COMP 4651 - Cloud Computing

Human Relations & Customer Relationship Management System

Oliver Archer (21159115), Vincent Yap (21158927)

Contents

I. Motivation

In our increasingly globalised world, the scope and demands of businesses have drastically evolved, and the necessity for infrastructure to support these new demands is evident. With the ubiquity of the internet even small businesses are likely to have customers – and with the rise of remote work, possibly even employees – in countries other than their own.

This entails a multitude of issues when managing employees and customer relations which we propose a cloud-based human resources customer relationship management (HR CRM) solution is well-suited to solve.

II. Specification

A. Problem Analysis

Our solution is tailored to address challenges frequently encountered by multinational corporations, with requirements naturally stemming from the legislative and operational constraints they face. To meet these demands, the software must ensure global accessibility at all times while securely managing sensitive data, such as payslips, paid leave records, and employee details. This gives rise to the following key requirements:

- 24 hour availability during the work week to cover all time zones
- Security measures for file storage
- Regulated access for employees only; minimising employee access to what is strictly necessary

Additionally, the system should provide a smooth experience for its users, meaning user experience features, such as a Single Sign-On (SSO) system, are important.

For a complete solution, further consideration could also be given to product development within the company. For example, a pre-production pipeline and environment could be integrated, allowing developers to internally validate new software functionality, testers to ensure these features match requirements and work as expected, and these changes to be deployed to the production environment. However, this will be omitted from the scope of the project.

B. Project Scope & Requirement Formalisation

In the interest of relevance to the module, and feasibility given the time frame for development, a decision to focus on the high availability, high redundancy, and security aspects of the cloud CRM solution was made.

This focus includes key elements such as inbound traffic filtering, HTTPS access to the web app, and the implementation of internal AWS service communications protocols in TLS 1.2.

Additionally, the aim is to also implement SSO login support using the SAMLv2 protocol.

We have based our assumptions on the typical scale of multinational corporations, which generally employ over 60,000 employees worldwide (Minbaeva and Navrbjerg 2011, p. 104). Consequently, our system is designed to meet the following development requirements:

- Secure communication via HTTPS and a firewall
- Defenses against Distributed Denial of Service (DDOS) attacks
- TLS 1.2 protocol for internal communication between AWS services
- 24-hour-a-day availability from all regions during the work week
- SSO in SAMLv2 for system security and user convenience
- Capacity to support 60,000 concurrent
- SFTP communication with third-party tools

Given the varying types of data to be stored at each index a No-SQL key-value database makes the most sense. This eliminates needing an overly complex SQL schema to accommodate the varying data stored on each employee.

III. Design

A. Load Balancing

To ensure the requirements for availability and redundancy are met, deploying multiple instances of the app alone is insufficient.

To address network load distribution, the methods discussed by Patil (2022) are implemented; utilizing the *Elastic Load Balancer (ELB)* module in AWS, network traffic is distributed across multiple Availability Zones (AZs) within a specific region. Additionally, incorporating the *Route 53* DNS module is critical for managing load balancing across regions. This ensures that, in scenarios where a particular region becomes overloaded, traffic can be rerouted to alternative regions to maintain optimal performance and reliability.

For example, if servers in North America (e.g., those located in Canada and the USA) experience high load or downtime, users accessing the application from this region could be redirected to servers in Europe or South America. This approach enhances system reliability and ensures uninterrupted service delivery.

B. Security & Authentication

In addition to the aforementioned AWS modules, user identification and authentication must be addressed; as highlighted by Ismailov (2024), *Amazon Cognito* emerges as a suitable solution, providing secure and scalable Customer Identity and Access Management (CIAM) for employees. Crucially for the application, it supports SAML, along with a range of other authentication protocols.

Furthermore, in order to facilitate HTTPS connectivity, between the open Internet and the application, and TLS 1.2 secured inter-service communication *AWS Certificate Manager (ACM)* is employed. This allows for use to seamlessly handle SSL and TSL certificate handling for the deployed application instances.

C. Compute

In terms of performance, supporting 60,000 concurrent users requires a relatively powerful *Elastic Cloud Computing (EC2)* instance specification. However, since multiple instances will be deployed to ensure high availability and address potential bottlenecks, the specifications of each instance can be reduced to handle approximately 30% of the load – equivalent to 18,000 concurrent users per instance.

