About Me

• Chief Architect at Cloudkick
• Developer on Apache HTTP Server
• Former VP Infrastructure @ ASF
• Libcloud developer!
About the Cloud
About the Cloud

• Awesome.
About the Cloud

- Awesome.
- Really.
About the Cloud

- Awesome.
- Really.
- Maybe not always.
About the Cloud

• Awesome.
• Really.
• Maybe not always.
• But mostly.
Services (SaaS)

- GMail, Google Docs, etc
- Most any website cloud be called SaaS.
- Cloudkick
Platforms (PaaS)

- Google AppEngine
- Heroku (Rails as PaaS)
- SalesForce.com / VMForce
Storage

- Amazon S3
- Rackspace CloudFiles
- Google Storage for Developers
Compute

- Amazon EC2
- Rackspace Cloud
- Linode
- GoGrid
- Voxel
- And many more!
I want a server.
I want a server: right now.
from libcloud.types import Provider
from libcloud.providers import get_driver
rs = get_driver(Provider.RACKSPACE)('rackspace-apikey')
rs.create_node('serverA')
About Libcloud

• Started in the summer of 2009
• Easy to use.
• Portable.
• Pure Python (proposed ports to others)
  • Socket & HTTP interfaces exist today!
Why?

- API Styles:
  - Amazon: XML + Custom HMAC Auth
  - Rackspace: JSON + Auth Tickets
  - SoftLayer: XML RPC + User / Password
Libcloud Today

- In the Apache Incubator
- 15 Providers:
  - Dreamhost, Amazon EC2, Enomaly ECP, Eucalyptus, GoGrid, IBM Developer Cloud, Linode, OpenNebula, Slicehost, SoftLayer, Rackspace, RimuHosting, Terramark, VMWare vCloud, Voxel, VPS.net
Libcloud APIs

- Originally 6 Core APIs
  - List Nodes
  - List Images
  - List Sizes
  - Create / Destroy / Reboot Node
list_nodes

foo = d.list_nodes()
for node in foo:
    print node.id
    print node.public_ip
list_images

images = d.list_images()

ubuntu = [i for i in images if i.name.find('Ubuntu') != -1]

print ubuntu[0].id
print ubuntu[0].name
list_sizes

sizes = d.list_sizes()
print sizes[0].id
print sizes[0].ram
print sizes[0].disk
print sizes[0].price
Create Node

images = d.list_images()
sizes = d.list_sizes()
print d.create_node(name="test22", image=images[0], size=sizes[0])
Reboot/Destroy

d.reboot(nodeA)

d.destroy(nodeB)
Locations!

loc = d.list_locations()
print loc[0].name
print loc[0].country
Extended APIs

- Providers inconsistent about services.
- Have a “ex_” prefix, documented per-driver.
- Amazon Security Groups:
  - amz.create_node('foo', ex_securitygroup='groupA')
Getting Started

- easy_install apache-libcloud
Get your Provider Info

• Amazon:

• Rackspace:
  • https://manage.rackspacecloud.com/APIAccess.do
List your Machines

from libcloud.types import Provider
from libcloud.providers import get_driver

d = get_driver(Provider.RACKSPACE)("xxxxxxx")

nodes = d.list_nodes()
for node in nodes:
    print "id: %s  name: %s  public_ips: %s" % (node.id, node.name, node.public_ip)
Cheapest 4 gig node outside the US

```python
possible = []
for d in drivers:
    loc = filter(lambda x: x.country != 'US',
                 d.list_locations())
    for l in loc:
        sizes = filter(lambda x: x.ram >= 4096, d.list_sizes(l))
        for s in sizes:
            possible.append({'size': s,
                             'location': l,
                             'driver': d})

best = sorted(possible, lambda x,y:
               x['size'].price < y['size'].price)[0]
print best
```
Integrating with Fabric

```python
env.hosts = [x.public_ip[0] for x in d.list_nodes()]

def hostname():
    run('hostname')
```
$ fab hostname
[173.45.245.33] run: hostname
[173.45.245.33] out: lctest3.k1k.me
[173.45.245.32] run: hostname
[173.45.245.32] out: lctest2.k1k.me

Done.
Disconnecting from 173.45.245.33... done.
Disconnecting from 173.45.245.32... done.
One more thing!

- deploy_node
  - Calls create_node
  - Consistent initial bootstrapping of machines.
- SSH Keys
- Configuration Management
Installing Puppet

```python
skey = SSHKeyDeployment(key)
sd = ScriptDeployment("apt-get install -y puppet")
msd = MultiStepDeployment([skey, sd])
node = d.deploy_node(name="lc-test", deploy=msd)
```
Up next for Libcloud
Image Formats

• Hazy world between Operating System, Configuration Management and the Sysadmin.

• People stick with Config Management, because dealing with base Images is painful today
Existing standards

- Amazon AMI
- VMWare Open Virtualization Format (OVF)
Hosting Provider Side

• Technical challenges
  • Most commercial hosting is Xen based
  • Most hosting companies aren’t giant tech companies
User Side

- Building Images is complicated.
- Versioning Sucks
- Time sink uploading
Proposed Image Format

- Based on Cloudlets Project
- JSON Metadata in single file
- Filesystem in a tarball
  - Versioned in DVCS
- Includes building server-side support infrastructure for Hosting providers!
- More details on mailing list
Multiple Languages

- Hundreds of emails exchanged on the mailing list.
- Interest, something will happen.
- If interested, join the lists, start hacking on code!
Contributing!

- Open Community just as important as open code -- everything on list or IRC.
- Hosting providers: Get your driver in!
- Hackers: Make cool tools!
- Sysadmins: Manage your infrastructure.
Related Projects

• JClouds
• Java
• Apache Deltacloud
  • Ruby, started by Redhat, just joined Apache Incubator in May
• Fog
  • Ruby
Questions?

- Apache Libcloud:
  

  #libcloud on Freenode IRC

- Slides online:

  [http://paul.querna.org/slides](http://paul.querna.org/slides)