Anders Rindal Loeng
Morten Rognhaug Bratberg
Eirik Granvin Bakke

Secure and Compliant Infrastructure as Code

Bachelor's thesis in Digital Infrastructure and Cyber Security
Supervisor: Eigil Obrestad
May 2022

Bachelor's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU
Norwegian University of
Science and Technology

Anders Rindal Loeng
Morten Rognhaug Bratberg
Eirik Granvin Bakke

Secure and Compliant Infrastructure as Code

Bachelor’s thesis in Digital Infrastructure and Cyber Security
Supervisor: Eigil Obrestad
May 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU
Kunnskap for en bedre verden

Abstract

Title	Secure and compliant Infrastructure as Code
Date	20.05.2022
Authors	Anders Rindal Loeng Morten Rognhaug Bratberg Eirik Granvin Bakke
Supervisor	Eigil Obrestad
Employer	Bouvet ASA
Keywords	Infrastructure as Code (IaC), Azure, Terraform, Cloud computing, Landing zone
Number of pages	88
Number of Appendix	7
Availability	Open
Abstract	A growing number of businesses wants to take advantage of the cloud to achieve their goals. Bouvet have a goal to help Small and Medium-sized Businesses achieve this. Before businesses can utilize the cloud, the necessary resources to support their needs must be created. Infrastructure as Code makes it possible to build cloud environments efficiently, make changes and keep track of changes. This project creates three templates that will automate the process of provisioning an infrastructure in Azure. The security and compliance of the templates are crucial factors for the project. One of the templates is a library of code blocks, where developers can select and combine parts to create a desired cloud infrastructure. The two other templates are specific cases which is built with resources from the base template. The two specific cases are for Internet of Things (IoT) integration and web page hosting in the cloud. The project includes configuration and use of CI/CD.

Sammendrag

Tittel	Secure and compliant Infrastructure as Code
Dato	20.05.2022
Forfatter	Anders Rindal Loeng Morten Rognhaug Bratberg Eirik Granvin Bakke
Veileder	Eigil Obrestad
Oppdragsgiver	Bouvet ASA
Nøkkelord	Infrastructure as Code (IaC), Azure, Terraform, Sky, Landingssone
Antall sider	88
Antall vedlegg	7
Tilgjengelighet	Åpen
Sammendrag	Et økende antall bedrifter ønsker å oppnå målene sine ved å utnytte seg av fordelene ved bruk av sky. Bouvet har et ønske om å hjelpe små og mellomstore bedrifter (SMB) oppnå sine mål. Før en bedrift kan utnytte seg av skytjenester, må de riktige resursene opprettes for deres behov. Infrastructure as Code gjør det mulig å bygge skymiljøer effektivt, gjøre endringer og holde kontroll på endringer. Dette prosjektet har tre templates laget for å automatisere prosessen ved å bygge en infrastruktur i Azure. Sikkerhet og compliance i templatene er avgjørende faktorer for prosjektet. Et av templatene fungerer som et bibliotek av ressurser, som utviklere kan ta kode ut fra, og sette sammen for å lage en infrastruktur i skyen. De to andre templatene er spesifikke scenario som er lagd basert på koden i biblioteket. De to scenarioene er hosting av nettside og integrasjon av IoT enheter med skyen. Prosjektet inneholder også konfigurasjon og bruk av CI/CD.

Contents

Figures

Tables

Code Listings

Acronyms

API Application programming interface. 63

ARM Azure Resource Manager. 27, 51

AWS Amazon Web Services. 22

CD Continuous Delivery. iii, iv, 3, 7, 24, 25, 70, 74

CDN Content Delivery Network. 54

CI Continuous Integration. iii, iv, 3, 7, 24, 25, 70, 74

CLI Command-line Interface. 10, 18, 22, 43, 51

DDoS Distributed Denial-of-Service. 66

GUI Graphical User Interface. 10, 27, 40, 42, 63, 88

HCL HashiCorp Configuration Language. 16

IaaS Infrastructure as a Service. 11

IaC Infrastructure as Code. iii, iv, 1–4, 7, 10, 16, 24, 26, 27, 32, 56, 59, 60, 80, 84

IoT Internet of Things. iii, iv, vii, xi, xii, 2, 3, 15, 16, 25, 30, 34, 40, 41, 44, 45, 47, 50–52, 57, 60, 62–64, 71, 87, 113, 117

ISO International Organization for Standardization. 9, 35, 87

ML Machine Learning. 41

NSG Network Security Group. 66

NSM Norwegian National Security Authority (Nasjonal sikkerhetsmyndighet). 35

NTNU Norwegian University of Science and Technology (Norges teknisk-naturvitenskapelige universitet). 4, 5

PaaS Platform as a Service. 11, 42

SaaS Software as a Service. 11

SIEM Security Information and Event Management. 43

SMB Small and Medium-sized Businesses. iii, iv, 1–3, 34, 35, 44–47, 60–62, 64, 79–81, 84, 85, 87

SME Small and Medium-sized Enterprises. 2

TDD Test Driven Development. 7

TLS Transport Layer Security. 14, 75

WAF Web Application Firewall. 54

Glossary

ACL is rules that is used to deny or allow traffic based on IP addresses or CIDR blocks, or access control to files, objects and systems. 36

Cache is a component that increases the speed of fetching data by storing a limited amount of data at a temporary location. Cache is information stored based on an earlier action, either a copy of data stored elsewhere or from a earlier compute. 38, 54, 65, 66

CIS Controls V8 stands for "Center for Internet Security", and is a standard for security controls to ensure best practice for data security.. 9

DIPA stands for "Data Protection Impact Assessment", this is an assessment of privacy consequences, to ensure that the privacy of those involved is protected.. 62

GDPR stands for General Data Protection Regulation, and is a law in the EU and EEA with the purpose of protecting data and privacy.. 8, 45, 46, 52, 56, 58, 64

ISO 27000 family is a series of international standard for information security management [1]. 35

mean time to recover(MTTR) is the average amount of time it takes for the system or resource to becomes fully functional after a failure.. 63

NIST SP800-53 stands for "National Institute of Standards and Technology", the Special Publication 800-53 is a standard for security and privacy controls.. 9

NoSQL is a non-relational database. That means the database is not stored with the linked together relations, that we find in SQL databases . 41, 54

PCI-DSS stands for "Payment Card Industry Data Security Standard", and is standard for security when payments is involved.. 9

SQL is a relational database, stored in rows and tables and is linked together. 41, 54

Chapter 1

Introduction

A growing number of businesses wants to take advantage of cloud computing to achieve their goals. The knowledge needed to make this happen is something not all Small and Medium-sized Businesses (SMB) have internally, and therefore use a service provider. For a service provider to be able to efficiently provide cloud infrastructure to customers, Infrastructure as Code (IaC) is an important tool. Having code for provisioning the infrastructure enables the provider to offer predictable timeframes and price for their customers. This project will help Bouvet reach their goals by developing Infrastructure as Code with security and compliance as the focal point.

1.1 Employer

Bouvet is a Norwegian IT consultancy, with offices in Norway and Sweden, which delivers services to both private and public sector. Their services ranges from designing, developing and advising on IT solutions and digital communication. Bouvet's ambition is to "be the most credible consultancy with the most satisfied employees and clients" [2]. Building secure and compliant infrastructures in the cloud is a crucial ability for Bouvet to reach their goals.

1.2 Field of Study

This section will give a basic introduction to the most central subjects for this thesis. The most central subjects are Infrastructure as Code (IaC), security and compliance. IaC is used to create the cloud infrastructure. Security and compliance are important factors for both the code and the infrastructure created.

1.2.1 Infrastructure as Code

Many Small and Medium-sized Businesses (SMB)[1] today wants to take advantage of the functionalities in the cloud. A cloud environment must in some way be managed and provisioned, this could either be manually done or automated through code. Infrastructure as Code (IaC) enables providers to build new environments more efficient and consistent by automation of these processes. IaC also makes it possible to track changes, roll back and recover the infrastructure. In this project IaC is used to create a library of code that Bouvet can make use of when creating new cloud environments, and two cases that provisions infrastructure for Internet of Things (IoT) and web hosting.

1.2.2 Compliance

The project will have focus on meeting general compliance for the use that the templates are intended for. This could either be meeting a compliance demand for a specific sector, or to comply with regulations from governing bodies relevant for the users. Both the IoT and web page cases can fall under compliance regulations depending on what data is collected.

1.2.3 Security

The project will have a focus on security for the infrastructure created. Security will be an important aspect for the infrastructure since it will potentially handle sensitive information. Through a secure infrastructure configuration, the service running in the cloud will only be available to users with the correct access rights. Security in the code is important in addition to the cloud infrastructure. The project will include security mechanism for code testing and safe usage of code.

1.3 Project Description

The purpose of this project is to develop templates for efficient provisioning of infrastructures in Azure. This will help Bouvet to explore the possible alternatives in the work of cloud migration, and help them improve in the work of consulting SMB with their cloud migration. This includes creating a standardized processes

[1]Small and Medium-sized Businesses (SMB) and Small and Medium-sized Enterprises (SME) is defined as the same and can be used interchangeably. This project will used Small and Medium-sized Businesses (SMB) for describing small and medium-sized business, enterprises and organizations.

and making the migration more adaptable and easy. This project will help the employer achieve this by developing IaC templates.

The project consists of 3 main components, the base template and 2 case templates. The base template is a generalized library with IaC for provisioning the most relevant resources in Azure for a SMB. From this library, Bouvet can make use of the code to create new IaC cloud environments. The base template is not meant to be provisioned as it is, but to be used as a library.

In addition to the base template, the project consists of IaC for two cloud environments that are ready to be provisioned and used. The two environments are IoT monitoring and web page hosting. For these 3 components the compliance and security aspects, related to their fields, will also be taken into consideration.

The project also includes IaC for setting up the prerequisites for provisioning. This consists of shared storage for configuration files and the necessary resources for Continuous Integration (CI)/Continuous Delivery (CD) through pipelines.

1.4 Delimitations

Decided limitation and framework for the project is the following;

- The resulting infrastructure code will be templates, not a “ready-to-go" infrastructure stack. The code must be modified based on the needs of the customer or project.
- Security and compliance will be prioritize over comprehensive scope. Not all possible needs and variants of a cloud environment will be covered.

The purpose of the limitations is to improve the final product, by giving the group more time to focus on the most important aspects of the task at hand.

1.5 Target Audience

The main target audience for this project will be the employer Bouvet and the examiners. For Bouvet, the final product is the most important aspect of this project. The thesis is also relevant because it provides the employer with information about the templates created.

Fellow students and people interested in the field of Infrastructure as Code and cloud computing are also a significant audience for this project. Fellow students may be interested in both the report and the product. The report can be used to learn about the subject or as an example of how a report is structured. The product can also provide the students with an example of how IaC templates are structured, as well as providing the possibility to test and see how they work. Since the repositories will be publicly available, people interested in the field might use the code to learn more about cloud infrastructure provisioning. People or organizations starting up with using provisioning tools for Azure, might use the code as inspiration for their own code.

Based on the target audience, it is expected that the reader of this report has some basic knowledge on the topic of cloud computing, Infrastructure as Code and cyber security.

1.6 Academic Background

The group consists of 3 students studying Digital Infrastructure and Cyber Security at NTNU in Gjøvik. The members have acquired relevant knowledge for this project through the Bachelor program. All members have obtained knowledge on the subject of digital infrastructure and cloud technologies through the courses "DCSG1001 - Infrastructure: basic skills", "DCSG1005 - Infrastructure: secure core services" and "DCSG2003 - Robust and scalable services". The course "IIKG3005 - Infrastructure as Code" was a optional course that all the members took, which was very relevant to the project itself since it gave an introduction to the field of Infrastructure as Code. Multiple courses from the Bachelor program gave the members knowledge on cyber security, but "DCSG2005 - Risk Management" was special relevant because it gave an introduction to the ISO standards used in this project.

1.7 Motivation

The group reasoning for choosing this project was all members have a interest in the field of Infrastructure as Code and automation. All members have experience with cloud administration and operation from courses at NTNU in the field of digital and cloud infrastructure. We wanted to increase or knowledge and experience on these topics.

1.8 Roles

The employer for the project is Bouvet ASA, represented by Lauritz Prag Sømme, Sebastian Slettebakken, Magnus Korneliussen and Herman Hallerud Wikstrøm.

Eigil Obrestad is the supervisor for the thesis. He works partly as an Assistant Professor at The Department of Information Security and Communication Technology, in addition to being a senior engineer at NTNUs IT department with responsibility for NTNUs private clouds.

1.9 Report Structure

Chapter 1 Introduction
A short introduction with basic information about the thesis and the project.

Chapter 2 Background
Introduces and explains the theory that is necessary for the reader to know about, to be able to follow the rest of the thesis.

Chapter 3 Development Process
Explains how the project work process was organized.

Chapter 4 Requirements for Base Template
Explains the expected requirements the finished base template should meet.

Chapter 5 Implementation Base Template
Description of the finished result of the base template in addition to how the template was created.

Chapter 6 Requirements for Case Templates
Explains the expected requirements the finished case templates should meet.

Chapter 7 Implementation of case templates
Description of the finished result of the case templates in addition to how the templates were created.

Chapter 8 Evaluation
An evaluation of the requirements and implementations for the base and case templates.

Chapter 9 Provisioning
Explains the setup and usage of the provisioning process used in the project.

Chapter 10 Assessments
An assessment of the work process and the project as a whole, including a potential improvements and future work.

Chapter 11 Conclusion
A conclusion for the thesis.

Chapter 2

Background

The purpose of the Background chapter is to give the reader a deeper introduction to the project and give theoretical information of existing solutions, tools and methods used in the project. This will improve the readers understanding of how and why the project was solved this way. Rules, standards and regulations used in this thesis, will also get a brief description with focus on their jurisdiction, applicability and purpose.

2.1 Infrastructure as Code

Infrastructure as Code (IaC) has become a stable practice in many of the biggest companies in the world. The point of IaC is to automate all the processes relating to creating and running the infrastructure. The idea of IaC stems from Software development, by incorporating the agile practices of Test Driven Development (TDD), Continuous Integration (CI) and Continuous Delivery (CD) to make changes to the infrastructure fast and reliable [3]. Increased use of cloud computing, has lead to automation of infrastructure being easier. Automation is by many considered as best practice in cloud infrastructure. With the use of agile practices, it often leads to increased quality of the end results, since making improvements is easier and encouraged. [3]

2.1.1 Why Automate?

When the system gets bigger, manually setting up the infrastructure will quickly become unmanageable and complex. For the IT staff to manually maintain an infrastructure like this, it will be very error prone, hard to recreate, lead to high downtime, making changes becomes hard and lead to an increase in time consumption. Most of these problems can be avoided by automating the processes related to the infrastructure. When automated, the infrastructure becomes more manageable, more reliable, easier to understand, lower downtime, and making changes becomes an easy task.

2.2 Compliance

Compliance is "the act of obeying an order, rule or request" [1]. For this project the main focus is compliance of IT infrastructure, and compliance in a cloud environment. Most IT related systems have a strict set of rules to follow, depending on the country, what kind of information is processed, industry requirements and more. An example of this is GDPR. When dealing with privacy information from users in the EU, there are strict rules on how you should store the data, for how long you can keep the data, security requirements for the systems, and more to avoid being fined. [4] The main goal of GDPR is to ensure that users privacy data do not get misused, and that it is stored properly to lower the risk of it getting lost in case of a data breach.

GDPR is an example of required compliance on a international level, when a set of criteria is met, and it will be enforced by a governing body. Not all compliance is required, there are forms of compliance that serves the purpose of guidance, and to achieve best practice. The ISO standards are an example of this. Most ISO standards for cloud and IT systems, are not required to be in compliance with by law. But being in compliance with them, is by many considered as best practice, and will help the companies credibility. Standards like ISO, is not something that is often required to follow at a international or national level, but can often be required for a company in different types of contracts, or be self enforced to follow for the employees in the company.

2.2.1 GDPR

As of 2018 [5] GDPR was implemented and made mandatory to follow for all companies operating in the EU, when handling personal data. The implementation of GDPR had a goal of giving the users more control of their personal data, and protect the users data from misuse. Compliance with GDPR is governed by the EU, and if companies are working with EU customers data, they are required to comply with GDPR regulation. When a company is found to not be following GDPR, sanctions might be imposed.

[1]The definition on "compliance" is retrieved from the Cambridge dictionary, at https://dictionary.cambridge.org/dictionary/english/compliance, accessed: 07.03.2022

2.2.2 ISO

ISO is a international organization, that was founded in 1947, with a goal of standardizing management, products, processes and more[6]. Depending on the regulatory processes in a sector, the ISO standards may be considered obligatory or just as guidance. For this project the most relevant ISO standards are from the ISO27000 series, which focuses on the information security aspects on a technical and organizational management level[7].

2.2.3 Compliance in Azure

Following compliance can in many cases be quite time-consuming[8], but in Microsoft's cloud provider solution, Azure, there are some tools that are available to make compliance easier for users to implement and follow. With Microsoft Defender for Cloud, there is a feature quite similar to Compliance Manager called "Regulatory Compliance", that helps customers manage their compliance requirements[9]. This feature shows users which resources in the environment that are not compliant with the Azure Security Benchmark V3, and other eventual standards that have been added. The Azure Security Benchmark V3 is a collection of best practices from the PCI-DSS, CIS Controls V8 and NIST SP800-53 standards, made by Microsoft[10]. This feature paired with Azures extensive resources on compliance documentations, makes following compliance a little simpler[2].

2.3 Security

Cybersecurity is the implemented protection against potential harm from a threat actor. The CIA triad is one of the most used models in the area of cybersecurity. CIA stands for Confidentiality, Integrity and Availability. Availability means that the components are up and running and are accessible for authorized user when needed. Integrity intend that data or resources have not been tampered with by an unauthorized user. Confidentiality is the control of access to data and resources for only the correct personnel and machines.

Cloud Security is functionality that exists to protect the resources in a cloud infrastructure. Responsibility for security in the cloud is divided between the cloud provider and the cloud user. The cloud provider has generally a responsibility for the security of physical resources, and in some cases the operating system and software they provide. The cloud users are generally responsible for what they put in the cloud, like data or running a SQL database.

[2]Azure documentation on compliance https://docs.microsoft.com/en-us/azure/compliance/

Testing the infrastructure code is an important part of keeping the infrastructure secure. The infrastructure can be tested on various levels. Testing can be done before the infrastructure is provisioned, by running security tests on the code. These tests can be smaller tests, like unit tests, that checks only small parts of the code. When the infrastructure is running, testing can be done on the whole environment through logging and monitoring. Monitoring of the infrastructure can show inconsistencies from normal behavior and use. Irregular behavior and use can either originate from fault in the resources itself or a malicious user. Test can also check for known security vulnerabilities and weaknesses [8].

2.4 Microsoft Azure

Azure was released in 2010, under the name Windows Azure, and offered web roles and SQL Databases. It was not before 2014 they adopted the name Microsoft Azure and expanded their services to offer networking, computing, storage and more [11]. Azure is a pay as you go service, where you only pay for what you use. This allows the service to be affordable for most companies regardless of their size.

2.4.1 What is Azure

Azure is at the core a public cloud, where Microsoft handles the underlying infrastructure for the environment. Azure allows users to build their own infrastructure, with Microsoft's resources. With Azure the users also have the opportunity to integrate their own local infrastructure, or a private cloud with their environment, and end up with a hybrid cloud solution [12].

2.4.2 How Azure works

To use Azure the users first needs an account, with a subscription for their environment. With the subscription the users get access to the different resources. Most features in Azure needs a Resource Group as the base. When the Resource Group is configured, it is then possible to create resources like VM's, networks, storage, etc. Azure is made to be beginner friendly, with an easy to use GUI, that doesn't require much prior knowledge. For users that want more IaC oriented solutions, they can use Azure's own CLI language "Azure CLI", their provisioning language "ARM" or use third party provisioning languages like Terraform.

2.4.3 Azure Resources

Azure has a large amount of resources to solve many different problems. The resources are a combination of PaaS, IaaS and SaaS solutions. This section will cover the resources used in this project, with a brief description of the functionality of each resource. The resources are divided in to tables, based on their usage and functionality. Resources availability are different, when it comes to where each resources is hosted. Most resources provided by Microsoft Azure have the possibility to be hosted in Norway, but there are instances where a resource must be hosted in another region.

Table 2.1: Azure Resources General [13–22]

Resource	**Description**
Resource Group	A container that groups resources together in Azure.
Subscription	A solution to organize resource groups and control access to the environment, and centralize monitoring and billing information.
Azure Active Directory	Active Directory hosted in Azure, for identity management.
Key vault	A service with the purpose of securely storing secrets, like password, certificates, ssh keys and other information that should be secret.
Dashboard	Creates organized views of resources in the subscription, with possibility to display monitoring, users, tasks and more.
Monitor	A solution in Azure to gain visibility, investigate, view, notify and analyze resources status, usage, health and performances.
Cost Management	Consist of budget and cost, and is used to monitor and analyse spendings. It can also notify user when they surpass a given limit, or when it is forecasted that the spendings will surpass this limit.
Log Analytics	A tool to store and query metric data collected from resources.
Microsoft Defender for Cloud	A tool for managing security posture and for threat protection.
Sentinel	Azure cloud solution for security information and event management (SIEM) and security orchestration, automation, and response (SOAR).
Application Insight	Gives detailed monitoring of web apps, like anomalies in performance and what users do with the app.

Table 2.1 Azure Resources General [13–22] gives a an explanation of the most common resources in Azure, that is utilized in this project. Both resource groups and subscriptions are used to group objects together. With the exception of Azure Active Directory and Key Vault, the other resources are used mainly for monitoring and logging, with some having a reactive task based on the data.

Table 2.2: Azure Resources storage [23–26]

Resource	**Description**
Storage Account	An administrative resource for storing the different types of data objects in an Azure environment.
Cosmosdb	A database service that can handle different types of data. It can handle both SQL and NoSQL data.
Azure SQL Server and Database	Fully managed PaaS. One server can contain multiple databases. Databases can be deployed as a single databases or as a part of an elastic pool, with multiple databases that shares resources such as CPU and memory.
Azure Cache for Redis	Cache for applications. For example a Web App can use Redis to cache objects from a database.

Tabel 2.2 gives an introduction to the resources in Azure, used to store data and information. This table shows the resources for storage, that is used in this project. The different resources provides the possibility to store different types of data.

Table 2.3: Azure networking resources [27–30]

Resource	**Description**
Content delivery network (CDN)	Point-of-presence servers that caches web content close to the users.
Application gateway	An application layer (OSI layer 7) load balancer in Azure for web traffic.
Web Application Firewall (WAF)	Configuration of firewall policies which can be applied to an Application gateway, CDN or Front Door in Azure, to defend against common exploits used in attacks on web services.
Network Security Group (NSG)	Filtering of network traffic based on source IP and port, destination IP and port and protocol.

Table 2.3 Azure networking resources [27–30] gives an intro to networking resources used in this project.

Table 2.4: Azure Resources computing [31–35]

Resource	Description
App service	A PaaS solution used to easily create and host web applications, mobile back-ends and APIs.
Function App	Serverless solution to group functions together, used for scheduling tasks and lightweight Web API.
Logic App	Platform for creating and running automated workflows that integrates apps, data, services, and systems.
Virtual Machines	VMs in Azure is a compute resource which is on-demand and scalable. Can be connected to via RD-P/SSH.
Virtual desktop	Service that provides virtual desktops and apps. Desktops can be connected to over TLS from a Remote Desktop client or in a browser.

The resources in table 2.4 Azure Resources computing [31–35] are resources that are a virtual representation of a computer or a server. App service, function app, and logic app have built in functionality to decrease the difficulty for the user.

Table 2.5: Azure IoT Resources [36–45]

Resource	Description
IoTHub	A central message hub in azure that is used for connecting and controlling IoT devices. Gives an overall view of the connected devices.
IoT Central	A Platform as a Service solution to handle and control IoT devices.
Eventhub	Big data streaming platform, used for processing events and view real time analytics from connected resoruces.
Stream Analytics job	Real-time analytics, for processing high volume of data from connected sources. Used for triggering different workflows, that can either alert, send data or store data for later use.
Digital Twins	A service for creating visual presentation of environments and large objects, to increase understanding of the object and connection of the IoT devices.
HDInsight	Data analytics service using open source frameworks like Hadoop, Apache Spark, Apache Kafka, R and more.
Data Explorer	Data analytics service that can analyze large amounts of data in near real time.
Data Lake	Storage service for big data analytics.
Machine Learning (ML)	Dedicated service for Machine Learning tasks.
Databricks	Data analytic platform, that has three different versions, Databricks SQL, Databricks Data Science and Engineering and Databricks Machine Learning.

Resources in tabel 2.5 Azure IoT Resources [36–45] are used alongside IoT, or to utilize information form IoT devices. The functionality of the resources in this table is to collect data from the devices, or to use and analyze this data. Resources in table 2.5 will be found in the case template for IoT vehicles, and in the IoT module in the Base template.

2.5 Terraform

Terraform is an Infrastructure as Code tool created by HasiCorp. Terraform is open-source, which means that everyone can use, configure and distribute the software and source code. Terraform is generally written in the HashiCorp Configuration Language (HCL) syntax, but it can also we written in JSON syntax. HCL syntax is usually preferred because it is easier for humans to write compared to JSON syntax. The code is written in text files with the .tf extension or .tf.json if using the JSON version of Terraform.

Terraform is a declarative language for provisioning infrastructure. A declarative language is a programing language that only declares the desired state, without specifying how to get to the desired state. Terraform is a provisioning tool, used for creating and maintaining infrastructures, in contrast to configuration tools, which is for is for configuring and maintaining compute resources. Idempotency is another benefit of Terraform. Idempotent code is code that will not change the result, when applied multiple times, given that the code is unchanged. In practice this means that the infrastructure will be unchanged, given the code is consistent, when running the code repetitively.

