PANDA Project Documentation

Release 0.1.0

PANDA Project

April 02, 2012

CONTENTS

CHAPTER

ONE

ABOUT

PANDA wants to be your newsroom data appliance. It provides a place for you to store data, search it and share it with the rest of your newsroom.

- Repository: https://github.com/pandaproject/panda
- Issues: https://github.com/pandaproject/panda/issues
- Documentation: http://panda.rtfd.org/
- Wiki: https://github.com/pandaproject/panda/wiki

CHAPTER

TWO

WHAT IS PANDA?

PANDA is:

- A place for journalists to store data.
- A search engine for your news data.
- A private archive of your newsworthy datasets.
- Extensible.
- Self-hosted.

PANDA is *not*:

- A publishing system.
- A universal backend for your newsapps.
- A platform for data visualizations.
- A highly structured datastore.
- Software as a Service.

CHAPTER

THREE

INSTALLATION

Setup:

3.1 Local development & testing

3.1.1 Install basic requirements

Use the tools appropriate to your operating system to install the following packages. For OSX you can use Homebrew. For Ubuntu you can use Apt.

- pip
- virtualenv
- virtualenvwrapper
- PostgreSQL

3.1.2 Set up PANDA

This script will setup the complete application, *except* for Solr. Be sure to read the comments, as some steps require opening additional terminals:

```
# Get source and requirements
git clone git://github.com/pandaproject/panda.git
cd panda
mkvirtualenv --no-site-packages panda
pip install -r requirements.txt

# Create log directory
sudo mkdir /var/log/panda
sudo chown $USER /var/log/panda

# Create data directories
mkdir /tmp/panda
mkdir /tmp/panda_exports

# Enter "panda" when prompted for password
createuser -d -R -S -P panda
createdb -O panda panda
python manage.py syncdb --noinput
```

```
# Start PANDA
python manage.py runserver
```

Open a new terminal in the PANDA directory and enter:

```
# Start the task queue
workon panda
python manage.py celeryd
```

Open *another* terminal in the PANDA directory and enter:

```
# Run a local email server
workon panda
fab local_email
```

3.1.3 Set up Solr

Installing Solr can be tricky and will vary quite a bit depending on your operating system. The following will get you up and running on OSX Lion, using Homebrew. If you've just started the PANDA server, open a new terminal in the PANDA directory and enter these commands:

```
# Get into the env
workon panda

# Install solr 3.4.0
brew update
brew install solr

# Create Solr home directory
sudo mkdir /var/solr
sudo chown $USER /var/solr

# This command will install all Solr configuration
fab local_reset_solr

# To start Solr
fab local_solr
```

3.1.4 Running Python unit tests

To run the unit tests, start Solr and execute the test runner, like so:

```
# Ensure you are in the PANDA source directory and your virtualenv is active
# You may need to customize the fabfile so it can find your Solr installation.
fab local_solr

# Quite a bit of output will be printed to the screen.
# Wait until you see something like
# 2011-11-02 14:15:54.061:INFO::Started SocketConnector@0.0.0.0:8983
# Then, open another terminal and change to your PANDA source directory.
workon panda
python manage.py test panda
```

3.1.5 Running Javascript unit tests

Running the Javascript unit tests requires that the application server is running (to render the the JST template map). To run the Javascript tests, first start the test server with `python manage.py runserver`, then open the file `client/static/js/SpecRunner.html` in your browser (e.g. `file://localhost/Users/onyxfish/src/panda/client/static/js/SpecRunner.html`.

3.2 Production deployment

We provide documentation for two different ways of installing PANDA for production use. Non-advanced users and those who desire the simplest setup possible should install their PANDA on Amazon's EC2 service. This is the best-supported option.

Those who are concerned about the security of storing their data in the cloud or who have access to dedicated hardware may choose to install PANDA on their own server. Although we make every effort to support this, we can not offer any guarantee of compatibility.

I am...

3.2.1 Installing on Amazon EC2

Registering for EC2

If you don't already have an Amazon Web Services account, you will need to register for one and set up your credentials. Visit the EC2 homepage and follow the "Sign Up Now" link.

Note: Although every effort has been made to make this process as streamlined as possible, if you've never set up a server before, you may find it rather daunting. In this case we suggest pairing up with an engineer until you are through the setup process.

Configuring your Security Group

Before setting up your PANDA server, you will need to configure your security group so that web requests will be able to reach it.

To do this, visit the Security Groups tab of the EC2 Management Console. Select the "default" security group from the list, and then click the "Inbound" tab in the bottom pane of the window. Add rules for `HTTP`, `HTTPS` and `SSH`. If you don't mind your PANDA being accessible to anyone on the internet, you can enter `0.0.0.0/0` in the **Source** field for each. More discerning users may wish to enter a private IP or subnet assigned to their organization.

Note: If you're not sure what to enter for the **Source** field it would be wise to consult with your IT department to find out if your organization has a private subnet.

Installation

Method #1 - Using an AMI

This is the absolute simplest way to make a PANDA. Visit the Instances tab and click "Launch Instance". Select "Launch Class Wizard" and click "Continue". Click the "Community AMIs" tab and search for `ami-fcf32095`.

It may take a moment to return a result. When it does, click "Select". On the next page you'll need to select an **Instance Type**. You are welcome to use (and pay for) a more powerful server, but PANDA has been optimized with the expectation that most organizations will run it on an `m1.small` instance. (At a cost of roughly $70 per month.) This should provide adequate capacity for small- to medium-sized groups. We don't recommend trying to run it on a `t1.micro` unless you will only be using it for testing.

