CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Enabling Secure Cloud Governance using Policy as Code

A code driven approach to automate the cloud governance

Master's thesis in Computer Systems and Networks

Arun Prakash Jothimani

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

Enabling Secure Cloud Governance using Policy as Code

A code driven approach to automate secure cloud governance

Arun Prakash Jothimani

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science And Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

Enabling Secure Cloud Governance using Policy as Code
A code driven approach to automate the cloud governance
Arun Prakash Jothimani

Supervisor: Ahmed Ali-Eldin Hassan, Chalmers CSE-NS
Advisor: Stefan Essman, Volvo Cars Corporation
Examiner: Tomas Olovsson, Chalmers CSE-NS

Master's Thesis 2022
Department of Computer Science And Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2022

Enabling Secure Cloud Governance using Policy as Code
A code driven approach to automate the cloud governance
Arun Prakash Jothimani
Department of Computer Science and Engineering
Chalmers University of Technology

Abstract

Cloud infrastructures are evolving at a rapid rate. Thus, it is important to ensure the stability and reliability of the cloud services [1] as they support many of today's critical systems. The Cloud Security Governance Deployment Framework [2] describes the critical security issues that must be considered and analyzed by the developer to ensure a secure cloud environment. With the rapid growth in the data and users, there is a need for solid rules to handle data storage and Identity Access Management(IAM). Lack of proper authentication management [3], user management, authorization management, access management, data management and monitoring can easily open doors for attackers to exploit the system [4]. The initiation, development, implementation, operation, and destruction phase has to be studied based on the cloud security critical domain guidelines, and risk considerations [2][5].
Policy-driven governance can be used to control the provisioning and consumption of cloud services. As discussed in [6], It is a challenging task to identify and implement scalable monitoring for different types of metrics [7] relevant to the Cloud infrastructure of the organization. The industrial state of the art in policy-based governance [8] has to meet the dynamic needs of the organization that changes throughout the period of time. The policy definition and evaluation are tightly coupled in one component via imperative languages, which hinders the easy evolution. This tightly coupled approach gives an opportunity to introduce policy-as-code [9] to modularize and decouple the policy environment for easy governance adoption. The policy based strategy will also provide a better solution to manage user credentials, user authentication and authorization [10].

Keywords : Cloud Governance, Policy, Infrastructure as code, Configuration Management, Automation

Acknowledgements

First and foremost, I would like to thank my industrial advisor Stefan Essman, Senior Network Architect, Volvo Cars and my academic supervisor Assistant professor Ahmed Ali-Eldin Hassan, Chalmers University of Technology, for their support, advice, and criticism during this entire thesis work. Needless to say, without their constant support and valuable ideas, this thesis work would not have been possible. Furthermore, I would like to thank my examiner, Associate professor Tomas Olovsson, Chalmers University of Technology, for his support and time throughout this work.

I also would like to mention special thanks to Christofer Åkerström, Team Manager, Volvo Cars and Adnan Sallova, Senior Cloud Solution Architect, Microsoft without whom this work would not have been possible. I extend my thanks to my brother Ramesh Jothimani for being a support line throughout my time at university.

Last but not least, I am thankful to my family and all my unnamed friends at Volvo, Chalmers for their help and support.

Arun Prakash Jothimani, Gothenburg, June 2021

Contents

List of Figures

1
Introduction

In recent times, cloud computing has become a buzzword for many industries. Numerous companies across the globe are rapidly migrating their infrastructure to the cloud and discussing the opportunities that can be reaped out of the multi-cloud strategy. This rise in popularity has introduced new challenges for cloud governance [11]. According to Gartner, "the rapid adoption of cloud services, along with an increasing number of cloud infrastructure and platform services, has created an explosion in the complexity and unmanaged risks". Although the cloud infrastructure has made it easy to create, read, update, and delete resources for the company, it can also lead to unintended security consequences, if the unrestricted resource access to the cloud is exploited. As the number of services and users added to the cloud infrastructure grows exponentially in the course of time, it becomes significant for a company to ensure that all the cloud services and users are strictly complying to the company policies and industrial standards [12].

1.1 Aim

The thesis work intends to benchmark the governance security baseline and identity baseline of the cloud services used by Volvo Cars against the industrial standards and identify the best possible technical options to follow and implement. The thesis studies the application of automation via Infrastructure as code [9] to enable the provisioning of resources to be more agile while maintaining the privacy and security requirements of the application to provide better control over the environment. Cloud governance [11] through configuration management is a necessary technique to manage the changes in a standard and repeatable way. The proposed Proof Of Concept, automates the process of policy creation, declarative configuration and deployment in the Cloud infrastructure.

1.2 Research Questions

The thesis answers the following research questions

- How to measure the benchmark score of the security and identity baseline?
- How to improve the existing security baseline?
- What tools can be used in automating the cloud governance baselines?
- What policies should be automated to get an ideal score ?
- What are the benefits in automating the cloud baseline?

1.3 Approach

The project begins with a study about the existing security and identity baseline of the organization. The relevant literature were identified to analyze the gaps with respect to the state-of-the-art. Cloud Adoption Framework Governance Benchmark Tool [1] has been employed to benchmark the security and identity baseline of the existing state and desired future state of the cloud infrastructure. To achieve the desired benchmark score, its very significant to identify the technical risks associated to the security baseline of the cloud infrastructure, formulate the policy requirements, and to explore the possible technical design options so that the best possible implementation can be chosen.

To enable infrastructure as a code, After the identification of the right provisioning and configuration tools, a prototype, policy-as-code with a declarative language has to be created that decouples the custom policy definition, execution and evaluation.

The improved security baseline of governance is evaluated by an execution over a test bed and with a set of developer interviews. Then the set of data is fed into the Cloud Adoption Framework Governance Benchmark Tool to identify if the system has reached the ideal benchmark score. The current and ideal benchmark score are generally calculated by the Cloud Adoption Framework Governance Benchmark Tool based on the collected current state and desired state input data.

1.4 Intended Project Result

This thesis is driven by the technical security challenges faced by Volvo Cars with respect to the enforcement of cloud governance policies when deploying their workload in the Microsoft Azure Cloud.

My contribution to the thesis is divided into two areas, a research-oriented approach to structure a standardized strategy to achieve the ideal cloud governance security baseline benchmark score for the cloud infrastructure of the organization. My next contribution is to automate the policy driven governance proposing a prototype, that can allow or deny the configurations based on the system's compliance with the proposed technical options.

As a part of this thesis work, The current security and identity baseline governance state of the organization is studied with the help of study guides and short developer interviews for the purpose of initial understanding. This is followed by the documentation of desired future state to be achieved after the incremental improvement to the security baseline of cloud governance. Thus, the thesis work intends to identify the risk tolerance indicators (Example: standard compliance trigger) and metrics, policy compliance processes, data classification schema and the security baseline tool chain associated with the cloud environment of the organization. As a part

[1] https://cafbaseline.com/

of policy compliance processes, the thesis work analyses the policy requirements in depth to propose the best possible technical options as actionable items. Finally, an automated formal reasoning prototype for the configuration analysis and mitigation is scripted by enabling Infrastructure-as-code.

1.5 Risk analysis and Ethical Consideration

Any of the confidential data used in the thesis work will not be exposed in the final thesis report. Required data are substituted with the corresponding mock data during the final report submission.

1.6 Limitations

Only the security baseline of Governance is in the scope of this thesis. At this moment, CI CD Integration is not considered into the account.

2
Background

Cloud computing has becomes the popular way of delivering computing services over the internet. The computing service can be servers, storage, databases, networking, software, analytics, and intelligence. The cloud computing model relies on pay as you go technique, where users get the sophistication to pay only for the amount of resources used by them.

2.1 Fundamental of Cloud Computing

Cloud computing is built on the top of utility computing, virtualization, service oriented architecture and parallel computing[13] leveraging the existing technologies to meet the current technical and economical requirement of the industry. The cloud computing model has enabled a way to offer general utilities that can be leased and released by the user via internet in an on demand fashion [14].

By definition, "A Cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers".- Buyya R et al [15].

Agility, reliability, scalability, pay-per-use, on demand service, resiliency are the core features of cloud computing [13]. Agility is the ability of the environment to respond and adopt to the changing conditions in a rapid phase. Reliability refers to the probability of getting the tasks done without any failure. The distributed nature of cloud makes it more reliable. Scalability is the ability of the cloud to increase or decrease its offering limit based on the user needs. Pay-per-use property enables an easy-to-use method where user can pay only for the resources he uses. On demand service enables user to use the cloud service whenever they want rather than integrating it as a permanent infrastructure. Resiliency of the cloud providers ensure the fault tolerance in the infrastructure. The performance analysis and monitoring can be easily achieved using web services.

2.2 Cloud computing - Service Models

The three types of cloud service models are

- Software-as-a-Service SaaS

- Platform-as-a-Service PaaS
- Infrastructure-as-a-Service IaaS

Software-as-a-Service

Software as a Service (SaaS) is a model via which a client can rent an application from the cloud provider and use it without actually installing the application in his local system [16]. Thus, the client can utilize the features of the licensed application using interfaces like browsers. The SaaS application are more focused towards the end user experience as the provider takes the responsibility to manage the underlying infrastructure [17]. This is a most preferred way to use the lightweight applications.

This model has lot of benefits. A few of the core benefits are listed below [16].

- Software licensing cost will be greatly reduced.
- No infrastructure overhead.
- Customers can easily access the SaaS application as they can be instantly deployed to millions of customers.
- Pushing patches and updates is super easy.

Platform-as-a-Service

Platform as a service provides an environment that can be utilized by the user to create, run and deploy the application [16] without any ambiguity of installing or configuring the development environment. As the PaaS is built on the top of IaaS, the advantages PaaS can offer includes hardware virtualization, dynamic resource allocation and reduced investment costs [18].

The services like compute, storage, version management, testing has enabled a rapid and efficient product development. PaaS is heavily used to design and deploy cloud based applications.

Infrastructure-as-a-Service

Infrastructure as a service offers servers, storage and virtualization as a service to the users via web based service. The users are given with wide range of permissions in the server to manage their infrastructure, ranging from operating systems to applications.

The term infrastructure includes but not limited to facility, communication networks, physical compute nodes, virtual machines and their operating systems, storage, etc [18]. These resources are highly scalable and provided on-demand, thus they are billed completely based on the usage. Infrastructure operation, hosting and maintenance are completely taken care by the provider. The major benefits are infrastructure scalability, on demand infrastructure setup, reduced ROI risk and low investment costs [16].

2.3 Major cloud service providers

Cloud service providers are vendors who help the user in accessing the data from a remote server. These providers have their own data centers to host the infrastructure and required services for the clients. Every organization choose their cloud providers based on their requirements, priorities and evaluation criteria.

Though there are numerous cloud service providers in the market, Major cloud providers offering SaaS, PaaS, IaaS services are Amazon Web Services by Amazon, Microsoft Azure by Microsoft and GCP by Google. Based on the analysis carried out during February 2020, Analysis reports confirm that AWS reserves 32.4 percent of market share, Azure reserves 17.6 percent of market share while Google Cloud has 6 percent of market share.

Amazon Web Services

AWS is launched by Amazon in the year 2006. AWS is dominating the cloud space with some enhanced features like configuration, monitoring, security, auto-scaling, etc. AWS is relying on a per-hour-billing model [19]. The key cloud tools associated with AWS are Athena, QuickSight, Sage Maker, Lex, Greengras IoT, AWS Lambda & Deep Lens.

The major benefits of AWS are its easy-to-use nature, flexible, cost-effective, reliable, secure, scalable, worldwide reach while it has some limitations as well. It is cost prohibitive and has costly technical support [19][20].

Microsoft Azure

Azure is launched by Microsoft in 2008. Azure in known for its better development and testing tools. Azure becomes more handy and reliable when there is a need to integrate with Microsoft products. It relies on the per minute - rounded up(prepaid or monthly) billing model. The key cloud tools associated with Azure are HD Insights, Azure Data Factory, Azure ML Studio, Azure Boot Services, Cognitive Service, IoT Hub, Functions [19].

