Amazon Web Services Workshop

Block 1: 10:45 - 11:10

Overview

BIOTEAM
Enabling Science

Who am I?

- I'm from the BioTeam
  - Independent consulting shop
  - Staffed by scientists forced to learn IT to get our own research done
- Found a fun business niche
  - Bridging the "gap" between science, IT & high performance computing
- This matters today because …
  - We've been doing production informatics work on Amazon AWS since 2007
  - Can speak from multiple AWS perspectives (Customer, Developer, Integrator)

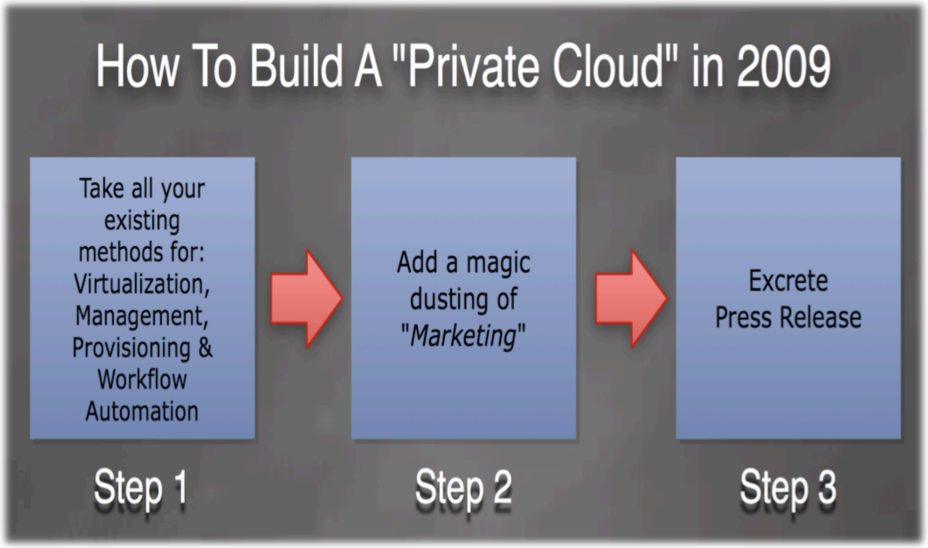
Scene from ancient history in a cloud-enabled world…

ISMB 2010 Cloud Workshop

Warning: we usually take 2 full days to talk about this stuff

- This will be a lightweight talk
  - Time constraints mean we can't do much of a deep dive into any single topic area
  - Taking a guess at the topics most of interest to this audience
  - Please stop me or ask questions at any time

- **We will not be talking about:**
  - Private Clouds
    - Why? They are just silly.



How To Build A "Private Cloud" in 2009

Step 1: Take all your existing methods for: Virtualization, Management, Provisioning & Workflow Automation

Step 2: Add a magic dusting of "*Marketing*"

Step 3: Excrete Press Release

*** I have seen my '09 graphic come to life in the real world. Psychic.*

Topics I do plan to talk about

- Infrastructure as a service (IaaS)
- Why cloud? Why Amazon?
- Mapping high performance computing (HPC) workflows onto cloud platforms
- Strategies, Reference Architectures & Best Practices
- Technical challenges & workarounds
- Horrible mistakes I've made and valuable lessons learned

Note: I am somewhat infamous for talking very fast and using huge slide decks. Some slides are included mainly as "reference" material for post-workshop slide handouts. Please feel free to interrupt & ask questions.

The “C” Word

- Impossible to avoid
- The term ‘cloud computing’ is effectively meaningless today – too many marketers have fuzzed and co-opted the term

- *I prefer “utility computing” myself*

- Now we have to define the term whenever we use it in public

Defining our Terms

- Gartner:
  - *"Cloud computing is a style of computing where scalable and elastic IT-enabled capabilities are delivered as a service to external customers using Internet technologies."*
- Jinesh Varia on AWS:
  - *"… Amazon Web Services (AWS) cloud provides a highly reliable and scalable infrastructure for deploying web-scale solutions, **with minimal support and administration costs, and more flexibility than you've come to expect from your own infrastructure**, either on-premise or at a datacenter facility."*

Today's Cloud Meta-Topics

- Laziness
- Beauty
- Money

The real reason I visit Amazon in Seattle, shhhhhh
(mocha made by Kelli @ Inner Chapters Café & Bookstore)

Laziness

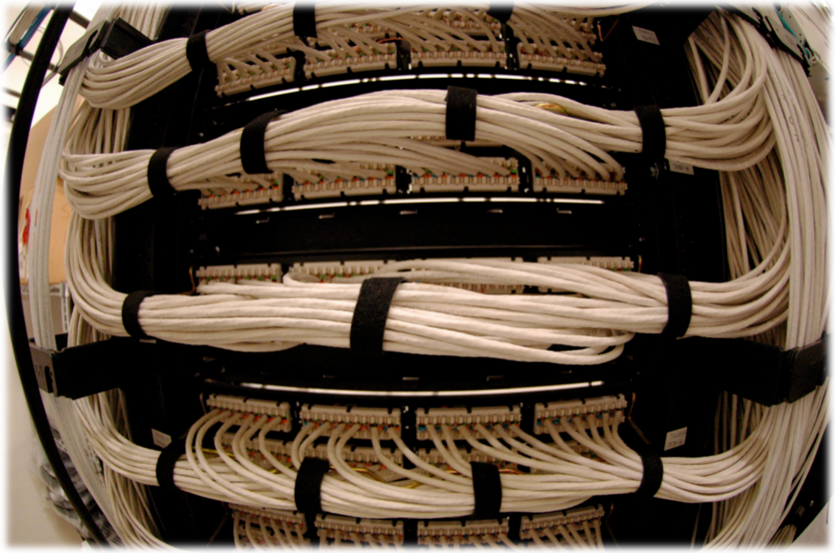
ISMB 2010 Cloud Workshop

- Larry Wall's 1st Great Virtue:
  - *"… the quality that makes you go to great effort* ***to reduce overall energy expenditure. It makes you write labor-saving programs that other people will find useful, and document what you wrote so you don't have to answer so many questions about it"***
- Scriptable IT Infrastructures are the latest boon for the perennially lazy (like myself)

Note subtle Amazon product plug above …

Beauty

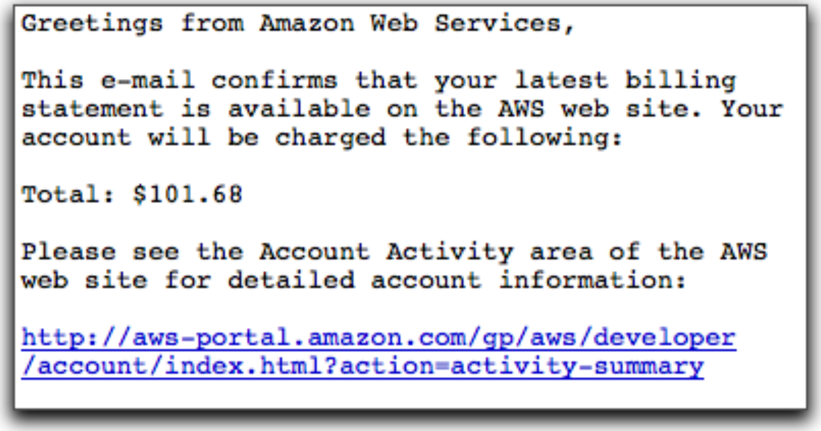
- Call me a nerd but the cloud gives us amazing new abilities:
  1. “Scriptable datacenters”
  2. Orchestrating complex systems & workflows with a few lines of code
  3. Infrastructure managed like it was source code



Can you believe that the Broad Institute @ MIT let me into their telco closets & machine rooms?

Money

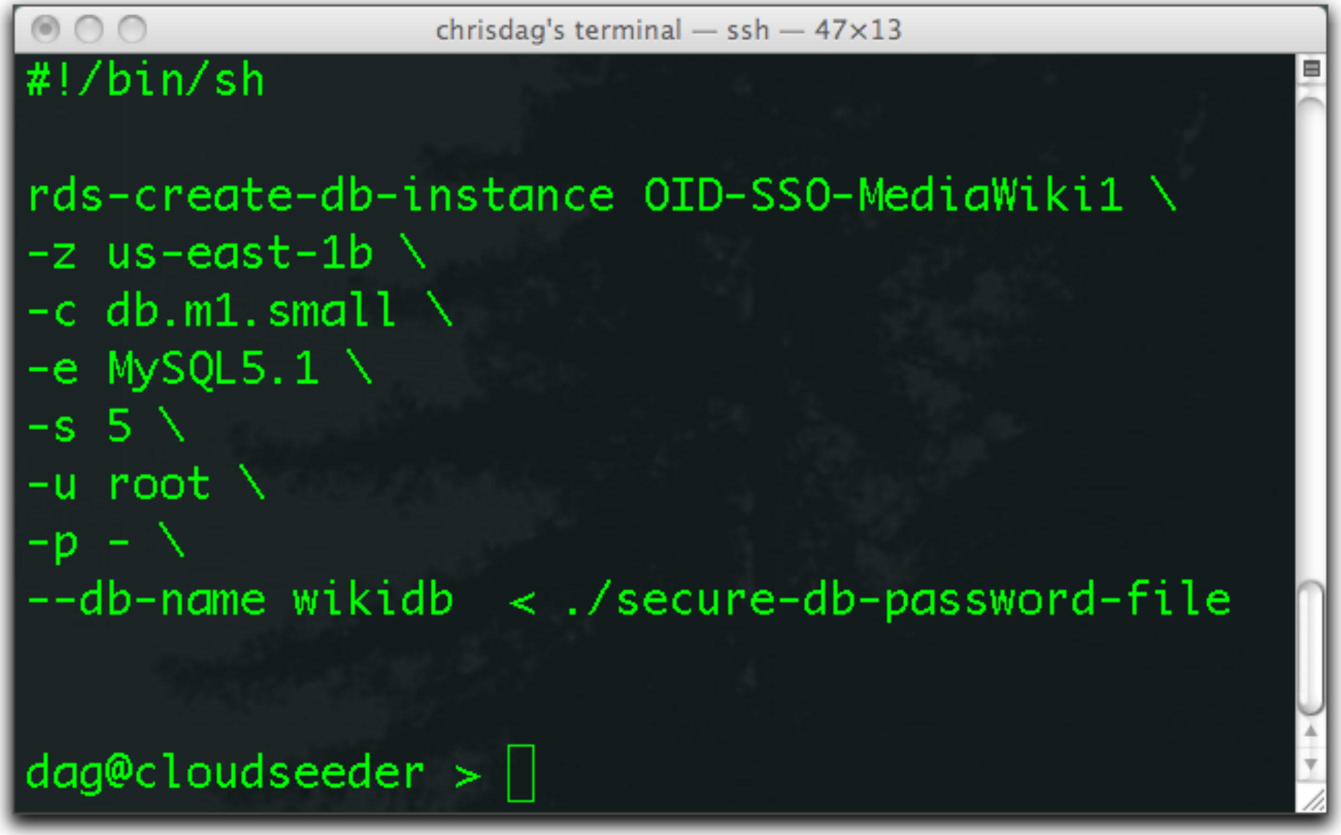
- If a cloud talk occurs without mention of $
  - … did it actually happen?



```
Greetings from Amazon Web Services,

This e-mail confirms that your latest billing
statement is available on the AWS web site. Your
account will be charged the following:

Total: $101.68

Please see the Account Activity area of the AWS
web site for detailed account information:

http://aws-portal.amazon.com/gp/aws/developer
/account/index.html?action=activity-summary
```