Terraform works with over 1700 different providers, including some of the most popular cloud service providers. The providers are categorized by HasiCorp in "Official", "Verified" and "Community". The different categories is defined in the following manner, "Official providers are owned and maintained by HashiCorp", and Verfied providers are third party providers that own and maintain them [46]. Community are published providers that are created by the individuals or groups in the Terraform community that is not verified [47]. The Azure provider is classified as "Official" by Terraform.

2.5.1 Modules

Modules are containers for multiple resources and code blocks that are connected. The modules have separate directories with Terraform files, and in some cases sub modules. The reusability of code is a key benefit of using modules. The root module is the module that is defined in the root folder of the working directory. Module blocks can be used from a module to refer to a child module [48]. In addition to modules in the local repository, Terraform provides the option to use published modules from sources like Terraform Registry, GitHub and more [49].

2.5.2 Terraform Blocks

This section will explain the different code block types in Terraform utilized in this project.

Module Block

Module blocks are code blocks used to call child modules. The module blocks are required to specify the source of the child module, like line 2 in Code listing 2.1, this can both be local and published modules. Input variables can be sent from the parent module to the child module by specifying a name and a value for the variable.

```
1 module "vnet" {
2   source                = "./modules/1-vnet"
3   location              = var.location
4   environment           = var.environment
5 }
```

Code listing 2.1: Module block

Resource Block

The resource block describes the infrastructure components being created, for an example see Code listing 2.2. Resource blocks are specific to the provider for the Terraform project, like Azure or AWS. The resource block creating a infrastructure resource for one provider is different to the resource block from another provider.

```
resource "azurerm_virtual_network" "vnet" {
    resource_group_name = azurerm_resource_group.main.name
    location            = var.location
    name                = "${var.environment}-${var.vnet_name}"
    address_space       = var.address_space
}
```

Code listing 2.2: Resource block

Output Block

Output blocks in a child module are used to send information to the parent module. In the root module the outputs from the child module can be used as inputs to other modules, for example a subnet id from network module can be used in virtual machine module. The outputs can also be printed to the CLI when running the code. For an example of a output block, see Code listing 2.3.

```
output "vnet_id" {
    value = azurerm_virtual_network.vnet.id
}
```

Code listing 2.3: Output block

Variable Block

Variables are declared and defined in variable blocks and then used in resource blocks. For how a variable block can look like, see Code listing 2.4

```
variable "location" {
  type          = string
  description   = "Location of the resources"
  default       = "norwayeast"
}
```

Code listing 2.4: Variables block

2.5.3 Declaring, Defining and Referencing Variables

There are multiple ways to make use of variables when using Terraform. This section explains the approach which is used in the project. This approach consists of four different ways of using variables.

Variables in Child Module

One way is to declare and define variables in a module's varables.tf file, and reference them in the main.tf file. This approach is used for variables that are only relevant for the module in question. Variables in a child module is shown in code listing 2.5 and 2.6.

```
resource "azurerm_resource_group" "main" {
  name                = var.resource_group_name
  location            = var.location
}
```

Code listing 2.5: main.tf

```
variable "resource_group_name" {
  type        = string
  default     = "main"
}

variable "location" {
  type          = string
  default       = "norwayeast"
}
```

Code listing 2.6: variables.tf

Variables in Root Module

Another way is to declare and define global variables in the root variables.tf file (Code listing 2.7). The variables are passed to the module (Code listing 2.8), then declared in the child modules variables.tf file (Code listing 2.9) and referenced in the child modules main.tf (Code listing 2.10). This allows the variables to be passed into multiple modules. In the code listings the location is defined in the root variables file, while the resource group name is defined in the child modules variables.tf. Both variables are then references in resource block.

```
variable "location" {
  type          = string
  description   = "Location of the resources"
  default       = "norwayeast"
}
```

Code listing 2.7: Root variables.tf

```
module "vnet" {
  source                = "./modules/1-vnet"
  location              = var.location
}
```

Code listing 2.8: Root main.tf

```
variable "resource_group_name" {
  type        = string
  default     = "main"
}

variable "location" {
  type          = string
  description   = "Location of the resources. inherited from root-variables"
  default       = ""
}
```

Code listing 2.9: Child module variables.tf

```
resource "azurerm_resource_group" "main" {
  name                = var.resource_group_name
  location            = var.location
}
```

Code listing 2.10: main.tf

Passing Variables Between Modules

When using modules to separate the Terraform code, some values from a resource in a module, can be necessary to access in another module. For example the ID of a subnet created in the network module, might be needed in the virtual machine module for the virtual machine resource.

In Terraform this is done by defining the value as an output in the module (Code listing 2.3), from which the value comes from, and passing it to the other module as shown in Code listing 2.11.

```
module "virtual-machine" {
    source              = "./modules/virtual-machine"
    subnet_id           = module.vnet.subnet_id
}
```

Code listing 2.11: Module block with variable from another module

Variables in .tfvars Files

The fourth way is using .tfvars files. These files are for different configuration between multiple environments by having one .tfvars file for each environment. The .tfvars file associated to the cloud environment is used when provisioning. The variables in .tfvars is passed through to the child modules main.tf file as shown with the environment variable in listing 2.12 to listing 2.16.

```
environment = "prod"
```

Code listing 2.12: production.tfvars

```
variable "environment" {
  type          = string
  description   = "(test/stage/prod) inherited from .tfvars files"
  default       = ""
}

variable "location" {
  type          = string
  description   = "Location of the resources"
  default       = "norwayeast"
}
```

Code listing 2.13: Root variables.tf

```
module "vnet" {
  source                = "./modules/1-vnet"
  location              = var.location
  environment           = var.environment
}
```

Code listing 2.14: Root main.tf

```
variable "environment" {
  type          = string
  description   = "(test/stage/prod) inherited from .tfvars files"
  default       = ""
}

variable "resource_group_name" {
  type        = string
  default     = "main"
}

variable "location" {
  type          = string
  default       = "norwayeast"
}
```

Code listing 2.15: Child module variables.tf

```
resource "azurerm_resource_group" "main" {
  name                = "${var.environment}-${var.resource_group_name}"
  location            = var.location
}
```

Code listing 2.16: Child module main.tf

The same codebase provisions the environments, and the .tfvars files creates differences between them. In this example, the .tfvars files are used to set the name for resource group. Other examples of differences between the environments, are for instance size of resources where testing environment require less than production, and thresholds for budget consumption alerts.

2.5.4 Terraform State

Terraform keeps track of the infrastructure that is provisioned with Terraform in a state file called terraform.tfstate. When running Terraform, it compares the infrastructure code to the actual resources in Azure. [50]

For projects with multiple team members, the state file should be accessible for everyone in the team. The state file can be kept in Azure and referenced in the Terraform code. This is a good practice, since team members and provisioning pipelines can reference the same state file. The state file will be in a locked state when a team member or a pipeline runs Terraform apply, stopping others from doing changes to the infrastructure at the same time. Since the state file can contain secrets, keeping it in Azure storage will reduce the risk of loosing secrets. This approach also ensures that the latest version of the state file is used, while also keeping older versions for the ability to roll back.

Local State

Local state is the default setting for Terraform, and the state file will be stored on the developers local computer. The state file will be named "terraform.tfstate" as default, unless another name is specified. The name can either be specified in the backend file or in the CLI when running the code.

Remote State

The Terraform state file can be stored in a different location than the developers local computer. Terraform support different backends like azurerm, s3 (AWS), Kubernetes, http and consul. For this project, Azure storage is used as backend for Terraform. In Azure the state will be stored in a blob container on a Storage Account. For a description of Storage account see Table 2.2. The Terraform backend requires the resources in Azure to be up and running before being used. The backend.tf file specifies what backend is being used. [51]

Advantages of Remote State

Version Control
Storing and referencing state file in Azure storage ensures that the correct version is used, compared to other approaches such as pushing and pulling it to a version control system.

State Locking
For teams with multiple members it is important to prevent simultaneous provisioning operations, since it can cause problems. Azure storage blobs supports locking. This means when someone or a pipeline runs terraform plan or apply, the state file is set to a locked state, and others have to wait.

Security
Azure storage automatically encrypts the data which is stored [52], and users must authenticate to get access to the state file.

2.5.5 Terraform Workflow

When provisioning infrastructure with Terraform, the procedure starts with the command `terraform init`, which does the initialization steps in the working directory [53]. An optional next step is to run `terraform plan` to output the changes that will be made to the infrastructure [54]. The way Terraform accomplishes this, is by updating the state file to match the actual resources in the cloud, every time plan or apply is ran. Then the state file is compared to the code and the differences are shown.

The `terraform apply` command first outputs the changes that will be made, then prompts for confirmation before it executes. The confirmation prompt can be skipped when provisioning with a pipeline. When provisioning different configurations to different environments using .tfvars files, the relevant file is stated in the commands: `terraform plan --var-file="prod/prod.tfvars"` and `terraform apply --var-file="prod/prod.tfvars"`.

To remove infrastructure components,`terraform destroy` is used. The command first outputs what will happen, and prompts for confirmation the same way as `terraform apply` does, before executing.

2.6 Scripting

When using IaC to build cloud environments there are some tasks where scripting is the desired method. This can be because of technical limitations of a IaC tool, or a choice made by the developers. Azure has its own command-line tool called Azure CLI, which can be used to provision and manage resources in Azure. The tool is cross-platform and the commands can be incorporated into both PowerShell and Linux shell scripts.

In the project, Azure CLI and bash scripting are used in addition to Terraform for managing resources in the cloud environment. Azure CLI is used for managing permissions in Azure, and bash scripting is used for initial configuration of a virtual machine used for running the pipelines.

2.7 Git

Git is a free open source version control system, that is meant to make file management easier for developers [55] [56]. Git makes file exchanging, changing and version controlling easy. When working together in teams, and making changes to the code, things can suddenly stop working. With Git there is a way to see the changes that where made, and by who, which could lead to the possible problem being found much quicker. There is also the option of going back to an earlier version that worked. It is possible to set up a private server with Git, since it is open source. But the more popular option is to use already established services, like GitHub, BitBucket or GitLab that provides this as a service.

For this project GitHub is used. GitHub is a free Git service that allows users to easily make and manage repositories. The repositories are used for storing Terraform code related to the project. The use of GitHub improves the groups managing, collaboration, version control of the files, and also make it easy to create backups of the work.

2.8 Provisioning

This section describes Continuous Integration (CI)/Continuous Delivery (CD), and how GitHub Actions can be used for deploying code.

2.8.1 Continuous Delivery Pipeline

A continuous delivery pipeline is a set of actions the code goes through. These actions can consist of multiple tests, and deployment to multiple environments. The pipelines can be defined as code. This makes tests and deployments consistent, which mitigates the risk of errors in manual configuration. Having a pipeline that runs the code directly from a repository, compared to deploying from multiple team members local computer, will remove the complication of keeping track of and deploying the correct version.

2.8.2 GitHub Actions

GitHub Actions is GitHub's platform for CI/CD. In GitHub Actions, workflows can be triggered when an event occurs in the repository or be triggered manually. Workflows can contain one or more jobs, that can run in sequential order or in parallel. Each job has one or more steps, which can be scripts you define, or an action, which can be apps from GitHub or self-written.

The server that runs the workflows is called runner and can be GitHub-hosted or self-hosted. A self-hosted runner can be a virtual machine in Azure and be a part of the infrastructure [57]. In this project GitHub Actions is utilized to test infrastructure code and to provision the IoT and web page cases.

Chapter 3

Development Process

This segment will explain how the code was developed under the course of the project, the different processes used, and decisions made related to the development process. The chapter will further cover the decision of choosing related IaC tool for the project. It will also explain how the code is structured, and the reasoning for the decisions. The point of this chapter is to give the reader insight into how the project was solved, by giving a better understanding of the workflow.

3.1 Development

The project was organized using Scrum as a development model, which gave the possibility to easily change and adopt frequently, throughout the course of the project. The Scrum sprints were two weeks long, and started with a sprint review meeting of the last sprint, and a sprint planning meeting for the next sprint. Sprint meetings were held on Mondays with a biweekly recurrence. The meetings between the employer and the group where held at the same days as the sprint meetings, and involved a brief introduction to status of the project, often through a recap of the sprint meetings. Further information about Scrum and the groups reasoning for choosing Scrum as a development model is explained in greater detailed in Appendix A Project Plan section 4.1.

The GANTT chart created in the Project plan (see Appendix A Chapter 6 on page 11), was used by the group through the whole project. The group followed the general flow of the chart, but adopted when needed. The specified dates from the chart was not followed completely, however the general flow of development was. The GANTT chart was a useful tool during the project, because it provided a visual presentation of the upcoming tasks, and deadlines for obligatory submissions.

The groups time sheet, (see Appendix C) is a document where all members could register when, what, for how long and with whom they worked with. The time sheet gave the group a structure way to keep track of the time spent by each member, and how much time was spent on each task. Time spent by the group and respectively for each member is presented for each week. The group used the time sheet to keep track, to see that the group was on schedule with the expected time to use on a Bachelor project from NTNU, of 1600 hours.

3.2 Assessing and deciding provisioning tool

Azure provides multiple options to provision infrastructure to the cloud. The infrastructure can be provisioned manually through the GUI, using scripts and commands, or through infrastructure provisioning languages. The project purpose is to create templates with the possibility for automation and reusability, because of this reason manual configuration and provisioning using GUI is not viable options.

Both scripting and Infrastructure as Code are two viable options. Azure CLI and Azure Powershell are the scripting options in Azure and can be run from Windows, Linux, MacOS, Docker or the Azure Cloud Shell[1], but the Azure AZ PowerShell module needs to run in PowerShell [58]. Infrastructure languages like Azure Resource Manager, Biceps or Terraform are options to provision infrastructures in Azure. For the project, these three languages mentioned are the only viable options, because the employer have experience with these languages.

3.2.1 Chosen provisioning tool

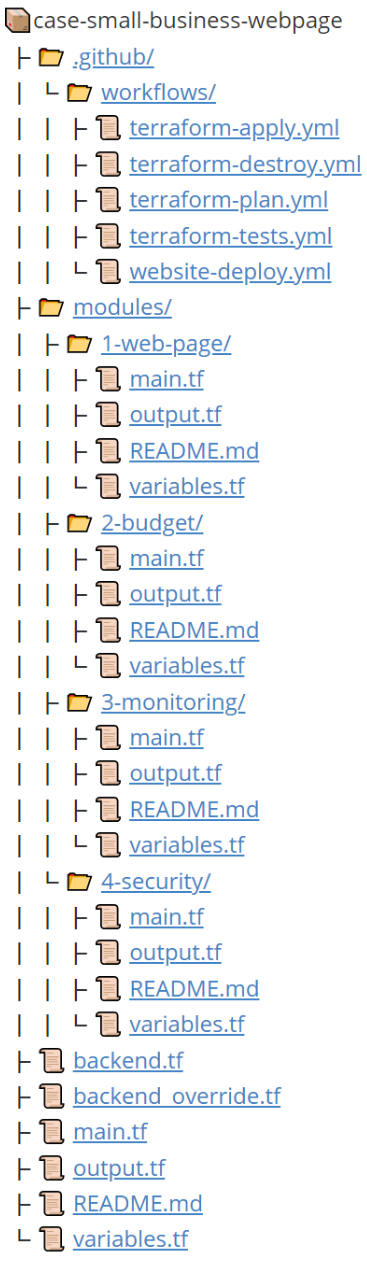
The group decided to use Terraform as the provisioning tool for this project, based on several factors. One aspect was that Terraform can be used with many different public and private clouds. Terraform will ease the change of cloud provider, compared to a cloud specific IaC tool like ARM. If the employer decides to swap from Azure to another cloud provider when using Terraform, the code will need to change, but the syntax of the code is the same. By using a cloud specific language, the change could be more difficult, because it will create the need for another language.

A negative factor when comparing Terraform to ARM and Biceps, is that Terraform can have some delay before new features in Azure are available. ARM and Biceps have all functionality available from the time the resource are available in Azure, because both languages are created and maintained by Microsoft.

[1]Azure Cloud Shell is a command line interface accessible through the Azure portal

The learning outcome for the group was another factor for choosing Terraform. Terraform provides more advanced functionality compared to ARM templates. The group members are also more likely to use Terraform in the future, because it can be used with multiple cloud providers.

3.3 Code structure



```
case-small-business-webpage
├ .github/
|  └ workflows/
|  |  ├ terraform-apply.yml
|  |  ├ terraform-destroy.yml
|  |  ├ terraform-plan.yml
|  |  ├ terraform-tests.yml
|  |  └ website-deploy.yml
├ modules/
|  ├ 1-web-page/
|  |  ├ main.tf
|  |  ├ output.tf
|  |  ├ README.md
|  |  └ variables.tf
|  ├ 2-budget/
|  |  ├ main.tf
|  |  ├ output.tf
|  |  ├ README.md
|  |  └ variables.tf
|  ├ 3-monitoring/
|  |  ├ main.tf
|  |  ├ output.tf
|  |  ├ README.md
|  |  └ variables.tf
|  └ 4-security/
|  |  ├ main.tf
|  |  ├ output.tf
|  |  ├ README.md
|  |  └ variables.tf
├ backend.tf
├ backend_override.tf
├ main.tf
├ output.tf
├ README.md
└ variables.tf
```

Figure 3.1: File and Folder Structure from the web page repository

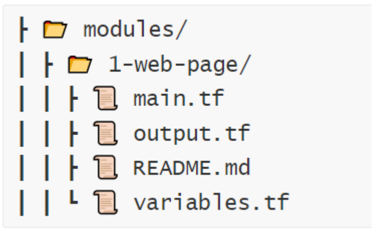
The repositories with the Terraform code consist of multiple files and directories. The code is divide into modules to increase the readability. The modules consist of code that creates a set of resources, that will typically belong together. Some of the code in a module is coupled, because the resource depends on one or multiple other resources. The modules are placed in the modules folder in the root folder, like demonstrated in Figure 3.1. The root folder of the repository is a module, which is referred to as the root module. A GitHub organization named "terraform-azure-iac" (https://github.com/terraform-azure-iac), contains all the code and repositories created for this project.

3.3.1 Repository

The product is divided into multiple repositories to create a clear structure and separation between code with different purposes. The final product is going to deliver both a base template and two case templates. The base template and the case template is created in different repositories to structure all the code, and increase the usability of the code. When the code is in different repositories it is easier to retrieve the needed code to create the resources with the wanted functionality. The base template repository is created to have a variety of different resources, and will function as a code library. The repository is divided into modules with the resources that are generally used together, to improve the easiness and time spent to create a case template from the base template. The case templates demonstrates the usage of the base template and how a template can be used for a specific case. The case templates have their respective repositories, to demonstrate how a potential case template can look.

The product also includes a repository for configuring remote backend for Terraform, and a repository for GitHub Actions self-hosted runner. These two parts of the project can be used with both the IoT case and the web page case, and other future cases.

3.3.2 Modules



```
├ 📂 modules/
| ├ 📂 1-web-page/
| | ├ 📜 main.tf
| | ├ 📜 output.tf
| | ├ 📜 README.md
| | └ 📜 variables.tf
```

Figure 3.2: A module in the repository

This project utilizes modules to group together coupled resources or resources with similar functionality, like in Figure 3.2. The purpose of dividing and grouping resources in modules is to increase reusability and readability. Modules will give the user an easier way of reusing only parts of the code, compared to all code being in one file. Another benefit of modules in Terraform is when testing the code, Terraform provides the option to only plan and run a specified module.

3.3.3 Files

Both the root modules and the child modules in both the base template and the case templates contains a set of Terraform files. All modules have a main.tf, variable.tf and output.tf files. These files contains resource blocks, variables blocks and output blocks, respectively. See Section 2.5 for an explanation of the blocks function and purpose. This separation of code and naming convention is a decision made by the group to create better reusability and readability. For Terraform the filenames do not affect the configuration, except for override files. [2]

The main.tf file is where code to create the infrastructure is, using the resource blocks. In addition to the resource blocks, the main.tf file in some cases contains data blocks [3]. The resource blocks will use the variables from the variable file when creating infrastructure objects, and the output file get information from the main file.

The root modules have in addition a file named backend.tf. This file have a Terraform block that configures the settings for Terraform, where the version and

[2]Terraform will use resource objects in files ending in _override.tf instead of the same resource objects in files without the _override.tf ending.

[3]Data blocks are code blocks being used to retrieve data from a remote source, like the cloud. An example can be to retrieve an id of an existing resource or subscription in the cloud

source of the provider is specified. In addition some authentication information to the cloud provider are set. For the project a backend_override.tf file was used, for the backend, to prevent authentication information being stored in GitHub. This file will also have the code for the state file. This will either be nothing, when utilizing local state [4], or a specification of the Azure resources used as the Terraform backend.

Dividing modules into multiple files for the different code blocks increases the reusability and readability of the modules. For making changes to the infrastructure resources as seamless as possible, variables is used. Naming, numbers, resource types, versions, notification receivers and more a set in the variable blocks in the variable files. Some changes still needs to be done in the resource blocks, because the resources can be very different based the configuration. If every aspect is set in the variable blocks, the complexity will increase exponentially.

The Terraform code can be used to provision different environments making use of the .tfvars files in the prod, stage and test folders, see Figure 3.1. The environments are normally separated into their own Azure subscriptions. Each .tfvars file is then used when provisioning to the corresponding Azure subscription. In this project, only one Azure subscription is used. Provisioning to multiple environments is simulated by provisioning to the subscription with one of the .tfvars files. How the variables from .tfvars files are passed in the code is explained in Section 2.5.3. The setup of multiple environments will be explained in greater detail in Section 9.5.

3.4 State

This project utilized both local and remote backend, which holds the state file. Each of the case templates have one state file that keeps track of the resources in Azure for that case. The two variants of state types provides different functionality and benefits as explained in Section 2.5.4 Terraform State. Local state was used in the start of the project, because it provided the option of dividing the workload based on different modules, and then use local state for testing on specific modules. The use of local state in this stage of the project prevented errors when working on the same code. The use of remote state became necessary when multiple members worked on the same module. Remote state also became needed for the pipeline runner GitHub Actions used for provisioning.

Before one can use storage in Azure as remote backend, the storage resources must be provisioned. In the project, local state is used when provisioning these resources. When the resources are in place, future IaC provisioning utilizes the

[4]Local state is the default setting for Terraform

remote backend to store the state file. This process and the technical implementation of local and remote state file is explained in Section 9.1.

Chapter 4

Requirements for Base Template

This chapter will go over the requirements for the base template, and the overall goals for the finished template. The base template will work as a resource library, that contains the different types of Azure resources a SMB might need. Because of this, the requirements and goals is relative open, with a few set expectations on the result from the employer.

4.1 Goals and requirements for the base template

The goal for the base template is to have a large set of infrastructure code, for the employer to easily copy and adopt based on the task at hand. The template is therefore divided into modules, to structure the code, for improved readability and improve the understanding of the coupling between resources. The modules in the base template covers different areas and resources. The size of the modules and the amount of resources they cover varies. Some modules give a more concrete insight to a smaller amount of resources, while other gives a larger picture, with presenting an example infrastructure and how some resources are linked together.

The finished base template should consist of many different resources in Azure, that can be combined into a infrastructure of choice, to solve a specific task for a SMB. For this the base template needs to cover different aspects that a SMB might need, like storage, monitoring, IoT device handling, information processing or setting up a web-page.

Goals
- Facilitate the use of a standardized method for building new infrastructure in Azure
- Have the capability to cover the most relevant needs for a SMB
- To improve time needed to build a new infrastructure in Azure
- Make the process of moving a service to the cloud less expensive
- The template should be easy to use and understand

Table 4.1: Goals for base template

The goals, stated in Table 4.1, were set by both the employer and the group. The goals for standardizing, meeting needs for SMB and making the template understandable was goals set to meet the requirements from the employer. The goal for reducing time and making the templates cost efficient, where goals that was set by the group, to make a potential cloud migration for a SMB more beneficial.

4.2 Security and Compliance

Security and compliance was an important part of the project, and needed to be addressed in both the base and case template. For the base template it was decided that some standards from the ISO 27000 family and the NSMs Grunnprinsipper for IKT-sikkerhet were needed to be taken into consideration when making the template. From the ISO 27000 family the most relevant standards to be considered was the ISO 27001, 27002 and 27017, which contains standards related to information security management, security controls and implementation and aspects related to the a cloud environment. Since the base template would function as a resource library, it was decided that it was not possible to have the base template fully compliant, since some of the aspects would only come into consideration when configuring for a scenario. The base template should therefor only follow the general ideas and controls from the ISO and NSM standards.

For achieving the best security in the Azure environment, the tools provided by Azure should be used. These tools is sufficient for the security aspects related to the functionality that the base template provides.

Chapter 5

Implementation Base Template

The base template is created as a library of resources that can be used to create an infrastructure for a specific case. It is designed to take parts of the template's resources, and modify them for a scenario. The module names creates an indication on what types of resources that can be found in the module. Some resources and modules can be provisioned almost without change, but other needs modification based on the specifications from the scenario. The base template has seven modules with resources. How the modules works will be explained in Section 5.1. The code for the base template can be seen in the repository terraform-azure-base-template (https://github.com/terraform-azure-iac/terraform-azure-base-template).

5.1 Modules

The modules in the base template are Key Vault, Budget monitoring, Monitoring, Web Page, Storage, IoT and Desktop Virtualization. The next sections will go into the different modules in detail. There are some overlap in both resources and functionality in the modules, with the purpose of increasing the usability of the base template.

5.1.1 Key Vault

The Key Vault module is a small and simple module that only creates one resource. The code defines a simple, but fully functional Key Vault with some basic permissions. The Key Vault is created in a way it is easy to add permissions and functionality like network ACL to the resources. For more info on Key Vault, see Table 2.1.

