Who am I?

- Chirag Jog
- CTO, Clogeny Technologies - Cloud Computing Experts
- Python developer
- Open Source Contributor – Linux Test Project, Linux Kernel, boto etc

Clogeny
Technologies Pvt.Ltd.

Innovation → Execution → Solution → Delivered

MANAGE YOUR AMAZON AWS ASSETS USING BOTO

Email: **chirag@clogeny.com**
Web: http://www.clogeny.com

Presentation - Talks

What is boto?

A Python interface to Amazon Web Services

Agenda

- Basics of Cloud Computing, Amazon Web Services
- Building blocks of one's cloud infrastructure – EC2, EBS, S3, ELB
- Introduction to Boto
- Using Boto – EC2, EBS, S3, ELB
- Sample scripts
- Coding Guidelines
- Manage Non-AWS Clouds

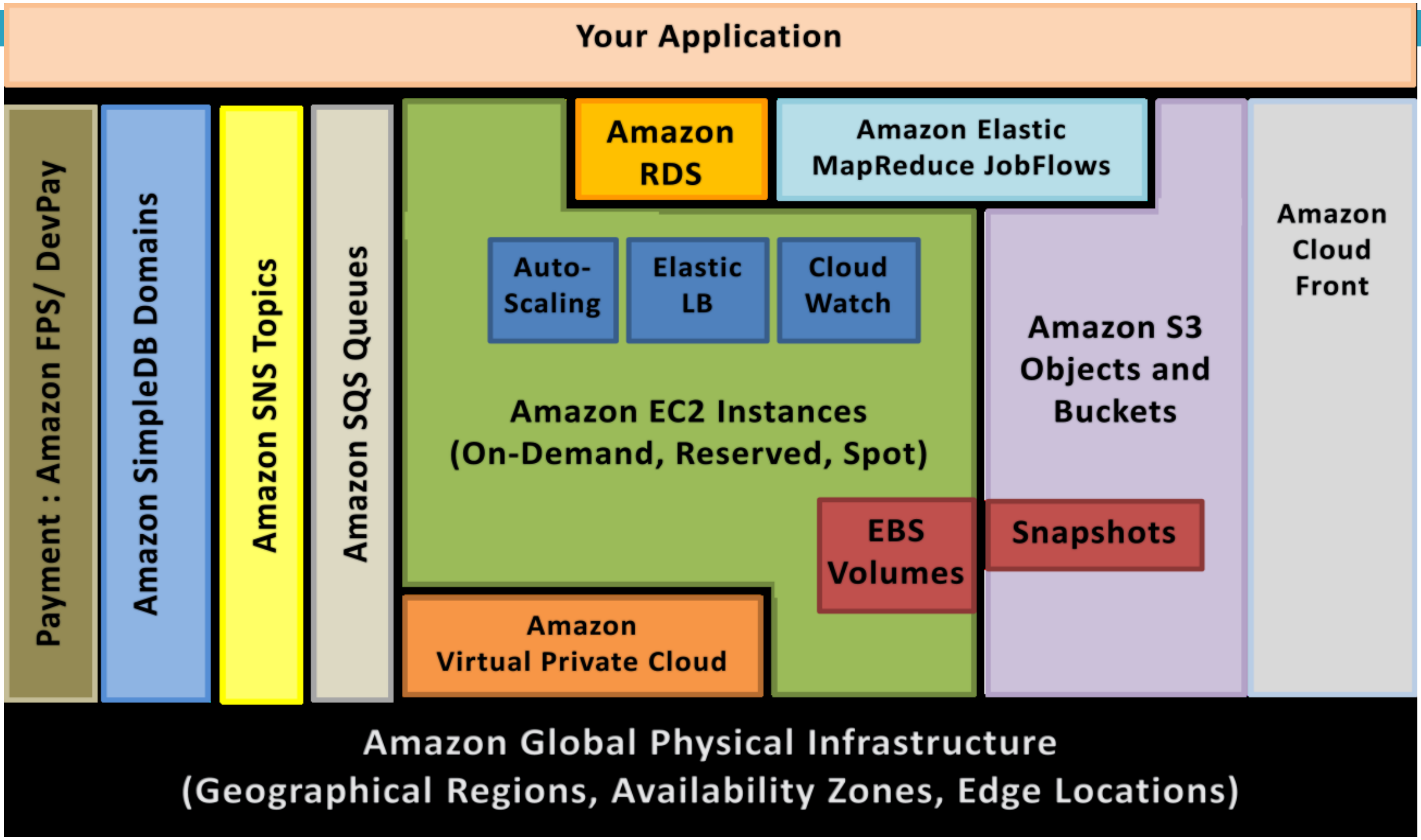
Cloud Computing in a Nutshell

- Next generation of computing after Mainframe, Personal computers, Client-Server and the Web
- New platform & delivery model providing dynamically scalable & often virtualized resources
- No CAPEX, only OPEX
  - 1000 servers for 1 hour = 1 server for 1000 hours!
- Commoditization of IT
- Economies of Scale
- Elasticity – On-Demand

Infrastructure-as-a-Service

- Pay-as-you-go Virtualized Resources – CPU, Storage, Network
- Infrastructure management services & tools
- Application cannot dynamically scale on-demand
- Local Server moved into the cloud – managing, patching, securing, monitoring is still a responsibility
- Extremely flexible
- Very little vendor lock-in
- Examples: Amazon EC2, Terremark vCloud, GoGrid Cloud, Rackspace Cloud

Amazon Web Services


Your Application
Payment : Amazon FPS/ DevPay
Amazon SimpleDB Domains
Amazon SNS Topics
Amazon SQS Queues
Amazon RDS
Amazon Elastic MapReduce JobFlows
Auto-Scaling
Elastic LB
Cloud Watch
Amazon EC2 Instances (On-Demand, Reserved, Spot)
EBS Volumes
Snapshots
Amazon S3 Objects and Buckets
Amazon Cloud Front
Amazon Virtual Private Cloud
