NETWORK STRUCTURE & CLOUD COMPUTING PROJECT REPORT

"SPPARD ON CLOUD"

APRIL 2016

Guided By-
Prof. Julian Cooper

By-
Aayush Shah
Dishu Jindal
Palash Kochar
Pawan Matta
Rimple Talati
Saurabh Goyal

Contents

INTRODUCTION

Cloud computing is Internet based development and use of computer technology. It is a paradigm shift where details are abstracted from the users who no longer need knowledge or expertise or control over the technology infrastructure "in the cloud" that supports them. It typically involves the provision of dynamically scalable and often virtualized resources as a service over the Internet.

'Cloud' refers to a distinct IT environment that is designed for the purpose of remotely provisioning scalable and measured IT resources. The term cloud is originated as a metaphor for the Internet. It is important to note the differences between the term 'cloud' and cloud symbol from Internet. There are multiple individual clouds that has a finite boundary and are accessible via the Internet. On one hand, where Internet provides open access to many web based IT resources, a cloud, on other hand typically privately owned offers access to IT resources that is metered.

Cloud computing typically provides 3 types of services: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). These services are available over the Internet to the whole world where the cloud acts as a single point of access for serving all the customers.

Types of Cloud

- Private Cloud – This is solely created for the internal purposes for a single organization. It is typically the first step of the organization to move into the cloud.
- Public Cloud – Multiple organizations rents the clouds services from cloud provider on-demand basis.
- Hybrid Cloud – This is composed of multiple internal or external cloud. For example, a cloud consumer may choose to deploy sensitive data to a private and less sensitive cloud services to a public cloud.
- Community Cloud – It is similar to public cloud except that its access is limited to a specific community of cloud consumers.

REQUIREMENTS

- Entire stack has to be on one either AWS or Azure.
- Java Spring developed login portal secured by self signed certificate
- Backend DB has to be either relational DB or NoSQL with sample data
- Java spring developed use case buttons
    - First button - creation of 3 user login to application and run 2 unique report
    - Second button - creation of 10 user login to application and run 6 unique report
    - Third button - creation of 17 user login to application and run 10 unique report
    - Fourth button - creation of 24 user login to application and run 14 unique report
    - Fifth button - user input fields to allow the ability to enter parameters for how many users login creation and how many reports those users are uniquely are running.
- Proper infrastructure alerts and triggers to allow for auto-scaling of resources to accommodate the additional load in application, network, data storage and usage with your environment

BUSINESS JUSTIIFICATION

To deploy a Spring MVC web application in AWS with the help of MySQL RDS database to scale in/out the server according to user traffic and load and hence avoiding latency and bottlenecks.

SYSTEMS AND DESCRIPTIONS

Platform

- Amazon Web Services

Typically when we need to choose a cloud based platform then only two of them comes in the mind – Microsoft's Azure or Amazon's AWS.

As a student, we had free credits for both of these platforms. So the question arises, which platform will be best suitable for our project?

When we started to research on these two platforms, we observed following comparisons:

Criteria	**AWS**	**Azure**
Instance Families	7	4
Instance Types	38	33
Zones	Yes	-
Ephemeral(Temporary)	Yes	Temporary storage – D Drive
Block Storage	EBS	Page Blobs
Relational DB	RDS	Relational DBs
NoSQL and Big Data	DynamoDB, EMR, Kinesis, Redshift	Windows Azure table, HDInsight
Virtual Network	VPC	VNet
DNS	Route53	-
Pricing	Per hour – rounded trip	Per minute – rounded up commitments

We had used both of them during our assignments and apart from the above comparisons we also felt that the performance speed of Azure is little slow as compared to AWS. So we decided to build our entire project on AWS.