5.1.2 Budget monitoring

Budget Monitoring has the purpose of giving the user control over the cost of their infrastructure in Azure. The module allows the user to monitor their spending in an environment, and get a notification when the spending goes over the set budget. The module is easy to implement with other modules and requiring minimal change. Most of the changes that needs to be done are either changes to variable's names and values, or direct changes in the main file, like changes to the notification settings. The budget monitoring is helpful for the goal of making it less expensive. With the information gained from monitoring the cost, the user will have a better understanding of the costs in Azure, and see potential resources that could be removed or improved.

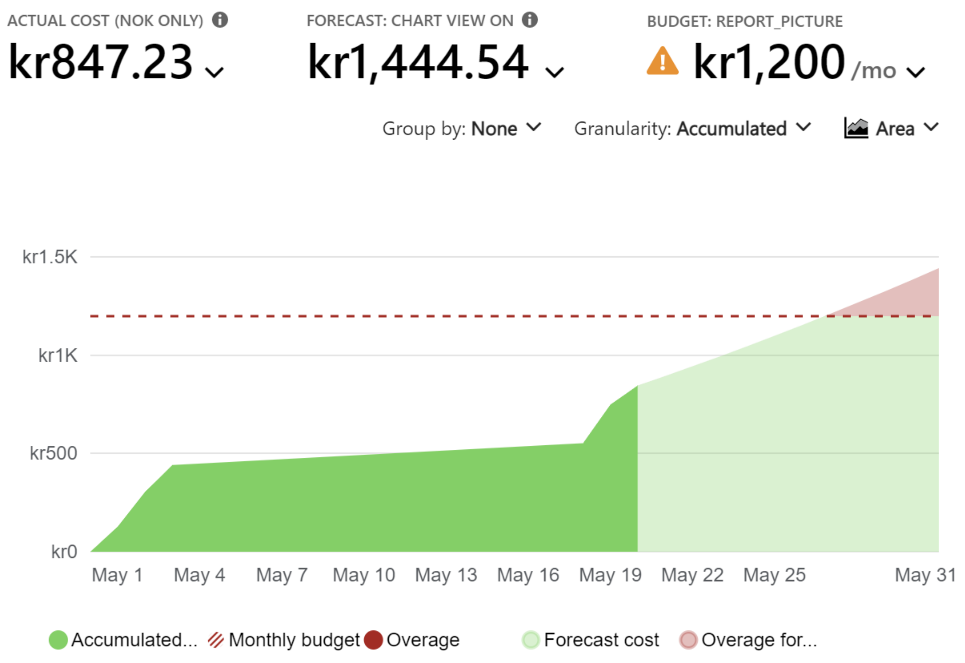


Figure 5.1: Budget graph for resource group

The notification alerts can send out alerts based upon 2 different metrics, actual or forecasted spending. For the alerts to trigger, the spending on the chosen metric, must meet the specified percentage of the total budget. This means that there could be an alert for when spending has reached 50% and one when 100% has been reached. Figure 5.1 shows a budget of 1200 NOK. With the code implementing a notification when the forecasted cost surpasses 100% of the budget. A notification will be sent to the user, because the forecasted cost is over the budget as it shows by the light red area in Figure 5.1.

For sending alerts to the users, an action group is needed. An action group is a defined collection of notification preferences [59]. The action group notifies users when an alert is triggered. In the budget module this will be when the environment passes the set budget. The action group will send notification to the specified receivers. For this module, the action group is configured to notify through email and webhook. When using a webhook, notifications can be sent to applications like Microsoft Teams. The action group can also be modified to send messages to other Azure resources like Event Hubs, Azure Functions or Logic Apps.

5.1.3 Monitoring

The monitoring for the case templates consists of mainly 3 different components. Metric alerts, Action Groups and Activity Log Alerts. The metric alerts are monitoring specified metrics on a resource in Azure. The metric can be things like errors, total connections, availability and etc. The metric alerts are defined in the Terraform code, and will trigger an alert when a set of predefined criteria are met. Example of this could be that the metric for total errors is equal to a set number that shows deviation from the norm.

While metric alerts show how a resource is doing, activity log alerts can tell what happened with or on a resource. Activity log alerts are set up on a resource, or the subscription, and targets a specific action to trigger an alert. The targeted resource and action to monitor is defined in Terraform, and will trigger an alert when the specified action happens. An example could be a deletion of a security group triggering an alert.

The notification process for alerts works in the same way as in Section 5.1.2. When the action group is defined, a Metric alert or a Activity Log alert, just needs to be associated with it. And the specified method for notifying in the action group will be used.

5.1.4 Web Page

The Web Page module is created to fit different users' approach to running a web page. The module is not fully functional as it is, but it creates a lot of different resources that can be paired together to create and operate a web page. The module contains network, load balancing, security, monitoring, database, Cache and computing resources.

The computing resources are the resources running the web page. If storage for the web page is needed, one can use the database resources. Redis Cache can be used to improve the speed of the web page, either by working as a Cache for the web page or the database. The network and load balancing resources are created with the purpose of connecting resources together, and dividing the load between resources. Security functionality is provided in the module to prevent successful attacks on the web page and to stop or log malicious, abnormal and unwanted activity.

The web page case in Section 7.2, presents a possible way how the resources in the base template can be used and modified to fit the need for a specific case.

5.1.5 Storage

In the storage module, there are different approaches for storing data. Most of the storage resources in Azure are used together with other resources, and are rarely needed on their own. In the module there are mainly 2 different types of storage, consisting of a solution for objects and a solution for relational data. The reason for this is that there are many different ways to store data in Azure, but which one to use is case specific and depends on the needs for an resource. This means that making a standardized example of all the different possibilities is not necessary, so the 2 most commonly used methods was chosen to be showcased. This module uses Blob as the solution for storing objects. This resource allows users to store files like videos, images or other types of data in the cloud environment. The other example implemented in the module, is for setting up and configuring SQL databases. These two options was included in the base template, since they could be used independently in a case scenario.

5.1.6 Internet of Things (IoT)

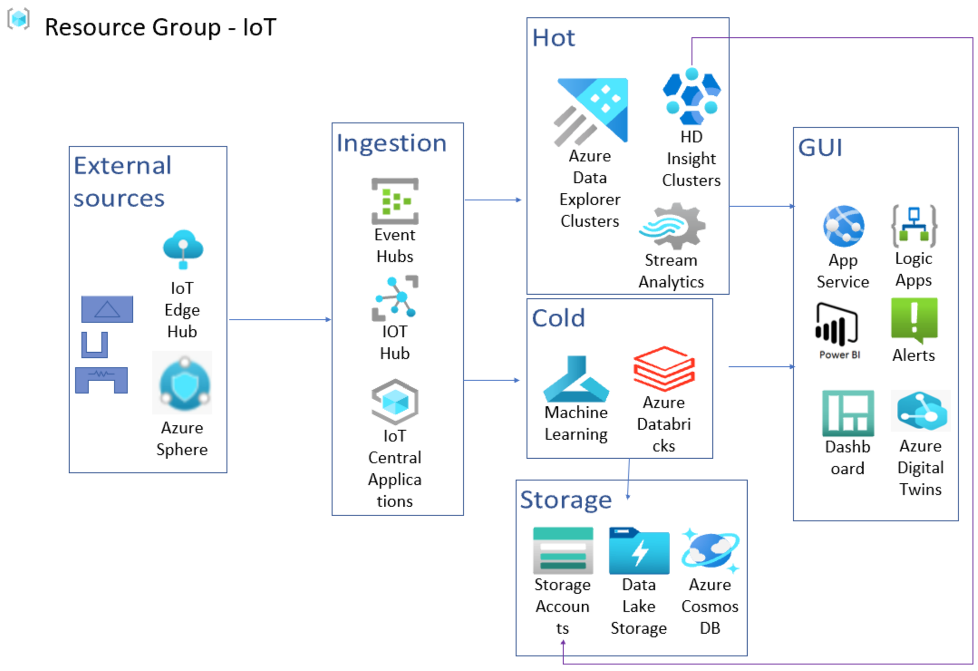


Figure 5.2: Visual presentation of the IoT module Infrastructure

The IoT module, visualized in Figure 5.2, provides a large library of Azure resources with functionality that can be used with IoT devices and handling their data. The base template is a library of code blocks and the module is created so the user can select the needed code for their case. Sections are created in the code to group resources with similar functionality together. Resources are divided into sections for ingestion, storage, GUI, hot path and cold path. The ingestion section consist of the resources for data collection from the IoT devices. The hot and cold path have the purpose of analyzing the data, but is split based on functionality and speed. The storage section is for storage resources, and the GUI section is for resources creating a interface for presenting the data collected to the end user.

Ingestion

The ingestion section consist of Iot Hub, IoT Central and Eventhub, see Table 2.5 for explanation of the resources. IoT Hub and IoT central are resources created with the purpose of connecting to IoT devices and structuring the data received. Event hub can also handle IoT data, but it is not created solely for IoT. The ingestion section for a specific case can contain one resource or a combination of multiple. IoT Central is in most cases easier to learn and understand compared to IoT Hub. IoT Hub require higher technical knowledge to use, but also has different functionality. The ingestion section contains the different resources to meet requirements and knowledge level from different users.

Data analyzing

This section will cover the hot path and cold path used to analyze the data from ingestion. The cold path are slower, but provides more advanced functionality. The resources in the cold path are Azure Machine Learning and Azure Databricks. Both the resources provides Machine Learning (ML) functionality. Benefits of cold path resources are that they can learn and adopt automatically. The hot path provides resources that can analyze data at near real-time. The benefit and the reason for the hot path resources are that they analyze the data consecutively. The hot path will provide information to the end user quicker than the cold path. The hot path section consist of Azure HDInsight, Azure Stream Analytics and Azure Data Explorer resources that all provides data analytic functionality. HDInsight uses existing open-source frameworks like Kafka, Spark and Storm for the clusters [60]. The two other resources are created to manage large of mount of data in near real-time, by utilizing Structured Query Language for Azure Stream Analytics and Kusto Query Language[1] for Azure Data Explorer and improving functionality by using functions from a programming language in the query language [61][62].

Storage

The storage sections presents possible resources that can be used as part of an infrastructure for handling IoT data. Data storage is important for both IoT data and log data, because it will be used to compare data to look for changes. The storage section contains three resources, a Storage Account, CosmosDB and a Data Lake, which are the most relevant storage resources for handling IoT data. In addition more storage resources can be found in the storage module, see Section 5.1.5. The resources supports storage of both structured and unstructured data. The Data Lake storage is designed for large amounts of data storage and high throughput. A benefit of Data Lake Storage is that it can handle different types of data, from raw data to structured data. Data Lake Storage can be integrated with multiple of the other resources in this module, some of them are IoT Hub, HDInsight and Power BI [62]. Storing raw and unstructured data is especially necessary, when storing unprocessed messages from IoT devices. The CosmosDB is integrated in the module because it can be configured to store both SQL and NoSQL data.

[1]Kusto Query Language is a query language, created by Microsoft, to handle structured, semi-structured and unstructured data.

User Interface

User Interface is created with the purpose to meet the potential requirements and knowledge of different users. The resources vary in implementation and adaptability. The App Service can be adapted completely to meet the needs, but it also require more time and knowledge. The user interface provided by the App Service needs to be created using code, based on the requirements of the user. Power BI and Dashboard have built in functionality to create tables and visual presentation of information and data. Logic App is created to send notifications based on event triggers, but can also create workflows integrating multiple components. The module creates the infrastructure resources, but to create the user interface, the resource needs to be configured. The user needs to decide which resources to use based on the time and knowledge they have with the functionality needed.

Digital Twins is a PaaS resource creating graphs and models to gain insight into a object like a building or a farm. The use of digital twins will be in cases where a user are monitoring a large space or object using multiple sensors, and need a visual presentation of status and connectivity of the sensors. In addition to the resources in the GUI sections, some of the resources in the ingestion and data analyzing sections have built in functionality to display data. The visual displays these resources have built in are limited compared to the resources from the Graphical User Interface section, and therefore the dedicated resources are created.

5.1.7 Desktop Virtualization

The desktop virtualization module is created to meet potential requirements for virtual desktops. Implementation of Azure Virtual Desktops is different from the traditional use of virtual machines with a public IP address and open RDP or SSH port. Azure Virtual Desktop is configured with a pool of VMs that users can get access to. Access control is managed by Azure Active Directory, and the users connect to the virtual machines by logging into the remote desktop application [2] or by logging in via a browser.

5.2 Security and Compliance

For achieving good security standard in the base template, the security features Microsoft Defender for Cloud and Azure Sentinel is enabled. The configuration for Microsoft Defender for Cloud, with the use of Terraform is simple, since there is not much to configure. For enabling the feature, its mostly about activating

[2]"Remote Desktop Application" is not the same as "Remote Desktop Connection (mstsc)"

the right parts of the subscription, to cover the resources that is found in the environment. The code will also be able to set up the contact information for the Defender, so that the right personnel get an alert when something happens.

Microsoft Defender for Cloud works in a way that is similar to the well-known Microsoft Defender, the main difference is that it's made to keep your cloud environment safe. The Defender will regularly scan for abnormalities in the environment, and alert the set personnel when something has happened. Defender for Cloud also has an integrated compliance feature, that shows the overall compliance with security standards in the environment. From this the defender will recommend changes to policies, resources, etc. to meet compliance.

Another security feature in Azure is Sentinel. This feature is different from the Defender for Cloud. Sentinel is considered as a cloud native SIEM solution. Defender for cloud handles incidents after they have happened, and the Sentinel will provide the tools that may indicate that an incident is about to take place or are taking place.

The base configuration for Sentinel can be done by Terraform, trough setting up a log analytic with the Sentinel resource. To get a fully functional Sentinel, a part of the configuration needs to be done with the GUI or through Azure CLI. The use of Sentinel requires a person with some knowledge in the field of incident handling to fully utilize the resource, so the use of Sentinel might not be applicable for every company. Sentinel is not an essential part of the overall security for the environment, but is a great tool for companies that is more exposed for attacks, and can help detect attacks that uses unpatched exploits or ongoing breaches in the environment.

Chapter 6

Requirements for Case Templates

The case templates will be used as examples on how the different resources in the base template can be used to build an infrastructure to solve a specific task. The case templates will each have a hypothetical scenario, which are IoT device management and web page for a SMB. The chapter is divided into sections describing the requirements for the IoT case, for the web page case, and lastly the requirements that is identical between both cases.

6.1 Requirements for IoT vehicles case

The hypothetical scenario this case template is based upon, is the collecting of different types of data, from vehicles in a company. The optimal user for this sort of template, is a SMB where their employees work is depending on a company car. The template will be able to monitor different types of IoT devices, and process the data that they produce. Further will the template have monitoring for the budgeting, and other metrics that might be interesting. For the template the needed compliance and security standards like ISO, will be followed. The IoT case template can be easily modified to fit other scenarios where SMBs use IoT to monitor other devices and objects.

6.1.1 Goals for IoT-vehicles case

Goals
- Be able to manage IoT devices and handle IoT data
- Be able to monitor azure resources (relevant and interesting statistics)
- Be able to monitor and manage the overall cost of the infrastructure
- Be able to ensure security and compliance of the environment

Table 6.1: Goals for case IoT template

Table 6.1 specifies the most important goals for the IoT template. The code in the case template should automatically set up the infrastructure needed, to get an environment for IoT devices to be connected to. When they are connected, the monitoring of the resources should alert the administrators if something is deviating from the norm. Since the target group for these templates are SMBs, the cost of the infrastructure is something to be considered. Therefor a dedicated monitoring of the total cost of the infrastructure is deemed necessary. The goal is to alert the administrators of the environment when the resources is costing more than the set budget. The template should also meet the necessary security and compliance standards.

6.1.2 Security and Compliance

GDPR is a relevant aspect to consider when collecting data from vehicles. This is because information about the employees' location and behavioral patterns can be collected and stored by the IoT solution. To be compliant with GDPR, the company must implement "appropriate technical and organizational measures for ensuring that, by default, only personal data which are necessary for each specific purpose of the processing are processed." [63].

With the use surveillance technology in Norway there are some aspects that are not able to be addressed with the code. When a company wants to use such technology, they have an obligation to inform the employees and discuss the circumstances around why and how its being used, as per the "Working Environment Act" [64] [65]. This aspect is something a potential user of the template must address.

6.2 Requirements for web page case

The web page template is developed for provisioning infrastructure for hosting a web application or a website for a SMB. This template has a large potential use, because most businesses today have the need for a basic web page. This template should be easy to adapt and combine with other templates or modules.

6.2.1 Goal for small business web page case

Goals
- Be able provision the resources necessary to host a web page
- The infrastructure to be able to handle different amount of traffic
- Be able to monitor azure resources (relevant and interesting statistics)
- Be able to monitor and manage the overall cost of the infrastructure
- Be able to ensure security and compliance of the environment

Table 6.2: Goals for web page case

Table 6.2 presents the most important goals for the SMB web page case. The code for the web page case, should automatically set up the infrastructure needed to host a fully functional web page. The infrastructure should be able to easily handle a fast growth in traffic to the web page. The case should also provide an option for storage, to store web page relevant content like inventory. Load balancing between the instances hosting the web page is another important requirement for the case, because this will improve the speed and availability of the web page.

6.2.2 Security and Compliance

Requirements to comply with laws and standards is highly dependent on what the web application is used for. The web page case is made to host a basic web page with information about the business. If personal data is stored in the web application, GDPR regulations must be followed. If the web application is used by businesses that must comply with laws and regulations of the respective industry sector, these must be considered.

6.3 Requirements in common

For most cases there are some shared requirements independent of the scenario. For the IoT and Web page case this is budget, monitoring and security and compliance. For a SMB the capability to monitor their spending is a essential tool for keeping their infrastructure cost effective, avoid unnecessary spending and evaluate the gains versus the cost of a service. The other part that all SMBs would want is a way to ensure that their infrastructure is functional, with the help of monitoring. With monitoring the SMBs are able to monitor and evaluate different metrics on important parts of their services, to ensure that everything is working as it should, with the feature of being notified when something is wrong. The monitoring of metrics can either help them get their infrastructure back to normal functionality, or help the administrators spot potential malicious activity in their environment.

In addition to the actual use of the case templates, to create an infrastructure in Azure, the purpose of the case templates is as well to demonstrate how to use the base template to create new templates. In addition to the actual code of the case template, it should be paired with a detailed description instructing how the template was made using the base template. This is to give the employer a greater understanding of how to use the base template to maximize their benefit.

6.3.1 Budget

Budget and spending are important factors for all businesses, and it is also important for the cloud infrastructure. Notifications should be sent to one or more persons or applications, at given spending events. There should be a possibility to set notifications at different amount spent for a period. In addition to notifications for the actual amount spent for the time period, there should also be the possibility for notification when the estimated spending for the period surpasses a percentage of the budget.

A scenario could be that they have a budget of x amount per month in the cloud, and they want to get updated on how much they spend. They could set up notification for when they have spent half the budget, and then check if they are half way through the month or if they need to cut some cost for the rest of the month. It can also be interesting to have multiple notifications on different percentage spent of the budget. Additionally to the actual spending they want to get notified when the estimated spending for the month surpasses the budget.

6.3.2 Monitoring

Monitoring in this context is referring to resources and functionality that observe and check for incidents in the cloud. An incident could be anything from a deletion or creation of a resource to latency or health changes of a service. For a web page the response time might be a significant factor while for an IoT infrastructure the number of connections and failed jobs might be more significant. The main idea for these requirements are the same for both cases, but the details of implementation will be different, because the cases use different resources with different metrics to monitor. When a incident is recorded, the resources need to send a notification. The notifications should be sent to the administrator responsible for the part of the infrastructure.

6.3.3 Security and Compliance

Both of the case templates needs to follow compliance and security practices for their respective scenarios. The case templates should follow the same standard as in Section 4.2, but since the case templates will be more specific, more controls and recommendations from the standards should be implemented. These implementations should either be done trough configuration with Terraform code, use of security features provided in Azure, or a third-party service.

The goal for the case templates is to follow security and compliance relevant for their respective scenarios, which will lead to an increase in overall security for the environment. To check if this goal is meet, it was decided to utilize Microsoft Defender for Cloud and Checkov for checking configurations, in the environment and in the code.

Microsoft Defender for Cloud has a feature for checking compliance in the environment, see Section 2.2.3 for the compliance feature. This feature will be used for checking compliance with ISO 27001 and the Azure Security Benchmark V3 for both of the case templates. Checkov is a third party tools used for finding misconfigurations in the Terraform code, that might lead to a security or compliance problems [66], this is achieved by checking the code with their own database and policies. The use of Checkov will help the templates with security and compliance, while getting rid of mistakes in the Terraform code.

Chapter 7

Implementation of case templates

The case templates are created based on the base template. This leads to it being some overlap between the code and functionality in the base template and the two cases. The parts of the cases vary in the amount that is duplicated from the base template. This chapter is separated, to distinguish the case specific segments from the equal segments.

The first sections have focus on the parts of the cases that are specific to the respective case. How each case are implemented is explained in great detail in the following sections. Lastly in this chapter there will be a section describing the similar functionality for the cases and the base template. The functionality in the last section is not identical, but have large overlaps. The adjustment made to fit the specific case will described, to present how the code was adjusted for the case.

7.1 IoT-Vehicles Specific

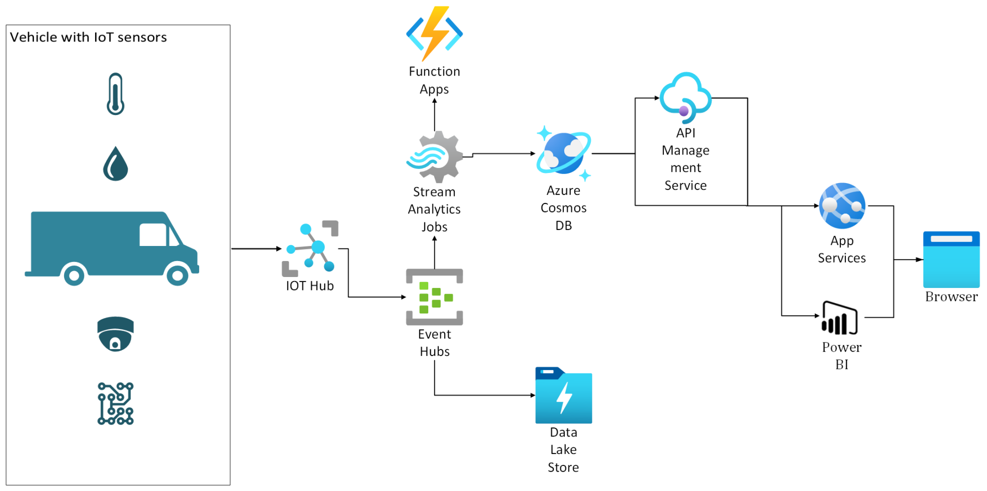


Figure 7.1: IoT infrastructure diagram

The case template consist of 4 different modules; IoT-Vehicles, Monitoring, Budget and Security. All 4 modules are made from their corresponding modules in base template, with some configuring to fit the specific scenario. The IoT-Vehicle module, as visualized in Figure 7.1, handles the connection and overview for the IoT devices, storing, processing and display of the data related to the IoT devices. The monitoring module handles alerts on relevant resources, when a metric or activity is meeting the set criteria. Budget keeps track of the spending on the environment and can be altered to meet the companies budget expectations. Security is enabling different security features in azure, to keep their environment as safe as possible, and compliant with regulations. This section will focus on the case specific module for IoT, while modules in common will be addressed in Section 7.3. Figure 7.1 is a visual representation of how the resources in the case specific module is connected, and how the data flows in the infrastructure. The code for the IoT case is in the "terraform-azure-iot" repository (https://github.com/terraform-azure-iac/terraform-azure-iot). The code for the implementation explained in section 7.1.1 through section 7.1.6 can be found in module 4-IoT.

7.1.1 Ingestion

IoTHub will be handling and connecting the different types of IoT devices to the cloud. The IotHub will have a connection string that the IoT devices will need to incorporate into their configuration. When the configuration for the device is set, the device will start to send data into the IotHub. IoTHub will make the data available for processing by other resources in Azure. The Terraform code will only set up the basics of the IotHub, so that the resource is available in the Azure environment. The connection and configuration of the devices is not within the scope of the project.

7.1.2 Information processing

For the IoT case template, the goal was to implement the underlying infrastructure to achieve the possibility of an information stream from the IoT devices to the users. The IoTHub will collect the data from the devices and send the data to an Event Hub. From the Event hub the data can either be stored into a Data Lake, or be processed by Stream Analytics and then stored into a database for further use. Stream Analytics will provide the capabilities to analyze large amount and complex data at real-time.

At this moment the case template will only set up the resources, and configure them together only to what the capabilities of the Terraform code allows. Overlying configuration, which data to show, and how it will be presented is at the moment not incorporated into the case template. This is due to the projects time allotment, and limitations of the Terraform provider. There will therefore be a need to configure this manually, and later automatize this through Azure CLI scripts, or trough extracting an ARM template of the finale setup, to achieve a automatized and reliable infrastructure.

7.1.3 Storage

The IoT module is created with two resources for data storage. The resources have different functionality and is created to store data from different resources and of different types. Data Lake storage is created with the purpose of storing all information from the IoT devices. Azure Cosmos DB will store analyzed data and make it available to the end user through the user interface. Cosmos DB creates the possibility for the App Service and Power BI to access data. The Cosmos DB SQL database stores the data in a structured format, but Cosmos DB can also handle unstructured data.

7.1.4 Function App

The Function App will have the job of reacting to web requests using HTTP triggers, and it also has the possibility to analyse IoT data. Code blocks can be directly stored in the resource and be activated to run at a schedule or when it is triggered.