Laziness

"Scriptable Infrastructure" is a BIG DEAL



```
#!/bin/sh

rds-create-db-instance OID-SSO-MediaWiki1 \
-z us-east-1b \
-c db.m1.small \
-e MySQL5.1 \
-s 5 \
-u root \
-p - \
--db-name wikidb  < ./secure-db-password-file


dag@cloudseeder > 
```

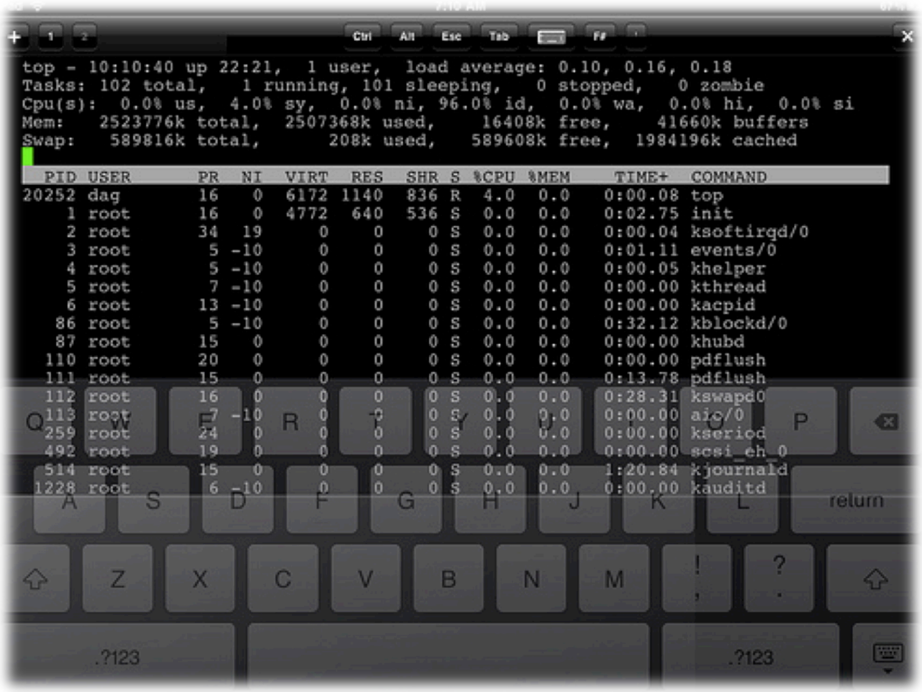
This single command will start a 5GB managed MySQL database in the Amazon cloud for $0.11/hour. The database is ***automatically*** patched, managed and backed up and can (optionally) be clustered for high availability across multiple datacenters. ***Can you do that in your datacenter today?***

It’s ALL scriptable …

- Servers
- Storage
- Operating System(s)
- Network
- Provisioning
- Management
- Monitoring & Scaling
- Accounting

Not hype. Real.

- Every facet of our IT infrastructure can now be automated and remotely controlled via simple scripts and API calls
- Benefits go way above and beyond simple IT Operations work, server "lights out management" features and what local VM systems provide
- *Nirvana for lazy nerds like myself*
  - *Concentrate on **getting SCIENCE done**, not babysitting datacenter racks*



```
top - 10:10:40 up 22:21,  1 user,  load average: 0.10, 0.16, 0.18
Tasks: 102 total,   1 running, 101 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0% us,  4.0% sy,  0.0% ni, 96.0% id,  0.0% wa,  0.0% hi,  0.0% si
Mem:   2523776k total,  2507368k used,    16408k free,    41660k buffers
Swap:   589816k total,      208k used,   589608k free,  1984196k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
20252 dag       16   0  6172 1140  836 R  4.0  0.0   0:00.08 top
    1 root      16   0  4772  640  536 S  0.0  0.0   0:02.75 init
    2 root      34  19     0    0    0 S  0.0  0.0   0:00.04 ksoftirqd/0
    3 root       5 -10     0    0    0 S  0.0  0.0   0:01.11 events/0
    4 root       5 -10     0    0    0 S  0.0  0.0   0:00.05 khelper
    5 root       7 -10     0    0    0 S  0.0  0.0   0:00.00 kthread
    6 root      13 -10     0    0    0 S  0.0  0.0   0:00.00 kacpid
   86 root       5 -10     0    0    0 S  0.0  0.0   0:32.12 kblockd/0
   87 root      15   0     0    0    0 S  0.0  0.0   0:00.00 khubd
  110 root      20   0     0    0    0 S  0.0  0.0   0:00.00 pdflush
  111 root      15   0     0    0    0 S  0.0  0.0   0:13.78 pdflush
  112 root      16   0     0    0    0 S  0.0  0.0   0:28.31 kswapd0
  113 root       7 -10     0    0    0 S  0.0  0.0   0:00.00 aio/0
  259 root      24   0     0    0    0 S  0.0  0.0   0:00.00 kseriod
  492 root      19   0     0    0    0 S  0.0  0.0   0:00.00 scsi_eh_0
  514 root      15   0     0    0    0 S  0.0  0.0   1:20.84 kjournald
 1228 root       6 -10     0    0    0 S  0.0  0.0   0:00.00 kauditd
```

So easy an iPad can control it.

Scriptable Infrastructure

- For the first time significant parts of our Research IT infrastructure might be 100% virtual and entirely controllable via scripts and APIs
- It's not rocket science
- Anyone can drive this stuff
    - *Especially motivated researchers*

Beauty

Beauty

ISMB 2010 Cloud Workshop

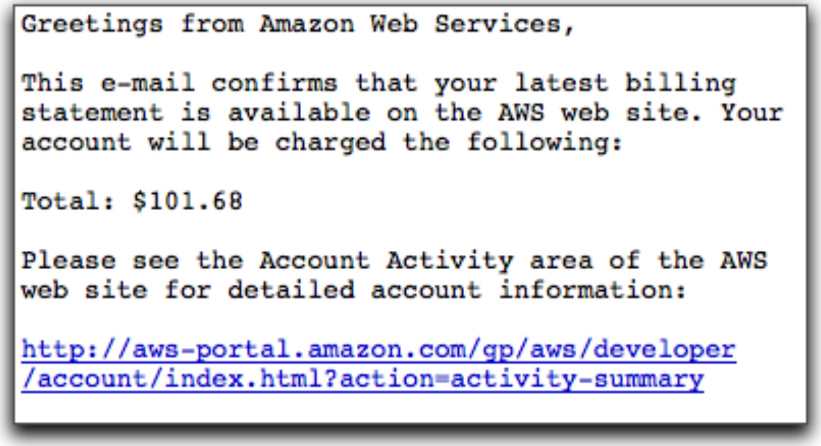
- "Scriptable Infrastructure" is just the baseline
  - The cool stuff happens when we build on top of these capabilities
- AWS enables us to ***orchestrate*** *vast arrays of complex systems, pipelines, workflows & applications*
  - *Without leaving the hammock*
- *Orchestrated systems working in concert are a beautiful thing.*

Money

Money

For anyone seriously looking at IaaS Cloud Platforms

- Can't escape it
- Critical to have a solid understanding of the financial issues


Greetings from Amazon Web Services,

This e-mail confirms that your latest billing statement is available on the AWS web site. Your account will be charged the following:

Total: $101.68

Please see the Account Activity area of the AWS web site for detailed account information:

http://aws-portal.amazon.com/gp/aws/developer/account/index.html?action=activity-summary

ISMB 2010 Cloud Workshop

Money

- IaaS is all about leveraging economies of scale
  - ▫ Providers can charge far less than your own true internal costs while still making a profit
- **If you don't honestly assess and track the true cost of providing IT services internally, you won't have the data you need to make informed decisions**
  - ▫ ... also more exposed to manipulation by people with personal agendas to push

Money: Cost-of-service Accounting

Amylin Pharmaceuticals Example:

- Hired a consultant specializing in IT cost guidance
- Huge effort to construct a spreadsheet that tracked the real cost of delivering each IT service
  - Result: When compared against actual budget, the spreadsheet was accurate to within $2K out of a 20M budget
- Incredible benefits from this data
  - Realized it cost $2M/year to run HR internally
  - Many IT staff simply "keeping lights running" and not driving business or scientific success
  - Information shared widely, senior managers really did not like being associated with the most expensive services
  - … lead to organizational changes & operational methods that deliver huge recurring savings

ISMB 2010 Cloud Workshop

Why Amazon?

Amazon Web Services

- Infrastructure as a Service (IaaS) platfom
- Virtualized compute, storage, databases, message passing and other foundational building blocks
- Available via the internet
- Accessible/controllable via software APIs
- Metered usage, pay as you go
- AWS infrastructure is insanely huge and distributed worldwide

ISMB 2010 Cloud Workshop

My $.02 - What AWS got right

- Pricing
- Very broad product & service offerings
- Ease of use
- Very large scale
- Significant expertise in operating at-scale in hostile networking environments

AWS Today

- Leading IaaS provider
- Multi-year head start on competitors
- Hard to compete with current AWS scale
- Very hard to compete with rate of AWS product roll-outs and enhancements
    - Significant announcements often several times per month

AWS Competition

- Google, Microsoft and (perhaps) RackSpace
- Other competitors are just fooling themselves
  - My personal (biased) belief about IaaS providers:
    - If you operate less than ~200,000 CPU cores I don’t believe that you have solved all of the technical and engineering issues
    - If you are smaller than Amazon, Microsoft or Google than I don’t believe you can operate at a scale large enough to pass significant $ savings down to me

Why BioTeam uses AWS

- Broad product stack provides **ALL** of the tools we need to build solutions and serve our clients
- **However**:
  - ▫ We are quite mercenary in choosing our IT solutions – we are platform/technology/vendor agnostic
  - ▫ If a better solution exists we will switch

AWS Rate of Change Examples

- Dec 2009
  - **Amazon VPC launch**
  - **AWS Spot Instance launch**
  - Windows Server 2008, SQL Server 2008 support
  - **AWS Import/Export launch**
  - US-West AWS region launch
- Feb 2010
  - SimpleDB consistency enhancements
  - Reserved Instances (Windows)
  - **m2.xlarge EC2 instance type**
  - **AWS Consolidated Billing**
  - S3 Object Versioning

The AWS Blog is a great resource: http://aws.typepad.com/aws/

AWS Rate of Change Examples

- March 2010
  - **S3 Import/Export**
    - **Raw drive support**
  - **S3 Versioning**
  - Combined bandwidth pricing
  - Reverse DNS for elastic IPs

- April 2010
  - SNS Service beta
  - RDS Europe launch
  - Singapore AWS Region w/ 2 availability zones launched

The AWS Blog is a great resource: http://aws.typepad.com/aws/

AWS Rate of Change Examples

- May 2010
  - **RDS Multi-AZ Deployment**
  - **S3 Reduced Redundancy Storage (RRS) launch**
  - **RDS support in AWS Console**

- June 2010
  - Elastic Map Reduce Updates
  - **S3 Import/Export API**
  - CloudFront HTTPS support
  - **S3 support in AWS Console**
  - CloudWatch metrics for EBS volumes