The core strength of Azure lies in its ability to integrate with Microsoft suit, billing adjustment, hybrid capability, technical support while its drawbacks inclines towards the limited DevOps support.

Google Cloud Platform

GCP is launched by Google in the year 2011. GCP shows promising results when the focus is on DevOps. CP relies on the per minute - rounded up (minimum 10 minutes) billing model [19]. The key cloud tools associated with GCP are BigQuery, Cloud DataFlow, Cloud ML Engine, Cloud IoT core and Cloud functions.

The GCP has DevOps support as its strongest feature. As it is pretty latest comparing other major providers, GCP is yet to come up with as many features as the consumers like to have [19][20].

2.4 Cloud Security Governance

Cloud computing has brought down the operating costs massively, improved the speed and agility of the system, but not without a limitation. The subscription to a cloud services brings a lot of security challenges to focus on. The most common challenges are data breach, identity access management, vulnerability remediation, data encryption and end point protection.

There could be multiple factors that could result in cloud security breach. One significant reason is ineffective cloud governance. The inadequate organizational responsibility and minimal enforcement of policies and procedures can cost a lot for the company in terms of reputation. Cloud security governance offers an effective security management in cloud, thus helping the organization in achieving their business goals.

An organization must have a standardized process [8] for cloud infrastructural security to make the environment more repeatable, consistent, and to avoid configuration errors during the application creation, deletion and modification. As Gartner states, "through 2022, at least 95% of cloud security will be the customer's fault". It also stats that "Soon, most of the attacks in the cloud environment will be the result of misconfigurations, lack of customizable security profiles, and auto-remediation by organizations in their day-to-day applications."

Volvo cars are slowly migrating their workloads to the cloud and exploring the application of automation to make the infrastructure more repeatable, scalable, automatable and auditable. As multiple teams are working on Azure, the development of policy as code will help in managing multiple subscriptions, enforcing regulatory requirements and standards to be followed. As the team has just started their transition, it is very significant to reach the ideal governance benchmark score. As a spark to the process, this thesis work benchmark the existing scenario in a scale using cloud adoption framework [1] and bridges the gap to help the team to achieve the ideal score.

Policy as Code has multiple applications that can benefit the company, for example, it can help the company to manage azure policies across multiple environments with ease. This gives very less or no possibility for human errors as the policies are deployed using DevOps/scripts. Enabling audit trails before making any configuration changes will act as the guardrails to protect the automated system from causing any

[1] https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/govern/security-baseline/

hazardous impacts.

Though the company has deployed few policies for secure governance, the system is in the early stage of the development and there exist, a lot of scope to study further to improve the policy driven approach to enable a secure governance. Rather than spending consistent efforts in educating the team to choose the best technical options for a number of governance issues and letting them manually enforce the security configurations on the cloud environment to satisfy corporate security standards, rules and effects over the resources to stay compliant with the company's security policies and service level agreements [21], the approach of having an automated policy driven governance is a better alternative.

There is always a possibility that multiple policies are assigned in a scope and changes may happen dynamically depending upon the requirement of the organization. All these significant problems calls for an improved approach to policy engineering with increased automation and efficient structuring of cloud governance [7]. This can help the project to stay compliant with industry standards as well as corporate or organizational standards [6] which is addressed by this thesis.

3
Method

3.1 Cloud Adoption Framework

The Cloud Adoption Framework is a proven guide designed to help you create and implement the business and technology strategies that enable your organization to succeed in the cloud [22].

This framework contains tips, documentation and tools which is relied on by business decision influencer, business decision maker, data professional, primary dev decision maker, IT decision maker to analyze the organization readiness, cost management, security baseline, resource consistency, deployment acceleration and identity baseline.

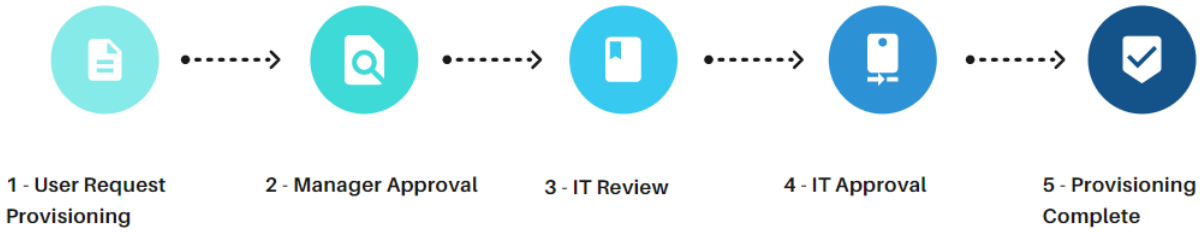

Figure 3.1: Traditional Approach

With cloud governance, the process becomes much faster and productivity increase in scale. The mechanisms offered by governance helps the organization to have complete control over the resources with proper prioritization. Azure Governance primarily relies on the implementation of two significant features, namely Azure Policy and Azure Cost Management.

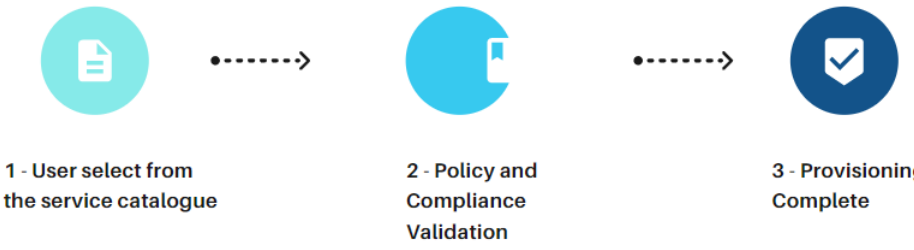

Figure 3.2: Cloud Governance Approach

Cloud Governance is an iterative process. The process flow has the following sequence.

- Methodology
- Benchmark
- Initial Governance Foundation
- Improvement on the initial governance foundation

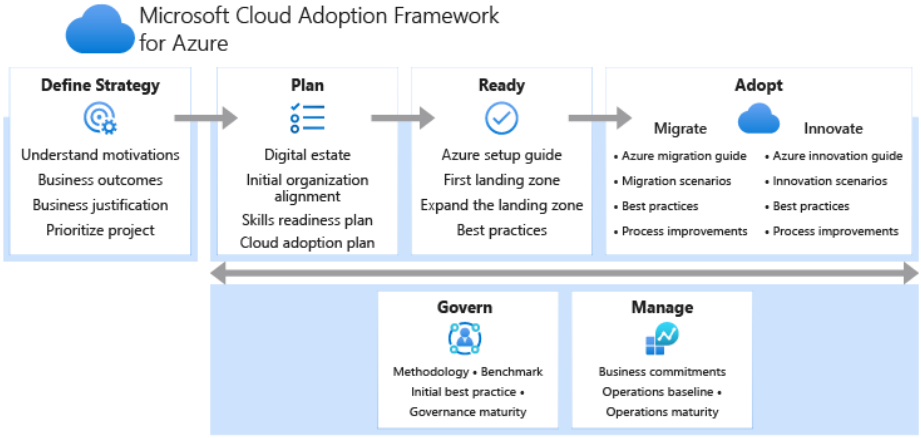

Figure 3.3: Cloud Governance Methodology

The above figure is taken from the official Microsoft site[1]. The cloud adoption framework Methodology has the following steps.

- Define Strategy
- Plan
- Ready
- Adopt
- Govern
- Manage

3.2 Infrastructure as Code

In this fast moving world, relying on a manual process for executing a number of complex tasks is highly impossible. The manual processes involve higher cost, shows inconsistency, and they lack agility as well. It is really cumbersome to maintain the compliance when the system is fueled by manual processes. Automation came as a one-stop solution to make the process faster and less prone to manual errors. This gives flexibility to the developers to focus on the core logic creation rather than battling with the everlasting production issues.

After the advent of DevOps, Infrastructure design has become a crucial phase in software life cycle. Infrastructure design will generally include a script that can reproduce the infrastructure based on the input from the user that contain details

[1]https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/operating-model/define

about cloud setup, Number of VMs, resource budget constrains, location of deployment, deployed policies, configuration details etc.

The usage of high level language to manage and automate policies borne fruitful results, which resulted in the advent of policy as code. It makes the environment scalable, repeatable, automatable and auditable with ease.

Documenting the Azure policies as code files is a very promising technique to repeat the deployment of the environment anytime with a single click. It also helps us in managing the azure policies across environments like production, development and testing without any manual dependencies. Automation in deployment help us remove the possibility for any human errors. Auditing the environment for change in Azure policy is also hassle-free when policy as code is in-place.

3.3 Omitted Results

The cloud adoption framework [2] has been employed to assess the existing cloud infrastructure of the company. Thus, aiming to bridge the gap between the existing score and industrial benchmark score.The current score of Security baseline and Identity baseline has been recorded by asking a set of questions (attached to appendix) to the architects of the Cloud Infra and Networks Team of Volvo Cars. The exact score cannot be disclosed here due to security reasons.Based on my research, I have come up with a set of recommendations over the policies that could be employed on the infrastructure and that is expected to evolve over the period of time based on the organizational needs. The percentage of difference observed in the benchmark score after the policy improvement has been recorded as metrics for the purpose of this report.

[2]https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/govern/security-baseline/

4

Results

The policies in Azure play a crucial role in identifying the security misconfigurations that can exist in the cloud infrastructure of the organization. When the policies are deployed, they analyze the compliance status of the resources in the cloud infrastructure as well as restrict the user from doing any future misconfiguration. To improve the Industrial benchmark score, the security infrastructure of the organization has been analyzed and a set of policies are recommended as a part of this research work to improve the security governance of the cloud infrastructure.

As a part of the thesis work, the areas like ports security management, Vulnerability remediation, data encryption while it is at rest & transit, Identity Access Management, End point protection are analyzed and potential measures to guardrail the infrastructure are charted as policy recommendations.

The recommended policies are taken into account and applied over the test bed cloud infrastructure, that resulted in potential increase in benchmark score when measured via cloud adoption framework. The scoring mechanism and its parameters are introduced below.

The score control has the following parameters.

- Max Score
- No Of Resources
- Current Score
- Potential Increase

Max Score

This is the maximum value that can be achieved when the system is kept completely under control.

No Of Resources

The existing contribution of every resource can be calculated by dividing that max score by the number of resources affected. Example $4/20 = 0.2$

Current Score

$Currentscore = [Scoreperresource] * [Numberofhealthyresources]$
$0.2 * 1healthyresource = 0.2$

Potential Increase

$Potentialscoreincrease = [Scoreperresource] * [Numberofunhealthyresources]$
$0.2 * 3unhealthyresources = 0.6$

The thesis work recommends the following set of polices to improve the security baseline of the cloud infrastructure.

4.1 Security Management ports

This control is made up of 8 points in total and comprises 3 recommendations. Though the management ports play a crucial role in the management, unfortunately they happen to be an entry point for attackers. It is very significant to limit the ports exposure and availability. It is highly recommended that, the ports should be enabled only for the recognized requests.

- Internet-facing virtual machines should be protected with network security groups.
- Management ports should be closed on virtual machines.
- Management ports of virtual machines should be protected with just-in-time network access control.

1. **Internet-facing virtual machines should be protected with network security groups**

Network Security Group [NSG] is a collection of zero or more rules within Azure subscription limits to filter the network traffic that moves to and from Azure resources in the Azure virtual network.

The unprotected VMs with open management ports can become prime target of RDP & SSH brute force attacks. There could be some scenarios where public facing VMs the requirements, For example DEV Environment. Network Security Groups are a potential way to limit the public access to VMs. NSGs Access control list can be assigned to VMs NIC or subnet which helps in controlling the network traffic with allow/deny options. Thus, for a better security the VM access to the internet must be restricted and NSG should be enabled on the subnet.