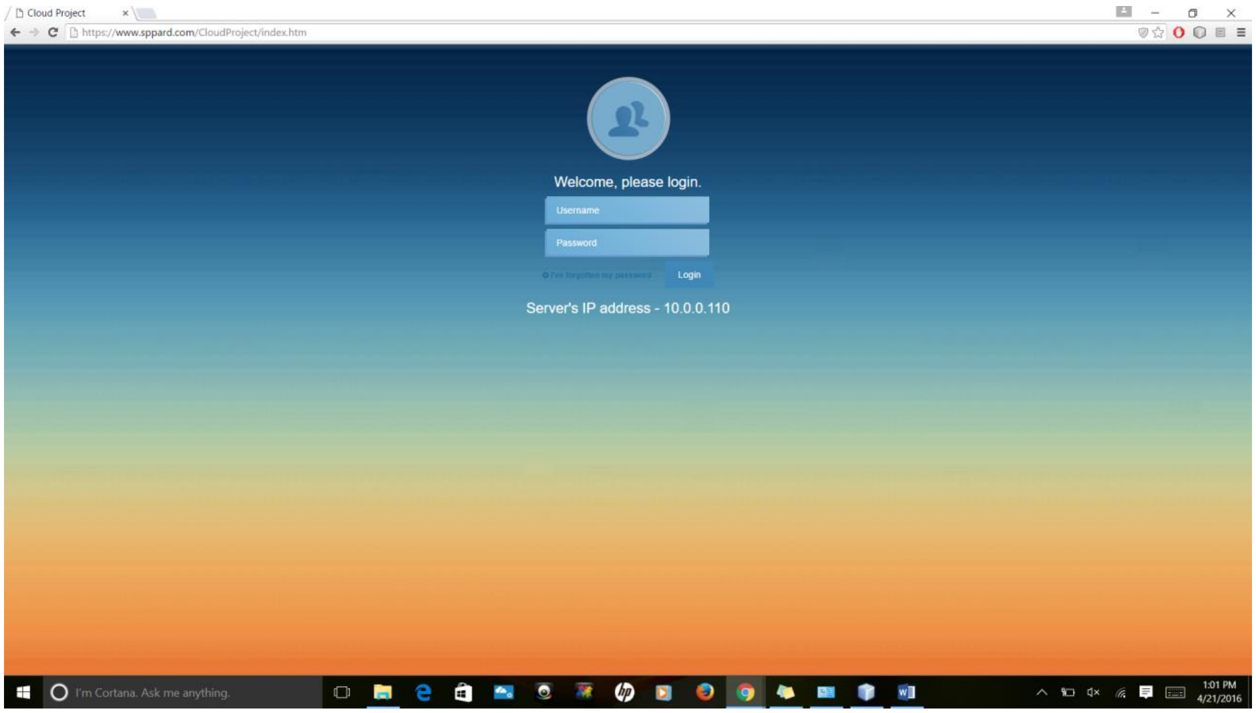
Techniques Used

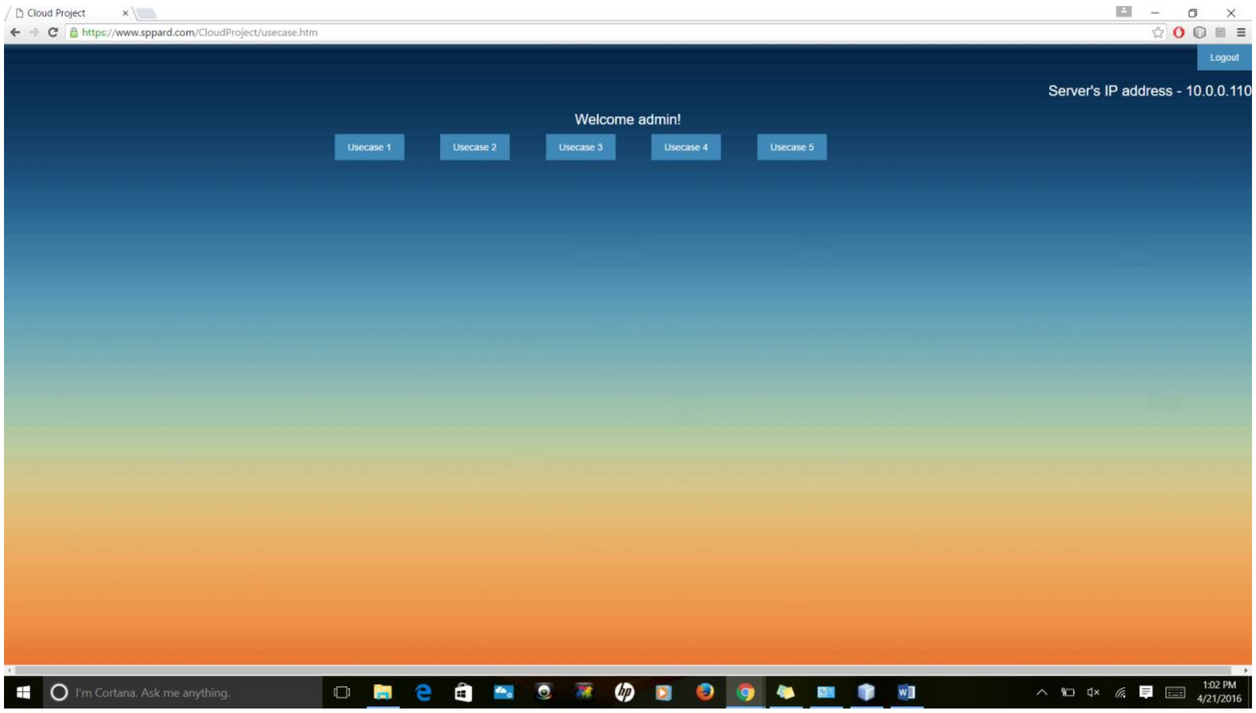
- Elastic Load Balancing
- Auto Scaling
- Cloud Watch
- Health Check

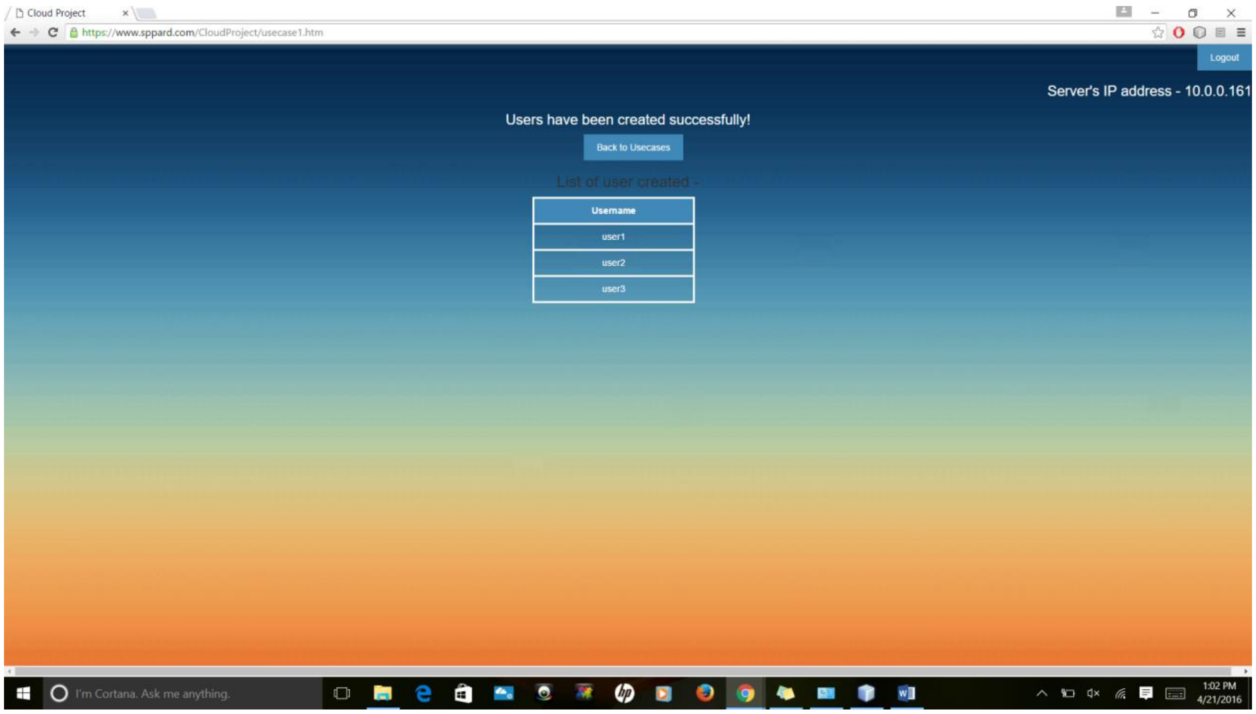
Web Application

- We are creating an E-commerce application https://www.sppard.com/ based on Spring MVC framework
- We are using MySQL with RDS as a database
- We created one table which consists of userID, username, passwords, product_names. The random data was filled by - https://www.devart.com/dbforge/sql/data-generator/

- Size of the dataset is 760MB and it has around 16 million records.
- We have written a Java code which allows to create certain number of users on click of Use Case buttons. We are also storing these created users in the session which shows up in the next page. Besides user creation, we are also generating certain number of unique reports on each Use case button and these reports are automatically saved on S3 bucket in AWS.


Cloud Project
https://www.sppard.com/CloudProject/index.htm
Welcome, please login.
Username
Password
Login
Server's IP address - 10.0.0.110
I'm Cortana. Ask me anything.
1:01 PM
4/21/2016


Cloud Project
https://www.sppard.com/CloudProject/usecase.htm
Logout
Server's IP address - 10.0.0.110
Welcome admin!
Usecase 1
Usecase 2
Usecase 3
Usecase 4
Usecase 5
I'm Cortana. Ask me anything.
1:02 PM
4/21/2016


Cloud Project
https://www.sppard.com/CloudProject/usecase1.htm
Logout
Back to Usecases
I'm Cortana. Ask me anything.
1:02 PM
4/21/2016