The Function App will react to the output sent from Stream Analytic resource. For the case the resource will react to IoT devices' errors it receives from Azure Stream Analytics.

7.1.5 API Management

API Management is created to manage multiple resources through a centralized API service. The API Management have a integration with Event Hub, Function App, Web App and the Monitor resources in Azure. Terraform does not provide the possibility to create the integration between a service or a resource and the API Management, therefor this is not a part of the template and needs to be done manually. The resource also have the feature to integrate with resources outside of the cloud if needed.

7.1.6 User Interface

The module is created with two resources, App Service and Power BI, to present data from the IoT devices. The App Service resource will work as a highly adaptable solution to present the data in the way wanted by the end user. The resource can also have triggers that send notifications to end users. Power BI will present the data using graphs and tables, and will be easily implemented using the built in functionality. The configuration of the two resources is not a part of the template, and needs to be done after the infrastructure is provisioned.

7.1.7 IoT - Security and Compliance

To be compliant with GDPR, as explained in chapter 6.1.2 - Security and Compliance for IoT, the company must implement technical and organizational measurements. The technical measurements configured in this template for complying, consists of protecting data in transit with TLS encryption [67], and at rest with Azure Data Encryption at rest [68]. More security implementations are described in Section 7.3.3.

7.2 SMB web page Specific

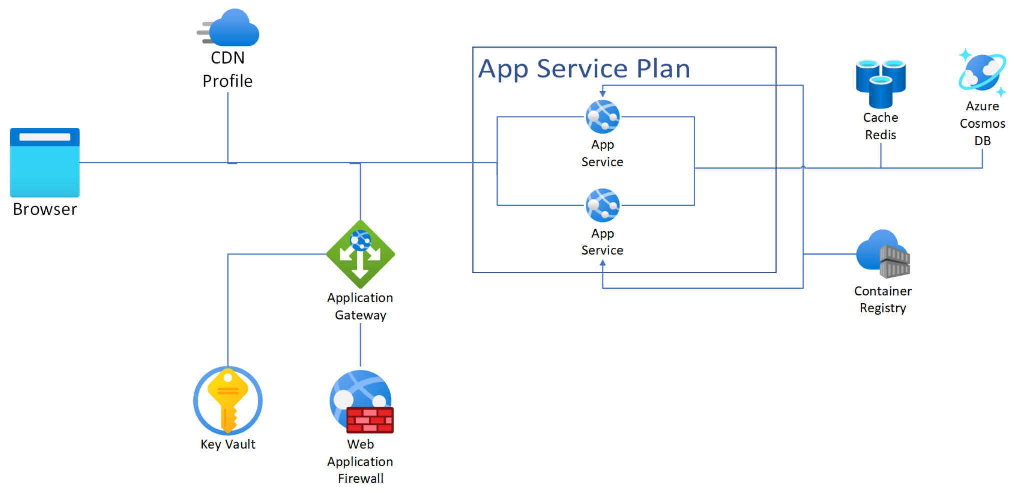


Figure 7.2: Web page infrastructure diagram

The web page case template consists of the 4 modules web-page, budget, monitoring and security. All 4 modules are created based on modules in the base template. The web page module, as visualized in Figure 7.2, provisions resources for hosting a web page or a web application. Budget, monitoring and security is configured in their own modules. Resources in the web page module will be explained in this section, while budget and monitoring is explained in Section 7.3 Things in Common. Figure 7.2 presents how the infrastructure resources is connected and how the general traffic flows when the web page is used. The code for the web page case template can be found in the "terraform-azure-webpage" repository, (https://github.com/terraform-azure-iac/terraform-azure-webpage). The code for the implementation explained in section 7.2.1 through section 7.2.6 can be found in module 1-web-page.

7.2.1 App Service

The App Service resources hosts the actual web content and web applications. Azure App Service Plan specifies the compute resources that the web applications run on. The web page case template includes two identical App Services that will run different code. An example of how the two App Services will be used, is one hosting the general home page and employee information code and the other hosting the web site's code for the company's previous projects.

The template creates an automatic scaling setting for the App Service. The automatic scaling feature scales the App Service out when the CPU usage of the resource exceeds the set limit. This means a new instance will be added to the App Service. A new instance is not a new resource, and do not require any configuration. The automatic scaling feature also scales the App Service inn if the CPU usage is under a given limit.

7.2.2 Application Gateway

The Application Gateway does the load balancing in front of the App Services. It is configured with a public IP and a backend configuration that tells which App Service to send traffic to. A SSL certificate in the Key Vault is also specified in the configuration, to secure the transport of data. One example of how the Application gateway can be used, is utilizing information from the example in Section 7.2.1. The Application Gateway will analyze the web address; when "/projects/" is in the URL, traffic will be sent to the App Service responsible for handling the request, while traffic to other web addresses will be sent to their corresponding App Service. A Web Application Firewall (WAF) feature is applied on the Application Gateway, to prevent and detect against common web hacking attack types.

7.2.3 Key Vault

Azure Key Vault, as explained in more detail in Table 2.1, is a service that securely stores confidential information. In this template, the Key Vault is used to store SSL certificates used by the Application Gateway. In this case the use of SSL certificates is necessary, because ensures secure transport of data.

7.2.4 Database

Azure Cosmos Database provides the option of storing data in different formats. A CosmosDB account is required for creating CosmosDB databases. The account can utilize different database types like SQL, MongoDB among other. The template creates a SQL database, but can easily be adapted to provision a NoSQL database.

7.2.5 Cache

Cache is used at two places in this templates infrastructure. Redis cache sits between the database and the App Service, and caches backend data for the App Service. The other caching functionality happens between the user and the Application Gateway. Here, the Content Delivery Network (CDN) can be configured to cache data in point-of-presence locations close to the users.

7.2.6 Container Registry

The web page infrastructure also consists of a private Docker container registry. Docker images can be pushed to the registry and deployed to App Service. How to utilize the web page infrastructure to build and deploy container images is explained in the next section. Images can also be built in the registry by using Azure Container Registry Tasks. Builds can be triggered when pushing changes to GitHub.

7.2.7 Demo Web Page

The web page case template includes a demo of how the resulting infrastructure can be used to deploy and host a web page. A GitHub Actions pipeline is used to build and deploy a docker container to the container registry, which then is deployed to the App Service. These steps are defined in the deployment pipeline shown in Code listing 7.1. The GitHub Actions Runner, which is explained in Section 9.2, is the compute resource running this process.

```yaml
name: website deploy

on:
  push:
    paths:
      - 'src/*'
  workflow_dispatch:

defaults:
  run:
    working-directory: ./src/

jobs:
  build-container-image:
    runs-on: self-hosted

    steps:
    - name: 'Checkout GitHub Action'
      uses: actions/checkout@main

    - name: 'Build and Push Image to ACR'
      uses: azure/docker-login@v1
      with:
        login-server: WebPageContainerRegistry.azurecr.io
        username: ${{secrets.TF_ARM_CLIENT_ID}}
        password: ${{secrets.TF_ARM_CLIENT_SECRET}}
    - run: |
        docker build . -t WebPageContainerRegistry.azurecr.io/test:${{
    github.sha }}
```

```
        docker push WebPageContainerRegistry.azurecr.io/test:${{ github
    .sha }}

  deploy-to-web-app:
    needs: build-container-image
    runs-on: self-hosted
    steps:

    - name: Webapp Deploy
      uses: azure/webapps-deploy@v2
      with:
        app-name: app-service-web-page-test
        publish-profile: ${{ secrets.AZURE_WEBAPP_PUBLISH_PROFILE }}
        images: 'WebPageContainerRegistry.azurecr.io/test:${{ github.
    sha }}'
```

Code listing 7.1: Website deployment pipeline

7.2.8 Web Page - Security and Compliance

As explained in 6.2.2 - Security and Compliance for the web page case, the configuration is dependent on what the web application will be used for. If personal information will be transferred, stored and processed, the IaC must be configured to comply to GDPR.

The template have integrated some security features on the resources to increase the security. The resources having internet access are configured to only allow HTTPS traffic. In addition TLS version 1.2, the newest version available in Azure, are configured on all resource with the configuration option. Newer TLS version have better security features compared to older versions. The SSL certificate used by the Application Gateway are stored securely in the Key Vault.

7.3 Things in Common

This section will go into detail on the parts in the case templates that overlap with each other and with the base template. This will give only a brief introduction on the topic or module, and reference to the base template implementation for more detailed information. Minor adjustment that is needed for the specific case in the implementation will be described.

7.3.1 Monitoring

The monitoring module for the cases is mainly doing the job of creating an action group, defining who alerts is sent to and metric alerts to the different resources relevant for the respective case. For the monitoring module, parameters for which metrics to monitor on what resource, is done trough the Terraform code. For both cases, the metrics added is chosen based on what information would be useful. Changes to the Terraform code, for removing, adding or edit existing metric alerts is an easy task. In addition to metric alerts, both cases have activity log alerts implemented. They are configured to alert when changes to a security group are made.

7.3.2 Budget

The configuration of budget and alerts on budgets are similar in the base template, IoT case and web page case. What the budget module is and how it is used, is explained in Section 5.1.2 Budget monitoring.

7.3.3 Security and Compliance

The security and compliance requirements described in the requirement chapters for base template (4.2), IoT (6.1.2) and web page (6.2.2) is accomplished trough multiple layers. With static code analysis, the infrastructure code is scanned for misconfigurations that can cause security or compliance issues in the infrastructure. With Azures Compliance "Manager" in the Defender for Cloud, the infrastructure in Azure is evaluated based on policies and standards.

Azure

The use of compliance and security features in Azure is explained in chapter 5.2. For the case templates the compliance feature integrated in the Microsoft Defender for Cloud, is also used to make a spreadsheet of current compliance problems in the environment. This is done to get an overview of the current configuration problems on the resources, to give a deeper understanding of what needs to be changed.

At this time, the compliance feature in the Defender only has some predefined policies available for import and managing by the feature. For other policies the Azure Policy tool can be used to manually add the policies, from other relevant standards like GDPR, but the templates does not contain a full import of these policies with Terraform.

For the security aspects in the case templates, Microsoft Defender for Cloud and Sentinel was used. Their configuration is not optimal with Terraform, so a limited configuration of the solutions is done. Terraform only allowed for some different subscriptions in the Defender to be turned on or of.

Code Analysis

Code analysis is done prior to provisioning by implementing Checkov for code security tests in the provisioning pipeline. The tests can be configured to check according to best practice policies and custom policies created by the user. Code security tests can be implemented into the development workflow. This can be done by triggering tests when pushing code to a repository, merging changes into main branch, or in a provisioning pipeline where infrastructure deployment is dependent on test success. The technical implementation is described in Section 9.3.

Chapter 8

Evaluation

This chapter will focus on the IaC product, and evaluate the result against the goals for each part of the product. The chapter is structured in three sections, where each section evaluates one template. Each section will start with an analysis of the product and whether the goals was reached. This section will also contain the reasoning for the resource and configuration choices. The second section will be an evaluation of the templates security and compliance.

8.1 Base template

The base template is a library of code, to be used to create infrastructure for different cases. This section will evaluate the base template, by analyzing if the goals and security and compliance requirements is met in the base template.

8.1.1 Goals

The goals for the base template is described in Table 4.1, and Chapter 5 explains the implementation of the decided solution. This section will evaluate if the goals was reached through the implementation.

Facilitate the use of a standardized method for building infrastructure

The base template facilitates a standardized method for building new infrastructure in Azure. By defining all infrastructure as code, building and making changes to environments can be done through defined methods. This way developers can go away from doing manual changes which are unknown to others and not stated in code, which can lead to unexpected consequences. Manual changes in one of the environments may cause tests to behave differently between the test runs, which can lead to unnecessary troubleshooting.

The base template is functioning as a library, and allows users to have the same basis when making new templates for specific cases. With the possibility to add more into the library, and by using the library, developers can share the same methods without difficulty. The building of new templates will then become more standardized.

Cover different needs for a SMB

Another goal for the base template is to cover different needs that SMBs might have for a cloud environment. The content of the base template covers some of the most relevant cloud infrastructure components for SMBs. This includes IaC for Key Vault, web page, storage, IoT, virtual desktops and monitoring of budget and resources. The functionality in the base template covers different basic tasks, that a SMB will benefit moving on to the cloud.

Improve the time needed to build a new infrastructure in Azure

To improve the time needed to build new infrastructures in Azure, the base template includes IaC for provisioning multiple types of resources. The base template includes modules consisting of individual resources, and modules that are composed of multiple resources to create useful environments in Azure. Use of the base template improves the time needed to build new infrastructures in Azure, compared to writing the code from scratch or manually configuring the environments.

Make the process of moving a service to the cloud less expensive

The goal of making the process less expensive is tightly connected with the goal of reducing the time needed to build a new infrastructure in Azure. The reason why time reduction is a important part of decreasing the cost is because it leads to the employer having to spend less man-hours. Less man-hours provides the possibility to reduce the price of the process of moving to cloud. A focus for the base template has also been to use the most cost efficient resources to solve a task. When multiple resources could provide similar functionality the lowest costing resources was used.

The template should be easy to use and understand

The template should be easy to use and understand, was also a goal of the base template. Dividing the code into modules and then separating the code into multiple files creates a structure for the repository that eases the use of the template. The README files explanations and descriptions of the files, folders and code structure improves the understanding of the repository and code. The naming policy for files and folder is a significant reason for reaching this goals, because it gives the user a easier understanding of the code placements.

8.1.2 Security and compliance

When making the base template one of the goals was to cover different needs for a SMB. With this goal there was also a need to ensure that the functionality in the base template was secure. In Section 4.2 the general standards followed for the base template is described. Compliance with these standards were automated to the extent that Terraform allowed, but some aspects would either require use of scripts or GUI, and due to time limits this was not a part of the scope. Also with the base template being a resource library, most of the functionality would not be fully functional or finished with configuration, which means that total compliance was not entirely possible with the base template. This meant the focus on compliance for the base template was not to have it fully compliant, but rather use the standards to guide the code to meet best practice.

The decision to make the base template a general resource library, instead of making specific templates for sectors, as the original project description mentioned (explained in Section 10.2.1), was based upon assessments made by the governing bodies in Norway. A research paper made for the Norwegian Parliament, about migrating government sectors to the cloud, pointed out potential problems with migrating sensitive task and information. The base template was accommodated to allow SMBs in the sectors for the government, to move simple functionality onto the cloud. To accommodate for the regulations, all the resources made in Azure was set to be located on data centers in Norway. This was to ensure that the data would be under jurisdiction of only the Norwegian government, to avoid the potential of other countries being able to access the data [69].

The other problem was for potential SMBs in a sector related to the military, based from a research paper made by the Norwegian military. The paper concluded that in the current state of cloud technologies, there are no real solutions to address handling of classified information in the cloud, since the security clearance of the systems have no clear guidelines [70]. Based on the issues raised by the government and military, it was clear that SMBs related to these sectors, would most likely only use cloud technologies to handle simple tasks. This meant that the

SMBs in these sector, would have most use out of the same functionality as SMBs in public sectors. Based on the possible issues raised by the governing bodies, the base template accommodates the issues in the code, to the extent that Terraform and Azure allowed.

Some of the security and compliance aspects are not able to be addressed in the code, or be automated, since it needs to be addressed by the company on a managerial level. For an example, when a company wants to do a cloud migration, they are obligated to do a risk assessment and a DIPA [71]. These are things that the base template cannot address.

8.2 IoT case

Both the IoT case's goals and the security and compliance requirements for the template will be evaluated. The section will cover an evaluation of whether the implementation have reached the requirements for the template. The requirements consist of both the goals for the template and the security and compliance requirements. The goals have a focus on the functionality of the template, while the security and compliance requirements are the controlled by the laws, rules and best practices for the fictional case.

8.2.1 Goals

The IoT case's main goals is specified in Table 6.1 and the explanation of the implementation, with the ambition of meeting the goals, is explained in Section 7.1. This section will evaluate if the goals for the template was reached through the implementation. The goal, "Be able to ensure security and compliance of the environment", will be covered in Section 8.2.2

Manage IoT devices and handle IoT data

The main goal of the IoT case is that the template should create an infrastructure able to manage and handle data from IoT devices. Iot Hub is an important resource for making the connection between any IoT device and the cloud. Since the resource have the functionality to both send and receive messages from an IoT device, it is an important part of reaching the goal. Storing the gathered IoT data is important to make it available in the future.

The reason for gathering IoT data is to make the information available to the end user. The template creates resources that can present the data to the end user.

In addition to present the data, the infrastructure will also provide the possibility to notify an end user based on an event. The notification option is not a requirement for the infrastructure, but it gives an improved user experience. Function App can send notifications that can help with reducing the users response time to an incident. Analyzing the data, using the Stream Analytics, will improve both the notification and the presentation of the IoT data. The API Management resource is not necessary for the infrastructure, however it provides the possibility to configure other resources. The addition of API Management will improve the user experience, because the user will have a single place where the they can configure other resources using an API.

Monitor Azure resources

The monitor module in the case template is important to reach the goal of monitoring Azure resources. It triggers notifications based on specified events on resources in the infrastructure. By sending notifications it will reduce the response time and therefor reduce the mean time to recover(MTTR). Important resources like IoT Hub, Event Hub, App Service and storage account are monitored. An example is the number of connections and failed jobs with the IoT Hub. It is important to control that an IoT devices is not missing, or that a additional IoT devices appears, because this can be security threat to the organization. Number of errors are measured to make sure the resources are fully functional. There are also metrics on the App Service to make sure the GUI is available to the end user. Monitoring of the Storage Account's number of transaction can reveal misconfigurations, or unwanted and malicious activity if more data than normal are being transferred.

Monitor and manage cost

Being able to monitor and manage the overall cost of the infrastructure, is another goal for the template. The template improves the users control of cost by notifying the user when the cost surpasses an specified amount. Separating cost by the subscription and a resource group is an advantage of the template. The subscription gives information about all the resources in the template. Additionally it provides the benefit of making any provisioned resources a part of the cost analysis, so there are not any unknown extra cost at the end of the period. The budget set on the IoT module's resource group gives a more detailed insight into the cost of the most important infrastructure, and what the specific resources are costing in the infrastructure. By notifying the user at given percentage of the budget, respectively for the subscription and the resource group, makes the template reach the goal of having control of the overall cost.

8.2.2 Security and compliance

For the IoT case the most important aspect to be considered is the possible complications when dealing with surveillance. Both the GDPR and governing bodies in Norway have very strict rules related to this. Before potential SMBs can use the case template, they need to do an assessment into why they need it, potential risks connected to it, what data they need to store and how they are going to handle that data. The Working environment act in Norway is what prevents companies from doing unnecessary surveillance of their employees. To be compliant they need to be transparent with them and make the reason and area of use for the surveillance clear [64] .

IoT Hub was chosen for the template, despite the fact that IoT Central is considerably easier to use and understand, because IoT Central is not currently available in Norway. Since data flow outside of Norway can lead to potential violations in some sectors, the best option in this case is to utilize IoT Hub.

For general security in the template, Defender for Cloud provides the necessary functionality by handling and alerting on potential incidents, while also recommending changes to be made in the environment to follow best practice. For IoT in Azure, there is a feature called Microsoft Defender for IoT, but this feature is not available for Terraform, and the cost for the feature could be undesirable to a SMB with the limited tasks it can do. With security there are only some aspects that can be addressed in the code, and with proper training for the employees, good policies and following best practice overall, the security implemented to this case should be enough to keep the environment secure.

To evaluate the security and compliance in the IoT template, as per Section 7.3.3, the compliance feature in Microsoft Defender for Cloud was used. From the use of this feature, a spreadsheet containing all missing compliance in the environment was put together. The spreadsheet contains missing compliance specifically for the Azure Security Benchmark V3 and ISO 27001:2013 standards, see Appendix D. The missing compliance was not addressed in the code, due to time constraint, with respect to the time it would have taken to automate and correct. For a new resource or changes to a resource to appear in the defender, it takes 24 hours or more. This means that the process for checking possible solution, in the code, is very time consuming. The other problem related to this process, is the constraints with Terraform, specifically problems with automating or that a fix is only available in the GUI. Based upon these factors, it was decided to deprioritize this task, for remaining problems see Appendix E.

8.3 Web page case

This section will cover an evaluation of the Web page case, and it is structured in the same way as the Section 8.2, but the requirements are different for the two templates. Like the previous section it will first be an evaluation of the goals and functionality of the template, followed by the evaluation of the security and compliance of the case.

8.3.1 Goals

The goals for the web page case is stated in Table 6.2, and how the case is implemented is explained in Section 7.2. This section will evaluate if the implementation of the case accomplished the goals. The goal, "Be able to ensure security and compliance of the environment", will be covered in Section 8.3.2

Provision the resources necessary to host a web page

The main goal for the web page case, is to be able to provision an environment in Azure that can host a web page. Through the implementation in Section 7.2, the resulting infrastructure is capable of hosting a web page. The web page can be built with the most common languages and frameworks, and deployed from a Git repository. The infrastructure is also able to host containerized web applications. The database resource is a part of reaching the goal by working as a backend for the web page or web application. Cache is configured both between the user and the web page, and between the App Service and the database and improves the speed of the web page.

Handle different amounts of traffic

The goal of having the infrastructure be able to handle different amounts of traffic, is accomplished by configuring the App Services to automatically scale based on resource usage. The App Service components will scale out [1] when the resource usage is over a threshold, to be able to handle increase in traffic. When the resource usage goes bellow a threshold, the components scale in to lower the cost. The load balancing functionality in the Application Gateway and internally in the App Service, are beneficial to the environment, because it balances the load between the App Service resources and instances.

[1]Scale out means adding instances, while scale up means adding resources to the instance

Monitor Azure resources

The monitoring module in the web page case lets the user monitor relevant and interesting metrics in the infrastructure. The web page's HTTP response codes is monitored for errors on client and server side, and the response time is monitored and set to alert when it is over a threshold. The module is also configured to monitor cache misses and errors in the Redis Cache, availability and latency of the Key Vault, indications of Distributed Denial-of-Service (DDoS) attack and activity on the Network Security Group (NSG).

Monitor and manage cost

To be able to monitor and manage the cost of the infrastructure, budgets with alerts are configured on subscription level and on resource group level. The subscription level budget gives administrators an overview of the cost in the environment, and the resource group level budget monitors a defined resource group. The budget module is configured to notify an administrator when the forecasted consumption reaches a threshold, and when the actual consumption exceeds a threshold.

8.3.2 Security and compliance

Many of the points in Section 8.1.2 and Section 8.2.2 stay true for the web page template. The security for the template is mainly done by Defender for Cloud, with some specific configuration done in the code to enable functionality, like only allowing HTTPS traffic. The compliance feature in Microsoft Defender for Cloud is also used to map compliance and unresolved problems for this template, see Appendix F and Appendix G. The compliance issues is not automated for the same reasons as in Section 8.2.2.

Chapter 9

Provisioning

The chapter explains the setup and usage of the provisioning process in the project. This includes setup and usage of remote backend and GitHub Actions. Section 9.3 goes through how remote backend and GitHub Actions is utilized to provision the Terraform code into Azure using pipelines. Several security considerations, provisioning to multiple Azure environments and static code testing is also discussed in this chapter.

Shared storage for Terraform state file must be provisioned before the pipeline-runner can use it. In this project, this is accomplished by separating this setup from provisioning of the cases. The remote backend and pipeline-runner is provisioned and configured first, then used in provisioning of the cases. The code for provisioning remote backend is in its own repository named "terraform-azure-backend" [1], and the code for GitHub Actions self-hosted runner in the "terraform-azure-github-actions-runner" repository [2].

9.1 Terraform Remote Backend

As explained in section 2.5.4, the state file should be accessible for everyone in the team, including the pipeline runner. This is accomplished by storing it in an Azure Storage Account and referencing it in the Terraform code.

[1]Remote backend repository: https://github.com/terraform-azure-iac/terraform-azure-backend
[2]GitHub Actions repository: https://github.com/terraform-azure-iac/terraform-azure-github-actions-runner

9.1.1 Provisioning the Remote Backend with Terraform

The setup of the remote backend is a two-stage operation. The first stage is to provision the resources, which is a resource group with storage account and storage container. The second stage is to reference these resources in the backend configuration, as shown in Code listing 9.2, when provisioning new infrastructures.

The stages to achieve this is:

1. Provision storage account and container without backend block (see code listing 9.1).
   This will create a terraform.tfstate locally
2. After provisioning; use the references to the storage account and container in the backend.tf when provisioning new infrastructure (code listing 9.2).

```
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = "~>2.0"
    }
  }
}

provider "azurerm" {
  features {}

  subscription_id = var.subscription_id
}
```

Code listing 9.1: backend.tf without backend block

```
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = "~>2.0"
    }
  }
   backend "azurerm" {
    resource_group_name  = "terraform"
    storage_account_name = "terraformbackend"
    container_name       = "terraformbackend"
    key                  = "terraform.tfstate"
  }
}

provider "azurerm" {
  features {}

  subscription_id = var.subscription_id
}
```

Code listing 9.2: backend.tf with backend block

The remote backend can now be used when provisioning new infrastructure. For provisioning with a pipeline, reference the storage account and container in the backend.tf file. For provisioning from local computer, add the references in backend.tf and run `terraform init`. This will offer to migrate the state file to Azure storage, as explained in figure 9.1. From now on, resources that is provisioned will be managed by the state file in Azure storage.

