**Technical design**

***Centralized Firewall***

*Semester 6 - Infrastructure*

***Fontys - Eindhoven***

**Text, logo, company name

Description automatically generated with medium confidence**

**Version**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Amendments** | **Status** |
| 1.0 | 11/14/2022 | Group 1 |  | In progress |
|  |  |  |  |  |

Table of Contents

[1. Introduction 3](#_Toc119504780)

[2. Network diagram 4](#_Toc119504781)

[3. Fontys Server - NetLab 5](#_Toc119504782)

[3.2 Terraform control 5](#_Toc119504783)

[4.1 Service: EC2 6](#_Toc119504784)

[4.1.1 Web Server 6](#_Toc119504785)

[4.1.2 Database 6](#_Toc119504786)

[4.1.3 Nat1 7](#_Toc119504787)

[4.2 Service: VPC 8](#_Toc119504788)

[4.2.3 Subnets 8](#_Toc119504789)

[4.2.4 Route tables 9](#_Toc119504790)

[4.2.5 Internet gateways 9](#_Toc119504791)

[5. Process diagram 10](#_Toc119504792)

# 1. Introduction

This document describes in technical detail what the multi-cloud environment with the firewall solution will look like based on the requirements we received. The specifications are explained and also why certain choices were made for the design. It is highly recommended to read our project plan before reading this document, because we explain the global information and requirements in detail and this document will be easier to understand with more knowledge about the project.

# 2. Network diagram

This is the first version of the Sogeti multi-cloud environment which we created for the first sprint. The goal of this sprint is to show to the stakeholder that that we can roll out the environment with IaC and apply firewall rules for the different VPC’s and subnets.

Diagram

Description automatically generated

Figure 1: First version of the cloud environment

The environment works as following. On the diagram we have two environments: The first area is a virtual environment from Fontys University called the NetLab. In this area we host server with Ansible and Terrafrom installed on it. From our Terraform control server we are able to apply and roll out changes to the AWS environment. The second area depicts the Sogeti cloud environment. This is the production network of our project. In the future this will contain a firewall for the Sogeti clients to connect to. The first public subnet contains a webserver which is available for public.

# 3. Fontys Server - NetLab

## 3.2 Terraform control

This virtual machine is used to run the Ansible/Terraform scripts. It is a central point from where the administrator can run numerous different scripts for employing instances on the AWS environment. The AWS CLI is installed on it and is connected to a the groups IAM account (Luuk or Alex).

Hostname: Terraform\_control\_vm

IP Address: 192.168.189.25/24

OS: Ubuntu 20.04.5 LTS

Since we are using a test environment, the Terrafrom control server is installed for now in the NetLab, due the fact that we want to manage the cloud environment remotely. When the firewall solution is going to be used in real-time, the control server will probably be installed in a different on-premise environment to manage the cloud environment remotely. For future use it is recommended to place the Terraform server in an on-site environment. This is because of the security risks if anyone could access the server.4. Amazon Web Services

We used the region Europe (Ireland) eu-west-1. This is due to a rule in the permissions on our AWS account and its not in our power to choose a region with better latency.

## 4.1 Service: EC2

This chapter contains and describes the different EC2 instances we use for our project. We use the different services to support the development of our cloud environment and test of the different resources are functioning as they should.