Once you've selected your instance type, click "Continue". Keep clicking "Continue" and accepting all the default options until the "Continue" button becomes a "Launch" button. Click "Launch".

Method #2 - Via SSH

This method is slightly more complex, but also provides greater feedback for users who want to understand more about how PANDA works.

Visit the Instances tab and click "Launch Instance". Select "Launch Class Wizard" and click "Continue". Click the "Community AMIs" tab and search for `ami-1af12273`. This is the official Ubuntu 11.10 AMI. It may take a moment to return a result. When it does, click "Select".

On the next page you'll need to select an **Instance Type**. See the notes above regarding instance types. We recommend you select `m1.small`.

Click "Continue" and keep clicking "Continue" and accepting all the default options until the "Continue" button becomes a "Launch" button. Click "Launch".

Once your new server is available, SSH into it and execute the following commands:

```
wget https://raw.github.com/pandaproject/panda/master/setup_panda.sh
sudo bash setup_panda.sh
```

The disadvantage of this method is that you will need to wait while the setup script is run. This normally takes 15-20 minutes.

Note: An installation log will be created at `/var/log/panda-install.log` in case you need to review any part of the process.

Verifying your instance

Once you've completed your selected installation method you'll want to verify that your new PANDA is available. You can browse directly using to your instance using its "Public DNS Name". Navigate to the EC2 Instances tab and select your instance. The public DNS name will be listed among the instance details in the bottom pane. It will look something like this: `ec2-50-16-157-39.compute-1.amazonaws.com`. Visit this in your browser, like so:

```
http://ec2-50-16-157-39.compute-1.amazonaws.com/
```

Once you have verified that your instance is online you may wish to configure DNS, E-mail and/or Secure connections (SSL).

3.2.2 Installing on your own hardware

Installation

Server requirements

PANDA requires a server running Ubuntu 11.10. Whether you want to run PANDA in a virtual machine or on the old Compaq under your desk, as long as it can run Ubuntu 11.10, you should be fine. (Performance, of course, may vary widely depending on the hardware.)

Running the install script

SSH into your server and run the following setup commands:

```
wget https://raw.github.com/pandaproject/panda/master/setup_panda.sh
sudo bash setup_panda.sh
```

Note that the setup script **must** be run with sudo.

An installation log will be created at `/var/log/panda-install.log` in case you need to review any part of the process.

Configuration:

3.3 Configuring DNS

3.3.1 Getting a domain name

To use a custom domain (or subdomain) with your PANDA, you will first need to requisition one from your IT department or purchase one from a registrar such as Namecheap or dnsimple.

Note: If you're using Amazon EC2, you will now want to set up an "Elastic IP" for your instance. This will give you a permanent address for your server. To do this, visit the EC2 Dashboard and click "Elastic IPs" in the left-hand rail. Click "Allocate New Address" in the toolbar and then "Yes, Allocate". Select the new IP address in the list and click "Associate Address" in the toolbar. Select your PANDA instance from the drop-down and click "Yes, Associate." You'll use this new IP address in the next step.

Create an ANAME record for your new domain, pointed to your server's IP address. The details of how to do this will depend on your registrar (or the procedures of your IT staff), but typically it's as simple as pasting the IP address into the correct box and clicking "save." It may take some time for your domain to become available, but often it is instantaneous. Trying visiting your new domain in your web browser and PANDA should load. If it does not, wait a while and try again.

3.3.2 Configuring PANDA

Once you have your new domain name directed to your PANDA, you'll want to configure outbound email to use the new domain. You can do this on the settings page:

```
http://localhost:8000/admin/settings
```

Replace `localhost:8000` with your PANDA's domain name.

You'll be prompted to log in. If this is your first time, you can use the default username, `panda@pandaproject.net`, and the default password `panda`.

Once you've logged in, you'll see a list of configuration options. In the section titled "Site domain settings," fill in your new domain name and click "Update Settings".

To test the new domain name, click the "Home" link at the top of the screen and then the link to "Add" a new User. Fill in your own email address and click "Save." You should get an activation email in your inbox. Click the activation link and verify that you return to your PANDA.

3.4 Configuring Email

3.4.1 Getting an SMTP server

Before you configure PANDA you're going to need access to an SMTP server for sending email. There are several ways you can get access to one of these.

Using your own SMTP

If you're organization already has an SMTP server then you may be able to use it for PANDA. Note, however, that if you host PANDA on EC2 and your SMTP server is internal to your organization you may not be able to reach it. It's best to discuss this possibility with the IT staff in charge of your email servers.

If you can use your own SMTP server then make sure you have its address, port number, username and password ready so you can fill them in later.

Using CritSend

If you don't have access to an SMTP server for your organization your best bet is probably to use an email service that provides SMTP access. We suggest CritSend because they are inexpensive (first 1,000 emails are free, next 20,000 are $10) and their registration process doesn't require that your website is public facing.

To sign up for CritSend visit their website and enter your email address in the signup box. You'll receive an activation email. Follow it and then click "My Account" and fill out the information they require. In order to prevent SPAM they do manually validate accounts, so make sure you use legitimate contact information. Once you save it will take up to 24 hours for your registration to be validated.