Network bandwidth must also be considered as it is critical for ensuring reliable, low-latency data transfer. Based on EC2 specifications (AWS 2024), the most suitable option for this use case is a compute-optimized instance type. This configuration is ideal as storage requirements are handled externally, making the storage-optimized instance type unnecessarily redundant. Additionally, a general-purpose instance type could be considered for the web app, given that the app functions as an interface hence not requiring as much expensive compute power.

Among the compute-optimized options, the *c8g.4xlarge* instance is particularly appropriate, offering 16 vCPUs, 32 GiB of RAM, and up to 15 Gbps of network bandwidth. This configuration balances performance and scalability, aligning well with the application's requirements.

D. Architecture

To achieve high availability and redundancy for the application, a multi-tiered architectural approach is adopted, leveraging key AWS services to ensure scalability, reliability, and security. The core components and their roles are as follows:

- **Global Deployment of EC2 Instances**: At least one *EC2* instance is deployed on each continent. Within each region, Cross Availability Zone replication is enabled using *Amazon Elastic Disaster Recovery (AWS DRS)* to safeguard against regional failures.

- **Regional Elastic Load Balancers (ELBs)**: Each region employs an *Elastic Load Balancer (ELB)* to distribute incoming network traffic evenly across its local EC2 instances and their replicas. This ensures a balanced workload within the region.
- **Global Traffic Management with Route 53**: *Route 53* DNS routing distributes traffic across all globally deployed instances. When a region becomes overloaded or experiences downtime, *Route 53* automatically redirects users to alternative regions, maintaining seamless access.
- **Secure Communication with AWS Certificate Manager (ACM)**: *ACM* is deployed in each region to manage SSL certificates for HTTPS access and TLS certificates for internal communication between AWS services, both within and across regions.
- **Enhanced Security Measures**: *AWS WAF* and *AWS Shield* are integrated to protect against DDoS attacks and other security threats. These services are applied globally across all AWS resources.

For user authentication, *Amazon Cognito* plays a central role. User pools are region-specific, therefore a centralized *Amazon Cognito* instance can be set up in the region hosting the organization's main IT infrastructure or headquarters, providing Single Sign-On (SSO) with SAMLv2 for all globally deployed instances. This centralized approach simplifies identity management and maintains secure access across the system. Additionally, by hosting the cognito instance by the headquarters it is likely to minimise the sign-on latency for the largest proportion of employees.

Data storage follows a similar centralized model. A NoSQL key-value database using *Amazon DynamoDB* is employed to store sensitive employee data. Consolidating storage in a single region minimizes the risks associated with duplicating sensitive data across multiple regions. For added reliability, the storage hub can be replicated in a secondary region to provide disaster recovery and redundancy. Considering the legislative repercussions of this decision, is far outside the scope of the project.

To facilitate communication between distributed components, asynchronous messaging is implemented via *AWS AppSync* or *Amazon Simple Notification Service (SNS)*. This pub-sub model ensures reliable message delivery without risking data duplication, supporting fault-tolerant operations across regions.

This architectural strategy centralizes critical resources like authentication and storage while maintaining regional compute and traffic management capabilities. Such a setup ensures scalability, fault tolerance, and operational efficiency across the global infrastructure.

Figure 1 provides a visual overview of the system's architecture, summarizing the components and their interactions to ensure high availability, redundancy, and security.

