# bodhi Documentation

***Release 2.4.0***

**Luke Macken**

**Mar 06, 2017**

# Contents

Contents:

CHAPTER 1

# bodhi CLI man page

## Synopsis

`bodhi` COMMAND SUBCOMMAND [OPTIONS] [ARGS]...

## Description

`bodhi` is the command line interface to bodhi, Fedora's update release management system. It can be used to create or modify updates and overrides.

## Options

Most of the commands will accept these three options:

`--help`

Show help text and exit.

`--password <text>`

A password to authenticate as the user given by `--user`.

`--staging`

Use the staging bodhi instance instead of the production instance.

`--user <username>`

Many commands accept this flag to specify which user's updates should be operated upon.

`--version`

Show version and exit. Not accepted by subcommands.

## Commands

There are two commands, `overrides` and `updates`. They are described in more detail in their own sections below.

`bodhi overrides <subcommand> [options] [args]`

Provides commands to aid in management of build overrides. Supports subcommands `query` and `save`, described below.

`bodhi updates <subcommand> [options] [args]`

Provides an interface to manage updates. Supports subcommands `comment`, `download`, `new`, `query`, and `request`, described below.

## Overrides

The `overrides` command allows users to manage build overrides.

`bodhi overrides query [options]`

The `query` subcommand provides an interface for users to query the bodhi server for existing overrides.

`bodhi overrides save [options] <nvr>`

Save the build root given by `<nvr>` as a buildroot override. The `save` subcommand supports the following options:

`--duration <days>`

The number of days the override should exist, given as an integer.

`--notes <text>`

Notes on why this override is in place.

## Updates

The `updates` command allows users to interact with bodhi updates.

`bodhi updates comment [options] <update> <text>`

Leave the given text as a comment on a bodhi update. The `comment` subcommand supports the following options:

`--karma [+1 | 0 | -1]`

The karma value you wish to contribute to the update.

`bodhi updates download [options]`

Download update(s) given by CVE(s), ID(s), or NVR(s). One of `--cves`, `--updateid`, or `builds` is required. The download subcommand supports the following options:

`--cves <cves>`

A comma-separated list of CVEs that identify updates you would like to download.

`--updateid <ids>`

A comman-separated list of update IDs you would like to download.

`--builds <nvrs`

A comma-separated list of NVRs that identify updates you would like to download.

`bodhi updates new [options] <builds>`

Create a new bodhi update containing the builds, given as a comma separated list of NVRs. The `new` subcommand supports the following options:

`--type [security | bugfix | enhancement | newpackage]`

The type of the new update.

`--notes <text>`

The description of the update.

`--notes-file <path>`

A path to a file containing a description of the update.

`--bugs <bugs>`

A comma separated list of bugs to associate with this update.

`--close-bugs`

If given, this flag will cause bodhi to close the referenced bugs automatically when the update reaches stable.

`--request [testing | stable | upush]`

The repository requested for this update.

`--autokarma`

Enable autokarma for this update.

`--stable-karma <integer>`

Configure the stable karma threshold for the given value.

`--unstable-karma <integer>`

Configure the unstable karma threshold for the given value.

`--suggest [logout | reboot]`

Suggest that the user logout or reboot upon applying the update.

`--file <path>`

A path to a file containing all the update details.

`bodhi updates query [options]`

Query the bodhi server for updates. The `query` subcommand supports the following options:

`--updateid <id>`

Query for the update given by id.

`--approved-since <timestamp>`

Query for updates approved after the given timestamp.

`--modified-since <timestamp>`

Query for updates modified after the given timestamp.

`--builds <builds>`

Query for updates containing the given builds, given as a comma-separated list.

`--bugs <bugs>`

Query for updates related to the given bugs, given as a comma-separated list.

`--critpath`

Query for updates submitted for the critical path.

`--cves <cves>`

Query for updates related to the given CVEs, given as a comma-separated list.

`--packages <packages>`

Query for updates related to the given packages, given as a comma-separated list.