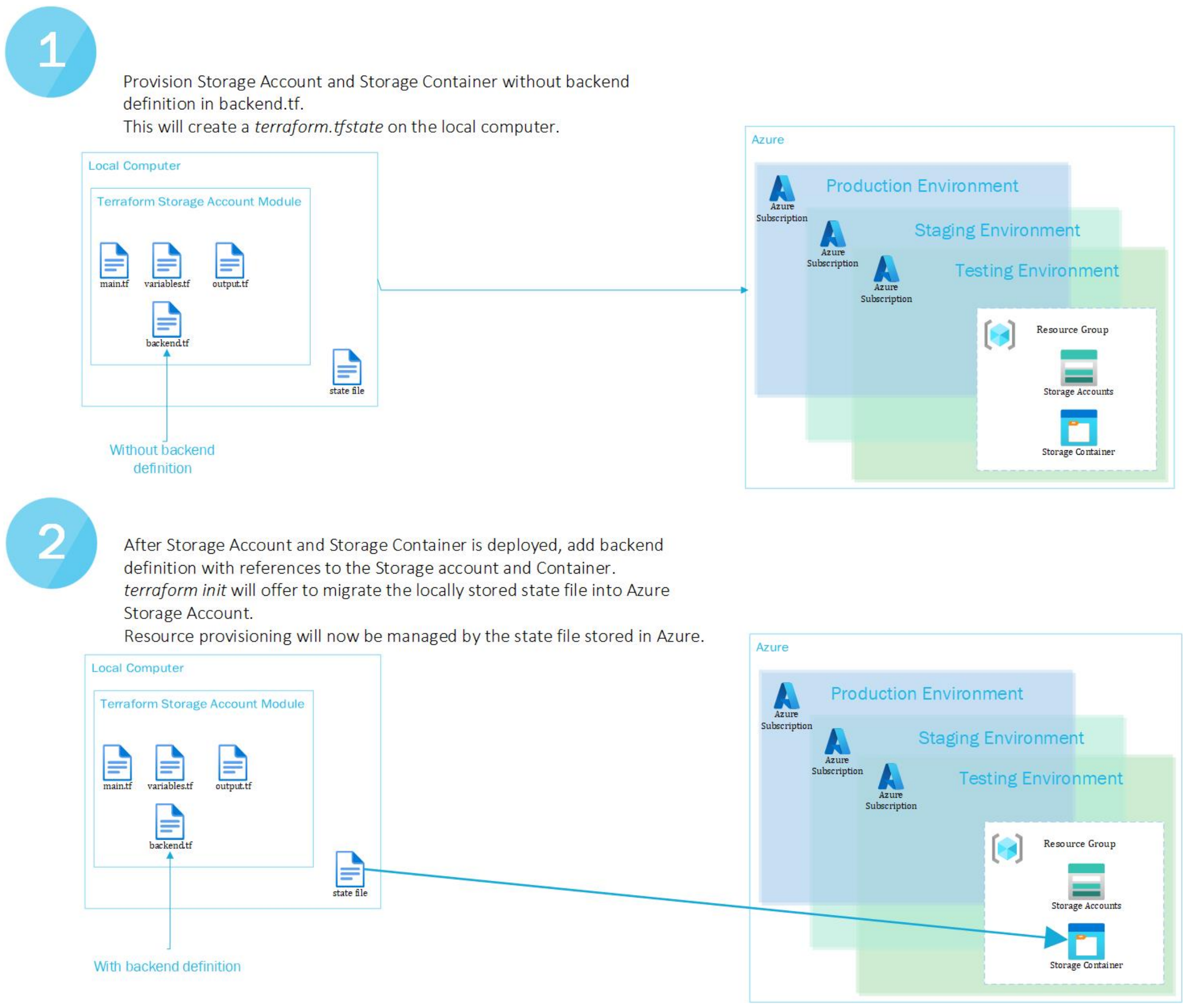

1
Provision Storage Account and Storage Container without backend definition in backend.tf.
This will create a *terraform.tfstate* on the local computer.
Local Computer
Terraform Storage Account Module
main.tf
variables.tf
output.tf
backend.tf
state file
Without backend definition
Azure
Production Environment
Azure Subscription
Staging Environment
Azure Subscription
Testing Environment
Azure Subscription
Resource Group
Storage Accounts
Storage Container
2
After Storage Account and Storage Container is deployed, add backend definition with references to the Storage account and Container.
terraform init will offer to migrate the locally stored state file into Azure Storage Account.
Resource provisioning will now be managed by the state file stored in Azure.
Local Computer
Terraform Storage Account Module
main.tf
variables.tf
output.tf
backend.tf
state file
With backend definition
Azure
Production Environment
Azure Subscription
Staging Environment
Azure Subscription
Testing Environment
Azure Subscription
Resource Group
Storage Accounts
Storage Container

Figure 9.1: The state file is migrated from local to remote

The backend in Azure is now managed by a local state. This means that it cannot be removed with terraform destroy without first going back to the local state. This is a positive effect, since the backend should continue to exist while other resources changes.

9.2 GitHub Actions Self-Hosted Runner

As explained in Section 2.8, GitHub Actions is GitHub's platform for CI/CD. This repository contains code for provisioning and configuring resources in Azure to run GitHub Actions.

Runners in GitHub are the servers that the pipeline runs on. These servers can be GitHub-hosted, or self-hosted. The repository provisions a self-hosted runner in Azure. The repository consists of Terraform modules for network, Linux virtual machine, Key Vault and Bastion. In addition, a PowerShell script that manages the Service Principal.

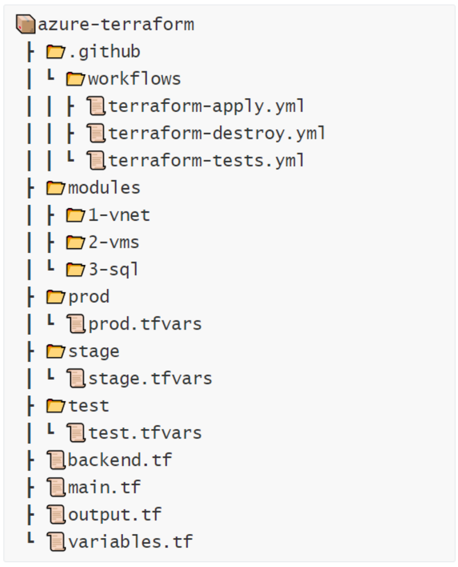
Azure Service Principal is an object in Azure Active Directory that is referenced in GitHub Secrets. The Service Principal gives the pipeline permission to deploy resources in Azure. The Linux virtual machine is the machine that runs the pipeline. Key vault is used to store the private SSH-key used to connect with SSH to the VM. Azure Bastion is a service that provides remote desktop and SSH connection to virtual machines without exposing the connections to the internet.

9.2.1 Provisioning GitHub Actions runner with Terraform

The GitHub Actions self-hosted runner must also be provisioned before it can be used. This means that a developer must run the Terraform code from a local computer. If the backend for Terraform is set up, as explained in Section 9.1.1, it can be used in this deployment.

The resources are provisioned with Terraform, and the virtual machine is configured with a bash script which is runs during the initial setup. The script configures the necessities for running the GitHub Actions pipelines. A token from GitHub has to be passed to the GitHub Actions configuration in the bash script. This is done by manually copying it from GitHub and inserting it in the root variables.tf file. The token is then passed into the script when Terraform is run.

9.3 Provisioning with pipeline



```
azure-terraform
├ .github
| └ workflows
| | ├ terraform-apply.yml
| | ├ terraform-destroy.yml
| | └ terraform-tests.yml
├ modules
| ├ 1-vnet
| ├ 2-vms
| └ 3-sql
├ prod
| └ prod.tfvars
├ stage
| └ stage.tfvars
├ test
| └ test.tfvars
├ backend.tf
├ main.tf
├ output.tf
└ variables.tf
```

Figure 9.2: Repository directory with pipeline workflows

The remote backend and GitHub Actions is utilized to provision the IoT and web page cases. The provisioning process is defined as code in pipeline workflow files that are a part of the repository. For GitHub Actions, the workflow files are in the .github/workflows subdirectory as shown in Figure 9.2.

9.3.1 Pipeline workflows

A pipeline can be triggered by an event in the repository or be triggered manually. Events can be a push to a branch or a pull request. Different triggers can start different pipeline workflows. A push to the repository may for instance trigger a workflow with code validation tests, as shown in Code listing 9.3.

```yaml
name: Terraform Tests

on:
  push:
    branches:
      - main

defaults:
  run:
    working-directory: .

jobs:
  terraform:
    runs-on: self-hosted

    env:
      ARM_CLIENT_ID: ${{secrets.TF_ARM_CLIENT_ID}}
      ARM_CLIENT_SECRET: ${{secrets.TF_ARM_CLIENT_SECRET}}
      ARM_SUBSCRIPTION_ID: ${{secrets.TF_ARM_SUBSCRIPTION_ID}}
      ARM_TENANT_ID: ${{secrets.TF_ARM_TENANT_ID}}

    steps:
      - uses: actions/checkout@v2

      - name: Setup Terraform
        uses: hashicorp/setup-terraform@v1

      - name: Terraform Init
        run: terraform init

      - name: Terraform format
        run: terraform fmt

      - name: Terraform Validate
        run: terraform validate

      - uses: terraform-linters/setup-tflint@v1
        name: Setup TFLint
        with:
          tflint_version: v0.29.0

      - name: Init TFLint
        run: tflint --init

      - name: Run TFLint
        run: tflint -f compact
```

Code listing 9.3: Pipeline workflow with code validation

A pull request can trigger a workflow with `terraform plan` which lets team members go through what the code will deploy. After code review when the pull request is approved, another workflow can be triggered which runs `terraform apply` and deploys the infrastructure from the code. A pipeline that provisions Terraform code is shown in code listing 9.4.

```
name: Terraform Apply

on:
  workflow_dispatch:

```

```
 6 defaults:
 7   run:
 8     working-directory: .
 9 
10 jobs:
11   terraform:
12     runs-on: self-hosted
13 
14     env:
15       ARM_CLIENT_ID: ${{secrets.TF_ARM_CLIENT_ID}}
16       ARM_CLIENT_SECRET: ${{secrets.TF_ARM_CLIENT_SECRET}}
17       ARM_SUBSCRIPTION_ID: ${{secrets.TF_ARM_SUBSCRIPTION_ID}}
18       ARM_TENANT_ID: ${{secrets.TF_ARM_TENANT_ID}}
19 
20     steps:
21       - uses: actions/checkout@v2
22 
23       - name: Setup Terraform
24         uses: hashicorp/setup-terraform@v1
25 
26       - name: Terraform Init
27         run: terraform init
28 
29       - name: Terraform Plan
30         run: terraform plan
31 
32       - name: Terraform Apply
33         run: terraform apply -input=false -auto-approve
```

Code listing 9.4: Pipeline workflow with infrastructure deployment

9.3.2 Authentication

Provisioning infrastructure in Azure with Terraform and GitHub Actions with self-hosted runner requires two authentication processes. One is authentication between the pipeline and the Azure tenant, and the other is between GitHub and the self-hosted runner in Azure.

Authentication from the pipeline to Azure happens with a Service Principal in Azure Active Directory. In Azure AD, you create a Service Principal with permission to deploy resources. This Service Principal's information is stored in GitHub Secrets, and referenced in environment variables in the pipeline files as shown in line 15-18 in code listing 9.4. The GitHub Actions self-hosted runner authenticates to GitHub repositories using a token from GitHub, which is passed as an input in the GitHub Actions configuration on the virtual machine.

The service principal used by GitHub Actions needs a set of permissions to be able to provision the infrastructures. It is possible to set the permissions with Terraform, but in this project the management of permissions is done outside of Terraform. The reasoning for this is to make it faster and easier to make changes to the permissions. For security reasons the service principal should not have permanent rights to do changes to the environment. By using Azure CLI to manage the permissions, they can be applied when needed and removed after.

9.4 Security Aspects

Utilizing CI/CD tools can cause security risks if configuration of permissions, authentication or secrets management is not optimal. In this section, security risks and how to mitigate them is explained.

9.4.1 Arbitrary Code Execution

Risk

The code that exists in the repository will be executed when the pipeline workflow is triggered. This can facilitate arbitrary code execution if a malicious user manages to get malicious code in the repository and trigger a pipeline run.

Mitigation

To mitigate these risks, some countermeasures must be in place. In this project, the Service Principal connected to GitHub Actions is given the permissions it needs, for the time it needs them.

Team members GitHub accounts is a potential way in for an attacker. An organization using GitHub can require two factor authentication for all users in the organization to mitigate the risk of a malicious user gaining access to the repository.

9.4.2 Secrets Management

Risk

Authentication between the pipeline and Azure requires that values from the service principal is referenced in the pipeline. Loosing these values is a risk because they can give an adversary a way to execute code in the Azure environment.

Mitigation

Secrets and other sensitive information used in workflows should be stored using GitHub Secrets and referenced with variables in the workflow file. A downside of this is that it adds one more place to set up, manage and update.

9.4.3 Self-Hosted Runner

Risk

The self-hosted runner can contribute to a higher security risk, since it has the necessary authentication and permissions to create and destroy resources in Azure.

Mitigation

To mitigate the risk, we must lower the probability for exploitation, and lower the impact if the runner has been exploited. To lower the probability for an adversary to exploit the runner virtual machine, it must be secured according to best practices, such as access control, use of SSH keys and management of authentication information. To lower the impact if the VM is exposed, we must configure its permissions according to the principle of least privilege. The Azure Active Directory object that gives the VM its permissions should only contain those permissions it needs. It should not be able to do privilege escalation to get more permissions in the cloud environment.

In this project, the self-hosted runner-vm is provisioned and configured without the need for a user to log in to it. If manual configuration is needed, a user can connect through bastion. This way the communication between the users local computer and Azure is over TLS, and no port is exposed to the internet.

9.5 Multiple environments

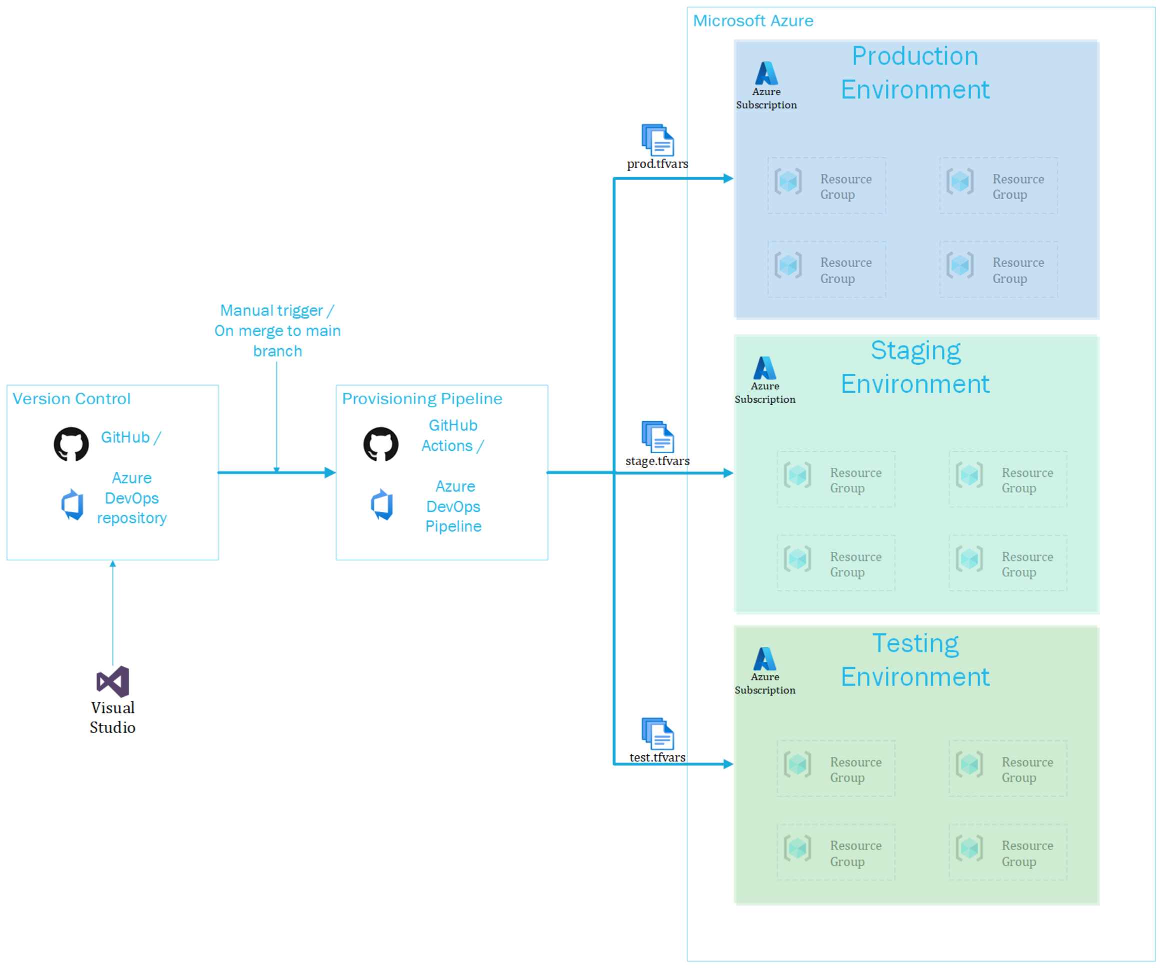


Figure 9.3: Provisioning to multiple environments

The Terraform code in the case repositories supports multi environment configuration with test, stage and production environments. In Azure, the three environments are separated into their own subscription. The same codebase provisions the three environments, and variables differentiates them, as shown in Figure 9.3. Examples of differences between the three environments are size of resources where test and stage environment require less than production, and thresholds for budget consumption alerts where consumption in production can differ from test and stage.

The case repositories use one .tfvars file for each environment. Variables set in the .tfvars files are passed to the resource provisioning code when Terraform runs. When provisioning from a local computer, the developer connects to the desired

Azure subscription and provisions the code with the associated .tfvars file. For example, when provisioning to test environment, the command is `terraform apply --var-file="test/test.tfvars"`.

When provisioning with GitHub Actions, separate jobs can provision to the different environments. Variables containing the ids of the Azure subscriptions are used in each job. The `needs` option in the pipeline creates dependencies between the jobs, so that provisioning to test environment must succeed before provisioning to stage can happen, and the same procedure before provisioning to production can happen.

9.6 Testing

The case template repositories includes pipelines for testing the infrastructure code. The tests have the two focus areas code syntax validation, and code security assessment. The syntax tests will fail if there are syntax errors in the code that would cause failure to provision. *Terraform format*, *Terraform validate* and *TFLint* are the tools used in the pipelines for syntax tests. Code security tests will analyse the code and output errors when finding misconfigurations that can lead to security or compliance issues. The tools used for the security testing, is *Checkow* and *Bridgecrew*.

```
name: Terraform Security Tests

on:
  workflow_dispatch:

defaults:
  run:
    working-directory: .

jobs:
  Checkov:
    name: Checkov Security Tests
    runs-on: self-hosted
    steps:
      - name: Run Checkov action
        id: checkov
        uses: bridgecrewio/checkov-action@master
        with:
          framework: terraform
          container_user: 1000

  Bridgecrew:
    name: Bridgecrew
    runs-on: self-hosted
    steps:
      - name: Run Bridgecrew
        id: Bridgecrew
        uses: bridgecrewio/bridgecrew-action@master
        with:
          api-key: ${{ secrets.BC_API_KEY }}
          soft_fail: false
          framework: terraform
```

Code listing 9.5: Pipeline with code security testst

Tests can be implemented in the provisioning workflow. In the provisioning pipeline, the provisioning job can be dependent on successful security tests. This way any misconfigured Terraform code will not be applied.

Chapter 10

Assessments

This chapter is an assessment of the whole project, including the thesis, product and the process the team members have been through. The result of the project will be compared to the goals set in the project plan in Appendix A. Some topics that have been discussed trough out the thesis, will be assessed in greater detail in this chapter. The whole project will also be taken under consideration when pointing out areas where the group members could have done things differently. In Section 10.4, some potential future work based on this project is presented. This is both continuous work on the result of this project, and potential new bachelor projects. At the end of the chapter, the group's work is evaluated.

10.1 Results

This section will analyze if the objectives for the project, defined in Appendix A chapter 1.2 was reached during the project. Both the impact and result goals from the project plan have focus on the product created. The scope and goals for the product changed during the project, which means that the result goals in Appendix A is not the end goals for the product. The result of the product was evaluated in Chapter 8, up against the new goals. The impact goal was partially covered in the Evaluation chapter, by evaluating the security and easiness of the templates. To summaries the template were relative secure and was easy to implement, and created resources for a cloud environment relevant for an SMB.

The learning outcome goal has focus on the group members' outcome from the project. The goal was to both improve our understanding and experience with cloud infrastructure and with project and team work. This was the group members' first project with this magnitude and size. Planning the project was crucial for the final result. The project presented the importance of a good plan and structure, as well as choosing the right development model, especially for a large project. During the project the members got a lot of experience with Azure, and how to provision a cloud infrastructure using Infrastructure as Code (IaC). It also provided a lot of knowledge about security in the cloud and existing compliance possibilities and frameworks.

10.2 Discussions

This section contains discussions about topics that have been mentioned through the thesis. These topics will be discussed in greater detail in the following section.

10.2.1 Changes to the scope

The scope for the project, based on the original assignment (Appendix B) from the employer, Bouvet, was changed to improve the result of the project. The original assignment states that the goal is to create general IaC templates for three sectors. The scope changed to creating one base template and two case specific templates.

The original scope was to create three templates for different sectors. The template should work as a basis for when the employer was creating the code for a new customer. An example would be that a new customer in the finance sector needed a cloud environment, so code from the corresponding template was used to meet their needs.

After thorough consideration and discussions, the final scope was to create one base template and two case templates. The base template should be a library of different resources, that can meet the potential demands of multiple different SMBs. In some way's, it is a combined template of multiple sector templates from the original assignment. The base template can be used for creating code for cases in multiple sectors. The case templates demonstrates how the base template can be used for different SMBs in different sectors.

There are multiple reasons for the changing of the scope. One reason is the new scope will create more value for Bouvet, because the three sector templates from the original assignment would have a lot of duplicated code. Duplication of code was reduced by creating a base template usable for all sectors. The research phase revealed that there were little changes between the compliance and security standards, for SMBs in different sectors, which was another negative side to creating multiple sector templates.

Another problem with the original scope, is that the standards and regulations for the sector might only be applicable for cloud infrastructure when used for certain tasks. Resulting in a lot of cases where only some tasks could be move to a public cloud, without a compromise in security for the SMB. Laws and regulations could only be applicable to a resource when doing a certain task, and not applicable when handling other tasks. This would lead to multiple different configuration of the same resource based on the usage of the resource. An example is a VM that is used to analyze data from the power plant, would have totally different requirements compared to a VM that will host a web page.

10.2.2 Provisioning

Azure subscription

One Azure subscription is used for the project. This creates some limitations when showing how the code can be used to provision to multiple environments, and how provisioning to a production environment can be dependent on first successfully provisioning to the test and stage environments. In the project this is solved by simulating multiple environments by provisioning with the test/stage/prod.tfvars files one at the time.

Terraform state file

The two cases in the project has one Terraform state file each, which manages all the resources. Resources in the cases are separated into modules which lets developers make changes to parts of the infrastructure. While the modules facilitates a micro stack pattern, the stack is still monolithic in the sense that one single state file manages all resources. If the state file is corrupted or unintentionally altered, the blast radius will be the whole infrastructure. An improvement would be to use one state file for each module. This change will require changing the backend configuration in Terraform, and the logic in the pipeline workflow files.

10.2.3 GitHub Actions

The GitHub Actions self-hosted runner in this project can only do provisioning in the Azure subscription it exists in. This means that to use this setup in a multi-environment setup, there must be a runner in each subscription. For cost and complication reasons, a GitHub Actions runner should exist outside of the subscriptions it will provision to, and have permissions to provision to these subscriptions.

As explained in Section 9.3.2, a token from GitHub authenticates between the runner and the repositories. In this project, retrieving the token from GitHub is a manual task that has to be done before provisioning the Terraform code for GitHub Actions runner. It exists an API for retrieving the token, but this is not made use of in the project. This is due to available time for the project, and the fact that it is still necessary to in some way authenticate to GitHub to retrieve the token.

10.2.4 Managing permissions

Managing and keeping control over users and applications permissions is an important aspect when working in the cloud. Several approaches is possible to solve this. A goal of automation is to minimize manual configuration, but a significant point is that it all has to start with someone or something that has the necessary permissions, and these permissions have to be managed.

Managing permissions with Terraform is possible. Since using Terraform is an idempotent operation, values for permissions can be changed, and running Teraform will apply the changes without altering other resources. The drawback of this approach, is the use of time for organizations that have implemented a workflow consisting of git development branches, pull request with code review and provisioning through pipelines.

For this project, Azure CLI is used to manage permissions for the Service Principal which connects the pipeline to Azure. The reasoning for this, is that the permissions should have a different life cycle compared to the rest of the infrastructure. While the virtual machine itself might continue to exist after some provisioning is done, its permissions should be removed to lower the impact if some unwanted code gets executed through the pipeline. This approach is also a simulation of how external services, such as Microsoft Identity Manager [72], can be incorporated to manage permissions.

Azure supports the option to create and use custom roles which can be assigned to users and applications such as Service Principals. The custom roles gives more detailed permissions. This can for instance be permitting the GitHub Actions Service Principal to deploy and destroy all resources except Key Vault and databases or only read and not alter data in Key Vault. Due to available time for the project, only Azure built-in roles are used.

10.3 Critics and potential improvements

This section will discuss some experiences that the group members had while working on the project, and reflections about the approach to solve the project. The section will address potential changes that could have improved the final result. It will also address any criticizable parts of the thesis, final product or the work process.

10.3.1 Writing the thesis

A change that could potentially improve the result, was to start the writing process earlier. Writing the implementation chapters during or straight after finishing each template could have improved the result of each template section in the thesis. This might reduced time spent on writhing the implementation since the code and choices where just made. Writing the requirements chapters before creating the code could have made the goals for the templates even clearer, but the downfall is that it could have made making changes to the expected result more problematic.