Once you receive notification that your registration has been validated, you'll be able to use the following settings to configure your PANDA:

- Host: `smtp.critsend.com` (if you're hosting on Amazon EC2 then use `aws-smtp.critsend.com`)
- Port: `587`
- User: `YOUR_CRITSEND_USERNAME`
- Password: `YOUR_CRITSEND_PASSWORD`
- Use TLS: `True`
- From address: `YOUR_FROM_ADDRESS`

If you wish to setup advanced features such as SPF and Domain Keys to ensure your email is not flagged as SPAM, CritSend has documentation on how to do this.

Using Gmail

Another possiblity is to use Gmail's SMTP servers for sending email. Note that this solution is somewhat hacky and probably shouldn't be relied on as a permanant solution.

You'll need to register for a new Gmail. Do **not** use your primary email account for this, but do register with a real name and contact information.

That's it. To use the Gmail SMTP servers, you'll use the following configuration for PANDA:

- Host: `smtp.gmail.com`
- Port: `587`
- User: `YOUR_NEW_EMAIL_ADDRESS`
- Password: `YOUR_NEW_PASSWORD`
- Use TLS: `True`
- From address: `YOUR_NEW_EMAIL_ADDRESS`

3.4.2 Configuring PANDA

Once you have your SMTP connection details ready. You're ready to configure your PANDA. Visit the settings page at:

```
http://localhost:8000/admin/settings
```

Replace `localhost:8000` with your PANDA's domain name.

You'll be prompted to login. If this is you're first time you can use the default username, `panda@pandaproject.net` and the default password `panda`.

Once you've logged you'll see a list of configuration options. In the section titled "Email settings", fill in the details of your SMTP connection and then click "Update Settings".

To test the new connection click the "Home" link at the top of the screen and then the link to "Add" a new User. Fill in your own email address and click Save. You should get an activation email in your inbox!

3.5 Configuring SSL

3.5.1 Getting your certificate

If you've decided to host your PANDA on Amazon's EC2 service or anywhere else that is accessible over the public internet then you should secure your site with an SSL certificate. Broadly, there are two ways you might go about this.

Buying a certificate

The best option is purchase a certificate from an official signing authority such as VeriSign or Digicert. However, a fully validated SSL certificate can cost hundreds of dollars per year.

Fortunantely, for the purposes of PANDA you should be fine with a much less expensive "Domain Validation" only certificate. This are available from many DNS registrars, such as Namecheap as well as dedicated providers like RapidSSL. Domain Validation certificates typically verify the email address in the WHOIS records of your domain registration and then issue instantly. Ideally, you wouldn't want to use one of these to secure a public website, but because PANDA is only used by members of your organization it is sufficient.

Once you've acquired a certificate you can copy it your server by running the following commands in the directory with the files:

```
scp -i <MY_EC2_KEY.pem> <MY_CERT.crt> ubuntu@<MY_PANDA_SERVER.com>:~/panda.crt
scp -i <MY_EC2_KEY.pem> <MY_KEY.key> ubuntu@<MY_PANDA_SERVER.com>:~/panda.key
```

Creating your own certificate

If you are operating your PANDA in a no-budget environment, then you may choose to generate your own "self-signed" certificate instead. This will provide all the benefits of encryption to your users, however, **each user will need to click through a browser warning that the site is not verified**. You will need to explicitly communicate to your users that this error message is normal. (In most browers they will only see it once.)

SSH into your server and run these commands:

```
# You'll be prompted for a passphrase.
# Use anything, but remember what it is.
openssl genrsa -des3 -out panda.key 1024

# After entering your passphrase you'll be prompted for a
# variety of information which you can either fill in or leave blank.
openssl req -new -key panda.key -out panda.csr

cp panda.key panda.key.org

# You'll need your passphrase again here.
openssl rsa -in panda.key.org -out panda.key
openssl x509 -req -days 365 -in panda.csr -signkey panda.key -out panda.crt

rm panda.csr
rm panda.key.org
```

3.5.2 Installing your certificate

Once you've purchased or created a certificate you'll need to install both it and your signing key in the correct location:

```
sudo mv panda.crt /etc/nginx/panda.crt
sudo chown root:root /etc/nginx/panda.crt
sudo chmod 644 /etc/nginx/panda.crt

sudo mv panda.key /etc/nginx/panda.key
sudo chown root:root /etc/nginx/panda.key
sudo chmod 644 /etc/nginx/panda.key
```

3.5.3 Turning it on

The last thing you'll need to do is reconfigure PANDA's webserver (Nginx) to use the new SSL certificate. Fortunantely, there is a one-size-fits-all solution for this. All you need to is SSH into your PANDA server and run the following commands:

```
sudo wget https://raw.github.com/pandaproject/panda/master/setup_panda/nginx_ssl -O /etc/nginx/sites-
sudo service nginx restart
```

Your PANDA should now redirect all unsecured requests to a secure `https://` url.

CHAPTER

FOUR

USAGE

For administrators:

4.1 User management

4.1.1 Default users

For demonstration and configuration purposes and PANDA ships with two default users, one administrator and one ordinary PANDA user. You will use the default administrator account to create your own adminstrator account. We strongly recommend that you then delete the demo users.

The default accounts are:

```
Administrator
Username: panda@pandaproject.net
Password: panda

PANDA user
Username: user@pandaproject.net
Password: user
```

4.1.2 Creating a new admin user

Visit the admin users page:

```
http://localhost:8000/admin/auth/user/
```

Replace `localhost:8000` with your PANDA's domain name.

When prompted to login use the default administrator credentials documented above.

In the upper-right corner click "Add user". Type in the email address and name and click "Save". If you've already setup Email then you will receive a registration email with an activation link. Ignore it. On the details page for your new user click the "change password form" link underneath the **Password** field. Enter your password and click "Change Password". Check the **Active**, **Staff status** and **Superuser status** checkboxes and click "Save".