`--pushed`

Query for updates that have been pushed.

`--pushed-since <timestamp>`

Query for updates that have been pushed after the given timestamp.

`--releases <releases>`

Query for updates related to a list of releases, given as a comma-separated list.

`--locked`

Query for updates that are currently locked.

`--request [testing | stable | unpush]`

Query for updates marked with the given request type.

`--submitted-since <timestamp>`

Query for updates that were submitted since the given timestamp.

`--status [pending | testing | stable | obsolete | unpushed | processing]`

Filter by status.

`--suggest [logout | reboot]`

Filter for updates that suggest logout or reboot to the user.

`--type [newpackage | security | bugfix | enhancement]`

Filter by update type.

`--user <username>`

Filter for updates by the given username.

`bodhi updates request [options] <update> <state>`

Request that the given update be changed to the given state. `update` should be given by update id, and `state` should be one of testing, stable, unpush, obsolete, or revoke.

## Examples

Create a new update with multiple builds:

```
$ bodhi updates new --user bowlofeggs --type bugfix --notes "Fix permission issues␣
↪during startup." --bugs 1393587 --close-bugs --request testing --autokarma --stable-
↪karma 3 --unstable-karma -3 ejabberd-16.09-2.fc25,erlang-esip-1.0.8-1.fc25,erlang-
↪fast_tls-1.0.7-1.fc25,erlang-fast_yaml-1.0.6-1.fc25,erlang-fast_xml-1.1.15-1.fc25,
↪erlang-iconv-1.0.2-1.fc25,erlang-stringprep-1.0.6-1.fc25,erlang-stun-1.0.7-1.fc25
```

## Bugs

If you find bugs in bodhi (or in the mage page), please feel free to file a bug report or a pull request:

```
https://github.com/fedora-infra/bodhi
```

CHAPTER 2

# bodhi-push man page

## Synopsis

`bodhi-push` [OPTIONS]

## Description

`bodhi-push` is used to select which packages to push out to the mirror network. It has various flags that can be used to select the package set, and then it emits a fedmsg with the list of packages to be mirrored.

## Options

`--help`

Show help text and exit.

`--builds TEXT`

A comma-separated list of builds to include in the push.

`--cert-prefix TEXT`

The prefix of a fedmsg cert used to sign the message.

`--releases TEXT`

A comma-separated list of releases to include in this push. By default, current and pending releases are selected.

`--request TEXT`

Push updates for the specified request. Defaults to testing,stable.

`--resume`

Resume one or more previously failed pushes.

`--staging`

Use the staging bodhi instance instead of the production instance.

`--username TEXT`

Your FAS user id.

`--version`

Show version and exit.

## Bugs

If you find bugs in bodhi (or in the mage page), please feel free to file a bug report or a pull request:

```
https://github.com/fedora-infra/bodhi
```

CHAPTER 3

# initialize_bodhi_db man page

## Synopsis

`initialize_bodhi_db CONFIG_URI`

## Description

`initialize_bodhi_db` is used to create the initial database tables for the Bodhi server. It uses the given `CONFIG_URI` to find the database settings to use.

## Example

```
$ initialize_bodhi_db /etc/bodhi/production.ini
```

## Bugs

If you find bugs in bodhi (or in the mage page), please feel free to file a bug report or a pull request:

```
https://github.com/fedora-infra/bodhi
```

CHAPTER 4

# Release notes

## 2.4.0

Bodhi 2.4.0 is a feature and bugfix release.

### Features

- The web interface now displays whether an update has autopush enabled (#999).
- Autopush is now disabled on any update that receives authenticated negative karma (#1191).
- Bodhi now links to Koji builds via TLS instead of plaintext (#1246).
- Some usage examples have been added to the `bodhi` man page.
- Bodhi's server package has a new script called `bodhi-clean-old-mashes` that can recursively delete any folders with names that end in a dash followed by a string that can be interpreted as a float, sparing the newest 10 by lexigraphical sorting. This should help release engineers keep the Koji mashing folder clean.
- There is now a `bodhi.client.bindings` module provided by the Bodhi client package. It contains Python bindings to Bodhi's REST API.
- The `bodhi` CLI now prints autokarma and thresholds when displaying updates.
- `bodhi-push` now has a `--version` flag.
- There are now man pages for `bodhi-push` and `initialize_bodhi_db`.

