DISQUS

Continuous ~~Deployment~~ **Everything**

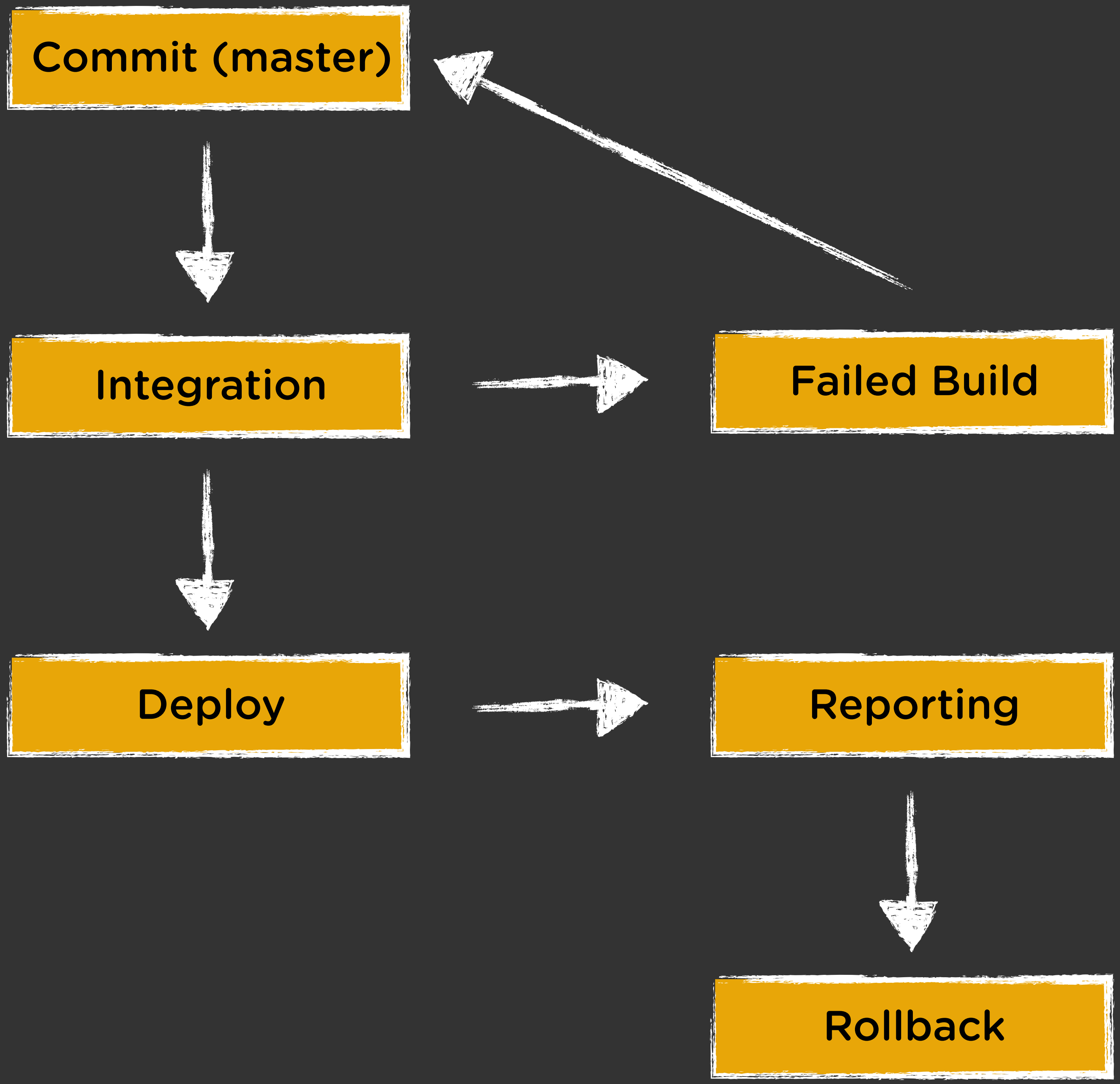
David Cramer
@zeeg

Continuous Deployment

Shipping new code as soon as it's ready

(It's really just super awesome buildbots)

Workflow


Commit (master)
Integration
Failed Build
Deploy
Reporting
Rollback

Pros

- Develop features incrementally
- Release frequently
- Smaller doses of QA

Cons

- **Culture Shock**
- Stability depends on test coverage
- Initial time investment

We mostly just care about iteration and stability

Painless Development

Development

- Production > Staging > CI > Dev
  - **Automate testing** of complicated processes and architecture
- Simple > complete
  - Especially for local development
- `python setup.py {develop,test}`
- Puppet, Chef, simple bootstrap.{py,sh}

Production

- PostgreSQL
- Memcache
- Redis
- Solr
- Apache
- Nginx
- RabbitMQ

Staging

- PostgreSQL
- Memcache
- Redis
- Solr
- Apache
- Nginx
- ~~RabbitMQ~~

CI Server

- Memcache
- PostgreSQL
- Redis
- Solr
- ~~Apache~~
- ~~Nginx~~
- ~~RabbitMQ~~

Macbook

- PostgreSQL
- ~~Apache~~
- ~~Memcache~~
- ~~Redis~~
- ~~Solr~~
- ~~Nginx~~
- ~~RabbitMQ~~

Bootstrapping Local

- Simplify local setup
  - `git clone dcramer@disqus:disqus.git`
  - `./bootstrap.sh`
  - `python manage.py runserver`