10.3.2 Different approach to the project

For the project it was decided to use Terraform to build and configure an environment, but during the course of the project, it was discovered that there was not enough time to do everything. A potential solution for this problem could have been to work the other way around, first build the infrastructure in GUI, with configurations, and then extract the template from the finished infrastructure. It is a possibility in Azure to extract ARM templates from running infrastructure, but it was decided that the best learning outcome for the group would be to not utilize this. The option was therefore not explored, and the full capabilities of the feature is therefore unknown. But its possible that solving the project with that method, could have lead to a better product for the employer.

10.3.3 Git development branches

During the project, the group has not utilized Git development branches. If it had been used, the group would have got experience with both the Git tool and the workflow. Different pipelines could have been incorporated in the workflow. For instance, a push to a development branch could trigger a code validation test, opening a pull request could trigger code security tests that needs to pass before merging into main, and merging into main could trigger a pipeline which provisions the infrastructure.

10.4 Future work

Future work will cover the potential tasks that will improve the product created or build on this project. A part will cover potential new Bachelor thesis's that can build on the result of this thesis.

10.4.1 Templates

Further work on the temples would include automating more of the provisioning, to avoid manual configurations of infrastructure. Due to limitations from Terraform, there are things in Azure that are not optimal to configure using Terraform. These problems could be solved by utilizing other IaC tools or through integrating more use of scripts. Scripts could solve some of the problems, by using Azure CLI to configure parts of the infrastructure that Terraform does not support.

The base template would also benefit from more functionality, to widen the possible overall use for potential SMB cases. The functionalities it currently supports are to accommodate smaller and generally useful use cases. The base template should also be extended to allow integration with other Microsoft services, like M365, to allow SMBs to move more of their environment to the cloud.

10.4.2 Compliance and Security

The current state of the templates does not handle the problems surrounding the automation of fixing compliance issues. At the time many of the issues would need manual configuration to be fixed, but parts of this could with further work be automated through the use of scripts and Logic Apps. The compliance in the templates has focus on general compliance, but this should also be extended to accommodate different case specific scenarios. Because of limited standards being currently available in Azure, other standards that are not included in the project, would need to be manually integrated into the environment through the use of policies. The process for making and automating the different needed policies would need a lot of time, which was the reason it was excluded from the scope.

Azure Sentinel could also use some further work, with automating the set up of the resource. The reason this was left out of the scope, is the fact that the use of Sentinel would require very specific knowledge in incident response, which a SMB might not have. This lead to the optimization of Sentinel to not be prioritized, but with more time it would be beneficial to complete.

10.4.3 Potential Bachelor Thesis

A potential bachelor thesis could be a general risk assessment of a cloud migration for a SMB. A deeper understanding of the potential risks a SMB might encounter with this process, will help the SMB make a choice on how a cloud migration will help them, and declare the potential problems they might encounter with the switch. A risk assessment will also help with the further development of this projects automated templates, and it may reveal potential risk factors that was not discovered.

Another thesis that might be beneficial for a SMB is to look into how to automate the complete set up of a M365 environment on the cloud. A SMB currently using the on premise solution [1], might benefit with a solution to help them move their current system on to the cloud. This could also further build on the current project, and allow the cloud infrastructure to be integrated with M365.

10.5 Teamwork evaluation

This section contains evaluations of the groups work. This involves how the group organized the work, how tasks were divided and our experiences with working on the project.

10.5.1 Organization

The group used scrum as a development model and a scrum board for managing tasks. This gave the group experience in working with the scrum workflow with backlog, sprints and daily stand-ups. Tasks were added to the backlog throughout the project, and during the sprint meetings the tasks for the sprint was picked out. The tasks that were not finished during a sprint, was moved to the next sprint.

The scrum board gave the group a clear outline of what tasks to do an a day-to-day basis, and the sprint meetings gave a good overview of the project as a whole. This agile approach made it possible to do drastic changes to the project in the beginning, and still be able to finish the project on time.

When working with this project, which consists of a code part and a thesis, the group experienced that the scrum framework and scrum board was most useful for coding tasks. The task of writing the thesis was a continuous task throughout the project period and it did not make sense to divide it into multiple tasks.

[1]The use of Active Directory and Office 365 locally on an on premise environment

10.5.2 Dividing the workload

During the project, the group did most of the work together. Since one of the team members had some previous experience with Azure and Terraform, the focus in the beginning was to get the knowledge level up for the whole group. While the two members fairly new to Azure and Terraform worked on this, the third team member had focus on provisioning. In addition, one team member had the responsibility for security tools in Azure.

10.5.3 Project as a form of work

By doing this project, the group members have acquired some important experiences. One of the experiences is the importance of stopping and taking a "step back" to get an overview of the project as a whole every two week or so. This made it possible to plan the tasks so that the project was finalized on time. Another experience was how important it is to be able to change plans. The project description was changed, and therefor also the planned schedule.

Chapter 11

Conclusion

A growing amount of SMBs want to move a part of their workload to the cloud. Bouvet is trying to help business with their cloud migration. Templates would help standardize the process and reduce time, resulting in a improved outcome. The templates should provision an infrastructure in Azure suited for a SMB. The project was determined to result in one base template, covering a large quantity of resources fitting a SMB, and two case templates showcasing the use of the base template for separate fictional cases.

The templates creates infrastructures suitable for a SMB use, but can also be applicable to larger business or organization. The Base template includes code for a large quantity of resources, and creates a good base for creating templates for a specific case. The base template have room for improvement through adding new resource blocks or new combination and connection of resources. Both case templates can potentially be used by multiple businesses. Considering most business today have a web page, the web page case will probably have a higher use rate. The IoT vehicles case have a smaller user group, but can be used for a SMB monitoring all types of IoT data through simple modifications.

For the project the focus on Security and Compliance was an important aspect, and the templates have followed some information security ISO standards and other relevant regulations for the specific cases. For the templates, best practice was implemented with Terraform to the extent that it allowed, with the goal of having it integrated and automated. For compliance, some aspects where able to be addressed in the templates, with specific choices like the location of the resources, while most of the compliance would be managed by an existing tool in Azure. For the compliance some issues with configurations was discovered, and some of them could not be resolved in the code, and would need additional integration of scripts and logic apps. Due to time limitations, it was decided that the fixes to the problems would not be automated, and that they would need to be fixed in the GUI.

During the course of the project there was discovered some limitations in the tools used, and the group feels that some aspects in the final product would need more time to be optimized. Overall is the group satisfied with the result of the project, and feels that the project has added some value to the task it set out to solve.

Bibliography

[1] I. G. Ltd. ‘Iso 27000 series of standard, The iso/iec 27000 family of information security standards.’ (), [Online]. Available: https://www.itgovernance.co.uk/iso27000-family.

[2] Bouvet. ‘About bouvet.’ (), [Online]. Available: https://en.bouvet.no/about-bouvet. (accessed: 17.01.2022).

[3] K. Moris, *Infrastructure as Code Dynamic Systems for the Cloud Age*, 2nd edition. O’Reilly Media, 2021, pp. 4–11, ISBN: 9781098114671.

[4] R. Koch. ‘Everything you need to know about gdpr compliance.’ (), [Online]. Available: https://gdpr.eu/compliance/. (accessed: 07.03.2022).

[5] E. Commission. ‘Eu data protection rules.’ (), [Online]. Available: https://ec.europa.eu/info/law/law-topic/data-protection/eu-data-protection-rules_en. (accessed: 07.03.2022).

[6] T. Holtebekk. ‘Iso.’ (), [Online]. Available: https://snl.no/ISO. (accessed: 07.03.2022).

[7] S. Norge. ‘It-sikkerhet - iso/iec 27000.’ (), [Online]. Available: https://www.standard.no/fagomrader/ikt/it-sikkerhet/. (accessed: 07.03.2022).

[8] K. Moris, *Infrastructure as Code Dynamic Systems for the Cloud Age*, 2nd edition. O’Reilly Media, 2021, pp. 108–110, ISBN: 9781098114671.

[9] Micorsoft. ‘Microsoft regulatory compliance.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/defender-for-cloud/regulatory-compliance-dashboard. (accessed: 07.03.2022).

[10] Microsoft. ‘Overview of the azure security benchmark (v3).’ (), [Online]. Available: https://docs.microsoft.com/en-us/security/benchmark/azure/overview. (accessed: 18.05.2022).

[11] M. Wali, *Learn Microsoft Azure : Build, Manage, and Scale Cloud Applications Using the Azure Ecosystem*, 1st edition. Packt Publishing, 2018, pp. 5–7, ISBN: 9781789617580.

[12] Microsoft. ‘What are public, private and hybrid clouds?’ (), [Online]. Available: https://azure.microsoft.com/en-in/overview/what-are-private-public-hybrid-clouds/#overview. (accessed: 10.03.2022).

[13] Microsoft. ‘Organize your azure resources effectively.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-setup-guide/organize-resources. (accessed: 09.05.2022).

[14] Microsoft. ‘What is azure active directory?’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-whatis. (accessed: 09.05.2022).

[15] Microsoft. ‘Azure key vault basic concepts.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/key-vault/general/basic-concepts. (accessed: 09.05.2022).

[16] Microsoft. ‘Create a dashboard in the azure portal.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-portal/azure-portal-dashboards. (accessed: 09.05.2022).

[17] Microsoft. ‘Azure monitor overview.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-monitor/overview. (accessed: 09.05.2022).

[18] Microsoft. ‘What is cost management + billing?’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/cost-management-billing/cost-management-billing-overview. (accessed: 09.05.2022).

[19] Microsoft. ‘Overview of log analytics in azure monitor.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-monitor/logs/log-analytics-overview. (accessed: 09.05.2022).

[20] Microsoft. ‘What is microsoft defender for cloud?’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/defender-for-cloud/defender-for-cloud-introduction. (accessed: 09.05.2022).

[21] Microsoft. ‘What is microsoft sentinel?’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/sentinel/overview. (accessed: 09.05.2022).

[22] Microsoft. ‘Application insights overview.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-monitor/app/app-insights-overview. (accessed: 09.05.2022).

[23] Microsoft. ‘Storage account overview.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/storage/common/storage-account-overview. (accessed: 09.05.2022).

[24] Microsoft. ‘Welcome to azure cosmos db.’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/cosmos-db/introduction. (accessed: 09.05.2022).

[25] Microsoft. ‘What is azure sql?’ (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-sql/azure-sql-iaas-vs-paas-what-is-overview?view=azuresql. (accessed: 09.05.2022).

[26] Microsoft. 'About azure cache for redis.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-cache-for-redis/cache-overview. (accessed: 09.05.2022).

[27] Microsoft. 'What is a content delivery network on azure?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/cdn/cdn-overview. (accessed: 09.05.2022).

[28] Microsoft. 'What is azure application gateway?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/application-gateway/overview. (accessed: 09.05.2022).

[29] Microsoft. 'What is azure web application firewall on azure application gateway?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/web-application-firewall/ag/ag-overview. (accessed: 09.05.2022).

[30] Microsoft. 'Network security groups.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/virtual-network/network-security-groups-overview. (accessed: 09.05.2022).

[31] Microsoft. 'App service overview.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/app-service/overview. (accessed: 09.05.2022).

[32] Microsoft. 'Introduction to azure functions.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview. (accessed: 09.05.2022).

[33] Microsoft. 'Https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-overview.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-overview. (accessed: 09.05.2022).

[34] Microsoft. 'Windows virtual machines in azure.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/virtual-machines/windows/overview. (accessed: 09.05.2022).

[35] Microsoft. 'What is azure virtual desktop?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/virtual-desktop/overview. (accessed: 09.05.2022).

[36] Microsoft. 'Iot concepts and azure iot hub.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/iot-hub/iot-concepts-and-iot-hub. (accessed: 09.05.2022).

[37] Microsoft. 'What is azure iot central?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/iot-central/core/overview-iot-central. (accessed: 09.05.2022).

[38] Microsoft. 'Azure event hubs — a big data streaming platform and event ingestion service.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/event-hubs/event-hubs-about. (accessed: 09.05.2022).

[39] Microsoft. ‘Welcome to azure stream analytics.’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-introduction`. (accessed: 09.05.2022).

[40] Microsoft. ‘What is azure digital twins?’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/digital-twins/overview`. (accessed: 09.05.2022).

[41] Microsoft. ‘What is azure hdinsight?’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/hdinsight/hdinsight-overview`. (accessed: 09.05.2022).

[42] Microsoft. ‘What is azure data explorer?’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/data-explorer/data-explorer-overview`. (accessed: 09.05.2022).

[43] Microsoft. ‘Introduction to azure data lake storage gen2.’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/storage/blobs/data-lake-storage-introduction`. (accessed: 09.05.2022).

[44] Microsoft. ‘What is azure machine learning?’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/machine-learning/overview-what-is-azure-machine-learning`. (accessed: 09.05.2022).

[45] Microsoft. ‘What is azure databricks?’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/databricks/scenarios/what-is-azure-databricks`. (accessed: 09.05.2022).

[46] Terraform. ‘Providers.’ (), [Online]. Available: `https://www.terraform.io/language/providers`. (accessed: 10.03.2022).

[47] Terraform. ‘What is terraform?’ (), [Online]. Available: `https://www.terraform.io/intro`. (accessed: 10.03.2022).

[48] Terraform. ‘Modules.’ (), [Online]. Available: `https://www.terraform.io/language/modules`. (accessed: 27.04.2022).

[49] Terraform. ‘Module sources.’ (), [Online]. Available: `https://www.terraform.io/language/modules/sources`. (accessed: 27.04.2022).

[50] several authors. ‘Store terraform state in azure storage.’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/developer/terraform/store-state-in-azure-storage?tabs=terraform`. (accessed: 14.03.2022).

[51] Terraform. ‘Azurerm.’ (), [Online]. Available: `https://www.terraform.io/language/settings/backends/azurerm`. (accessed: 12.04.2022).

[52] several authors. ‘Azure storage encryption for data at rest.’ (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/storage/common/storage-service-encryption`. (accessed: 14.03.2022).

[53] Terraform. ‘Command: Init.’ (), [Online]. Available: `https://www.terraform.io/cli/commands/init`. (accessed: 29.04.2022).

[54] Terraform. 'Command: Plan.' (), [Online]. Available: https://www.terraform.io/cli/commands/plan. (accessed: 29.04.2022).

[55] Git. 'Git.' (), [Online]. Available: https://git-scm.com/. (accessed: 10.03.2022).

[56] D. Spinellis, 'Git,' *IEEE Software*, vol. 29, no. 3, pp. 100–101, 2012. DOI: 10.1109/MS.2012.61.

[57] several authors. 'Understanding github actions.' (), [Online]. Available: https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions. (accessed: 10.03.2022).

[58] Microsoft. 'Choose the right azure command-line tool.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/developer/azure-cli/choose-the-right-azure-command-line-tool. (accessed: 13.04.2022).

[59] several authors. 'Create and manage action groups in the azure portal.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/azure-monitor/alerts/action-groups. (accessed: 08.04.2022).

[60] Microsoft. 'What is azure hdinsight?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/hdinsight/hdinsight-overview. (accessed: 28.04.2022).

[61] Microsoft. 'Welcome to azure stream analytics.' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-introduction. (accessed: 29.04.2022).

[62] Microsoft. 'What is azure data explorer?' (), [Online]. Available: https://docs.microsoft.com/en-us/azure/data-explorer/data-explorer-overview. (accessed: 29.04.2022).

[63] E. Commission. 'Art. 25 gdpr - data protection by design and by default.' (), [Online]. Available: https://gdpr-info.eu/art-25-gdpr/. (accessed: 13.04.2022).

[64] Arbeidstilsynet, 'Act relating to working environment, working hours and employment protection, etc. (working environment act),' pp. 14–15. [Online]. Available: https://www.arbeidstilsynet.no/globalassets/regelverkspdfer/working-environment-act, (accessed: 19.05.2022).

[65] Datatilsynet. 'Gps og sporing av yrkesbiler.' (), [Online]. Available: https://www.datatilsynet.no/personvern-pa-ulike-omrader/personvern-pa-arbeidsplassen/overvaking-kjoretoy/. (accessed: 19.05.2022).

[66] Bridgecrew. 'What is checkov?' (), [Online]. Available: https://www.checkov.io/1.Welcome/What%20is%20Checkov.html. (accessed: 18.05.2022).

[67] Microsoft. 'Encrypt sensitive information in transit.' (), [Online]. Available: https://docs.microsoft.com/en-us/security/benchmark/azure/baselines/iot-hub-security-baseline#dp-4-encrypt-sensitive-information-in-transit. (accessed: 13.04.2022).

[68] Microsoft. 'Azure data encryption at rest.' (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/security/fundamentals/encryption-atrest#encryption-at-rest-in-microsoft-cloud-services`. (accessed: 13.04.2022).

[69] arbeidsgruppen. 'Kartlegging av hindringer i regelverk for bruk av skytjenester.' (), [Online]. Available: `https://www.regjeringen.no/contentassets/d48b5b5895e54d679fef76e0860140a8/skytjenester_arbeidsgrupperapport.pdf`. (accessed: 07.04.2022).

[70] A. B. Ketil Lund Frank Trethan Johnsen. 'Bruk av skytjenester i forsvaret – muligheter og utfordringer.' (), [Online]. Available: `https://publications.ffi.no/nb/item/asset/dspace:6936/20-00136.pdf`. (accessed: 07.04.2022).

[71] Datatilsynet. 'Skytjenester.' (), [Online]. Available: `https://www.datatilsynet.no/personvern-pa-ulike-omrader/internett-og-apper/skytjenester/`. (accessed: 07.04.2022).

[72] Microsoft. 'Microsoft identity manager documentation.' (), [Online]. Available: `https://docs.microsoft.com/en-us/microsoft-identity-manager/`. (accessed: 06.05.2022).

[73] Microsoft. 'Security recommendations - a reference guide.' (), [Online]. Available: `https://docs.microsoft.com/en-us/azure/defender-for-cloud/recommendations-reference`. (accessed: 25.05.2022).

Appendix A

Project Plan

The Project plan is the groups plan for the Bachelor project. The plan consist of a introduction to the Project and the set limits for the work. Organization and planning is other parts of the project that is described in the Project plan.

Project Plan

Bachelor thesis DCSG2900 - 2022

Morten Bratberg

Anders R. Loeng

Eirik Granvin Bakke

Table of content

1. Introduction and Objective

1.1 Background

Bouvet is a Norwegian IT consultancy, with offices in Norway and Sweden, which delivers services to both private and public sector. Their services ranges from designing, developing and advising on IT solutions and digital communication. Bouvet's ambition is to "be the most credible consultancy with the most satisfied employees and clients" [1].

Building secure and compliant infrastructures in the cloud is a crucial ability for Bouvet to reach their goals. This project will help Bouvet reach their goals by developing infrastructure as code with security and compliance as the focal point.

1.2 Project Objectives

1.2.1 Learning Outcome

The goal of this project is to take a deeper dive into the security of infrastructure architectures in a cloud environment, like Microsoft Azure. With this project we will get real life experience with working on a bigger project with a client. This opportunity allows us to get some practise on how to plan, manage and work on a bigger project. Hopefully this will give us some new insides into how things work, and at the end, give us the opportunity to reflect on the project to see what could have been done different/better.

1.2.2 Impact

Bouvet describes themselves as a company that wants to "lead the way and build tomorrows society" [1]. And a part of that is to help small/medium companies to utilize and set up cloud environments. Developing secure generalized templets for Bouvet, will make that task easier, and the solution more secure for the companies.

1.2.3 Result

The goal is to make one generalized template, which will be altered to fit the security needs for 3 different sectors. These templates will reduce the time Bouvet employees uses to create a new cloud environment for a customer. The templates will also increase the security of the infrastructures.

1.3 Project limitations

After thorough discussions with the employer, the limitation for this project is:

- Deadline for the project is 20. May 2022
- The templates need to be compatible with Azure
- One basic template, with a focus on security through logs, firewalls and segmentation.
- Customized templates for different sectors based on the basic template

1.4 Resource Needs

In addition to our own equipment and resources, Bouvet will give the group access to Azure as an environment to test and develop the templates.

2. Scope

2.1 Field of study

Cloud services is a core functionality for most small and medium enterprises. The cloud infrastructure that supports the enterprises demands must be provisioned and managed. Infrastructure as code (IaC) enables providers to build new environments more efficient and consistent by automation of these processes. IaC also makes it possible to track changes, roll back and recover the infrastructure.

2.2 Delimitations

Decided limitation and framework for the project by the group after thorough discussions.

- The resulting infrastructure code will be templates, not a “ready-to-go" infrastructure stack. The code must be modified based on the needs of the customer or project
- We will prioritize security and compliance over comprehensive scope. We don’t plan to cover all possible needs and variants of a cloud environment

This limitations to the project work will potentially improve the finale product because it gives the group more time to focus on the most important aspects of the task.

2.3 Project Description

The project consists of 3 main components:

1.Develop IaC templates

A set of templates of infrastructure as code for building environments in Azure. This will result in one basic template that can be easily edited to fit different requirements, and three templates that are specifically designed for small and medium-sized businesses (SMB) in the different sectors. The three templates will each provision an infrastructure in Azure with the required resources, security measures and compliance. Variables should be used in the templates to ease to job of alter the templates for different users.

2.Describe different sectors requirements

The different sectors requirements to resources, security and compliance will be described. A typical SMBs requirements for resources will be the basis for the content of the infrastructure. Relevant standards for cloud security and compliance will be the basis for security and compliance for the infrastructure.

3.Security in use of templates

The security in the code is one aspect. We will look at the total security of the templates, and find policies, tools or methods of use, that makes the templates usability and security optimal.

3. Project Organizing

3.1 Roles and Responsibility

Employer: Bouvet AS

Team leader: Eirik Granvin Bakke

Group members: Morten Bratberg, Anders R. Loeng

Supervisor: Eigil Obrestad, NTNU

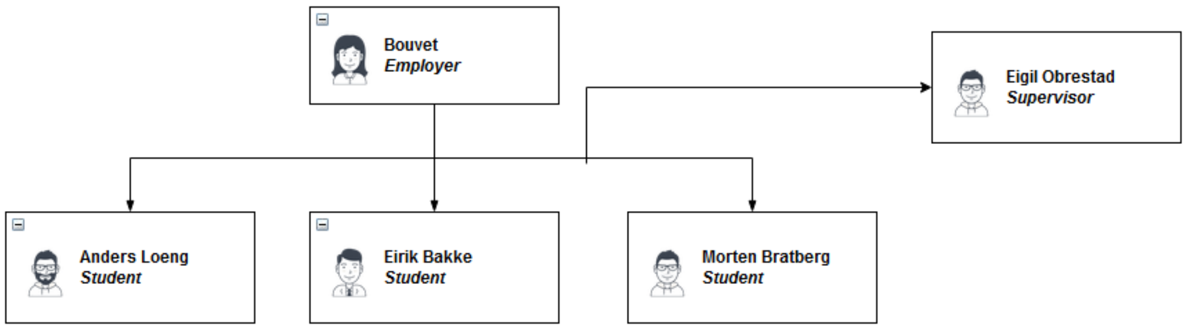


Figure 1: Organizational chart for the project

3.2 Routines, Rules and Requirements

- It is expected that the group members meet up at the designated time slot, if a member is late, this does not constitute a sanction. If the other group members start to see a pattern from a group member, this will be first discussed between members. If this then is repeated, or the problem isn't resolved, group supervisor will then be contacted, and other consequences might be implemented.
- It is expected that every member registered a minimum of 30 work hours each week.
- Exemptions to the 30 hours weeks needs to be informed the other group members a minimum of 1 week in advanced. Then the member has 2 weeks to catch up on lost time.
- It is expected that every member attends all the meetings with the supervisor and employer, unless otherwise is agreed upon. We will have weekly meetings with the supervisor from 13:00 to 13:30 on Fridays.
- There should be notes taken from every meeting with the supervisor and the employer.

- Project costs that are not covered by NTNU or the Employer will the group members share.
- All work by any member should be documented, with date and time and what was done.
- If a member cannot complete their task, they should notify the rest of the group in a reasonable time in advance of the deadline, so a decision could be taken on how to solve the problem.

4. Planning, Follow up and Reporting

4.1 Choosing Development Model

The conditions for choosing development model are the possibility to do research during the project, having a clear overview of what to do when and being able to make changes along the way. Based on these conditions, scrum will be used as development model for this project.

4.1.1 How are we going to use the model?

The project is going to be carried out in 2-week long sprints, where we use a Scrum board with a backlog, which we then decide what is going to be done. Each day we have 2 meetings, a meeting at the beginning, talking about what we are going to work on that day. One at the end, talking about what we have done, and possible complications. During the sprints we will have at least 1 meeting with the client, and 2 meetings with the supervisor, if deemed necessary. During the sprints we can add tasks, if there is something new that comes up. At the end of a sprint, we plan the new sprint, and possibly extend the periods of some tasks if they take longer time than expected. We will not be using the typical roles from Scrum, like project owner, scrum master and develop team member, since the team size is quite manageable.

4.1.2 Schedule for the sprints

Sprint nr	Start date	Finished date
1	10. January 2022	31. January 2022
2	31. January 2022	13. February 2022
3	14. February 2022	27. February 2022
4	28. February 2022	13. mars 2022
5	14. Mars 2022	27. Mars 2022
6	28. mars 2022	10. April 2022
7	11. April 2022	24. April 2022
8	25. April 2022	8. May 2022
9	9. May 2022	20. May 2022

Table 1: Sprint schedules

4.2 Plan for Status Meetings

Weekly meetings are scheduled with the supervisor. The meeting will be used to give a short status update, discuss experiences and information, and receive tips and feedback.

It is scheduled biweekly meetings with Bouvet, to update the employer on the progress and receive/gather necessary information.

There will be Scrum meetings at the start and end of each day, to update the task the members have completed and what task they are going to work on. The meetings at the end of the day will be used to inform what tasks have been completed and delegate upcoming work.