### Bugs

- Users' e-mail addresses will now be updated when they log in to Bodhi (#902).
- The masher now tests for `repomd.xml` instead of the directory that contains it (#908).
- Users can now only upvote an update once (#1018).

- Only comment on non-autokarma updates when they meet testing requirements (#1009).
- Autokarma can no longer be set to NULL (#1048).
- Users can now be more fickle than ever about karma (#1064).
- Critical path updates can now be free of past negative karma ghosts (#1065).
- Bodhi now comments on non-autokarma updates after enough time has passed (#1094).
- `bodhi-push` now does not crash when users abort a push (#1107).
- `bodhi-push` now does not print updates when resuming a push (#1113).
- Bodhi now says "Log in" and "Log out" instead of "Login" and "Logout" (#1146).
- Bodhi now configures the Koji client to retry, which should help make the masher more reliable (#1201).
- Bodhi is now compatible with Pillow-4.0.0 (#1262).
- The bodhi cli no longer prints update JSON when setting the request (#1408195).
- Bodhi's signed handler now skips builds that were not assigned to a release.
- The comps file is now cloned into an explicit path during mashing.
- The buildsystem is now locked during login.

## Development improvements

- A great deal of tests were written for Bodhi. Test coverage is now up to 81% and is enforced by the test suite.
- Bodhi's server code is now PEP-8 compliant.
- The docs now contain contribution guidelines.
- The build system will now fail with a useful Exception if used without being set up.
- The Vagrantfile is a good bit fancier, with hostname, dnf caching, unsafe but performant disk I/O, and more.
- The docs now include a database schema image.
- Bodhi is now run by systemd in the Vagrant guest.
- The Vagrant environment now has several helpful shell aliases and a helpful MOTD to advertise them to developers.
- The development environment now uses Fedora 25 by default.
- The test suite is less chatty, as several unicode warnings have been fixed.

## Dependency change

- Bodhi server now depends on click for `bodhi-push`.

## Release contributors

The following contributors submitted patches for Bodhi 2.4.0:

- Trishna Guha
- Patrick Uiterwijk
- Jeremy Cline

- Till Mass
- Josef Sukdol
- Clement Verna
- andreas
- Ankit Raj Ojha
- Randy Barlow

## 2.3.3

Bodhi 2.3.3 converts koji auth to be done with krb5 and fixes one bug:

- Use krb5 for koji (#1129).
- Disable caching koji sessions during mashing process (#1134).

Thanks to Patrick Uiterwijk for contributing both of these commits!

## 2.3.2

Bodhi 2.3.2 is a bugfix release that addresses the following issues:

- `push.py` now defaults to the current releases (#1071).
- Fixed a typo in the masher in sending an ostree compose message (#1072).
- Fixed a typo in looking up an e-mail template (#1073).
- The fedmsg name is now passed explicitly (#1079).
- The man page was corrected to state that builds should be comma separated (#1095).
- Fixed a race condition between robosignatory and the signed handler (#1111).
- Fix querying the updates for resumption in `push.py` (e7cb3f13).
- `push.py` now prompts for the username if not given (abeca57e).

### Release contributors

The following contributors authored patches for 2.3.2:

- Patrick Uiterwijk
- Randy Barlow

## 2.3.1

Bodhi 2.3.1 fixes #1067, such that edited updates now tag new builds into the `pending_signing_tag` instead of the `pending_testing_tag`. This is needed for automatic signing gating to work.

## 2.3.0

Bodhi 2.3.0 is a feature and bug fix release.