- Need to test dependancies?
  - virtualbox + `vagrant up`

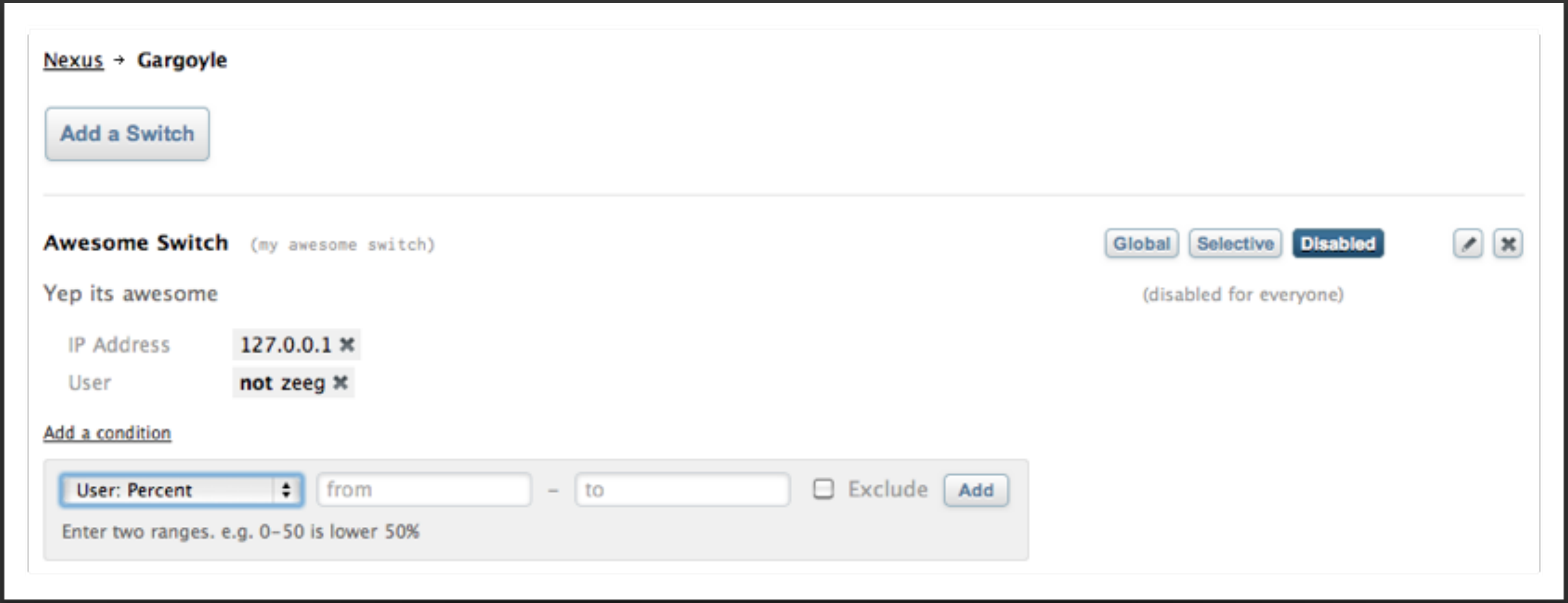
“Under Construction”

- Iterate quickly by hiding features
- **Early adopters** are free QA

```
from gargoyle import gargoyle

def my_view(request):
    if gargoyle.is_active('awesome', request):
        return 'new happy version :D'
    else:
        return 'old sad version :('
```

Gargoyle

Deploy features to portions of a user base at a time to ensure **smooth, measurable releases**



Nexus → **Gargoyle**

Awesome Switch (my awesome switch)

Yep its awesome

(disabled for everyone)

IP Address	127.0.0.1 ✖
User	not zeeg ✖

Add a condition

User: Percent from – to ☐ Exclude Add

Enter two ranges. e.g. 0-50 is lower 50%

Being **users of our product**, we **actively use early versions of features** before public release

Conditions in Gargoyle

```
from gargoyle import gargoyle
from gargoyle.conditions import ModelConditionSet,
                                Percent, String

class UserConditionSet(ModelConditionSet):
    # percent implicitly maps to ``id``
    percent = Percent()
    username = String()

    def can_execute(self, instance):
        return isinstance(instance, User)

# register with our main gargoyle instance
gargoyle.register(UserConditionSet(User))
```

Without Gargoyle

```
SWITCHES = {
    # enable my_feature for 50%
    'my_feature': range(0, 50),
}

def is_active(switch):
    try:
        pct_range = SWITCHES[switch]
    except KeyError:
        return False

    ip_hash = sum([int(x) for x
                   in ip_address.split('.')])

    return (ip_hash % 100 in pct_range)
```

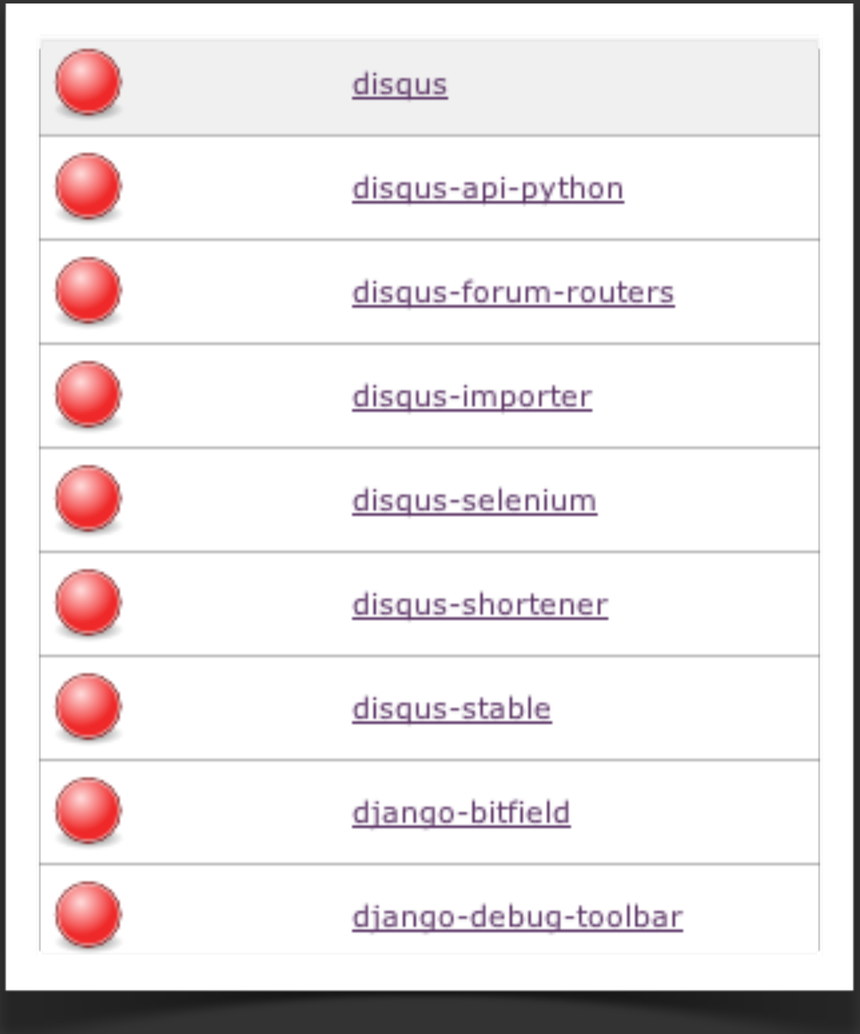
If you use Django, use Gargoyle

Integration

(or as we like to call it)

