CheeseHead Hosting

Cheesehead hosting

Technical Manual

Author: Group 1 Date: 10/01/2024 Version: 1.0

Table of Contents

Introduction

This document provides step-by-step instructions for setting up, maintaining, and troubleshooting various components of our network. The document contains instructions on IAM policies & roles, network configuration, security groups and instances, API Gateway, Step Functions, Lambda Functions, ECS, DNS, CloudWatch, and CloudTrail.

1. IAM policies & roles

1.1 Policies

In our network we have created three policies that we have used to grant permissions to AWS services to access resources.

1.1.1 ECStoEFS

The following policy grants ECS task definitions access to a specific EFS that will store the database from the Cheeseheadhosting website.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Policies* and then click on the *Create policy* button.
3. Choose the *JSON tab* to enter the JSON code.
4. Go to the GitLab repository and paste on the JSON code.
   NOTE: If you use another EFS then we use. Update the JSON code with the ARN of your EFS.
5. Click on the *Review policy* button.
6. Provide a *name* for example "ECStoEFS" and *description* for your policy.
7. Click on the *Create policy* button to create the policy.

1.1.2 ECStoEFSgeneral

The following policy grants ECS task definitions access to EFS that will store the customer databases from the website once they deploy an environment.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Policies* and then click on the *Create policy* button.
3. Choose the *JSON tab* to enter the JSON code.
4. Go to the GitLab repository and paste on the JSON code.
   NOTE: the ARN can be different because of the account ID please update the ARN with the correct one.
5. Click on the *Review policy* button.
6. Provide a *name* for example "ECStoEFSgeneral" and *description* for your policy.
7. Click on the *Create policy* button to create the policy.

1.2.3 LambdaSSM

The following policy is used to grant Lambda access to certain SSM commands and check the instance status. This policy is used to trigger an Ansible script on the operational EC2 host.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Policies* and then click on the *Create policy* button.
3. Choose the *JSON tab* to enter the JSON code.
4. Go to the GitLab repository and paste on the JSON code.
5. Click on the *Review policy* button.
6. Provide a *name* for example "LambdaSSM" and *description* for your policy.
7. Click on the *Create policy* button to create the policy.

1.2.4 VPC-Flow-logs

The following policy is used to grand VPC permissions to store the logs in an CloudWatch log group.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Policies* and then click on the *Create policy* button.
3. Choose the *JSON tab* to enter the JSON code.
4. Go to the GitLab repository and paste on the JSON code.
5. Click on the *Review policy* button.
6. Provide a *name* for example "VPC-Flow-logs" and *description* for your policy.
7. Click on the *Create policy* button to create the policy.

1.2 Roles

1.2.1 ecsTaskECStoEFSgeneral

The following IAM role is used by ECS tasks definitions to get access to EFS in order to store the customer databases on EFS.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Role* and then click on the *Create role* button.
3. On step 1 of select trust entity, select *AWS service*.
4. In the use case select *Elastic Container service* and click on *Elastic Container Service task.*
5. Click on the *next* button on the bottom right of the page.
6. Attach the following permission policies *AmazonECSTaskExecutionRolePolicy* & *ECStoEFS*
7. Click on the *next* button on the bottom right of the page.
8. Provide a *name* for example "ecsTaskECStoEFSgeneral" and *description* for your policy.
9. Click on the *Create role* button to create the role.

1.2.2 ecsTaskECStoEFS

The following IAM role is used by ECS task definition to get access to a specific EFS to store (read/write) the cheeseheadhosting database to an specific EFS.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Role* and then click on the *Create role* button.
3. On step 1 of select trust entity, select *AWS service*.
4. In the use case select *Elastic Container service* and click on *Elastic Container Service task.*
5. Click on the *next* button on the bottom right of the page.
6. Attach the following permission policies *AmazonECSTaskExecutionRolePolicy* & *ECStoEFSgeneral*
7. Click on the *next* button on the bottom right of the page.
8. Provide a *name* for example "ecsTaskECStoEFS" and *description* for your policy.
9. Click on the *Create role* button to create the role.

1.2.3 LambdaSSMrole

The following role is used by an AWS lambda function called LambdaTriggerTerraformSSM this Lambda function triggers the Ansible script on an EC2 instance by checking if the instance is running and send an SSM command to run the Ansible script.

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Role* and then click on the *Create role* button.
3. On step 1 of select trust entity, select *AWS service*.
4. In the use case select *Lambda*.
5. Click on the *next* button on the bottom right of the page.
6. Attach the following permission policies.
   - AmazonSESfullAccess
   - AmazonSSMFullAccess
   - AmazonSSMManagedIstanceCore
   - AWSLambdaBasicExecutionRole
   - LambdaSSM
7. Click on the *next* button on the bottom right of the page.
8. Provide a *name* for example "LambdaSSMrole" and *description* for your policy.
9. Click on the *Create role* button to create the role.

1.2.4 VPC Flow logs

1. Open the AWS Management Console and go to the IAM dashboard.
2. In the left-hand menu, click on *Role* and then click on the *Create role* button.
3. On step 1 of select trust entity, select *AWS service*.
4. In the use case select *Lambda*.
5. Click on the *next* button on the bottom right of the page.
6. Attach the following permission policies.
   - CloudWatchEventsBuiltInTargetExecutionAccess
   - CloudWatchEventsInvocationAccess
   - VPC-flow-logs
7. Click on the *next* button on the bottom right of the page.
8. Provide a *name* for example "VPC-flow-logs" and *description* for your policy.
9. Click on the *Create role* button to create the role.
10. Select the newly created policy
11. Go to trust relationships
12. Click on edit trust policy
13. Replace "service": "lambda.amazonaws.com" with "Service": "vpc-flow-logs.amazonaws.com"
14. Click on save.