Manual Steps

1. Select a VM from the list.
2. Assign the relevant NSG to the NIC or subnet for the VM you're protecting:
   (a) To assign the NSG to the VM's subnet (recommended):
      i. In the Networking page, select the 'Virtual network/subnet'.

ii. Open the 'Subnets' menu.
iii. Select the subnet where your VM is deployed.
iv. Select the NSG to assign to the subnet and click 'Save'.

(b) To assign the NSG to the NIC:
i. In the Networking page, select the network interface that's associated with the selected VM.
ii. In the Network interfaces page, select the 'Network security group' menu item.
iii. Click 'Edit' at the top of the page.
iv. Follow the on-screen instructions and select the Network Security Group to assign to this NIC.

Policy Details

- Policy Name: *Internet-facing virtual machines should be protected with network security groups*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/f6de0be7-9a8a-4b8a-b349-43cf02d22f7c
- Effect : AuditIfNotExists

2. **Management ports should be closed on your virtual machines**

The management ports include ports that are used as a point to access Azure virtual machines using protocols like Remote Desktop Protocol and Secure Shell Protocol. Once these ports are compromised, the attacker may utilize the VM as a starting point to compromise the other related machines in the network. During inbound port selection, you can witness that RDP or SSH are selected by default in the Azure portal. It is recommended

- To disable the public access of RDP & SSH to the Azure virtual machines.
- To deny VMs with NICs having public IP addresses.

To limit the resource access level, the NSGs can be used along with JIT to manage the inbound VM Access. Two popular strategies to achieve this are

- DNAT Rules on Azure Firewall
- Azure Bastion

DNAT Rules on Azure Firewall

Azure Firewall Destination Network Address Translation (DNAT) can be configured to manage the inbound internet traffic to the subnets. It provides the functionality of inbound internet traffic translation and filtering. The NAT rule is utilized in the translation of Firewall public IP address and ports into private IP address and port.

Azure Bastion

It enables secure RDP/SSH connectivity to the virtual machines directly from Azure portal over TLS.

Manual Steps

1. Select a VM from the list.
2. Deny/Improve the rules under Networking Blades. Focus on the rules that allow management ports (e.g. RDP-3389, WINRM-5985, SSH-22).
3. Save.

Policy Details

- Policy Name: *Internet-facing virtual machines should be protected with network security groups*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/f6de0be7-9a8a-4b8a-b349-43cf02d22f7c
- Effect : AuditIfNotExists

3. **Management ports of virtual machines should be protected with just-in-time network access control**

The just-in-time (JIT) access is a fundamental security practice to create a set of rules that can be applied over the network security group, in turn which can be used to lock down the inbound traffic to offer a privilege access to the system for the pre-determined period of time.

Enabling just-in-time access control could protect the VM from internet based brute force attacks.

Manual Steps

1. Remediation can be done from the Azure Security Center.
2. User can define the ports to be considered, at JIT VM Configuration page.
3. Save.

Policy Details

- Policy Name: *Management ports of virtual machines should be protected with just-in-time network access control*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/f6de0be7-9a8a-4b8a-b349-43cf02d22f7c
- Effect : AuditIfNotExists

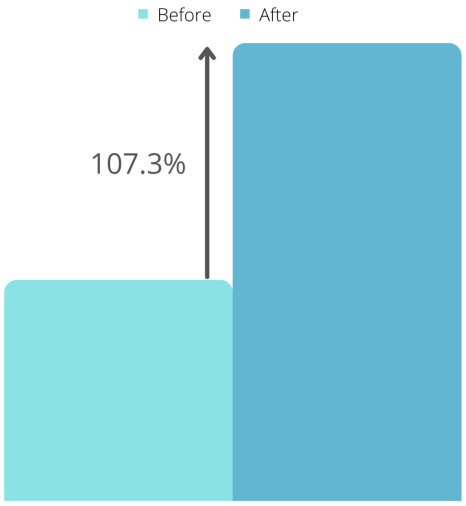

Figure 4.1: Security Management ports

After the implementation of above recommended policies, the cloud adoption framework was used to measure the increase in the benchmark score. As a result, I observed that there is an increase of 107.3 percent comparing the existing benchmark score.

4.2 Remediate Vulnerabilities

1. Azure Defender for SQL should be enabled on your SQL servers

Azure Defender for SQL helps in detecting the anomalous activities, mitigating the vulnerabilities that could be a potential threat to the database.

Policy Details

- Policy Name: *Internet-facing virtual machines should be protected with network security groups*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/f6de0be7-9a8a-4b8a-b349-43cf02d22f7c
- Effect : AuditIfNotExists

2. Vulnerability assessment should be enabled on your SQL servers

Enabling vulnerability assessment can help in discovering and tracking the vulnerability, thus a possibility to remediate.

Policy Details

- Policy Name: *Vulnerability assessment should be enabled on your SQL servers*
- Definition ID: Vulnerability assessment should be enabled on your SQL servers
- Effect : AuditIfNotExists

3. Vulnerability assessment findings on your SQL databases should be remediated

SQL Vulnerability assessment provides the data by scanning the database, that exposes any misconfigurations, excessive permissions, and unprotected sensitive data.

Policy Details

- Policy Name: *Vulnerabilities on your SQL databases should be remediated*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/feedbf84-6b99-488c-acc2-71c829aa5ffc
- Effect : AuditIfNotExists

4. Vulnerability assessment findings on your SQL servers on machines should be remediated

SQL Vulnerability assessment provides the data by scanning the database, that exposes any misconfigurations, excessive permissions, and unprotected sensitive data.

Policy Details

- Policy Name: *Vulnerability assessment findings on your SQL servers on machines should be remediated*
- Definition ID:
- Effect : AuditIfNotExists

5. A vulnerability assessment solution should be enabled on your virtual machines

Microsoft Azure offers end point protection that prevents the Azure services from getting affected by the malware such as viruses and spyware. This feature can be utilized by applying relevant policies over the cloud infrastructure to monitor any kind of potential threats.

Policy Details

- Policy Name: *Vulnerabilities on the SQL servers on the machine should be remediated*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/6ba6d016-e7c3-4842-b8f2-4992ebc0d72d
- Effect : AuditIfNotExists

6. Vulnerabilities in Container Registry images should be remediated

This policy assess every container images for any possible vulnerabilities, thus exposing any potential threat to the images. It is highly recommended to resolve the vulnerabilities as soon as it is detected by the policy.

Policy Details

- Policy Name: *Vulnerabilities in Azure Container Registry images should be remediated.*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/5f0f936f-2f01-4bf5-b6be-d423792fa562
- Effect : AuditIfNotExists

7. Azure Policy Add-on for Kubernetes should be installed and enabled on your clusters

There exists an add-on which can be utilized from GitHub, to enhance the security in Azure. Azure Policy Add-on for Kubernetes to extend Gatekeeper v3 to enable at-scale enforcement, safeguarding the cluster in a centralized and consistent manner should be enabled. This installation can be made mandatory by enabling this policy in the cloud infrastructure.

Policy Details

- Policy Name: *Azure Policy Add-on for Kubernetes service (AKS) should be installed and enabled on the clusters*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/0a15ec92-a229-4763-bb14-0ea34a568f8d
- Effect : Audit

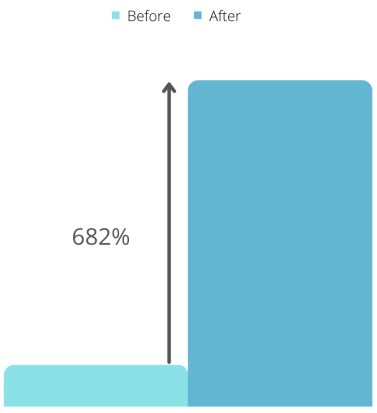

Figure 4.2: Remediate Vulnerabilities

After the implementation of above recommended policies, the cloud adoption framework was used to measure the increase in the benchmark score. As a result, I observed that there is an increase of 700 percent comparing the existing benchmark score.

4.3 Encrypt data in transit

The popular attacks like man-in-the-middle attacks, eavesdropping and session hijacking target the data in transit. If no proper data protection strategy is in place, the organization will be in the danger of exposing the sensitive data.The data transmission may be from client to server, server to server, or between the in-house system & third party system.

Some notable recommendations are:

- Web Application should only be accessible over HTTPS.
- Function App should only be accessible over HTTPS.
- API App should only be accessible over HTTPS.
- Only secure connections to your Azure Cache for Redis should be enabled.
- Secure transfer to storage accounts should be enabled.
- TLS should be updated to the latest version for your API app.
- Enforce SSL connection should be enabled for PostgreSQL database servers.
- TLS should be updated to the latest version for the web app.
- TLS should be updated to the latest version for the function app.
- FTPS should be required in the function App.
- FTPS should be required on the web App.
- FTPS should be required in the API App.

1. Web Application should only be accessible over HTTPS

Enabling HTTPS protects the data from eavesdropping stacks targeted on them during the transit at network layer Manually, we can redirect all HTTP traffic to HTTPS by following steps:

- Open App service.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *Web Application should only be accessible over HTTPS*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/a4af4a39-4135-47fb-b175-47fbdf85311d
- Effect : Audit

2. Function App should only be accessible over HTTPS

Enabling HTTPS protects the data from eavesdropping stacks targeted on them during the transit at network layer Manually, we can redirect all HTTP traffic to HTTPS by following steps:

- Open App service.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *Function App should only be accessible over HTTPS*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/6d555dd1-86f2-4f1c-8ed7-5abae7c6cbab
- Effect : Audit

3. API App should only be accessible over HTTPS

Enabling HTTPS protects the data from eavesdropping stacks targeted on them during the transit at network layer Manually, we can redirect all HTTP traffic to HTTPS by following steps:

- Open App service.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *API App should only be accessible over HTTPS*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/b7ddfbdc-1260-477d-91fd-98bd9be789a6
- Effect : Audit

4. Only secure connections to your Azure Cache for Redis should be enabled

Enabling the connections via SSL to Redis Cache ensures that data flowing between the server and service are protected from man-in-the-middle attack, eavesdropping, and session-hijacking.

Manual Steps

- Move to the Redis Cache.
- Select Advanced Settings.
- Enable Allow access only via SSL & Save.

Policy Details

- Policy Name: *API App should only be accessible over HTTPS*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/b7ddfbdc-1260-477d-91fd-98bd9be789a6
- Effect : Audit, Deny

5. Secure transfer to storage accounts should be enabled
Enabling secure transfer forces, the storage account to accept requests from the secure connections, thus helping the system to resist any man-in-the-middle attack, eavesdropping and session hijacking.

Manually, we can redirect all HTTP traffic to HTTPS by following steps
- Open App service.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *Secure transfer to storage accounts should be enabled*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/404c3081-a854-4457-ae30-26a93ef643f9
- Effect : Audit, Deny

6. TLS should be updated to the latest version for your API app

It is always recommended to be at the latest version of TLCS.

Manual Steps

- Open App service.
- Select TLS/SSL settings.
- Select the latest TLS version under protocol settings section.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *Latest TLS version should be used in your API App*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/8cb6aa8b-9e41-4f4e-aa25-089a7ac2581e
- Effect : AuditIfNotExists

7. Enforce SSL connection should be enabled for PostgreSQL database servers
Azure database for PostgreSQL server can be securely connected to the client application via a Secure Socket Layer. This helps the system to resist the man-in-the-middle attacks, as the data stream involved in the communication is completely encrypted.

Manual Steps

- Select the Azure Database for PostgreSQL.
- Enable SSL Connection under Connection Security section.