Jenkins

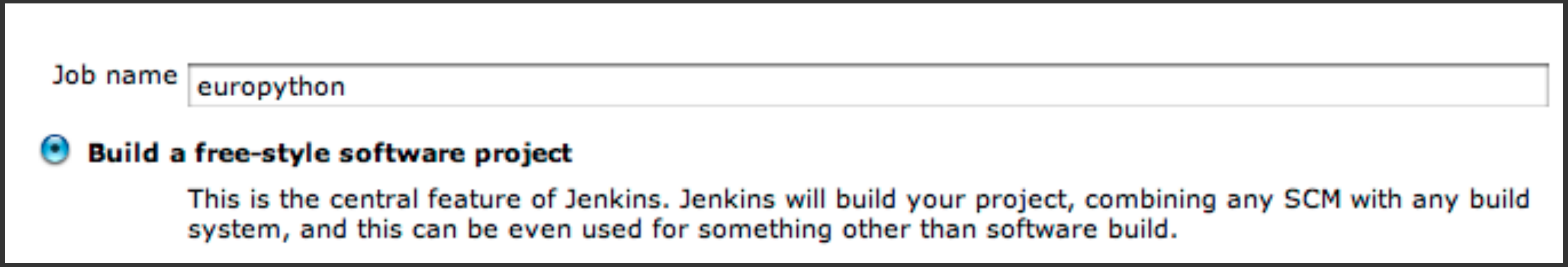
Integration is Required



🔴	disqus
🔴	disqus-api-python
🔴	disqus-forum-routers
🔴	disqus-importer
🔴	disqus-selenium
🔴	disqus-shortener
🔴	disqus-stable
🔴	django-bitfield
🔴	django-debug-toolbar

Deploy only when **things wont break**

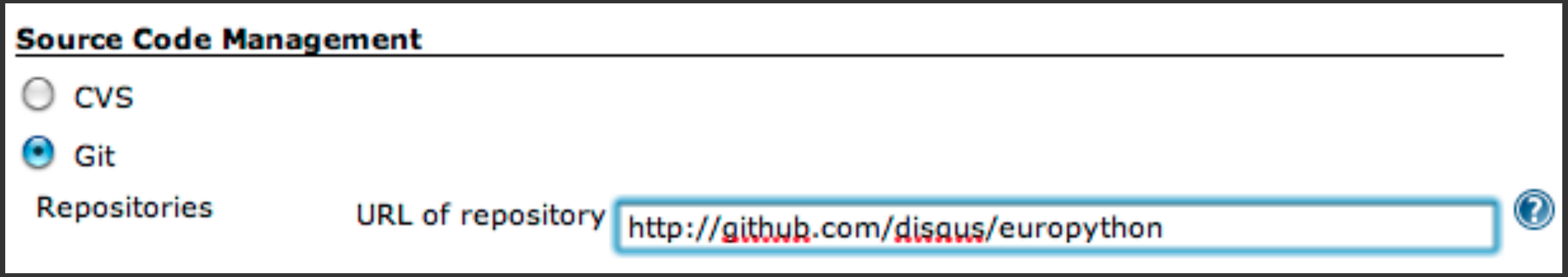
Setup a Jenkins Build



Job name: europython

[✓ Build a free-style software project]

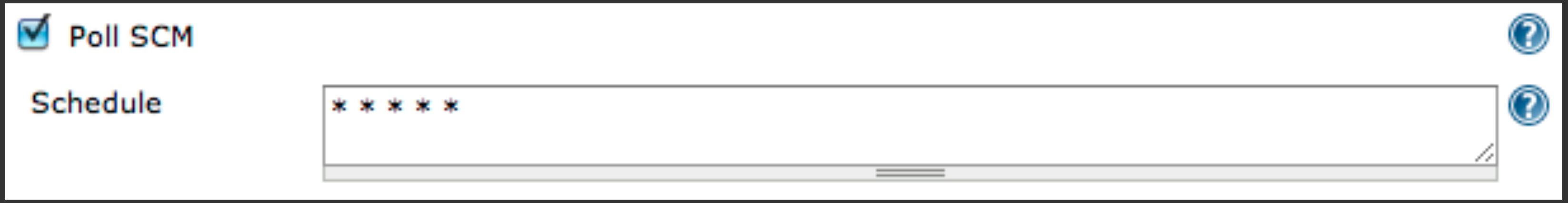
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