The AWS Blog is a great resource: http://aws.typepad.com/aws/

ISMB 2010 Cloud Workshop

AWS Product Stack

Putting the pieces in context

AWS Product Stack

Remember

- Amazon is providing us with low-level infrastructure services (“the plumbing”)
- We take these low-level primitives and assemble/ orchestrate them into what we desire

Amazon Web Services Products

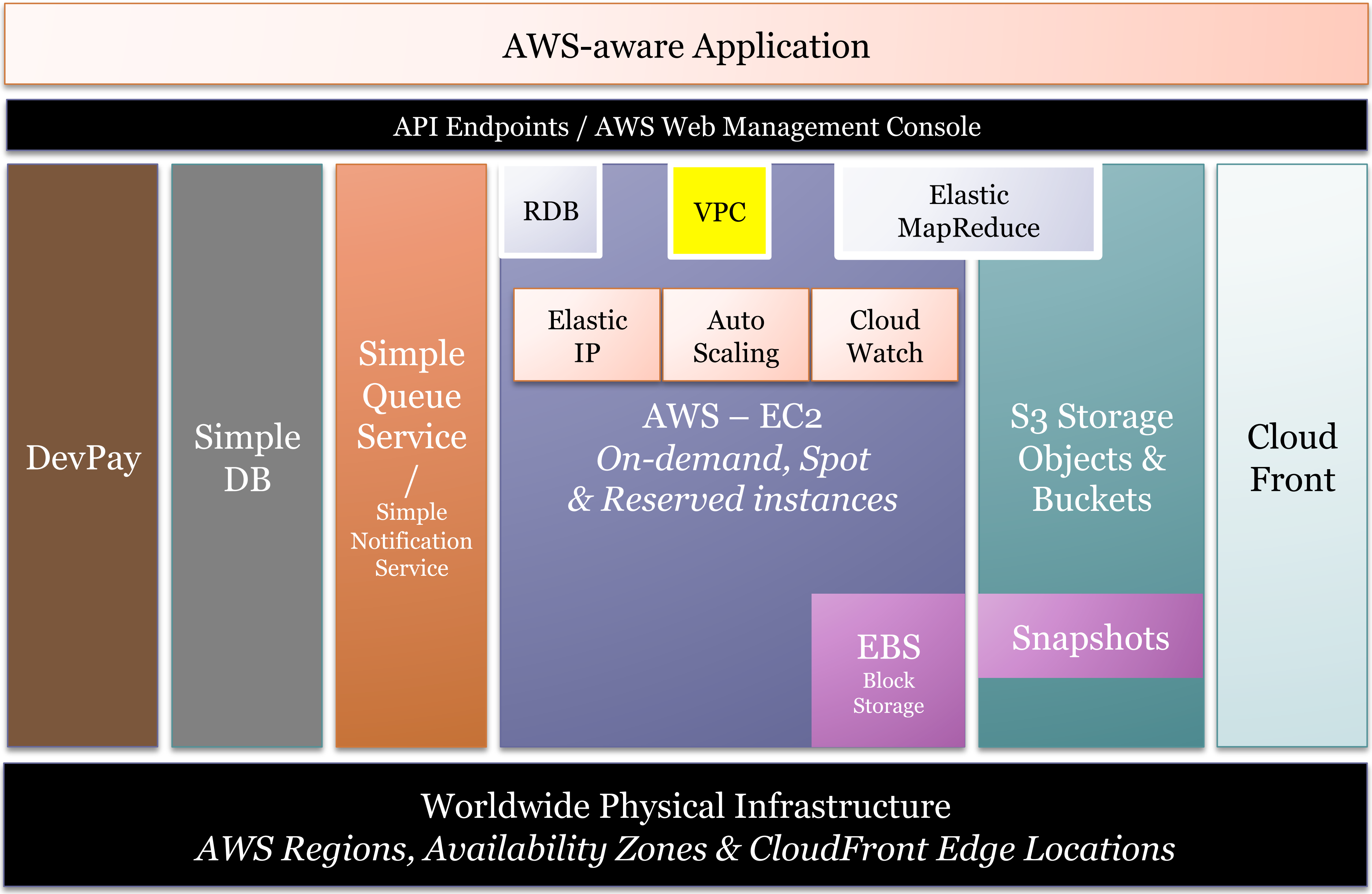
ISMB 2010 Cloud Workshop

- Servers & Compute
  - EC2
    - *On-demand, Spot & Reserved*
  - Elastic MapReduce
  - Auto Scaling
- Internet Content Distribution
  - CloudFront
- Database
  - SimpleDB
  - Relational Database Service (RDS)

Billing & Payment Services

  - DevPay
  - Flexible Payment Service (FPS)

- Messaging
  - Simple Queue Service (SQS)
  - Simple Notification Service (SNS)
- Monitoring
  - CloudWatch
- Networking
  - Virtual Private Cloud (VPC)
  - Elastic Load Balancing
  - Elastic IP
- Storage
  - Simple Storage Service (S3)
  - Elastic Block Storage (EBS)
  - AWS Import/Export
- Support
  - AWS Premium Support



AWS-aware Application

API Endpoints / AWS Web Management Console

DevPay

Simple DB

Simple Queue Service / Simple Notification Service

RDB

VPC

Elastic MapReduce

Elastic IP

Auto Scaling

Cloud Watch

AWS – EC2
On-demand, Spot
& Reserved instances

EBS
Block Storage

S3 Storage Objects & Buckets

Snapshots

Cloud Front

Worldwide Physical Infrastructure
AWS Regions, Availability Zones & CloudFront Edge Locations

Not all services are of equal interest

Of significant interest to science users

- EC2 compute
- S3 object storage
- S3 import/export
- EBS block storage
- SQS
- SimpleDB
- RDB
- VPC
- Elastic MapReduce

Of lesser or moderate interest

- CloudFront
- DevPay
- Auto-scale
- Reserved instances
- Spot instances

ISMB 2010 Cloud Workshop

AWS Services BioTeam uses the most …

- EC2 virtual servers
  - On-demand, Reserved and Spot Instances (we use them all)
- S3 Object storage
  - Used for: incoming data, outgoing analysis results
- EBS Block storage
  - Backups & Data sharing among collaborators
  - Making storage go faster in HPC workflows
  - Faster booting, persistent EC2 virtual servers
- SQS
  - Message passing within our scientific workflows & pipelines
- SimpleDB
  - State-keeping and status information on pipelines & workflows

Elastic Compute Cloud “EC2”

Our baseline foundation for building workflows & applications

Amazon EC2

Core concept

- Web service for requesting virtual server instances with “minimal friction”
- Usage
  1. Select or create an Amazon Machine Image (“ami”)
  2. Set network access/security policies
  3. Select from available ‘instance types’
  4. Start as many server ‘instances’ as you desire
  5. Optional decisions (some at additional cost…)
     - Select locations, enable monitoring, enable elastic IP, attach EBS storage volume, join VPC, etc.
- Pay by the hour for what you consume

Amazon EC2

Three Different Ways To Purchase EC2 Services

- On-demand Instances
  - □ Most common & most popular
- Reserved Instances
  - □ Pre-pay upfront for lower recurring charges & guaranteed access
- Spot Instances
  - □ Auction market for unused EC2 capacity – set bids on what you'd like to pay for EC2 server time

Amazon EC2

Geographic Options

- Four current availability zones
  - ▫ US-East (Northern VA)
  - ▫ US-West (Northern CA)
  - ▫ EU (Ireland)
  - ▫ APAC (Singapore)
- This is important because:
  - ▫ Some pricing varies by zone
  - ▫ High Availability achieved by spanning zones
  - ▫ Data protection laws (!)

ISMB 2010 Cloud Workshop

Simple Storage Service (S3)

Objects & buckets in the cloud …

Amazon S3

- "Storage for the internet"
    - ▫ Web service interface for storing and retrieving any amount of data, at any time, from anywhere on the internet
    - ▫ Core design goals revolve around scalability, reliability, speed and low expense

Amazon S3

What it is

- Read/write/delete unlimited number of objects ranging from 1 byte to 5 gigabytes
- Standard SOAP and REST programmer interfaces
- Objects are stored in buckets, referenced via a developer assigned key
- Bucket & key mechanism can mimic a traditional hierarchical storage grouping (very common)
- Buckets live in one region
- Security & ACL model for public & private content

Amazon S3

Data Protection & Reliability

- Data is written to multiple datacenters within a region synchronously before SUCCESS is returned to any PUT or COPY operation

- S3 Versioning (new in 2010)
  - Available in all regions
  - Can preserve, retrieve & restore every version of every object stored in S3
  - Defaults to most current, versioning use is optional

Amazon S3

Reduced Redundancy Storage (S3 RRS)

- S3 Standard Storage
  - Designed to provide 99.999999999% durability and 99.99% availability of objects over a given year.
  - Designed to sustain the concurrent **loss of data in two facilities**
- S3 Reduced Redundancy Storage (RRS)
  - Designed to provide 99.99% durability and 99.99% availability of objects over a given year. This durability level corresponds to an average annual expected loss of 0.01% of objects.
  - Designed to sustain the **loss of data in a single facility**
- Pricing Difference Example
  - S3: $.15/GB for first 50TB
  - S3 RSS: $.10/GB for first 50TB

Amazon S3

Many different API language bindings, use what you prefer

- Ruby:
  - http://rightaws.rubyforge.org/right_aws_gem_doc/
  - http://s3sync.net/wiki
- PHP:
  - http://undesigned.org.za/2007/10/22/amazon-s3-php-class
- Python:
  - http://code.google.com/p/boto/
- Perl:
  - http://search.cpan.org/dist/Net-Amazon-S3/
- Java:
  - http://jets3t.s3.amazonaws.com/toolkit/toolkit.html
- C#:
  - http://www.codeplex.com/ThreeSharp
- C:
  - http://libs3.ischo.com/index.html

Amazon S3

GUI Tools

- S3Fox
  - http://www.s3fox.net/
- s3sync (ruby vesion)
  - http://s3sync.net/wiki
- And many more …
  - Many different commercial and open source GUIs, shells and interfaces are available
  - All aimed at easier interaction with S3 without direct exposure to API calls

Elastic Block Storage (EBS)

Sometimes you want your storage to look like a disk drive …

Amazon EBS

- S3 storage is ***object*** based
- EBS is ***block-storage***
  - ***It looks like a disk drive and you can treat it as such***
- EBS volumes:
  - Created via API or GUI tools
  - Looks just like a naked disk drive – partitioning & formatting required
  - Can be attached and detached from EC2 servers in same AZ
  - Easy snapshots for replication or backup
- EBS used for
  - Situations where block storage is required
  - Speeding up file IO via striping across many EBS volumes
  - Making persistent EC2 virtual servers that also boot really fast

ISMB 2010 Cloud Workshop

ISMB 2010 Cloud Workshop

Message Passing & State Keeping

Amazon services that help us build scientific pipelines and workflows

Amazon SQS

- Simple Queuing Service
  - Purpose-built system for passing messages
  - This is the “glue” that ties everything together
- Why is this needed?
  - Data on most EC2 servers dies with the instance
  - Need a way to store messages that is fast, reliable, scalable and independent of our compute & storage systems
- The benefit:
  - This is the glue that lets us build agile & elastic systems that can grow and shrink according to need or demand
  - Individual pieces of our pipelines and workflows (compute, storage, etc.) can scale up and down automatically with ease because our messaging is totally independent & asynchronous