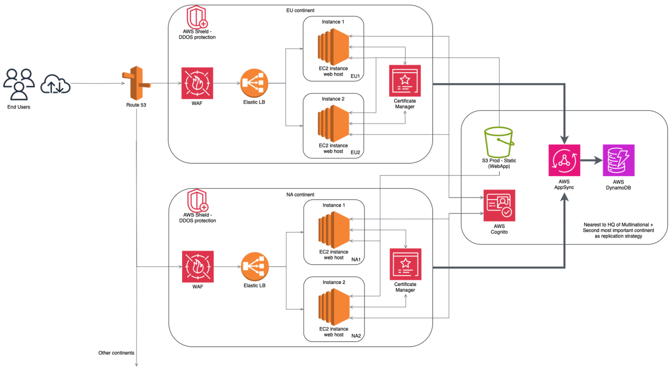


Fig. 1. Architecture proposal

IV. Implementation

A. *AWS* CloudFormation

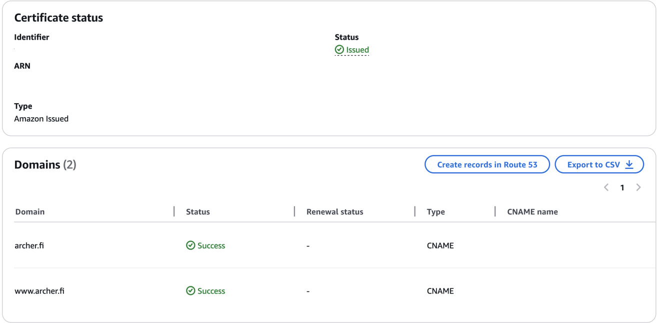
AWS CloudFormation is used to model the AWS architecture allowing for it to be quickly deployed. Templates, written in YAML or JSON, describe the required resources, such as Amazon EC2 instances or Amazon RDS DB instances, these templates can then be stored in version control tools to allow for different interactions of the architecture to be easily accessible. The provisioning and configuration of these resources are handled automatically by *CloudFormation*, removing the need for manual setup and dependency management. This approach ensures infrastructure is deployed consistently and reliably.

For this project, we have submitted YAML *CloudFormation* templates to deploy several AWS services to fit our architecture. These templates include configurations for EC2 instances integrated with AWS Certificate Manager, an Elastic Load Balancer to manage traffic across the EC2 instances, and the setup of Amazon Cognito with a fixed login URL to support Single Sign-On (SSO) functionality. We have not configured *Route 53* due to limitations with the AWS accounts provided for the module, however this is a small component of the architecture and could easily for example be replaced by Cloudflare's DNS service.

B. EC2 instances, ELB, WAF and Certificate handling

For the regional compute stacks the *CloudFormation* template matches that of the proposed architecture.

It creates two EC2 instances with an Ubuntu AMI. This is accompanied by an ELB with HTTP listeners set to listen for HTTPS and SSH traffic. HTTPS requires an SSL certificate so a certificate manager is included. To run the template a reference to an AWS certificate must be passed to AWS. We generated a certificate using a domain we owned however, this domain is not otherwise used.



Certificate status

Identifier

Status
Issued

ARN

Type
Amazon Issued

Domains (2)

Domain	Status	Renewal status	Type	CNAME name
archer.fi	Success	-	CNAME	
www.archer.fi	Success	-	CNAME	

Fig. 2. AWS Certificate

C. Cognito

AWS Cognito is a service designed to address two key aspects of identity and access management:

- Determining identity (authentication)
- Controlling access (authorization)

These concerns are managed through two primary features of Cognito: *User Pools* (authentication) and *Identity Pools* (authorization).

User Pools provide functionality for tasks such as user registration, login, and account recovery. Integration with third-party identity providers, including Facebook and Google, is also supported, allowing for a more flexible approach to authentication.

By contrast, *Cognito Identity Pools* enable users to be authorized for access to AWS services. For instance, permissions can be granted to upload files to an S3 bucket or invoke an API Gateway endpoint by configuring an Identity Pool.

In the demonstration, a *User Pool* was created with a straightforward login page to allow users to sign up. The configuration was completed using the provided YAML template located in the root of the repository.

V. Conclusion

In conclusion, while our demonstration is minimal, it shows the viability of our cloud-based architecture and the benefits of hosting the application on the cloud.

References

AWS (2024). *Amazon EC2 Instance types*. URL: https://aws.amazon.com/ec2/instance-types/ (Accessed 28. Nov. 2024).

Ismailov, Shukhrat (Feb. 29, 2024). *AWS Cognito and AWS IAM (Identity and Access Management)*. URL: https://medium.com/@shukhrat.ismailov05/aws-cognito-and-aws-iam-identity-and-access-management-7704c73613d9 (Accessed 28. Nov. 2024).

Minbaeva, Dana and Steen Erik Navrbjerg (2011). *Employment Practices of Multinational Companies in Denmark: Result Report*. English. Denmark: Københavns Universitet. ISBN: 978-87-91833-61-8.

Patil, Jayendra (Feb. 15, 2022). *AWS High Availability & Fault Tolerance Architecture*. URL: https://jayendrapatil.com/aws-high-availability-fault-tolerance-architecture-certification (Accessed 28. Nov. 2024).