Source Code Management

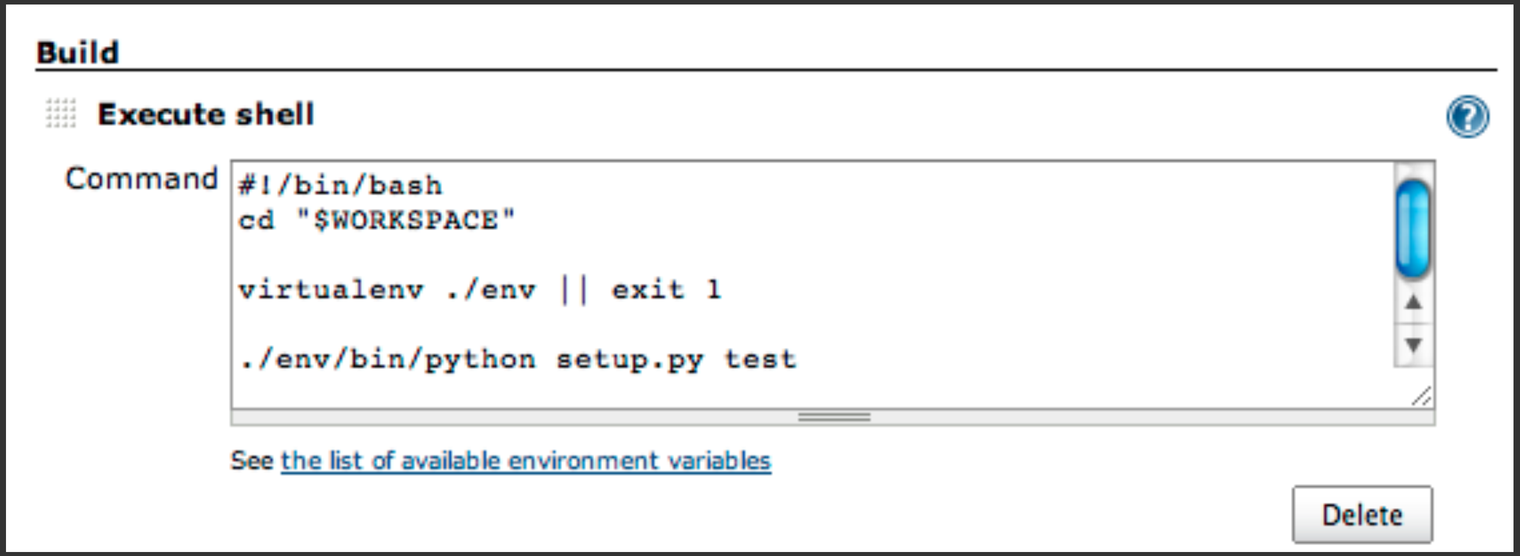
CVS [✓ Git]

Repositories — URL of repository: http://github.com/disqus/europython



[✓] Poll SCM

Schedule: `* * * * *`



Build

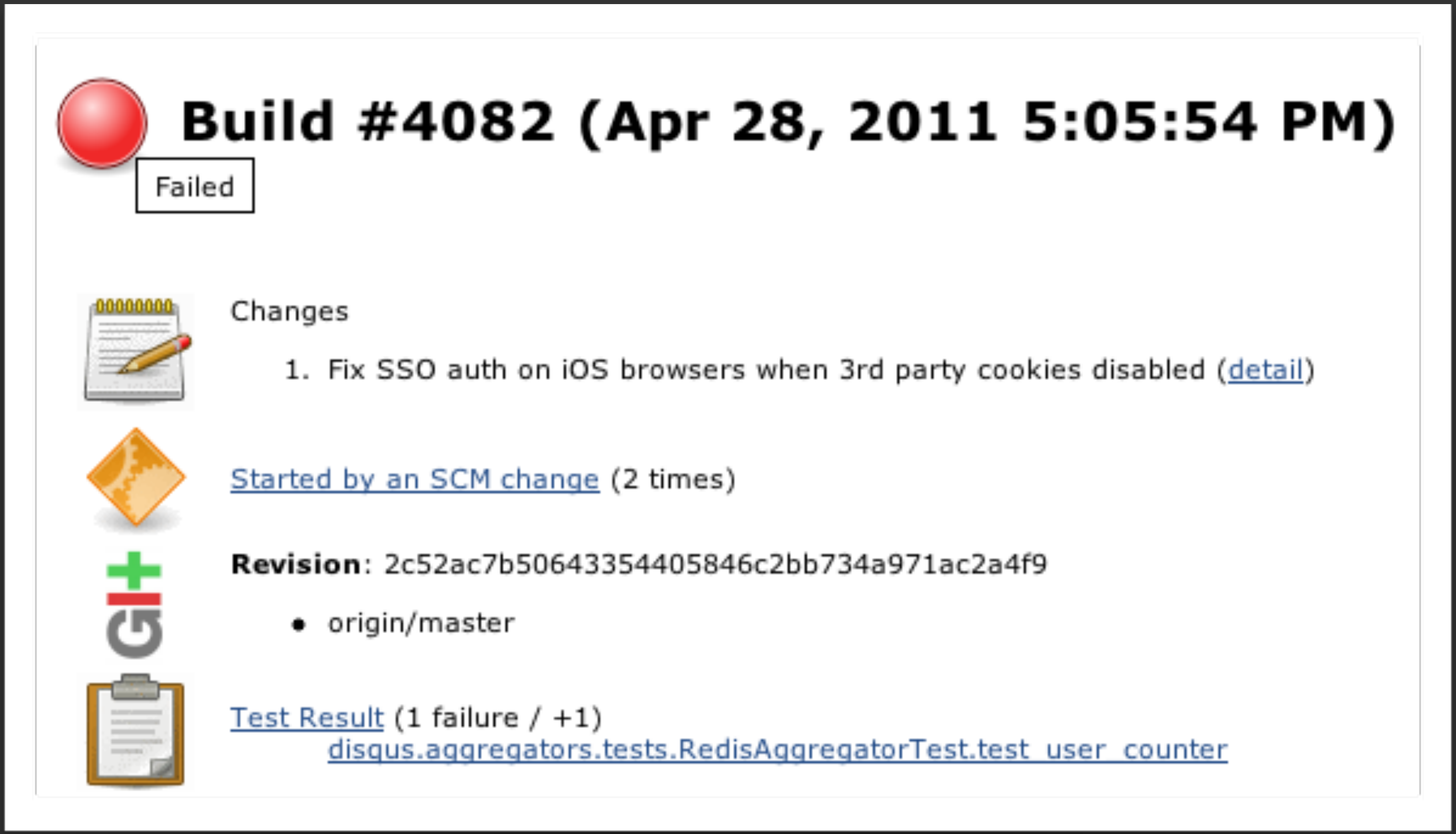
Execute shell

Command:

```
#!/bin/bash
cd "$WORKSPACE"

virtualenv ./env || exit 1

./env/bin/python setup.py test
```