Policy Details

- Policy Name: *Enforce SSL connection should be enabled for PostgreSQL database servers*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/d158790f-bfb0-486c-8631-2dc6b4e8e6af
- Effect : Audit

8. TLS should be updated to the latest version for your web app

It is always recommended to be at the latest version of TLS

- Open App service.
- Select TLS/SSL settings.
- Select the latest TLS version under protocol settings section.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *Latest TLS version should be used in your Web App*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/f0e6e85b-9b9f-4a4b-b67b-f730d42f1b0b
- Effect : AuditIfNotExists

9. TLS should be updated to the latest version for your function app

It is recommended to always be at the latest version of TLS.

Manual Steps

- Open App service.
- Select TLS/SSL settings.
- Select the latest TLS version under protocol settings section.
- Toggle HTTPS Only to ON under TLS/SSL Settings.

Policy Details

- Policy Name: *Latest TLS version should be used in your Function App*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/f0e6e85b-9b9f-4a4b-b67b-f730d42f1b0b
- Effect : AuditIfNotExists

10. FTPS should be required in your function App

Always enable FTPS for availing enhanced security.

Manual Steps

- Open App service of your API App.
- Select General Settings tab under Configuration.
- Select FTPS only under FTP State.

Policy Details

- Policy Name: *FTPS should be required in your function App*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/399b2637-a50f-4f95-96f8-3a145476eb15
- Effect : AuditIfNotExists

11.FTPS should be required in your web App

Always enable FTPS for availing enhanced security.

Manual Steps

- Open App service of your API App.
- Select General Settings tab under Configuration.
- Select FTPS only under FTP State.

Policy Details

- Policy Name: *FTPS should be required in your web App*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/4d24b6d4-5e53-4a4f-a7f4-618fa573ee4b
- Effect : AuditIfNotExists

12.FTPS should be required in your API App

Manual Steps

- Open App service of your API App.
- Select General Settings tab under Configuration.
- Select FTPS only under FTP State.

Policy Details

- Policy Name: *FTPS should be required in your API App*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/9a1b8c48-453a-4044-86c3-d8bfd823e4f5
- Effect : AuditIfNotExists

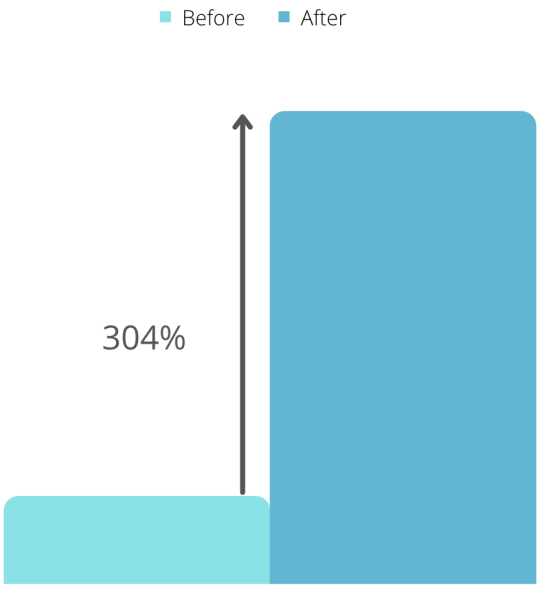

Figure 4.3: Encrypt data in transit

After the implementation of above recommended policies, the cloud adoption framework was used to measure the increase in the benchmark score. As a result, I observed that there is an increase of 304 percent comparing the existing benchmark score.

4.4 Enable encryption at rest

- Disk encryption should be applied on virtual machines.
- Transparent Data Encryption on SQL databases should be enabled.
- Service Fabric clusters should have the ClusterProtectionLevel property set to EncryptAndSign.
- Bring your own key data protection should be enabled for PostgreSQL servers.

1. Disk encryption should be applied on virtual machines

By default, managed disks are encrypted at rest by default Azure storage service encryption using Microsoft managed keys. If that's not the case, Azure Disk Encryption is the way to go. The machine can be protected by either one pass encryption or two pass encryption. One pass encryption is enabled if all the InstanceView.disks elements have encryptionSetttings.enabled == True OR Resource.ADE.Version (vm extension) starts with 1 pass major version Two pass encryption enabled if the storageProfile.OsDisk.encryptionSetttings.enabled == True Azure Disk Encryption uses Bit Lock feature of windows to enable full disk encryption.

Policy Details

- Policy Name: *Disk encryption should be applied on virtual machines*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/0961003e-5a0a-4549-abde-af6a37f2724d
- Effect: AuditIfNotExists

2. Transparent Data Encryption on SQL databases should be enabled

Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics are protected at rest by Transparent Data Encryption. Generally, all the data, log files and backs ups stored in the persistent storage are termed as data at rest. TDE enables encryption without any specific changes required to the application. Database Encryption Key (DEK) is the symmetric key used by TDE to encrypt the entire database. DEK is usually protected by service managed certificate or an asymmetric key stored in Azure key vault.

Page level encryption is performed by TDE. It encrypts the data before it gets written to disk and decrypt when the read operation is performed. To avoid the data vulnerability, it is always recommended to keep the data encryption ON.

Policy Details

- Policy Name: *Transparent Data Encryption on SQL databases should be enabled*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/17k78e20-9358-41c9-923c-fb736d382a12
- Effect: AuditIfNotExists

3. Service Fabric clusters should have the ClusterProtectionLevel property set to EncryptAndSign

Three levels of protection provided for the node to node communication by the service fabric are None, Sign and EncryptAndSign. This protection level ensures that all node to node messages are encrypted and digitally signed.

Policy Details

- Policy Name: *Service Fabric clusters should have the ClusterProtectionLevel property set to EncryptAndSign*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/617c02be-7f02-4efd-8836-3180d47b6c68
- Effect: Audit, Deny

4. Bring your own key data protection should be enabled for PostgreSQL servers

Service managed keys has the data encrypted by default. The user who owns the customer-managed keys (CMK) is responsible to encrypt the data using Azure Key Vault. The customer managed key data should be encrypted at rest for Postgres Servers.

Policy Details

- Policy Name: *Bring your own key data protection should be enabled for PostgreSQL servers*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/18adea5e-f416-4d0f-8aa8-d24321e3e274
- Effect : AuditIfNotExists

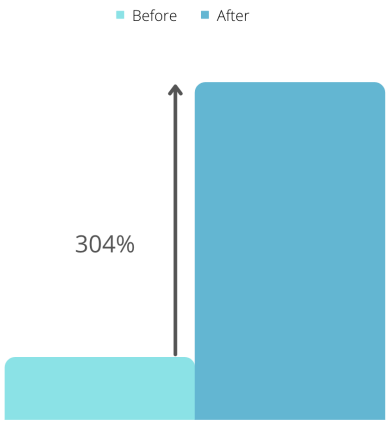

Figure 4.4: Encrypt data in rest

After the implementation of above recommended policies, the cloud adoption framework was used to measure the increase in the benchmark score. As a result, I observed that there is an increase of 217.5 percent comparing the existing benchmark score.

4.5 Manage access and permissions

Some notable recommendations are

- External accounts with owner permissions should be removed from your subscription.
- External accounts with write permissions should be removed from your subscription.
- Deprecated accounts with owner permissions should be removed from your subscription.
- Deprecated accounts should be removed from your subscription.

- There should be more than one owner assigned to your subscription.
- Role-Based Access Control should be used on Kubernetes Services.
- Service Fabric clusters should only use Azure Active Directory for client authentication.
- Azure Policy Add-on for Kubernetes should be installed and enabled on your clusters.
- Service principals should be used to protect your subscriptions instead of Management Certificates.
- Storage accounts public access should be disallowed.
- Managed identity should be used in your function app.
- Managed identity should be used in your web app.
- Managed identity should be used in your API app.
- Function apps should have Client Certificates (Incoming client certificates) enabled.
- Authentication to Linux machines should require SSH keys.
- Guest Configuration extension should be installed on your machines.

1. External accounts with owner permissions should be removed from your subscription

External accounts, identified with different domain names and owner permission, should be removed to get rid of the unmonitored access. Else, this could be a trapdoor for the attackers to have data access.

Manual Steps

- Open access control page.
- Choose Role assignments.
- Find the external users with owner access and Click Remove.

Policy Details

- Policy Name: *External accounts with owner permissions should be removed from your subscription*
- Definition ID: http://providers/Microsoft.Authorization/policyDefinitions/f8456c1c-aa66-4dfb-861a-25d127b775c9
- Effect: AuditIfNotExists

2. External accounts with write permissions should be removed from your subscription

The Log analytics agent plays a significant role as they collect the security events to the configured workspace.

A list of monitoring agent health issues can be found at Azure Security Center Troubleshooting Guide.

Policy Details

- Policy Name: *External accounts with write permissions should be removed from your subscription*
- Definition ID:
- Effect: AuditIfNotExists

3. Deprecated accounts with owner permissions should be removed from your subscription

The blocked user accounts that serves no purpose in the subscription should be removed as they may provide a way for the attackers to access the data.

Manual Steps

- Open the access control page.
- Click the role assignment tab.
- Select the blocked user accounts and click remove.

Policy Details

- Policy Name: *Deprecated accounts with owner permissions should be removed from your subscription*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/ebb62a0c-3560-49e1-89ed-27e074e9f8ad
- Effect : AuditIfNotExists

4. Deprecated accounts should be removed from your subscription

The blocked user accounts that serves no purpose in the subscription should be removed as they may provide a way for the attackers to access the data.

- Open the access control page.
- Click the role assignment tab.
- Select the blocked user accounts and click remove.

Policy Details

- Policy Name: *Deprecated accounts should be removed from your subscription*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/6b1cbf55-e8b6-442f-ba4c-7246b6381474
- Effect : AuditIfNotExists

5. There should be more than one owner assigned to your subscription

Assign more than one subscription owner to achieve access redundancy.

Manual Steps

- Open IAM Page.
- Open Add role assignment pane by clicking Add.
- Select Owner role in the Role drop down list.
- Select a user from the list.
- Click Save.

Policy Details

- Policy Name: *There should be more than one owner assigned to your subscription*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/09024ccc-0c5f-475e-9457-b7c0d9ed487b
- Effect : AuditIfNotExists

6. Role-Based Access Control should be used on Kubernetes Services

RBAC is used for managing permissions in Kubernetes Service Clusters and configuration of authorization policies.

Manual Steps

- Open Azure Kubernetes Services.
- Click Add. Enter the Cluster's configuration.
- Enable the RBAC Settings in Authentication tab.

Policy Details

- Policy Name: *Role-Based Access Control should be used on Kubernetes Services*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/ac4a19c2-fa67-49b4-8ae5-0b2e78c49457
- Effect : Audit

7. Service Fabric clusters should only use Azure Active Directory for client authentication

Restrict Client authentication only to Azure Active Directory in Service Fabric.

Policy Details

- Policy Name: *Service Fabric clusters should only use Azure Active Directory for client authentication*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/b54ed75b-3e1a-44ac-a333-05ba39b99ff0
- Effect : Audit, Deny

8. Azure Policy Add-on for Kubernetes should be installed and enabled on your clusters

Azure Policy Add-on for Kubernetes service (AKS) extends Gatekeeper v3, that acts as the admission controller webhook for Open Policy Agent.

Policy Details

- Policy Name: *Azure Policy Add-on for Kubernetes service (AKS) should be installed and enabled on your clusters*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/0a15ec92-a229-4763-bb14-0ea34a568f8d
- Effect : Audit

9. Service principals should be used to protect your subscriptions instead of Management Certificates

Service principals with resource manager can be used for secure automated resource management. They play a significant role in limiting the damage caused by certificate compromise.

- To replace management certificates with service principals
- Create Service principal with the certificate
- Select the required subscription from the list
- Choose the required management certificates under settings and delete the existing management certificates to replace them with the created service principals

Policy Details

- Policy Name: *Service principals should be used to protect your subscriptions instead of management certificates*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/6646a0bd-e110-40ca-bb97-84fcee63c414
- Effect: AuditIfNotExists

10. Storage accounts public access should be disallowed

Anonymous public read access to storage account (containers and blobs) should be restricted unless it has a necessity.