Amazon Global Physical Infrastructure (Geographical Regions, Availability Zones, Edge Locations)

Building Blocks of an infrastructure

- Compute – Amazon EC2
- Storage – Amazon S3, EBS, SimpleDB, RDS
- Network - Elastic Load Balancer, Auto Scaling

Introduction to boto

- An integrated interface to current and future infrastructural services offered by Amazon Web Services
- Website: http://boto.cloudhackers.com/
- Source Code: http://github.com/boto
- IRC: #boto on FreeNode
- License: MIT License

How easy is it to use boto?

- easy_install boto
- >>> from boto.ec2.connection import EC2Connection
- >>> conn = EC2Connection('<aws access key>', '<aws secret key>')
- >>> images = conn.get_all_images()
- >>> image = images[0]
- >>> reservation = image.run()
- >>> instance = reservation.instances[0]
- >>> instance.state
- >>> instance.update()

How easy is it to use boto?

- Demo & Explanation

Managing instances

- Starting an instance
- Image.run(
  - instance_type, # server size – m1.large, c1.medium
  - key_name, # SSH Keypair Name to access the machine
  - placement, # Region – us-east-1, us-west-1
  - security_groups # List Firewall rules
  - disable_api_termination # Avoid termination errors
-)

Managing instances

- Listing Instances
  - conn = EC2Connection('<aws access key>', '<aws secret key>')
  - conn.get_all_instances()
- Stopping an instance
  - Instance.stop()
- Terminating an instance
  - Instance.terminate()

Automate Server Creation/Termination

- Sample Script and Demo

Managing Volumes (Elastic Block Storage)

- Create Volumes
  - volume = conn.create_volume(size, zone)
- Attach Volumes
  - volume.attach(instanceid, device)
- Backup Volumes – Create Snapshots
  - conn.create_snapshot(volume.volume_id)
- Detach Volumes
  - volume.detach()
- Delete Volumes
  - volume.delete()
- Create Volumes from snapshots
- volume = conn.create_volume(size, zone, snapshot_id)

