Swinburne University of Technology

Faculty of Science, Engineering and Technology

COS20009 Cloud Computing Architecture

Mid-Semester Project

Important Dates

- 16 April 2018: Design document submitted to Turnitin, review and presentation (in class)
- 30 April 2018: Implementation Complete. Documentation submitted to Turnitin.

Introduction

- InnoMed is a startup software as a service (SaaS) company.
- It has built an online medical social networking and diagnosis assistance application for users in APAC, the US, and Europe.
- The application connects patients and doctors to allow online appointments, remote consultation, remote diagnosis, electronic prescription transfer, and payment services.
- The application allows customers to upload documents and images. Text is extracted from documents, and images are converted into multiple formats.
- The application has not yet been launched publically.

Overview

Current Environment

- InnoMed's development and test infrastructure is deployed with a server hosting company.
- InnoMed uses Microsoft Windows servers to host their web and application tiers with Microsoft SQL Server Standard Edition backend databases.
- The application launch date is coming soon and InnoMed expects many users to start using the application.
- InnoMed has decided to use cloud technologies to support its rapid growth.
- InnoMed has hired you to architect an infrastructure in AWS to meet their application needs.

Current Architecture

Web Tier

- Two physical servers (Two CPUs / 4-GB memory)
- Microsoft Windows 2016 Base with IIS
- HAProxy load balancer used to balance traffic between the web servers

Application Tier

- Two physical servers (Four CPUs / 16-GB memory)
- Microsoft Windows 2016 Base with IIS
- HAProxy load balancer used to balance traffic between app servers

Database Tier

- One physical server (Eight CPUs / 32-GB memory / 5-TB storage)
- SQL Server Standard Edition – v12.xx
- Microsoft Windows 2016 Base
- DBAs access and manage the database, but no RDMBS or advanced configuration is required.

Requirements

General

InnoMed plans to move their application into AWS.
The plan includes:

- Configuring access permissions to conform with AWS best practices.
- Configuring auditing to track all user actions.
- Building networks that conform to AWS best practices while providing all the necessary network services to the applications in their different environments.
- Building an architecture that matches the current architecture at the server hosting company and that can handle doubling the number of servers.
- Ensuring that no code changes are required.

User Authentication

- Users with access to AWS are in three groups:
- System Administrator group: 2 users
- Database Administrator group: 2 users
- Monitor group: 4 users
    - Monitoring infrastructure resources for the application (EC2, S3, RDS)
- Follow AWS best practices for distributing permissions.
- The InnoMed application must read and write to S3 buckets.
- A password policy should enforce the following:
    - A password with at least 8 characters, 1 uppercase and 1 lowercase letter, 1 number, and 1 special character
    - Forced password change every 90 days
    - No re-use of the previous three passwords
- All administrators require programmatic access and AWS Management Console access.
    - When signing in to the console, each administrator is required to provide a user name, a password, and a random generated code provided by the Virtual MFA.
- All other users should only have AWS Management Console access, using a combination of user name and password.

Auditing

Administrators must be able to track every AWS action in the account, including:

- Every service or object that was created or configured.
- The user who created the object or configured the service.

- The interface used for each creation or configuration action.
    - For example: AWS Management Console or CLI
- The source of each creation or configuration action.
    - For example: IP address or DNS alias

Networking and Security

The new architecture must conform to AWS best practices.
This includes:

- Achieve high availability for all tiers to reduce downtime.
- Control access to the applications and limit public entry points.
- Minimize IP address usage to reduce the attack surface.
- Maintain separate networks for InnoMed's development/testing environment and production environment. The web tier load balancer can receive requests from the Internet on port 80.
- Web tier servers can receive requests from the web tier load balancer only on port 80.
- The Application Load Balancer can receive requests from web tier servers only on port 8080.
- Application tier servers can receive requests from the application tier load balancer only on port 80.
- Database servers can receive requests from application servers only on port 1433.

Web Tier and Application Tier

- All the instance names in the web tier should be tagged as Key = Name and value = web-tier.
- All the instance names the in application tier should be tagged as
  Key = Name and value = app-tier.
- All instances in the application tier must support EBS optimization. (Note!!!)
- Load balancers for web tier and application tier must support HTTP, HTTPS, and TCP protocols.

Business Continuity

The new architecture should be designed for business continuity.
This includes:

- The web and application tiers should be self-healing.
    - If a server becomes unavailable it will be replaced by a new server.
    - A server is considered to be unavailable if the operating system or application fails to respond.
- The database tier should support Multi-AZ deployment.
- The architecture should handle doubling the number of servers to support its rapid growth

Task 1 – Create an implementation design document

This task can be done individually or in a team of 2. You will present your design to the class.

The document should be submitted to Turnitin and contain the following:

1. **AWS diagram** - showing the VPC, availability zones, subnets, services (ELBs, EC2 instances, scaling groups, etc.) and links between each of the services. Icons for various formats are available at https://aws.amazon.com/architecture/icons/

2. **IAM** - define the users, groups and roles that you will be creating. List the AWS permissions that will assignment to each role/group.

Group/Role #	Group/Role Name	Permissions
Group	e.g. DBAdmin	AmazonRDSFullAccess
Group		
Group		
Role		

3. **Password policy** – define policy noting any special requirements for particular groups.

4. **VPCs** - List the VPCs that need to be created including: Name (e.g. Production), Region, AZs to be used, number of subnets required and CIDR range. (Check your selected region supports RDS Multi-AZ).

5. **Subnets** – for the Production VPC only, define each of the subnets including the name (the name should indicate what tier the subnet is e.g. "WebPrivate1"), type (public/private), AZ and subnet CIDR address range.