2. Network configuration.

2.1 VPCs

Within the network there are 4 VPCs created to separate the CheeseHeadHosting network from the customer network. There also is a separation between public and private VPCs to increase the security of the network.

2.1.1 Cheesehead hosting VPC public.

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *VPC* and then click on the *Create VPC* button.
3. Provide a meaningful name in the *Name tag* field such as Cheesehead hosting VPC public.
4. Specify the IPv4 CIDR block for your VPC, in this case 172.16.0.0/16
5. Select no IPv6 CIDR block under IPv6 CIDR block.
6. Under Tenancy select *default*.
7. On the bottom-right click on *create VPC*.
8. Find the VPC you have just created. Select the VPC and click on *action* and click on *edit VPC settings*.
9. Under DNS settings check the boxes for *Enable DNS resolution* and *Enable DNS hostnames*.
10. On the bottom-right click *Save*.

2.1.1.1 VPC flowlogs

In the network there are 3 VPC flow logs configured to capture and monitor traffic.

2.1.1.1.1 VPC flow logs accept

1. Select the *VPC* for which you want to create *flow logs*.
2. In the left *navigation pane*, choose *Flow Logs*.
3. Click on "Create Flow Log."
4. Provide a name for your flow log, e.g., vpc-flow-accept.
5. Choose *ACCEPT* as the *Traffic Type.*
6. Set the *retention* period to *10 minutes*.
7. Set the *destination* log group to *send to CloudWatch logs.*
8. Choose the *IAM role VPC-flow-logs* that has the necessary permissions.
9. Choose a log file format, choose *text.*
10. Click *Create Flow Log*.

2.1.1.1.1 VPC flow logs reject

1. Select the *VPC* for which you want to create *flow logs*.
2. In the left *navigation pane*, choose *Flow Logs*.
3. Click on "Create Flow Log."
4. Provide a name for your flow log, e.g., vpc-flow-reject.
5. Set the *retention* period to *10 minutes*.
6. Choose *REJECT* as the *Traffic Type.*
7. Set the *destination* log group to *send to CloudWatch logs.*
8. Choose the *IAM role VPC-flow-logs* that has the necessary permissions.
9. Choose a log file format, choose *text.*
10. Click *Create Flow Log*.

2.1.1.1.1 VPC flow logs all

1. Select the *VPC* for which you want to create *flow logs*.
2. In the left *navigation pane*, choose *Flow Logs*.
3. Set the *filter* to capture all traffic.
4. Set the *retention* period to *10 minutes*.
5. Click on "Send to an Amazon S3 bucket."
6. Provide the ARN of the VPC-flow-log bucket.
7. Choose a *log record format* choose *AWS default format.*
8. Choose a log file format, choose *text.*
9. Set the box for *partion logs by time* to *every 24 hours.*
10. Click *Create Flow Log*.

2.1.2 Cheesehead hosting VPC Private.

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *VPC* and then click on the *Create VPC* button.
3. Provide a meaningful name in the *Name tag* field such as Cheesehead hosting VPC private.
4. Specify the IPv4 CIDR block for your VPC, in this case 172.17.0.0/16
5. Select no IPv6 CIDR block under IPv6 CIDR block.
6. Under Tenancy select *default*.
7. On the bottom-right click on *create VPC*.
8. Find the VPC you have just created. Select the VPC and click on *action* and click on *edit VPC settings*.
9. Under DNS settings check the boxes for *Enable DNS resolution* and *Enable DNS hostnames*.
10. On the bottom-right click *Save*.

2.1.3 Customer_VPC_public

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *VPC* and then click on the *Create VPC* button.
3. Provide a meaningful name in the *Name tag* field such as Customer_VPC_private.
4. Specify the IPv4 CIDR block for your VPC, in this case 10.1.0.0/16.
5. Select no IPv6 CIDR block under IPv6 CIDR block.
6. Under Tenancy select *default*.
7. On the bottom-right click on *create VPC*.
8. Find the VPC you have just created. Select the VPC and click on *action* and click on *edit VPC settings*.
9. Under DNS settings check the boxes for *Enable DNS resolution* and *Enable DNS hostnames*.
10. On the bottom-right click *Save*.

2.1.4 Customer_VPC_private

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *VPC* and then click on the *Create VPC* button.
3. Provide a meaningful name in the *Name tag* field such as Customer_VPC_private.
4. Specify the IPv4 CIDR block for your VPC, in this case 10.0.0.0/16.
5. Select no IPv6 CIDR block under IPv6 CIDR block.
6. Under Tenancy select *default*.
7. On the bottom-right click on *create VPC*.
8. Find the VPC you have just created. Select the VPC and click on *action* and click on *edit VPC settings*.
9. Under DNS settings check the boxes for *Enable DNS resolution* and *Enable DNS hostnames*.
10. On the bottom-right click *Save*.

2.2 subnets

There are various subnets in the CheeseHeadHosting network to create a separation between operational instances, the subnets for websites, database, and the NAT instance. This will allow for additional security.

2.2.1 Operational subnet

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *subnets* and then click on the *Create subnet* button.
3. For VPC ID select the *cheesehead hosting VPC private* ID.
4. Provide a meaningful name in the *Name tag* field such as operational subnet.
5. Availability zone can be set to *no preference*.
6. Provide an IPv4 subnet CIDR block, (e.g. 172.17.2.0/28)
7. In the right bottom corner click on create subnet.