Server's IP address - 10.0.0.161

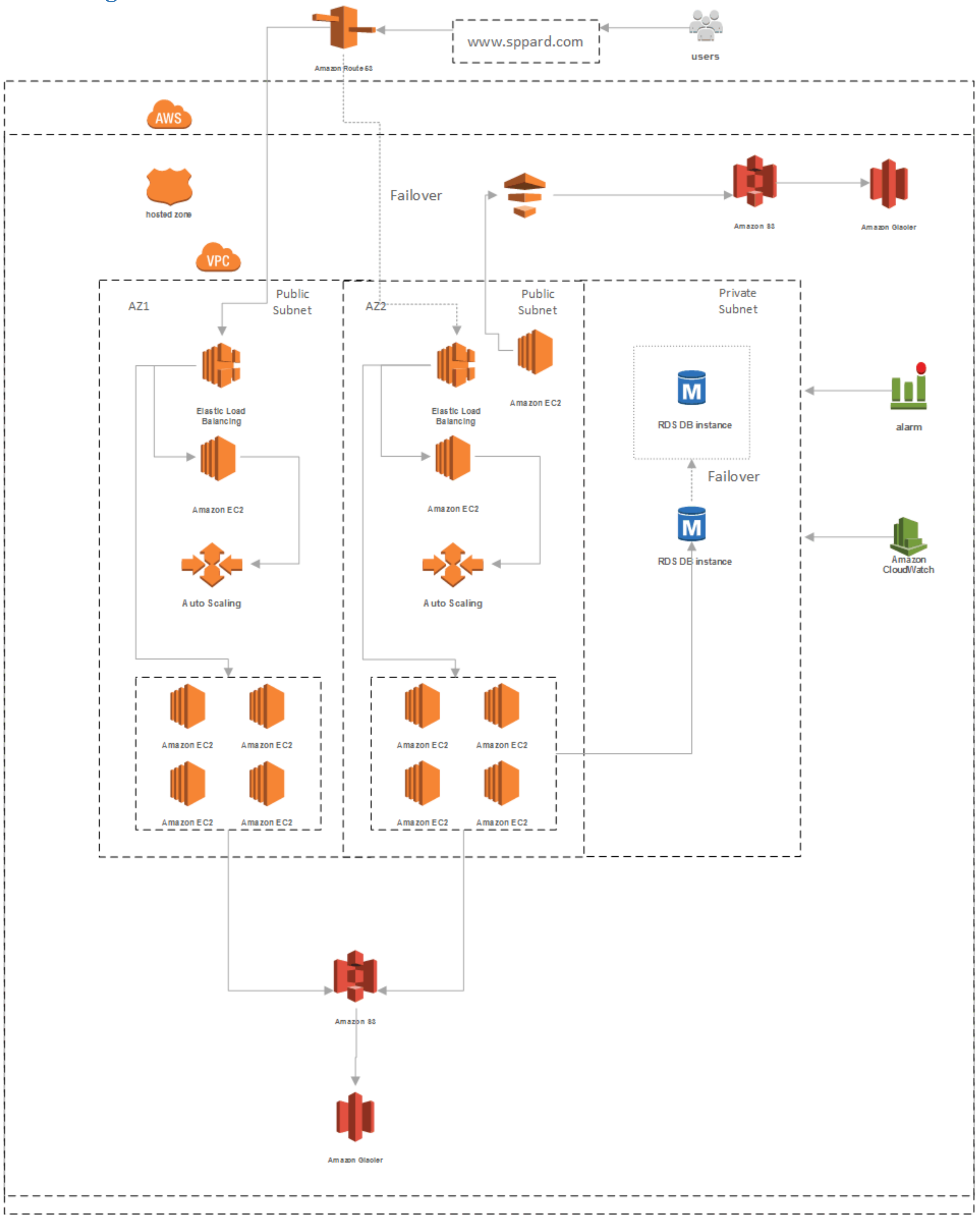
Users have been created successfully!

List of user created -

Username
user1
user2
user3

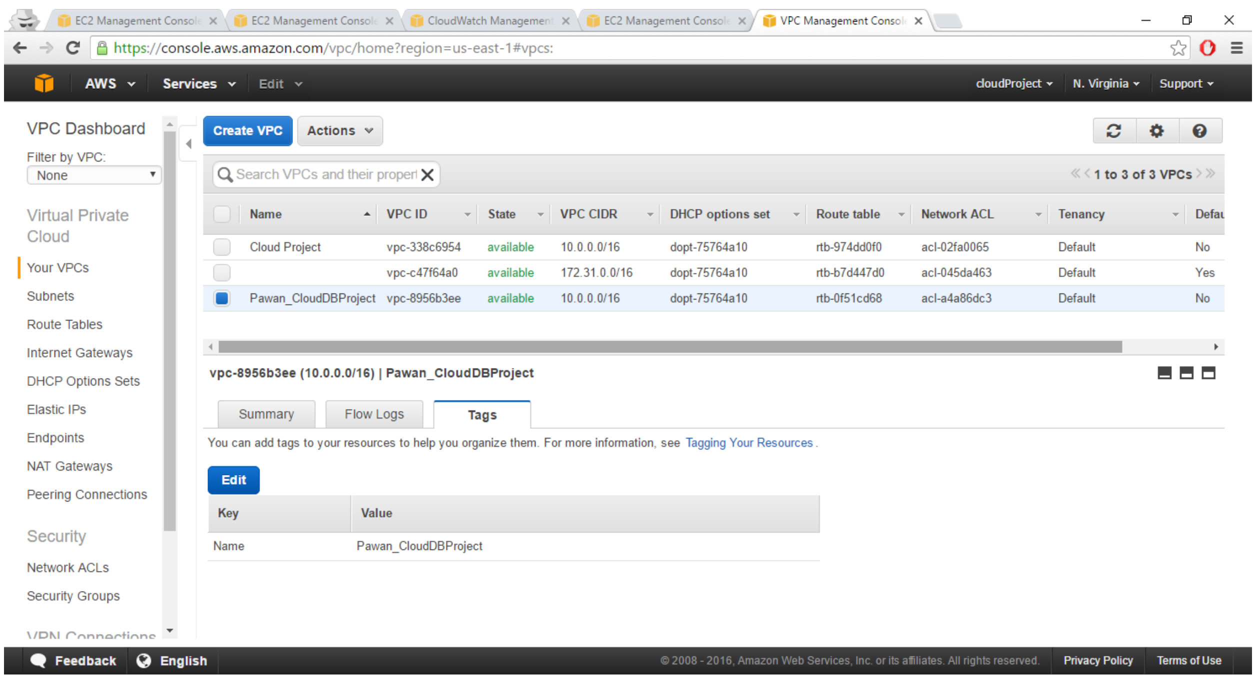
CLOUD ARCHITECTURE

Visio Diagram



Virtual Private Cloud

- Virtual private cloud (VPC) is a virtual network dedicated to our AWS account. It is logically isolated from other virtual networks in the AWS cloud.
- We had created VPC with Public and Private Subnets.
- EC2 instances, RDS instance, Load Balancer, Auto Scaling Group and security groups are inside VPC



	Name	VPC ID	State	VPC CIDR	DHCP options set	Route table	Network ACL	Tenancy	Defau
☐	Cloud Project	vpc-338c6954	available	10.0.0.0/16	dopt-75764a10	rtb-974dd0f0	acl-02fa0065	Default	No
☐		vpc-c47f64a0	available	172.31.0.0/16	dopt-75764a10	rtb-b7d447d0	acl-045da463	Default	Yes
■	Pawan_CloudDBProject	vpc-8956b3ee	available	10.0.0.0/16	dopt-75764a10	rtb-0f51cd68	acl-a4a86dc3	Default	No

vpc-8956b3ee (10.0.0.0/16) | Pawan_CloudDBProject

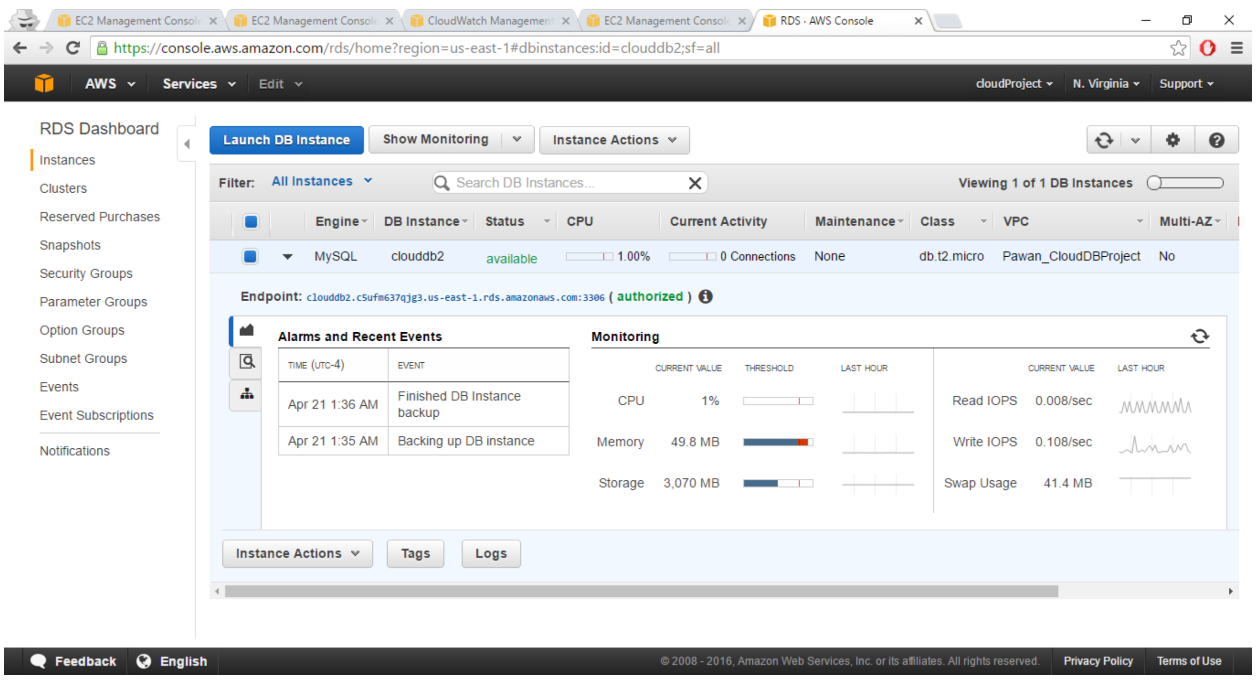
You can add tags to your resources to help you organize them. For more information, see Tagging Your Resources.