6. **Instance details –** describe the type, size and justification for the EC2 instances you will use in each tier.

Tier	Tag*	OS	Type	Size	Justification	# of instances	User Data?
Web	Key = Name Value = web-tier						
App	Key = Name Value = app-tier						
DB	Key = Name Value = db-tier						

* Tags must be configured as shown to meet the lab objectives

7. **Load balancer details**

Load Balancer	Name*	External/Internal	Subnets	SG Name*	Rule	Source
For Web Tier	web-elb			web-elb-sg		
For App Tier	app-elb			app-elb-sg		

8. **Security group details –** describe the type, size and justification for the EC2 instances you will use in each tier.

Instance Tier	SG Name*	Rule	Source
Web Tier	web-tier-sg		
App Tier	app-tier-sg		
Database Tier	db-tier-sg		

* Names must be configured as shown to meet the lab objectives

9. Auto Scaling Launch Configuration

Tier	OS	Type	Size	Configuration Name*	Role	Security Group
Web				WebTier		
App				AppTier		

10. Auto Scaling Group

Tier	Launch Configuration*	Group Name*	Group Size	VPC	Subnets	ELB	Tags
Web	WebTier	WebTier					
App	AppTier	AppTier					

* Name must be configured as shown to meet the lab objectives

Task 2 – Implement your design

Overview

Use your solution design to implement a ***subset*** of your solution in the Qwiklabs lab environment. This allows you to practice the configuration and validate your design. It is necessary to configure the entire environment in the time.

There is no application code available in the lab environment to test your solution.

Trackers

The lab environment includes tracking of specific objectives. Due to the limitations of the tracking system it help to complete the objectives in the order specified by the tracker.

Configuration

A ***subset*** of your designed solution should configured in the lab environment.
We recommend you configure the following services:

1. Users, and roles With the correct policies and associations
2. Production VPC
3. Internet Gateway
4. Subnets for all tiers two availability zones
5. Route tables and routes With correct subnet associations
6. Security Groups for all servers and load balancers
7. Auditing Of all console and API user actions
8. Load Balancers
9. Launch Configurations
10. Auto Scaling Groups
11. Compute Instances running web services
12. Storage With a lifecycle policy

Validating Your Solution

The following tasks can completed to validate your own solution.

Viewing web pages from the web tier instances

• If your web tier instances deploy with user data to configure a web service, then requests to the Internet facing load balancer should provide a web page in response.

Auto Scaling Groups

• Manually increase the number of desired instances in the application tier group from two to four. The new instances should be deployed across multiple availability zones automatically.
• Check the application load balancer has been automatically updated with the two new instances.

Auditing

• Move a user from the Monitor group to one of the administrator groups.
• View the audit logs to verify the change was logged.

IAM Roles

• View the configuration of the web tier instances and verify they are running with the appropriate IAM role that allows them to use the storage service.

Task 3 Documentation of your implementation

Document your implementation solution with screenshots and produce a CloudFormer configuration export. The document will be submitted to Turnitin by the due date and included in your Portfolio.

Mid-Semester Project Marking Scheme

Design

AWS Diagram VPC(s), Subnets, Components/Services (ELBs, EC2 instances, scaling groups, IGW, NATs, S3, etc.) clearly illustrated Highly available all tiers Minimal attack surface	3
IAM and Passwords Groups, roles and policies correctly defined	2
Security groups defined	2
EC2 instances correctly specified (with rationale) Scaling policy defined	3

Implementation check list

The following will be checked in you implementation. Make sure your portfolio has sufficient evidence for this to be checked.

1. IAM – Groups, user and roles (3 marks)
   1.1. Specified Groups created with appropriate permissions
   1.2. Users added to groups
   1.3. MFA enabled for SysAdmin user
   1.4. Password policy created as specified
   1.5. S3 access role for EC2
2. Log API calls with CloudTrail (2 marks)
   2.1. Tracker Log all API calls
   2.2. Log saved to S3 bucket with appropriate lifecycle
3. VPC (3 marks)
   3.1. VPC created
   3.2. Public and private subnets in two AZs – appropriate CIDR ranges.
   3.3. NAT instance or gateway created
   3.4. Make NAT accessible internet in Main Route Table, and associate with private subnets
   3.5. Internet gateway created for VPC
   3.6. New (Main = NO) route table created that routes to IGW and is associated with Public subnets
4. Security groups (at least 3) created with appropriate protocols and sources (3 marks)
5. Web Servers (3 marks)
   5.1. Web server configuration instance created with appropriate AMI, user data, specified role and tag Name
   5.2. Image created from Web tier configurations instance
   5.3. Web tier load balancer (classic) created in VPC with specified name, associate with appropriate subnets in both AZs, and security groups.
   5.4. Launch Configuration created based your saved AMI and with specified Names. Enable CloudWatch and add appropriate user data, storage, security group etc.
   5.5. Auto Scaling Group created with appropriate policy (desired, min and max sizes)
   5.6. ASG configured to received traffic from Web tier ELB
   5.7. Tag created Name = **web-tier**
6. App Servers (3 marks)
   6.1. App tier configuration AMI, ELB, Launch configuration, ASG created (as for Web servers)
7. Create RDS db (3 marks)
   7.1. DB subnet groups created
   7.2. Create RDS instance of appropriate type created, SQL Standard ed., db.m4 2xlarge (Lab 5.3)
   7.3. Multi-AZ, DB instance id, master username and password set
   7.4. Appropriate security group selected, DB not public accessible.