Amazon SQS in the real world

- **SQS messages can be ideal "work units" for people building scientific pipelines in the cloud**
- But:
  - Messages have an 8K size limit
- Thus:
  - Many people (including BioTeam) use SQS messages to pass pointers to where the "real" work data is stored …
  - Example:
    - A BioTeam SQS message often contains just the name of an S3 bucket
    - The actual bucket contains:
      - All necessary job/task data including parameters, input data and directions on where to place the output

ISMB 2010 Cloud Workshop

Amazon SimpleDB

- Simple "cloudy" database service
- Familiar but not identical to what people experienced with relational databases will expect
- BioTeam uses SimpleDB for:
  - Storing all state, error and status information about our scientific pipelines and workflows
  - Hard lesson learned after making initial mistakes with SQS queues
  - Huh?
    - Hard to globally poll the state of an SQS queue
    - Polling the queue requires iterating over messages and this resets visibility windows, timeouts and other critical parameters
    - Way easier to make status monitoring "dashboard" or "qstat clone" that queries SimpleDB rather than SQS

Putting it all together

Orchestrating AWS services into a scientific pipeline

Generic Science Pipeline on the Amazon Cloud

- Servers
  - We ***design & deploy*** via EC2 On-demand servers
  - We ***test*** via EC2 Spot-instance servers
- Storage
  - Applications, source code & static data on EBS disks
  - Workflow input data & results stored in S3 buckets
- Message Passing
  - Incoming jobs are encoded as SQS messages in an "input queue"
  - Results are written out to S3 via an SQS "outbound queue"
- Getting work done
  - EC2 nodes start up, connect to the SQS Input Queue and start processing "work units"
  - State, status and error information written into SimpleDB table
- Status Monitoring
  - Status Dashboard queries SimpleDB to find out what is going on

End;

- End of Block 2

- **Want these slides?**
  - **Check http://blog.bioteam.net**

- Want actual hands-on training in a small class environment?
  - 2-day "AWS Fundamentals for Science & Engineering" classes
  - http://bioteam.net/services/cloud-training.html
  - Upcoming open classes
    - September 2010 – Providence, Rhode Island
    - October 2010 – Hannover, Germany
    - More to be announced …

Mapping informatics to the cloud

Block 2: 11:15 – 11:40

Informatics on the cloud: the good, the bad & the ugly

BIOTEAM
Enabling Science

HPC & AWS: A whole new world

Radical changes coming to our happy cluster/HPC world …

- Research IT folk have spent careers tuning systems for sharing resources among multiple groups
- But …
  - Utility model offers dedicated resources
  - EC2 not particularly architected for our needs
  - Best Practices & reference designs will all change

ISMB 2010 Cloud Workshop

HPC & AWS: A whole new world

2010 Current State

- Still a challenge to integrate local resources with cloud-based systems
  - ▫ "Cloud bursting" (ugh!)
  - ▫ Networking, bandwidth & data motion still the largest issues
- Clever people working on some of the most vexing issues
  - ▫ Amazon VPC
  - ▫ SDM-managed Grid Engine
  - ▫ Univa UniCloud
  - ▫ CloudSwitch Inc.
  - ▫ Aspera Software

ISMB 2010 Cloud Workshop

HPC & AWS: A whole new world

2010 Current State

- Most people in our community are …
  - ▫ Moving entire workflows rather than "linking grids"
    - Cherry picking the workflows that make sense of course!
  - ▫ Integrating existing workflows with tasks/steps performed on the cloud
    - Or decoupling existing pipelines in preparation for such a transition
  - ▫ Moving test, development & prototype systems over

AWS was not designed for us

- Built to support internet-scale “web 2.0” style applications
- Designed for wide-spread scaling, elasticity and high levels of redundancy at lowest possible delivery cost to customer
- Virtual ***everything*** *is slow*

AWS was not designed for us

- Focus on improving latency for different user base
  - Latency at the edge (CloudFront)
  - Not latency at the core
- Servers and services that can span datacenters, regions and continents
  - Great for load balancing & high availability
  - Not so great for high performance

This will (probably) change

- NDA issues could make this a tricky topic
- Amazon never discusses unreleased products
- However:
  - ▫ Amazon "gets it" with respect to HPC issues on the current AWS platform
  - ▫ Amazon has made statements expressing strong support and interest in the HPC market
  - ▫ Amazon has a great track record of rapid product advancement, enhancements and development
  - ▫ Infer what you will
    - (and keep pestering AWS product reps!)

HPC & Amazon EC2

Current State

- New feature/enhancements rate is impressive
- However
  - EC2 instances still poor on IO operations
  - EBS would be awesome with some enhancements
    - Many-reader/One-writer option for EBS volume would be fantastic
    - *EBS volumes bootstrapped from EBS snapshot is one currently workable option …*
  - Still poor support for latency-sensitive things and workflows that prefer tight topologies

ISMB 2010 Cloud Workshop

HPC & Amazon EC2

This matters because …

- Compute power is trivial to acquire in 2010
- Life science often performance-bound by speed of storage & IO operations
- We are currently being buried under mountains of data

AWS & Internet Networking

Current State

- Can be challenging
- EC2 instances use private hostname & IP space
- EC2 instances have public IP endpoints
  - … but they don't necessarily "know" them
  - Internet traffic to/from instance via upstream NAT
- Dealing with NAT and the private IP space can be a hassle
  - This is why MANY best-practice deployments implement their own software-based VPN layers

AWS & Internet Networking

Amazon Virtual Private Cloud ("VPC")

- Amazon VPC designed to solve these issues
- It can
  - Extend your existing subnet(s) & IP space into EC2
  - Seamlessly integrate EC2 instances with your production network
  - Route internet-destined traffic through a security endpoint on your network first
- However
  - Current state not as cool/usable as it seems
  - Currently listed as tested/approved gateways:
    1. Cisco Integrated Services routers w/ Cisco IOS 12.4 (or later)
    2. Juniper J-Series routers w/ JunOS 9.5 (or later)
    3. Juniper SSG series w/ ScreenOS 6.1 or later
    4. Juniper ISG series / ScreenOS 6.1 or later

AWS & Internet Networking

Amazon Virtual Private Cloud ("VPC")

- I would be using VPC today if I could
- Would love to be able to demo it
    - Sadly I don't have expensive Cisco & Juniper endpoints handy (although Juniper SSG & ISG are within reason ..)
    - I suspect may potential VPC customers are in this position
- Still searching for HOWTO or whitepaper on how to configure Linux/OpenVPN, OpenBSD/IPSEC or some other server-based endpoint for VPC gateways

HPC & AWS Internal Networking

What you need to know:

- Absolutely no guarantee that instances within a single EC2 reservation will be on the same subnet
- Your control over topology is largely constrained to picking what global region you want to operate in
- This can freak out certain MPI stacks and other software that ***assumes*** a consistent flat subnet topology

HPC & AWS Internal Networking

And one more thing …

- Elastic IPs are fantastic
- Cost is reasonable enough that it should be considered for important parts of AWS workflows or command/control/management systems
- However
  - Never use one for internal data transfer
    - AWS data transit fees will apply
  - Make sure your instance-to-instance communication and data passing is via the internal private IP space and network

HPC & AWS Summary

- Virtualized networking is ‘reasonable’ but there are certainly issues that need to be worked around
- Network latency can be high
- Virtualized storage I/O is far slower than anything we can do with local resources. Absolute fact.
- Still hard to share data/storage across many systems
- Inability to currently request EC2 nodes that are “close” in network topology terms is problematic (but likely to change)
- MapReduce is not a viable solution for everyone

HPC on Amazon Web Services

Moving forward …

Scalable Architectures

Pick your battles …

- AWS is meant to enable 'frictionless' resource scaling (and shrinking) as needed
- This paradigm can not be used to advantage on all scientific workflows
- For each workflow
  - ▫ Identify the monolithic components & bottlenecks
  - ▫ Can steps be decoupled? Can work units be shrunk?
  - ▫ For best effect workflows and work streams will likely have to refactored

Is hadoop in your future?

Look towards hadoop

- Hadoop and implementations like Elastic MapReduce may represent a popular 'way forward' for many HPC cloud apps
- Likely worth spending some time evaluating and experimenting with it
- List of available hadoop-aware bio* apps seems to be increaseing

Scalable Architectures

Characteristics of a scalable cloud workflow

1. Increasing available resources results in increased performance or throughput
2. Low operational/administrative burden
3. Resilient, self-repairing and fault tolerant
4. More cost-efficient as it grows
   - Cost per work-unit is smaller as system scales

Tips & Tricks

Design for failure

- Always design and implement mechanisms for automatic recovery from failure
- Pessimism is a good thing in this context
- Think long and hard about how to ***detect*** failure
- Think long and hard about how to ***respond*** to failure
  - Without human intervention ideally …
- Good designs should not care about instance failures, sudden terminations and unplanned reboots

Tips & Tricks

Design Challenge

- If you can design your application workflow so that it functions while running on EC2 Spot Instances you have likely solved 95% of potential failure points
  - This will pay of no matter how/where you run
- Even if you never run in production under Spot Instances it might be a desirable design pattern to enforce stateless and exceptionally fault-tolerant architectures

Tips & Tricks

Fault Tolerance Strategies

1. Have a sensible backup & restore mechanism for your data, apps & results
   - Automate it
2. Build workflows that can resume work upon reboot
3. Configuration management is critical
4. Keep instances stateless as much as possible
   - State information can live elsewhere
     - EBS, S3 Buckets, SimpleDB, RDB, SQS Queues, etc.

Tips & Tricks

Serial or batch processing at-scale

- Loose coupling is ideal
- Break worfklow steps up into independent components
- Design components so they can scale up and down independently from everything else
- Try for stateless and asynchronous communication

Tips & Tricks

Serial or batch processing at-scale

- If you have independent loosely connected components that can self-scale and communicate asynchronously
  - You are in cloud nirvana
  - Application or workflow will scale horizontally
  - Well suited for the brave new world of multi-tenancy and cloud operation

Tips & Tricks

Serial or batch processing at-scale

- Amazon provides tools to assist with decoupling
- Often this involves SQS Queues
- Used for
  - Buffering between components (handle workload size changes)
  - Message/work/task passing among components & pipelines

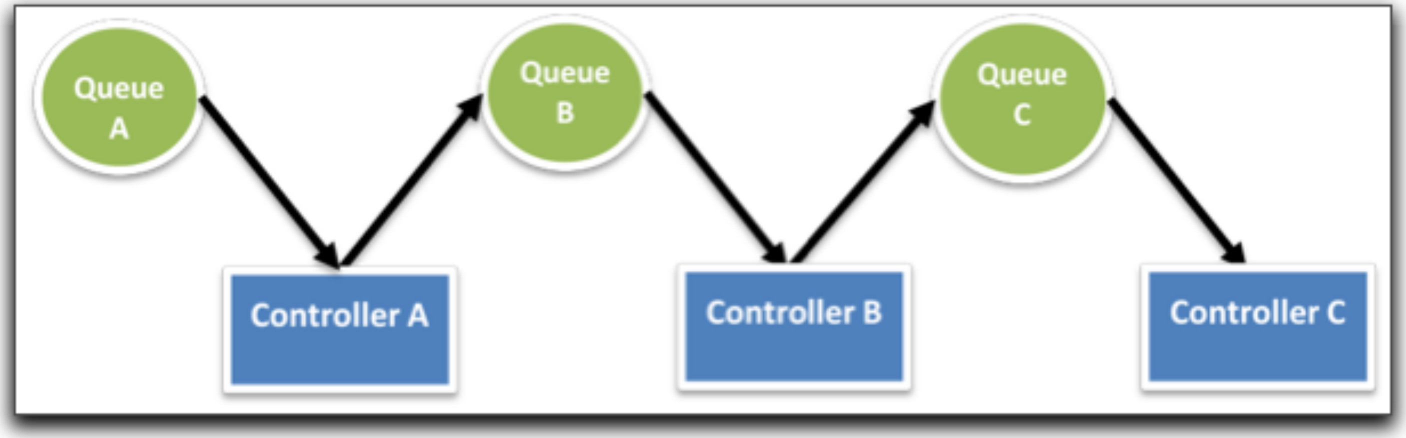


Image source: Jinesh Varia, "Architecting for the Cloud" AWS whitepaper.