- Open storage account.
- Select Configurations from settings menu.
- Disable the "Allow Blob public access" icon.

Policy Details

- Policy Name: *Storage account public access should be disallowed*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/4fa4b6c0-31ca-4c0d-b10d-24b96f62a751
- Effect: Audit, Deny

11. Managed identity should be used in your function app

Azure resources will have an identity in Azure AD and that could be used to obtain Azure Active Directory tokens

Manual Steps

- Open App service.
- Select identity under platform feature.
- More details at https://docs.microsoft.com/en-us/azure/app-service/overview-managed-identity?tabs=dotnet.

Policy Details

- Policy Name: *Managed identity should be used in your function app*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/0da106f2-4ca3-48e8-bc85-c638fe6aea8f
- Effect : AuditIfNotExists

12. Managed identity should be used in your web app

Azure resources will have an identity in Azure AD and that could be used to obtain Azure Active Directory tokens

Manual Steps

- Open App service.
- Select identity under platform feature.
- More details at https://docs.microsoft.com/en-us/azure/app-service/overview-managed-identity?tabs=dotnet.

Policy Details

- Policy Name: *Managed identity should be used in your web app*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/2b9ad585-36bc-4615-b300-fd4435808332
- Effect : AuditIfNotExists

13. Managed identity should be used in your API app

Azure resources will have an identity in Azure AD and that could be used to obtain Azure Active Directory tokens

Manual Steps

- Open App service.
- Select identity under platform feature.

Policy Details

- Policy Name: *Managed identity should be used in your API app*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/c4d441f8-f9d9-4a9e-9cef-e82117cb3eef
- Effect : AuditIfNotExists

14. Function apps should have Client Certificates (Incoming client certificates) enabled

Only the client requests with valid certificates should be able to reach the app.

Manual Steps

- Open App Service.
- Select Configuration.
- In General settings tab, set requirements for incoming client certificates.

Policy Details

- Policy Name: *Function apps should have Client Certificates (Incoming client certificates) enabled*
- Definition ID:
- Effect : Audit

15. Authentication to Linux machines should require SSH keys

In spite of being secure, there is still a possibility of brute force attacks while using passwords with VM. It's always recommended to rely on public private keys.

Manual Steps

- Create SSH Key pair for the Linux virtual machine.
- Disable password authentication in Linux VM Configuration.
- Update the SSH keys in ARM templates.

Policy Details

- Policy Name: *Authentication to Linux machines should require SSH keys*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/630c64f9-8b6b-4c64-b511-6544ceff6fd6
- Effect: AuditIfNotExists

16. Guest Configuration extension should be installed on your machines

Installing Guest Configuration extension helps in ensuring secure configuration for OS and environmental settings

Manual Steps

- Register the subscription to Guest Configuration resource provider.
- Install the Guest Configuration extension.
- Enable system assigned managed identity.

Policy Details

- Policy Name: *Guest Configuration extension should be installed on your machines*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/ae89ebca-1c92-4898-ac2c-9f63decb045c
- Effect: AuditIfNotExists

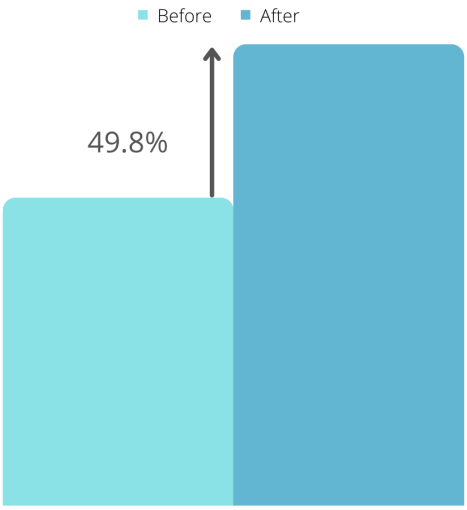

Figure 4.5: Manage access and permissions

After the implementation of above recommended policies, the cloud adoption framework was used to measure the increase in the benchmark score. As a result, I observed that there is an increase of 49.8 percent comparing the existing benchmark score.

4.6 End point protection

Some notable recommendations are

- Log Analytics agent should be installed on your virtual machine.
- Log Analytics agent health issues should be resolved on your machines.
- Install endpoint protection solution on virtual machines.
- Endpoint protection health issues should be resolved on your machines.
- Log Analytics agent should be installed on your virtual machine scale sets.
- Endpoint protection solution should be installed on virtual machine scale sets.
- Endpoint protection health failures should be remediated on virtual machine scale sets.

1. Log Analytics agent should be installed on your virtual machine

Log Analytics agent collects data from various security-related configurations and event logs of the Virtual Machine for the analysis of security vulnerabilities and threats. Log Analytics agent is also required for monitoring the CMs used by Azure managed services like Azure Kubernetes Service or Azure Service Fabric. It was highly recommended to enable automatic deployment of the agent.

Policy Details

- Policy Name: *Log Analytics agent should be installed on your virtual machine*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/a4fe33eb-e377-4efb-ab31-0784311bc499
- Effect: AuditIfNotExists

2. Log Analytics agent health issues should be resolved on your machines

The Log Analytics agent is very important for monitoring the virtual machine. So, it is highly recommended to ensure that agent is installed and operating as expected.

Policy Details

- Policy Name: *Log Analytics agent health issues should be resolved on your machines*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/d62cfe2b-3ab0-4d41-980d-76803b58ca65
- Effect : AuditIfNotExists

3. Install endpoint protection solution on virtual machines

It's recommended to have endpoint protection installed on the virtual machines.

Policy Details

- Policy Name: *Monitor missing Endpoint Protection in Azure Security Center*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/af6cd1bd-1635-48cb-bde7-5b15693900b9
- Effect: AuditIfNotExists

4. Endpoint protection health issues should be resolved on your machines

The end point protection health should be monitored and resolved as soon as possible.

Policy Details

- Policy Name: *Monitor missing Endpoint Protection in Azure Security Center*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/af6cd1bd-1635-48cb-bde7-5b15693900b9
- Effect: AuditIfNotExist

5. Log Analytics agent should be installed on your virtual machine scale sets

Log Analytics agent collects data from various security-related configurations and

event logs of the Virtual Machine for the analysis of security vulnerabilities and threats. Log Analytics agent is also required for monitoring the CMs used by Azure managed services like Azure Kubernetes Service or Azure Service Fabric. It was highly recommended to enable automatic deployment of the agent.

Policy Details

- Policy Name: *Log Analytics agent should be installed on your virtual machine scale sets for Azure Security Center monitoring*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/a3a6ea0c-e018-4933-9ef0-5aaa1501449b
- Effect: AuditIfNotExist

6. Enabling end point protection on the virtual machine sets will help in protecting them from threats and vulnerabilities

Policy Details

- Policy Name: *Endpoint protection solution should be installed on virtual machine scale sets*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/26a828e1-e88f-464e-bbb3-c134a282b9de
- Effect: AuditIfNotExist

7. Endpoint protection health failures should be remediated on virtual machine scale sets

To protect the virtual machines from threats and vulnerabilities, It is always recommended to remediate endpoint protection health failures.

Policy Details

- Policy Name: *Endpoint protection solution should be installed on virtual machine scale sets*
- Definition ID: /providers/Microsoft.Authorization/policyDefinitions/26a828e1-e88f-464e-bbb3-c134a282b9de
- Effect: AuditIfNotExist

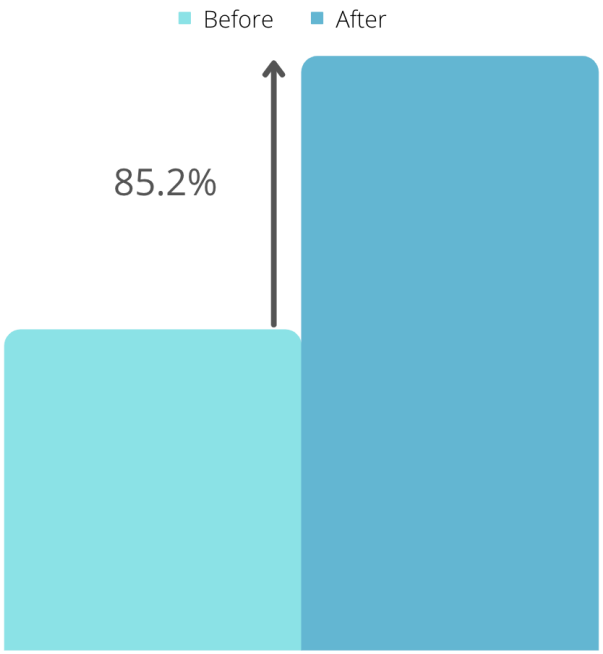

Figure 4.6: End Point Protection

After the implementation of above recommended policies, the cloud adoption framework was used to measure the increase in the benchmark score. As a result, I observed that there is an increase of 85.2 percent comparing the existing benchmark score.

5

Infrastructure Automation

Cloud infrastructure automation made jobs of develops easy as it enables them to create, modify and manage the resources in an automated, fast and efficient way. Infrastructure automation has really improved the security, resilience, tolerance and governance to the next level. Infrastructure as code defines the infrastructure in the configuration files and spins the cluster automatically by launching the configuration.

Initially, the current benchmark score calculated via Cloud Adaption Framework gave insights about the existing security loopholes and practices in the system. The security risks detected by the Azure security center are taken into account and polices to address them are listed out during the thesis work.

This thesis utilizes popular infrastructure automation tools like terraform and packer for automating the policy infrastructure using the recommended policies. The core language used for automation was HashiCorp Programming Language.

Terraform

Terraform is a popular open source tool from HashiCorp that helps in defining, launching and managing the cloud infrastructure via simple declarative programming language called HashiCorp Configuration Language [HCL]. This tool can be used to manage the infrastructure of public clouds (like Azure, AWS, GCP, etc) and private virtualization platforms like OpenStack [23]. The terraform scripts has the extension of ".tf". Every configuration file will have multiple code blocks corresponding to the targeted infrastructure resource. Terraform understands both JSON and HCL as its configuring language [24]. When the infrastructure is scripted as tf file, terraform load, parse and interpret the script block by block. Terraform, provider, variable, locals, resource, module, output are the blocks of Terraform based on their order of execution [25].

Packer

Packer is also an open source tool from Hashicorp that helps in the creation, management and consumption of cross-platform images in a centralized way from a single source template. Packer registry contains information like image creation date, image location and other build details. Packer images enable the deployment of provisioned and configured machines in seconds. As they have the potential to create reusable identical images for multiple platforms, portability is achieved.

Architecture

The infrastructure automation system is built to spin up the reusable, configurable infrastructure with necessary policies enabled as a guardrail to protect the system and reduce the security risks.

