Architect Proposal for assignment #2

Zhaolei 2018-05-28

# Background and assumptions

At the beginning, startup company may have not enough experiences and budget on AWS. They want Invest iteratively.

Hourly downtime is acceptable at the beginning. And finally they want a manageable, secure, scalable, high performance, efficient, elastic, highly available, fault tolerant and recoverable architecture that allows the them to organically grow.

# Architecture proposal

I will propose a road map to resolve your problems because it can be implemented iteratively and more realistic. And the final solution architecture diagrams are steps 5 and 6.

If you have enough experiences and budget, you can skip all the previous steps and directly go to the final architecture.

You are always able to controlling traffic to virtual machines with security groups.

The other environment can be the same as the production environment but use the appropriate instances cloned by OpsWorks or CloudFormation. And you can use VPC and subnet to separate them.

Follow the instructions below to migrate to AWS step by step:

## Migrate to AWS with minimal workload and cost

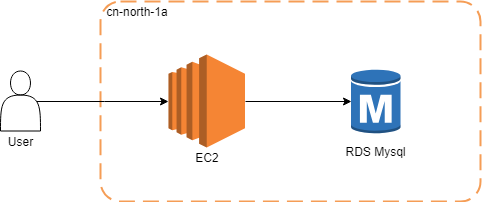
|  |  |  |
| --- | --- | --- |
| Options | Minimize | CDN powered |
| architecture diagram | C:\Users\DELL\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\7A8F09EC.tmp | C:\Users\DELL\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\C221E0F0.tmp |
| Benefits | * Minimal workload and cost * Higher availability powered by AWS EC2 | * Very low latency by CloudFront(CDN) |

It is best to use [Elastic IP](https://ap-southeast-1.console.aws.amazon.com/ec2/v2/home?region=ap-southeast-1#Addresses:sort=PublicIp)(EIP) binding to the EC2 instance so you don’t need to change IP after instance replacing. You can have one Elastic IP (EIP) address associated with a running instance at no charge.

We can use CloudWatch + CloudFormation to do the Disaster Recovery. But the following solutions is recommended because they are easier and more powerful.

You can also add a SPA layer to reduce network traffic and increase app respond speed. But it may increase the initial load speed. You can use S3 to host the static web content.

## Double performance and throughput by decouple app server and DB (use RDS and one more ec2 instance)



The [Amazon Relational Database Service](https://ap-southeast-1.console.aws.amazon.com/rds/home?region=ap-southeast-1) (RDS) offers ready-to-use relational databases. Automatically backup and updated database version, easily scale.

Multi-AZ is not recommended at first because it is expensive and you can enable it at any time with a few mouse clicks (AWS will do the hard work for you, like Master-slave config).

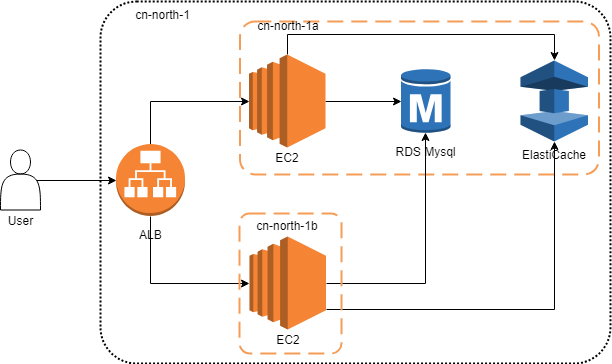
You can automate configuration management for Disaster Recovery (self-healing) and productivity by [Elastic Beanstalk](https://ap-southeast-1.console.aws.amazon.com/elasticbeanstalk/home?region=ap-southeast-1#/welcome) or [OpsWorks Stacks](https://console.aws.amazon.com/opsworks/landing/choice?region=ap-southeast-1):

|  |  |  |
| --- | --- | --- |
| Options | [Elastic Beanstalk](https://ap-southeast-1.console.aws.amazon.com/elasticbeanstalk/home?region=ap-southeast-1#/welcome) | [OpsWorks Stacks](https://console.aws.amazon.com/opsworks/landing/choice?region=ap-southeast-1) |
|  | C:\Users\DELL\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\DE945AA0.tmp | C:\Users\DELL\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\836053B9.tmp |
| Characteristic | Simplest | Flexible |
| Use when | Usable | Elastic Beanstalk cannot fit your application because of some customization requirements. |

Both tools can be used on GUI, which have clearly guide and not need to read lots of documents.

