Manual Installation Instructions

This documentation is for Release 7.0 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
- Release 6.3-6.5 version of this page: v.201

These instructions provide a recipe for building your own all-in-one Avalon system from scratch on CentOS or Red Hat Enterprise Linux version 7.x. 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

The instructions below require being run as root unless specifically noted otherwise.

Storage requirement

Avalon and components need about 20GB of disk space to install.

Open ports requirement

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
8980	HTTP (Nginx)	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

```
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`
```

Reboot to apply change

```
shutdown -r now
```

Install EPEL

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

```
yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

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

MariaDB

MariaDB

MariaDB is now the default database system for CentOS/RHEL7 and can be used interchangeably with MySql. MySql or PostgreSQL can be substituted if desired.

Avalon uses MariaDB for storing search queries, user data and roles, and as a back end our encoding dashboard.

Install MariaDB server

```
yum install mariadb-server
systemctl start mariadb
```

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>
```

Create Fedora user and database

Enter the mariadb client

```
mysql
mariadb> create database fcrepo CHARACTER SET utf8 COLLATE utf8_general_ci;
mariadb> create user 'fcrepo'@'localhost' identified by '<fcrepo_password>';
mariadb> grant all privileges on fcrepo.* to 'fcrepo'@'localhost';

mariadb> create database rails CHARACTER SET utf8 COLLATE utf8_general_ci;
mariadb> create user 'rails'@'localhost' identified by '<rails_pasword>';
mariadb> grant all privileges on rails.* to 'rails'@'localhost';

mariadb> flush privileges;
```

Check your work and exit

```
mariadb> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| fcrepo             |
| performance_schema |
| test               |
+--------------------+
5 rows in set (0.00 sec)
mariadb> exit;
Bye
```

Configure Tomcat for Fedora

Append the following to `/etc/tomcat/tomcat.conf`

```
JAVA_OPTS="-Dfcrepo.modeshape.configuration=classpath:/config/jdbc-mysql/repository.json -Dfcrepo.mysql.
username=fcrepo -Dfcrepo.mysql.password=<fcrepo_password> -Dfcrepo.mysql.host=localhost -Dfcrepo.mysql.
port=3306 -Dfcrepo.home=/var/avalon/fedora/"
```

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-4.7.5/fcrepo-webapp-4.7.5.war -O /usr/share
/tomcat/webapps/fedora4.war
```

Restart Tomcat

```
systemctl restart tomcat
```

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/6.6.6/solr-6.6.6.tgz
tar xzf solr-6.6.6.tgz solr-6.6.6/bin/install_solr_service.sh --strip-components=2
bash ./install_solr_service.sh solr-6.6.6.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

Media Streaming Server

An HLS-enabled server like Wowza, Adobe Media Server (commercial) or Nginx + the HLS module (open-source) can take an mp4 created by Avalon 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 `/lib/systemd/system/nginx.service`

```
[Unit]
Description=The NGINX HTTP and reverse proxy server
After=syslog.target network.target remote-fs.target nss-lookup.target

[Service]
Type=forking
PIDFile=/run/nginx.pid
ExecStartPre=/usr/sbin/nginx -t
ExecStart=/usr/sbin/nginx
ExecReload=/usr/sbin/nginx -s reload
ExecStop=/bin/kill -s QUIT $MAINPID
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

Add `nginx` user and let it own nginx stuff

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

Make 8980 bindable and start nginx

```
semanage port -a -t http_port_t  -p tcp 8980
systemctl start nginx
```

Later: 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 & Mediainfo

Download prebuilt ffmpeg

The following prebuilt binaries are provided by a third party. Proceed with caution.

Download and install ffmpeg (for transcoding & thumbnails)

```
mkdir -p /tmp/ffmpeg && cd /tmp/ffmpeg
curl https://johnvansickle.com/ffmpeg/releases/ffmpeg-release-amd64-static.tar.xz | tar xJ
cp `find . -type f -executable` /usr/bin/
```

Install Mediainfo (for technical metadata)

```
yum install mediainfo
```

HTTPD

Install and start the httpd service.

```
yum install httpd
systemctl start httpd
```

Apache Passenger and Ruby

Change current user to avalon then install RVM and ruby 2.4.1

```
yum install ruby sqlite-devel # Needed to build Ruby using RVM.
useradd avalon
su - avalon
curl -L https://get.rvm.io | bash -s stable --ruby=2.5.5
```

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

```
source /home/avalon/.rvm/scripts/rvm
rvm use 2.5.5
exit
```

Install Passenger apache module requirements (as root)

```
yum install pygpgme curl
curl --fail -sSLo /etc/yum.repos.d/passenger.repo https://oss-binaries.phusionpassenger.com/yum/definitions/el-passenger.repo

yum install mod_passenger || yum-config-manager --enable cr && yum install mod_passenger
```

Create a virtual host for avalon

```
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 production
```

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
```

Modify /etc/httpd/conf.d/passenger.conf

```
PassengerRuby /home/avalon/.rvm/rubies/ruby-2.5.5/bin/ruby
```

Validate passenger install and restart apache

```
passenger-config validate-install
systemctl start httpd
```

Avalon

Grab Avalon code from github

```
git clone https://github.com/avalonmediasystem/avalon.git /var/www/avalon
chown -R avalon:avalon /var/www/avalon
```

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

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

Configure database settings

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

Replace database.yml with the correct values for your production environment. Note that the pool setting should be equal or exceed the number of concurrent jobs in Sidekiq.