Key	Value
Name	Pawan_CloudDBProject

Feedback
English
© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Amazon RDS for MySQL

- Amazon RDS makes it easy to set up, operate, and scale MySQL deployments in the cloud.
- It has following features –
  - Scalable
  - Cost-efficient
  - Resizable hardware capacity
- We had launched the RDS instance in the private subnet of VPC in order to secure the data.
- During configuration we had selected MYSQL, specified DB details and db.t2.micro as DB instance class. On click of configured advanced settings, we selected the VPC that we have created and gave other information.



Engine	DB Instance	Status	CPU	Current Activity	Maintenance	Class	VPC	Multi-AZ
MySQL	clouddb2	available	1.00%	0 Connections	None	db.t2.micro	Pawan_CloudDBProject	No

Endpoint: clouddb2.c5ufm637qjg3.us-east-1.rds.amazonaws.com:3306 (authorized)

Alarms and Recent Events

TIME (UTC-4)	EVENT
Apr 21 1:36 AM	Finished DB Instance backup
Apr 21 1:35 AM	Backing up DB instance

Monitoring

	CURRENT VALUE
CPU	1%
Memory	49.8 MB
Storage	3,070 MB
Read IOPS	0.008/sec
Write IOPS	0.108/sec
Swap Usage	41.4 MB

RDS connection with MySQL workbench

- As the RDS is in private subnet so no one access it from outside. In order to load the data in to RDS, we have created a VPN instance in our VPC through which we can connect the RDS instance.
- In order to create the VPN instance, we had used an AMI from Amazon marketplace and used openVPN to connect to the VPN instance.

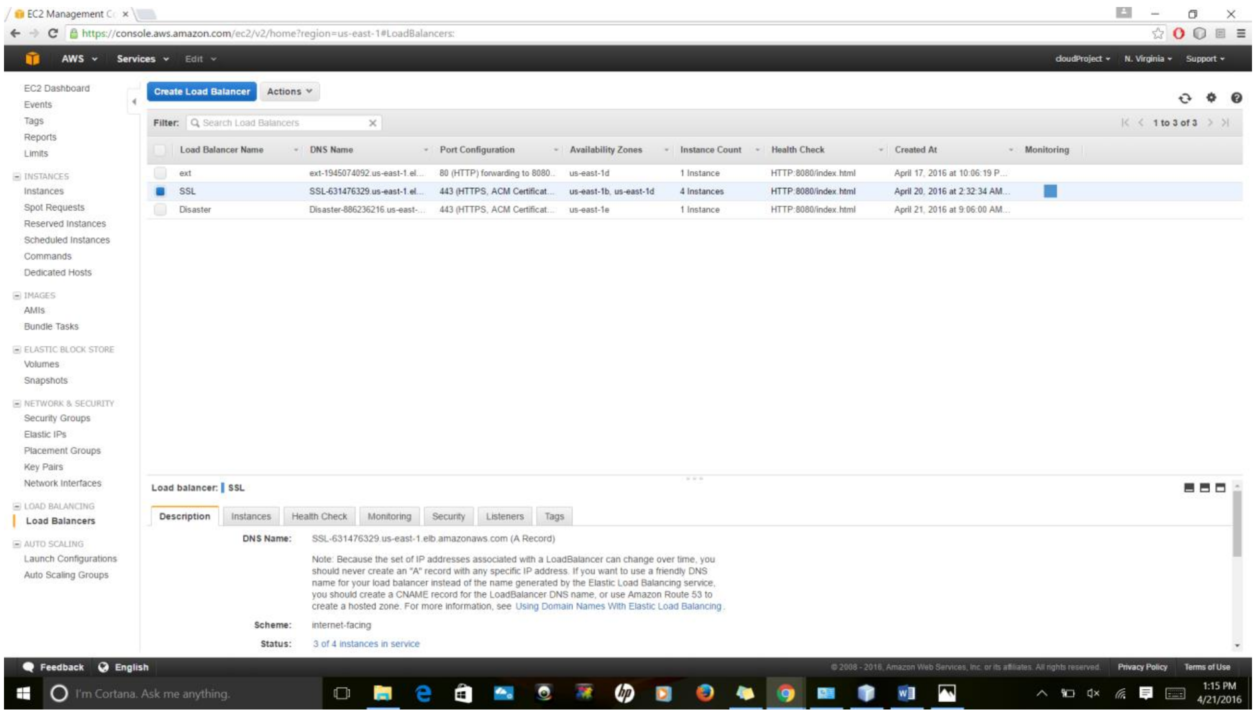
Advantages of VPN –

- VPN offers securely connection to a remote network via the Internet.
- These are very useful for connecting multiple networks together securely.
- VPN encrypts Internet traffic which helps to stymie other people who may be trying to capture your passwords

Elastic Load Balancer

- It is a load balancing solution that automatically scales its request-handling capacity in response to incoming application traffic.
- ELB solutions often provide scalable cloud computing capacity.
- It has following features –
  - Detection of unhealthy instances
  - Spreading instances across healthy channels only.
  - Flexible cipher support.
  - Centralized management of SSL certificates.
- As our application is secured through self-signed certificate, we have to attach SSL certificate in load balancer so that it can forward HTTPS request to our EC2 instances hosting the web application. We need to add listener by:-

- LB Protocol – HTTPS
- LB Port – 443
- Instance Protocol – HTTPS



Load Balancer Name	DNS Name	Port Configuration	Availability Zones	Instance Count	Health Check	Created At	Monitoring
ext	ext-1945074092.us-east-1.el...	80 (HTTP) forwarding to 8080...	us-east-1d	1 Instance	HTTP:8080/index.html	April 17, 2016 at 10:06:19 P...	
SSL	SSL-631476329.us-east-1.el...	443 (HTTPS, ACM Certificat...	us-east-1b, us-east-1d	4 Instances	HTTP:8080/index.html	April 20, 2016 at 2:32:34 AM...	
Disaster	Disaster-886236216.us-east-...	443 (HTTPS, ACM Certificat...	us-east-1e	1 Instance	HTTP:8080/index.html	April 21, 2016 at 9:06:00 AM...	

Load balancer: SSL

DNS Name: SSL-631476329.us-east-1.elb.amazonaws.com (A Record)

Note: Because the set of IP addresses associated with a LoadBalancer can change over time, you should never create an "A" record with any specific IP address. If you want to use a friendly DNS name for your load balancer instead of the name generated by the Elastic Load Balancing service, you should create a CNAME record for the LoadBalancer DNS name, or use Amazon Route 53 to create a hosted zone. For more information, see Using Domain Names With Elastic Load Balancing.

