Architecture for migrating to AWS:

# Problem Statement:

Consider a monolithic java application stack consisting of Apache Web Server, Apache Tomcat application server with Active MQ and Oracle and MongoDB backend. Propose a solution to migrate this application stack to AWS. Mention all the AWS services you would use and how you would maintain HA and Load Balancing (consider app to be stateless). Mention rationale for each design decision.

**I): *Migration to AWS based tools***

In this approach we shift from the existing tool stack to AWS managed services. It is important to note that typically moving to alternate AWS provided(managed) services, they offer almost same or sometimes less technical features, the tradeoff is the great cost cutting, super-fast scalability, avoiding the overhead management efforts.

1. *Apache Web Server and tomcat server:*

The Apache Web Server and the Apache Tomcat application are to be deployed as part of an LC-ASG-ELB setup spanning across more than one AZ's.

This would ensure HA w.r.t unavailability of Availability Zones.

To maintain stateless-ness we need store the session data / user data off-server using memcache/RDS. This is a sorted issue if we are already using the existing DB's for this purpose (Mondo DB or Oracle as mentioned)

During launching of new servers, the application will automatically be deployed using a repo such as GitHub or AWS codecommit. Further to automate the process AWS Code Deploy can be used which automatically keeps track of new instances and deploys the code as required. If there is any pre compilation required as part of the deployment a Jenkins server can be created which will do the needful, this Jenkins job can be triggered using a web-hook which is configured into the user data of the Launch Configuration.

To maintain true statelessness, it is also needed that the logs are stored away from the instance (a) By enabling access logs on the ELB[1].

(b) Enabling VPC flow logs on the ELB NIC.[2]

(c) using X\_HTTP\_Forwaded and further offloading from the server , while this will store the access logs to store the error logs persistently autoscaling hooks can be used so that once the server is being scaled in by the ASG[3], the error logs and other required data will be offloaded to an other server or cloud watch Logs.

The Security group rules of the webservers would permit only HTTP traffic only from ELB, no other ports will be opened for security reasons.

Every instance launched is associated with an IAM role to facilitate EC2-run-commands for any adhoc requirements, this is also useful for various other purposes like for code deployment or for authenticating for other services [4] [5].

To serve static images/videos and fonts, they will be placed in an S3 bucket maintaining uniformity in referencing. If the application is to be catered to a global audience or to reduce latency even further, AWS Cloud Front can be used to server from the S3 [6].

*b)ActiveMQ :*

The Active MQ messaging service will be replaced with AWS Simple Queue Service (SQS)[7], the in-depth features of SQS is beyond the scope of this document but can be readily referred from AWS documentation[7].

The functionalities of SQS and Active MQ are almost same with pro's and con's on both sides. But the weighted advantage in SQS' side is the compatibility of it being an AWS provided managed service in an AWS stack - bringing in a pool of advantages[8] [9].

Notably unlike ActiveMQ, SQS is a polling based mechanism and not push based mechanism. This can be tackled by configuring work around using SNS[10] and HTTP like[11], a more custom solution can be made based on the specific application needs

*c)Oracle:*

Oracle is offered as a managed service from AWS as part of the AWS RDS service offering. In order to maintain HA, RDS comes with an inbuilt option for a multi-AZ deployment with automatic synchronous replication among these instances.

All the standard set of features of RDS are as applicable[12] [13]. Oracle AWS RDS comes with a BYOL model to facilitate usage of the existing license hassle free.

Based on the application stack, if the RDS need not be accessed via internet, it will not be made publicly accessible and also traffic will be restricted to AWS services using SG.