Log out and log back in with your new superuser. You may now delete the default administrator.

4.1.3 Creating new PANDA users

Note: Setup Email before you do this.

Visit the admin users page:

```
http://localhost:8000/admin/auth/user/
```

Replace `localhost:8000` with your PANDA's domain name.

In the upper-right corner click "Add user". Type in the email address. You may choose to also enter the user's name or leave it blank. They will have an opportunity to add/update it. Click "Save". Your new user will receive a registration email with an activation link.

4.1.4 Creating new users in bulk

If you need to create a lot of users you can also use your administrative API key to create new users via the API.

4.2 Managing dataset categories

PANDA allows you to organize your datasets into categories that you define. By default it comes configured with a small handful of categories we imagine everyone will need, but you will almost certainly want to add your own.

Note: Categories are intended to be used as **broad** groupings of datasets ("Crime" or "Education"), not projects ("Medicaid Investigation 2012") or tags ("president"). Trying to use them in these latter ways is strongly discouraged.

Categories can be maintained by visiting the categories section of the admin:

```
http://localhost:8000/admin/panda/category/
```

Replace `localhost:8000` with your PANDA's domain name.

This interface should be largely self-explanatory. When creating a new category a url slug will automatically be generated based on the name you provide. In less you feel the need to edit them for brevity, these default slugs are usually fine.

4.3 Managing user API keys

Under the hood PANDA relies on API Keys to allow users to access the application. These keys also allow the user to programmatically access PANDA if they have the know-how. Every user is automatically issued an API key when they are created. It is not possible to use PANDA without a valid API key.

From time to time it may be necessary to revoke a user's API key and issue them a new one. Normally you would want to do this if you were concerned that their key had been leaked to a third-party or otherwise compromised by someone who should not have access.

Warning: If it is the user whom you are trying to keep from accessing PANDA, the right way to do so is to deactivate their account. The only reason to delete a user's API key is if you plan on creating a new one for them.

To regenerate a user's API key go to the user admin page:

```
http://localhost:8000/admin/auth/user/
```

Replace `localhost:8000` with your PANDA's domain name.

Select a user from the list and scroll down to the bottom of their page. Check the **Delete** checkbox on the right-hand side of the "Api Keys" section. Click "Save and continue editing".

Scroll to the bottom of the page once more and click "Add another Api Key". Click "Save". Your user now has a new, valid API key and the old one can no longer be used to access your PANDA.

For users:

Coming soon...

CHAPTER

FIVE

SERVER MAINTENANCE

5.1 Systems Administration (Ops)

If you are an advanced user and you're going to run a PANDA server, you may want to know how it all works. This page lays out the users, services and files that are part of the standard PANDA setup.

This might also be thought of as a guide to what you may need to change in the event you want to run PANDA in a non-standard configuration.

5.1.1 Users

panda

Runs the `uwsgi` and `celeryd` services and owns their logs. Owns `/var/lib/panda/media` (compressed assets) and `/var/lib/panda/uploads` (uploaded files). Only this user should be used to execute Django management commands, etc:

```
sudo -u panda -E python manage.py shell
```

Note: Note when executing command as the `panda` user, it's often necessary to use the `-E` option to sudo, so that the `DEPLOYMENT_TARGET` environment variable will be preserved.

---

postgres

Runs the `postgresql` service and owns its database files and logs.

root

Owns everything else, including the panda source code in `/opt/panda`.

solr

Runs the `solr` service and owns its files (`/opt/solr`), indices and logs.

ubuntu

The standard Ubuntu login user. Must be used to SSH into the system and run sudo. Has read-only access to files and logs, but can not execute any system commands.

www-data

Runs the `nginx` service and owns its logs.

5.1.2 Services

celeryd

Task processing.

nginx

Web server. Runs on port 80.

postgresql

Database. Runs on port 5432 locally. Not accessible from remote hosts.

solr

Search engine. Runs on 8983. Not accessible from remote hosts.

uwsgi

Python application server. Runs over a socket at `/var/run/uwsgi/wsgi.sock`.

5.1.3 Files

- `/var/lib/panda/media` - compressed assets
- `/var/lib/panda/uploads` - file uploads
- `/opt/panda` - application source code
- `/opt/solr` - Solr application
- `/var/log/celeryd` - destination for celery logs if output was not captured and routed to panda logs (exists only as a failsafe)
- `/var/log/nginx` - destination for access logs
- `/var/log/panda` - destination for application logs (errors and task processing)
- `/var/log/uwsgi.log` - destination for uwsgi logs (normally just startup and shutdown)

5.2 Backing up your PANDA

If you are using PANDA in production you will want to ensure that you perform frequent backups. For archival purposes the most crucial thing to store is the data files themselves, however, you will probably also want to backup your search indexes and database.

5.2.1 Amazon EC2 backups

Creating a backup

If you are hosting your PANDA on Amazon EC2 then you can make use of EC2's "snapshot" ability for your backups. In essence a snapshot is an exact copy of any EBS volume. You can use this ability to create backups your PANDA server and, if you have migrated your files and indexes, the volumes they reside on as well. We provide a script to automate this process. You will need to have at least some experience with administering a server in order to use this script.

To perform a one-time backup SSH into your server and execute the following commands:

```
cd /opt/panda/scripts
sudo python backup_volumes.py
```

When asked for your Amazon Access and Secret keys you can paste them into the console. They will not be displayed or logged. Your PANDA will be unavailable during the backup process, which may take some time (especially if your volumes are large).