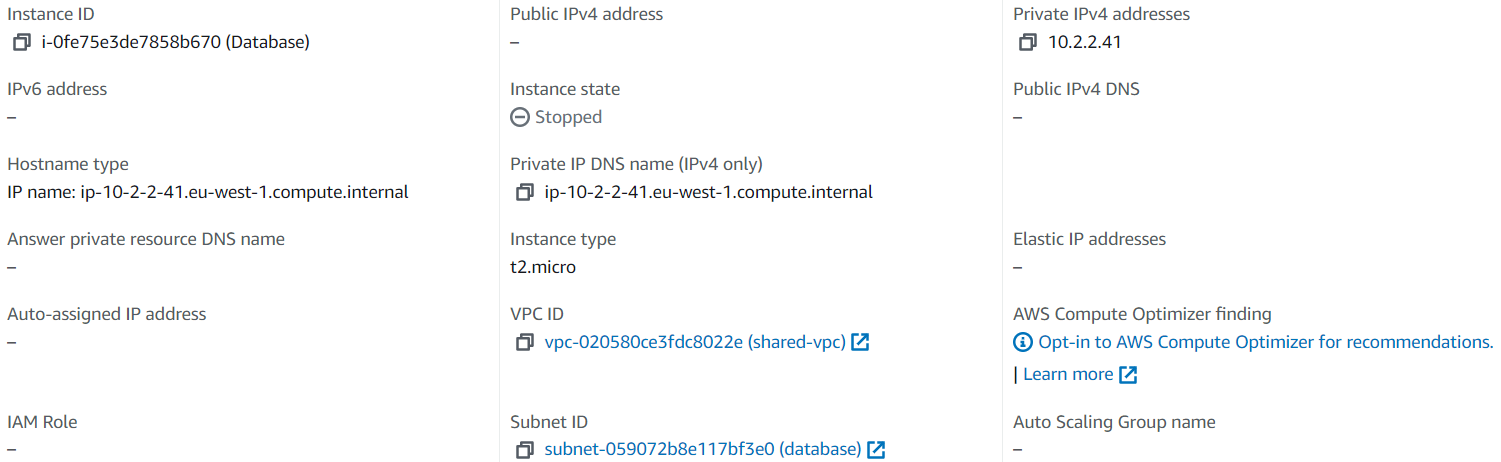
### 4.1.1 Web Server

Graphical user interface, text, application

Description automatically generated

This instance is for our public webserver. This EC2 instance helps us testing the AWS security rules. This compute instance is available in a public subnet so it should be accessible to public connections. In the future this instance could be removed.

### 4.1.2 Database



The database supports our webserver. At this moment the database is placed in a private subnet to show that this resource can only be accessed by the webserver. When the testing sprint is done, this resource can be removed from the environment.

### 4.1.3 Nat1

Graphical user interface, text, application

Description automatically generated

## 4.2 Service: VPC

**4.2.1 web-vpc**

Graphical user interface

Description automatically generated with medium confidence

This is the VPC we use for the webserver. This VPC is accessible to the public to view the content on the webserver. The webserver VPC serves a demo purpose to show that we can control access between the VPCs

**4.2.2 shared-vpc**

The database is in a private VPC disconnected from the public VPC. The database is not accessible for public view on the internet, because of security reasons.