```
production:
  adapter: mysql2
  host: localhost
  database: rails
  username: rails
  password: rails
  pool: 20
  timeout: 5000
```

Install the mysql2 adapter

```
yum install cmake #<--will be required for rugged gem
yum install mariadb-devel
```

Install gems

Run the bundle install

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

# as avalon
su - avalon
cd /var/www/avalon
gem install bundler
bundle install --with mysql production --without development test
exit
```

Finish configuring Avalon

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

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

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

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

Create streaming directory

```
# as root
mkdir -p /var/avalon/derivatives
chown avalon:avalon /var/avalon/derivatives
```

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/production.local.yml

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

Change the secrets.yml file:

```
export RAILS_ENV=production
rake secret
```

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

More information: Configuration Files#config/secrets.yml

Create controlled_vocabulary.yml

```
cp config/controlled_vocabulary.yml.example config/controlled_vocabulary.yml
```

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
```

Install yarn and node modules

```
# as root
curl --silent --location https://dl.yarnpkg.com/rpm/yarn.repo | sudo tee /etc/yum.repos.d/yarn.repo
yum install yarn

# as avalon
su - avalon
cd /var/www/avalon
yarn install
```

Precompile assets

```
# as avalon
RAILS_ENV=production bundle exec rake assets:precompile
```

Restart Apache

```
# as root
systemctl restart httpd
```

Install ImageMagick

```
# as root
yum install imagemagick
```

Sidekiq

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

Install Redis

```
# as root
yum install redis
systemctl start redis
```

Install Sidekiq

```
# as root
wget https://raw.githubusercontent.com/mperham/sidekiq/master/examples/systemd/sidekiq.service -O /lib/systemd
/system/sidekiq.service
```

Edit the following lines in sidekiq.service

```
WorkingDirectory=/var/www/avalon
ExecStart=/bin/bash -lc '/home/avalon/.rvm/gems/ruby-2.5.5/bin/bundle exec sidekiq -e production'
User=avalon
Group=avalon
```

```
# as root
systemctl start sidekiq
```

Sidekiq logs to STDOUT.

tmp Error after uploading file

When ingesting a media file, you may encounter an error message saying that file:///tmp/filename can't be accessed or located. This may result from the protected temp file settings that are defaults in CentOS 7. Fix by changing "true" to "false" for PrivateTmp in these files in /usr/lib /systemd/system:

sidekiq.service PrivateTmp=false
nginx.service PrivateTmp=false
httpd.service PrivateTmp=false

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.

Authentication Strategy

Avalon comes with Persona by default but it can be configured to work with other authentication strategies by using the appropriate omniauth gems. The following example is applicable to Indiana University CAS, it may need some adjustments in order to work with other CAS implementation.

Add to `Gemfile`

```
gem 'net-ldap'
gem 'omniauth-cas', :git => "https://github.com/cjcolvar/omniauth-cas.git"
```

Install new gems

```
bundle install
```

Add to `config/initializers/my-ldap.rb`

```
module Avalon
      MY_GUEST_LDAP = Net::LDAP.new
      MY_GUEST_LDAP.host = "eads.myuni.edu"
      MY_GUEST_LDAP.authenticate 'cn=******,ou=Accounts,dc=eads,dc=myuni,dc=edu', '******'

      GROUP_LDAP = Net::LDAP.new
      GROUP_LDAP.host = "ads.myuni.edu"
      GROUP_LDAP.authenticate 'cn=******,ou=Accounts,dc=ads,dc=myuni,dc=edu', '******'
      GROUP_LDAP_TREE = "dc=ads,dc=myuni,dc=edu"
end
```

Add `config/initializers/user_auth_cas.rb`

```
require 'net/ldap'

User.instance_eval do
  def self.find_for_cas(access_token, signed_in_resource=nil)
    logger.debug "#{access_token.inspect}"
    #data = access_token.info
    username = access_token.uid
    email = nil

    user = User.where(:username => username).first

    unless user
      if email.nil?
        tree = "dc=ads,dc=myuni,dc=edu"
        filter = Net::LDAP::Filter.eq("cn", "#{username}")
        email = Avalon::GROUP_LDAP.search(:base => tree, :filter => filter, :attributes=> ["mail"]).first.mail.
first
      end
      user = User.find_or_create_by_username_or_email(username, email)
      raise "Finding user (#{ user }) failed: #{ user.errors.full_messages }" unless user.persisted?
    end
    user
  end
end
```

Add to `config/settings/production.local.yml`

```
auth:
  configuration:
    - :name: My University
      :logo: my_logo.png
      :provider: :cas
      :params:
        :host: cas.myuni.edu
        :login_url: /cas/login
        :service_validate_url: /cas/validate
        :logout_url: /cas/logout
        :ssl: true
```

Using the System

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

Create an admin account

You can 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 mariadb on
chkconfig --level 345 nginx on
chkconfig --level 345 sshd on
chkconfig --level 345 redis on
chkconfig --level 345 sidekiq on
chkconfig --level 345 httpd on
```