See the list of available environment variables

Reporting is Critical



Build #4082 (Apr 28, 2011 5:05:54 PM)

Failed

Changes

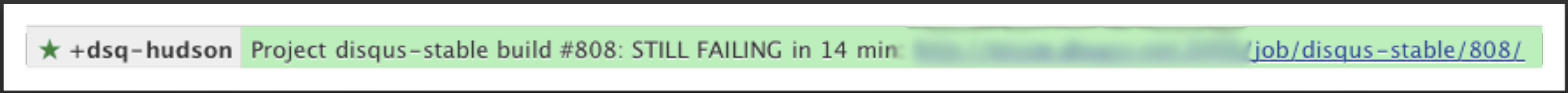
1. Fix SSO auth on iOS browsers when 3rd party cookies disabled (detail)

Started by an SCM change (2 times)

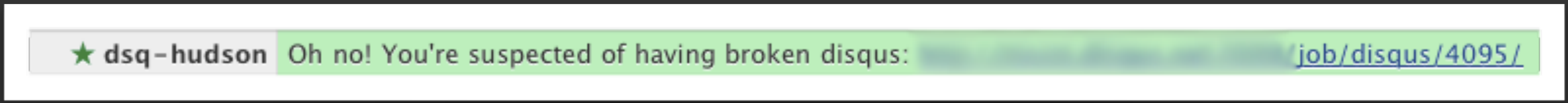
Revision: 2c52ac7b50643354405846c2bb734a971ac2a4f9

- origin/master

Test Result (1 failure / +1)
disqus.aggregators.tests.RedisAggregatorTest.test_user_counter



★ +dsq-hudson Project disqus-stable build #808: STILL FAILING in 14 min: [redacted]/job/disqus-stable/808/



★ dsq-hudson Oh no! You're suspected of having broken disqus: [redacted]/job/disqus/4095/

CI Requirements

- Developers **must** know when they've broken something
  - IRC, Email, IM
- Support proper reporting
  - XUnit, Pylint, Coverage.py
- Painless setup
  - apt-get install jenkins *

https://wiki.jenkins-ci.org/display/JENKINS/Installing+Jenkins+on+Ubuntu

Shortcomings

- False positives lower awareness
  - Reporting isn't accurate
  - Services fail
  - Bad Tests
- Not enough code coverage
  - Regressions on untested code
- Test suite takes too long
  - Integration tests vs Unit tests
  - SOA, distribution

Fixing False Positives

- Re-run tests several times on a failure
- Report continually failing tests
  - Fix continually failing tests
- Rely less on 3rd parties
  - Mock/Dingus

Maintaining Coverage

- Raise awareness with reporting
  - Fail/alert when coverage drops on a build
- Commit tests with code
  - Coverage against commit diff for untested regressions
- Drive it into your culture

Speeding Up Tests

- Write true unit tests
  - vs slower integration tests
- Mock 3rd party APIs
- Distributed and parallel testing
  - http://github.com/disqus/mule

Mule

- **Unstable**, will change a lot
- Mostly Django right now
  - Generic interfaces for **unittest2**
- Works with multi-processing and **Celery**
- Full **XUnit** integration
- **Simple** workflow
  - `mule test --runner="python manage.py mule --worker $TEST"`

Deploy (finally)

How DISQUS Does It

- Incremental deploy with Fabric
- Drop server from pool
- Pull in requirements on each server
    - Isolated virtualenv's built on each server
- Push server back online

How You Can Do It

```
# fabfile.py
from fabric.api import *

def deploy(revision):
    # update sources, virtualenv, requirements
    # ...

    # copy ``current`` to ``previous``
    run('cp -R %(path)s/current %(path)s/previous' % dict(
        path=env.path,
        revision=revision,
    ))

    # symlink ``revision`` to ``current``
    run('ln -fs %(path)s/%(revision)s %(path)s/current' % dict(
        path=env.path,
        revision=revision,
    ))

    # restart apache
    run('touch %(path)s/current/django.wsgi')
```

How YOU Can Do It (cont.)

```
# fabfile.py
from fabric.api import *

def rollback(revision=None):
    # move ``previous`` to ``current``
    run('mv %(path)s/previous %(path)s/current' % dict(
        path=env.path,
        revision=revision,
    ))

    # restart apache
    run('touch %(path)s/current/django.wsgi')
```

Challenges

- PyPi works on server A, but not B
- Scale
- CPU cost per server
- Schema changes, data model changes
- Backwards compatibility

PyPi is Down

- http://github.com/disqus/chishop

Help, we have 100 servers!

- Incremental (ours) vs Fanout
- Push vs Pull
  - Twitter uses BitTorrent
- Isolation vs Packaging (Complexity)

SQL Schema Changes

1. Add column (NULLable)
2. Add app code to **fill column**
3. **Deploy**
4. Backfill column
5. Add app code to **read column**
6. **Deploy**

Updating Caches

- Have a **global version number**
  - `CACHE_PREFIX = 9000`
- Have a data model cache version
  - `sha1(cls.__dict__)`
- Use multiple caches

Reporting

It’s Important!

MERV'S MANURE
SAM'S FANS

<You> Why is mongodb-1 down?