Scheme: internet-facing

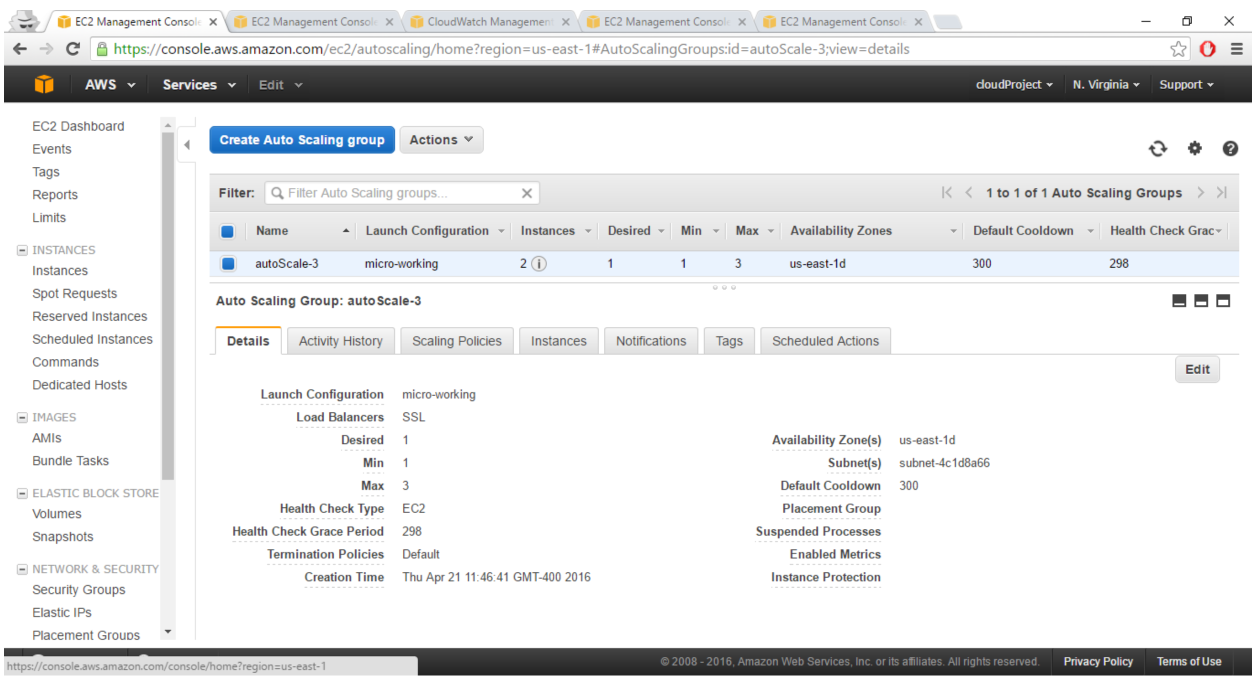
Status: 3 of 4 instances in service

Web Application Security

- To secure our application, we have used self-signed certificate.
- We had uncommented the connector port – 8443 and added keystore file and keystore pass so that it accepts HTTPS requests.
- We had done changes to redirect 8080 to 8443 in server.xml and added security constraint in application's web.xml file.

Auto Scaling

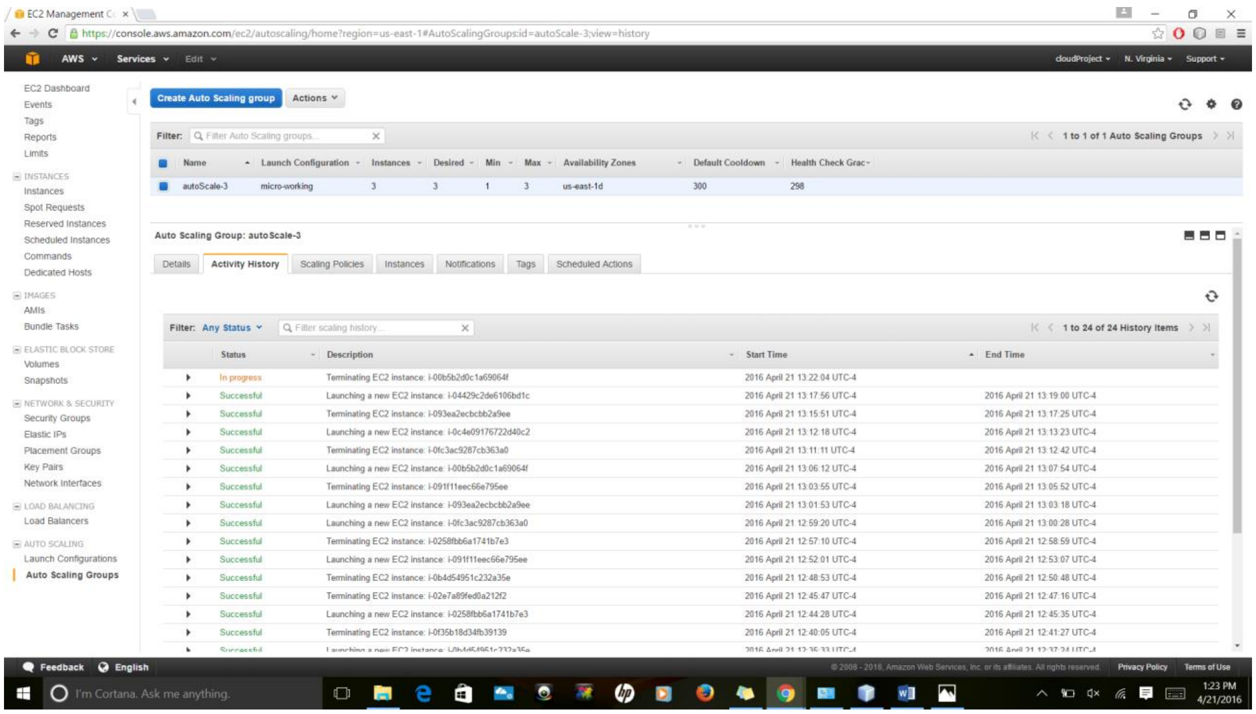
- Auto Scaling allows us to scale our Amazon EC2 capacity up or down automatically according to conditions we define.
- In the launch configuration, we have used pre-configured AMI of EC2 instance that had tomcat server and war file of our application.
- We are using m3.large instance class for the EC2 instances that will be created.
- Next, we added security group that will control the traffic of our EC2 instances.
- Now, we had created auto scaling group. We set the minimum ass 1 and maximum as 3 instances.
- We have used Cloud Watch to monitor our server instances and at certain event like CPU utilization above or below the specified limit, it will automatically launch or decrease instances based on the AMI template that we define in launch configurations. These instances are attached to the Load Balancer which have been added in this ASG.