Tips & Tricks

Dealing with slow IO

- Ideal situation
  - Design around slow IO
  - Increase performance/throughput via scaling out onto more EC2 instances
- When that is not possible
  - Elastic Block Store (EBS) can help

Tips & Tricks

Dealing with slow IO

- In almost all cases an EBS volume will be faster than local instance storage
  - Difference is more pronounced on larger instance types
- EBS can help with
  - Instance boot times
  - Deep instance customization & state preservation
  - Attaching faster storage to an instance
  - Sharing data via EBS snapshots
    - This is good for internal use
    - … also a great way to share data with collaborators

Tips & Tricks

Data Movement – Think Parallel

- I am guilty of using the wget/curl paradigm when pulling from S3
  - ▫ Nothing about S3 storage requests needs to be serial or sequential
  - ▫ Request parallelization via multiple concurrent threads is popular and recommended
  - ▫ Same thing can be done to SimpleDB GET and BATCHPUT requests if needed

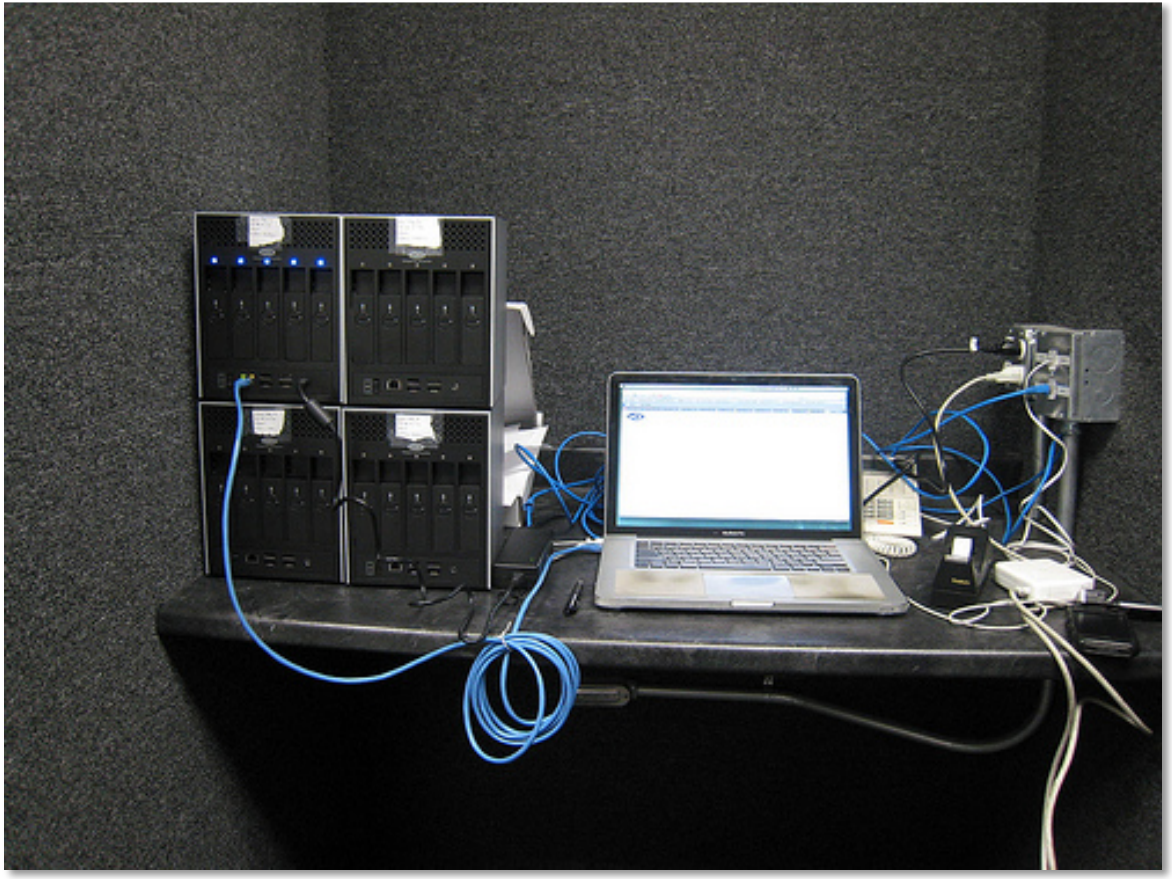
Tips & Tricks

Data Movement – Think Parallel

- In 2009 we had to pull ~20TB from S3
- We used a simple Java S3 library that was multi threaded to pull tens of thousands of objects from S3 buckets
    - ▫ Easily able to sustain 50 megabytes/second download from both a 300mbit and 1gbit circuits
    - ▫ CPU & RAM limits biggest bottleneck for us
- Depending on your location*
    - ▫ We believe that AWS S3 is not the bottleneck
    - ▫ Saturation of your internet connection should be possible

Data movement in the real world

Example: Data Export Station using portable LaCie RAID enclosures



Data movement in the real world

Example: eSATA hotswap drive enclosure

Data movement in the real world

Note: data management, movement & QC is non-trivial in the real world

End;

- End of Block 2

- **Want these slides?**
  - **Check http://blog.bioteam.net**

- Want actual hands-on training in a small class environment?
  - 2-day "AWS Fundamentals for Science & Engineering" classes
  - http://bioteam.net/services/cloud-training.html
  - Upcoming open classes
    - September 2010 – Providence, Rhode Island
    - October 2010 – Hannover, Germany
    - More to be announced …

ISMB 2010 Cloud Workshop

Best Practices & Lessons Learned

Block 3: 11:45 – 12:10

An attempt at some practical advice

BIOTEAM
Enabling Science

Mega Lesson #1

- You need at least 2 cloud deployment strategies:
  1. Handling legacy applications & pipelines
  2. Handling new or re-architected applications

Mega Lesson #1, continued

- Why?
  - To take best advantage of the cloud you need to rewrite, redesign and re-architect
  - But:
    - Often you will have applications or workflows where it is simply not worthwhile to do any significant engineering changes – you just want to run them "in the cloud"

- Thus:
  - Best to build strategies for handling both "new" and "old" situations

Mega Lesson #1, continued

- Legacy Approach Should Include
  - Methods for replicating on the cloud what you are already doing inhouse
    - Provisioning systems with informatics binaries & data
    - Building Grid Engine compute farms & clusters
    - Additional methods for getting data in, results out as well as status/state monitoring

Mega Lesson #1, continued

- “Cloudy” Approach Should Include
  - Basic reference architecture for doing things in the new scalable & elastic “cloud way”
  - Loosely coupled, asynchronous systems with many potential scale up|down points that communicate via message passing systems and store state information in something like SimpleDB

Mega Lesson #2

- Use of a centralized configuration management system is **absolutely essential**
- It does not matter **what** you use, just that you use one
- Why?
  - The configuration management system is the heart and soul of your methods for automatically automating and orchestrating the deployment and management of large & complex systems
  - Delivers the most value and provides the most operational efficiency

Mega Lesson #2, continued

- It is a **huge mistake** to constantly bundle and re-bundle Amazon EC2 AMI images in order to update software, install new code or add new features
- **Why?**
  - Each time you re-bundle an AMI you waste at least 40 minutes of your life. It's a hugely significant time sink.
  - The more AMI images you have to maintain the greater the wastage & inefficiency

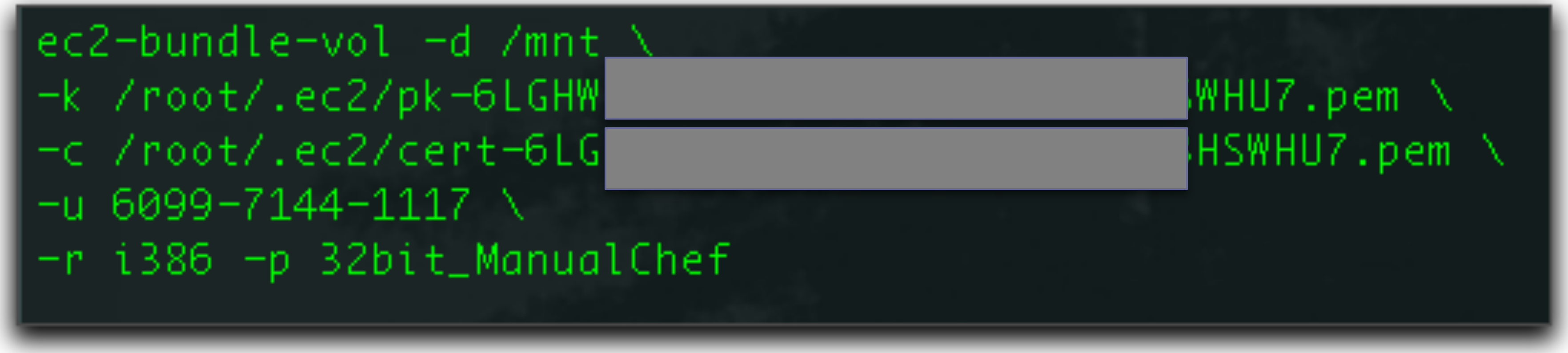
Mega Lesson #2, continued

- With a configuration management system you maintain a very small number of EC2 AMIs that you can then “orchestrate” or provision into ***whatever*** you need

- After learning this lesson the hard way, BioTeam now manages only four EC2 server images:
  - 32bit Linux (S3 boot & EBS boot versions)
  - 64bit Linux (S3 boot & EBS boot versions)

Step 1 – Make a bundle

ISMB 2010 Cloud Workshop

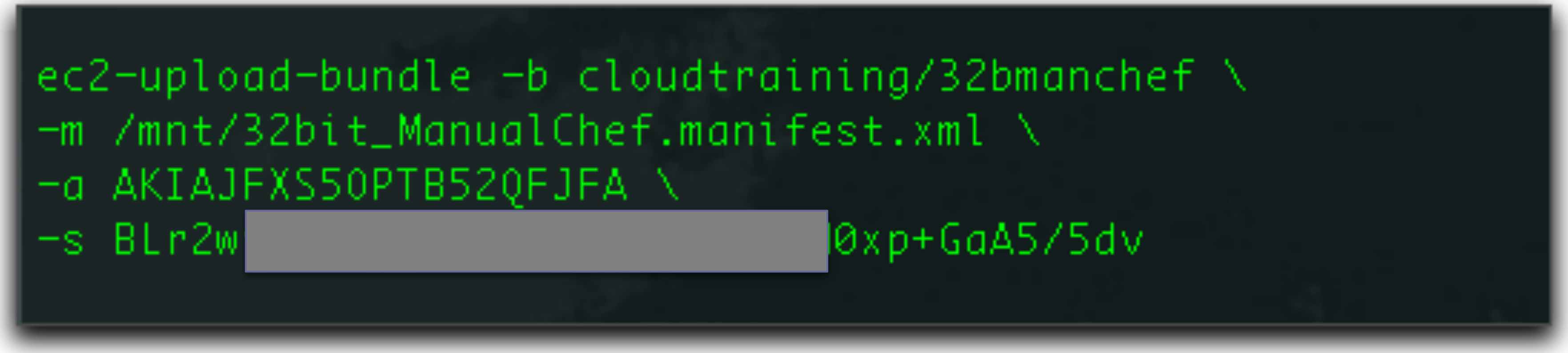


```
ec2-bundle-vol -d /mnt \
-k /root/.ec2/pk-6LGHW[redacted]WHU7.pem \
-c /root/.ec2/cert-6LG[redacted]HSWHU7.pem \
-u 6099-7144-1117 \
-r i386 -p 32bit_ManualChef
```

