
almir Documentation

Release 0.1.8.0

Domen Kožar

January 22, 2014

Author Domen Kožar <domen@dev.si>

Source code [github.com](#) project

Bug tracker [github.com](#) issues

Generated January 22, 2014

License GPL v3

Version 0.1.8.0

Features

- supports bacula-director version 5.x.x
- supports sqlite, postgresql, mysql databases
- complete read-only interface for bacula-director
- interactive web console frontend to bconsole
- export data to excel, pdf and clipboard
- supports python2.6 and python 2.7
- 100% test coverage

Overview

Almir is a [bacula](#) web interface for administrators written in Python. It is designed with simplicity in mind, although backup management is never simple.

It is named after [Almir Karič](#), a fellow developer from **Slovenia**, whose dream was to live in San Francisco. Two years after he moved, he died in a car crash. This is his gift for eternity.

Almir is open source software licensed under **GPL v3**, started by [Domen Kožar](#) in 2011.

1.1 Demo

<http://almir-demo.domenkozar.com/>

1.2 Design goals

- each release is pinned to bacula-director specific major version
- simple is better than complicated
- inline documentation
- convention over configuration
- can act as view interface only (optional configuration functionality)
- plugin into existing bacula instance
- total control of bacula through existing bacula api

1.3 Installation

Almir will install everything inside one directory, which must be empty. Application is meant to be self-contained, meaning no additional administrator is needed besides upgrading to a newer version. Almir should be always installed on a system together with bacula-director.

1.3.1 Prerequisites

- Bacula 5.x.x is installed and bacula-director is running
- installed python2.6 or python2.7 (compiled with sqlite support)
- using postgresql: make sure *postgresql.conf* includes a line *client_encoding = utf8*

1.3.2 Installer (interactive)

Install prerequisites (Debian based):

```
$ sudo apt-get install git bacula-console python-distribute gcc python-dev wget
```

Note: Installer will ask you few questions about SQL database and configuration for bconsole.

Install almir (recommended: under same user as bacula):

```
$ cd /some/empty/directory/to/install/almir/  
$ sh -xec "$(wget -O - https://raw.githubusercontent.com/iElectric/almir/master/install_production.sh) "
```

You can continue with configuring *Configuring Nginx as a frontend*.

1.3.3 Manual (not recommended)

For security and *unix freaks, here is a step by step description what interactive install does behind the scene. Taking manual steps to install will ensure you are missing lovely time with your beloved one this weekend and replacing that with mild headache (specially if you are not familiar with python deployment quirks).

Interactive install also handles upgrades transparently. Almir is developed with agile workflow with small incremental versions of new changes. You will have to dig in yourself how to upgrade environment upon a new release. Still stubborn? Let's go!

- Download latests release from github (as tarball or git clone) from *latests* branch (install_production.sh takes care of that otherwise)
- unpack into empty directory
- Make sure you use python2.7 or python2.6, since other versions are not supported
- Populate *production.ini* file for daemon configuration, taken from *buildout.d/templates/production.ini.in* (*almir/scripts/configure_deploy.py* takes care of that interactively, then runs buildout to output configuration file from the template)
- Install all python dependencies from *setup.py* file, preferably within virtualenv (buildout takes care of that and pins them down to known workable set of versions, found in *buildout.d/versions.cfg*)
- Once you have installed dependencies (with *python setup.py install*) inside virtualenv or system python (REALLY not recommended), you should have *pserve* binary installed in *bin/* directory
- run it like so: *bin/pserve production.ini*
- make sure daemon runs at reboot, configure log rotation

Happy? Let's see until first upgrade.

1.3.4 Configuring Nginx as a frontend

It is wise to use frontend HTTP server and proxy HTTP requests to python web server. Following is an example for nginx, you could also use papache2 or lighthttpd.

You would normally put this in */etc/nginx/sites-enabled/almir.mywebsite.com.conf*

```
server {  
    listen          80;  
    server_name    almir.mywebsite.com;
```

```

location / {
    proxy_pass http://localhost:2500;

    proxy_set_header    X-Real-IP    $remote_addr;
    proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header    Host          $http_host;
    proxy_redirect       off;

    # optional authentication - recommended
    auth_basic "Restricted";
    # how to correctly write htpasswd: http://wiki.nginx.org/HttpAuthBasicModule#auth_basic_
    auth_basic_user_file /some/directory/to/install/almir/.htpasswd;
}
}

```

Then run:

```
$ /etc/init.d/nginx reload
```

Now try to access <http://almir.mywebsite.com/> (if you have an error, follow instructions at *Reporting bugs*)

1.3.5 Configuring Apache2 as a frontend

```
<VirtualHost *:80>
```

```
ServerName almir.mydomain.com
DocumentRoot "/var/www/almir.mydomain.com"
```

```
ProxyPreserveHost On
```

```
<Location />
ProxyPass http://almir.mydomain.com:2500/
ProxyPassReverse http://almir.mydomain.com:2500/
</Location>
```

```
ErrorLog /var/log/httpd/almir.mydomain.com-error.log
CustomLog /var/log/httpd/almir.mydomain.com-access.log combined
```

```
</VirtualHost>
```

Do not forget to restrict access to almir, either by IP or by username/password.

