AWS Architecture Design – Assignment #2 for EC2 Core SSA

AWS Solution Design and Proposal Document for "XXX" Company

Wentao,Cui

1 Assumption of the Case

2 Executive Summary

2.1 Requirements Analysis

XXX company is new startup company. Currently the business is in the early stages with good development trend, but like most startup company, it's still with uncertainty. The main architecture and technologies are based on LAMP stack with Apache, PHP and MySQL. The below aspects of requirement are considered by customer for future's business extension.

- Functionalities
- Security
- High Availability
- Scalability
- Performance
- Data Backup and Recovery
- Flexibility
- Cost

2.2 Solution Abstract and Benefits

In this solution, the following AWS services are mainly used and involved. They can provide and meet the all requirements mentioned in above chapter.

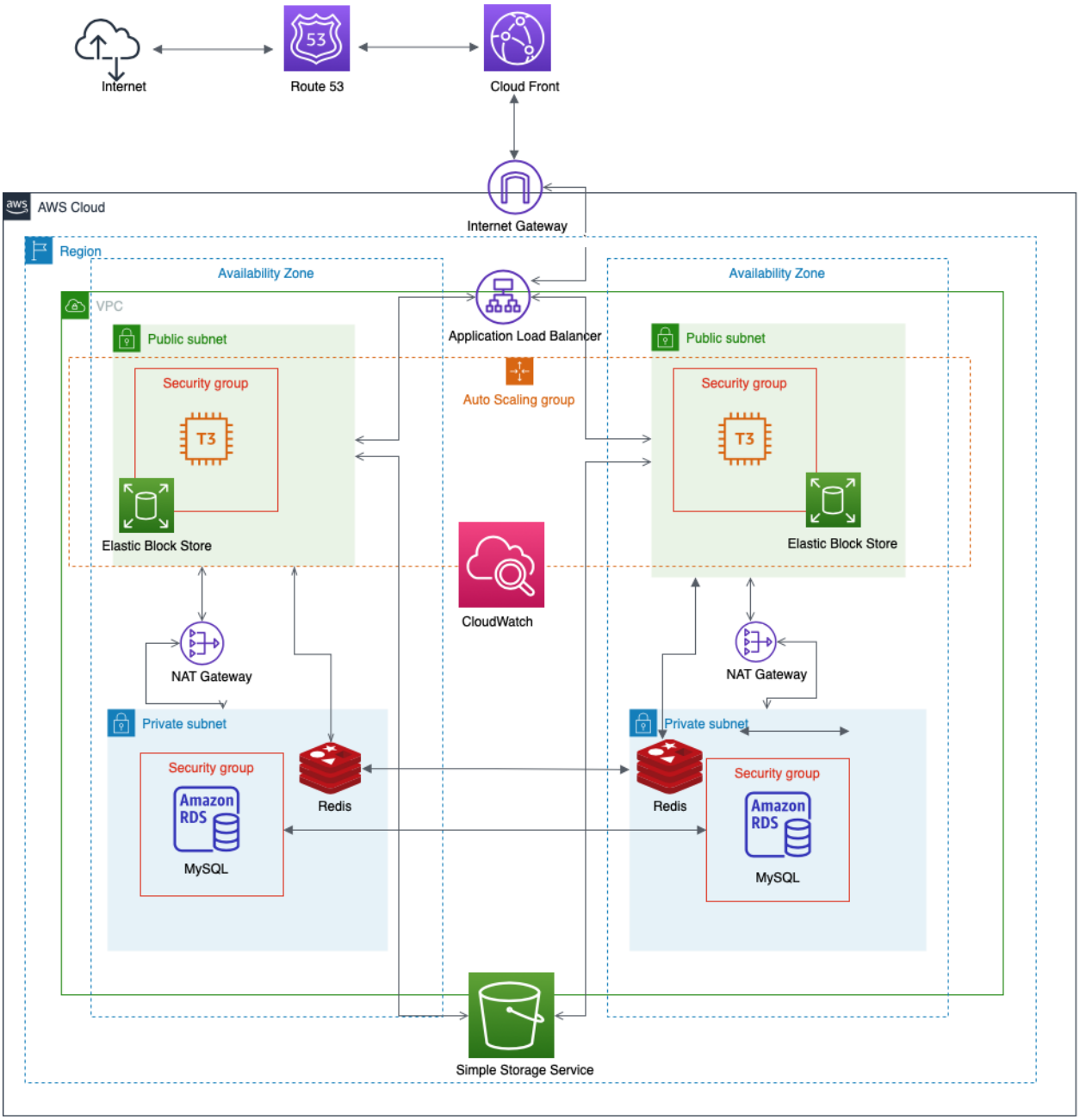
For detailed description for each service below, please refer to the following chapter.