2.2.2 Cheesehead-website-subnet-1b

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *subnets* and then click on the *Create subnet* button.
3. For VPC ID select the *cheesehead hosting VPC public* ID.
4. Provide a meaningful name in the *Name tag* field such as cheesehead-website-subet-1b.
5. Set the availability zone to eu-west-1b.
6. Provide an IPv4 subnet CIDR block, (e.g. 172.16.1.0/28)
7. In the right bottom corner click on create subnet.

2.2.3 cheesehead-website-subnet-1c

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *subnets* and then click on the *Create subnet* button.
3. For VPC ID select the *cheesehead hosting VPC public* ID.
4. Provide a meaningful name in the *Name tag* field such as cheesehead-website-subet-1b.
5. Set the availability zone to eu-west-1c.
6. Provide an IPv4 subnet CIDR block, (e.g. 172.16.3.0/28)
7. In the right bottom corner click on create subnet.

2.2.4 database-subnet

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *subnets* and then click on the *Create subnet* button.
3. For VPC ID select the *cheesehead hosting VPC private* ID.
4. Provide a meaningful name in the *Name tag* field such as database-subnet.
5. Set the availability zone to eu-west-1c.
6. Provide an IPv4 subnet CIDR block, (e.g. 172.17.1.0/28)
7. In the right bottom corner click on create subnet.

2.2.5 NAT subnet

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *subnets* and then click on the *Create subnet* button.
3. For VPC ID select the *cheesehead hosting VPC private* ID.
4. Provide a meaningful name in the *Name tag* field such as NAT subnet.
5. Set the availability zone to eu-west-1c.
6. Provide an IPv4 subnet CIDR block, (e.g. 172.17.254.0/28)
7. In the right bottom corner click on create subnet.

2.3 Internet gateways

There are four internet gateways in the network that allow communication to the internet. All the internet gateways are attached to different VPCs.

2.3.1 Cheesehead VPC public igw

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *internet gateways* and then click on the *Create internet gateway* button.
3. Enter a name for the Internet Gateway such as Cheesehead VPC public igw.
4. On the bottom left click on "Create Internet Gateway."
5. Select the newly created internet gateway from the list.
6. Click on the *action button* and select *attach to VPC* button.
7. In the pop-up window, select the cheesehead hosting VPC public.
8. Finally click on *attach internet gateway*.

2.3.2 Cheesehead VPC private NAT igw

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *internet gateways* and then click on the *Create internet gateway* button.
3. Enter a name for the Internet Gateway such as Cheesehead VPC private NAT igw.
4. On the bottom left click on "Create Internet Gateway."
5. Select the newly created internet gateway from the list.
6. Click on the *action button* and select *attach to VPC* button.
7. In the pop-up window, select the cheesehead hosting VPC private.
8. Finally click on *attach internet gateway*.

2.3.3 Customer_IGW_Public

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *internet gateways* and then click on the *Create internet gateway* button.
3. Enter a name for the Internet Gateway such as Customer_IGW_Public.
4. On the bottom left click on "Create Internet Gateway."
5. Select the newly created internet gateway from the list.
6. Click on the *action button* and select *attach to VPC* button.
7. In the pop-up window, select the Customer_VPC_Public.
8. Finally click on *attach internet gateway*.

2.3.4 Customer_IGW_Private

1. Open the AWS Management Console and go to *VPC*.
2. In the left-hand menu, click on *internet gateways* and then click on the *Create internet gateway* button.
3. Enter a name for the Internet Gateway such as Customer_IGW_Private.
4. On the bottom left click on "Create Internet Gateway."
5. Select the newly created internet gateway from the list.
6. Click on the *action button* and select *attach to VPC* button.
7. In the pop-up window, select the Customer_VPC_ Private.
8. Finally click on *attach internet gateway*.

2.4 VPC peering.

To enable communication between the 2 VPC's (public and private) a Peering connection is needed. The configuration is as follows:

1. In AWS go to VPC
2. Under Virtual Private cloud go to peering connections
3. Create a new peering connection.
4. Name the peering connection and select the 2 VPC's you want to connect (requester and accepter)
5. Create the peering connection.
6. In the overview select the newly created peering connection and click on actions > Accept request
7. The peering connection has been configured.

Repeat these steps for both the client and hosting VPC's.

2.5 Routing tables

Our environment has 3 manually made routing tables. These are as follows:

2.5.1 VPC public routing table

This routing table includes routing to the peering gateway and internet gateway. Like the name suggest it is created in the public VPC for websites etc.

To replicate this routing table, follow the following steps:

1. Go to AWS VPC and go to *Route tables* under *Virtual private cloud.*
2. Click *Create route table*. Name the table and select the appropriate VPC.
3. Click on *Create route table.*
4. Select the route table and go to *Routes.* Click on *Edit routes.*
5. Add the following routes to the routing table:

Destination	**Target**
0.0.0.0/0	<internet gateway VPC public>
172.16.0.0/16	Local
172.17.0.0/16	<peering connection>

2.5.2 VPC private database routing table

This routing table includes the routing to the peering gateway and the NAT instance's ENI for internet access.