A planning meeting will take place at the start of each sprint, to decide the task for the sprint. At the end of the sprint there will be a meeting to review the sprint. It will be used to check if the goals and tasks for the sprint was completed and discuss potential areas of improvement.

5. Quality Assurance

5.1 Documentation, Standards and Configuration Management

5.1.1 Documentation

The code will have comprehensive documentation on what the code does, how it works and how to use it. To document and visualize the infrastructure, we will create infrastructure charts of the environments that the IaC templates provisions.

References used when gathering information during research will be documented.

All meetings with the employer and the supervisor will be documented with what we discussed and determined. All work by the group members will be documented in the shared timesheet, with the time spent and a high-level description of the work. All documentations will have designated places in SharePoint, so that it will be easily found when needed.

5.1.2 Standards

International, regional and national standards regarding security aspects relevant to the project e.g., ISO, Enisa and NSM standards will be followed. The sectors or industries chosen to develop IaC templates for, might have standards and regulations that the infrastructure must comply with.

5.2 Tools

5.2.1 SharePoint

The group will use SharePoint to share documents and research material. This will also be the main platform for storing documentation.

5.2.2 Version Control System

GIT will be used as version control system for our code, documentation and the report. Azure DevOps Services, GitHub and GitLab are candidates for VCS.

5.2.3 Overleaf

Overleaf is an online LaTeX editor that is used to create technical documents. The group is going to use Overleaf to write the thesis. Overleaf and Latex gives the group an easier way to structure their document, because it uses a syntax-based language to structure the document.

5.2.4 Trello

Trello is an online collaboration tool that lets user's organic tasks on a board. The Trello board is a simple but useful tool that can be used as an aid in Scrum and Kanban workflow and will be used as a digital representation of the Scrum board.

5.2.5 Communication Platforms

Discord and teams will be the two main platforms used for communication during the project. Discord will be mostly used by the group members when working together or having team meetings. Teams will be used for meetings with the supervisor and Bouvet.

5.3 Risk Management

In this part we have looked at the different types of risk that we have deemed as possible problems, for the progression/completion of the project. Below you can find the tables explaining how the risk is calculated, the meaning of the values, the risk scenario with a measure and then the residual risk after implementing the measures.

5.3.1 Risk Matrix

		Probability			
I		1 unlikely	2 less likely	3 Probable	4 Very likely
m	1 Insignificant	1	2	3	4
p	2 Low	2	4	6	8
a	3 Moderate	3	6	9	12
ct	4 Sever	4	8	12	16

Table 2: Probability multiplied by impact gives us a risk value

5.3.2 Risk value

Severity rating - Colour	Meaning
12-16 – Severe	Things that seriously compromise the progression of the project, or potentially lead to complete stop of the progression. Thorough measures are necessary
8-9 – Moderate	Things that can hinder the progress of the project, or cause delay. Measures are necessary
4-6 - Low	Things that can influence the progress of the progress of the project. Simple measures should be implemented
1-4 - insignificant	Should not impact the progress of the project in any significant way. The risk is deemed acceptable, thus no measures needed

Table 3: Risk Value paring with the risk rating and the meaning of the rating

5.3.3 Inherent risk

Risk	Impact	Probability	IxP=R	Controls
Member of the group becomes ill	Moderate	Less likely	3x2=6	Good communication between members, so that the workload of the ill member gets reassigned. Make sure to follow regulations regarding Covid.
Multiple members of the group become ill	Severe	Unlikely	4x1=4	Make sure to follow regulations regarding Covid.
Internal disagreement(s) that causes the progress to halt	Moderate	Unlikely	3x1=3	The group have decided several rules that we must follow. If We cannot come to an agreement, we will escalate the issue to our supervisor.
We do something that leads to sensitive / secret information being leaked	Severe	Unlikely	4x1=4	Make sure to use secure storage systems, with a secure password and 2FA. Don't download sensitive

				information members computers. Don’t share sensitive information with unsafe communication options.
Misunderstandings of the requirements or goals for the product	Moderate	Less likely	3x2=6	Make sure to plan for enough time to agree and understand the requirements and goals. Express all our questions to Bouvet.
We are over ambitious with the task at hand and are not able to finish the bachelor assignment	Severe	Unlikely	4x1=4	Put early focus into planning the span of the project and have realistic goals. Have good communication with the client and clarify what their minimum requirements are. If the members at some point suspect that the workload is too much, discuss the potential issue with the client.
Bouvet change cloud platform they want the templates for	Moderate	Unlikely	3x1=3	If it is late in the project, we will continue the work as it was originally planned. If the changes are made early, we will implement it and write templates for that cloud platform
We lose authentication information for cloud environment, which leads to unwanted access to an environment	Moderate	Unlikely	3x1=3	The group member will not store authentication information on VCS, in other cloud storages, or in clear text anywhere. We will use a password manager to store the authentication information for the cloud environment.
Unable to access Azure (1 day or more)	Low	Unlikely	2x1=2	We postpone testing the code and start working on other parts of the code or write on the thesis
All data in Overleaf is deleted (If we are going to use it)	Severe	Unlikely	4x1=4	Weekly backups of the data in overleaf to our VCS or more frequent if big changes are made We use Word and when overleaf is up aging we paste it in to overleaf
Unable to access the VCS we are using (1 day or more)	Low	Unlikely	2x1=2	We share the changes through another channel, like another VCS or SharePoint. And have

				a cloned repo on a member's computer.
VCS repo is deleted	Moderate	Unlikely	3x1=3	Every member of the group has a copy of the VCS repo that is no more than 72 hours old

Table 4: Potential risk scenarios, calculated risk before controls and possible measures

5.3.4 Residual Risk

Risks	Inherent Risk	Residual Risk
Member of the group becomes ill	Low 6	Low 4
Multiple members of the group become ill	Low 4	Low 4
Internal disagreement(s) that causes the progress to halt	Insignificant 3	Insignificant 2
We do something that leads to sensitive / secret information being leaked	Low 4	Low 4
Misunderstandings of the requirements or goals for the product	Low 6	Insignificant 3
We are over ambitious with the task at hand and are not able to finish the bachelor assignment	Low 4	Low 4
Bouvet change cloud platform they want the templates for	Insignificant 3	Insignificant 2
We lose authentication information for cloud environment, which leads to unwanted access to an environment	Insignificant 3	Insignificant 3
Unable to access Azure (1 day or more)	Insignificant 2	Insignificant 2
All data in Overleaf is deleted (If we are going to use it)	Low 4	Insignificant 2
Unable to access the VCS we are using (1 day or more)	Insignificant 2	Insignificant 1
VCS repo is deleted	Insignificant 3	Insignificant 2

Table 5: Residual risk for the scenarios after the implementation of measures

6. Work Schedule

This Gannt schema is based on theVertex42 template [3].

Bachelor

Secure and compliant Infrastructure as Code

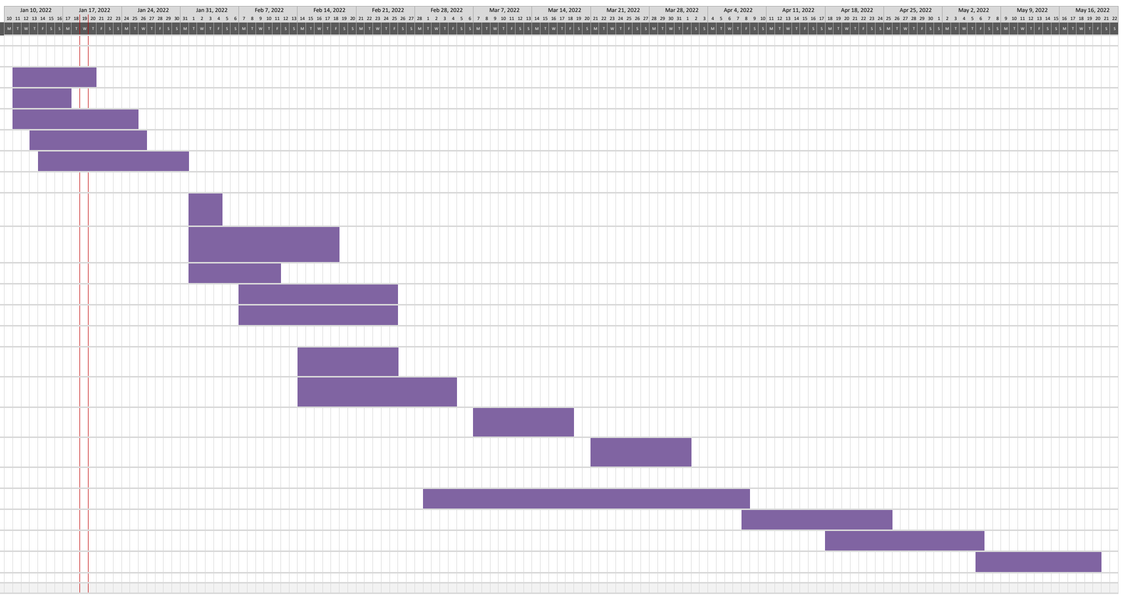
Anders R. Loeng
Morten Bratberg
Eirik Granvin Bakke

SIMPLE GANTT CHART by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Project Start:	11-Jan-22
Display Week:	1-Jan

TASK	ASSIGNED TO	PROGRESS	START	END
Pilot project				
Contract between the members of the group			11-Jan-22	20-Jan-22
Choose process modell			11-Jan-22	17-Jan-22
Make GANTT			11-Jan-22	25-Jan-22
Contract between the group and Bouvet			13-Jan-22	26-Jan-22
Write the pilot project			14-Jan-22	31-Jan-22
Reasearch phase				
Decide what sectors to make template for and what IaC tools to use			1-Feb-22	4-Feb-22
Reasearch different sectors requirments to the cloud infrastructure			1-Feb-22	18-Feb-22
Reasearch IaC tools			1-Feb-22	11-Feb-22
Reasearch IaC security tools			7-Feb-22	25-Feb-22
Reasearch Security in the cloud (Azure)			7-Feb-22	25-Feb-22
Making the product				
Make a basic IaC template			14-Feb-22	25-Feb-22
Make the IaC code/documentation for sector 1			14-Feb-22	4-Mar-22
Make the IaC code/documentation for sector 2			7-Mar-22	18-Mar-22
Make the IaC code/documentation for sector 3			21-Mar-22	1-Apr-22
Writing the Bachelor Thesis				
First draft of the Bachelor thesis			1-Mar-22	8-Apr-22
Second draft of the Bacehlor thesis			8-Apr-22	25-Apr-22
Finale draft of the Bachelor thesis			18-Apr-22	6-May-22
Deliver the final Bachelor thesis			6-May-22	20-May-22

Insert new rows ABOVE this one



7. References

[1] Bouvet. About Bouvet [Internet]. Norge: Bouvet; No date [cited: 17. January 2022]. Available from: https://en.bouvet.no/

[2] Palmquist S, Lapham M, Miller S, Chick T, Ozkaya I. Parallel Worlds: Agile and Waterfall Differences and Similarities [Internet]. Pittsburgh, Pennsylvania: Carnegie Mellon University; 2013 [cited: 13. January 2022] Available from: https://apps.dtic.mil/sti/pdfs/ADA610501.pdf

[3] Vertex42, Simple Gantt Chart [Internet]. Simple Gantt Chart [cited: 11. January 2022] https://templates.office.com/en-us/simple-gantt-chart-tm16400962

Appendix B

Assignment from Employer

This appendix is the project assignment from the employer.

Secure and compliant Infrastructure as Code

Oppgaven

Infrastructure as Code (IaC) er stadig hyppigere brukt i sammenheng med oppsett av skytjenester. Behovet for gode grunnmaler som kan endres ved nødvendighet blir mer og mer aktuelt. Det ferdige skymiljøet som blir resultatet av disse grunnmalene må være sikre, skalerbare, sporbare og kvalitetssikret. I tillegg, ulike bransjer har ulike behov når det kommer til sikkerhet i skymiljøene sine. Helsesektoren og banknæringen kan ha sammenfallene krav fra myndigheter, men også helt ulike. Dette bør gjenspeiles i en god grunnmal. Det er gunstig med en mal for å kjapt kunne spinne opp et skymiljø for en virksomhet.

Malene kan lages i Azure, ARM templates, med PowerShell eller ved bruk av rammeverk som Azure Bicep. Skytjenesten som er blant de mest ettertraktete i Norge er Microsoft Azure, og vi i Bouvet ønsker derfor at grunnmalene lages for Azure, men det er også en mulighet å bruke Terraform.

Oppgavens mål

Målet med oppgaven er å opprette en eller flere maler for utrulling av infrastruktur som vi kan bruke som en grunnlinje for nåværende eller fremtidige kunder. Dette er for å forenkle prosessen med å bistå en kunde til å enten migrere til skyen, forbedre skymiljøet sitt eller å starte opp et nytt. Samtidig er det viktig for kundene å etterleve krav og sikkerhetskrav de måtte ha til sine tjenester. Derfor bør etterlevelse av krav reflekteres i malene. Vi ser for oss at det lages maler for tre sektorer med ulike krav. For eksempel: finans, helse, industri (olje, off-shore, kraft, stålverk), staten, forsvaret

Vi er ute etter:

- Lage en basisgrunnmal som kan brukes til å generere en "landing zone" i sky(helst azure) som et fundament for alle bransjer
- Tilpasse basisgrunnmal for ulike bransjer iht. bransjens krav og sikkerhetskrav
- God og oversiktlig dokumentasjon på hvordan grunnmalene funker, og hvordan de skal brukes

Kontakt:

sebastian.slettebakken@bouvet.no lauritz.somme@bouvet.no magnus.korneliussen@bouvet.no

Appendix C

Time sheet

The time sheet was the groups way of logging the work. The time sheet was made in excel. The total excel document was divide into 5 worksheet. One worksheet for every group member, and one for the group work and one summary that calculated the amount of work done by each member. The group and personal worksheet was structured the same. The time spent and the date was noted as well as a short description of the work done. The group worksheet also provided the option to note who worked together. The summary is structured by weeks and how much each persons has worked alone, together and in total, and a summary for the whole group each week. The reason for the fluctuation in time spent between the weeks are because of seminars and group work in another subject. The Appendix only present the summary worksheet, because it gives a view in to the execution of the project.

	Indivduelt arbeid			Sammarbied			Totalt			Totalt	Hva ble gjort
uke nr	Anders	Morten	Eirik	Anders	Morten	Eirik	Anders	Morten	Eirik	Gruppa	
2	05:30	03:30	07:15	16:30	16:30	16:30	22:00	20:00	23:45	65:45	Sprint 1 - Sign Contract w/ Bovuet - GANTT GANTT - Decide development model - Project Plan - Learn overleaf/LaTex - research Azure - research Terraform
3	10:00	12:00	23:30	15:30	17:15	17:15	25:30	29:15	40:45	95:30	
4	15:00	07:00	19:00	17:00	17:00	17:00	32:00	24:00	36:00	92:00	
5	12:30	13:00	15:00	20:00	20:00	20:00	32:30	33:00	35:00	100:30	Sprint 2 - Decide IaC tool - research & decide pipeline - research Azure - research Terraform
6	13:30	07:00	16:00	23:00	23:00	23:00	36:30	30:00	39:00	105:30	
7	06:00	09:00	07:00	09:45	09:45	09:45	15:45	18:45	16:45	51:15	Sprint 3 (Base template) - module backend - module storage - module key vault - module web page - create visio diagram for base infrastructure
8	14:00	14:00	15:30	22:00	22:00	22:00	36:00	36:00	37:30	109:30	
9	10:00	09:00	09:45	10:00	10:00	10:00	20:00	19:00	19:45	58:45	Sprint 4 - module Security/Defender for Cloud - module monitoring - module budget - module web page - Bachelor struktur m.m.
10	14:30	14:15	15:00	19:15	19:15	19:15	33:45	33:30	34:15	101:30	
11	11:30	11:30	10:00	07:00	07:00	07:00	18:30	18:30	17:00	54:00	Sprint 6 - decide sectors/cases for case template - case IoT - case Web Page - Bachelor skriving
12	03:45		04:00	31:15	06:45	31:15	35:00	06:45	35:15	77:00	
13	07:00	18:00	07:15	26:15	21:15	30:00	33:15	39:15	37:15	109:45	Sprint 7 - base tmp module desktop virtualization - case IoT - case Web Page - Bachelor skriving
14	16:30	21:45	18:00	22:00	22:00	22:00	38:30	43:45	40:00	122:15	
15	08:00	03:30	08:00	12:30	12:30	12:30	20:30	16:00	20:30	57:00	Sprint 7 - base tmp module desktop virtualization - case IoT - case Web Page - Bachelor skriving
16	07:00	14:30	05:00	17:30	09:45	17:30	24:30	24:15	22:30	71:15	
17	12:00	09:00	15:30	22:00	25:00	25:00	34:00	34:00	40:30	108:30	Sprint 7 - code review - Bachelor skriving
18	07:30	03:30	07:30	22:00	22:00	15:30	29:30	25:30	23:00	78:00	
19	04:00	06:00	04:00	26:00	26:00	10:30	30:00	32:00	14:30	76:30	Sprint 8 - code review & touch up - Bachelor skriving
20	10:30	08:00	07:00	40:00	40:00	40:00	50:30	48:00	47:00	145:30	
21	17:00	04:00	10:00	19:15	35:15	35:15	36:15	39:15	45:15	120:45	
Totalt	205:45	188:30	224:15	398:45	382:15	401:15	604:30	570:45	625:30	1800:45	

Appendix D

IoT missing compliance

The spreadsheet show the current compliance problems with the standards Azure Security Benchmark v3 and ISO 27001:2013, for the IoT case template. The spreadsheet lists the relevant sections of the standards, and shows which is compliant, what resource is affected and the reason for not being compliant. For further information about the problems causing the compliance issues, see Appendix E.

		Total Amount of policies	Compliante Policies	Missing compliance	Complian ce	Resoruce Type	Reason for no compliance
NS. Network Secuirty	NS-1	5	4	Policy 1	Partial	VM, Subnets	Non internett facing vm should be protected
	NS-2	29	23	Policy 1,2,3,4,5,6	True	Azure Resoruces, Storage Account, Key Vault	Networking, endpoint, private, firewall
	NS-3	4	4		True	VM, Azure Resoruce	
	NS-5	1	1		True	Virtual Network	
	NS-6	2	2		True	Azure Resources	
	NS-7	1	1		True	VM	
	NS-8	3	3		True	Azure Resoruces, Web App	
	NS-10	1	1		True	Subscriptions	
IM. Identity Managment	IM-1	2	2		True	Azure Resources	
	IM-3	4	2	Policy 1,2	False	Web App, Azure Resoruce	Maneged identity should be used
	IM-6	5	5		True	Subscriptions, VM	
PA. Privileged Access	PA-1	4	2	Policy 1, 2	False	Subcriptions	More than 3 owners, External accounts
	PA-2	1	1		True	VM	
	PA-4	5	4	Policy 1	Partial	Subscriptions	External Accounts with owner
	PA-7	2	2		True	Azure Resource, Azure Role Definition	
DP. Data Protection	DP-2	5	4	Policy 1	Partial	Azure Resoruce, Subscription	Defender should be active for SQL Server
	DP-3	15	13	Policy 1,2	Partial	Azure Resources, Web App, Storage Account	FTPS should be required
	DP-4	4	3	Policy 1	False	Azure Resources, VM	VM's Should encrypt temp disk/cache
	DP-5	9	9		True	Azure Resources	
	DP-6	2	2		True	Azure Resources	
	DP-7	1	1		True	Azure Resource	
	DP-8	6	3	Policy 1,2,3	False	Key Vaults, Subscriptions	Firewall, purge protection, private endpoint
AM. Asset Management	AM-2	2	2		True	Storage Account, VM	
	AM-5	2	2		True	VM	
LT. Logging and Threat Detection	LT-1	13	12	Policy 1	Partial	Subscriptions, Azure Resources	Defender should be active for SQL Server
	LT-2	13	12	Policy 1	Partial	Subscriptions, Azure Resources	Defender should be active for SQL Server
	LT-3	13	9	Policy 1,2,3,4	False	Event hub namespace, Stream analytics, iot	Diagnostic logs should be enabled
	LT-5	5	5		True	Azure Resource, Subscriptions, VM	
	LT-6	1	1		True	Azure Resource	
IR. Incident Response	IR-2	3	3		True	Subscriptions	
	IR-3	11	10	Policy 1	Partial	Subscriptions, Azure Resources	Defender should be active for SQL Server
	IR-4	1	1		True	Virtual Network	
	IR-5	11	10	Policy 1	Partial	Subscriptions, Azure Resources	Defender should be active for SQL Server
PV. Posture and Vulnerability Management	PV-2	27	25	Policy 1,2	Partial	Azure Resources, Web App	SSl Certificates, and client certificates
	PV-4	10	9	Policy 1	Partial	Azure Resources, VM	Guest Configuration should be installed on VM
	PV-5	3	3		True	Azure Resources, VM	
	PV-6	16	16		True	Azure Resources, VM, Container Host	
ES. Endpoint Security	ES-1	1	1		True	Subscriptions	
	ES-2	5	5		True	Azure Resources, VM	
	ES-3	1	1		True	VM	
BR. Backup and	BR-1	4	3	Policy 1	False	Azure Resources, VM	Azure backup should be enabled for VM

Recovery	BR-2	4	3	policy 1	False	Azure Resoruces, VM	Azure backup should be enabled for VM
DS. DevOps Security	DS-6	2	2		True	Azure Resoruce, Container Host	

		Total Amount of policies	Compliante Policies	Missing compliance	Compliance	Resoruce Type	Reason for no compliance
A.6. Organization of Information Secuity	A.6.1.2.	2	1	Policy 1	False	Subscription	Amount of owners
A.9. Access Control	A.9.1.2.	8	5	Policy 1,2,3	False	VM, Storage Account	Auditing, Guest configuration
	A.9.2.3.	7	5	Policy 1, 2	False	Subscription, Azure Resoruces, Role Definitions	External Accounts, Usage of Auditing
	A.9.2.4.	7	5	Policy 1,2	False	Subscriotion, VM	Guest Config, Auditing
	A.9.2.5.	4	3	Policy 1	False	External account with owner	External Accounts
	A.9.2.6.	2	2		True	Subscription	
	A.9.4.2.	3	3		True	Subscription	
	A.9.4.3.	8	8		True	Azure Resoruces, VM	
A.10. Cryptography	A.10.1.1.	12	11	Policy 1	Partial	Azure Resoruces, VM, Web App, Storage Account	encrypt disk/cache
A.12. Operations Security	A.12.4.1.	6	5	Policy 1	False	Azure Resoruces	Auditing Diagnostic setting
	A.12.4.2.	6	5	Policy 1	False	Azure Resoruces	Auditing Diagnostic setting
	A.12.4.3.	6	5	Policy 1	False	Azure Resoruces	Auditing Diagnostic setting
	A.12.5.1.	1	1		True	VM	
	A.12.6.1.	4	4		True	VM	
	A.12.6.2.	1	1		True	VM	
A.13. Commications Security	A.13.1.1.	2	1	Policy 1	False	VM, Storage Account	Restrict access with firewall
	A.13.2.1.	2	2		True	Azure Resoruce, Storage Account	

Appendix E

IoT compliance/security problems

The spreadsheet shows the current security and compliance issues for the IoT case template. The spreadsheet lists the problem, severity, a short description, affected resources, standard and the option for resolving the issues. The problems which contains fix, enforce, logic app or deny are currently not automated, and is easily fixed in GUI. The manual indicates that the problems would need a integration of script for automation, or is a more complex issue to resolve in GUI. For the project most of these problems was fixed with GUI, due to the amount of time it would take to automate. NOTE: The text for description in the spreadsheet, is auto generated text from the compliance manager in Microsoft defender for cloud [73].