Once the script has completed your backup will be created and your PANDA will be running again.

Restoring a backup

Restoring a backup created with snapshots is a matter of restoring each volume and ensuring they are mounted correctly. However, because one of the volumes that will need to be restored is the root volume of the instance, we have to do a little magic. To restore the root volume:

- On the EC2 Instances tab locate the instance you wish to restore.
- Right-click that instance and select "Launch More Like This".
- Proceed through the wizard, modifying the **Instance Type**, if desired.
- Launch your new instance.
- Once the new instance is available, select it and find its instance ID near the top of the bottom pane. It will look like `i-76acf913`. Note this, you will need it in a minute.
- Right-click on your instance and "Stop" it. (Don't "Terminate" it!)
- Once it's stopped, navigate to the Volumes tab.
- Find the volume which has the instance ID you noted in the "Attachment Information" column.
- Select this volume and click "Detach Volume."
- Once it is detached, delete it.
- On the EC2 Snapshots tab locate the latest snapshot of your root volume, in the description it will say "mounted at /". Note the Snapshot ID. It will look like `snap-f8b6c49f`. You will need this in a moment.
- Select your snapshot and click "Create Volume". Make it 8GB.

- Got back to the Volumes tab. You should see your new volume in the list. Its identifiable by the snapshot ID you noted in the last step.
- Select it and click "Attach Volume".
- Find your instance in the list using the instance ID you noted before. It should still be in "stopped" state. Set the **Device** to `/dev/sda1`. Attach it!

If you have run either the files or indexes storage migration scripts then you will also need to restore your additional storage volumes. These are more straight-forward:

- Create a volume from each snapshot in the same manner you created the root volume.
- Make note of the device in the description of each volume. It looks like `/dev/sdg`.
- When attaching each volume in the instance, use this value as the **Device**.

Once all volumes have been created and attached, navigate back to the Instances tab, find your instance in the list, right-click it and select "Start". Once it is finished booting up you will have successfully restored your PANDA backup!

5.2.2 Self-install backups

If you chose to self-install PANDA your backup options may vary widely. If possible you should consider periodically disabling the PANDA services and backing up the entire filesystem. If you have files and indexes stored on separate devices you will, of course, want to back those up as well. Using compressed and/or incremental backups will likely save you a significant amount of storage space.

5.3 Adding more storage to your PANDA

PANDA stores both raw data files and search indexes. Both of these can be quite large. At some point you may find you need to upgrade your storage for one or both. This document describes how to do this.

> **Warning:** All procedures described on this page **can destroy your data**. Please take every precaution to ensure your data is backed up before beginning.

5.3.1 Amazon EC2 upgrade

If you are hosting your PANDA on Amazon EC2 and used our installation guide then we provide scripts to upgrade your file and index storage. You will need to have at least some experience with administering a server in order to use these scripts.

To upgrade your file storage SSH into your server and execute the following commands:

```
cd /opt/panda/scripts
sudo python migrate_files_volume.py
```

To upgrade your search index storage the second command would be:

```
sudo python migrate_solr_volume.py
```

Both scripts will prompt you for some information. When asked for your Amazon Access and Secret keys you can paste them into the console. They will not be displayed or logged. You will also need to enter a size (in gigabytes) for your new volume.

Once the script has completed your files or indexes will be on the new volume and your PANDA will be running again.

Note: If you run this script more than once (i.e. migrating from an added volume to a new added volume) then the old volume will not be detached from your instance or destroyed. Describing how to do this is beyond the scope of this documentation.

5.3.2 Self-install upgrade

Although we can't provide a detailed upgrade guide for self-installed PANDA instances, the basic outline of what you need to do is simple:

- Take your PANDA out of service.
- Add a new storage device and mount it at a temporary location.
- Copy the contents of either `/var/lib/panda` (for files) or `/opt/solr/panda/solr` to the temporary location.
- Unmount the device, delete the original files and remount the device at the original location.
- Restart your PANDA.

Don't forget to update `/etc/fstab` so that the new device will be automatically mounted after a reboot.

5.4 Upgrading your PANDA

We have yet to settle on a definitive upgrade strategy for PANDA, but broadly speaking upgrades will proceed as follows:

When we release a new version (e.g. 1.1), we will also release an upgrade script that migrates the previous version (e.g. 1.0) forward. These scripts will be self-contained and function much like the backup and storage migration scripts. Each script will ensure that your data is securely migrated to the new version of the system.

We anticipate releasing more detailed instructions alongside our next release.

CHAPTER

SIX

EXTENDING PANDA

6.1 API Documentation

The PANDA application is built on top of a REST API that can be used to power custom applications or import/export data in novel ways.

The PANDA API follows the conventions of Tastypie except in important cases where doing so would create unacceptable limitations. If this documentation seems incomplete, refer to Tastypie's page on Interacting with the API to become familiar with the common idiom.

Note: You will probably want to try these URLs in your browser. In order to make them work you'll need to use the `format`, `email`, and `api_key` query string parameters. For example, to authenticate as the default administrative user that comes with PANDA, append the following query string to any url described on this page:

```
?format=json&email=panda@pandaproject.net&api_key=edfe6c5ffd1be4d3bf22f69188ac6bc0fc04c84b
```

All endpoints that return lists support the `limit` and `offset` parameters for pagination. Pagination information is always returned in the embedded `meta` object.

6.1.1 Users

User objects can be queried to retrieve information about PANDA users, however, passwords and API keys are not included in responses.