To replicate this routing table, follow the following steps:

1. Go to AWS VPC and go to *Route tables* under *Virtual private cloud.*
2. Click *Create route table*. Name the table and select the appropriate VPC.
3. Click on *Create route table.*
4. Select the route table and go to *Routes.* Click on *Edit routes.*
5. Add the following routes to the routing table:

Destination	**Target**
0.0.0.0/0	<ENI NAT instance>
172.16.0.0/16	<peering connection>
172.17.0.0/16	Local

2.5.3 VPC private NAT routing table

This routing table routes the traffic in the NAT subnet which serves as a buffer between the internet and the private subnet. To make this work you must ensure that stop source / destination checking on the NAT instance is stopped. To check or do this follow the following steps:

1. Go to AWS EC2 and go to the instances tab.
2. Select your NAT instance from the list and click on actions > networking.
3. Select *Change Source / destination check.*
4. Check *Stop* under Source / destination checking and click on *Save*.

To replicate this routing table, follow the following steps:

1. Go to AWS VPC and go to *Route tables* under *Virtual private cloud.*
2. Click *Create route table*. Name the table and select the appropriate VPC.
3. Click on *Create route table.*
4. Select the route table and go to *Routes.* Click on *Edit routes.*
5. Add the following routes to the routing table:

Destination	**Target**
0.0.0.0/0	<Internet gateway NAT subnet>
172.16.0.0/16	<peering connection>
172.17.0.0/16	Local

3. Security group and instances

3.1 Security groups

3.1.1 Cheesehead-db-SG

1. Open the AWS Management Console and go to *EC2*.
2. In the left-hand menu, click on *security groups* and then click on the *Create security group* button.
3. Enter a name for the security group (e.g cheesehead-db-sg).
4. Add a description of the security group.
5. Add the following rules
    - MYSQL/Aurora - TCP 3306 - 172.17.1.0/28
        - Click on the "Add Rule" button.
        - Choose "MYSQL/Aurora" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "3306" for the Port Range.
        - Enter "172.17.1.0/28" for the Source.
    - Rule 2: MYSQL/Aurora - TCP 3306 - 172.17.254.0/28
        - Click on the "Add Rule" button.
        - Choose "MYSQL/Aurora" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "3306" for the Port Range.
        - Enter "172.17.254.0/28" for the Source.
    - Rule 3: SSH - TCP 22 - 172.17.254.0/28
        - Click on the "Add Rule" button.
        - Choose "MYSQL/Aurora" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "3306" for the Port Range.
        - Enter "172.17.254.0/28" for the Source.
    - Rule 4: All ICMP - IPv4 - ICMP All - 172.17.254.0/28
        - Click on the "Add Rule" button.
        - Choose "MYSQL/Aurora" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "3306" for the Port Range.
        - Enter "172.17.254.0/28" for the Source.

    - Rule 5: MYSQL/Aurora - TCP 3306 - 172.16.1.0/28
        - Click on the "Add Rule" button.
        - Choose "MYSQL/Aurora" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "3306" for the Port Range.
        - Enter "172.17.254.0/28" for the Source.
    - Rule 6: All Traffic - All - All - 0.0.0.0/0 (Outbound)
        - Click on the "Add Rule" button.
        - Choose "All Traffic" for the Type.
        - Choose "All" for the Protocol.
        - Enter "All" for the Port Range.
        - Enter "0.0.0.0/0" for the Source.
6. On the right bottom click on create security group.

3.1.2 Cheesehead-NAT-SG

1. Open the AWS Management Console and go to *EC2*.
2. In the left-hand menu, click on *security groups* and then click on the *Create security group* button.
3. Enter a name for the security group (e.g cheesehead-NAT-sg).
4. Add a description of the security group.
5. Add the following rules
    - Rule 1: SSH - TCP 22 - 0.0.0.0/0
        - Click on the "Add Rule" button.
        - Choose "SSH" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "22" for the Port Range.
        - Enter "0.0.0.0/0" for the Source.
    - Rule 2: All Traffic - All - All - 172.17.2.0/28
        - Click on the "Add Rule" button in the Outbound Rules section.
        - Choose "All Traffic" for the Type.
        - Choose "All" for the Protocol.
        - Enter "All" for the Port Range.
        - Enter "172.17.2.0/28" for the Destination.

- Rule 3: All Traffic - All - All - 172.17.1.0/28
  - Click on the "Add Rule" button in the Outbound Rules section.
  - Choose "All Traffic" for the Type.
  - Choose "All" for the Protocol.
  - Enter "All" for the Port Range.
  - Enter "172.17.1.0/28" for the Destination.
- Rule 4: All ICMP - IPv4 - ICMP All - 0.0.0.0/0
  - Click on the "Add Rule" button in the Outbound Rules section.
  - Choose "All ICMP - IPv4" for the Type.
  - Choose "ICMP" for the Protocol.
  - Enter "All" for the Port Range.
  - Enter "0.0.0.0/0" for the Destination.
- Rule 6: All Traffic - All - All - 0.0.0.0/0 (Outbound)
  - Click on the "Add Rule" button.
  - Choose "All Traffic" for the Type.
  - Choose "All" for the Protocol.
  - Enter "All" for the Port Range.
  - Enter "0.0.0.0/0" for the Source.

6. On the right bottom click on create security group.

Cheesehead-op-SG

1. Open the AWS Management Console and go to *EC2*.
2. In the left-hand menu, click on *security groups* and then click on the *Create security group* button.
3. Enter a name for the security group (e.g cheesehead-op-sg).
4. Add a description of the security group.
5. Add the following rules
   - Rule 1: All TCP - TCP 0 - 65535 - 0.0.0.0/0*

     This rule was made to allow Lambda to trigger the instance for testing.

     - Click on the "Add Rule" button.
     - Choose "All TCP" for the Type.
     - Choose "TCP" for the Protocol.
     - Enter "0 - 65535" for the Port Range.
     - Enter "0.0.0.0/0" for the Source.