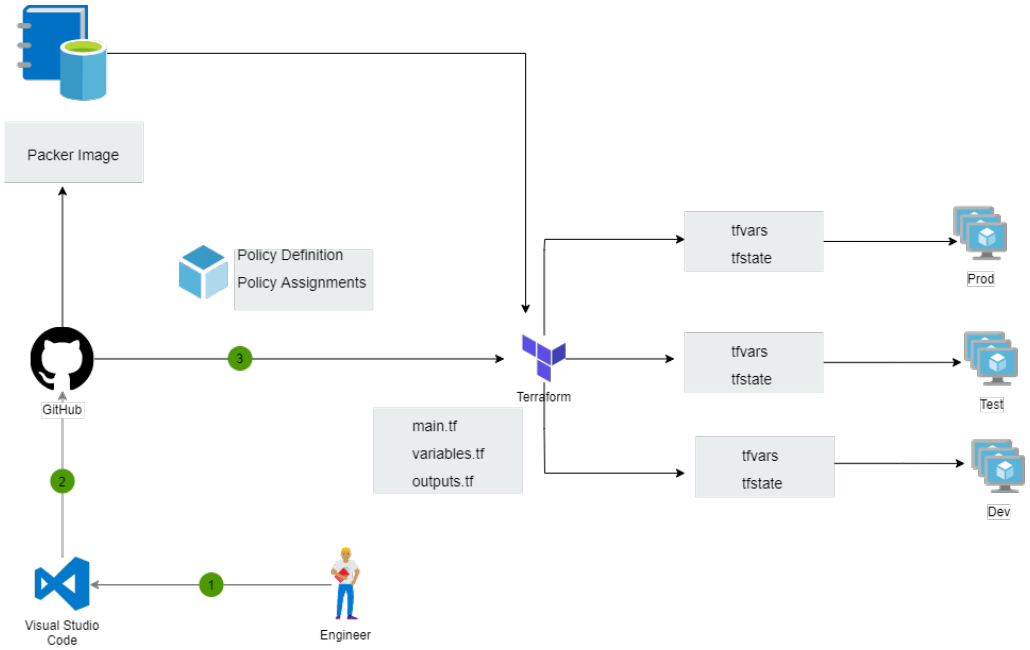

Figure 5.1: Policy as Code Architecture

The set of policies is fed into the automation system as terraform files. Different policy sets used in this automation are classified as below

- Cloud governance policy set
- Monitoring governance policy set
- Tag governance policy set
- IAM governance policy set
- Security governance policy set
- Data protection governance policy set

The terraform block will have details like version and provider. The provider block has tenant and subscription ID to connect with the target public cloud infrastructure. The project is divided into two parts.

The first part is to assign the set of policies to the cloud environment. The second part is to deploy an infrastructure into the cloud environment to verify if the guardrails ensure that the deployed infrastructure is 100 percent complying to the policies.

Part 1:

- Two modules, namely policy-assignment and policy definition, are created.
- Policy definition module has all the required policy sets to be deployed.
- The command "terraform init" is executed to initialize the working directory containing all the source configuration files.
- Thus, the overall plan will be displayed, and it gets saved in a file named "solution.plan".
- The command "terraform apply" is executed to apply the configurations.

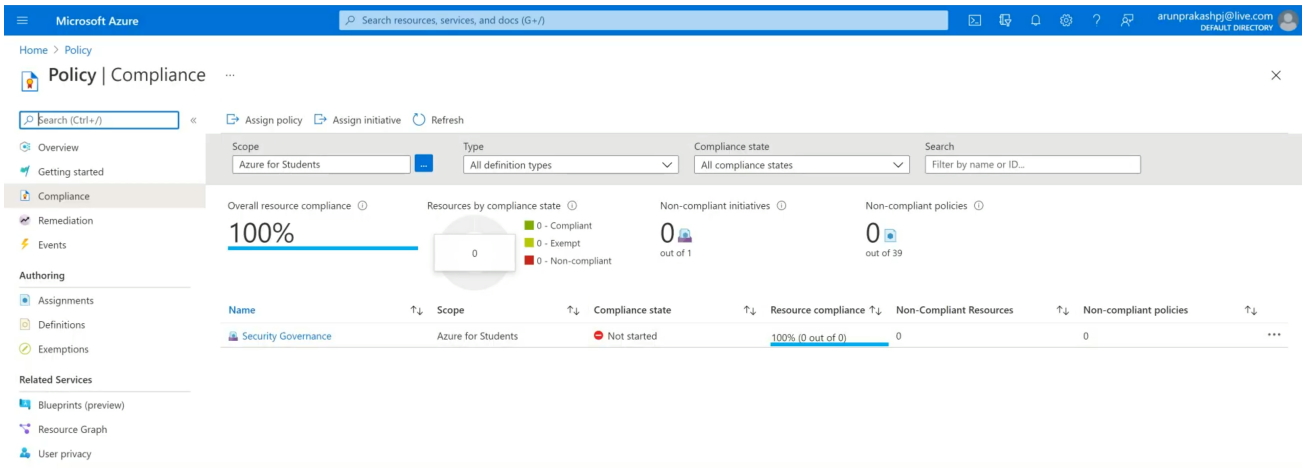

Figure 5.2: Policy Assignment

Part 2 :

- The tool packer is used to create infrastructure image via declarative configurations.
- The packer image is deployed into the target infrastructure using terraform.
- Every infrastructure resource is audited for compliance.
- Every infrastructure resource is audited for compliance.
- If they comply with the deployed policies, the configurations are applied.
- If they don't comply with the deployed policies, the configurations are denied.

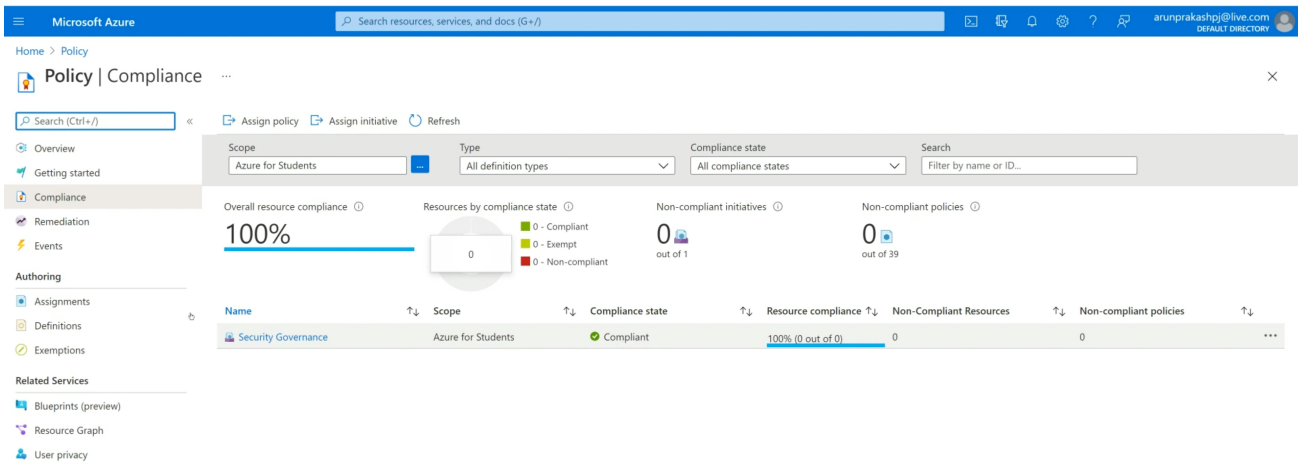

Figure 5.3: Compliance State

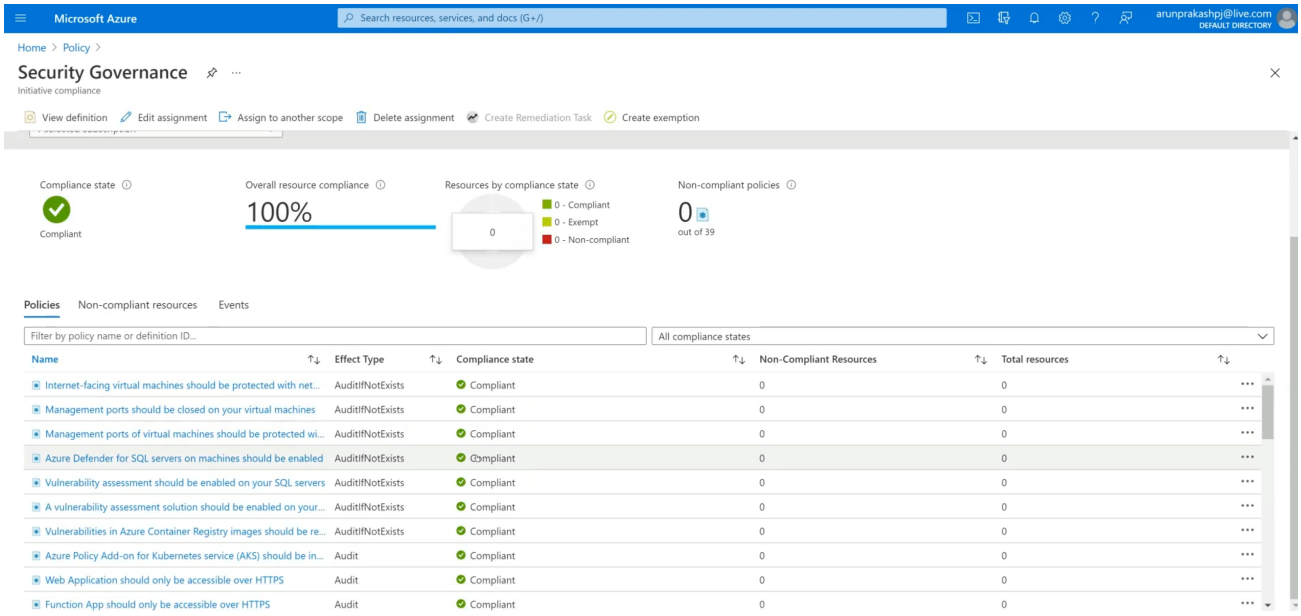

Figure 5.4: Compliance Evaluation

6
Discussion

Provisioning is the primary step to initiate an infrastructure. The most popular vendor specific infrastructure tools are CloudFormation for AWS, Cloud Deployment Manager for the Google Cloud Platform and Azure Resource Manager for Microsoft. There also exist quite a lot of open sources provisioning tools like terraform, pulumi, ansible, etc. This thesis work mainly focuses on Azure and opted for Terraform over Azure Resource Manager for provisioning the infrastructure. The core reason to choose terraform over Azure Resource Manager is due to the fact that terraform is open source, declarative and cloud-agnostic. The infrastructure configuration written using terraform can be utilized to spin up the infrastructure in any cloud, as it follows a declarative configuration approach.

6.1 Terraform Architecture

Terraform is an open source tool from Hashicorp, that provides flexible abstraction of resources and providers. This tool helps us to visualize almost everything, from physical hardware, virtual machines, DNS providers, etc. Terraform has been written in Go Language, which is known for its efficient concurrency capabilities. The Terraform has two significant modules, namely Terraform Core and Terraform Plugins.

6.1.1 Terraform Core

Terraform Core is a statically-compiled binary which acts as the entry point for anyone using terraform. The primary responsibilities of terraform core are reading the configuration files and modules, resource state management, resource graph construction, plan execution and enabling communication with plugins over RPC.

6.1.2 Terraform Plugins

Terraform plugins are executable binaries invoked by terraform core over RPC. The implementation of specific provisioners is exposed by these plugins. Each plugin is executed separately as an individual process, and they rely on an RPC interface to communicate with the main terraform binary. Though terraform has several built in provisioners, based on the need, sometimes the providers are dynamically discovered on the fly. The primary responsibilities of plugins are to initialize any libraries that are employed in making API calls, infrastructure provider authentication and defining resources that could be mapped to services.

6.2 HashiCorp Configuration Language

Most of the HashiCorp products rely on a configuration language called HashiCorp Configuration Language. This language syntax is built around two key syntax constructs, namely arguments and blocks.

6.2.1 Arguments

Every argument names, block type names, input variables or any other resources in terraform are called Identifiers. An identifier is allowed to have a letters, digits, underscores and hyphens. But the identifier cannot start with a digit. Argument assigns a value to any particular identifier. The syntax is given below.

```
  imgId = "xyz123"
```

The identifier "imgId" before the equals sign is the argument name, and the expression "xyz123" after the equal sign is its value.

6.2.2 Blocks

A block can be defined as a container to hold a group of configurations. Most of the tool features are implemented as top-level blocks that includes resources, data sources, input variables, output values, etc. A snippet of a resource block used during the thesis is given below for a reference.

```
  resource "azurerm_virtual_network" "main" {
  name                = "${var.prefix}-network"
  address_space       = ["10.0.0.0/24"]
  location            = azurerm_resource_group.main.location
  resource_group_name = azurerm_resource_group.main.name
  tags                = var.tags
}
```

A resource block is defined in the example above. After every block type, we define the required label names. Here, "azurerm_virtual_network" and "main" are the labels. These labels change based on the requirement, and it's not a mandatory element. After labels, the block body is delimited by the "{" and "}" delimiters. In the block body, further arguments and blocks can be added and sometimes nested if required.