Example User object:

```
{
    date_joined: "2011-11-04T00:00:00",
    email: "panda@pandaproject.net",
    first_name: "Redd",
    id: "1",
    is_active: true,
    last_login: "2011-11-04T00:00:00",
    last_name: "",
    resource_uri: "/api/1.0/user/1/"
}
```

Schema

```
http://localhost:8000/api/1.0/user/schema/
```

List

```
http://localhost:8000/api/1.0/user/
```

Fetch

```
http://localhost:8000/api/1.0/user/[id]/
```

Create

To create a new user, POST a JSON document containing at least `username` and `email` properties to http://localhost:8000/api/1.0/user/. Other properties such as `first_name` and `last_name` may also be set. If a `password` property is specified it will be set on the new user, but it will not be included in the response. If `password` is omitted the user will need to set a password before they can log in (not yet implemented).

6.1.2 Tasks

The Task API is read-only.

Example Task object:

```
{
    end: "2011-12-12T15:11:25",
    id: "1",
    message: "Import complete",
    resource_uri: "/api/1.0/task/1/",
    start: "2011-12-12T15:11:25",
    status: "SUCCESS",
    task_name: "panda.tasks.import.csv",
    traceback: null
}
```

Schema

```
http://localhost:8000/api/1.0/task/schema/
```

List

```
http://localhost:8000/api/1.0/task/
```

List filtered by status

List tasks that are PENDING (queued, but have not yet started processing):

```
http://localhost:8000/api/1.0/task/?status=PENDING
```

Note: Possible task statuses are `PENDING`, `STARTED`, `SUCCESS`, and `FAILURE`.

List filtered by date

List tasks that ended on October 31st, 2011:

```
http://localhost:8000/api/1.0/task/?end__year=2011&end__month=10&end__day=31
```

Fetch

```
http://localhost:8000/api/1.0/task/[id]/
```

6.1.3 Data Uploads

Due to limitations in upload file-handling, it is not possible to create Uploads via the normal API. Instead data files should be uploaded to http://localhost:8000/data_upload/ either as form data or as an AJAX request. Examples of how to upload files with curl are at the end of this section.

Example DataUpload object:

```
{
    columns: [
        "id",
        "first_name",
        "last_name",
        "employer"
    ],
    creation_date: "2012-02-08T17:50:09",
    creator: {
        date_joined: "2011-11-04T00:00:00",
        email: "user@pandaproject.net",
        first_name: "User",
        id: "2",
        is_active: true,
        last_login: "2012-02-08T22:45:28",
        last_name: "",
        resource_uri: "/api/1.0/user/2/"
    },
    data_type: "csv",
    dataset: "/api/1.0/dataset/contributors/",
    dialect: {
        delimiter: ",",
        doublequote: false,
        lineterminator: "
        ",
        quotechar: """,
        quoting: 0,
```

```
        skipinitialspace: false
    },
    encoding: "utf-8",
    filename: "contributors.csv",
    id: "1",
    imported: true,
    original_filename: "contributors.csv",
    resource_uri: "/api/1.0/data_upload/1/",
    sample_data: [
        [
            "1",
            "Brian",
            "Boyer",
            "Chicago Tribune"
        ],
        [
            "2",
            "Joseph",
            "Germuska",
            "Chicago Tribune"
        ],
        [
            "3",
            "Ryan",
            "Pitts",
            "The Spokesman-Review"
        ],
        [
            "4",
            "Christopher",
            "Groskopf",
            "PANDA Project"
        ]
    ],
    size: 168
}
```


Schema

```
http://localhost:8000/api/1.0/data_upload/schema/
```

List

```
http://localhost:8000/api/1.0/data_upload/
```

Fetch

```
http://localhost:8000/api/1.0/data_upload/[id]/
```

Download original file

```
http://localhost:8000/api/1.0/data_upload/[id]/download/
```

Upload as form-data

When accessing PANDA via curl, your email and API key can be specified with the headers `PANDA_EMAIL` and `PANDA_API_KEY`, respectively:

```
curl -H "PANDA_EMAIL: panda" -H "PANDA_API_KEY: edfe6c5ffd1be4d3bf22f69188ac6bc0fc04c84b" \
-F file=@README.csv http://localhost:8000/data_upload/
```

Upload via AJAX

```
curl -H "PANDA_EMAIL: panda" -H "PANDA_API_KEY: edfe6c5ffd1be4d3bf22f69188ac6bc0fc04c84b" \
--data-binary @test.csv -H "X-Requested-With:XMLHttpRequest" http://localhost:8000/data_upload/?qqfil
```

Note: When using either upload method you may specify the character encoding of the file by passing it as a parameter, e.g. `?encoding=latin1`

6.1.4 Related Uploads

Due to limitations in upload file-handling, it is not possible to create Uploads via the normal API. Instead related files should be uploaded to http://localhost:8000/related_upload/ either as form data or as an AJAX request. Examples of how to upload files with curl are at the end of this section.