Automate backups of server/datastore

- Sample script and Demo

Managing Elastic Load Balancers

- Create Load Balancer
  - conn.create_load_balancer(
  - Name,
  - Zone List,
  - Port Mapping List)
  - eg. conn.create_load_balancer('my_lb', ['us-east-1a', 'us-east-1b'], [(80, 80, 'http'), (443, 8443, 'tcp')])
- Attach Load Balancer
  - elb_conn = ELBConnection(…)
  - elb_conn.get_all_load_balancers()
  - elb.register_instances(instance_id)
- Detach Load Balancer
  - elb.deregister_instances(instance_id)

Automate Load Balancer Setup

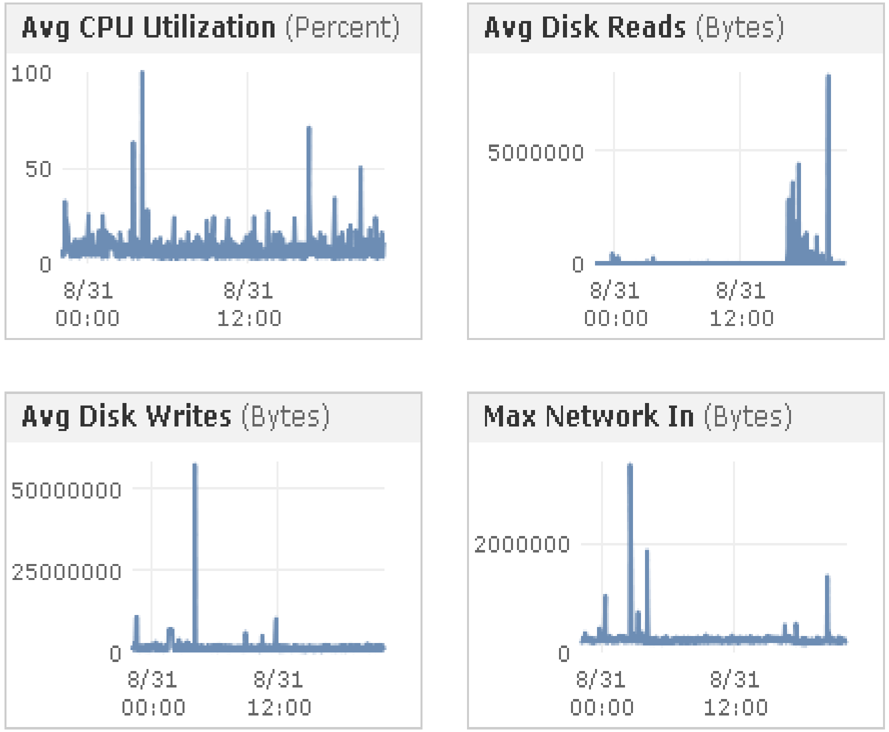
- Sample script and Demo

Automate Load Balancer Setup

- Sample script and Demo

Amazon Cloud Watch

- Monitoring of AWS Cloud resources in real-time
  - Basic 5-min monitoring – free
  - Detailed monitoring at 1 minute interval – charged
- Create Custom Metrics
  - Memory, Free Space
  - Users in my DB
- Set alarms & notifications
- View graphs and statistics



Amazon Cloud Watch

- Monitoring of AWS Cloud resources in real-time
  - Basic 5-min monitoring – free
  - Detailed monitoring at 1 minute interval – charged
- Create Custom Metrics
  - Memory, Free Space
  - Users in my DB
- Set alarms & notifications
- View graphs and statistics

Amazon Cloud Watch – Custom Metrics

- Custom metrics - use case
- Custom metrics using boto
- cw_connect = CloudWatchConnection(
  - aws_access_key,
  - aws_secret_access_key)
- cw_connect.put_metric_data(
  - namespace,
  - metric_name,
  - count,
  - unit="Count")

Putting all the pieces together

- Demo and scripts
  - Provision servers
  - Create snapshots
  - Create AMIs

Coding Precautions

- Retry, retry and retry
- Check for status always
  - Attach
  - Detach
  - Create
  - Delete
- Clean up after any exceptions
  - Servers
  - Volumes

Libraries besides boto

- Apache Libcloud
  - Python library abstracts away cloud provider API
  - A unified interface to the clouds
  - Amazon AWS, Rackspace, Gogrid and IBM
- Non Python libraries
  - Jcloud
  - Deltacloud

Questions and Discussions