- Rule 2: SSH - TCP 22 - 172.17.2.0/28
    - Click on the "Add Rule" button.
    - Choose "SSH" for the Type.
    - Choose "TCP" for the Protocol.
    - Enter "22" for the Port Range.
    - Enter "172.17.2.0/28" for the Source.
- Rule 3: SSH - TCP 22 - 172.17.254.0/28
    - Click on the "Add Rule" button.
    - Choose "SSH" for the Type.
    - Choose "TCP" for the Protocol.
    - Enter "22" for the Port Range.
    - Enter "172.17.254.0/28" for the Source.
- Rule 4: All Traffic - All - All - 0.0.0.0/0 (Outbound)
    - Click on the "Add Rule" button.
    - Choose "All Traffic" for the Type.
    - Choose "All" for the Protocol.
    - Enter "All" for the Port Range.
    - Enter "0.0.0.0/0" for the Source.

6. On the right bottom click on create security group.

Cheesehead-web-SG

1. Open the AWS Management Console and go to *EC2*.
2. In the left-hand menu, click on *security groups* and then click on the *Create security group* button.
3. Enter a name for the security group (e.g cheesehead-web-sg).
4. Add a description of the security group.
5. Add the following rules.
    - Rule 1: MYSQL/Aurora - TCP 3306 - 172.17.1.0/28
        - Click on the "Add Rule" button.
        - Choose "MYSQL/Aurora" for the Type.
        - Choose "TCP" for the Protocol.
        - Enter "3306" for the Port Range.
        - Enter "172.17.1.0/28" for the Source.

- Rule 2: Custom TCP - TCP 21-22 (sFTP) - 0.0.0.0/0
  - Click on the "Add Rule" button.
  - Choose "Custom TCP" for the Type.
  - Choose "TCP" for the Protocol.
  - Enter "21-22" for the Port Range.
  - Enter "0.0.0.0/0" for the Source.
  - Optionally, enter "Allow (s)FTP traffic" as a description.
- Rule 3: SSH - TCP 22 - 172.17.254.0/28 (Inbound)
  - Click on the "Add Rule" button.
  - Choose "SSH" for the Type.
  - Choose "TCP" for the Protocol.
  - Enter "22" for the Port Range.
  - Enter "172.17.254.0/28" for the Source.
- Rule 4: HTTPS - TCP 443 - 0.0.0.0/0
  - Click on the "Add Rule" button.
  - Choose "HTTPS" for the Type.
  - Choose "TCP" for the Protocol.
  - Enter "443" for the Port Range.
  - Enter "0.0.0.0/0" for the Source.
- Rule 5: HTTP - TCP 80 - 0.0.0.0/0
  - Click on the "Add Rule" button.
  - Choose "HTTP" for the Type.
  - Choose "TCP" for the Protocol.
  - Enter "80" for the Port Range.
  - Enter "0.0.0.0/0" for the Source.
- Rule 6: All Traffic - All - All - 0.0.0.0/0 (Outbound)
  - Click on the "Add Rule" button.
  - Choose "All Traffic" for the Type.
  - Choose "All" for the Protocol.
  - Enter "All" for the Port Range.
  - Enter "0.0.0.0/0" for the Source.

6. On the right bottom click on create security group.

EFS-SG

1. Open the AWS Management Console and go to *EC2*.
2. In the left-hand menu, click on *security groups* and then click on the *Create security group* button.
3. Enter a name for the security group (e.g EFS-SG).
4. Add a description of the security group.
5. Add the following rules.
    - Rule 1: All Traffic - All - All - 0.0.0.0/0 (Inbound)
        - Click on the "Add Rule" button.
        - Choose "All Traffic" for the Type.
        - Choose "All" for the Protocol.
        - Enter "All" for the Port Range.
        - Enter "0.0.0.0/0" for the Source.
    - Rule 2: All Traffic - All - All - 0.0.0.0/0 (Outbound)
        - Click on the "Add Rule" button.
        - Choose "All Traffic" for the Type.
        - Choose "All" for the Protocol.
        - Enter "All" for the Port Range.
        - Enter "0.0.0.0/0" for the Source.
6. On the right bottom click on create security group.

3.2 EFS

1. Open the AWS Management Console and go to *EFS.*
2. Click on *Create file system* to begin creating a new EFS.
3. Click on *Customize*.
4. Enter a name for your EFS (E.G, Project-database).
5. Set the *file system type* to *Regional.*
6. *Check* the box for *enable automated backups.*
7. Leave the rest of the settings default & click next on the right bottom side.
8. When selecting a VPC choice for the private VPC of your network, in this case it will be cheesehead hosting VPC private.
9. Select a security group that allows inbound traffic on port 2049, following this manual this is EFS-SG.
10. Finally click create at the *right bottom* corner.

3.3 EC2 Instances

3.3.1 NAT instance

1. Navigate to EC, click on “launch instance”, give it a name and select the “ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20230516” AMI. For instance, type choose t2.micro, within the menu create a new key pair to access the instance. Scroll down to the network settings, click on edit, choose the correct VPC (cheesehead VPC private), subnet and security groups (cheesehead-NAT-SG). Make sure storage is set to 8 GiB of GP2 and click on “Launch instance”.

2. After launching the instance select the instance.

3. Click on *actions*.

4. Click on *networking*.

5. Click on *change source/destination check*.

6. *Check* the box for *Stop* and click *save*.