## Increase throughput and availability of App server by load balancing

Powered by Elastic Beanstalk / OpsWorks, we can easily double throughput and availability by adding a peer server managed by an [ELB/ALB](https://ap-southeast-1.console.aws.amazon.com/ec2/v2/home?region=ap-southeast-1#LoadBalancers:sort=loadBalancerName):



[Elastic Load Balancing](https://ap-southeast-1.console.aws.amazon.com/ec2/v2/home?region=ap-southeast-1#LoadBalancers:sort=loadBalancerName) can recreate your server by cooperating with Beanstalk / OpsWorks when the health check is failed.

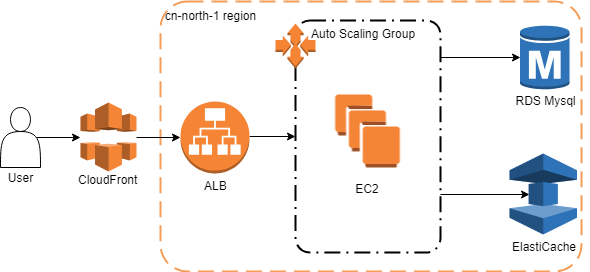
In OpsWorks stack, if you want more flexibility you can try Time-based and Load-based Instances to manage throughput in a more flexible way. It applicable to the situations that you can predict your load.

[ElastiCache](https://ap-southeast-1.console.aws.amazon.com/elasticache/home?region=ap-southeast-1) is a Redis web service. First, refactor your application into stateless by persisting states to [ElastiCache](https://ap-southeast-1.console.aws.amazon.com/elasticache/home?region=ap-southeast-1). Then you can distribute traffic to peer servers.

We recommend to go with the modern Application(ALB) or Network Load Balancer because they are in most cases more cost efficient and more feature rich. But ALB may not well integration with the OpsWork stacks. If OpsWork cannot find ALB, you can use classic ELB.

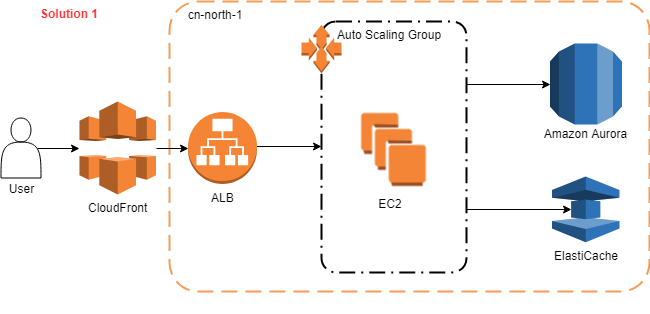
## Make app servers unlimited flexibility by auto scaling

When your applications are stateless, [Auto Scaling](https://amazonaws-china.com/autoscaling/) can ensures you have the correct number of EC2 instances available to handle your application load. You create collections of EC2 instances (called Auto Scaling groups), specify desired instance ranges for them, and create scaling policies that define when instances are provisioned or removed from the group:



Auto-scaling is not by default available on OpsWorks, and there is no build in way to have an auto-scaling group associated with your OpsWorks stack, but it's possible with a bit of work. Read about it [here](https://aws.amazon.com/blogs/devops/auto-scaling-aws-opsworks-instances/).

## Make MySQL RDS unlimited storage and flexibility by Aurora



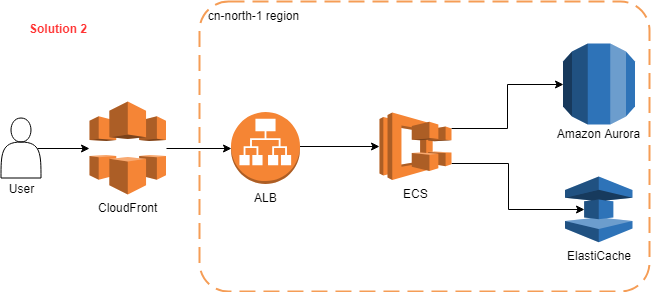
Solution 1

[Amazon Aurora](https://ap-southeast-1.console.aws.amazon.com/rds/home?region=ap-southeast-1) is a MySQL- and PostgreSQL-compatible enterprise-class database, starting at <$1/day. Aurora supports up to 64TB of auto-scaling storage capacity, 6-way replication across three availability zones, and 15 low-latency read replicas. It is less cost and complexity than multi-AZ RDS.