6.3 Features And Execution

Terraform helps to define the infrastructure as a code to manage end-to-end infrastructure in a declarative fashion that can create, manage and destroy the resources

in a very quick time. A sample snippet is provided below to display the declarative configuration which is used as one of the input.

```
variable "num_of_vms" {
  description = "Number of VM resources to be
                 created behind the load balancer"
  default     = 2
  type        = number
}
```

This variable block expects the user to give the number of virtual machines during the execution of infrastructure setup. In case if the user is not providing any inputs, then default number is taken as the input.

The policy recommendation list generated by the thesis work has been configured into the variable block with a label "security_governance_policies". A snippet is displayed below that has a default policy list. Thus, anytime if the user ignores the policies list, the default policy recommendations will be applied over the infrastructure.

```
variable "security_governance_policies" {
  type        = list
  description = "List of policy definitions for the security_governance policyset"
  default = [
    "Internet-facing virtual machines
    should be protected with network security groups",
    "Management ports should be
    closed on your virtual machines",
    "Management ports of virtual machines
    should be protected with just-in-time network access control",
...
  ]
}

data "azurerm_policy_definition" "builtin_policies_security_governance" {
  count        = length(var.builtin_policies_security_governance)
  display_name = var.builtin_policies_security_governance[count.index]
}
```

Whenever "terraform init" command is executed, the required community modules are automatically downloaded. Then the user can plan a dry run before applying the configurations over the infrastructure. When "terraform plan" command is executed, terraform allows the user to make safe changes to the infrastructure with mapped resources in a dry run mode. Once the plan is satisfactory, "terraform apply"

command is executed to apply the configuration changes to the infrastructure.

To replicate the same infrastructure from an image, a web server was developed as a part of this thesis work. The snippet of the web server is displayed below. The parameter "managed_image_name" denotes the image that contains the infrastructural changes.

```
{
  "variables": {
    "client_id": "{{env `AZ_PACKER_ID` }}",
    "tenant_id": "{{env `AZ_TENANT_ID` }}",
    "client_secret": "{{env `AZ_PACKER_SECRET` }}",
    "subscription_id": "xyz"
  },
  "builders": [{
    "type": "azure-arm",

    "client_id": "{{user `client_id`}}",
    "client_secret": "{{user `client_secret`}}",
    "subscription_id": "{{user `subscription_id`}}",

    "os_type": "Linux",
    "image_publisher": "Canonical",
    "image_offer": "UbuntuServer",
    "image_sku": "18.04-LTS",
    "azure_tags": {
      "Name": "volvo-webserver"
    },
    "managed_image_resource_group_name": "packer-rg",
    "managed_image_name": "webserverPackerImage",

    "location": "North Europe",
    "vm_size": "Standard_B1s"
  }],
  "provisioners": [{
    "inline": [
        "apt-get update",
        "apt-get upgrade -y"
    ],
    "inline_shebang": "/bin/sh -x",
    "type": "shell",
    "execute_command": "chmod +x {{ .Path }}; {{ .Vars }} sudo -E sh '{{ .Path }}'"
  }]
}
```

6.4 Merits and Demerits

- Terraform is an open source tool that has cloud-agnostic uniform declarative syntax for infrastructure as code. It has a plugin based structure, facilitating easy extension of the software functionality. The other impressive feature is its ability to generate dependency graphs for the complete Infrastructure as Code. As it is an open source tool, it is really an inexpensive and effective tool to build the infrastructure.

- The major disadvantage of this tool is its inability to have any automated rollback in case of incorrect resource changes. Though this tool is an open source, collaboration and security features are available as premium features exclusively available only in expensive enterprise plans.

7
Conclusion

The work presented in this thesis described the process of assessing the security & identity baseline of the organization's cloud infrastructure, thus helping them to improve the cloud governance using the application of infrastructure automation. The Cloud Adoption Framework has been employed to assess the security and identity baseline of the cloud infrastructure. The input to the Cloud Adoption framework has been collected from a team of 12 people working in Network Security and Cloud Infrastructure team of the organization. Based on the results delivered by the cloud adoption framework, the current security and identity benchmark score has been identified and potential improvement over the benchmark score was explored to improve it to the highest possible level.

As a part of the assessment, security management ports, data encryption during transit/rest, identity access management, end point protection mechanisms of the cloud infrastructure is studied and guidelines to improve them are drafted. The drafted guidelines are used as a source to come up with a set of policy recommendations to make an effective guardrail over the system.These guardrails have been raised once the recommended policies are deployed over the cloud infrastructure.

The application of automation was very beneficial in achieving an ideal environment. The tools like terraform and packer are used for building the infrastructure using policy as code. Part 1 of the automation focuses on deployment of policies into the cloud infrastructure, thus guardrails has been enabled to place restrictions over the cloud infrastructure. The part 2 of the thesis focused on the infrastructure automation. If the deployed infrastructure has some resources violating the guard rails, the configurations has been denied from getting deployed.

The thesis has improved the security and identity baseline scores of the cloud infrastructure to a tremendous level, and automation made the process repeatable and superfast. The outcome of the thesis can be used by the organization to improve the security standards of their cloud infrastructure. The automation script can be used to deploy a repeatable infrastructure over any region, as the code is completely written with declarative configuration.

Bibliography

[1] X. Chen, C. Chen, Y. Tao, and J. Hu, "A cloud security assessment system based on classifying and grading," *IEEE Cloud Computing*, vol. 2, no. 2, pp. 58–67, 2015.

[2] T. Kao, C. Mao, C. Chang, and K. Chang, "Cloud ssdlc: Cloud security governance deployment framework in secure system development life cycle," in *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, 2012, pp. 1143–1148.

[3] H. Chang and E. Choi, "User authentication in cloud computing," in *Ubiquitous Computing and Multimedia Applications*, T.-h. Kim, H. Adeli, R. J. Robles, and M. Balitanas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 338–342.

[4] I. Indu, P. R. Anand, and V. Bhaskar, "Identity and access management in cloud environment: Mechanisms and challenges," *Engineering Science and Technology, an International Journal*, vol. 21, no. 4, pp. 574–588, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2215098617316750

[5] H. Heier, H. P. Borgman, and B. Bahli, "Cloudrise: Opportunities and challenges for it governance at the dawn of cloud computing," in *2012 45th Hawaii International Conference on System Sciences*, 2012, pp. 4982–4991.

[6] K. Bratanis and D. Kourtesis, "Introducing policy-driven governance and service level failure mitigation in cloud service brokers: Challenges ahead," in *Service-Oriented Computing – ICSOC 2013 Workshops*, A. R. Lomuscio, S. Nepal, F. Patrizi, B. Benatallah, and I. Brandić, Eds. Cham: Springer International Publishing, 2014, pp. 177–191.

[7] M. Al-Ruithe, E. Benkhelifa, and K. Hameed, "Key dimensions for cloud data governance," in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, 2016, pp. 379–386.

[8] M. Al-Ruithe and E. Benkhelifa, "Cloud data governance maturity model," in *Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing*, ser. ICC '17. New York, NY, USA: Association for Computing Machinery, 2017. [Online]. Available: https://doi.org/10.1145/3018896.3036394

[9] M. Artac, T. Borovssak, E. Di Nitto, M. Guerriero, and D. A. Tamburri, "Devops: Introducing infrastructure-as-code," in *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, 2017, pp. 497–498.

[10] A. Sharma, S. Sharma, and M. Dave, "Identity and access management- a comprehensive study," in *2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 1481–1485.

[11] B. Thuraisingham, "Cloud governance," in *2020 IEEE 13th International Conference on Cloud Computing (CLOUD)*, 2020, pp. 86–90.

[12] R. Farrell, "Securing the cloud—governance, risk, and compliance issues reign supreme," *Information Security Journal: A Global Perspective*, vol. 19, no. 6, pp. 310–319, 2010. [Online]. Available: https://doi.org/10.1080/19393555.2010.514655

[13] S. Namasudra, P. Roy, and B. Balusamy, "Cloud computing: Fundamentals and research issues," in *2017 Second International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM)*, 2017, pp. 7–12.

[14] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of internet services and applications*, vol. 1, no. 1, pp. 7–18, 2010.

[15] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities," in *2008 10th IEEE international conference on high performance computing and communications*. Ieee, 2008, pp. 5–13.

[16] M. U. Bokhari, Q. M. Shallal, and Y. K. Tamandani, "Cloud computing service models: A comparative study," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, 2016, pp. 890–895.

[17] E. Evans and R. Grossman, "Cyber security and reliability in a digital cloud," *US Department of Defense Science Board Study*, 2013.

[18] J. Gibson, R. Rondeau, D. Eveleigh, and Q. Tan, "Benefits and challenges of three cloud computing service models," in *2012 Fourth International Conference on Computational Aspects of Social Networks (CASoN)*, 2012, pp. 198–205.

[19] P. Dutta and P. Dutta, "Comparative study of cloud services offered by amazon, microsoft & google," *International Journal of Trend in Scientific Research and Development*, vol. 3, no. 3, pp. 981–985, 2019.

[20] B. Gupta, P. Mittal, and T. Mufti, "A review on amazon web service (aws), microsoft azure & google cloud platform (gcp) services," 2021.

[21] J. García-Galán, P. Trinidad, O. F. Rana, and A. Ruiz-Cortés, "Automated configuration support for infrastructure migration to the cloud," *Future Generation Computer Systems*, vol. 55, pp. 200 – 212, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X15000618

[22] Microsoft, "Cloud adoption framework," 2021. [Online]. Available: https://azure.microsoft.com/sv-se/cloud-adoption-framework/

[23] L. R. de Carvalho and A. Patricia Favacho de Araujo, "Performance comparison of terraform and cloudify as multicloud orchestrators," in *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, 2020, pp. 380–389.

[24] N. Sabharwal, S. Pandey, and P. Pandey, "Getting started with hashicorp terraform," in *Infrastructure-as-Code Automation Using Terraform, Packer, Vault, Nomad and Consul*. Springer, 2021, pp. 11–45.

[25] R. Modi, "Getting started with terraform," in *Deep-Dive Terraform on Azure*. Springer, 2021, pp. 43–75.