Auto Scaling Group: autoScale-3

Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
autoScale-3	micro-working	2	1	1	3	us-east-1d	300	298

Launch Configuration	micro-working		
Load Balancers	SSL		
Desired	1	Availability Zone(s)	us-east-1d
Min	1	Subnet(s)	subnet-4c1d8a66
Max	3	Default Cooldown	300
Health Check Type	EC2	Placement Group	
Health Check Grace Period	298	Suspended Processes	
Termination Policies	Default	Enabled Metrics	
Creation Time	Thu Apr 21 11:46:41 GMT-400 2016	Instance Protection	



Auto Scaling Group: autoScale-3

Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
autoScale-3	micro-working	3	3	1	3	us-east-1d	300	298

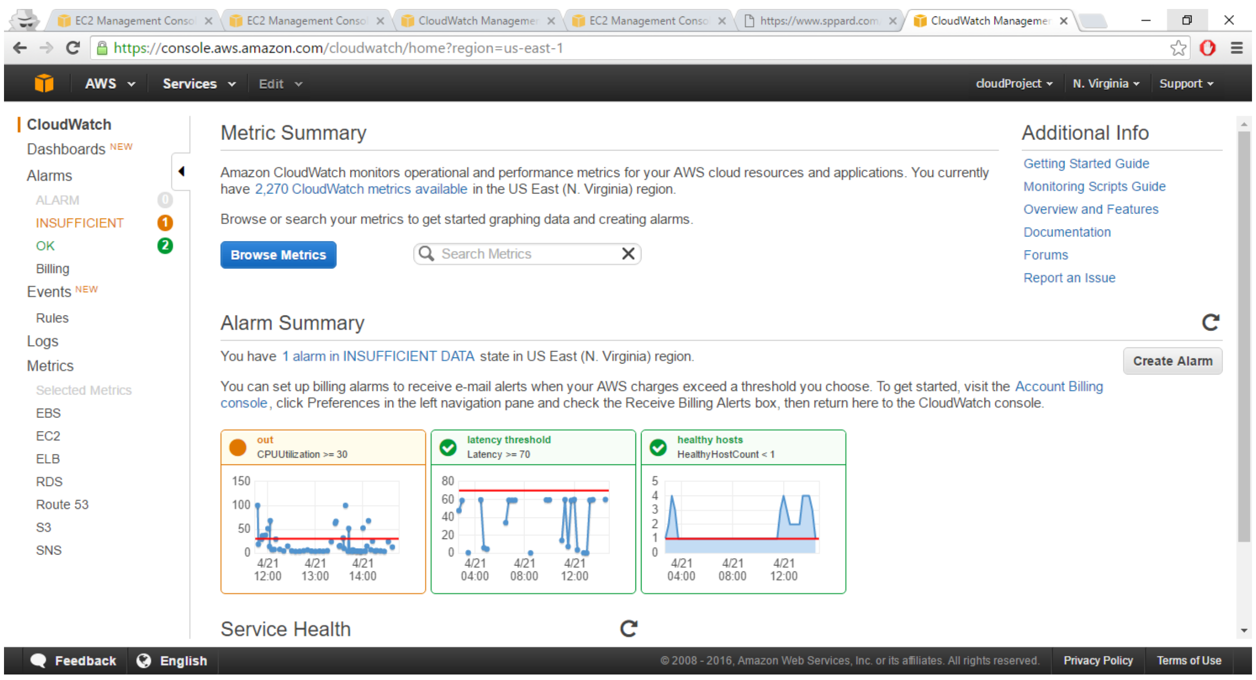
1 to 24 of 24 History Items

Status	Description	Start Time	End Time
In progress	Terminating EC2 instance: i-00b5b2d0c1a69064f	2016 April 21 13:22:04 UTC-4	
Successful	Launching a new EC2 instance: i-04429c2de6106bd1c	2016 April 21 13:17:56 UTC-4	2016 April 21 13:19:00 UTC-4
Successful	Terminating EC2 instance: i-093ea2ecbcbb2a9ee	2016 April 21 13:15:51 UTC-4	2016 April 21 13:17:25 UTC-4
Successful	Launching a new EC2 instance: i-0c4e09176722d40c2	2016 April 21 13:12:18 UTC-4	2016 April 21 13:13:23 UTC-4
Successful	Terminating EC2 instance: i-0fc3ac9287cb363a0	2016 April 21 13:11:11 UTC-4	2016 April 21 13:12:42 UTC-4
Successful	Launching a new EC2 instance: i-00b5b2d0c1a69064f	2016 April 21 13:06:12 UTC-4	2016 April 21 13:07:54 UTC-4
Successful	Terminating EC2 instance: i-091f11eec66e795ee	2016 April 21 13:03:55 UTC-4	2016 April 21 13:05:52 UTC-4
Successful	Launching a new EC2 instance: i-093ea2ecbcbb2a9ee	2016 April 21 13:01:53 UTC-4	2016 April 21 13:03:18 UTC-4
Successful	Launching a new EC2 instance: i-0fc3ac9287cb363a0	2016 April 21 12:59:20 UTC-4	2016 April 21 13:00:28 UTC-4
Successful	Terminating EC2 instance: i-0258fbb6a1741b7e3	2016 April 21 12:57:10 UTC-4	2016 April 21 12:58:59 UTC-4
Successful	Launching a new EC2 instance: i-091f11eec66e795ee	2016 April 21 12:52:01 UTC-4	2016 April 21 12:53:07 UTC-4
Successful	Terminating EC2 instance: i-0b4d54951c232a35e	2016 April 21 12:48:53 UTC-4	2016 April 21 12:50:48 UTC-4
Successful	Terminating EC2 instance: i-02e7a89fed0a212f2	2016 April 21 12:45:47 UTC-4	2016 April 21 12:47:16 UTC-4
Successful	Launching a new EC2 instance: i-0258fbb6a1741b7e3	2016 April 21 12:44:28 UTC-4	2016 April 21 12:45:35 UTC-4
Successful	Terminating EC2 instance: i-0f35b18d34fb39139	2016 April 21 12:40:05 UTC-4	2016 April 21 12:41:27 UTC-4

Cloud Watch

- Amazon CloudWatch is a component of Amazon Web Services (AWS) that provides monitoring for AWS resources such as Amazon EC2 instances, Load Balancers and Amazon RDS instances and the customer applications running on the Amazon infrastructure.

- We had set the Alarm to monitor the CPU usage of our EC2 instances and then used this data to determine whether to launch additional instances to handle increased load
- An alarm can have three possible states:
    - OK—The metric is within the defined threshold
    - ALARM—The metric is outside of the defined threshold
    - INSUFFICIENT_DATA—The alarm has just started, the metric is not available, or not enough data is available for the metric to determine the alarm state
- We had set 1 alarm to trigger:
    - If the CPU utilization is greater than 30% for 1 minute then ASG will scale out by 1 instance.



Metric Summary

Amazon CloudWatch monitors operational and performance metrics for your AWS cloud resources and applications. You currently have 2,270 CloudWatch metrics available in the US East (N. Virginia) region.

Browse or search your metrics to get started graphing data and creating alarms.

Additional Info

Alarm Summary

You have 1 alarm in INSUFFICIENT DATA state in US East (N. Virginia) region.