### Features

- The package input field is now autofocused when creating new updates (#876).
- Releases now have a `pending_signing_tag` (3fe3e219).
- fedmsg notifications are now sent during ostree compositions (b972cad0).
- Critical path updates will have autopush disabled if they receive negative karma (b1f71006).
- The e-mail templates reference dnf for Fedora and yum for Enterprise Linux (1c1f2ab7).
- Updates are now obsoleted if they reach the unstable threshold while pending (f033c74c).
- Bodhi now gates Updates based on whether they are signed yet or not (#1011).

### Bugs

- Candidate builds and bugs are no longer duplicated while searching (#897).
- The Bugzilla connection is only initialized when needed (950eee2c).
- A sorting issue was fixed on the metrics page so the data is presented correctly (487acaaf).
- The Copyright date in the footer of the web interface is updated (1447b6c7).
- Bodhi will comment with the required time instead of the elapsed time on updates (#1017).
- Bodhi will only comment once to say that non-autopush updates have reached the threshold (#1009).
- `/masher/` is now allowed in addition to `/masher` for GET requests (cdb621ba).

### Dependencies

Bodhi now depends on fedmsg-atomic-composer >= 2016.3, which addresses a few issues during mashing.

### Development improvements

Bodhi 2.3.0 also has a few improvements to the development environment that make it easier to contribute to Bodhi or improve Bodhi's automated tests:

- Documentation was added to describe how to connect development Bodhi to staging Koji (7f3b5fa2).
- An unused `locked_date_for_update()` method was removed (b87a6395).
- The development.ini.example base_address was changed to localhost so requests would be allowed (0fd5901d).
- The `setup.py` file has more complete metadata, making it more suitable for submission to PyPI (5c201ac2).
- The #bodhi and #fedora-apps channels are now documented in the readme file (52093069).
- A new test has been added to enforce PEP-8 style and a few modules have been converted to conform (bbafc9e6).

### Release contributors

The following contributors authored patches for 2.3.0:

- Josef Sukdol
- Julio Faracco
- Patrick Uiterwijk
- Randy Barlow
- Richard Fearn
- Trishna Guha

## 2.2.4

This release fixes two issues:

- #989, where Karma on non-autopush updates would reset the request to None.
- #994, allowing Bodhi to be built on setuptools-28.

## 2.2.3

This release fixes #951, which prevented updates with large numbers of packages to be viewable in web browsers.

## 2.2.2

This is another in a series of bug fix releases for Bodhi this week. In this release, we've fixed the following issues:

- Disallow comment text to be set to the NULL value in the database (#949).
- Fix autopush on updates that predate the 2.2.0 release (#950).
- Don't wait on mashes when there aren't any (68de510c).

## 2.2.1

Bodhi 2.2.1 is a bug fix release, primarily focusing on mashing issues:

- Register date locked during mashing (#952).
- UTF-8 encode the updateinfo before writing it to disk (#955).
- Improved logging during updateinfo generation (#956).
- Removed some unused code (07ff664f).
- Fix some incorrect imports (9dd5bdbc and b1cc12ad).
- Rely on self.skip_mash to detect when it is ok to skip a mash (ad65362e).

## 2.2.0

Bodhi 2.2.0 is a security and feature release, with a few bug fixes as well.

### Security

This update addresses CVE-2016-1000008 by disallowing the re-use of solved captchas. Additionally, the captcha is warped to make it more difficult to solve through automation. Thanks to Patrick Uiterwijk for discovering and reporting this issue.

### Features

- Bodhi's `approve_testing.py` script will now comment on updates when they have reached a stable karma threshold (5b0d1c7c).
- The web interface now displays a push to stable button when the karma reaches the right level when autokarma is disabled (#772 and #796).
- Masher messages now have an "agent", so it is possible to tell which user ran the mash (45e4fc9f).
- Locked updates now list the time they were locked (#831).
- Bugs are closed and commented on in the same Bugzilla POST (#404).
- Karma values equal to 0 are no longer displayed with a green background to better distinguish them from positive karma reports (#799).
- Updates display a link to the feedback guidelines (#865).
- The new CLI now has a man page (95574831).
- The CLI now has a `--version` flag (#895).

