AWS Architecture Design for a Web Based Software Solutions

Background:

Suppose a start-up company in the early stages of their operations. Currently their architecture uses a LAMP (Wordpress/PHP) stack all running on one physical server. Like many small start-ups they are confident that they will be the next big in market and expect significant, rapid growth in the near future. With this in mind, they are concerned about the following:

- ✓ Scalability: scaling to meet the demand, but with uncertainty around when and how much this demand will be they are very concerned about buying too much infrastructure too soon or not enough too late.
- ✓ Accessibility: ability to configure their database and data access layer for high performance and throughput.
- ✓ Latency: making the user experience in the browser very low latency even though a large portion of their user base will be from far away
- ✓ Load Balance: effective distribution of load
- ✓ Failover: a self-healing infrastructure that recovers from failed service instances
- ✓ Disaster Recovery: lack of provision of Web/App/RDS for natural disaster recovery.
- ✓ Security Optimisation: security of data at rest and in transit.

Target:

Recommend a manageable, secure, scalable, high performance, efficient, elastic, highly available, fault tolerant and recoverable architecture that allows the start-up to grow up organically. The architecture should specifically address the requirements/concerns as described above.

Action:

This is a high level reference for a web based services start-up company which hopes for a significant growth in near future. This proposed architecture is based on AWS solutions and inherently enables various enterprise capabilities to the overall solution. Why AWS is good fit to design for web based software solutions:

- ✓ *AWS Solutions elasticity nature enables the enterprise cloud capabilities to the solution:* Since growth of the company is not guaranteed and to keep the solution cost effective over the time, the solution should be easily re-sizeable to match the growth. AWS solutions readily bring these capabilities at every stage of the solution to the desk. AWS components are elastic in nature, that is they can auto re-size to a given load at run time. Say, using EC2 you can have more servers added to the solution if the load on servers increases and vice-versa. AWS pay-as-you pricing model keeps investments as well elastic. That is when you have more load, you use more resources then you pay for added resources and vice-versa. Choosing AWS enables the elasticity to the overall solution for any loads over any time.

- ✓ *AWS solution is a Scalability booster for handling the unpredicted traffic loads:* It's very often heard that upcoming web based solutions suffer performance during peak loads. This is a result of resources not being scalable to meet the unexpected increase in load at real time. AWS auto scale provisioning of VM instances helps you add more web servers readily as needed to serve the increased loads. They can be taken offline or removed when the load

returns to normal as needed. This capability boosts the scalability of overall solution to the ever changing traffic loads.

✓ *Easy manageability of the solution deployments and maintenance:* Being a start-up and web based solution; company may often need to deploy changes to the system. They would require a consistent close of production environment at every level of their Development, QA testing, Staging and Prod. Yet again the on-demand provisioning of resources in AWS improves turnaround time to deploy and test the changes at every stage to ensure stable release of product in the Production. This minimizes the hardware and maintenance costs involved at every stage of development cycle.