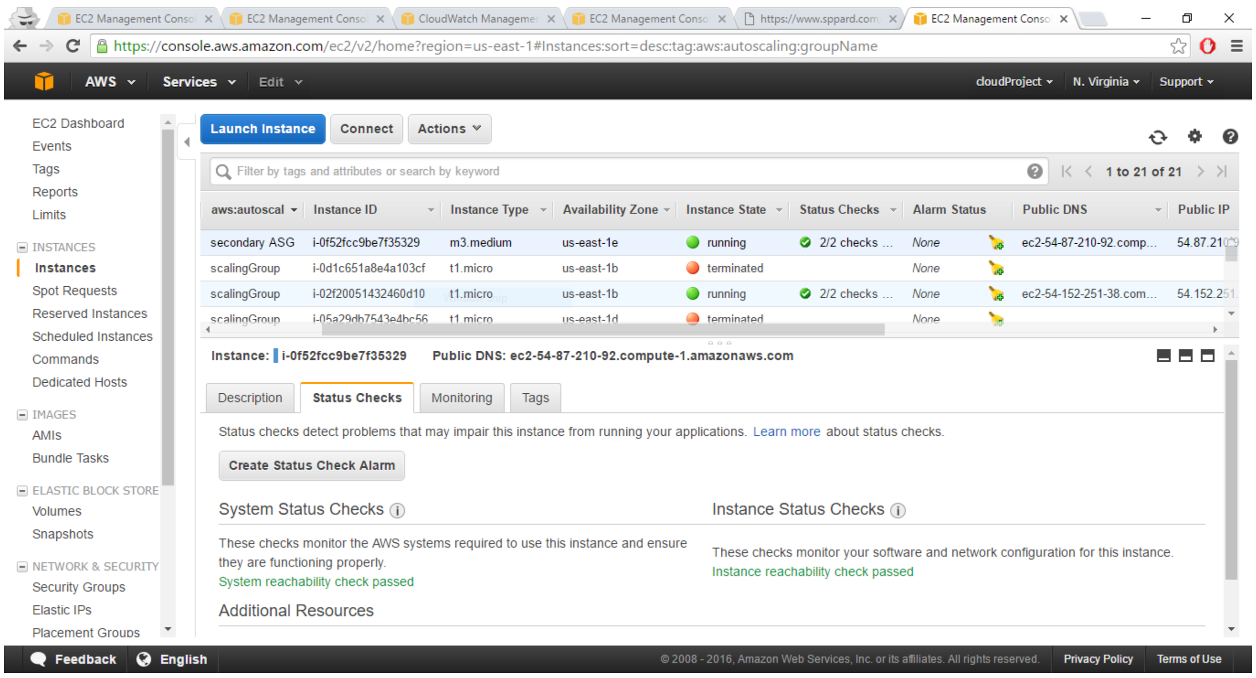
You can set up billing alarms to receive e-mail alerts when your AWS charges exceed a threshold you choose. To get started, visit the Account Billing console, click Preferences in the left navigation pane and check the Receive Billing Alerts box, then return here to the CloudWatch console.

Service Health

© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Launching an EC2 instance

- First we launched an EC2 instance, installed TOMCAT and deployed the war file of our application and created its AMI. This AMI will be used by ASG to launch new instances.


https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=desc:tag:aws:autoscaling:groupName

1 to 21 of 21

aws:autoscal	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
secondary ASG	i-0f52fcc9be7f35329	m3.medium	us-east-1e	running	2/2 checks ...	None	ec2-54-87-210-92.comp...	54.87.210
scalingGroup	i-0d1c651a8e4a103cf	t1.micro	us-east-1b	terminated		None		
scalingGroup	i-02f20051432460d10	t1.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-152-251-38.com...	54.152.251
scalingGroup	i-05a29db7543e4bc56	t1.micro	us-east-1d	terminated		None		

Instance: i-0f52fcc9be7f35329 Public DNS: ec2-54-87-210-92.compute-1.amazonaws.com

Status checks detect problems that may impair this instance from running your applications. Learn more about status checks.

System Status Checks

These checks monitor the AWS systems required to use this instance and ensure they are functioning properly.

System reachability check passed

Instance Status Checks

These checks monitor your software and network configuration for this instance.

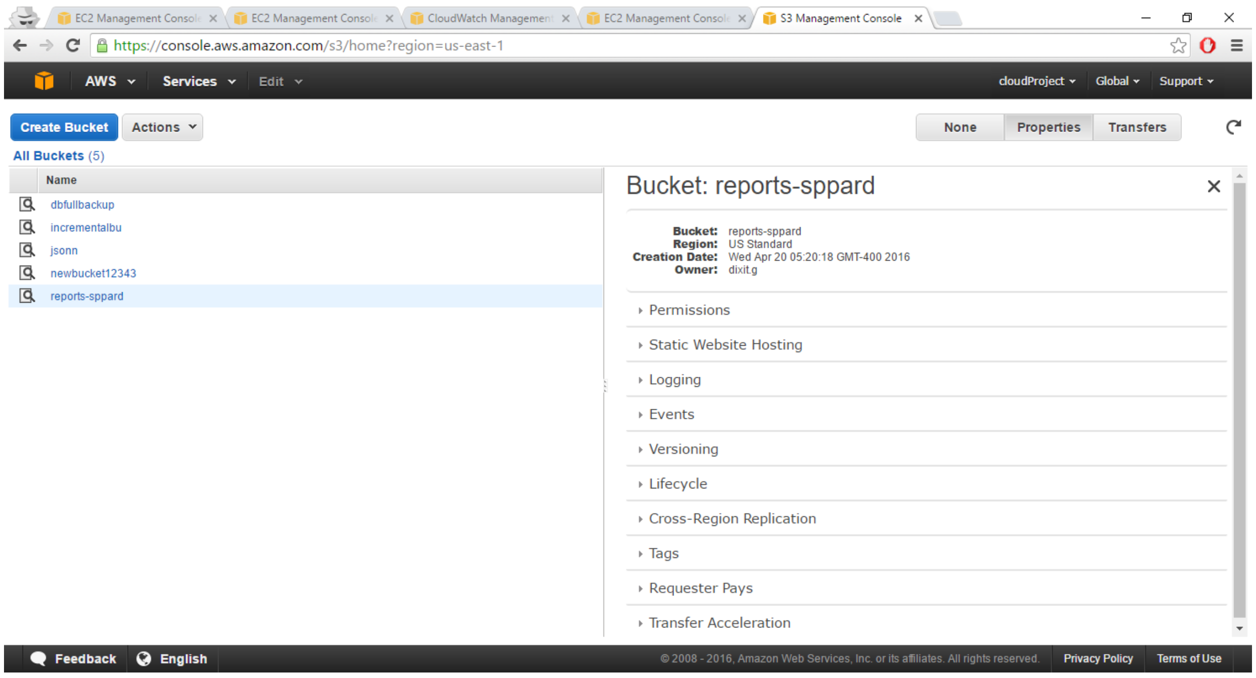
Instance reachability check passed

Additional Resources

© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Simple Storage Service (S3)

- It is an online file storage web service.
- We have used S3 to directly save the generated reports on click of each Use case button.


https://console.aws.amazon.com/s3/home?region=us-east-1

All Buckets (5)

Name
dbfullbackup
incrementalbu
jsonn
newbucket12343
reports-sppard

Bucket: reports-sppard

Bucket:	reports-sppard
Region:	US Standard
Creation Date:	Wed Apr 20 05:20:18 GMT-400 2016
Owner:	dixit.g

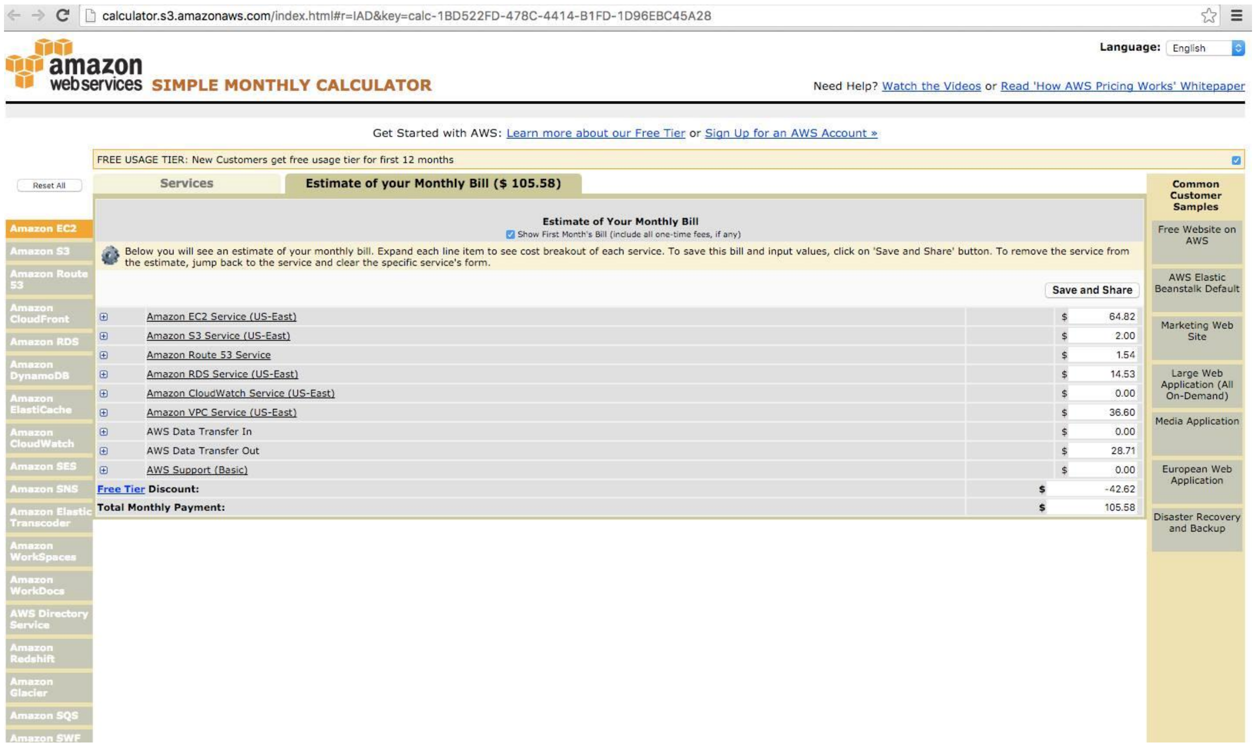
© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Cloud Pricing Model for the next 3-6 months