A picture containing graphical user interface

Description automatically generated

### 4.2.3 Subnets

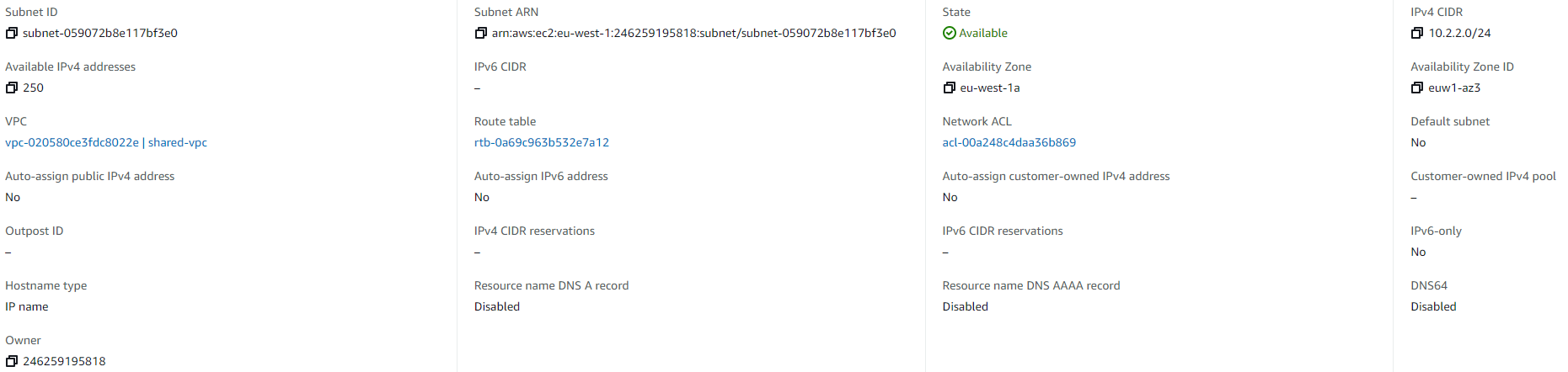
**Web-pub3**

Graphical user interface, application

Description automatically generated

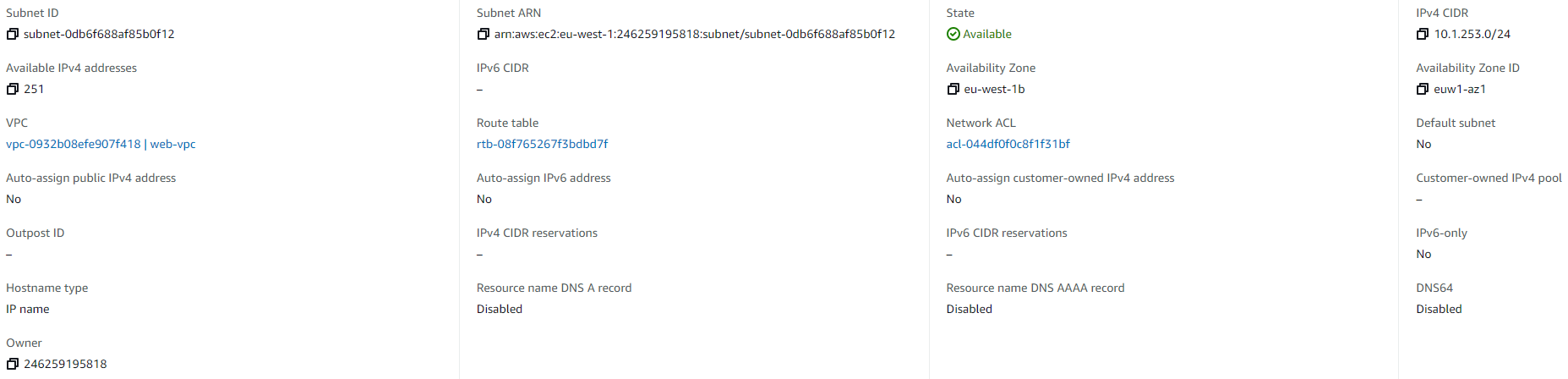
Web-pub3 is a public subnet which we used for testing our environment. The subnet is currently empty and could be removed in the future.

**Database**



This subnet is a private subnet in which contains our database. The subnet is set to private for security purposes. When the testing sprint is finished, this database subnet could be removed if we have no further need of RDS.

**web-pub2**



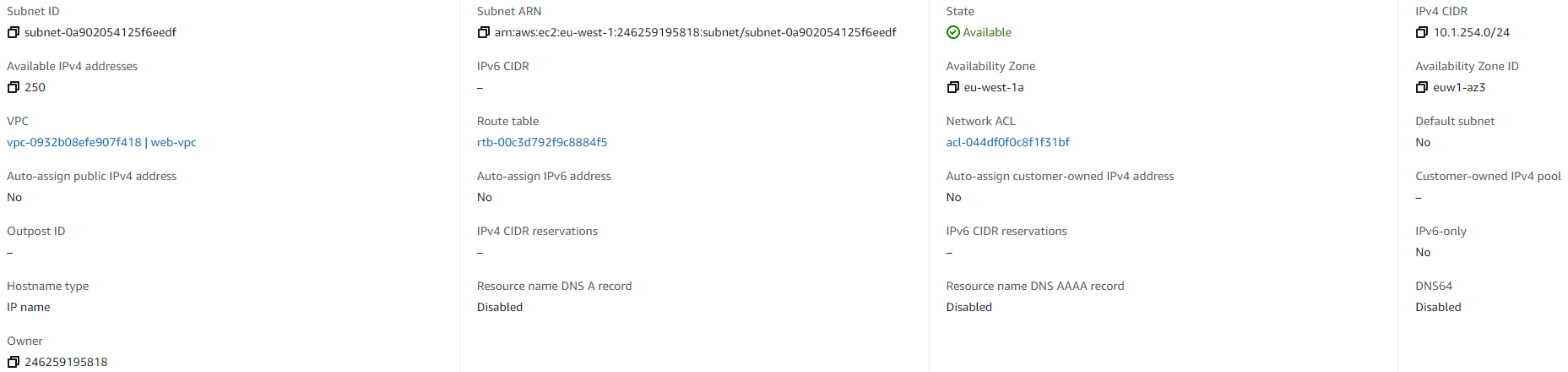
Web-pub2 is a public subnet which we used for testing our environment. The subnet is currently empty and could be removed in the future.