1.4 Upgrading to a newer release

Run:

```
$ cd almir_install_directory
$ git pull
$ python bootstrap.py
$ bin/buildout
$ bin/supervisorctl shutdown
$ bin/supervisord
```

You can also use that in crontab to auto upgrade on new releases, if you are crazy enough. You would probably extra check if upgrade is needed, something like running following and checking for any output:

```
$ git log latests..origin/latests
```

1.5 Reporting bugs

Check if an issue already exists at <https://github.com/iElectric/almir/issues>, otherwise add new one with following information:

- bacula-director version, operating system and browser version
- include screenshot if it provides any useful information
- pastebin (<http://paste2.org>) output of `$ cat ALMIR_ROOT/var/logs/almir-stderr*`
- pastebin `ALMIR_ROOT/buildout.cfg`, but be careful to *remove any sensitive data*

1.6 Filesystem structure

TODO ;)

2.1 Setup developer environment

- `sudo aptitude install git gcc python-dev bacula-console`
- `git clone https://github.com/iElectric/almir.git almir`
- `cd almir`
- `cp buildout.d/templates/buildout.cfg.in buildout.cfg`
- `vim buildout.cfg # configure variables`
- `python bootstrap.py`
- `bin/buildout`
- `bin/pserve --reload development.ini`

2.2 Running Python tests

Easy as:

```
$ bin/nosetests
```

By default it will run against sqlite fixture, you can also tell nosetests to use mysql fixture (you need to import sql manually for now):

```
$ DATABASE="mysql" bin/nosetests
```

Or just specify sqlalchemy engine:

```
$ ENGINE="sqlite:///var/lib/bacula/bacula.db" bin/nosetests
```

2.3 Running Javascript tests

Install and configure phantomjs (webkit headless testing):

```
$ sudo apt-get install libqtwebkit-dev
$ git clone git://github.com/ariya/phantomjs.git && cd phantomjs
$ qmake-qt4 && make
$ sudo cp bin/phantomjs /usr/local/bin/
```

Run tests:

```
$ cd ../almir
$ ./travis_qunit_tests.sh
```

2.4 Coding conventions

- PEP8 except for 80 char length rule
- add changelog, test and documentation with code in commits
- same applies to javascript
- jslinted javascript

2.5 Releasing almir

```
$ bin/fullrelease
$ git checkout latests
$ git merge master
$ git push
# update http://readthedocs.org/dashboard/almir/versions/
```

Changelog

3.1 0.1.8 (unreleased)

- Nothing changed yet.

3.2 0.1.7 (2013-03-27)

- [bug] Add also LICENSE to MANIFEST.in [Domen Kožar]

3.3 0.1.6 (2013-03-27)

- [bug] Add .ini to MANIFEST.in [Domen Kožar]

3.4 0.1.5 (2013-03-27)

- [feature] Refactor the package a bit, so it's easier to package it for Linux distributions [Domen Kožar]
- [bug] Update MANIFEST.in so all files are included in release [Domen Kožar]
- [bug] Add new bootstrap.py and pin down zc.buildout version to avoid upgrading zc.buildout to 2.0 [Domen Kožar]
- [feature] Add apache2 configuration example [Iban]

3.5 0.1.4 (2013/03/23)

- brownbag release

3.6 0.1.3 (2012/08/27)

- [bug] upgraded doctutils as it was failing buildout [Domen Kožar]
- removed some dependencies on production, upgraded zc.buildout to 1.6.3 for faster installation [Domen Kožar]

- determine version from distribution metadata [Domen Kožar]

3.7 0.1.2 (2012/05/31)

- [bug] interactive installer would swallow error message when SQL connection string was not formatted correctly
- [bug]: #7: don't word wrap size columns
- [feature] add manual install steps
- [bug] #4: client detail page did not render if client had no successful backup
- [bug] #5: correctly parse scheduled jobs (choked on Admin job)
- [feature] use python2.7 or python2.6, otherwise abort installation

3.8 0.1.1 (2012/04/18)

- [bug] fix support for postgresql 9.1 [Domen Kožar]
- [feature] add reboot crontab for almir daemon [Domen Kožar]
- [bug] MySQL database size calculation was wrong, sometimes crashing the dashboard [Domen Kožar]
- [bug] console command list was not ordered and search box was not shown [Domen Kožar]
- [bug] bconsole did not accept non-ascii input [Domen Kožar]

3.9 0.1 (2012/04/06)

- Initial version [Domen Kožar]

Source documentation

4.1 almir – Main package

4.1.1 `almir.forms` – HTML forms definitions

4.1.2 `almir.meta` – Configuration for models

4.1.3 `almir.models` – SQLAlchemy models

4.1.4 `almir.views` – Pyramid views

4.1.5 `almir.lib` – Non MVC code

`almir.lib.bacula_base64` – Bacula custom base64 implementation

`almir.lib.bconsole` – Python interface to bconsole

`almir.lib.console_commands` – Parsed list of bconsole commands

`almir.lib.sqlalchemy_custom_types`

`almir.lib.sqlalchemy_declarative_reflection`

`almir.lib.sqlalchemy_lowercase_inspector`

`almir.lib.utils` - General utilities

4.1.6 `almir.scripts` – Runnable scripts package

`almir.scripts.configure_deploy` – Ask few questions and configure almir

`almir.scripts.parse_console_commands` – Parse help commands from bconsole source

4.1.7 `almir.tests` – Tests package

`almir.tests.test_functional` – Functional tests

Indices and tables

- *genindex*
- *modindex*
- *search*