<Ops> It’s down? Must have crashed again

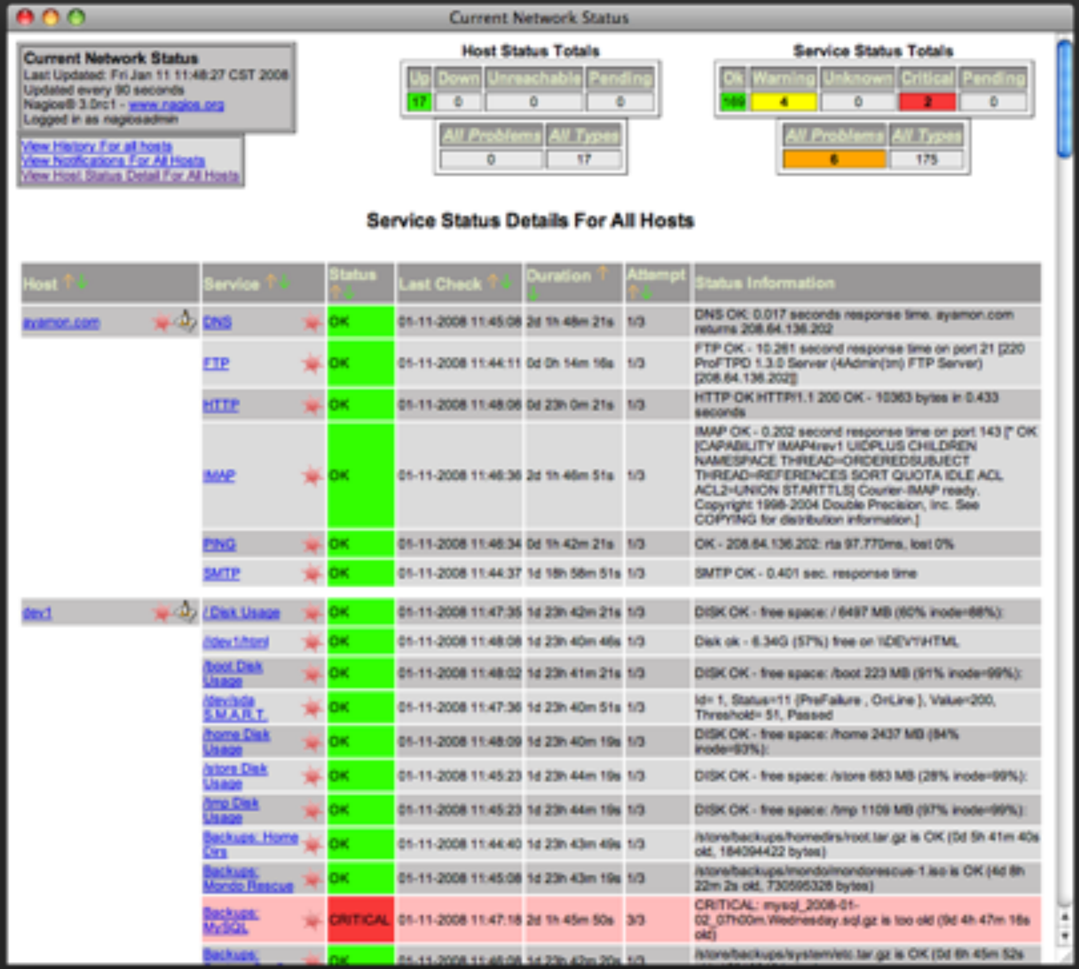
Meaningful Metrics

- Rate of traffic (not just hits!)
  - Business vs system
- Response time (database, web)
- Exceptions
- Social media
  - Twitter

Standard Tools

Graphite

Nagios



Current Network Status

Current Network Status
Last Updated: Fri Jan 11 11:48:27 CST 2008
Updated every 90 seconds
Nagios® 3.0rc1 - www.nagios.org
Logged in as nagiosadmin