**nat-pub**

Graphical user interface

Description automatically generated

**web-pub**

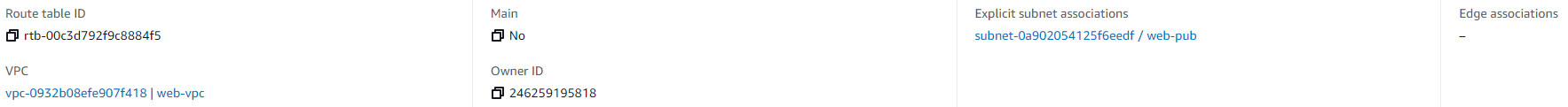


### 4.2.4 Route tables

There are four route tables created for the public and private subnets. The tables specify how packets are forwarded between the subnets within the VPC.





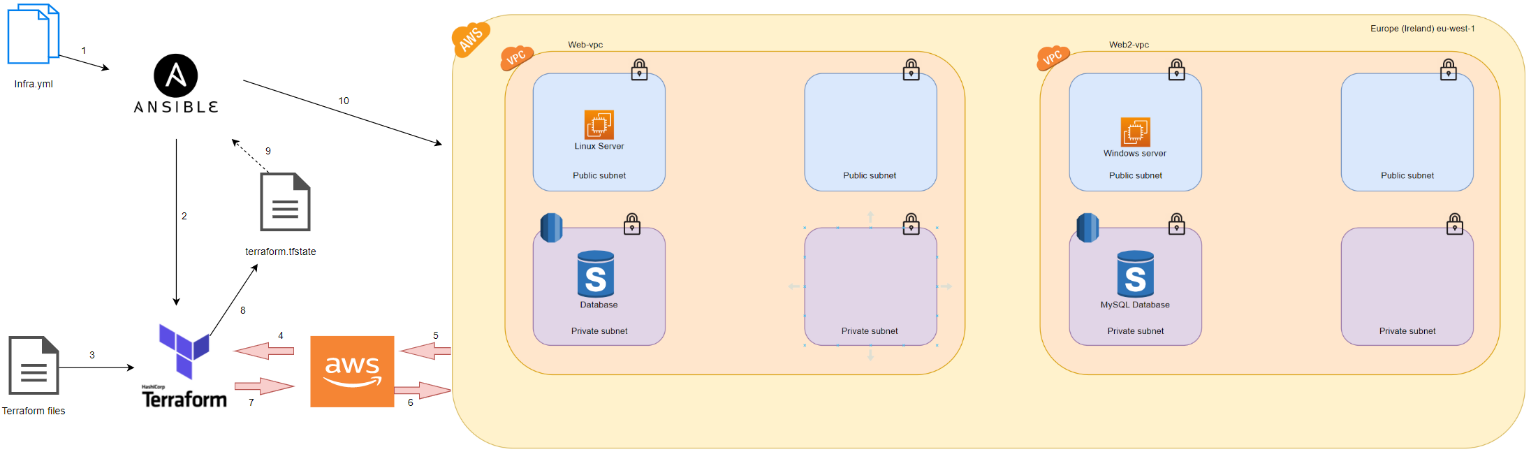




### 4.2.5 Internet gateways

A Internet Gateway is created for the public and private VPC’s because these VPC’s need to have internet access.5. Process diagram

## 5.1 Part 1



This process diagram includes every single step that is needed to create the whole infrastructure and the configuration. The first step is to have an Ansible file, which includes the beginning of everything. The Ansible file includes executing the Terraform files and checking the servers to see if they are running. Moreover, it also includes installing Apache for the webserver with all the needed configuration to have a website up and running. Talking about the Terraform files, which are six in total, and every single file is connected to each other. More specifically, We have files for the VPCs, EC2 instances, NAT instance, and network. In this way, the whole infrastructure is created. In that way, I could save a lot of time with this automated creation of the infrastructure and skip the manual configuration in this phase. The benefit of Terraform is that we tell the software what we want, instead of writing all the steps in a file.