3.3.2 Operational instance

1. Navigate to EC, click on “launch instance”, give it a name and select the “Ubuntu 22.04” AMI. For instance, type choose t2.micro, within the menu create a new key pair to access the instance. Scroll down to the network settings, click on edit, choose the correct VPC (cheesehead VPC private), subnet and security groups (cheesehead-op-SG). Make sure storage is set to 8 GiB of GP2 and click on “Launch instance”.

2. Use SSH to log into the NAT instance, copy the newly created SSH key from your system to the NAT instance and again use SSH to log into the OP instance. Execute the following commands:

- curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
- unzip awscliv2.zip
- sudo ./aws/install
- $ UBUNTU_CODENAME=jammy
- $ wget -O- "https://keyserver.ubuntu.com/pks/lookup?fingerprint=on&op=get&search=0x6125E2A8C77F2818FB7BD15B93C4A3FD7BB9C367" | sudo gpg --dearmour -o /usr/share/keyrings/ansible-archive-keyring.gpg
- $ echo "deb [signed-by=/usr/share/keyrings/ansible-archive-keyring.gpg] http://ppa.launchpad.net/ansible/ansible/ubuntu $UBUNTU_CODENAME main" | sudo tee /etc/apt/sources.list.d/ansible.list
- $ sudo apt update && sudo apt install ansible

3. For installing Terraform follow the official guide on this website, https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli .

4. Create a folder called “Project”, in here create a file called "ECSclient.tf” and another called “ch-playbook.yaml” and paste the code from the GIT repository in these files.

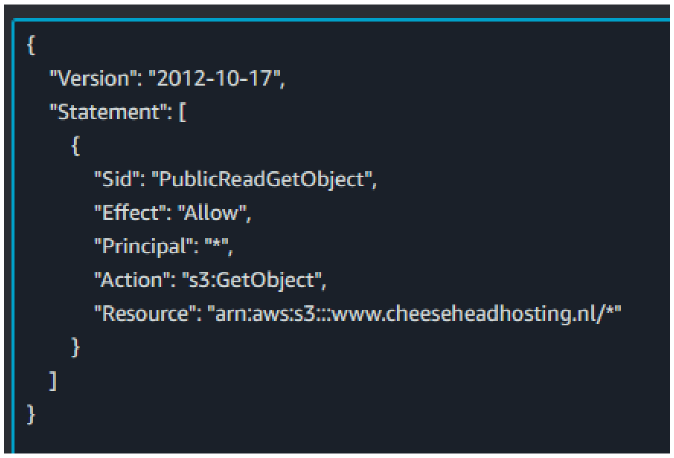
3.3 S3

www.cheeseheadhosting.nl bucket

This bucket hosts the maintenance site which gets shown if all website servers are down. This bucket is linked to the www.cheeseheadhosting.nl failover record.

To recreate this bucket follow the following steps:

1. Go to AWS S3
2. Click on *Create Bucket*
3. Name the new bucket www.cheeseheadhosting.nl and deselect block all public access
4. Leave the rest default and click on *Create Bucket*
5. Click on the newly created bucket and go to the *Properties* tab
6. Scroll down to *Static website hosting* and click on *edit*
7. Under static website hosting select *Enable,* under hosting type select *Host a static website* and specify *index.html* under *Index document*.
8. Click *Save changes*
9. Go to the *Permissions* tab
10. Under *Bucket policy* click edit and change the policy so it mirrors the one below:



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::www.cheeseheadhosting.nl/*"
    }
  ]
}
```

11. Click on *Save changes*
12. Go back to the *Objects* tab
13. Click on *Upload* and add the index.html file
14. Click on *Upload* again

snoweek8 bucket

1. Navigate to AWS S3, click on “Create bucket”, enter the name of the bucket and check “block all public access”, click on “Create bucket”.

Ch-bt01bucket

1. Navigate to AWS S3, click on “Create bucket”, enter the name of the bucket and check “block all public access”, click on “Create bucket”.

4. API Gateway

4.1 Create the API Gateway

1. Navigate to “API Gateway” on the AWS console.
2. Scroll down to “REST API” and click on “Build”.
3. Provide a name and choose “Regional” as endpoint type.

4.2 Configure the API

1. Click on the newly created API and create a resource inside with the name and path of “Website”.

2. Within this resource, navigate to the POST method and edit the integration request.

3. Select “AWS Service”, the region your state machine is located in and select the “Step Functions” service.

4. Add the right execution role as you have created in chapter 1, set “Content Handling” to “Passthrough” and click on “Save”.

5. Navigate to “Stages” and click on “Create Stage”, set a name and click on “Save”.

6. Go back your API and click on “Deploy”, select your newly created stage and click “Deploy”.

5. Step functions

5.1 Step function

1. Create a new state machine

2. edit the state machine

3. open the *code* tab

4. paste the contents of the *stepfunction.json* in git into the left panel

5. Save

Note: This step function makes use of several Lambda scripts which it calls using specific ARN's. Please alter these within the step designer so it uses **the correct Lambda ARN's of your environment**.

5.2 Related Lambda scripts

Our step function is built to use the following lambda scripts & roles. For more information see chapter 6.

LamdaTriggerTerraformSSM

This function triggers the ansible and terraform script running on the operator instance using instance ID. It uses the lambdassmrole role to connect to the operator EC2 instance and execute the relevant commands.

Note: When working with this lambda make sure it has access to SSM and/or uses the specified lamdassmrole permissions.

Project-CheckFile-EC2

This function checks the existence of the error file that could be generated after LamdaTriggerTerraformSSM has ran. If this file is detected the step function marks it as an error and sends the failure mail. If not, it continues and sends the success mail to the client.

Note: This lambda script also needs access to SSM to check the filesystem for the error file. In our current situation it also makes use of the lambdassm role.

Project-Send-Email

This function is used to send the e-mail to the client. It uses the default lambda role made upon creation which only has permissions to create logs in CloudWatch. When working with this function within the step machine make sure that the payload is entered as specified in the Json code. Using this custom payload with the input payload it fills in the parts of the e-mail.

6. Lambda functions

6.1 Lambda for SSM

1. Navigate to “Lambda” in the AWS console.

2. Click on “Create function”.

3. Give the function the name” LambdaTriggerTerraformSSM”, select Python 3.11 as “Runtime”, leave the Architecture at x86_64 and click on “Create function”.

4. Navigate to the GIT page and copy the Python code in the “LambdaTriggerTerraformSSM” file that is located in the Lambda folder.

5. Go back to your Lambda function and paste the code you just copied inside and click “Deploy”.

6. Scroll up and click on “Add Trigger”, select API Gateway as a source, select “Use existing API”, click on the API you created in chapter 4 and click on “Add”.

6.2 Lambda for file check

1. Navigate to “Lambda” in the AWS console.

2. Click on “Create function”.

3. Give the function the name” Project-CheckFile-EC2”, select Python 3.11 as “Runtime”, leave the Architecture at x86_64 and click on “Create function”.

4. Navigate to the GIT page and copy the Python code in the “Project-CheckFile-EC2” file that is located in the Lambda folder.

5. Go back to your Lambda function and paste the code you just copied inside and click “Deploy”.

6.3 Lambda for sending Email.

1. Navigate to “Lambda” in the AWS console.

2. Click on “Create function”.

3. Give the function the name” Project-Send-Email”, select Python 3.11 as “Runtime”, leave the Architecture at x86_64 and click on “Create function”.

4. Navigate to the GIT page and copy the Python code in the “Project-Send-Email” file that is located in the Lambda folder.

5. Go back to your Lambda function and paste the code you just copied inside and click “Deploy”.

7. ECS

7.1 ECS task definitions

7.1.1 DataBase

1. Navigate to the GIT (using a terminal) and go to the Database directory inside the Docker folder, build a Docker image from the Docker file located there and push to Docker Hub.

2. Go to “Elastic Container Service” on the AWS console, navigate to “Task definitions” and click on “Create new task definition”.

3. Give the task a name, select “AWS Fargate” as launch type, allocate 0.5vCPU and 1GB memory, select the execution role you created in chapter 1, enter the name of your container and include the repository url and image with its tag. Open port 3306, make sure CloudWatch log collection is enabled, scroll down to “Volume 1”, give it a name, select “EFS” as volume type, select the file system you created, enter the directory and click “Create”.

7.1.2 Website

1. Navigate to the GIT (using a terminal) and go to the Website directory inside the Docker folder, build a Docker image from the Docker file located there and push to Docker Hub.

2. Go to “Elastic Container Service” on the AWS console, navigate to “Task definitions” and click on “Create new task definition”.

3. Give the task a name, select “AWS Fargate” as launch type, allocate 0.5vCPU and 1GB memory, select the execution role you created in chapter 1, enter the name of your container and include the repository url and image with its tag. Open port 443 and 80, add an environment variable with the key “MYSQL_DATABASE” and value “project-database.project-database”, make sure CloudWatch logging is enabled and click “Create”.

7.2 ECS cluster

1. Navigate to “Clusters” within ECS.

2. Click on “Create cluster”, give your cluster a name, select “AWS Fargate” as infrastructure and click on “Create”.

7.3 ECS services

7.3.1 Database service

1. Click on your newly created cluster, go to “Services” and click on “Create”, select “FARGATE” as capacity provider, specify “Service” as application type, select the task definition for the database, give a name to the service and set “desired tasks” to 1. Scroll down and enable “Service discovery”, enter a name for the Namespace, enter a name for the Service discovery name, make sure the DNS record is set to type A with a 15 second TTL. Scroll down to “Networking”, select the correct VPC, subnets, security group(s), disable “Public IP” and click “Create”.

7.3.1 Website service

1. Click on your newly created cluster, go to “Services” and click on “Create”, select “FARGATE” as capacity provider, specify “Service” as application type, select the task definition for the website, give a name to the service and set “desired tasks” to 1. Scroll down to “Networking”, select the correct VPC, subnets, security group(s), enable “Public IP” and click “Create”.