### Bugs

- Locked updates that aren't part of a current push will now be pushed and warnings will be logged (bf4bdeef). This should help us to fix #838.
- Don't show users an option to push to stable on obsoleted updates (#848).
- taskotron updates are shown per build, rather than per update (ce2394c6, 8e199668).
- The Sphinx documentation now builds again (b3f80b1b).
- Validator messages are now more useful and helpful (#630).
- The Bodhi CLI no longer depends on the server code to function (#900).
- Private bugs will no longer prevent the updates consumer from continuing (#905).
- bootstrap is now included in the setuptools manifest for the server package (#919).

### Commit log

The above lists are the highlights of what changed. For a full list of the changes since 2.1.8, please see the changelog.

CHAPTER 5

# Developer documentation

This page contains information for developers who wish to contribute to Bodhi.

## Contribution guidelines

Before you submit a pull request to Bodhi, please ensure that it meets these criteria:

- All tests must pass.
- New code should have 100% test coverage. This one is particularly important, as we don't want to deploy any broken code into production.
- New functions, methods, and classes should have docblocks that explain what the code block is, and describing any parameters it accepts and what it returns (if anything).
- New code should follow PEP-8. You can use the `flake8` utility to automatically check your code. There is a `bodhi.tests.test_style.TestStyle.test_code_with_flake8` test, that is slowly being expanded to enforce PEP-8 across the codebase.

## Create a Bodhi development environment

There are two ways to bootstrap a Bodhi development environment. You can use Vagrant, or you can use virtualenv on an existing host.

### Vagrant

Vagrant allows contributors to get quickly up and running with a Bodhi development environment by automatically configuring a virtual machine. Before you get started, ensure that your host machine has virtualization extensions enabled in its BIOS so the guest doesn't go slower than molasses. To get started, simply use these commands:

```
$ sudo dnf install ansible libvirt vagrant-libvirt vagrant-sshfs
$ sudo systemctl enable libvirtd
$ sudo systemctl start libvirtd
$ cp Vagrantfile.example Vagrantfile
# Make sure your bodhi checkout is your shell's cwd
$ vagrant up
```

`Vagrantfile.example` sets up a port forward from the host machine's port 6543 into the Vagrant guest's port 6543, so you can now visit http://localhost:6543 with your browser to see your Bodhi development instance if your browser is on the same host as the Vagrant host. If not, you will need to connect to port 6543 on your Vagrant host, which is an exercise left for the reader.

### Quick tips about the Bodhi Vagrant environment

You can ssh into your running Vagrant box like this:

```
# Make sure your bodhi checkout is your shell's cwd
$ vagrant ssh
```

Once you are inside the development environment, there are a helpful set of commands in your `.bashrc`:

- `bdocs`: Build Bodhi's documentation.
- `blog`: View Bodhi's log.
- `brestart`: Restart the Bodhi service.
- `bstart`: Start the Bodhi service.
- `bstop`: Stop the Bodhi service.
- `btest`: Run Bodhi's test suite.

Keep in mind that all `vagrant` commands should be run with your current working directory set to your Bodhi checkout. The code from your development host will be mounted in `/home/vagrant/bodhi` in the guest. You can edit this code on the host, and the vagrant-sshfs plugin will cause the changes to automatically be reflected in the guest's `/home/vagrant/bodhi` folder.

The development server is run inside the Vagrant environment by the `bodhi.service` systemd unit. You can use `pshell` and `tools/shelldb.py` to get a Python shell quickly set up with a nice environment for you to hack in:

```
[vagrant@localhost bodhi]$ pshell development.ini
Python 2.7.12 (default, Sep 29 2016, 13:30:34)
[GCC 6.2.1 20160916 (Red Hat 6.2.1-2)] on linux2
Type "help" for more information.

Environment:
  app          The WSGI application.
  registry     Active Pyramid registry.
  request      Active request object.
  root         Root of the default resource tree.
  root_factory Default root factory used to create `root`.

Custom Variables:
  m            bodhi.server.models
  t            transaction

>>> execfile('tools/shelldb.py')
```