- EC2
- EBS
- RDS
- Network - VPC
- Storage - S3
- Route53
- ACM
- IAM
- ElastiCache
- CloudFront
- CloudWatch

3 Solution Design

3.1 Architecture Overview

The following diagram shows the general design of whole solution architecture.



3.2 Design Details

3.2.1 EC2

EC2, one of most popular of AWS services, and mainly consists in below the capability of

- Virtual Machine (EC2)
- Store data (EBS)
- Load Balancer(ELB)
- Auto-scaling group(ASG)
- Images (AMIs)
- EBS Snapshots

Below is detailed design of EC2 service of current solution,

- Instance type: T2 or T3
  - ⇒ e.g. "T3" is general purpose type instance, T3 instances offer a balance of compute, memory, and network resources and are designed for applications with moderate CPU usage that experience temporary spikes in use.
- AMI: Customized image, which including EC2 user data. By using AMIs, customer can easily manage and replicate for multiple environments. Customer also can launch new EC2 instance with different type instance to support vertical scalability
- EBS: Provide local data storage for EC2 instance. In this solution, it's mainly used to store PHP files of Webserver, and other utility files
- EBS Snapshot
  - ⇒ Incremental backup
  - ⇒ Stored in S3
  - ⇒ Can copy and move across AZ or Region
  - ⇒ EBS volumes can be restored by snapshots
  - ⇒ Can be Automated using Amazon Date Lifecycle Manager
- ELB: EC2 Load Balancer, provides high availability, which means running application / system in at least 2 data centers(AZ). The goal is to e.g. survive a data center loss. The following design factors are involved in current solution
  - ⇒ Expose single point of access(DNS)
  - ⇒ ALB – Application Load Balancer is used
  - ⇒ Expose HTTPS/HTTP endpoint to other services, e.g. CloudFront/Route53
- ASG: Auto-scaling-group, provides horizontal scalability for EC2 instances. The below design factors are involved in current solution
  - ⇒ Desired capacity: 2, Minimum capacity: 1, Maximum capacity:4
  - ⇒ Policy type:
    - Target tracking scaling, execute policy when: "CPU utilization at around 80"
    - Simple Scaling: When a CloudWatch alarm is triggered, ASG CPU>=80%, then add 1 unit
  - ⇒ Attached to ALB
- AMI: customized image, which is used to create EC2 instances. It provides below advantages:
  - ⇒ Pre-installed packages needed
  - ⇒ Faster boot time
  - ⇒ Install app ahead of time, for faster deploys when auto-scaling
  - ⇒ AMI is stored in S3, which is durable, cheap and resilient storage

⇒ Quite inexpensive

3.2.2 RDS

RDS, means Relational Database Service. It's managed DB service use SQL as a query language. It mainly consists in below the capability of

- Automated provisioning
- Continuous backups and restore, e.g. to a specific timestamp
- Monitoring dashboards
- Read replicas for improved read performance
- Multi-AZ setup for Disaster Recovery
- Scaling capability (vertical and horizontal)
- Storage backed by EBS

Below is detailed design of RDS service of current solution,

- Instance type: db.t3.micro/db.t3.small
- Read replicates for read scalability: master and slave cross AZ
- Backups: are automatically enabled in RDS
- Snapshots: Manually triggered, and retention of backup for as long as you want
- Disaster Recovery
  - ⇒ SYNC replication
  - ⇒ One DNS name, make app failover to standby automatically
  - ⇒ Failover in case of loss of AZ, network instance or storage failure
  - ⇒ Encryption is enabled to make sure data security, e.g. replication between master and slave

3.2.3 Network – VPC

Amazon Virtual Private Cloud (VPC) is a service that lets you launch AWS resources in logically isolated virtual network that you define. It mainly consists in below the capability of

- Subnets
- Route Tables
- Internet Gateway
- Elastic IPs
- NAT Gateway
- Security Groups
- Firewall /VPN

Below is detailed design of VPC service of current solution,

- Subnets
  - ⇒ public subnet for webserver
  - ⇒ private subnet for RDS/MySQL
- Security Groups:
  - ⇒ MySQL
  - ⇒ ALB
  - ⇒ ElastiCache
  - ⇒ Ec2-Alb: used to connection between EC2 and MySQL

- NAT gateway: Allows instances in the private subnets to connect to the internet
- Elastic IP: used and attached in NAT gateway
- Internet gateways: helps VCP instances to connect with the internet
- DNS: enabled

3.2.4 S3

S3 is one of the main building block of AWS. It's widely popular and deserves its own section. Many websites use Amazon S3 as backbone or integration.