*d)Mongo DB :*

Mongo DB, an open source document oriented database program is like a NoSQL DB. Though this is directly not offered as part of the AWS managed services stack, a very similar NoSQL DB - DynamoDB is offered from AWS.

https://ci6.googleusercontent.com/proxy/RnNZfQn2o2xpggJQqefCOervMbPIci5mujDPJnvl43kv6Rtxjyh5gHN_JKVzeU-aaGz3pePFgxfoAAtZJZNx8mveVTc-11j98EfuAJVcumUenA=s0-d-e1-ft#https://ssl.gstatic.com/ui/v1/icons/mail/images/cleardot.gif While AWS elastic search, a tool which queries based on JSON formats and documents is primarily used for searching and indexing, given the true nature of application this can be used in addition to the DynamoDB or as a replacement all together[14].

## **II) Using same technology stack**

In this approach we stick to the same tools as what the current stack is using, but we only migrate to AWS hosted servers.

Though this approach maintains the ease by not requiring to migrate to new tools and the overhead attached to developers to adapt to that, this is a fairly complex approach in terms of architectural and maintenance viewpoints.

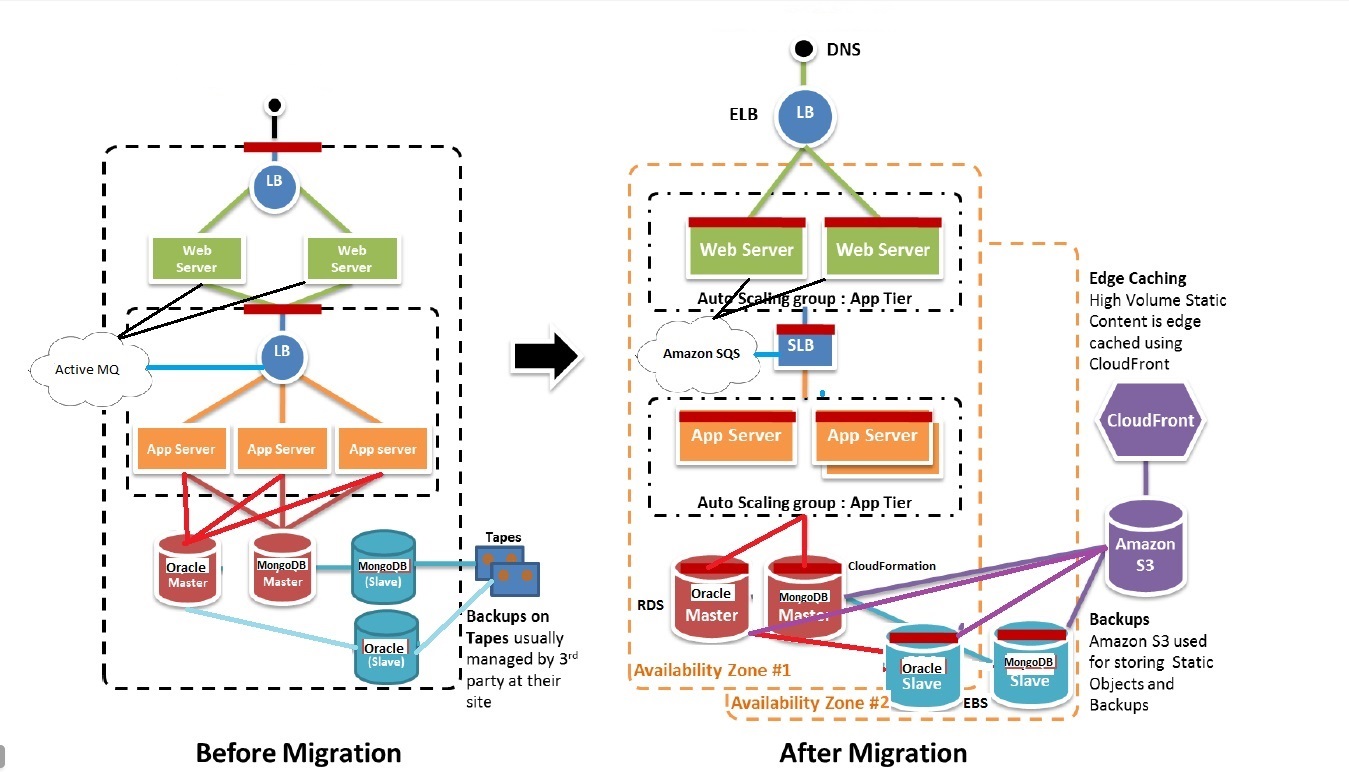
In this approach only AWS EC2 + S3 + VPC will be used, the standard set of ASG + ELB will be implemented on the stack.