Once you've run that `execfile('tools/shelldb.py')` tools command, it's pretty easy to run database queries:

```
>>> db.query(m.Update).filter_by(alias='FEDORA-2016-840ff89708').one().title
<output trimmed>
u'gtk3-3.22.1-1.fc25'
```

It is possible to connect your Vagrant box to the staging Koji instance for testing, which can be handy at times. You will need to copy your `.fedora.cert` and `.fedora-server-ca.cert` that you normally use to connect to Koji into your Vagrant box, storing them in `/home/vagrant`. Once you have those in place, you can set `buildsystem = koji` in your `development.ini` file.

When you are done with your Vagrant guest, you can destroy it permanently by running this command on the host:

```
$ vagrant destroy
```

## Virtualenv

Virtualenv is another option for building a development environment.

### Dependencies

```
sudo dnf install libffi-devel postgresql-devel openssl-devel koji
pcaro-hermit-fonts freetype-devel libjpeg-turbo-devel python-pillow
zeromq-devel liberation-mono-fonts
```

### Setup virtualenvwrapper

```
sudo dnf -y install python-virtualenvwrapper python-createrepo_c
```

Add the following to your *~/.bashrc*:

```
export WORKON_HOME=$HOME/.virtualenvs
source /usr/bin/virtualenvwrapper.sh
```

### Set PYTHONPATH

Add the following to your *~/.bashrc*

```
export PYTHONPATH=$PYTHONPATH:$HOME/.virtualenv
```

Then on the terminal

```
source ~/.bashrc
```

### Clone the source

```
git clone https://github.com/fedora-infra/bodhi.git
cd bodhi
```

### Bootstrap the virtualenv

```
./bootstrap.py
workon bodhi-python2.7
```

### Setting up

```
python setup.py develop
```

```
pip install psycopg2
```

### Create the development.ini file

Copy `development.ini.example` to `development.ini`:

```
cp development.ini.example development.ini
```

### Run the test suite

```
python setup.py nosetests
```

### Import the bodhi2 database

```
curl -O https://infrastructure.fedoraproject.org/infra/db-dumps/bodhi2.dump.xz
sudo -u postgres createdb bodhi2
xzcat bodhi2.dump.xz | sudo -u postgres psql bodhi2
```

**Note:** If you do not have a PostgreSQL server running, please see the instructions at the bottom of the file.

### Adjust database configuration in development.ini file

Set the configuration key sqlalchemy.url to point to the postgresql database. Something like:

```
sqlalchemy.url = postgresql://postgres:anypasswordworkslocally@localhost/bodhi2
```

### Upgrade the database

```
alembic upgrade head
```

### Run the web app

```
pserve development.ini --reload
```

### Setup the postgresql server

#### 1. Install postgresql

```
dnf install postgresql-server
```

#### 2. Setup the Database

As a privileged user on a Fedora system run the following:

```
sudo postgresql-setup initdb
```

#### 3. Adjust Postgresql Connection Settings

As a privileged user on a Fedora system modify the pg_hba.conf file:

```
vi /var/lib/pgsql/data/pg_hba.conf
```

Then adjust the content at the bottom of the file to match the following.

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD

# "local" is for Unix domain socket connections only
local   all             all                                     peer
# IPv4 local connections are *trusted*, any password will work.
host    all             all             127.0.0.1/32            trust
# IPv6 local connections are *trusted*, any password will work.
host    all             all             ::1/128                 trust
```

If you need to make other modifications to postgresql please make them now.

#### 4. Start Postgresql

As a privileged user on a Fedora system run the following:

```
sudo systemctl start postgresql.service
```

## Database Schema

The Bodhi database schema can be seen below.


Fig. 5.1: Database schema.

CHAPTER 6

# Bodhi API

## /updates

## /releases

## /builds

## /users

CHAPTER 7

# Indices and tables

- genindex
- modindex
- search