Example RelatedUpload object:

```
{
    creation_date: "2012-02-08T23:14:35",
    creator: {
        date_joined: "2011-11-04T00:00:00",
        email: "user@pandaproject.net",
        first_name: "User",
        id: "2",
        is_active: true,
        last_login: "2012-02-08T22:45:28",
        last_name: "",
        resource_uri: "/api/1.0/user/2/"
    },
    dataset: "/api/1.0/dataset/master-4/",
    filename: "PANDA.1.png",
    id: "1",
    original_filename: "PANDA.1.png",
    resource_uri: "/api/1.0/related_upload/1/",
    size: 58990
}
```

Schema

```
http://localhost:8000/api/1.0/related_upload/schema/
```

List

```
http://localhost:8000/api/1.0/related_upload/
```

Fetch

```
http://localhost:8000/api/1.0/related_upload/[id]/
```

Download original file

```
http://localhost:8000/api/1.0/related_upload/[id]/download/
```

Upload as form-data

When accessing PANDA via curl, your email and API key can be specified with the headers `PANDA_EMAIL` and `PANDA_API_KEY`, respectively:

```
curl -H "PANDA_EMAIL: panda" -H "PANDA_API_KEY: edfe6c5ffd1be4d3bf22f69188ac6bc0fc04c84b" \
-F file=@README.csv http://localhost:8000/related_upload/
```

Upload via AJAX

```
curl -H "PANDA_EMAIL: panda" -H "PANDA_API_KEY: edfe6c5ffd1be4d3bf22f69188ac6bc0fc04c84b" \
--data-binary @test.csv -H "X-Requested-With:XMLHttpRequest" http://localhost:8000/related_upload/?q
```

6.1.5 Categories

Categories are identified by slug, rather than by integer id (though they do have one).

Example Category object:

```
{
    dataset_count: 2,
    id: "1",
    name: "Crime",
    resource_uri: "/api/1.0/category/crime/",
    slug: "crime"
}
```

Schema

```
http://localhost:8000/api/1.0/category/schema/
```

List

When queried as a list, a "fake" category named "Uncategorized" will also be returned. This category includes the count of all Datasets not in any other category. It's slug is `uncategorized` is 0, but it can only be accessed as a part of the list.

```
http://localhost:8000/api/1.0/category/
```

Fetch

```
http://localhost:8000/api/1.0/category/[slug]/
```

6.1.6 Datasets

Datasets are identified by slug, rather than by integer id (though they do have one).

Example Dataset object:

```
{
    categories: [ ],
    columns: [
        "id",
        "first_name",
        "last_name",
        "employer"
    ],
    creation_date: "2012-02-08T17:50:11",
    creator: {
        date_joined: "2011-11-04T00:00:00",
        email: "user@pandaproject.net",
        first_name: "User",
        id: "2",
        is_active: true,
        last_login: "2012-02-08T22:45:28",
        last_name: "",
        resource_uri: "/api/1.0/user/2/"
    },
    current_task: {
        creator: "/api/1.0/user/2/",
        end: "2012-02-08T17:50:12",
        id: "1",
        message: "Import complete",
        resource_uri: "/api/1.0/task/1/",
        start: "2012-02-08T17:50:12",
        status: "SUCCESS",
        task_name: "panda.tasks.import.csv",
        traceback: null
    },
    data_uploads: [
        {
            columns: [
                "id",
                "first_name",
                "last_name",
                "employer"
            ],
```

```
            creation_date: "2012-02-08T17:50:09",
            creator: {
                date_joined: "2011-11-04T00:00:00",
                email: "user@pandaproject.net",
                first_name: "User",
                id: "2",
                is_active: true,
                last_login: "2012-02-08T22:45:28",
                last_name: "",
                resource_uri: "/api/1.0/user/2/"
            },
            data_type: "csv",
            dataset: "/api/1.0/dataset/contributors/",
            dialect: {
                delimiter: ",",
                doublequote: false,
                lineterminator: "
                ",
                quotechar: """,
                quoting: 0,
                skipinitialspace: false
            },
            encoding: "utf-8",
            filename: "contributors.csv",
            id: "1",
            imported: true,
            original_filename: "contributors.csv",
            resource_uri: "/api/1.0/data_upload/1/",
            sample_data: [
                [
                    "1",
                    "Brian",
                    "Boyer",
                    "Chicago Tribune"
                ],
                [
                    "2",
                    "Joseph",
                    "Germuska",
                    "Chicago Tribune"
                ],
                [
                    "3",
                    "Ryan",
                    "Pitts",
                    "The Spokesman-Review"
                ],
                [
                    "4",
                    "Christopher",
                    "Groskopf",
                    "PANDA Project"
                ]
            ],
            size: 168
        }
    ],
    description: "",
```

```
    id: "1",
    initial_upload: "/api/1.0/data_upload/1/",
    last_modification: null,
    last_modified: null,
    last_modified_by: null,
    name: "contributors",
    related_uploads: [ ],
    resource_uri: "/api/1.0/dataset/contributors/",
    row_count: 4,
    sample_data: [
        [
            "1",
            "Brian",
            "Boyer",
            "Chicago Tribune"
        ],
        [
            "2",
            "Joseph",
            "Germuska",
            "Chicago Tribune"
        ],
        [
            "3",
            "Ryan",
            "Pitts",
            "The Spokesman-Review"
        ],
        [
            "4",
            "Christopher",
            "Groskopf",
            "PANDA Project"
        ]
    ],
    slug: "contributors"
}
```


Schema

```
http://localhost:8000/api/1.0/dataset/schema/
```

List

```
http://localhost:8000/api/1.0/dataset/
```

List filtered by category

```
http://localhost:8000/api/1.0/dataset/?category=[slug]
```

Search for datasets

The Dataset list endpoint also provides full-text search over datasets' metadata via the `q` parameter.