In current solution, s3 is mainly used for store for inactive data and objects greater than 6 months, to support and meet customer's archival strategy requirement. In future, more static type files, e.g. image, video, could also be added into S3.

Below is detailed design of S3 service of current solution,

- Create S3 bucket with globally unique name
- Bucket Version: enabled
- Permission: block all public access
- Customer can create backup script to backup data to S3. Below is a sample design for this
  - ⇒ Sample backup script: backup inactive data greater than 6 months to S3 bucket by using AWS CLI

```
#!/bin/bash

# Example of S3 backup script

TIME=`date +%b-%d-%y`
FILENAME=opencart-webserver-backup-$TIME.tar.gz      # The filename including the date.
SRCDIR=/var/www/html                                 # Source backup folder.
DESDIR=./backup                                      # Destination of backup file.
~
mkdir $DESDIR
~
find $SRCDIR -mtime +60 | xargs tar -cpzf $DESDIR/$FILENAME
aws s3 cp $DESDIR/$FILENAME s3://opencart-s3-backup-bucket/
```

  - ⇒ Create scheduled task to execute above script. For example, execute crontab -e, and add one line "00 04 * * 1,5 /bin/bash /home/ec2-user/opencart-backup.sh"

3.2.5 Route53

Route53 is managed DNS, a collection of rules and records which helps clients understand how to reach a server through URLs.

Below is detailed design of current solution:

- Register a new public domain name, e.g. www.yibinet.link
- Add new record with CNAME type, route traffic coming from www.yibinet.link to another domain name and to some AWS resources, e.g. CloudFront or ALB
- Simple routing policy is used, but customer can change this to support more robust policy, e.g. there are more than one load balancers

3.2.6 ACM

ACM - AWS Certificate Manager is the service used to manage certificates. In current solution, a certificate is allocated for domain name to provide and support TLS/SSL.

3.2.7 IAM

AWS Identity and Access Management (IAM) enables customer to manage access to AWS services and resources securely. By using IAM, customer can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources. Below is detailed design of current solution

- Create a new user “aws_cli_admin” with “Programmatic access” access type
- Attach “AdministratorAccess” policy to user
- The user “aws_cli_admin” is used by backup script, to backup data to S3 bucket
- Because there is “delivery team expands” requirement, customer can create more users with different permissions to execute different tasks

3.2.8 ElastiCache

ElastiCache, managed Redis/Memcached service, mainly used for in-memory data store, sub-millisecond latency access. Below is detailed design for ElastiCache of current solution

- 3-node Redis cluster with Multi AZ
- Backup/Snapshot is enabled
- Change and customize PHP file to optimize and accelerate MySQL database query performance
- Below is sample PHP source code fragment, which shows the MySQL query optimization

```
<?php
$redis = new Redis();
$redis->connect('opencart-redis-cache.sjnnna.ng.0001.apse1.cache.amazonaws.com', 6379);

$key = 'PRODUCTS';
if (!$redis->get($key)) {
   $source = 'MySQL Server';
   $database_name     = 'opencart_db';
   $database_user     = 'admin';
   $database_password = '';
   $mysql_host       = 'opencart-database.cfngwnpejidk.ap-southeast-1.rds.amazonaws.com';

   $pdo = new PDO('mysql:host=' . $mysql_host . '; dbname=' . $database_name, $database_user, $database_password);
   print_r( "db connect ok!");

   $sql  = "SELECT * FROM oc_products";
   $stmt = $pdo->prepare($sql);
   $stmt->execute();

   while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
     $products[] = $row;
   }

   $redis->set($key, serialize($products));
   $redis->expire($key, 10);

} else {
    $source = 'Redis Server';
    $products = unserialize($redis->get($key));
}
```

3.2.9 CloudWatch

Amazon CloudWatch is a monitoring and observability service built for DevOps engineers, developers, site reliability engineers (SREs), and IT managers. CloudWatch provides you with data and actionable insights to monitor your applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health.