DO

Step 2 – Upload bundle to S3 bucket

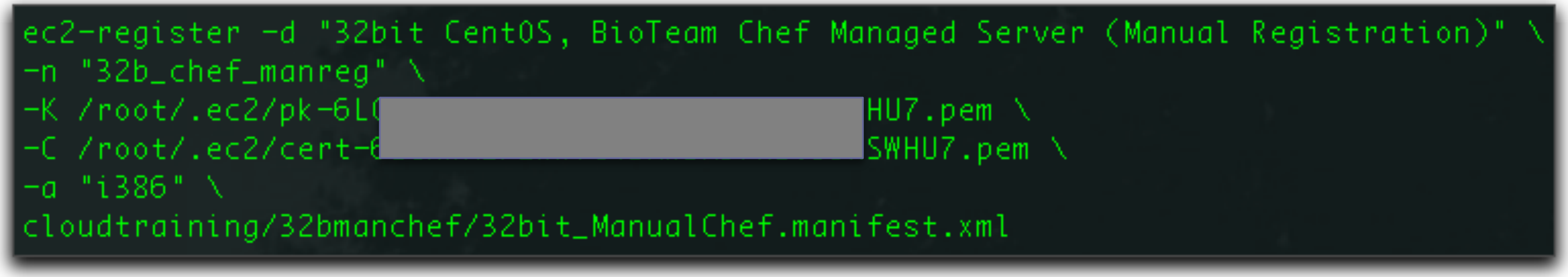
ISMB 2010 Cloud Workshop



```
ec2-upload-bundle -b cloudtraining/32bmanchef \
-m /mnt/32bit_ManualChef.manifest.xml \
-a AKIAJFXS5OPTB52QFJFA \
-s BLr2w[redacted]0xp+GaA5/5dv
```

DO NOT

Step 2 – Register & receive new AMI ID



```
ec2-register -d "32bit CentOS, BioTeam Chef Managed Server (Manual Registration)" \
-n "32b_chef_manreg" \
-K /root/.ec2/pk-6L[redacted]HU7.pem \
-C /root/.ec2/cert-6[redacted]SWHU7.pem \
-a "i386" \
cloudtraining/32bmanchef/32bit_ManualChef.manifest.xml
```

DO NOT DO THIS! (*often*)

Configuration Management

There Is More Than One Way To Do It (pick what you like …)

- Chef - http://wiki.opscode.com/display/chef/Home
- Puppet - http://reductivelabs.com/trac/puppet/
- CFEngine - http://www.cfengine.org/
- SystemImager - http://wiki.systemimager.org/index.php/Main_Page
- PoolParty - http://auser.github.com/poolparty/
- Genome - http://genome.et.redhat.com/
- JEOS - http://en.wikipedia.org/wiki/Just_enough_operating_system
- And many more …

Systems Orchestration

- **Bork**!
- BioTeam's preferred configuration management system
- OpsCode Chef
  - http://www.opscode.com

ISMB 2010 Cloud Workshop

Chef is

At a high level …

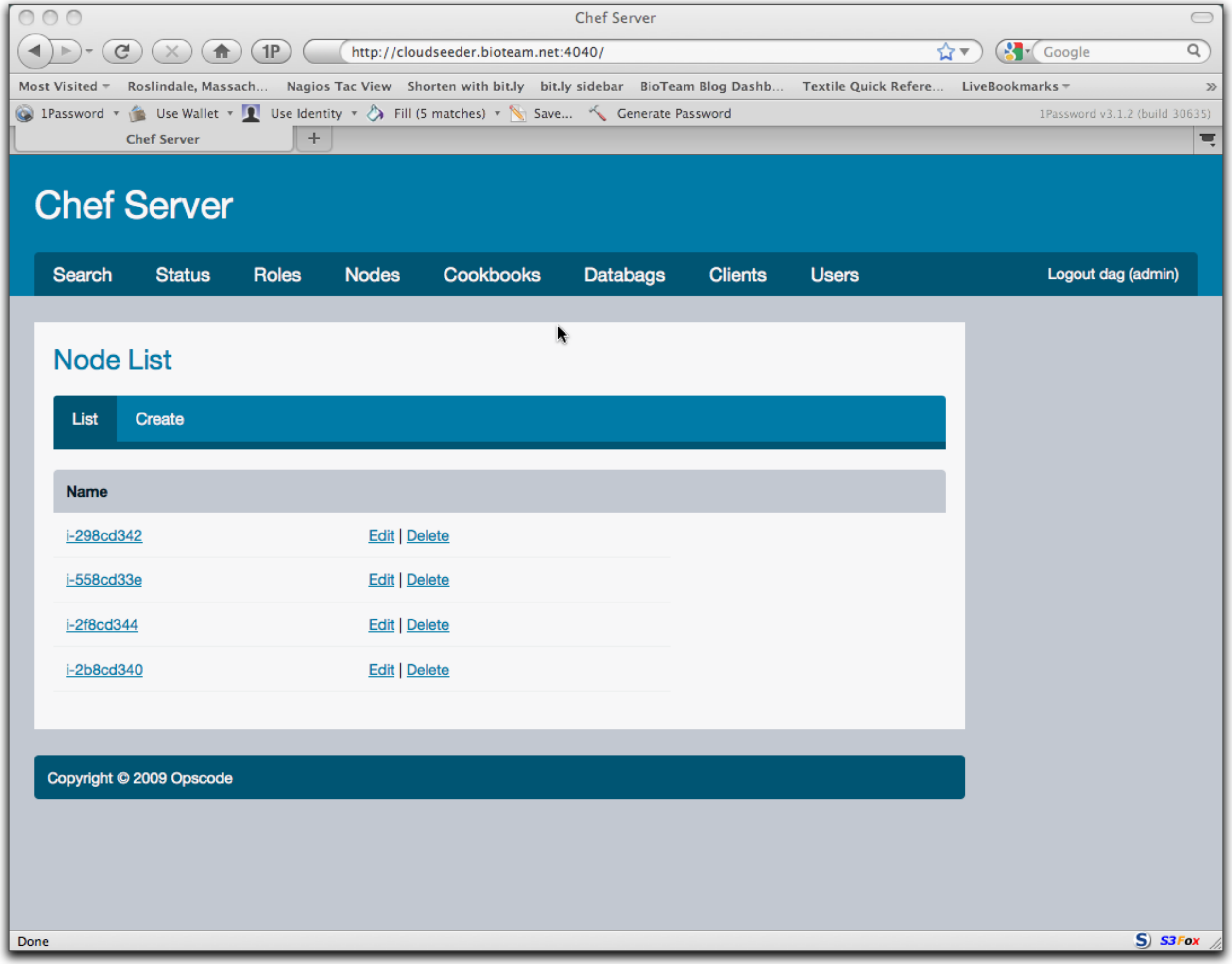
- Library for configuration management
- A standalone configuration management system
- A full featured systems integration platform
- **… natively aware of cloud platforms and cloud instance metadata**

Cooking with Chef

Why?

- New product built from years of CM experience
- Experience & techniques we can leverage
    - ‘Puppet’ developer spent years with ‘cfengine’
    - Chef developers spent years as ‘puppet’ consultants
- And at 2010 BioITWorld Cloud Workshop
    - … first three speakers mentioned that they used Chef - totally unplanned & unexpected

Chef Server Web UI



Chef Server

Node List

Name	
i-298cd342	Edit | Delete
i-558cd33e	Edit | Delete
i-2f8cd344	Edit | Delete
i-2b8cd340	Edit | Delete

Copyright © 2009 Opscode

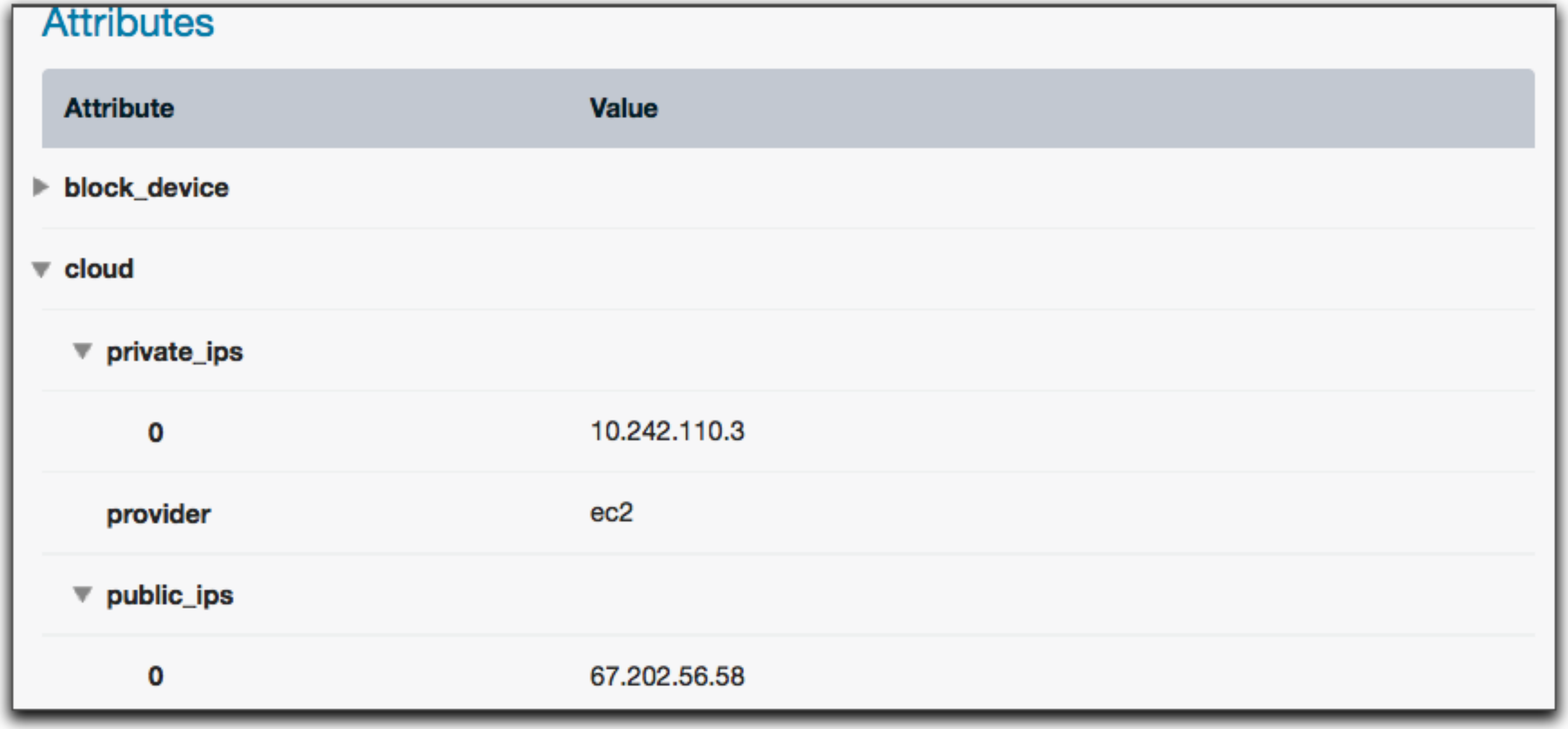
As a result of Chef …

BioTeam now manages only 4 Linux AMI's in the cloud

- … using knife, chef-solo or Chef Server we can orchestrate AWS services into whatever we need in a matter of minutes.