Enable ElastiCache’s Cluster and Multi-AZ gain unlimited storage and availability of cache.

## Containerization by ECS (or EKS)

You can use Amazon Elastic Container Service ([ECS](https://ap-southeast-1.console.aws.amazon.com/ecs/home?region=ap-southeast-1#/clusters)) or Amazon Elastic Container Service for Kubernetes ([EKS)](https://amazonaws-china.com/cn/eks/) to improve the efficiency of resources usage:



Solution 2

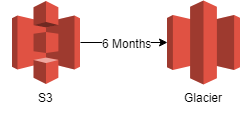
This architecture applicable to the situations that you budget is limited and don’t afraid of the challenge of using cutting-edge technology. You can get the best ROI and less vendor lock-in from the solution. But it may significantly increase your architectural complexity and initial time cost.

You can read a post [here](https://amazonaws-china.com/cn/blogs/compute/automatic-scaling-with-amazon-ecs/) to config auto-scaling for your ECS cluster.

## Others

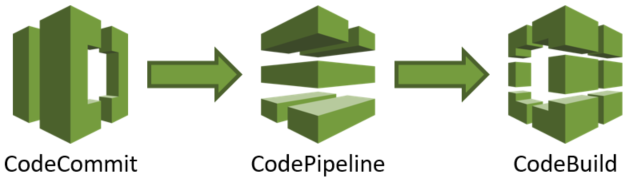
### Archival strategy for inactive objects greater than 6 months with Glacier

Backup or structured data stored in S3 can be configured by a lifecycle rule to move objects to Glacier automatically:



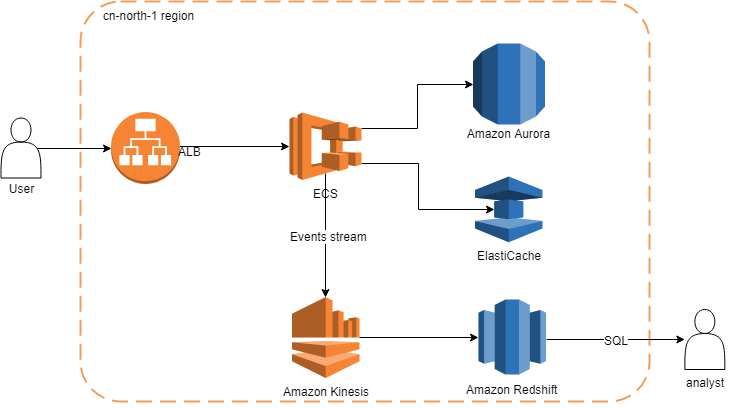
|  |  |  |
| --- | --- | --- |
|  | S3 | Glacier |
| Storage Costs for a GB per month in US East (N. Virginia) | 0.023 USD | 0.004 USD |
| Costs for inserting data | Low | High |
| Costs for retrieving data | Low | High |
| Accessibility | Immediate upon request | One minute to twelve hours after request. Faster retrieval is more expensive. |
| Durability | Designed for annual durability of 99.999999999% | Designed for annual durability of 99.999999999% |

### Continue delivery



You can implement Continue Delivery pipeline by AWS developer tools such as [CodeCommit](https://ap-southeast-1.console.aws.amazon.com/codecommit/home?region=ap-southeast-1#/introduction)(Git), [CodePipeline](https://ap-southeast-1.console.aws.amazon.com/codepipeline/home?region=ap-southeast-1)(CD), [CodeDeploy](https://ap-southeast-1.console.aws.amazon.com/codedeploy/home?region=ap-southeast-1), [CodeBuild](https://ap-southeast-1.console.aws.amazon.com/codebuild/home?region=ap-southeast-1)(CI).

## OLAP



A [Kinesis](https://ap-southeast-1.console.aws.amazon.com/kinesis/home?region=ap-southeast-1#/intro) stream is an ordered sequence of data records like kafka. It can be used for Asynchronous decoupling for your system, which is also necessary for Microservices architecture.

You can use Lambda or EMR (spark) to do ETL if necessary. And if you don’t want the cost of a data warehouse, you can simply store the data to S3 and use Athena to analyze them.

With all preparations above, you can easily migrate to microservice architecture to gain more advantages with more AWS services, like API Gateway, Lambda, etc.