In current solution, CloudWatch is mainly used to monitor EC2 auto-scaling-group, to add/remove EC2 instance dynamically. Below is detailed design for CloudWatch

- Create new alarm
  - ⇒ Metrics: “By Auto Scaling Group” -> “CPUUtilization”
  - ⇒ Statistic: Average
  - ⇒ Period: 5 minutes
  - ⇒ Conditions: Greater/Equal (80%)
  - ⇒ Auto Scaling action: “Add 1 instance”

3.2.10 CloudFront

AWS CloudFront, Content Delivery Network(CDN), improves read performance, content is cached at the edge. Customers can access content globally with low latency, high transfer speeds.

TBD: Customer need to evaluate how to use CloudFront to optimize PHP website.

3.3 EC2 instance selection: AMD vs INTEL vs ARM

3.3.1 General Introduction

- **Intel.** Intel is the most popular and well-known maker of processors. Manufacturers like Dell, Apple, Samsung and HP all use Intel processors in their computers. Intel processors are the most stable and offer the best all-round performance. The current i3, i5, and i7 models represent entry, middle and high level hardware.
- **AMD.** AMD is Intel’s biggest competitor, offering processors that are similar to Intel’s, but at a, for the most part, cheaper price. The majority of computer manufacturers, except for Apple, also offer products with AMD processors. AMD’s Athlon processors are budget models while Phenom and FX are mainstream and high level respectively.
- **ARM.** ARM processors are generally used in smartphones, mobile devices and tablets. Apple’s iPhone and iPad; Samsung’s Galaxy line and HTC devices all use some form of ARM processor in their mobile devices. A rule of thumb is, if it doesn’t have AMD or Intel in the name, it’s most likely an ARM processor

3.3.2 Detailed Technical Introduction

- CISC vs RISC: Intel processors use Complex Instruction Set Computing while ARM uses Reduced Instruction Set Computing

- The ARM processors use only one cycle to execute a command, hence, it reduces functions. While the Intel processors use a simpler command code, it must go through several cycles before the action is complete.
- Power Consumption: ARM processors use less battery life and also have a reduced operating temperature than intel processors. Intel processors are focused on performance mainly
- Speed: ARM chips are usually slower than inter counter parts

3.3.3 How to benchmark for CPU

Different benchmark tools can be used to evaluate CPU performance, such as Geekbench, Cinebench and so on.

For example, Geekbench, a cross-platform benchmark tool, which can be used for measuring system performance. For CPU, it mainly included below features,

- Single-core vs Multi-core
- Cross-Platform, INTEL vs ARM

Below is example of the steps for AWS ec2 instance performance evaluating by using Geekbench

```
ssh -i opencart-key-pair.pem  ec2-user@ec2-54-169-154-124.ap-southeast-1.compute.amazonaws.com
sudo wget -0 http://cdn.geekbench.com/Geekbench-5.1.0-Linux.tar.gz
sudo tar -xzvf Geekbench-5.1.0-Linux.tar.gz
cd Geekbench-5.1.0-Linux
sudo ./geekbench5
```

3.4 Cost Saving Consideration

For EC2 instance, there are below types instance,

- On Demand Instance: short workload, predicable price. Pay for what you use, and has highest cost but no upfront payment
- Reserved Instance curl https://www.yibinet.link/test.html
    - ⇒ Reserved instance: long workload
    - ⇒ Convertible Reserved Instance: long workload with flexible instances
    - ⇒ Scheduled Reserved Instances: e.g. every Monday between 10am and 14pm
- Spot Instance: show workload, for cheap, and can lose (less reliable)
- Dedicated Instance: no customer share your hardware
- Dedicated Hosts: bool an entire physical server, control instance placement
-

For cost optimization consideration, it's suggested to select below combination

- Reserved Instance with period > 1 or 3 years
- Pay upfront if it's possible
- Add on-demand and spot instance for temporary jobs, e.g. temporary short-period high traffic, batch jobs, data analysis, image/video processing and so on

4 References

- https://www.hireanitexpert.com/intel-amd-and-arm-processors-explained/
- https://www.alphr.com/features/390064/arm-vs-intel-processors-what-s-the-difference