Chef understands cloud metadata

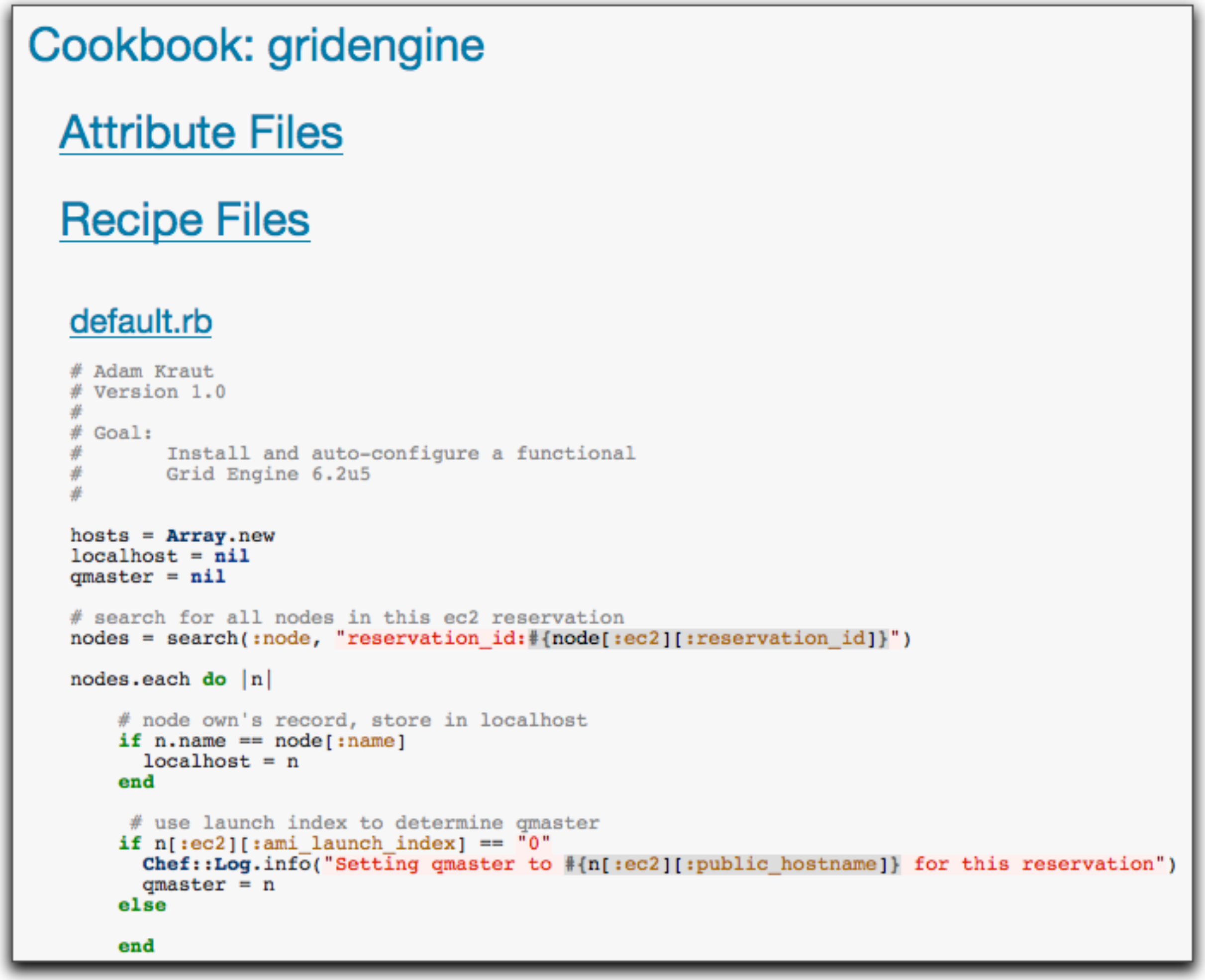
ISMB 2010 Cloud Workshop



Attributes

Attribute	Value
▶ block_device	
▼ cloud	
▼ private_ips	
0	10.242.110.3
provider	ec2
▼ public_ips	
0	67.202.56.58

Snippit of our Grid Engine recipe



Cookbook: gridengine

Attribute Files

Recipe Files

default.rb

```
# Adam Kraut
# Version 1.0
#
# Goal:
#       Install and auto-configure a functional
#       Grid Engine 6.2u5
#

hosts = Array.new
localhost = nil
qmaster = nil

# search for all nodes in this ec2 reservation
nodes = search(:node, "reservation_id:#{node[:ec2][:reservation_id]}")

nodes.each do |n|

    # node own's record, store in localhost
    if n.name == node[:name]
      localhost = n
    end

     # use launch index to determine qmaster
    if n[:ec2][:ami_launch_index] == "0"
      Chef::Log.info("Setting qmaster to #{n[:ec2][:public_hostname]} for this reservation")
      qmaster = n
    else

    end
```

Chef lets you …

Treat your infrastructure as code

- Manage configuration as idempotent resources
- Put resources together as recipes
- Group recipes into roles
- Track it all like source code
- Configure your systems

Chef in action

- We’ll use Chef to drive our demos in the next section of the workshop

Other Best Practices …

- Warning:
  - Next set of slides may reference Amazon products or product features that we did not cover in our short overview presentation in Block 1
  - Even though we did not cover these they remain a core part of our collected set of best practices
  - Feel free to stop me or ask questions if you don't know what something is

Next set of recommendations covers

- AWS Accounts
- Credential Management
- Configuration Management
- AWS Zones & Regions
- AWS Elastic IP addresses
- EC2 Instance Metadata
- Monitoring & Logging
- Identity Management
- AWS S3
- AWS EBS
- AWS RDS
- Message Passing
- Structured Data
- Architecture & Deployment

AWS Account Best Practices

- Map AWS accounts to email aliases, not individual staff addresses
- Use consolidated billing
  - Provides major advantages
- Protect payer account with MFA token
- Protect sensitive S3 owner accounts with MFA token

Credential Best Practices

- Manage access keys carefully
- Implement key rotation process in critical workflows
  - Important to build key rotation into your app at the beginning
  - Much harder to implement after the fact
  - Testing is critical!

Quick AWS Credential Reference

If you want to …	Use this credential
Make REST Query API calls to AWS products	Access Keys
Make SOAP API calls to AWS products	X.509 Certificates (except for S3 which requires your Access Keys)
View secured AWS support, forums or management console web pages	Sign-in credential (MFA optional)
Use EC2 command line utilities	X.509 Certificates
Launch or login to an EC2 instance	Key Pairs
Bundle a Linux/UNIX AMI image	X.509 Certificates & AWS Account ID to create the bundle, Access Keys for uploading the bundle to S3
Bundle a Windows AMI image	Access Keys for bundle & S3 upload
Share AMI or EBS volume snapshot	AWS Account ID of other user (no hyphens)

ISMB 2010 Cloud Workshop

Configuration Management

- Hopefully we have convinced you of the necessity for CM
- Use one
  - CM systems that are ‘aware’ of EC2 instance metadata and user-data are ***extremely*** useful

Best Practices - AWS Zones & Regions

- Understand the difference between AWS Regions & Availability Zones
  - US-East vs. us-east-1b
- Pick a preferred region
- Pick a preferred zone within the preferred region
- Understand what services span zones & regions (and those that do not)

Elastic IP Addresses

- Use Elastic IP for essential systems & services
- ***Never*** use Elastic IP address for internal data transfer or communication
  - Internal communication within AWS is free but if you route through Elastic IPs you will incur unnecessary bandwidth charges

EC2 Instance Metadata

- It is embarrassing how long it took me to discover this EC2 feature
- It is a HUGE thing and you need to understand it
- You should:
  - Make effective use of this extremely useful data
  - Make effective use of instance user-data for bootstrapping

Monitoring & Logging

- Know what metrics are important and which are not
- Figure out if AWS CloudWatch is OK or if something else is required
- Custom monitoring practices per-role or per-workflow is perfectly OK
- Logging to an external, centralized system is extremely beneficial

- BioTeam usually:
  - Configures EC2 systems with Syslog-NG set up to log messages and events to a central persistent log host.
  - This is especially useful in the cloud where EC2 server instances may die without warning. A central loghost is great for debugging and forensics

Identity Management

- AWS keys for primary access
- Create Service accounts if necessary
- Use CM system to push out additional SSH keys etc.
- ***Never bake passwords or credentials into systems***
- Use external authentication service for apps that require credentials
  - It is very very trivial to hook your systems up to a central identity management and authorization system (LDAP, OpenID, etc.)
  - Products like Amazon VPC can even securely connect your cloud nodes to existing Active Directory servers for instance
  - Central access control is far safer and far easier to manage than manual handling of access information or (worse) embedding accounts and passwords directly into server AMI images

AWS S3

- Target as your primary data store
- Optimize bucket layout for parallel get/put operations
- Remember that S3 bucket names have to be globally unique worldwide
  - ▫ This means that it is critical that any code you have that automatically creates buckets MUST be able to handle and work around name conflicts
- For very large S3 use cases there are fundamental things like object naming schemes and bucket organization methods that can have real-world performance impact

AWS EBS

- Use when S3 is not optimal
- Understand when and how to build EC2 servers that boot off of EBS
- Master effective use of snapshots
- IO profiling is essential
- Understand when to stripe EBS and when not to
    - Be wary of internet HowTos on this subject, their requirements may be quite different from yours
    - Currently most of the information on the internet regarding EBS performance tuning is written expressly for database people seeking to improve random IO performance – this is often NOT what scientists need most