Severity	Name	Description	Status	Initiatives	Resoruce	Options
Low	Diagnostic logs in Azure Stream Analytics should be enabled	Enable logs and retain them for up to a year. This enables you to recreate activity trails for investigation purposes when a security incident occurs or your network is compromised.	Unhealthy	ASC Default	Stream Analytics	Fix, Enforce
Medium	Function apps should have Client Certificates (Incoming client certificates) enabled	Client certificates allow for the app to request a certificate for incoming requests. Only clients with valid certificates will be able to reach the app.	Unhealthy	ASC Default	App Service	Logic app, manual
Low	Audit usage of custom RBAC rules	Audit built-in roles such as 'Owner, Contributer, Reader' instead of custom RBAC roles, which are error prone. Using custom roles is treated as an exception and requires a rigorous review and threat modeling	Unhealthy	ISO: 27001:2013	Role Definitions	Manual
Medium	Diagnostic logs in App Service should be enabled	Audit enabling of diagnostic logs on the app.This enables you to recreate activity trails for investigation purposes if a security incident occurs or your network is compromised	Unhealthy	ASC Default	App Service	Logic app, manual
High	FTPS should be required in web apps	Enable FTPS enforcement for enhanced security	Unhealthy	ASC Default	App Service	Logic app, manual
Low	Deploy the Linux Guest Configuration extension to enable Guest Configuration assignments on Linux VMs	This policy deploys the Linux Guest Configuration extension to Linux virtual machines hosted in Azure that are supported by Guest Configuration. The Linux Guest Configuration extension is a prerequisite for all Linux Guest Configuration assignments and must be deployed to machines before using any Linux Guest Configuration policy definition. For more information on Guest Configuration, visit https://aka.ms/gcpol.	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
Medium	Key vaults should have purge protection enabled	Malicious deletion of a key vault can lead to permanent data loss. A malicious insider in your organization can potentially delete and purge key vaults. Purge protection protects you from insider attacks by enforcing a mandatory retention period for soft deleted key vaults. No one inside your organization or Microsoft will be able to purge your key vaults during the soft delete retention period.	Unhealthy	ASC Default	Key Vaults	Logic app, manual, Deny
Medium	Guest Configuration extension should be installed on machines	To ensure secure configurations of in-guest settings of your machine, install the Guest Configuration extension. In-guest settings that the extension monitors include the configuration of the operating system, application configuration or presence, and environment settings. Once installed, in-guest policies will be available such as 'Windows Exploit guard should be enabled	Unhealthy	ASC Default	Virtual Machine	Fix, Manual
Low	Azure Backup should be enabled for virtual machines	Protect the data on your Azure virtual machines with Azure Backup. Azure Backup is an Azure-native, cost-effective, data protection solution. It creates recovery points that are stored in geo-redundant recovery vaults. When you restore from a recovery point, you can restore the whole VM or specific files.	Unhealthy	ASC Default	Virtual Machine	Enforce, Manual
Medium	Managed identity should be used in web apps	For enhanced authentication security, use a managed identity. On Azure, managed identities eliminate the need for developers to have to manage credentials by providing an identity for the Azure resource in Azure AD and using it to obtain Azure Active Directory (Azure AD) tokens.	Unhealthy	ASC Default	App Service	Logic app, Manual

High	Microsoft Defender for SQL servers on machines should be enabled	Microsoft Defender for SQL is a unified package that provides advanced SQL security capabilities. It includes functionality for surfacing and mitigating potential database vulnerabilities, detecting anomalous activities that could indicate a threat to your database, and discovering and classifying sensitive data. Important: Remediating this recommendation will result in charges for protecting your SQL servers on machines. If you don't have any SQL servers on machines in this subscription, no charges will be incurred. If you create any SQL servers on machines on this subscription in the future, they will automatically be protected and charges will begin at that time.	Unhealthy	ASC Default	Subscriptio n	Fix, Enforce
Medium	Machines should have a vulnerability assessment solution	Defender for Cloud regularly checks your connected machines to ensure they're running vulnerability assessment tools. Use this recommendation to deploy a vulnerability assessment solution.	Unhealthy	ASC Default, ISO: 27001:2013	Virtual Machine	Fix, Manual
Low	Audit Linux machines that do not have the passwd file permissions set to 0644	Requires that prerequisites are deployed to the policy assignment scope. For details, visit https://aka.ms/gcpol. Machines are non-compliant if Linux machines that do not have the passwd file permissions set to 0644	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
Medium	Storage account should use a private link connection	Private links enforce secure communication, by providing private connectivity to the storage account	Unhealthy	ASC Default	Storage Account	Logic app, enforce
Medium	Private endpoint should be configured for Key Vault	Private link provides a way to connect Key Vault to your Azure resources without sending traffic over the public internet. Private link provides defense in depth protection against data exfiltration.	Unhealthy	ASC Default	Key Vaults	Logic app, Manual
High	FTPS should be required in function apps	Enable FTPS enforcement for enhanced security	Unhealthy	ASC Default	App Service	Logic app, Manual
Medium	Storage account public access should be disallowed	Anonymous public read access to containers and blobs in Azure Storage is a convenient way to share data, but might present security risks. To prevent data breaches caused by undesired anonymous access, Microsoft recommends preventing public access to a storage account unless your scenario requires it.	Unhealthy	ASC Default	Storage Account	Fix, Deny
Medium	Web apps should request an SSL certificate for all incoming requests	Client certificates allow for the app to request a certificate for incoming requests. Only clients that have a valid certificate will be able to reach the app.	Unhealthy	ASC Default	App Service	Logic app, Manual
Medium	Managed identity should be used in function apps	For enhanced authentication security, use a managed identity. On Azure, managed identities eliminate the need for developers to have to manage credentials by providing an identity for the Azure resource in Azure AD and using it to obtain Azure Active Directory (Azure AD) tokens.	Unhealthy	ASC Default	App Service	Logic app, Manual
Low	Diagnostic logs in IoT Hub should be enabled	Enable logs and retain them for up to a year. This enables you to recreate activity trails for investigation purposes when a security incident occurs or your network is compromised.	Unhealthy	ASC Default	IoT Hub	Fix, Manual
High	Container hosts should be configured securely	Remediate vulnerabilities in security configuration on machines with Docker installed to protect them from attacks.	Unhealthy	ASC Default	Container Host	Manual

Medium	Endpoint protection health issues on machines should be resolved	Resolve endpoint protection health issues on your virtual machines to protect them from latest threats and vulnerabilities. See the documentation for the endpoint protection solutions supported by Defender for Cloud and the endpoint protection assessments.	Unhealthy	ASC Default	Virtual Machine	Logic app, Manual
Low	Diagnostic logs in Event Hub should be enabled	Enable logs and retain them for up to a year. This enables you to recreate activity trails for investigation purposes when a security incident occurs or your network is compromised.	Unhealthy	ASC Default	Event Hub Namespace	Fix, Enforce
High	A maximum of 3 owners should be designated for subscriptions	To reduce the potential for breaches by compromised owner accounts, we recommend limiting the number of owner accounts to a maximum of 3	Unhealthy	ASC Default, ISO: 27001:2013	Subscriptio n	Manual
Low	Audit Linux machines that have accounts without passwords	Requires that prerequisites are deployed to the policy assignment scope. For details, visit https://aka.ms/gcpol. Machines are non-compliant if Linux machines that have accounts without passwords	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
High	External accounts with owner permissions should be removed from subscriptions	Accounts with owner permissions that have different domain names (external accounts), should be removed from your subscription. This prevents unmonitored access. These accounts can be targets for attackers looking to find ways to access your data without being noticed.	Unhealthy	ASC Default, ISO: 27001:2013	Subscriptio n	Manual
Low	Audit Linux machines that allow remote connections from accounts without passwords	Requires that prerequisites are deployed to the policy assignment scope. For details, visit https://aka.ms/gcpol. Machines are non-compliant if Linux machines that allow remote connections from accounts without passwords	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
Low	Audit diagnostic setting	Audit diagnostic setting for selected resource types	Unhealthy	ISO: 27001:2013		Manual
Medium	Storage accounts should restrict network access using virtual network rules	Protect your storage accounts from potential threats using virtual network rules as a preferred method instead of IP-based filtering. Disabling IP-based filtering prevents public IPs from accessing your storage accounts	Unhealthy	ASC Default	Storage Account	Logic app, manual, Deny
Medium	Firewall should be enabled on Key Vault		Unhealthy	ASC Default	Key Vaults	Logic app, enforce
High	Virtual machines should encrypt temp disks, caches, and data flows between Compute and Storage resources	By default, a virtual machine's OS and data disks are encrypted-at-rest using platform-managed keys; temp disks and data caches aren't encrypted, and data isn't encrypted when flowing between compute and storage resources. For a comparison of different disk encryption technologies in Azure, see https://aka.ms/diskencryptioncomparison. Use Azure Disk Encryption to encrypt all this data.	Unhealthy	ASC Default, ISO: 27001:2013	Virtual Machine	Manual
Low	Access to storage accounts with firewall and virtual network configurations should be restricted	Review the settings of network access in your storage account firewall settings. We recommended configuring network rules so that only applications from allowed networks can access the storage account. To allow connections from specific internet or on-premise clients, access can be granted to traffic from specific Azure virtual networks or to public internet IP address ranges.	Unhealthy	ISO: 27001:2013	Storage Account	Manual, Deny

Medium	Azure Cosmos DB accounts should have firewall rules	Firewall rules should be defined on your Azure Cosmos DB accounts to prevent traffic from unauthorized sources. Accounts that have at least one IP rule defined with the virtual network filter enabled are deemed compliant. Accounts disabling public access are also deemed compliant.	Unhealthy	ASC Default	Azure Cosmos DB	Logic app, manual, Deny
Low	Non-internet-facing virtual machines should be protected with network security groups	Protect your non-internet-facing virtual machine from potential threats by restricting access to it with a network security group (NSG). NSGs contain a list of Access Control List (ACL) rules that allow or deny network traffic to your VM from other instances, whether or not they're on the same subnet. Note that to keep your machine as secure as possible, the VM's access to the internet must be restricted and an NSG should be enabled on the subnet.	Unhealthy	ASC Default	Virtual Machine	Manual

Appendix F

Web page missing compliance

The spreadsheet show the current compliance problems with the standards Azure Security Benchmark v3 and ISO 27001:2013, for the web page case template. The spreadsheet lists the relevant sections of the standards, and shows which is compliant, what resource is affected and the reason for not being compliant. For further information about the problems causing the compliance issues, see Appendix G.

		Total Amount of policies	Compliante Policies	Missing compliance	Compliance	Resoruce Type	Reason for no compliance
NS. Network Secuirty	NS-1	5	4	Policy 1	Partial	VM, Subnets	Non-internet-facing vm should be protected by security groups
	NS-2	29	21	Policy 1,2,3,4,5,6,7,8	False	Azure Resoruces, Key vault, Storage Account, container, key vault	Firewall, private endpoint, dissallow public access, private link, restrict access
	NS-3	4	4		True	Virtual network, Azure Resources	
	NS-5	1	0	Policy 1	False	Virtual Network	DDoS protection should be active
	NS-6	2	1	Policy 1	False	Azure Resroces, Application Gateway	WAF should be active
	NS-7	1	1		True	VM	
	NS-8	3	3		True	Azure Resoruces, Web Application	
	NS-10	1	1		True	Subscriptions	
IM. Identity Managment	IM-1	2	2		True	Azure Resources	
	IM-3	4	4		True	Azure Resoruces, Web Application	
	IM-6	4	4		True	Subscriptions, VM	
PA. Privileged Access	PA-1	4	2	Policy 1, 2	False	Subcriptions	More than 3 owners, External accounts
	PA-2	1	1		True	Azure Resources	
	PA-4	5	4	Policy 1	Partial	Subscriptions	External Accounts with owner
	PA-7	2	2		True	Azure Resources, Azure Role Definitions	
DP. Data Protection	DP-2	5	4	Policy 1	Partial	Azure Resource, Subscriptions	Defender should be active for SQL
	DP-3	15	13	Policy 1,2	Partial	Azure Resources, Storage Account, Redis Cache, Web Application	Web Application only accessiable over https and FTPS required
	DP-4	4	3	Policy 1	False	Azure Resroces, VM	VM's Should encrypt temp disk/cache
	DP-5	9	9		True	Azure Resources	
	DP-6	2	2		True	Azure Resources	
	DP-7	1	1		True	Azure Resource	

	DP-8	6	2	Policy 1,2,3,4	False	Key vault, Subscriptions	Defender should be active for key vault, firewall, purge protection and Private endpoint
AM. Asset Management	AM-2	2	2		True	Storage Account, VM	
	AM-5	2	2		True	VM	
LT. Logging and Threat Detection	LT-1	13	12	Policy 1	Partial	Subscriptions , Azure Resrouce	Defender should be active for sql
	LT-2	13	12	Policy 1	Partial	Subscriptions , Azure Resrouce	Defender should be active for sql
	LT-3	13	10	Policy 1,2,3	False	Azure Resoruces, Key vault, Web Application	Diagnostic log should be enabled for key vault, App service and Event hub
	LT-5	5	5		True	Azure Resoruce, Subscriptions , VM	
	LT-6	1	1		True	Azure Resrouce	
IR. Incident Response	IR-2	3	3		True	Subscription	
	IR-3	11	10	Policy 1	Partial	Subscriptions , Azure Resrouce	Defender should be active for sql
	IR-4	1	1		True	Virtual Network	
	IR-5	11	10	Policy 1	Partial	Subscriptions , Azure Resrouce	Defender should be active for sql
PV. Posture and Vulnerability Management	PV-2	27	26	Policy 1	Partial	Azure Resoruces, Web Application	Web Apps should request SSL certificate
	PV-4	10	9	Policy 1	Partial	Azure Resoruces, VM	Guest Configuration should be installed on VM
	PV-5	3	3		True	Azure Resoruces, VM	
	PV-6	16	14	Policy 1,2	Partial	Azure Resoruces, VM, Container Host	Should be configured securely
ES. Endpoint Security	ES-1	1	1		True	Subscriptions	
	ES-2	5	4	Policy 1	Partial	Azure Resoruces, VM	Endpoint protection should be installed
	ES-3	1	1		True	Azure Resrouce	
BR. Backup and Recovery	BR-1	4	3	Policy 1	False	Azure Resoruces, VM	Azure backup should be enabled for VM
	BR-2	4	3	policy 1	False	Azure Resoruces, VM	Azure backup should be enabled for VM
DS. DevOps Security	DS-6	2	1	Policy 1	False	Azure Resoruce, Container Host	Should be configured securely

		Total Amount of policies	Compliante Policies	Missing compliance	Compliance	Resoruce Type	Reason for no compliance
A.6. Organization of Information Secuity	A.6.1.2.	2	1	Policy 1	False	Subscription	Amount of owners
A.9. Access Control	A.9.1.2.	8	5	Policy 1,2,3	False	Storage Account, VM	Audit, Guest Configuration and Audit vm that allow remote connection
	A.9.2.3.	7	5	Policy 1, 2	False	Subscription, Azure Resoruces, Role Definitions	External Accounts, Usage of Auditing
	A.9.2.4.	7	5	Policy 1,2	False	Subscriotion, VM	Auditing, Guest Configuration
	A.9.2.5.	4	3	Policy 1	False	External account with owner	
	A.9.2.6.	2	2		True	Subscription	
	A.9.4.2.	3	3		True	Subscription	
	A.9.4.3.	8	8		True	VM, Azure Resources	
A.10. Cryptography	A.10.1.1.	12	10	Policy 1, 2	Partial	Azure Resoruces, VM, Web Application, Redis Cache	Auditing not stroing passwords, encrypt disk/cache and deploy guest configuration
A.12. Operations Security	A.12.4.1.	6	5	Policy 1	False	Azure Resources	Auditing Diagnostic setting
	A.12.4.2.	6	5	Policy 1	False	Azure Resources	Auditing Diagnostic setting
	A.12.4.3.	6	5	Policy 1	False	Azure Resources	Auditing Diagnostic setting
	A.12.5.1.	1	1		True	Azure Resources	
	A.12.6.1.	4	3	Policy 1	False	Azure Resoruces, VM	Should be configured securly
	A.12.6.2.	1	1		True	Azure Resoruce	
A.13. Commications Security	A.13.1.1.	2	1	Policy 1	False	Vm, Storage Account	Access to storage account through firewall
	A.13.2.1.	2	2		True	Azure Resoruces, Storage Account	

Appendix G

Web page compliance/security problems

The spreadsheet shows the current security and compliance issues for the web page case template. The spreadsheet lists the problem, severity, a short description, affected resources, standard and the option for resolving the issues. The problems which contains fix, enforce, logic app or deny are currently not automated, and is easily fixed in GUI. The manual indicates that the problems would need a integration of script for automation, or is a more complex issue to resolve in GUI. For the project most of these problems was fixed with GUI, due to the amount of time it would take to automate. NOTE: The text for description in the spreadsheet, is auto generated text from the compliance manager in Microsoft defender for cloud [73].

Severity	Name	Description	Status	Initiatives	Resoruce	Options
High	Microsoft Defender for SQL servers on machines should be enabled	Microsoft Defender for SQL is a unified package that provides advanced SQL security capabilities. It includes functionality for surfacing and mitigating potential database vulnerabilities, detecting anomalous activities that could indicate a threat to your database, and discovering and classifying sensitive data. Important: Remediating this recommendation will result in charges for protecting your SQL servers on machines. If you don't have any SQL servers on machines in this subscription, no charges will be incurred.If you create any SQL servers on machines on this subscription in the future, they will automatically be protected and charges will begin at that time.	Unhealthy	ASC Default	Subscription	Fix, Enforce
High	A maximum of 3 owners should be designated for subscriptions	To reduce the potential for breaches by compromised owner accounts, we recommend limiting the number of owner accounts to a maximum of 3	Unhealthy	ASC Default, ISO: 27001:2013	Subscription	Manual
High	FTPS should be required in web apps	Enable FTPS enforcement for enhanced security	Unhealthy	ASC Default	App Service	Logic App
High	Microsoft Defender for Containers should be enabled	Microsoft Defender for Containers provides hardening, vulnerability assessment and run-time protections for your Azure, hybrid, and multi-cloud Kubernetes environments. You can use this information to quickly remediate security issues and improve the security of your containers.	Unhealthy	ASC Default	Subscription	Fix, Enforce
High	External accounts with owner permissions should be removed from subscriptions	Accounts with owner permissions that have different domain names (external accounts), should be removed from your subscription. This prevents unmonitored access. These accounts can be targets for attackers looking to find ways to access your data without being noticed.	Unhealthy	ASC Default, ISO: 27001:2013	Subscription	Manual
High	Container hosts should be configured securely	Remediate vulnerabilities in security configuration on machines with Docker installed to protect them from attacks.	Unhealthy	ASC Default	Container Host	Manual
High	Endpoint protection should be installed on machines	To protect machines from threats and vulnerabilities, install a supported endpoint protection solution.	Unhealthy	ASC Default	Virtual Machine	Fix
High	Virtual machines should encrypt temp disks, caches, and data flows between Compute and Storage resources	By default, a virtual machine's OS and data disks are encrypted-at-rest using platform-managed keys;temp disks and data caches aren't encrypted, and data isn't encrypted when flowing between compute and storage resources. For a comparison of different disk encryption technologies in Azure, see https://aka.ms/diskencryptioncomparison. Use Azure Disk Encryption to encrypt all this data.	Unhealthy	ASC Default, ISO: 27001:2013	Virtual Machine	Manual

Medium	Web Application should only be accessible over HTTPS	Use of HTTPS ensures server/service authentication and protects data in transit from network layer eavesdropping attacks.	Unhealthy	ASC Default, ISO: 27001:2013	App Service	Fix
Medium	Key vaults should have purge protection enabled	Malicious deletion of a key vault can lead to permanent data loss. A malicious insider in your organization can potentially delete and purge key vaults. Purge protection protects you from insider attacks by enforcing a mandatory retention period for soft deleted key vaults. No one inside your organization or Microsoft will be able to purge your key vaults during the soft delete retention period.	Unhealthy	ASC Default	Key vault	Logic App, Deny
Medium	Storage account public access should be disallowed	Anonymous public read access to containers and blobs in Azure Storage is a convenient way to share data, but might present security risks. To prevent data breaches caused by undesired anonymous access, Microsoft recommends preventing public access to a storage account unless your scenario requires it.	Unhealthy	ASC Default	Storage Account	Fix, Deny
Medium	Guest Configuration extension should be installed on machines	To ensure secure configurations of in-guest settings of your machine, install the Guest Configuration extension. In-guest settings that the extension monitors include the configuration of the operating system, application configuration or presence, and environment settings.	Unhealthy	ASC Default	Virtual Machine	Fix
Medium	Storage accounts should restrict network access using virtual network rules	Protect your storage accounts from potential threats using virtual network rules as a preferred method instead of IP-based filtering. Disabling IP-based filtering prevents public IPs from accessing your storage accounts.	Unhealthy	ASC Default	Storage Account	Logic App, Deny
Medium	Azure DDoS Protection Standard should be enabled	Defender for Cloud has discovered virtual networks with Application Gateway resources unprotected by the DDoS protection service. These resources contain public IPs. Enable mitigation of network volumetric and protocol attacks.	Unhealthy	ASC Default	Virtual Network	Manual
Medium	Container registries should not allow unrestricted network access	Azure container registries by default accept connections over the internet from hosts on any network. To protect your registries from potential threats, allow access from only specific public IP addresses or address ranges. If your registry doesn't have an IP/firewall rule or a configured virtual network, it will appear in the unhealthy resources.	Unhealthy	ASC Default	Container Registries	Logic App
Medium	Container registries should use private link	Azure Private Link lets you connect your virtual network to Azure services without a public IP address at the source or destination. The private link platform handles the connectivity between the consumer and services over the Azure backbone network. By mapping private endpoints to your container registries instead of the entire service, you'll also be protected against data leakage risks	Unhealthy	ASC Default	Container Registries	Logic App, Enforce

Medium	Machines should have a vulnerability assessment solution	Defender for Cloud regularly checks your connected machines to ensure they're running vulnerability assessment tools. Use this recommendation to deploy a vulnerability assessment solution.	Unhealthy	ASC Default, ISO: 27001:2013	Virtual Machine	Fix
Medium	Azure Cosmos DB accounts should have firewall rules	Firewall rules should be defined on your Azure Cosmos DB accounts to prevent traffic from unauthorized sources. Accounts that have at least one IP rule defined with the virtual network filter enabled are deemed compliant. Accounts disabling public access are also deemed compliant.	Unhealthy	ASC Default	Azure Cosmos DB	Logic App, Deny
Medium	Web apps should request an SSL certificate for all incoming requests	Client certificates allow for the app to request a certificate for incoming requests. Only clients that have a valid certificate will be able to reach the app.	Unhealthy	ASC Default	App Service	Logic App
Medium	Diagnostic logs in App Service should be enabled	Audit enabling of diagnostic logs on the app. This enables you to recreate activity trails for investigation purposes if a security incident occurs or your network is compromised	Unhealthy	ASC Default	App Service	Logic App
Medium	Firewall should be enabled on Key Vault	Key vault's firewall prevents unauthorized traffic from reaching your key vault and provides an additional layer of protection for your secrets. Enable the firewall to make sure that only traffic from allowed networks can access your key vault.	Unhealthy	ASC Default	Key vault	Logic App, Enforce
Medium	Private endpoint should be configured for Key Vault	Private link provides a way to connect Key Vault to your Azure resources without sending traffic over the public internet. Private link provides defense in depth protection against data exfiltration.	Unhealthy	ASC Default	Key vault	Logic App
Medium	Storage account should use a private link connection	Private links enforce secure communication, by providing private connectivity to the storage account	Unhealthy	ASC Default	Storage Account	Logic App, Enforce
Low	Audit usage of custom RBAC rules	Audit built-in roles such as 'Owner, Contributer, Reader' instead of custom RBAC roles, which are error prone. Using custom roles is treated as an exception and requires a rigorous review and threat modeling	Unhealthy	ISO: 27001:2013	Role Definitions	Manual
Low	Web Application Firewall (WAF) should be enabled for Application Gateway	Deploy Azure Web Application Firewall (WAF) in front of public facing web applications for additional inspection of incoming traffic. Web Application Firewall (WAF) provides centralized protection of your web applications from common exploits and vulnerabilities such as SQL injections, Cross-Site Scripting, local and remote file executions. You can also restrict access to your web applications by countries, IP address ranges, and other http(s) parameters via custom rules.	Unhealthy	ASC Default	Application Gateway	Logic App, Deny
Low	Audit Linux machines that do not have the passwd file permissions set to 0644	Requires that prerequisites are deployed to the policy assignment scope. For details, visit https://aka.ms/gcpol. Machines are non-compliant if Linux machines that do not have the passwd file permissions set to 0644	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual

Low	Audit Linux machines that have accounts without passwords	Requires that prerequisites are deployed to the policy assignment scope. For details, visit https://aka.ms/gcpol. Machines are non-compliant if Linux machines that have accounts without passwords	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
Low	Audit Linux machines that allow remote connections from accounts without passwords	Requires that prerequisites are deployed to the policy assignment scope. For details, visit https://aka.ms/gcpol. Machines are non-compliant if Linux machines that allow remote connections from accounts without passwords	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
Low	Access to storage accounts with firewall and virtual network configurations should be restricted	Review the settings of network access in your storage account firewall settings. We recommended configuring network rules so that only applications from allowed networks can access the storage account. To allow connections from specific internet or on-premise clients, access can be granted to traffic from specific Azure virtual networks or to public internet IP address ranges.	Unhealthy	ISO: 27001:2013	Storage Account	Manual, Deny
Low	Audit diagnostic setting	Audit diagnostic setting for selected resource types	Unhealthy	ISO: 27001:2013		Manual
Low	Azure Backup should be enabled for virtual machines	Protect the data on your Azure virtual machines with Azure Backup. Azure Backup is an Azure-native, cost-effective, data protection solution. It creates recovery points that are stored in geo-redundant recovery vaults. When you restore from a recovery point, you can restore the whole VM or specific files.	Unhealthy	ASC Default	Virtual Machine	Logic App, Enforce
Low	Non-internet-facing virtual machines should be protected with network security groups	Protect your non-internet-facing virtual machine from potential threats by restricting access to it with a network security group (NSG). NSGs contain a list of Access Control List (ACL) rules that allow or deny network traffic to your VM from other instances, whether or not they're on the same subnet. Note that to keep your machine as secure as possible, the VM's access to the internet must be restricted and an NSG should be enabled on the subnet.	Unhealthy	ASC Default	Virtual Machine	Manual
Low	Diagnostic logs in Key Vault should be enabled	Enable logs and retain them for up to a year. This enables you to recreate activity trails for investigation purposes when a security incident occurs or your network is compromised.	Unhealthy	ASC Default	Key vault	Fix, Enforce
Low	Diagnostic logs in Event Hub should be enabled	Enable logs and retain them for up to a year. This enables you to recreate activity trails for investigation purposes when a security incident occurs or your network is compromised.	Unhealthy	ASC Default	Event hub namespace	Fix, Enforce

Low	Deploy the Linux Guest Configuration extension to enable Guest Configuration assignments on Linux VMs	This policy deploys the Linux Guest Configuration extension to Linux virtual machines hosted in Azure that are supported by Guest Configuration. The Linux Guest Configuration extension is a prerequisite for all Linux Guest Configuration assignments and must be deployed to machines before using any Linux Guest Configuration policy definition.	Unhealthy	ISO: 27001:2013	Virtual Machine	Manual
Low	Machines should be configured securely		Unhealthy	ASC Default, ISO: 27001:2013	Virtual Machine	Manual

Anders Loeng, Morten Bratberg, Eirik Bakke

Secure and Compliant Infrastructure as Code

NTNU

Norwegian University of
Science and Technology