A

Appendix 1

A.1 Cloud Adoption Framework

Figure A.1: CAF Part 1

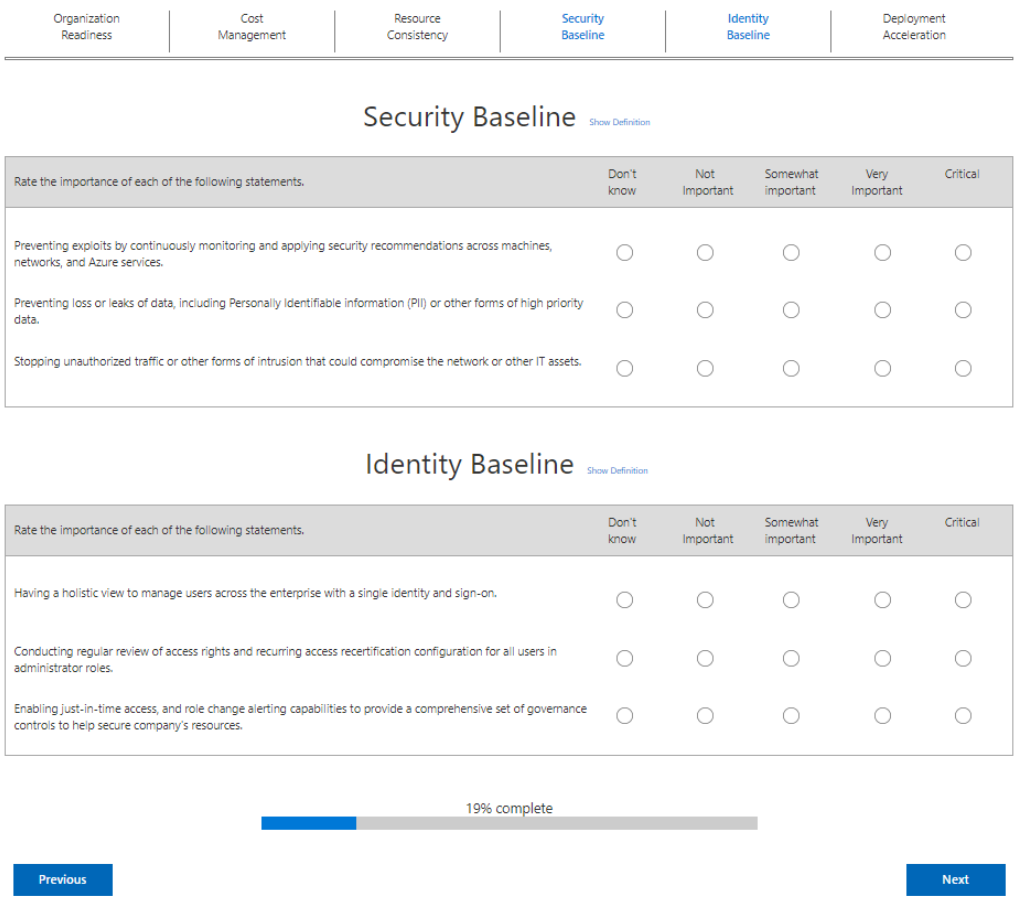

Figure A.2: CAF Part 2

Figure A.3: CAF Part 3

A.2 Code Snippet

main.tf

```
terraform {
```

```
  required_version = ">= 0.13"
  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~> 2.33.0"
    }
  }
}

provider "azurerm" {
  skip_provider_registration = true
  features {}
}

module "policy_assignments" {
  source = "./modules/policy-assignments"
  security_governance_policyset_id = module.policyset_definitions.securit

}


module "policyset_definitions" {
  source = "./modules/policyset-definitions"

}
```

policy-assignments
main.tf

```
resource "azurerm_policy_assignment" "security_governance" {
  name                 = "security_governance"
  scope                = data.azurerm_subscription.current.id
  policy_definition_id = var.security_governance_policyset_id
  description          = "Assignment of the Security Governance initiativ
  display_name         = "Security Governance"
  location             = "europewest"
  identity { type = "SystemAssigned" }
}
```

output.tf

```
output "security_governance_assignment_id" {
  value       = azurerm_policy_assignment.security_governance.id
  description = "The policy assignment id for security_governance"
}
```

```
output "security_governance_assignment_identity" {
  value       = azurerm_policy_assignment.security_governance.identity
  description = "The policy assignment identity for security_governance"
}
```

variables.tf

```
data "azurerm_subscription" "current" {
}

variable "security_governance_policyset_id" {
  type        = string
  description = "The policy set definition id for security_governance"
}
```

policyset-definitions
main.tf

```
resource "azurerm_policy_set_definition" "security_governance" {

  name         = "security_governance"
  policy_type  = "Custom"
  display_name = "Security Governance"
  description  = "Contains common Security Governance policies"

  metadata = <<METADATA
    {
    "category": "${var.policyset_definition_category}"
    }
METADATA

  dynamic "policy_definition_reference" {
    for_each = data.azurerm_policy_definition.builtin_policies_security_g
    content {
      policy_definition_id = policy_definition_reference.value["id"]
      reference_id         = policy_definition_reference.value["id"]
    }
  }
}
```

output.tf

```
output "security_governance_policyset_id" {
  value       = azurerm_policy_set_definition.security_governance.id
  description = "The policy set definition id for security_governance"
}
```

variable.tf

```
variable "policyset_definition_category" {
  type        = string
  description = "The category to use for all PolicySet defintions"
  default     = "Custom"
}

variable "builtin_policies_security_governance" {
  type        = list
  description = "List of policy definitions for the security_governance p
  default = [
    "Internet-facing virtual machines should be protected with network se
    "Management ports should be closed on your virtual machines",
    "Management ports of virtual machines should be protected with just-i


    "Azure Defender for SQL servers on machines should be enabled",
    "Vulnerability assessment should be enabled on your SQL servers",

    "A vulnerability assessment solution should be enabled on your virtua
    "Vulnerabilities in Azure Container Registry images should be remedia
    "Azure Policy Add-on for Kubernetes service (AKS) should be installed


    "Web Application should only be accessible over HTTPS",
    "Function App should only be accessible over HTTPS",
    "API App should only be accessible over HTTPS",
    "Only secure connections to your Azure Cache for Redis should be enab
    "Secure transfer to storage accounts should be enabled",
    "Latest TLS version should be used in your API App",
    "Enforce SSL connection should be enabled for PostgreSQL database ser
    "Latest TLS version should be used in your Web App",
    "Latest TLS version should be used in your Function App",
    "FTPS should be required in your Web App",


    "Disk encryption should be applied on virtual machines",
    "Transparent Data Encryption on SQL databases should be enabled",
    "Service Fabric clusters should have the ClusterProtectionLevel prope


    "External accounts with owner permissions should be removed from your
    "External accounts with write permissions should be removed from your
    "Deprecated accounts with owner permissions should be removed from yo
```

```
    "Deprecated accounts should be removed from your subscription",
    "There should be more than one owner assigned to your subscription",
    "Role-Based Access Control (RBAC) should be used on Kubernetes Servic
    "Service Fabric clusters should only use Azure Active Directory for c
    "Service principals should be used to protect your subscriptions inst


    "Managed identity should be used in your Function App",
    "Managed identity should be used in your Web App",
    "Managed identity should be used in your API App",

    "Function apps should have 'Client Certificates (Incoming client cert
    "Authentication to Linux machines should require SSH keys",
    "Guest Configuration extension should be installed on your machines",


    "Log Analytics agent should be installed on your virtual machine for
    "Log Analytics agent health issues should be resolved on your machine


    "Log Analytics agent should be installed on your virtual machine scal
    "Endpoint protection solution should be installed on virtual machine



  ]
}

data "azurerm_policy_definition" "builtin_policies_security_governance" {
  count        = length(var.builtin_policies_security_governance)
  display_name = var.builtin_policies_security_governance[count.index]
}
```

Packer Code

webserver.json

```
{
  "variables": {
    "client_id": "{{env `AZ_PACKER_ID` }}",
    "tenant_id": "{{env `AZ_TENANT_ID` }}",
    "client_secret": "{{env `AZ_PACKER_SECRET` }}",
    "subscription_id": "xyz"
  },
  "builders": [{
    "type": "azure-arm",

    "client_id": "{{user `client_id`}}",
```

```
    "client_secret": "{{user `client_secret`}}",
    "subscription_id": "{{user `subscription_id`}}",

    "os_type": "Linux",
    "image_publisher": "Canonical",
    "image_offer": "UbuntuServer",
    "image_sku": "18.04-LTS",
    "azure_tags": {
      "Name": "volvo-webserver"
    },
    "managed_image_resource_group_name": "packer-rg",
    "managed_image_name": "webserverPackerImage",

    "location": "North Europe",
    "vm_size": "Standard_B1s"
  }],
  "provisioners": [{
    "inline": [
        "apt-get update",
        "apt-get upgrade -y"
    ],
    "inline_shebang": "/bin/sh -x",
    "type": "shell",
    "execute_command": "chmod +x {{ .Path }}; {{ .Vars }} sudo -E sh '{{
  }]
}
```

variable.tf

```
variable "prefix" {
  description = "The prefix which should be used for all resources in thi
  default = "volvo-azure-webserver"
}

variable "location" {
  description = "The Azure Region in which all resources in this example
  default = "North Europe"
}


variable "resource_group" {
  description = "Name of the resource group, including the -rg"
  default     = "volvo-WSproject-rg"
  type        = string
}

variable "tags" {
```

```
  description = "A map of the tags to use for the resources that are depl
  type        = map(string)
  default = {
    Name = "volvo-azure-webserver"
  }
}

variable "username" {
  description = "Enter username to associate with the machine"
}

variable "password" {
  description = "Enter password to use to access the machine"
}

variable "packer_resource_group" {
  description = "Resource group of the Packer image"
  default     =  "packer-rg"
  type        = string
}

variable "packer_image_name" {
  description = "Image name of the Packer image"
  default     =  "webserverPackerImage"
  type        = string
}

variable "num_of_vms" {
  description = "Number of VM resources to be  created behnd the load bal
  default     = 2
  type        = number
}
```

main.tf

```
provider "azurerm" {
  features {}
}

resource "azurerm_resource_group" "main" {
  name     = var.resource_group
  location = var.location
  tags     = var.tags
}

resource "azurerm_virtual_network" "main" {
  name                = "${var.prefix}-network"
```

```
  address_space       = ["10.0.0.0/24"]
  location            = azurerm_resource_group.main.location
  resource_group_name = azurerm_resource_group.main.name
  tags                = var.tags
}

resource "azurerm_subnet" "main" {
 name                 = "${var.prefix}-subnet"
 resource_group_name  = azurerm_resource_group.main.name
 virtual_network_name = azurerm_virtual_network.main.name
 address_prefixes     = ["10.0.0.0/24"]
}

resource "azurerm_network_interface" "main" {
 count               = var.num_of_vms
 name                = "${var.prefix}-${count.index}-nic"
 location            = var.location
 resource_group_name = azurerm_resource_group.main.name

 ip_configuration {
   name                          = "mainConfiguration"
   subnet_id                     = azurerm_subnet.main.id
   private_ip_address_allocation = "Dynamic"
 }
 tags = var.tags
}

resource "azurerm_public_ip" "main" {
  name                = "${var.prefix}-public-ip"
  resource_group_name = azurerm_resource_group.main.name
  location            = var.location
  allocation_method   = "Static"
  tags                = var.tags
}

resource "azurerm_lb" "main" {
 name                = "${var.prefix}-lb"
 location            = azurerm_resource_group.main.location
 resource_group_name = azurerm_resource_group.main.name

 frontend_ip_configuration {
   name                 = "publicIPAddress"
   public_ip_address_id = azurerm_public_ip.main.id
 }

 tags = var.tags
```

```
}

resource "azurerm_lb_backend_address_pool" "main" {
 resource_group_name = azurerm_resource_group.main.name
 loadbalancer_id     = azurerm_lb.main.id
 name                = "BackEndAddressPool"
}


resource "azurerm_availability_set" "availset" {
 name                         = "availset"
 location                     = azurerm_resource_group.main.location
 resource_group_name          = azurerm_resource_group.main.name
 platform_fault_domain_count  = 2
 platform_update_domain_count = 2
 managed                      = true
tags = var.tags
}

data "azurerm_resource_group" "packer_rg" {
  name = var.packer_resource_group
}

data "azurerm_image" "image" {
  name                = var.packer_image_name
  resource_group_name = data.azurerm_resource_group.packer_rg.name
}


resource "azurerm_virtual_machine" "main" {
  count                            = var.num_of_vms
  name                             = "${var.prefix}${count.index}-vm"
  resource_group_name              = azurerm_resource_group.main.name
  location                         = azurerm_resource_group.main.location
  vm_size                             = "Standard_B1s"
  network_interface_ids = [element(azurerm_network_interface.main.*.id, c

  storage_image_reference  {
    publisher = "Canonical"
    offer     = "UbuntuServer"
    sku       = "18.04-LTS"
    version   = "latest"
  }

  os_profile {
   computer_name  =  "hostname${count.index}"
```

```
    admin_username                  = "testadmin"
    admin_password                  = "Password1234"
  }

  os_profile_linux_config {
    disable_password_authentication = false
  }

  storage_os_disk  {
    name              = "WSdisk${count.index}"
    caching           = "ReadWrite"
    create_option     = "FromImage"
    managed_disk_type = "Standard_LRS"
  }

}
```