is present, you need to add a cron job. This cron job can be created by the whenever gem from reading `config/schedule.rb`. To preview, run

```
whenever
```

this will translate content in `schedule.rb` to cron job syntax. Once verified, run the following to write job to crontab

```
whenever --update-crontab
```

You should get the cron job automatically if you were deploying from Capistrano.

Using the System

You should be able to visit the webpage with just the hostname (ie http://localhost)

Create an admin account

Click on "Sign in" in the upper right corner of the website main page. Set up a default identity with administrative privileges using the following properties.

```
archivist1@example.com
<some password>
```

You can also create an account from the command line in the root of your avalon install:

```
bundle exec rake avalon:user:create avalon_username=user@example.com avalon_password=password
avalon_groups=administrator
```

Additional information

You can find specific information about using the system in the Collection Manager's Guide. Sample content is available for your convenience. Upload new items individually or by batch directly via SFTP using the avalondrop account you created above.

Configure additional feataures

Known Issues - a list of bugs, workarounds, and cautions.

Restarting the Server

Before you restart your Avalon server, you'll want to make sure all of the services necessary to run Avalon will start automatically after the restart. Run these commands once and you should be set:

```
chkconfig --level 345 tomcat on
chkconfig --level 345 mysqld on
chkconfig --level 345 sshd on
chkconfig --level 345 red5 on
chkconfig --level 345 httpd on
chkconfig --level 345 matterhorn on
```