Note: By default search results are complete Dataset objects, however, it's frequently useful to return simplified objects for rendering lists, etc. To return simplified objects just add `simple=true` to the query.

```
http://localhost:8000/api/1.0/dataset/?q=[query]
```

Fetch

```
http://localhost:8000/api/1.0/dataset/[slug]/
```

Create

To create a new Dataset, `POST` a JSON document containing at least `name` and `data_upload` properties to `/api/1.0/dataset/`. The `data_upload` property may be either an embedded Upload object, or a URI to an existing Upload (for example, `/api/1.0/upload/17/`). Other properties such as `description` may also be set.

Note: The slug field is normally read-only. If you need to create a Dataset with a "well known" slug, you may `PUT` the document to that slug and it will be created.

For example, if I wanted to create a dataset that I knew would be accessible at `/api/1.0/dataset/my-slug/`, I could `PUT` my JSON document to that URL and it would be created. If a document with this slug already exists it will be overwritten!

Import

Begin an import task. Any data previously imported for this dataset will be lost. Returns the original dataset, which will include the id of the new import task:

```
http://localhost:8000/api/1.0/dataset/[id]/import/
```

6.1.7 Data

Data objects are referenced by a unicode `external_id` property, specified at the time they are created. This property must be unique within a given `Dataset`, but does not need to be unique globally. Data objects are accessible at per-dataset endpoints (e.g. `/api/1.0/dataset/[slug]/data/`). There is also a cross-dataset Data search endpoint at `/api/1.0/data`, however, this endpoint can only be used for search–not for create, update, or delete. (See below for more.)

Warning: The `external_id` property of a Data object is the only way it can be accessed through the API. In order to work with Data via the API you must include this property at the time you create it. By default this property is `null` and the Data can not be accessed except via search.

An example `Data` object with an `external_id`:

```
{
    "data": [
        "1",
        "Brian",
```

```
        "Boyer",
        "Chicago Tribune"
    ],
    "dataset": "/api/1.0/dataset/contributors/",
    "external_id": "1",
    "resource_uri": "/api/1.0/dataset/contributors/data/1/"
}
```

An example `Data` object **without** an `external_id`, note that it also has no `resource_uri`:

```
{
    "data": [
        "1",
        "Brian",
        "Boyer",
        "Chicago Tribune"
    ],
    "dataset": "/api/1.0/dataset/contributors/",
    "external_id": null,
    "resource_uri": null
}
```

Schema

There is no schema endpoint for Data.

List

When listing data, PANDA will return a simplified `Dataset` object with an embedded `meta` object and an embedded `objects` array containing `Data` objects. The added Dataset metadata is purely for convenience when building user interfaces.

`http://localhost:8000/api/1.0/dataset/[slug]/data/`

Search

Full-text queries function as "filters" over the normal `Data` list. Therefore, search results will be in the same format as the list results described above:

`http://localhost:8000/api/1.0/dataset/[slug]/data/?q=[query]`

For details on searching Data across all Datasets, see below.

Fetch

To fetch a single `Data` from a given `Dataset`:

`http://localhost:8000/api/1.0/dataset/[slug]/data/[external_id]/`

Create and update

Because Data is stored in Solr (rather than a SQL database), there is no functional difference between Create and Update. In either case any Data with the same `external_id` will be overwritten when the new Data is created. Because of this requests may be either `POST`'ed to the list endpoint or `PUT` to the detail endpoint.

An example POST:

```
{
    "data": [
        "column A value",
        "column B value",
        "column C value"
    ],
    "external_id": "id_value"
}
```

This object would be `POST`'ed to:

```
http://localhost:8000/api/1.0/dataset/[slug]/data/
```

An example `PUT`:

```
{
    "data": [
        "new column A value",
        "new column B value",
        "new column C value"
    ]
}
```

This object would be `PUT` to:

```
http://localhost:8000/api/1.0/dataset/[slug]/data/id_value/
```

Bulk create and update

To create or update objects in bulk you may `PUT` an array of objects to the list endpoint. Any object with a matching `external_id` will be deleted and then new objects will be created. The body of the request should be formatted like:

```
{
    "objects": [
        {
            "data": [
                "column A value",
                "column B value",
                "column C value"
            ],
            "external_id": "1"
        },
        {
            "data": [
                "column A value",
                "column B value",
                "column C value"
            ],
            "external_id": "2"
```

```
        }
    ]
}
```

Delete

To delete an object send a `DELETE` request to its detail url. The body of the request should be empty.

Delete all data from a dataset

In addition to deleting individual objects, its possible to delete all objects within a dataset, by sending a `DELETE` request to the root per-dataset data endpoint. The body of the request should be empty.

`http://localhost:8000/api/1.0/dataset/[slug]/data/`

6.1.8 Global search

Searching all data functions slightly differently than searching within a single dataset. Global search requests go to their own endpoint:

`http:://localhost:8000/api/1.0/data/?q=[query]`

The response is a `meta` object with paging information and an `objects` array containing simplified `Dataset` objects, each of which contains its own `meta` object and an `objects` array containing `Data` objects. **Each Dataset contains a group of matching Data.**

When using this endpoint the `limit` and `offset` parameters refer to the `Datasets` (that is, the **groups**) returned. If you wish to paginate the result sets within each group you can use `group_limit` and `group_offset`, however, this is rarely useful behavior.

CHAPTER

SEVEN

AUTHORS

The PANDA Board:

- Brian Boyer
- Joe Germuska
- Ryan Pitts

Lead Developer:

- Christopher Groskopf

Contributors:

- Dane Springmeyer
- David Eads

CHAPTER

EIGHT

LICENSE

CHAPTER

NINE

INDICES AND TABLES

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