View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
17	0	0	0

All Problems	All Types
0	17

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
169	4	0	2	0

All Problems	All Types
6	175

Service Status Details For All Hosts

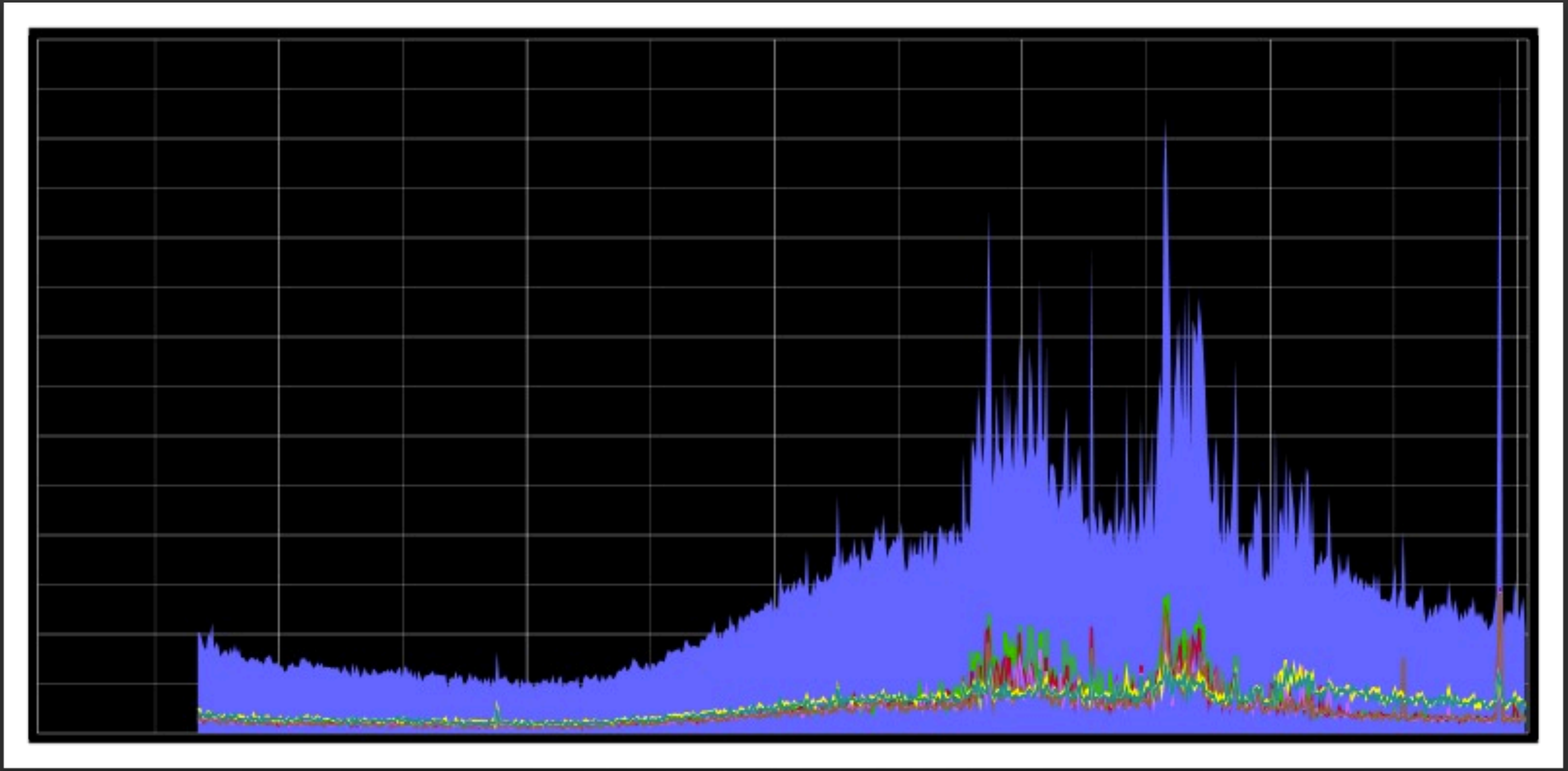
Host	Service	Status	Last Check	Duration	Attempt	Status Information
ayamon.com	DNS	OK	01-11-2008 11:45:08	2d 1h 48m 21s	1/3	DNS OK: 0.017 seconds response time. ayamon.com returns 208.64.136.202
	FTP	OK	01-11-2008 11:44:11	0d 0h 14m 16s	1/3	FTP OK - 10.261 second response time on port 21 [220 ProFTPD 1.3.0 Server (4Admin(tm) FTP Server) [208.64.136.202]]
	HTTP	OK	01-11-2008 11:48:06	0d 23h 0m 21s	1/3	HTTP OK HTTP/1.1 200 OK - 10363 bytes in 0.433 seconds
	IMAP	OK	01-11-2008 11:46:36	2d 1h 46m 51s	1/3	IMAP OK - 0.202 second response time on port 143 [* OK [CAPABILITY IMAP4rev1 UIDPLUS CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT THREAD=REFERENCES SORT QUOTA IDLE ACL ACL2=UNION STARTTLS] Courier-IMAP ready. Copyright 1998-2004 Double Precision, Inc. See COPYING for distribution information.]
	PING	OK	01-11-2008 11:46:34	0d 1h 42m 21s	1/3	OK - 208.64.136.202: rta 97.770ms, lost 0%
	SMTP	OK	01-11-2008 11:44:37	1d 18h 58m 51s	1/3	SMTP OK - 0.401 sec. response time
dev1	/ Disk Usage	OK	01-11-2008 11:47:35	1d 23h 42m 21s	1/3	DISK OK - free space: / 6497 MB (60% inode=68%):
	/dev1/html	OK	01-11-2008 11:48:08	1d 23h 40m 46s	1/3	Disk ok - 6.34G (57%) free on \\DEV1\HTML
	/boot Disk Usage	OK	01-11-2008 11:48:02	1d 23h 41m 21s	1/3	DISK OK - free space: /boot 223 MB (91% inode=99%):
	/dev/sda S.M.A.R.T.	OK	01-11-2008 11:47:36	1d 23h 40m 51s	1/3	Id= 1, Status=11 (PreFailure , OnLine), Value=200, Threshold= 51, Passed
	/home Disk Usage	OK	01-11-2008 11:48:09	1d 23h 40m 19s	1/3	DISK OK - free space: /home 2437 MB (84% inode=93%):
	/store Disk Usage	OK	01-11-2008 11:45:23	1d 23h 44m 19s	1/3	DISK OK - free space: /store 683 MB (28% inode=99%):
	/tmp Disk Usage	OK	01-11-2008 11:45:23	1d 23h 44m 19s	1/3	DISK OK - free space: /tmp 1109 MB (97% inode=99%):
	Backups: Home Dirs	OK	01-11-2008 11:44:40	1d 23h 43m 49s	1/3	/store/backups/homedirs/root.tar.gz is OK (0d 5h 41m 40s old, 184094422 bytes)
	Backups: Mondo Rescue	OK	01-11-2008 11:45:08	1d 23h 43m 19s	1/3	/store/backups/mondo/mondorescue-1.iso is OK (4d 8h 22m 2s old, 730595328 bytes)
	Backups: MySQL	CRITICAL	01-11-2008 11:47:18	2d 1h 45m 50s	3/3	CRITICAL: mysql_2008-01-02_07h00m.Wednesday.sql.gz is too old (9d 4h 47m 16s old)
	Backups:	OK	01-11-2008 11:48:08	1d 23h 42m 20s	1/3	/store/backups/system/etc.tar.gz is OK (0d 6h 45m 52s

Wednesday, June 22, 2011

Using Graphite

```
# statsd.py
# requires python-statsd

from pystatsd import Client
import socket

def with_suffix(key):
    hostname = socket.gethostname().split('.')[0]
    return '%s.%s' % (key, hostname)

client = Client(host=STATSD_HOST, port=STATSD_PORT)

# statsd.incr('key1', 'key2')
def incr(*keys):
    keys = [with_suffix(k) for k in keys]:
    client.increment(*keys):
```

Using Graphite (cont.)



(Traffic across a cluster of servers)

Logging

- Realtime
- Aggregates
- History
- Notifications
- Scalable
- Available
- Metadata

Logging: Syslog

- ✓ Realtime
- x Aggregates
- ✓ History
- x Notifications
- ✓ Scalable
- ✓ Available
- x Metadata

Logging: Email Collection

- ✓ Realtime
- x Aggregates
- ✓ History
- x Notifications
- x Scalable
- ✓ Available
- ✓ Metadata

(Django provides this out of the box)