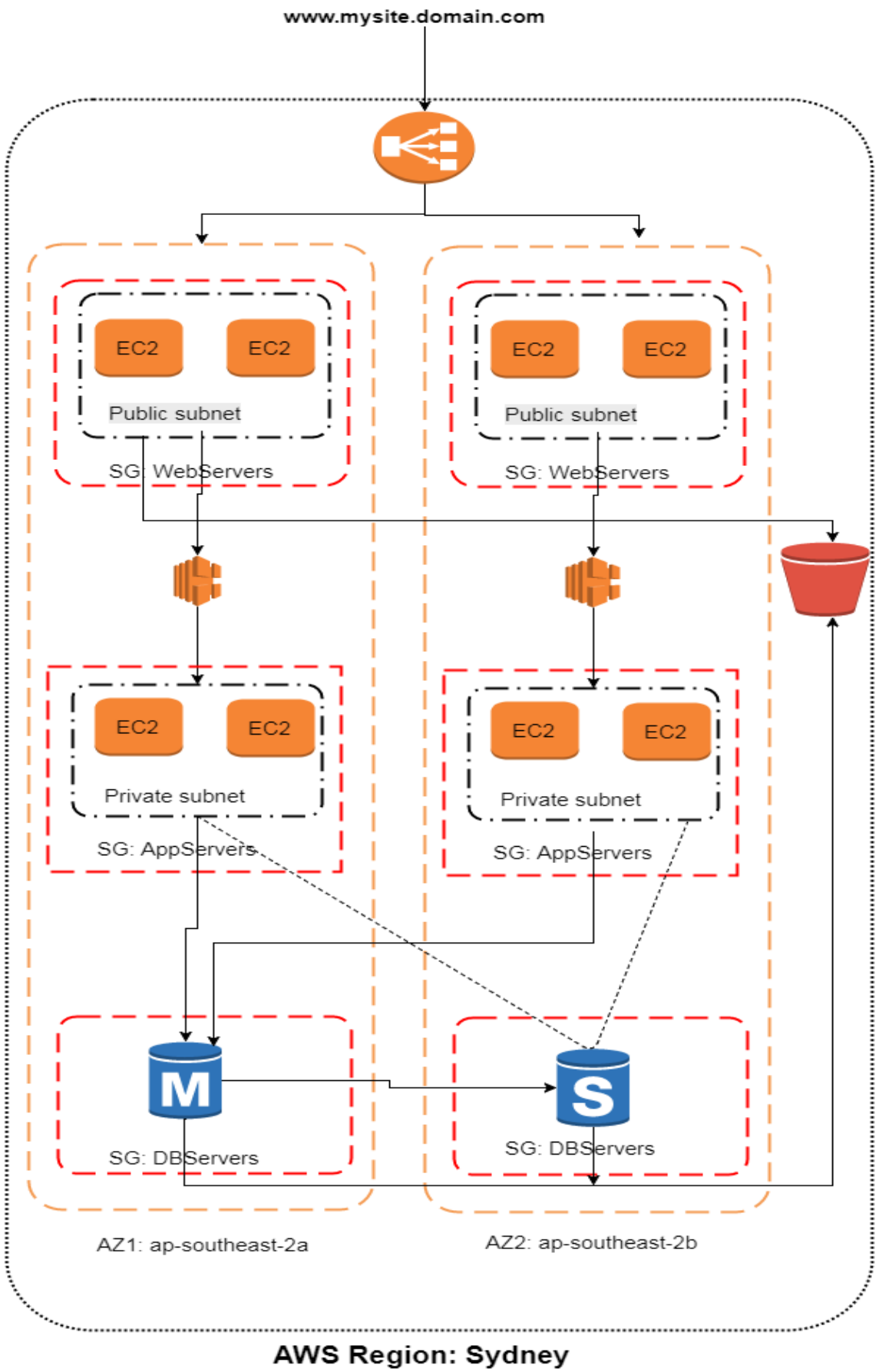
Design/Architecture:

To design this architecture there are key components and resources are needed to going forward. Following are the various layers at which a specific AWS skillsets are needed to manage and own the responsibility.

✓ Infrastructure/Network Tire: Manages external/internal network configurations and security via AWS Route 53, ELB, Multi-Availability Zones, SG, IGS, NAT, Public/Private Subnet, Elastic IPs, Security Groups,
✓ Web Server Tire: Manages Web Servers via AWS EC2, AMI images, etc. Responsible for configuring and maintaining web server instances to handle the web requests. Responsible for configuring and maintaining App server instances to handle the application requests
✓ Application Server Tire: Manages App Servers via AWS EC2, AMI images, etc.,
✓ Database Server Tire: Manages Database Servers and Data via AWS RDS, IOPS Volumes, Replication, and Backup. Responsible for configuring and maintaining the backend database access, security, performance and availability.

Result/Outcome: Automation through CloudFormation

With all of the above AWS features and capabilities the proposed architecture takes the start-up company. You will get ready the whole system within 20-30 minutes through AWS CloudFormation. You may want to just review the pricing of the AWS components on pay-as-you-use mode to make a call on migration to AWS.


www.mysite.domain.com
EC2
EC2
Public subnet
SG: WebServers
EC2
EC2
Public subnet
SG: WebServers
EC2
EC2
Private subnet
SG: AppServers
EC2
EC2
Private subnet
SG: AppServers
M
SG: DBServers
S
SG: DBServers
AZ1: ap-southeast-2a
AZ2: ap-southeast-2b
AWS Region: Sydney