Regardless of the approaches taken from the proposal some common AWS specific leverages and best practices are to be maintained. The stacks have to be deployed using orchestration tools such as CFT.

AWS EC2 offers reserved instances which can be purchased for fixed capacity to cut down on costs, on demand instances can be used when there is a spike.

Standard AWS IAM best practices are to be followed in the usage of account[15] [16] .

Architecture diagram[17]:



##### References

[1] “Access Logs for Your Classic Load Balancer - Elastic Load Balancing.” [Online]. Available: http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/access-log-collection.html. [Accessed: 17-Dec-2016].

[2] “VPC Flow Logs – Log and View Network Traffic Flows | AWS Blog.” [Online]. Available: https://aws.amazon.com/blogs/aws/vpc-flow-logs-log-and-view-network-traffic-flows/. [Accessed: 17-Dec-2016].

[3] “Auto Scaling Documentation,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/documentation/autoscaling/. [Accessed: 17-Dec-2016].

[4] “Amazon Elastic Compute Cloud (EC2) Documentation,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/documentation/ec2/. [Accessed: 17-Dec-2016].

[5] “IAM Roles for Amazon EC2 - Amazon Elastic Compute Cloud.” [Online]. Available: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html. [Accessed: 17-Dec-2016].

[6] “Amazon Simple Storage Service (S3) Documentation,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/documentation/s3/. [Accessed: 17-Dec-2016].

[7] “Amazon Simple Queue Service (SQS) Documentation,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/documentation/sqs/. [Accessed: 17-Dec-2016].

[8] “AWS Developer Forums: Migration from ActiveMQ to SQS - listen ...” [Online]. Available: https://forums.aws.amazon.com/message.jspa?messageID=305762. [Accessed: 17-Dec-2016].

[9] “Top Ten Differences Between ActiveMQ and Amazon SQS – The Dulin Report.” [Online]. Available: https://thedulinreport.com/2015/09/05/top-ten-differences-between-activemq-and-amazon-sqs/. [Accessed: 17-Dec-2016].

[10] “Amazon Simple Notification Service (SNS) Documentation,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/documentation/sns/. [Accessed: 17-Dec-2016].

[11] eran, “Using Amazon’s Simple Notification Service (SNS) & Simple Queue Service (SQS) For a Reliable Push Based Processing of Messages,” *Forecast: Cloudy*, 12-Jul-2011. .

[12] “Amazon Relational Database Service (RDS) Documentation,” *Amazon Web Services, Inc.* [Online]. Available: //aws.amazon.com/documentation/rds/. [Accessed: 17-Dec-2016].

[13] “Oracle on Amazon RDS - Amazon Relational Database Service.” [Online]. Available: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP\_Oracle.html. [Accessed: 17-Dec-2016].

[14] “Elasticsearch as a NoSQL Database,” *Elastic Blog*, 15-Sep-2013. [Online]. Available: http://www.elastic.co/blog/found-elasticsearch-as-nosql. [Accessed: 17-Dec-2016].

[15] “IAM Best Practices - AWS Identity and Access Management.” [Online]. Available: http://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html. [Accessed: 17-Dec-2016].

[16] “Adhere to IAM Best Practices in 2016 | AWS Security Blog.” [Online]. Available: https://aws.amazon.com/blogs/security/adhere-to-iam-best-practices-in-2016/. [Accessed: 17-Dec-2016].

[17] <http://sagyoceancloud.blogspot.com/2017/03/consider-monolithic-java-application.html>