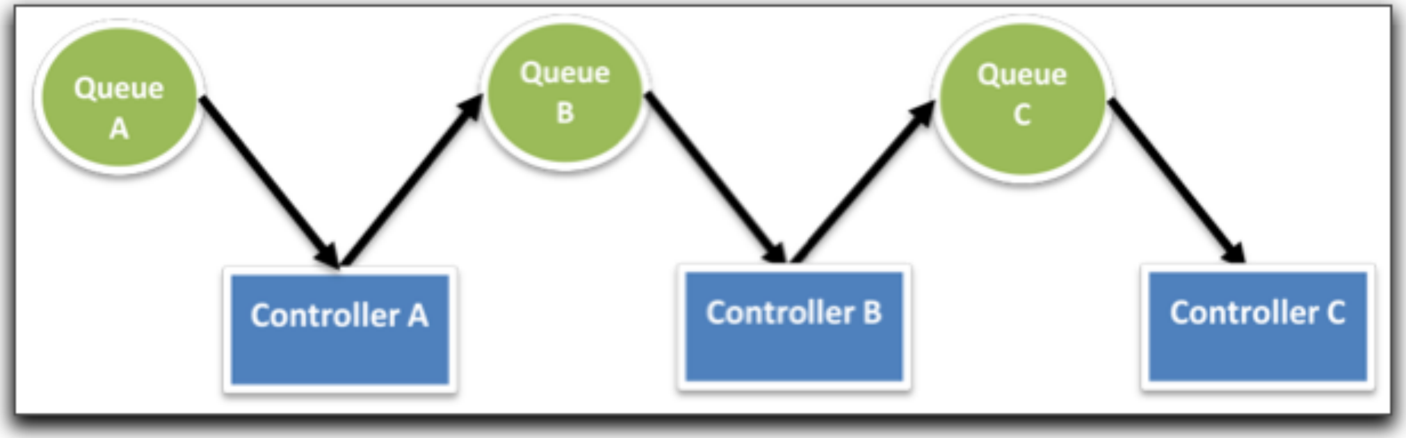
AWS RDS

- Not rocket science, it's just a managed mySQL service
- Know the product well enough to understand when it could offer:
    - Time savings
    - Operational effort savings
    - Feature savings

Message Passing

- Use producer, consumer, and controllers
- SQS is great but there are alternatives if you need them
- Use portable message formats (JSON, XML)
- Do not rely on persistence

ISMB 2010 Cloud Workshop



Structured Data

- Find a natural domain decomposition
- SimpleDB is fine but alternatives exist if needed
- Use S3 keys to get around attribute size limits
- Choose a consistency model that works best for your app

Architecture & Deployment

- Pick your battles
- Design for failure
- Automate backups
- Strive for success under Spot Instances
- Learn to live with eventual consistency
- Decouple process steps
- Glue steps together with message passing
- Don't leave state on ephemeral systems

End;

- End of Block 3
- **Want these slides?**
  - **Check http://blog.bioteam.net**
- Want actual hands-on training in a small class environment?
  - 2-day "AWS Fundamentals for Science & Engineering" classes
  - http://bioteam.net/services/cloud-training.html
  - Upcoming open classes
    - September 2010 – Providence, Rhode Island
    - October 2010 – Hannover, Germany
    - More to be announced …

Amazon Web Services Workshop

Block 4: 12:15 – 12:40

Demo Time

Live Demo Time

- Lets hope this works
- Would like to demo some common methods we use (also a great way to show off what Chef can do …)
- Demos
  1. Provision app + data onto a node
  2. Set up a Grid Engine system in the cloud
  3. Workflow system w/ message passing demo

Demo 1- Provisioning EC2 Node

- Why? This is part of the "legacy" use case
  - Need to run apps in the cloud just like we do inhouse with nothing fancy.
- Sometimes we just want to:
  - Boot a 'vanilla' server AMI on EC2
  - Install & compile a bioinformatics tool ('maq')
  - Get some work done
- How?
  - Trivial. It's just a Chef Recipe to us.

Demo 1 – Recap

- { Assuming it worked! }
- Questions:
  - We just showed how simple it is to 'orchestrate' a server into having whatever we want on it …
  - How could we extend this?
  - What about instance user-data? Bootstrapping?

Demo 2 – Simple SGE Cluster

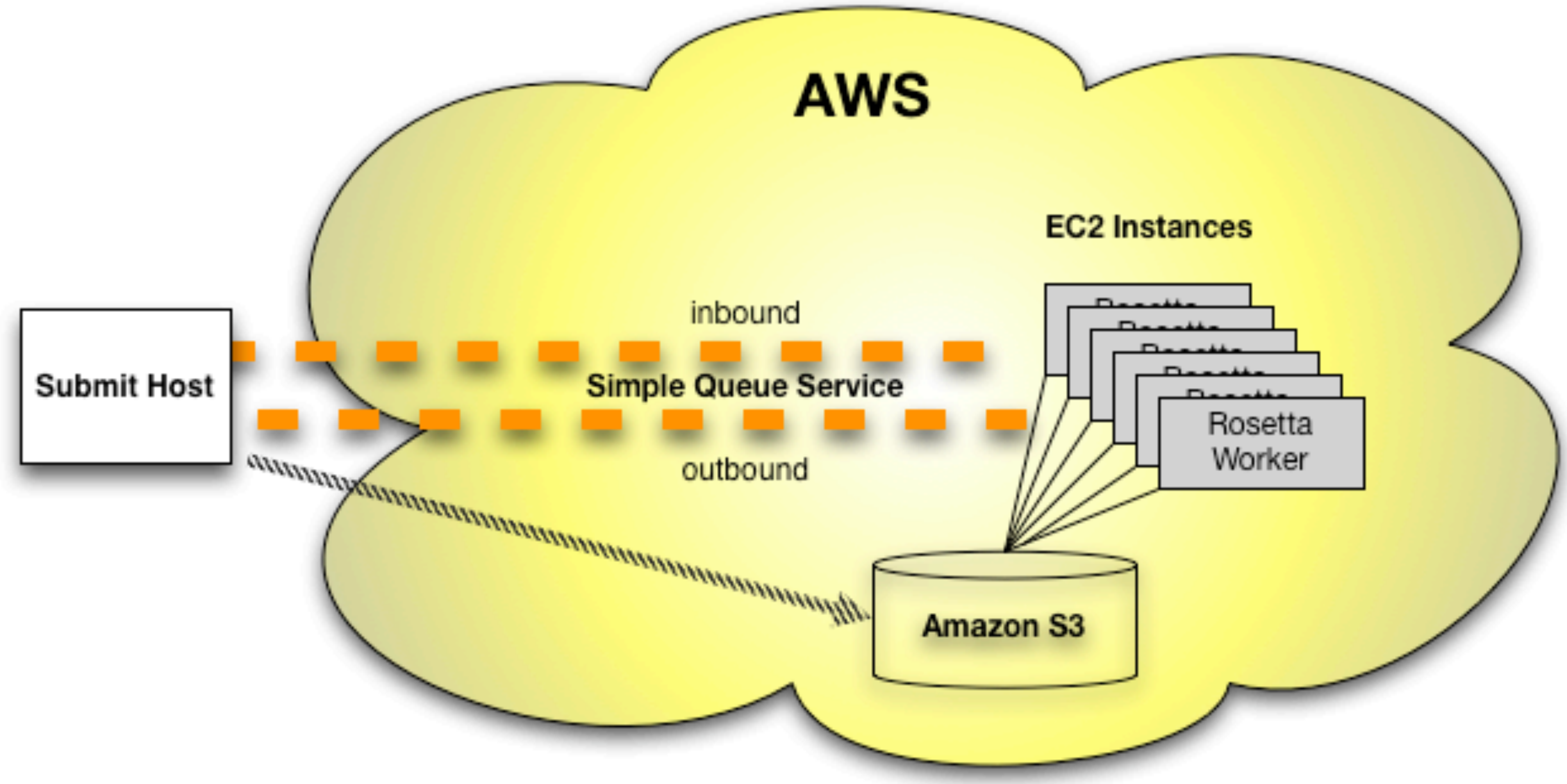
- Still part of the “legacy” use case
  - Sometimes it is useful to replicate our cluster/compute-farm inside the cloud
  - Chef Orchestration also makes this trivial
- We will:
  - Demo a single-node self-organizing Grid Engine system
  - Why single node? We don’t have the time in this session to wait for many nodes to boot & provision themselves …

Demo 3 – Cloud Workflow Demo

- Imagine this type of workflow:
  1. Upload informatics jobs as "work units"
  2. Notify an "inbound" SQS queue that work is ready
  3. Start EC2 node(s) processing the "work" queue
     - When complete, notify a second SQS queue that tasks are ready for QC
  4. Perform Quality Control on results
     - If pass, notify SQS "Outbound" results queue
     - If fail, resubmit work unit to "Inbound" queue

BIOTEAM
Enabling Science

Demo 3 – “Cloudy” Workflow

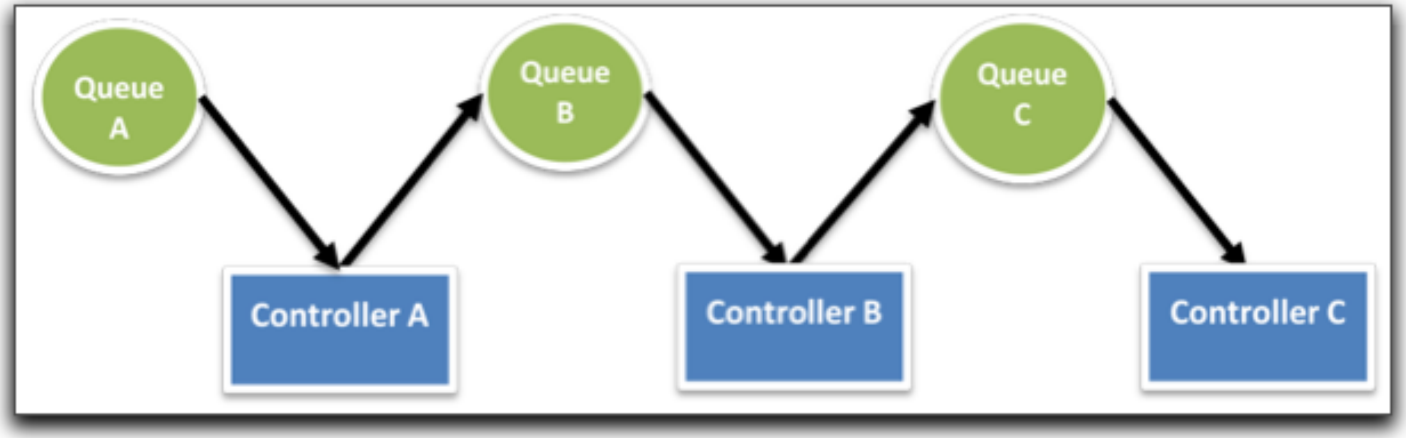


Demo 3 – Behind the scenes

- Consumer.rb script
  - Uploads jobs and notifies SQS Queue "Queue A" that work is ready to be processed
- ControllerA script
  - Iterates over work units in SQS Queue "Queue A"
  - When work completes, notify SQS "Queue B"
- ControllerB script
  - Iterates over work units in SQS Queue "Queue B"
  - Performs Quality Control tests on work results
  - If fail, resubmit work unit to to "Queue B"
  - If pass, notify SQS Queue "Queue C"
- Consumer.rb script
  - Iterate over messages in "Queue C"
  - Deliver results & cleanup

Demo 3 – "Cloudy Workflow"

- Questions:
  - What is the point of doing this?
  - What advantages does having Incoming, QC and Outbound SQS queues give us?
    - *Hint: mostly has to do with elasticity and the ability to scale up and scale down worker, QC and results-handling nodes as needed.*



End;

- End of Block 4

- **Want these slides?**
  - ▫ **Check http://blog.bioteam.net**

- Want actual hands-on training in a small class environment?
  - ▫ 2-day "AWS Fundamentals for Science & Engineering" classes
  - ▫ http://bioteam.net/services/cloud-training.html
  - ▫ Upcoming open classes
    - September 2010 – Providence, Rhode Island
    - October 2010 – Hannover, Germany
    - More to be announced …

Thanks!