It turns out that estimate cost of ‘sppard’ application is $59.98 per month which brings in $359.88 for 6 months.

Link to Billing Calculator:

http://calculator.s3.amazonaws.com/index.html#r=IAD&key=calc-1BD522FD-478C-4414-B1FD-1D96EBC45A28



calculator.s3.amazonaws.com/index.html#r=IAD&key=calc-1BD522FD-478C-4414-B1FD-1D96EBC45A28
amazon web services SIMPLE MONTHLY CALCULATOR
Language: English
Need Help? Watch the Videos or Read 'How AWS Pricing Works' Whitepaper
Get Started with AWS: Learn more about our Free Tier or Sign Up for an AWS Account »
Reset All
Services
Estimate of your Monthly Bill ($ 105.58)
Amazon EC2
Amazon S3
Amazon Route 53
Amazon CloudFront
Amazon RDS
Amazon DynamoDB
Amazon ElastiCache
Amazon CloudWatch
Amazon SES
Amazon SNS
Amazon Elastic Transcoder
Amazon WorkSpaces
Amazon WorkDocs
AWS Directory Service
Amazon Redshift
Amazon Glacier
Amazon SQS
Amazon SWF
Common Customer Samples
Free Website on AWS
AWS Elastic Beanstalk Default
Marketing Web Site
Large Web Application (All On-Demand)
Media Application
European Web Application
Disaster Recovery and Backup
Save and Share

FREE USAGE TIER: New Customers get free usage tier for first 12 months

Estimate of Your Monthly Bill

Show First Month's Bill (include all one-time fees, if any)

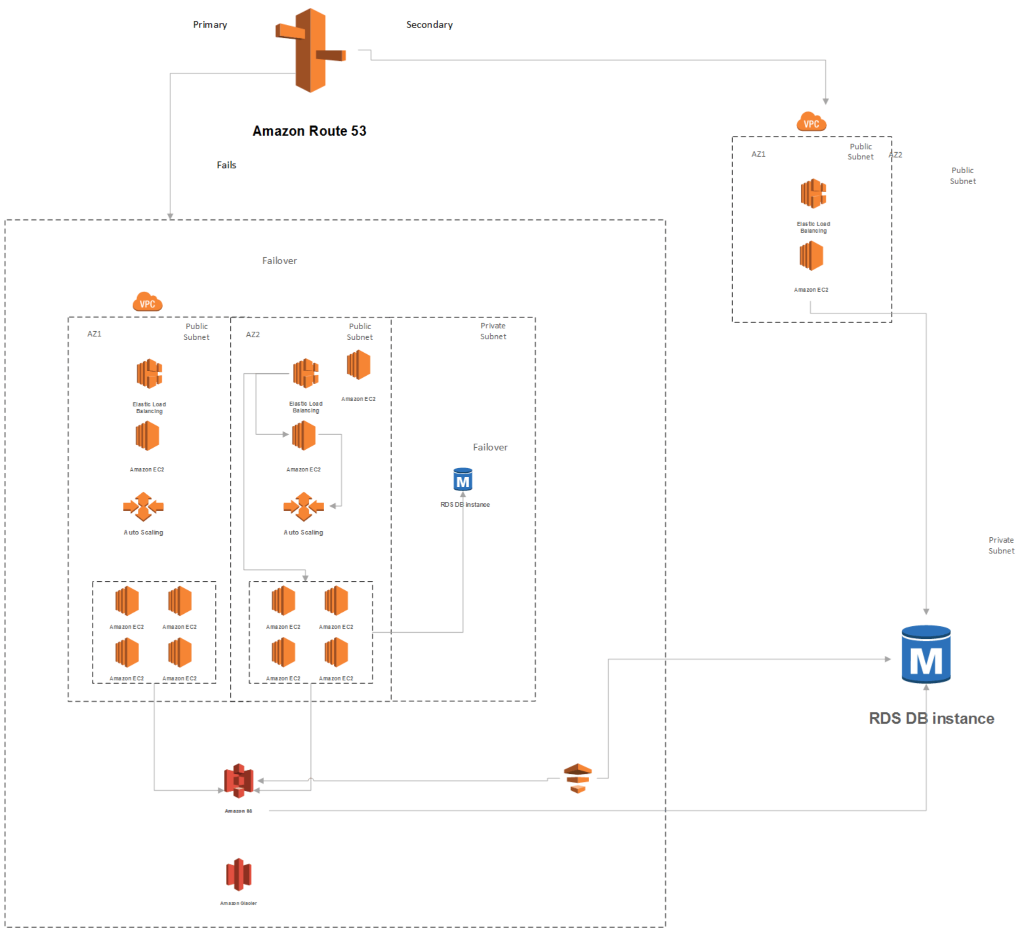
Below you will see an estimate of your monthly bill. Expand each line item to see cost breakout of each service. To save this bill and input values, click on 'Save and Share' button. To remove the service from the estimate, jump back to the service and clear the specific service's form.

Amazon EC2 Service (US-East)	$	64.82
Amazon S3 Service (US-East)	$	2.00
Amazon Route 53 Service	$	1.54
Amazon RDS Service (US-East)	$	14.53
Amazon CloudWatch Service (US-East)	$	0.00
Amazon VPC Service (US-East)	$	36.60
AWS Data Transfer In	$	0.00
AWS Data Transfer Out	$	28.71
AWS Support (Basic)	$	0.00
Free Tier Discount:	**$**	-42.62
Total Monthly Payment:	**$**	105.58

Disaster Recovery Procedures

Disaster Recovery is a process in which if a primary network or a dedicate server in a zone or a zone in a whole collapses or crashes, there should be a procedure in which we try to bring the whole consistent system as a whole.

Until the system is down, we point the Route53 on another system which is nowhere connected to the earlier primary, system. The entire load which was directed to the primary network or system gets redirected to the new system/zone or the database



Steps for Disaster Recovery Plan

1. First we would be taking the entire backup on a weekly basis and saving that into an S3 bucket using an data pipeline
2. We would then take a incremental backup and then again store that into the S3 bucket with a different name
3. We would create a New SQL which is consistent using this CSV file or utilize the RDS instance which is already available since it's a multi -az

References

https://aws.amazon.com/documentation/

http://searchaws.techtarget.com/

http://whatiscloud.com/