Logging: Sentry

- ✓ Realtime
- ✓ Aggregates
- ✓ History
- ✓ Notifications
- ✓ Scalable
- ✓ Available
- ✓ Metadata

http://github.com/dcramer/django-sentry

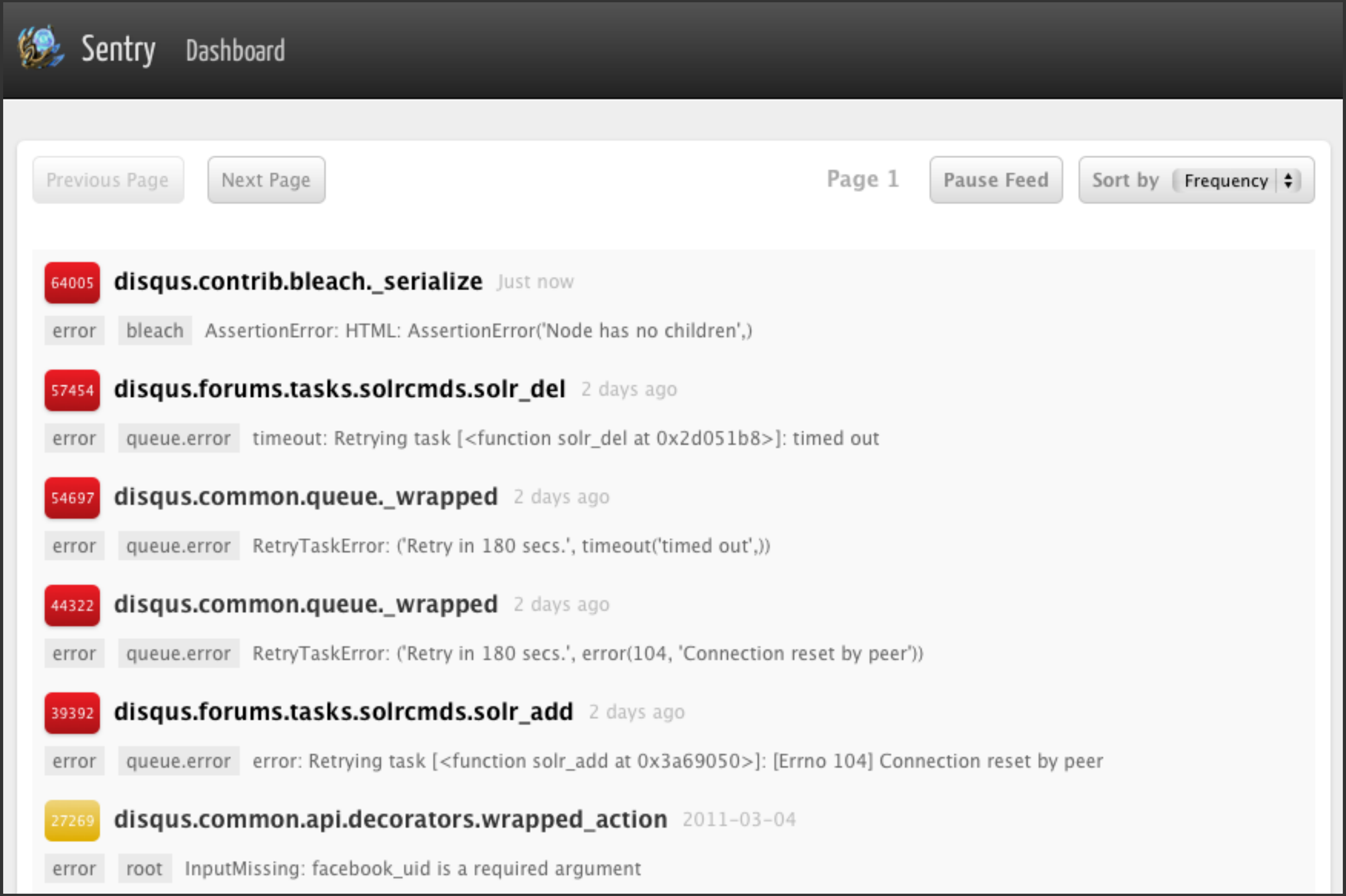
Setting up Sentry (1.x)

```
# setup your server first
$ pip install django-sentry
$ sentry start

# configure your Python (Django in our case) client
INSTALLED_APPS = (
    # ...
    'sentry.client',
)

# point the client to the servers
SENTRY_REMOTE_URL = ['http://sentry/store/']

# visit http://sentry in the browser
```

Setting up Sentry (cont.)

```
# ~/.sentry/sentry.conf.py

# use a better database
DATABASES = {
    'default': {
        'ENGINE': 'postgresql_psycopg2',
        'NAME': 'sentry',
        'USER': 'postgres',
    }
}

# bind to all interfaces
SENTRY_WEB_HOST = '0.0.0.0'

# change data paths
SENTRY_WEB_LOG_FILE = '/var/log/sentry.log'
SENTRY_WEB_PID_FILE = '/var/run/sentry.pid'
```

Sentry (demo time)



64005 **disqus.contrib.bleach._serialize** Just now
error bleach AssertionError: HTML: AssertionError('Node has no children',)

57454 **disqus.forums.tasks.solrcmds.solr_del** 2 days ago
error queue.error timeout: Retrying task [<function solr_del at 0x2d051b8>]: timed out

54697 **disqus.common.queue._wrapped** 2 days ago
error queue.error RetryTaskError: ('Retry in 180 secs.', timeout('timed out',))

44322 **disqus.common.queue._wrapped** 2 days ago
error queue.error RetryTaskError: ('Retry in 180 secs.', error(104, 'Connection reset by peer'))

39392 **disqus.forums.tasks.solrcmds.solr_add** 2 days ago
error queue.error error: Retrying task [<function solr_add at 0x3a69050>]: [Errno 104] Connection reset by peer

27269 **disqus.common.api.decorators.wrapped_action** 2011-03-04
error root InputMissing: facebook_uid is a required argument

Wrap Up

Getting Started

- Package your app
- Ease deployment; fast rollbacks
- Setup automated tests
- Gather some easy metrics

Going Further

- Build an immune system
  - Automate deploys, rollbacks (maybe)
- Adjust to your culture
  - CD doesn’t “just work”
- SOA == great success

DISQUS

Questions?

psst, we’re hiring
jobs@disqus.com

References

- **Gargoyle** (feature switches)
  https://github.com/disqus/gargoyle
- **Sentry** (log aggregation)
  https://github.com/dcramer/django-sentry (1.x)
  https://github.com/dcramer/sentry (2.x)
- **Jenkins CI**
  http://jenkins-ci.org/
- **Mule** (distributed test runner)
  https://github.com/disqus/mule

code.disqus.com