8. DNS

8.1 Route53

Within AWS Route53 we use the cheeseheadhosting.nl public hosted zone. To replicate this zone follow the following steps:

1. go to AWS Route53 and click on *Hosted zones*
2. Click on *Create hosted zone*
3. Under domain name fill in cheeseheadhosting.nl. Leave the rest on default and click on *Create hosted zone*
4. The hosted zone has been created

To replicate the apex record follow the following steps:

1. Go to the hosted zone and click on *Create record*
2. Select *Simple routing* and click on next
3. Click on *Define simple record*
4. Leave the subdomain field empty
5. Under *Value/Route traffic to* select *Alias to another record in this hosted zone*
6. Fill in *www.cheeseheadhosting.nl*
7. Click on *Define simple record*
8. Click on *Create records*

To replicate the wildcard record follow the following steps:

1. Go to the hosted zone and click on *Create record*
2. Select *Simple routing* and click on next
3. Click on *Define simple record*
4. Put a * in the subdomain field
5. Under *Value/Route traffic to* select *Alias to another record in this hosted zone*
6. Fill in *www.cheeseheadhosting.nl*
7. Click on *Define simple record*
8. Click on *Create records*

To replicate the web record follow the following steps:

1. Go to the hosted zone and click on *Create record*
2. Select *Simple routing* and click on next
3. Click on *Define simple record*
4. Put web in the subdomain field
5. Under *Value/Route traffic to* select *Alias to S3 website endpoint*
6. Select your region
7. Select your S3 endpoint (in our case s3-website.eu-central-1.amazonaws.com.)
8. Click on *Define simple record*
9. Click on *Create records*

Create the health check for the load balancer:

1. Under route 53 go to *Health checks*
2. Click on *Create health check*
3. Name the health check accordingly
4. Select *Domain name*
5. Fill in the DNS name of the load balancer under *Domain name*
6. Click on *Next*
7. Click on *Create health check*

To replicate the www failover record follow the following steps:

1. Go to the hosted zone and click on *Create record*
2. Select *Failover routing* and click on next
3. Put www in the subdomain field
4. Click on *Define failover record*
5. Under *Value/Route traffic to* select *Alias to Application and Classic Load Balancer*
6. Select your region
7. Select the load balancer
8. Under *Failover record type* choose *Primary*
9. Under *Health check ID* select the load balancers health check
10. Click on *Define failover record*
11. Click once more on *Define failover record*
12. Under *Value/Route traffic to* select *Alias to another record in this hosted zone*
13. Select *web.cheeseheadhosting.nl*
14. Set *Failover record type* to *Secondary*
15. Click on *Define failover record*
16. Finally, click on *Create records*

8.2 Certificate manager

8.2.1 cheeseheadhosting.nl

1. Open the AWS Management Console and go to *certificate manager*.
2. In the left-hand menu, click on *list certificates*.
3. Click on *Request*.
4. Select *request a public certificate* and click *Next*.
5. For *fully qualified domain name* fill in *cheeseheadhosting.nl*
6. For *validation method* click *email*.
7. Choose *RSA 2048* for *key algorithm*.
8. On the bottom right corner click on *request*.
9. Wait a few minutes and confirm the request you will receive in the mail.

8.2.2 *.cheeseheadhosting.nl

1. Open the AWS Management Console and go to *certificate manager*.
2. In the left-hand menu, click on *list certificates*.
3. Click on *Request*.
4. Select *request a public certificate* and click *Next*.
5. For *fully qualified domain name* fill in **.cheeseheadhosting.nl*
6. For *validation method* click *email*.
7. Choose *RSA 2048* for *key algorithm*.
8. On the bottom right corner click on *request*.
9. Wait a few minutes and confirm the request you will receive in the mail.

8.2.3 www.cheeseheadhosting.nl

1. Open the AWS Management Console and go to *certificate manager*.
2. In the left-hand menu, click on *list certificates*.
3. Click on *Request*.
4. Select *request a public certificate* and click *Next*.
5. For *fully qualified domain name* fill in www.*cheeseheadhosting.nl*
6. For *validation method* click *email*.
7. Choose *RSA 2048* for *key algorithm*.
8. On the bottom right corner click on *request*.
9. Wait a few minutes and confirm the request you will receive in the mail.

9. CloudWatch

1. Open the CloudWatch Service on AWS
2. Go to Dashboards
3. Click “General Dashboard. This is the appropriate dashboard that was made for the case study project.
4. Now you are able to see all the appropriate data that is properly named so you know what each.
5. VPC flow log analysis was added to monitor day to day activity of the company. This allows data to be set in the dashboard, but specific data can also be queried vis Athena as a table has already been created for it. Click “Athena” on the AWS home page, then when creating queries, ensure to use the “vpc_flow_logs” table.
6. According to the needs of the user, custom alarms can be implemented and custom thresholds can also be set.
    a. In the CloudWatch screen, click “In alarm”. Then click “Create Alarm”. Click “Select Metric”.
    b. In the following pop up screen, there are specific metrics that can be selected for alarms.
    c. During the configuration, you will be prompted to create a new SNS topic or select an existing one. You can create a new one if you please and name the topic and your email.
    d. You can also add lambda actions to this alarm if it is triggered to run a specific lambda function. If you want to add the DDoS alarm, the lambda function is named ddos-detection. You must also go to “Lambda” and add your SNS topic ARN within the code where it is prompted. If you would like to add your own CRON expression to this, go to “Amazon EventBridge”, click “Create Rule”. Give it a name and click the “Schedule” button. Then click “Continue to create rule” and enter the cron expression according to how many times you would like the lambda function to be run. After this is done, select the lambda function target by just searching “Lambda” within the target and enter the DDoS detection lambda function ARN which is “arn:aws:lambda:eu-west-1:349962368193:function:ddos-detection”. Once this is done, the rule can be created.

10. CloudTrail

Cloudtrail data is set to be stored in an S3 Bucket as well as imported to CloudWatch as a log group so it can be used in the Dashboard that can be accessed in the “CloudWatch” bookmark.

1. In order to access the CloudTrail trail, select “CloudTrail” on the AWS home page and then select the “Dashboard” tab. Then select “SNOWEEK16” and there you can see all of the data for the trail such as the bucket location.
2. In order to access the bucket, just click the link to the “Trail log location”” section In the general details for the CloudTrail trail after following step #1.
3. There is also an Athena table to allow queries to be done for future infrastructure engineers in the company that will be able to query data and set alarms, etc. according to the needs of the company
    a. Click “Athena” on the AWS home page.
    b. When making queries use the “cloudtrail_logs_snoweek8_cloudtrail” table. This was done because analyzing the logs from an S3 bucket makes it harder to see what is going on, so it can either be queried in Athena or analyzed via the CloudWatch dashboard.